Report Number: WUCS-96-17

1996-01-01

A Pilot Study of Speech and Pen User Interface For Graphical Editing

Karl E. Schmidt

As computer size continues to decrease and new user interface technologies become more ubiquitous, the conventional keyboard and mouse input interfaces are becoming harder to design into newer machines and less practical for use in some applications. The pen is one input technology more suited for the upcoming generation of smaller computers using direct manipulation interfaces. However, a pen-only user interface relies on continuous gesture and handwriting tecognizers that are often slow, inaccurate, and error prone for command and text entry. Speech recognition is an input modality that can input commands quickly and potentially be a fast text entry... **Read complete abstract on page 2.**

A Pilot Study of Speech and Pen User Interface For Graphical Editing

Karl E. Schmidt

Complete Abstract:

As computer size continues to decrease and new user interface technologies become more ubiquitous, the conventional keyboard and mouse input interfaces are becoming harder to design into newer machines and less practical for use in some applications. The pen is one input technology more suited for the upcoming generation of smaller computers using direct manipulation interfaces. However, a pen-only user interface relies on continuous gesture and handwriting tecognizers that are often slow, inaccurate, and error prone for command and text entry. Speech recognition is an input modality that can input commands quickly and potentially be a fast text entry mechanism, but lacks the capability of direct object manipulation and has inaccurate recognition. The combination of both pen and voice input should complement each other for direct graphic manipulation applications. This thesis compares the speed, usability, user-friendliness, and accuracy of a pen-only graphical editor against a pen-with-speech graphical editor. Two versions of a graphical editor were developed which have the same functionality. One is controlled by pen input alone and the other is controlled by both pen and speech input. The pen-only editor used the tool bar for command entry and character handwriting recognition for text entry. The pen-with-speech editor used speech recognition for both command and text entry. In a pilot study using both editors, 13 computer science graduate students were asked to draw a petri net, a state diagram, a flowchart, and a dataflow diagram. Shape entry was facilitated by automatic shape recognition that transformed continuous drawing information into a perfected shape. Experimental results comparing the editor's user interfaces were then analysed. Results show that the addition of speech made the editor slightly faster. Experimental subjects claimed this editor was more usable, perceived to be faster, and preferred to use. About half of the subjects found the editor with speech not to be more user-friendly than the pen-only editor. The accuracy of character recognition for the pen-with-speech editor was significantly inferior to the pen-only editor's handwriting recognition. The low recognition accuracy was caused by the speech recognizer's inability to distinguish between similar sounding letters.

A PILOT STUDY OF SPEECH AND PEN USER INTERFACE FOR GRAPHICAL EDITING

Karl E. Schmidt

WUCS-96-17

June 1996

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

WASHINGTON UNIVERSITY
SEVER INSTITUTE OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE

A PILOT STUDY OF SPEECH AND PEN USER INTERFACE FOR GRAPHICAL EDITING

by

Karl E. Schmidt

Prepared under the direction of Professor Takayuki Dan Kimura

---

A thesis presented to the Sever Institute of
Washington University in partial fulfillment
of the requirements for the degree of
MASTER OF SCIENCE

May, 1996

Saint Louis, Missouri

WASHINGTON UNIVERSITY
SEVER INSTITUTE OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE

ABSTRACT

A PILOT STUDY OF SPEECH AND PEN USER INTERFACE FOR GRAPHICAL EDITING

by Karl E. Schmidt

ADVISOR: Professor Takayuki Dan Kimura

May, 1996
Saint Louis, Missouri

As computer size continues to decrease and new user interface technologies become more ubiquitous, the conventional keyboard and mouse input interfaces are becoming harder to design into newer machines and less practical for use in some applications. The pen is one input technology more suited for the upcoming generation of smaller computers using direct manipulation interfaces. However, a pen-only user interface relies on continuous gesture and handwriting recognizers that are often slow, inaccurate, and error prone for command and text entry. Speech recognition is an input modality that can input commands quickly and potentially be a fast text entry mechanism, but lacks the capability of direct object manipulation and has inaccurate recognition. The combination of both pen and voice input should complement each other for direct graphic manipulation applications. This thesis compares the speed, usability, user-friendliness, and accuracy of a pen-only graphical editor against a pen-with-speech graphical editor.

Two versions of a graphical editor were developed which have the same functionality. One is controlled by pen input alone and the other is controlled by both pen and speech input. The pen-only editor used the tool bar for command entry and character handwriting recognition for text entry. The pen-with-speech editor used speech recognition for both command and text entry. In a pilot study using both editors, 13

computer science graduate students were asked to draw a petri net, a state diagram, a flowchart, and a dataflow diagram. Shape entry was facilitated by automatic shape recognition that transformed continuous drawing information into a perfected shape. Experimental results comparing the editor's user interfaces were then analyzed. Results show that the addition of speech made the editor slightly faster. Experimental subjects claimed this editor was more usable, perceived to be faster, and preferred to use. About half of the subjects found the editor with speech not to be more user-friendly than the pen-only editor. The accuracy of character recognition for the pen-with-speech editor was significantly inferior to the pen-only editor's handwriting recognition. The low recognition accuracy was caused by the speech recognizer's inability to distinguish between similar sounding letters.

to Mom and Dad

Table of Contents

List of Tables

List of Figures

Acknowledgments

I would foremost like to thank Dr. Takayuki Dan Kimura for his help both during the research and the writing of this thesis. I also owe a great debt of gratitude to both Mitsubishi Electric Corporation of Japan for their donation of the experiment pen computer and Palm Computing for letting me use their beta Windows version of Graffiti for free.

The development of the Picasso editors used for this thesis integrated the work of many previous and current students. None of my research could be accomplished without Mahesh Tharamal's code for the original X Window version of Picasso. This code included the excellent shape recognition algorithm designed by Van Vo. In addition, I want to thank Arun Eledath, Pete Gonzalez, Cyrus Mody, Naeem Bari and Matthew Blain for all their contributions to the current versions of Picasso.

I would also like to thank the 13 experiment subjects who donated their time for my research. Finally, I want to thank both Dr. Kenneth Goldman and Dr. Stan Kwasny for their comments and suggestions.

Chapter 1

Introduction

In the last half a century, the world has seen computers transform from large room-sized calculators into sleek portable design and information tools. These new machines are now constantly changing the way that people work, play, and think. This computer evolution has affected people far outside technical circles. People who once claimed that computers were not for them have been enticed or forced to use them. The increase of the computer user population has produced a large base of novice users that want to harness the power of modern computers without needing an extensive technical background.

The computer science field of human-computer interaction studies the methods of communication between a computer and it's user. For computer interaction to occur, a common interface must be devised from which both computer and user can translate information to and from. In the first generation of computers, interfaces were designed such that users were required to do much more information translation than the computers. Because of this, computer users consisted mainly of specialists whom knew the details of how computers work. Today's average user does not understand computers at this low level. In order to compensate, modern computers need to interface with their users at a level much closer to natural human communication.

Early computer interfaces consisted of users typing textual keyboard input and the computer displaying textual video output. In 1981, Xerox introduced Star. Star was the first commercial computer system to incorporate the modern GUI (Graphic User Interface) keyboard and mouse interface that is commonplace today. In a retrospective

article by many of Star's designers [10], the logic behind this interface shift is explained as:

> Traditional computer systems require users to remember and type a great deal just to control the system. This impedes learning and retention, especially by casual users. Star's designers favored an approach emphasizing recognition over recall, seeing and pointing over remembering and typing...They wanted users to feel that they are manipulating data directly, rather than issuing commands to the system to do it.

For the last several years new user interface technology has been rapidly evolving from the standard GUI interface. There are several major factors that have been motivating this evolution. The demand for smaller and lighter computers will soon force the exclusion of a keyboard from portable computers [4]. The power of today's CPUs have reached a stage where they are potent enough to run the complex AI recognition software necessary to process input from speech, handwriting, gesture, and other interfaces. Finally, virtual reality has driven researchers to find new ways to communicate with the computer and to use multiple communication channels cooperatively in parallel.

In a pen user interface, computer users convey information with the computer similar to the way that people communicate using pen and paper. Users are able to communicate information through handwriting, pointing, and drawing on a video display. Based on it's interaction methodology, the pen user interface has been heralded as more natural and powerful than the mouse user interface [11, 12, 14, 18]. A computer with a pen interface can control an application entirely by direct manipulation combined with handwriting and gesture input.

At the 1991 IEEE Workshop on Visual Languages, a panel of researchers predicted that the pen user interface would be significant in the next generation of portable computers [13]. Nearly 5 years later, the panel's foresight is becoming apparent. Pen computers are compact portable computers that include a flat-screen display and a pen user interface. Because of their self-contained functional interface capabilities, these computers are gaining popularity as people look for a convenient solutions for lightweight mobile computing [5].

Speech recognition has been considered by many to be the dominant input interface of the future. The technology is only now reaching a state of usability for large scale applications outside of research institutions. Research has shown that speech can be used to reduce task completion time and increase user satisfaction in applications when combined with mouse and keyboards [19, 28, 32, 34]. Researchers are considering the possibility that the addition of voice recognition to pen-only interfaces will show similar benefits to the overall interface [20, 25, 38].

A pen-only input user interface relies on continuous gesture and handwriting recognizers which are often slow, inaccurate, and error prone for command and text entry [3, 13, 15, 35]. Speech recognition has been shown to be a faster and often more favorable input interface than keypresses [19, 29, 32, 34] and mouse [17, 28] for giving commands in various engineering, control, and design systems. Also, it is possible that the input speed of speech recognition shown in giving speech commands could be used as a fast text entry mechanism in a keyboardless system. Speech recognition interfaces lack the pen's capability of direct manipulation in addition to having significant recognition inaccuracies of it's own [6, 30]. Oviatt proposes the following strategy, "Combine naturally complementary modalities in a manner that optimizes the individual strengths of each, while simultaneously overcoming each of their weaknesses." [25] Using this strategy, one could see how pen and voice could complement each other well for in a combined interface.

A multimodal interface is an interface which uses multiple communication methodologies together in synergy to accomplish tasks. Some research has examined the design of pen and voice multimodal interfaces [9, 22, 23, 24, 37]. These studies have shown that pen and speech recognition are capable of complimenting each other favorably. However these studies based their results on simulated experiments combined with assumptions rather than on a designed system analysis. This thesis takes an experimental approach to find whether pen-based graphical editing can benefit from the addition of speech recognition.

This thesis is arranged as follows. Chapter 2 gives the problem definition and background research for this thesis. Chapter 3 details the method for the proposed experiment to determine the benefits of speech recognition. Chapter 4 first details the

development of the experiment and the experimental tools. The chapter goes on to explain the execution and results of the experiment. In Chapter 5, the experiment is critically evaluated. The future directions of research are discussed in Chapter 6 and the conclusion is given in Chapter 7.

Chapter 2

Problem

This chapter defines the problem to be solved by this research and its significance.

2.1 Problem Definition

The problem is to prove the following thesis: Speech recognition can increase the speed, accuracy, usability, and user-friendliness of a pen-only interface in editing graphic diagrams.

According to Philip Cohen and Sharon Oviatt, "Portable computing and communications devices will soon be too small to allow for use of a keyboard, implying that the input modalities for such machines will most likely be digitizing pen and voice..." [4] Although new trends in the industry point to pen and voice interfaces having a major impact in the future [5, 24], our literature search has found little scientific research showing that a multimodal pen and voice interface is useable in an application that was not simulated. This simulated research has a human intermediary whom intelligibly translates spoken and written input to computer application actions. While simulated research is an excellent way to investigate the potential characteristics of an interface, additional experimental research should be performed on developed applications to investigate the behavior of existing interface technology.

2.2 Related Work

The pen user interface is more natural and powerful than the mouse user interface [12, 14, 18], mainly because the pen can control location accurately. Due to this high location accuracy, the pen can draw and point better than the mouse.

Apte & Kimura conducted a study showing that drawing and editing diagrams with a graphics editor is twice as fast when using pen as compared to mouse [1]. In this study, an editor was designed which uses a shape recognition algorithm to transform user drawn stokes into symmetrical perfected shapes [2, 11]. Experimental subjects were required to draw four common diagrams. In order to bypass text entry interface issues, the study did not include text in the diagrams. Two diagrams were drawn with a pen-only user interface and two diagrams were drawn with a mouse-only user interface. The order that the diagrams were drawn and the input modality used was varied between subjects. The subjects in the study found that the pen-based editor was easier for resizing and moving shapes. However, half the subjects did not like the small screen size of the pen computer.

Even with the pen's powerful input characteristics, a pen-only interface might not commonly replace the keyboard and mouse [25]. A pen-only interface can not duplicate a keyboard's functionality effectively in most applications. Handwriting is rarely as fast as typing and has a lower accuracy rate [3, 15]. Moving the pen to button bars and through menus slows down user command selection as compared to accelerator keys. Gestures can speed up command entry for pen significantly and are often preferred over the keyboard [31, 39]. However, gestures are often ambiguous and accurate gesture recognition is extremely difficult [35]. Although algorithm advances will improve recognizer error rates and lessen recognizer ambiguity, a keyboard is a much quicker interface for text and command entry. Just as modern GUI systems improve the keyboard interface by adding mouse's direct manipulation functionality [10], another input device added to the pen-only interface should be able to improve the overall interface.

Speech is one of the most natural and quickest forms of communication for humans [16]. One would expect the same for human-computer communication. Damper argues that speech will only be useful as a 1-out-of-many selection device [6]. Damper goes on to

state, "Contrary to apparently widely-held belief, speech is a poor candidate as a universal input medium."

Murray et al. conducted an extensive listening typewriter simulation [21]. Subjects responded to speech input in a polarized manor with extreme positive and extreme negative opinions. Murray et al. concluded, "Speech-only listening typewriters are slow and inefficient, and are thus likely to only be acceptable in situations where 'hands-free' input is absolutely essential."

Despite the previous research, except as 'hands-free' interfaces, it is still unclear whether complex speech-only input interfaces are useful. However, the following research shows that speech input can improve user interfaces when combined with other input devices.

Gale Martin was one of the first researchers who investigated multimodal speech interfaces [19]. Speech and other input modalities were used to replace text keyboard commands in a VLSI chip design package. The original interface used text keyboard commands to specify actions, and mouse for drawing and pointing operations. Martin compared the time to complete command sequences with a non-menu mouse interface, accelerator key input, and the original text interface with the speech interface. There was negligible advantage to inputting the commands via speech rather than the mouse. However, there was a 24% speed advantage for speech over the accelerator keypresses and a 108% speed advantage over the original text command interface. In addition, users were able to complete 62% of the tasks with the speech interface as compared to 38% of the tasks when speech was unavailable. Martin noticed that users spent more time looking at the design on the monitor rather than the keyboard. She also concluded that one reason that speech improved efficiency is because speech provides an additional response channel over which the workload can be spread.

Lewis, Petty, and Shneiderman conducted an experiment comparing speech activated and mouse activated commands for word processing applications [17]. The speech interface reduced the average task time 18.67% over the mouse interface. The subjects also preferred using the speech interface. However, subjects did comment on problems with the speech recognition including recognition accuracy, background noise problems, inadequate feedback, and slow response time.

Pausch & Leatherby conducted a study investigating whether the addition of speech recognition to the MacDraw graphical editor has any utility [28]. Experimental subjects were allowed to use the keyboard for text entry but not for command entry. Command entry was accomplished by either speech or mouse input interfaces. Mouse input was used for all other functionality. Four diagrams were drawn by each subject. The results of the study showed a 21.23% overall task reduction time using voice commands, with a 56.78% reduction for one task. Pausch & Leatherby concluded:

> We believe that using voice in parallel with the mouse provides substantial speedup for two major reasons. First, the user is no longer required to make large mouse motions to reach menus. Second, the task has many operations where commands are given in conjunction with screen locations, which are naturally parallel operations.

As a follow up study, Pauch & Leatherby did the same experiments using mouse and keyboard accelerators for commands [29]. The results showed that keyboard accelerators had a 14.51% task reduction time for subjects that memorized the accelerators and a 9.92% reduction for the other subjects. Based on the sum of the two studies, Pauch & Leatherby concluded that speech recognition reduced task time more than keyboard accelerators.

The previous research shows that speech recognition has improved both keyboard interface and mouse interface applications. This research also shows that both speech and pen interfaces can improve graphical editing tasks independent of each other. The following research from simulated multimodal pen and voice applications shows that speech recognition has the potential to improve a pen-only user interface.

Oviatt & Olsen conducted a study to examine how people might interact with a multimodal pen/voice interface [24]. This was done by simulating applications involving form completion. Results showed that subjects liked to say words, but write digits. Constrained input formats had a higher percentage of written input in comparison to unconstrained formats. The study showed that more subjects tended to like to write data and use speech for commands. Cohen & Oviatt later summarized user preference in this study, "In a recent comparison of spoken, written, and combined pen/voice input, it was

found that 56-89% of users preferred interacting multimodally, which was perceived to be easier and more flexible." [4]

Oviatt later conducted another study examining people's interactions with a simulated dynamic interactive map manipulation system [22]. The study collected data for map manipulation tasks which varied in communication modality and presentation format. The communication modalities studied were speech-only, pen-only, and multimodal pen/voice. The presentation formats were either a highly structured map or a minimally structured map. Results showed that 94.5% of the subjects preferred using multimodal pen/voice input, 5.5% of the subjects preferred pen-only input, and no subjects preferred speech-only input. In addition, when users had a choice of which input modality to use, they chose multimodal pen/voice 100% of the time. Speech-only interfaces were found to be 32% faster than pen-only interfaces. Oviatt expected this speed difference to be larger and attributes the low speed difference to the speed impact that gestures have on the pen-only interface. Oviatt comments on task completion time using the multimodal interface:

> With respect to relative efficiency, time required to complete map-based tasks actually was shorter during multimodal[1] then speech-only input, primarily because location and shape can be designated more precisely, rapidly, and with less effort and error using the pen. Specifically, task completion time was 10% faster during pen/voice input, even though it included 13% writing. Perhaps the greatest speed advantage of multimodal input accrued from pen-based pointing and graphic marks to designate a point, line, or area on the map display, which then avoided the need to speak complex location descriptions.

In summary, previous research show that pen/voice is better than pen in specific tasks such as form entry, text manipulation, and map manipulation. This thesis demonstrates the same for graphical editing tasks.

---

[1] There is a probably a typographical error in the original paper here.

Chapter 3

Method

This chapter describes the plan that was used to solve the research problem of the thesis. Our approach was experimental. Two versions of a graphical editor were constructed: One with pen input only and the other with multimodal pen and speech recognition input.

The editors were used by a group of subjects for tasks of drawing graphical editing diagrams. In the past, experiments studying user interface merits with graphical editors have had their users draw four graphical diagrams [1, 28, 29]. These same experiments measured speed improvement by the percentage of overall task time reduction and obtained subjective information through asking the subjects questions at the end of the experimental trial. We followed the same methodology used in these previous studies to collect data.

3.1 Domain

The study of a multimodal pen/voice graphical editor was chosen as a test bed to investigate the merits of a pen/voice user interface over a pen-only user interface. Graphical editing was selected as the domain of the experiment because it is a common task which has already been shown to benefit from the pen user interface [1]. A graphical editor fully exercises direct manipulation, command entry, and text entry. An additional benefit of the selection was that graphical editing research would directly contribute to future CASE tool, graphics package, and visual language research and design.

3.2 Tools

In order to compare two different user interfaces for a graphical editor impartially, two versions of a graphical editor needed to be designed with equivalent functionality. One editor contained a pen interface for input and the other had a multimodal pen with speech recognition user interface for input. From this point forward, the graphical editor with pen input only will be called the pen-only editor and the graphical editor with multimodal pen and speech recognition will be called the pen/voice editor.

Typical functionality for a graphical editor that was implemented consists of drawing, selection, moving, resizing, labeling, grouping, and cut/copy/paste operations. To implement the this functionality, the pen-only editor required a pen computer with handwriting recognition for text entry. Since gesture recognition systems are currently too inaccurate for use [35], gesture commands were not implemented. The pen/voice editor also required a pen computer and a speech recognizer was required for command and text entry. A user-dependent speech recognizer was used since they have error rates roughly three to five times smaller than speaker independent recognizers [30]. The tradeoff was that the speech recognition vocabulary needs to be trained. Finally, previous research has shown that adding shape recognition to a pen-based editor can improve the editor's speed [1, 2, 11]. Based on this research, shape recognition was added to both the pen-only and pen/voice editors.

3.3 Tasks

Tasks consisted of drawing graphic diagrams. These diagrams were selected such that they tested the functionality of the user interfaces of the editors thoroughly.

3.4 Measurements

Task completion time was observed from the tasks that were performed. The overall task reduction times were then calculated from this data. After a subject had finished all the drawing tasks, a questionnaire was completed by the subject in order to collect individualized data.

Chapter 4

Experiment

This chapter describes a pilot study design of the experiment outlined in Chapter 3. Two versions of a graphical editor were developed with identical functionality. One was controlled by pen input alone and the other was controlled by both pen and speech recognition input. The pen-only editor used the tool bar for command entry and character handwriting recognition for text entry. The pen/voice editor used speech recognition for both command and text entry. . The interface methodology for the functionality in the editors was determined through previous research and pre-experiment testing.

Thirteen computer science graduate students were selected as experimental subjects. Each subject was asked to perform four tasks in sequence. Each task consists of drawing a pre-defined diagram using either the pen-only or the pen/voice editor. The task completion time was recorded by the experimenter. Afterwards, the subject was asked to answer a written questionnaire.

The following sections present detailed descriptions of the experiment. The next chapter gives a critical evaluation of the experiment's results.

4.1 Goal

This experiment was designed as a pilot study to demonstrate the utility of the pen/voice user interface over the pen-only user interface in the application area of graphic editing.

4.2 Tasks

The drawing tasks assigned for each subject satisfied the following requirements:

- The tasks consisted of diagrams that were categorically familiar to the subjects.
- The sum of the tasks were complex enough to extensively utilize the editor's functionality.
- Both command and character entry oriented tasks were to be performed.

The following four diagrams were chosen as the objects of the experimental tasks:

4.2.1 The Petri Net Diagram

The diagram shown in Figure 4-1 was chosen to examine the time involved in drawing a diagram with extensive command entry. To complete the diagram efficiently, the subject needed to use a balanced mixture of drawing, moving, resizing, selection, and cut/copy/paste actions. Text entry is minimal in the diagram. The subjects were told to keep the general structure of the diagram rather than trying to duplicate it.

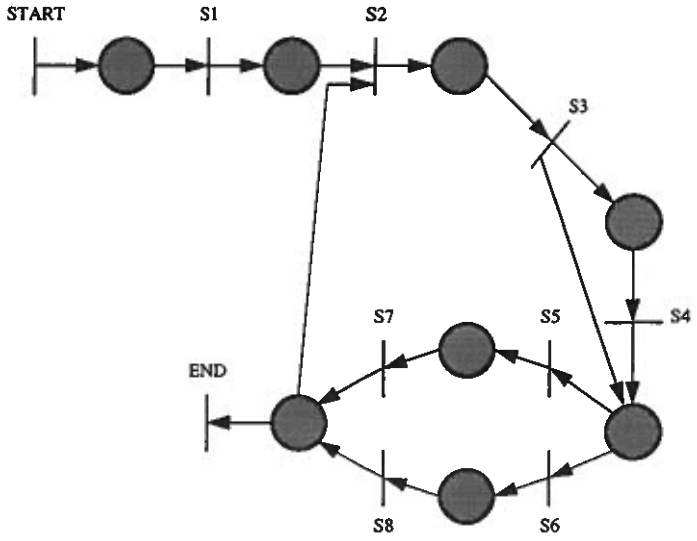

FIGURE 4-1. The Petri Net Diagram

Table 4-1 shows the entry requirements for the Petri net diagram.

TABLE 4-1. Petri Net Diagram Entry Requirements

Requirement	Value
Shapes	Ellipses - 8 Lines - 11 Directed Lines - 20 **Total - 39**
Text Characters	24

4.2.2 The State Diagram

The diagram is shown in Figure 4-2. Text entry is predominantly digits. Many parts of the diagram are symmetric, so subjects who were experienced with graphical editors were expected to finish the diagram fairly quickly. In addition, because of symmetry of the diagram it was expected that it would be easy to design an quality diagram. Hence subjects were required to focus on aesthetics while constructing this diagram.

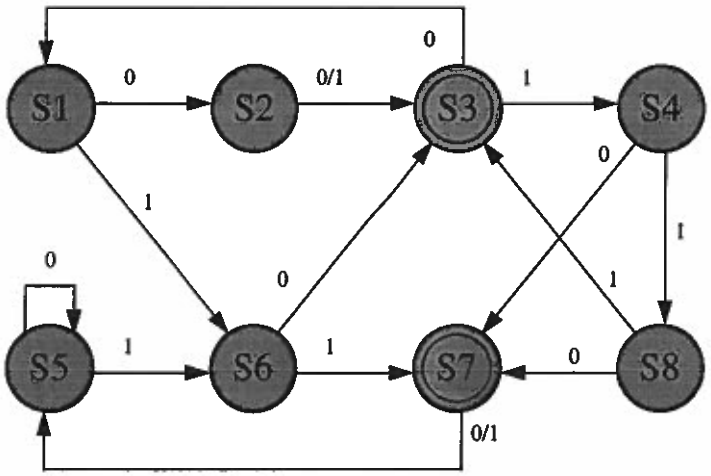

FIGURE 4-2. The State Diagram

Table 4-2 shows the entry requirements for the state diagram.

TABLE 4-2. State Diagram Entry Requirements

Requirement	Value
Shapes	Ellipses - 10 Lines - 6 Directed Lines - 14 **Total - 30**
Text Characters	34

4.2.3 The Flowchart Diagram

The diagram shown in Figure 4-3 tested the usability of the text entry methods. There are fewer shapes to draw than in the Petri net and the state diagram. Because of the variety of the shapes, this diagram tested editing skills involving the shape recognizer. To simplify shape entry, all lines in the diagram have arrowheads.

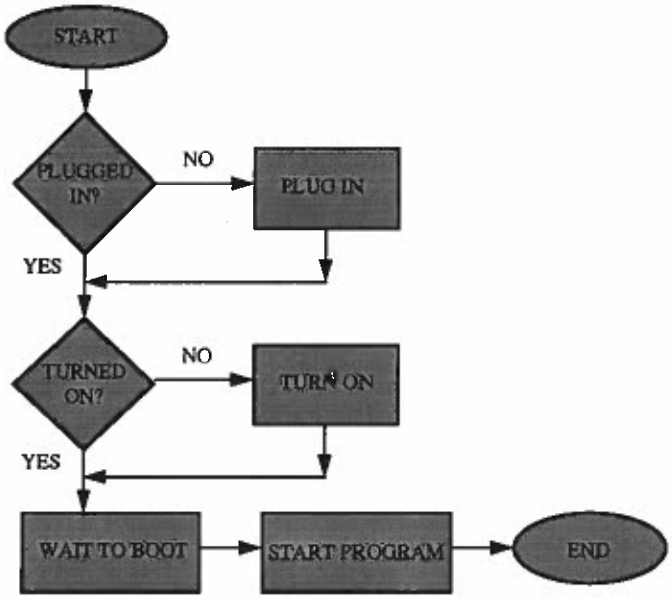

FIGURE 4-3. The Flowchart Diagram

Table 4-3 shows the entry requirements for the flowchart diagram.

TABLE 4-3. Flowchart Diagram Entry Requirements

Requirement	Value
Shapes	Rectangles - 4 Ellipses - 2 Diamonds - 2 Directed Lines - 10 **Total - 18**
Text Characters	78

4.2.4 The Dataflow Diagram

The diagram shown in Figure 4-4 was designed to test out the overall speed of each interface. The subjects were told to keep the aesthetics to a minimum. The diagram is simplistic and contains a mix of text entry, copy/paste, and line drawing.

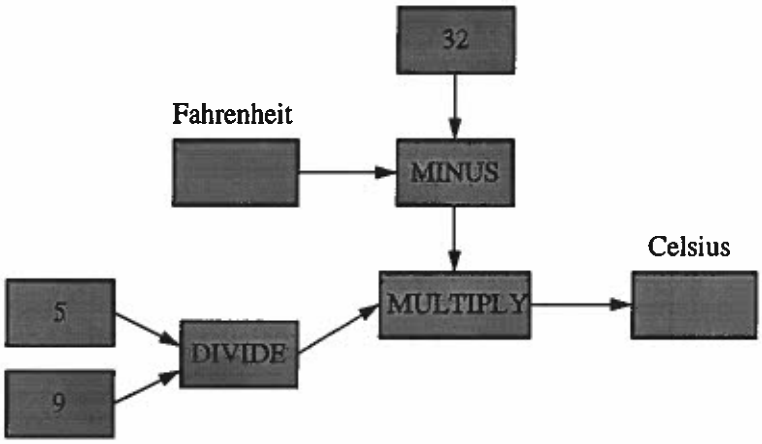

FIGURE 4-4. The Dataflow Diagram

Table 4-4 shows the entry requirements for the dataflow diagram.

TABLE 4-4. Dataflow Diagram Entry Requirements

Requirement	Value
Shapes	Rectangles - 8 Directed Lines - 7 **Total - 15**
Text Characters	40

4.2.5 All Diagrams

By examining Tables 4-1 through 4-4, one can see that there is a dispersal of the number of shapes and text characters throughout the diagrams. In addition, diagrams having more shapes have fewer characters than diagrams that have fewer shapes.

The aesthetics of the drawn diagrams were subjective. Subjects were told that they should have connecting lines touch other shapes at both ends. Specific instructions were given to subjects for each diagram which included aesthetic requirements.

4.3 Graphical Editors

There were two versions of the Picasso graphical editor specially developed for this pilot study. In the following subsections, the characteristics and features of the two editors which are relevant to the study are discussed.

4.3.1 Design Issues

Discussed here are the key design issues of the study: how and where voice is used in graphical editing which involves manipulation of objects through commands.

Commands

All actions that are used for graphical editing are initiated through commands. Commands consist of an operator which specifies an action, an operand which specifies

the recipient(s) of the action, and the parameter(s) of the action. Some commands have implied operands, e.g. 'cut' in which an operand is the previously selected objects. Some commands require not only the operand but also auxiliary parameters, e.g. 'move' commands need location and distance information besides the object to be moved.

Parameters consist of either discrete or continuous information. When continuous information such as distance and direction are entered through a discrete interface, there needs a large data range space to simulate continuous values. Pen and voice are naturally used as continuous interfaces. However, when voice is used as speech recognition, it becomes a discrete interface.

For graphical editing, there are two general categories of commands. One category consists of operators with implied or no operands. The other category requires operators, operands consisting of shapes, and parameters. The Pen and speech interfaces can specify both categories using either pen strokes or speech words.

Pen strokes have advantages over other interfaces because they can specify operator and operand information simultaneously. For this reason, pen strokes are faster than discrete interfaces. This pen stroke interface characteristic stems from the ability to specify two dimensions of continuous data from screen coordinates. Operators are specified through gesture recognition of the stroke. Operands and parameters can be specified by the starting and ending location of the stroke.

An advantage of speech words over other interfaces is that input does not rely on location and hence is usually quicker to specify. Speech words are beneficial when one dimensional discrete information such as an operator need to be entered. Specification of operands and parameters requires additional speech words and therefore more entry time.

For command entry in a graphical editor, using speech words should be faster than pen strokes for entering operators with either implied or no operands. Pen strokes should be faster for operators that have specified operator and parameter values. Preliminary study of a developed speech-only graphical editor, detailed in Appendix A.7, shows that this analysis of command entry is correct. The editors used in the pilot study were developed based on this analysis.

Text Entry

Picasso uses text in two ways: one as an object and the other as a label of an object. When text is used as a label of an object, the text's attributes such as size and location are constrained by the object.

Based on the technology we had available for handwriting recognition for our pen/only editor, it was decided that text entry would be accomplished through discrete character input for both methods of text entry. Speech character recognition was chosen over handwriting character recognition in the pen/voice editor because speech offered a discrete, rather than a continuous, input interface to produce discrete output. Theoretically, a discrete input interface should be faster to use, because continuous information does not need to be translated to discrete information. Unlike character handwriting technology, speech technology is designed to work with words and phrase input. Because of this, an accurate method for speech character entry needed to be found.

Initial testing of character entry using the intuitive method of spoken characters showed poor recognition accuracy. Similar sounding letters such as A/K, B/D/V, P/T/3, and U/2 were often confused for one another. The two options available for improved recognition were either to find a way to make the sound of spoken characters more distinct or using words for characters. Since using words for characters would involve significant time for memorization, an intuitive method for improving spoken letter was sought.

The most straightforward way to make spoken characters more distinct was to have the character repeated twice. Since digit characters sound dissimilar, digits only needed to be spoken once. This also avoids recognition inaccuracy between letters and digits.

A six subject study was done comparing single spoken character and double spoken character entry. Subjects were asked to train the spoken character set and then repeat each character five times. Each spoken character was trained four times. Statistics on the number of times characters were recognized inaccurately were collected. Subjects 1 through 3 tried double spoken character entry first and subjects 4 through 6 tried the single spoken character entry method initially. Tables 4-5 and 4-6 show the error statistics for both methods.

TABLE 4-5. Number of Errors with Single Letter Character Entry

Character	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	Totals
A	4		1	3	5	2	**15**
B	5	3	4	2	5	5	**24**
C	3		2	1		1	**7**
D		4	4	1	1	2	**12**
E	3		2	4	1		**10**
F	1						**1**
G	1	1	3	1			**6**
H			2				**2**
I			2				**2**
J					1		**1**
K		1					**1**
L					1		**1**
M							
N	1				2		**3**
O	1						**1**
P	5	3	1	4	4	1	**18**
Q	1	3				1	**5**
R							
S	2						**2**
T	3	2	1	2		2	**10**
U							
V		1	2	2		5	**10**
W		1					**1**
X							
Y							
Z	1		2				**3**
0			1				**1**
1							
2			2	1		1	**4**
3	5				1		**6**
4					2		**2**
5							
6	2				4		**6**
7							
8	1		1	1	1	1	**5**
9				2			**2**
Totals	**39**	**19**	**30**	**24**	**28**	**21**	**161**

TABLE 4-6. Number of Errors with Double Letter Character Entry

Character	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5	Subject 6	**Totals**
AA							
BB		1	1		3	3	**8**
CC	1						**1**
DD	5	1	2	1	1	3	**13**
EE							
FF							
GG	1	5					**6**
HH							
II					2		**2**
JJ		1			1		**2**
KK	3						**3**
LL							
MM		3			1		**4**
NN							
OO							
PP			1				**1**
QQ							
RR							
SS	1	3					**4**
TT	1		1			1	**3**
UU							
VV		2	1			1	**4**
WW							
XX			1		3		**4**
YY							
ZZ			1				**1**
0							
1							
2							
3							
4				1			**1**
5			2				**2**
6							
7							
8				1			**1**
9					1		**1**
Totals	**12**	**16**	**10**	**3**	**12**	**8**	**61**

There was 2.6 times the number of errors using the single spoken letter method as compared to the double spoken letter method. The double spoken letter method averaged about 10 errors per trial for an overall 94% accuracy rate. These results seemed reasonable enough to try using this method for speech character entry.

4.3.2 Tools

Technologies

There were 5 tools used in this experiment. They are detailed in the next five sub-sections. The Picasso graphical editor was used in the experiment and runs on a Mitsubishi Amity SV pen computer. Picasso can obtain character input through the external Graffiti handwriting recognizer program [26, 27]. Dragon VoiceTools speech recognition [7, 8] and a shape recognition algorithm [2] are designed into Picasso. Figure 4-5 visualizes the interaction of the tools used in the experiment.

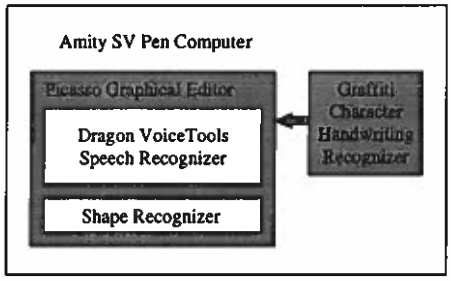

FIGURE 4-5. The Interaction of the Experiment Tools

The Picasso Graphical Editor

The original Picasso graphical editor was developed by Mahesh Tharamal for research purposes [36]. For this experiment, two versions of the Picasso graphical editor were developed for the Microsoft[2] Windows[3] 3.1 platform. One version, shown in Figure 4-6, is controlled by pen input only. The other version is controlled by multimodal pen and speech recognition input. The user manuals for these versions of Picasso are included in the Appendix A.

---

[2] Microsoft is a registered trademark of Microsoft Corporation.

[3] Windows is a trademark of Microsoft Corporation.

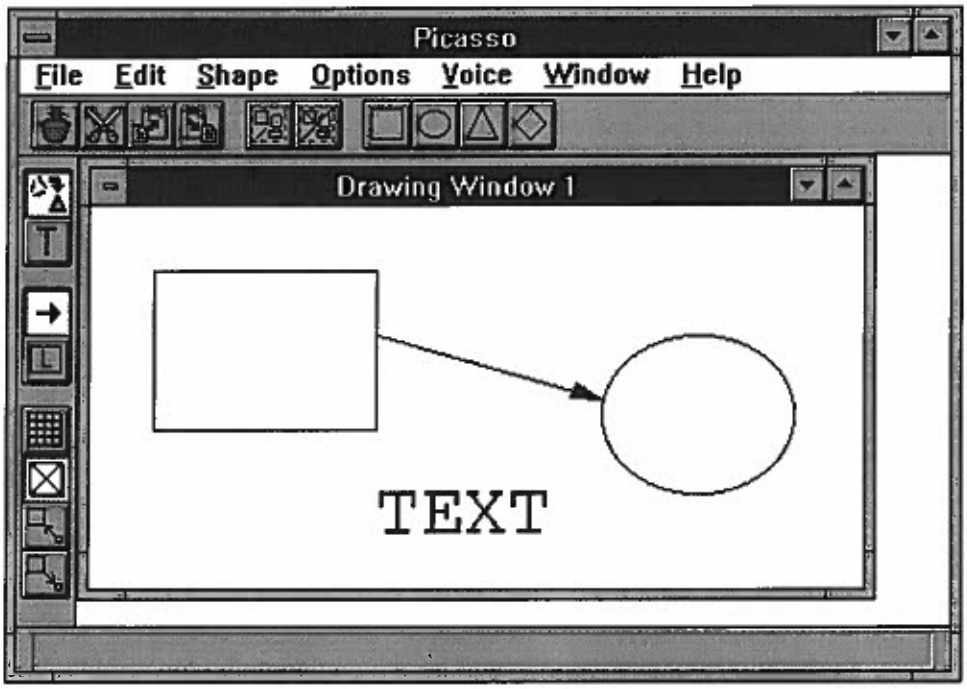

FIGURE 4-6. The Pen-only Picasso Graphical Editor

The following actions are available for both versions of Picasso: draw, move, resize, label, delete, cut, copy, paste, group, ungroup, and change the shape of the last drawn object.

The following are options that both versions of Picasso can toggle on and off: arrow heads for lines, automatic text labels for objects, display grid, snap to grid, and microphone for the pen/voice version. There are commands available to shrink and expand the grid. These grid control commands are available whether or not view grid is selected since there is a snap to grid option. The only option defaulted on is snap to grid. The direct manipulation of graphical objects for drawing, moving, resizing, and selection is done through the pen interface. The pen-only version of Picasso uses the tool bar for command entry and the Graffiti[4] character handwriting recognizer for text entry. The pen/voice version of Picasso uses built-in Dragon VoiceTools[5] speech recognition for both command and text entry. Section B.4 contains functionality reference sheets.

---

[4] Graffiti is a trademark of Palm Computing, Inc.

[5] Dragon VoiceTools and VoiceTools are trademarks of Dragon Systems, Inc.

There are two modes in the Picasso editor: Shape recognition mode translates what is drawn with the pen into a line, rectangle, ellipse, triangle, or diamond. Text mode is a text entry mode where text objects can be entered.

Text entry is entered using a character-by-character approach for both editors. In order to improve recognition accuracy, speech character recognition letters are entered by saying a letter twice. For example, to enter the character T, the spoken phrase 'tee tee' would be used. Digits and punctuation characters are only spoken once.

The Mitsubishi Amity SV Pen Computer

The Mitsubishi Amity SV pen computer has a monochrome gray scale display at 1024x768 resolution. The computer contains 20 megabytes of memory and 80 megabytes of hard drive space. The pen computer is extended by a New Media Multimedia Combo[6] sound card which uses a Shure[7] SM-10A microphone for sound input. The operating system is Microsoft Windows for Workgroups 3.1 with the Microsoft Pen SDK installed.

The Dragon VoiceTools Speech Recognizer

The speech recognition engine used in this experiment was Dragon VoiceTools from Dragon Systems[8]. The version of VoiceTools that was used for the experiment is designed for the Microsoft Windows 3.1 operating system. A 20 MHz 386 or better CPU, 5 megabytes of memory, and a multimedia sound card with a high quality microphone is required [8]. VoiceTools allows vocabularies of up to 10,000 words or phrases with 1,000 words or phrases active at any time.

VoiceTools has speaker independent and speaker dependent voice models. The speaker dependent model was used for this experiment to increase accuracy [30]. The drawback

---

[6] Multimedia Combo is a trademark of New Media Corporation

[7] Shure is a registered trademark of Shure Brothers, Inc.

[8] Dragon Systems is a registered trademark of Dragon Systems, Inc.

of using speaker dependent speech recognition models is that the vocabulary needs to be trained. The greater the number of times a word or phrase is trained, the higher the recognition rate but the vocabulary takes longer to train. Also, the smaller the vocabulary size, the better the recognition rate. In this experiment, each word or phrase in the vocabulary was trained four times. This is one less repetition than the recommended five repetitions for an application [7]. This was necessary because time was constrained in the experiment and is justified because the vocabulary was a relatively small 66 words. Vocabulary information is stored separately from user specific information so that multiple users can train their voice to the same vocabulary easily.

VoiceTools has the capability of continuous digit recognition. However, continuous digit recognition requires a great deal of memory and CPU resources. In this experiment, only discrete recognition was used in order to maximize speed, and keep a consistent speech interface between all spoken words and phrases.

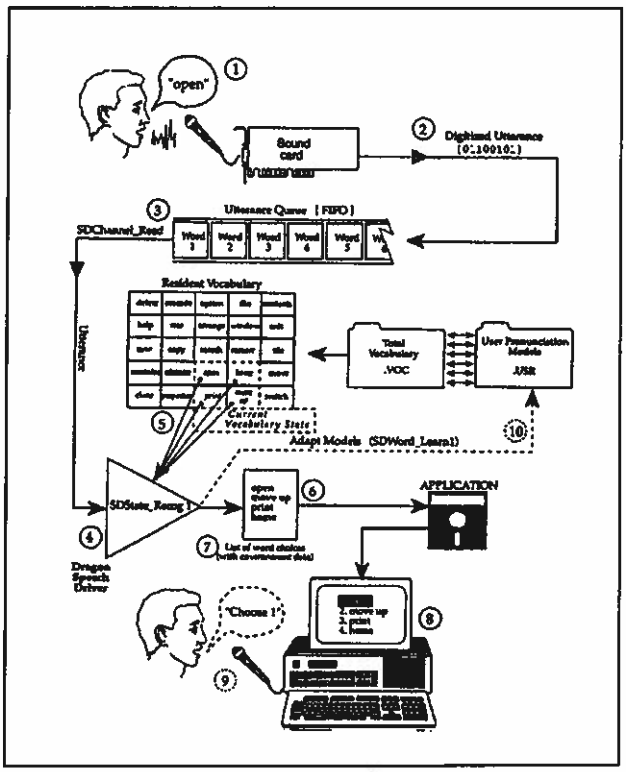

FIGURE 4-7. Sequence of Steps for Dragon VoiceTools Speech Recognition[9]

[9] Figure taken from page 3-10 of the Dragon VoiceTools Programmer's Guide [7]

Figure 4-7 shows the sequence of events that take place during Dragon VoiceTools speech recognition. The Dragon programmer's guide describes these steps as follows:

1. The user speaks into the microphone.
2. The sound card converts the electrical signal from the microphone into digital form.
3. The Dragon Speech Driver places the digitized utterance in the utterance queue.
4. Your application tells the Dragon Speech Driver to start recognition from the appropriate vocabulary list (state).
5. The Dragon Speech Driver compares the utterance to the active speech models.
6. The Dragon Speech Driver returns a list of most likely words to your application.
7. Your application chooses a word from the list and gets environment data (for example, keystrokes) for the word.
8. Your application takes some action based on the word that was spoken.
9. Your application asks you to choose the word spoken in step 1 from a list of possible words (optional).
10. Your application asks the Dragon Speech Driver to adapt the speech and language models for the word (optional). [7]

Another speech recognition system that was considered for this experiment was IBM Continuous Speech Series. Only VoiceTools and Continuous Speech Series were considered based on price, the ability to be used in the Microsoft Windows 3.1 environment, and the ability to use a PCMCIA sound card that could be used in the Mitsubishi Amity SV pen computer. Continuous Speech Series was not used because of the uncertainty about running on our pen computer platform.

The speech recognition for the pen/voice editor needs to be trained before it is used. There are 26 speech commands, 26 letters, 10 digits and 4 miscellaneous characters in the speech vocabulary for a total of 66 words.

The Graffiti Character Handwriting Recognizer

Graffiti, from Palm Computing[10], is a handwriting recognition program which translates a specially designed single stroke drawing into an alphanumeric character [26, 27]. Graffiti is capable of inputting lowercase and uppercase letters, digits, common punctuation, and many commons symbols. In order to input the characters, they must be entered as show in Figure 4-8. Palm Computing's studies show that "most users can learn to enter these strokes fast enough to achieve more than 30 words per minute with 100% accuracy." [27]

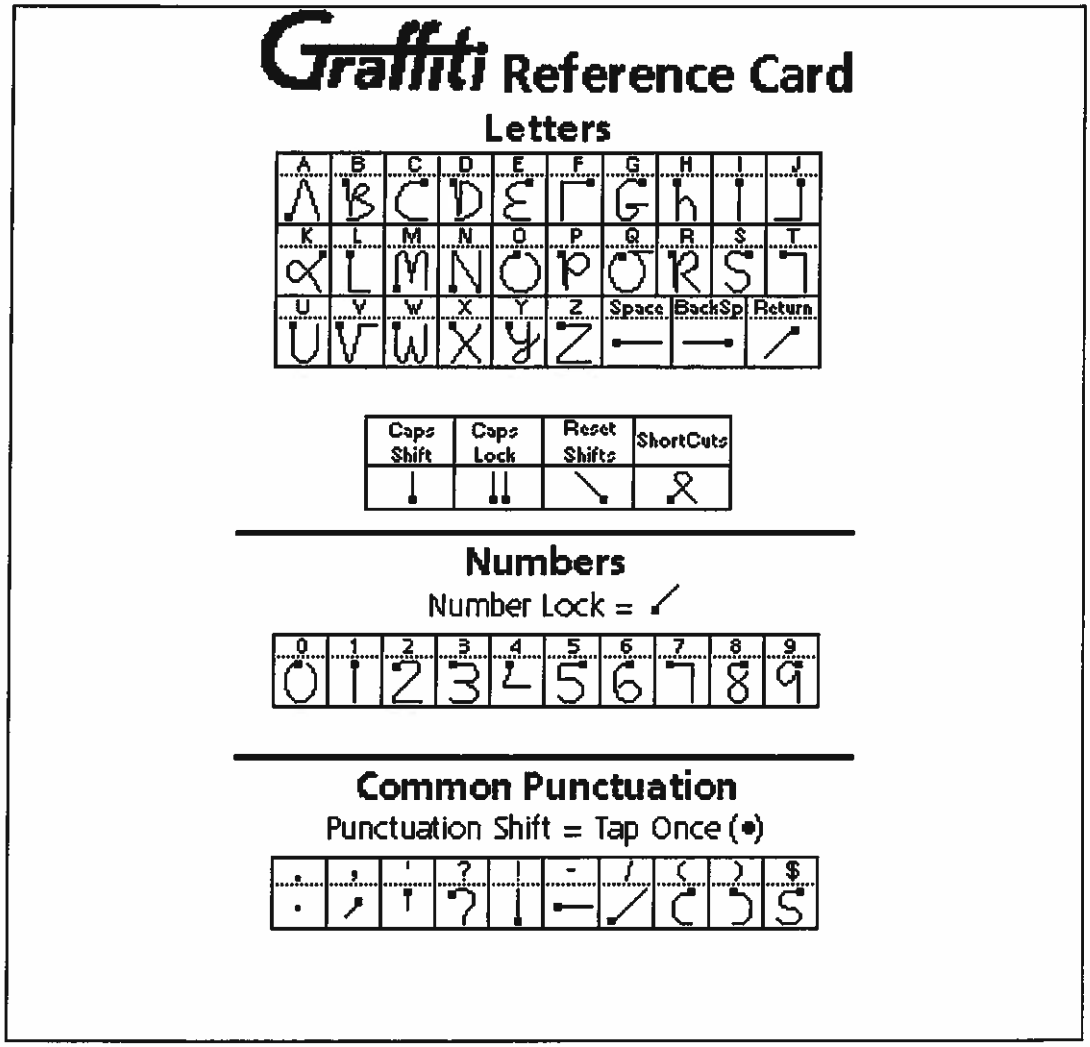

FIGURE 4-8. Graffiti Reference Card

[10] Palm Computing is a trademark of Palm Computing, Inc.

The Shape Recognizer

The shape recognition engine used in this experiment was developed by Van Vo [2]. It transforms continuous drawing information into a perfected shape. The recognition engine recognizes rectangles, ellipses, circles, diamonds, triangles, and lines. All the diagrams in Figures 4-1 through 4-4 consist of these shapes. Entered shapes can contain multiple pen strokes as long as they are entered before a set time-out period. The recognition rate of the shape recognition has been measured up to 98% accuracy. Figure 4-9 shows an example of a shape recognition transformation.

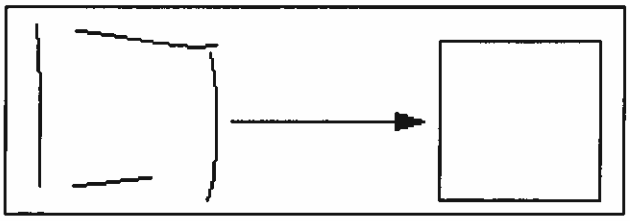

FIGURE 4-9. An Example Shape Recognition Transformation

4.4 Subjects

Computer Science graduate students were selected for this pilot study for several reasons. One reason was that they have an extensive computer background and can to learn use the two versions of the editor quickly. They also can give comparative feedback about the interfaces that novice users would not be able to give. Finally, they are readily available and cooperative. One problem with this subject set selection is that long time computer users might have problems using interfaces which vary significantly from what they are accustomed to.

Originally twelve subjects were chosen for the experiment. A thirteenth subject was added to collect results for a task that was unfinished by a previous subject.[11] This number was partially chosen for practical reasons and past similar research used either twelve or sixteen subjects [1, 28, 29].

---

[11] In the middle of drawing the Petri net diagram with the voice/pen editor, subject 9 accidentally selected all the objects and then spoke the delete command. The task was not redone.

4.5 Procedures

4.5.1 Task Assignment

There were four diagrams that were drawn in this experiment. These diagrams are shown in Figures 4-1 through 4-4. Each subject drew all four diagrams. Two were drawn with the pen-only editor and two were drawn with the pen/voice editor. The editor used for each diagram was varied between subjects. The six possible editor task combinations for the subject experiments are show in Table 4-7.

TABLE 4-7. Editor Task Combinations for the Experiment Diagrams

	Petri Net Diagram	Flowchart Diagram	State Diagram	Dataflow Diagram
Combination 1	PEN	PEN	PEN/VOICE	PEN/VOICE
Combination 2	PEN	PEN/VOICE	PEN/VOICE	PEN
Combination 3	PEN/VOICE	PEN/VOICE	PEN	PEN
Combination 4	PEN/VOICE	PEN	PEN	PEN/VOICE
Combination 5	PEN/VOICE	PEN	PEN/VOICE	PEN
Combination 6	PEN	PEN/VOICE	PEN	PEN/VOICE

Since the experiment was designed to have twelve subjects and six editor task combinations, each tasks combination was done by two subjects. Subjects were matched up randomly with the task combination they did. Subjects alternated editors between tasks so that they did not become accustomed to a particular editor. Since there were two subjects for each task combination, one subject started with the pen-only editor and one started with the voice/pen editor.

Each of the four diagrams can be drawn with two different editors and hence there are eight different diagram input tasks. In order to distribute all combinations the diagram input tasks evenly through the four subject tasks, twenty-four subjects were needed. In order to design the experiment for twelve subjects, diagram tasks were distributed arbitrarily so that each task appeared three times for each subject task order number. Table 4-8 shows the final ordering of tasks for the twelve subjects.

TABLE 4-8. Diagram Input Tasks for Each Subject

	Task 1	Task 2	Task 3	Task 4
Subject 1	Petri Net (P)	State Diag (PV)	Flowchart (P)	Dataflow (PV)
Subject 2	State Diag (PV)	Petri Net (P)	Dataflow (PV)	Flowchart (P)
Subject 3	Dataflow (P)	Flowchart (PV)	Petri Net (P)	State Diag (PV)
Subject 4	Flowchart (PV)	Dataflow (P)	State Diag (PV)	Petri Net (P)
Subject 5	State Diag (P)	Petri Net (PV)	Dataflow (P)	Flowchart (PV)
Subject 6	Petri Net (PV)	State Diag (P)	Flowchart (PV)	Dataflow (P)
Subject 7	Flowchart (P)	Dataflow (PV)	State Diag (P)	Petri Net (PV)
Subject 8	Dataflow (PV)	Flowchart (P)	Petri Net (PV)	State Diag (P)
Subject 9	Flowchart (P)	Petri Net (PV)	Dataflow (P)	State Diag (PV)
Subject 10	Petri Net (PV)	Flowchart (P)	State Diag (PV)	Dataflow (P)
Subject 11	State Diag (P)	Dataflow (PV)	Petri Net (P)	Flowchart (PV)
Subject 12	Dataflow (PV)	State Diag (P)	Flowchart (PV)	Petri Net (P)

4.5.2 Instructions

The following sub-sections overview the experiment that was performed. All handouts, instructions, and diagrams for the experiment are given in Appendix B.

Background

Each of the 12 subjects chose a time that was convenient for them to do their experiment trial during a one week period of time. The subject was asked to read the thesis experiment preparation handout in Section B.1 before the experiment. This handout told the user the goal of the experiment, the format of the experiment, and some instructions on how to do the experiment. All subjects said that they read the handout before doing the experiment.

The experiment trials were conducted in an isolated setting so that the subjects didn't feel pressured and background noise would not affect the speech recognition. Because of the large amount of material that the subject needed to learn, the subject was free to ask

questions at any point during the experiment trial. In addition, the subjects had access to the reference sheets in Section B.4 that overview the editor's functionality. A subject's experiment trial was designed to take approximately two hours. When a subject completed their part of the experiment, they were told not to discuss anything about the experiment with other subjects.

Execution

Each experiment started with some introductory statements. This was followed by three experimental phases. The exact steps that took place during these phases are given in the tester instructions included in the Section B.2.

The first phase of the experiment was a preparation phase. The subject learned how to use the Graffiti character handwriting recognition program, followed by learning the basic functionality of Picasso common to both editors. Finally, the subject trained the speech recognition vocabulary used in the pen/voice version of the editor.

The second phase of the experiment involved learning how to use both editors and drawing the four diagrams shown in Figures 4-1 through 4-4. The subject started the phase by learning the version of the editor which will be used to draw the first diagram task. This was immediately followed by drawing the first diagram task. Then the subject learned to use the other version of the editor and drew the second diagram task. Finally, the subject drew the third and fourth diagram tasks.

During the diagram drawing tasks, the subjects were timed and all recognition engine uses and errors[12] were counted. The subjects were told only to try to enter a text character that was not being recognized three times before moving on. All questions, relative to the editor, that were asked during the drawing time were counted to act as a gauge of how well the subject understood the version of the editor that was being used. The subject's diagram drawings were saved to obtain more data. All experiment statistics were collected on either the pen or the pen/voice testing sheets shown in Section B.7.

---

[12] Both mis-recognition and non-recognition errors were counted.

The final phase of the experiment consisted of closing remarks and a questionnaire. The questionnaire consisted of both circle the appropriate response and short answer questions. Questions were asked to determine the subject's computer interface and graphical editing background. Questions were also asked to determine the subject's preferences and opinions of the two versions of the editor. The subject was also able to write any remarks about the fairness of the experiment and any general comments.

4.6 Data

The following sub-sections give the calculated results of the experiments. The subject's diagram drawings are included in Appendix C.

4.6.1 Background Experience of the Subjects

Figure 4-10 shows the subject responses and response totals for the first seven questions on the questionnaire that are trying to gauge the computer, interface, and graphical editing experience of the subjects. The figure can be read as follows: a little - 2 [1,3] conveys that 2 subjects, which were subject 1 and subject 3, picked a little for their response.

1. Graphical Editor Experience:
   - none - 1 [12]
   - a little - 2 [1,3]
   - some - 6 [2,4,7,8,11,13]
   - a lot - 2 [5,9]
   - loads of - 2 [6,10]

2. PenLab Editor Experience:
   - have - 4 [2,3,5,6]
   - have none - 9 [1,4,7,8,9,10,11,12,13]

3. GUI Computer Usage:
   - never - 0
   - sometimes - 5 [1,4,7,8,13]
   - always - 8 [2,3,5,6,9,10,11,12]

4. Pen Computer Experience:
   - have - 5 [2,3,5,6,7]
   - have none - 8 [1,4,8,9,10,11,12,13]

5. Shape Recognition Experience:
   - have - 4 [2,3,5,6]
   - have none - 9 [1,4,7,8,9,10,11,12,13]

6. Handwriting Character Recognition Experience:
   - have - 4 [2,3,5,7]
   - have none - 9 [1,4,6,8,9,10,11,12,13]

7. Speech Recognition Experience:
   - have - 2 [2,8]
   - have none - 11 [1,3,4,5,6,7,9,10,11,12,13]

FIGURE 4-10. Totals of the Questionnaire Experience Questions

The relevance of this data is to determine the background knowledge of the selected subjects.

4.6.2 Opinions of the Subjects

Figure 4-11 shows the subject responses and response totals for the four questions on the questionnaire that ask users their preferences and opinions about the two editors. The figure can be read as follows: neither - '2 [10,13]' conveys that 2 subjects, which were subject 10 and subject 13, picked neither for their response.

8. Faster Editor:
   pen-only - 1 [9] voice/pen - 12 [1,2,3,4,5,6,7,8,10,11,12,13]
   neither - 0

9. More Usable Editor:
   pen-only - 2 [1,9] voice/pen - 11 [2,3,4,5,6,7,8,10,11,12,13]
   neither - 0

10. More User-Friendly (Comfortable) Editor:
   pen-only - 4 [1,3,7,9] voice/pen - 7 [2,4,5,6,8,11,12]
   neither - 2 [10,13]

11. Preferred Editor:
   pen-only - 2 [1,9] voice/pen - 11 [2,3,4,5,6,7,8,10,11,12,13]
   neither - 0

FIGURE 4-11. Totals of the Questionnaire Editor Opinion Questions

The relevance of this data is to show the subjective opinions of the subject population.

4.6.3 Statistics of Diagram Drawings

For each of the four types of diagrams drawn by the subject, there are three different tables. These tables consist of Tables 4-9 through 4-20. The first table gives the statistics for the pen-only editor. The second and third tables gives the statistics for the pen/voice editor. Table 4-21 through Table 4-23 gives overall diagram statistics.

Individual Diagram Statistics

The pen-only tables includes Graffiti character error counts and recognition accuracy, the Graffiti characters that subjects had difficulty writing, the number of questions asked while drawing the diagrams, the number of imperfections in the final diagram, and the diagram drawing time for each subject.

The first pen/voice tables includes speech character error counts and recognition accuracy, the speech characters that subjects had difficulty speaking, the number of characters that are incorrect in the final diagram, the number of questions asked while drawing the diagrams, the number of imperfections in the final diagram, and the diagram drawing time for each subject.

The second pen/voice tables includes the number of speech commands used, the number of speech command errors, the accuracy rate of the commands, the speech commands that subjects had difficulty speaking, and the diagram drawing time for each subject.

The accuracy rates given in the three tables are approximations. The subjects spoke and wrote too quickly to mark down every time the speech and handwriting recognizer was used for both command and text entry. The number of recognizer errors is accurate since subjects always paused whenever an error occurred. The number of characters spoken in a diagram is approximated as the number of characters in the diagram added to twice the number of errors[13] subtracted from the number of incorrect characters in the completed diagram. Recognizer accuracy results were calculated from the recognition errors and uses. All pen-only diagrams had no character errors in the completed diagram.

Any helpful comments I gave the subjects while drawing were counted as questions that the subject asked. The number of imperfections in the final diagram was calculated by approximating the number of objects that needed actions to make the target diagram. Diagram imperfections do not include any wrong characters in the completed speech diagrams.

---

[13] Twice the number of errors are counted to account for the error character and it's corresponding backspace.

Petri Net Diagram

TABLE 4-9. Pen-only Editor Petri Net Diagram Statistics Table

Subject Number (Task #)	Char Rec Errors	Char Rec Accuracy Rate	Char Rec Problem Characters (2+ Errors)	Questions Asked or Comments	Diagram Imperfect	Diagram Drawing Time
1 (Task 1)	5	85%	BKSP	4	3	15:22
2 (Task 2)	6	83%	N	3	0	18:29
3 (Task 3)	0	100%		3	2	25:08
4 (Task 4)	0	100%		2	4	12:45
11 (Task 3)	2	93%		0	1	15:16
12 (Task 4)	3	90%		1	1	15:49
AVERAGE	**3**	**91%**		**2**	**2**	**17:08**

TABLE 4-10. Pen/Voice Editor Petri Net Diagram Statistics Table 1

Subject Number (Task #)	Speech Rec Character Errors	Speech Rec Character Accuracy Rate	Speech Rec Problem Characters (2+ Errors)	Final Wrong Chars	Questions Asked or Comments	Diagram Imperfect	Diagram Drawing Time
5 (Task 2)	3	90%		0	1	1	12:49
6 (Task 1)	16	69%	A, D, S, T	4	1	8	15:20
7 (Task 4)	7	82%	A	0	0	3	17:01
8 (Task 3)	15	N/A	A, R, S, 5	N/A	5	N/A	23:08
10 (Task 1)	2	93%		0	4	3	15:36
13 (Task 2)	6	83%	D, T	0	4	1	15:09
AVERAGE	**8**	**82%**		**1**	**3**	**3**	**16:31**

TABLE 4-11. Pen/Voice Editor Petri Net Diagram Statistics Table 2

Subject Number (Task #)	Speech Command Used	Speech Command Errors	Speech Command Accuracy Rate	Speech Rec Problem Commands (2+ Errors)	Time to Draw Diagram
5 (Task 2)	46	0	100%		12:49
6 (Task 1)	67	12	82%	arrows	15:20
7 (Task 4)	36	4	89%	delete	17:01
8 (Task 3)	161	18	89%	arrows, copy, delete, group	23:08
10 (Task 1)	46	2	96%	delete	15:36
13 (Task 2)	81	13	84%	deselect all, paste	15:09
AVERAGE	**73**	**8**	**89%**		**16:31**

State Diagram

TABLE 4-12. Pen-only Editor State Diagram Statistics Table

Subject Number (Task #)	Char Rec Errors	Char Rec Accuracy Rate	Char Rec Problem Characters (2+ Errors)	Questions Asked or Comments	Diagram Imperfect	Diagram Drawing Time
5 (Task 1)	3	93%		0	1	17:40
6 (Task 2)	2	95%	/	3	2	12:09
7 (Task 3)	1	97%		1	2	11:36
8 (Task 4)	4	90%		6	1	21:16
11 (Task 1)	5	89%	/, 0	4	3	17:36
12 (Task 2)	6	87%	/	7	4	26:43
AVERAGE	**4**	**91%**		**4**	**2**	**17:50**