A Comparison of Speech vs Typed Input

Alexander G. Hauptmann and Alexander I. Rudnicky

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

ABSTRACT

We conducted a series of empirical experiments in which users were asked to enter digit strings into the computer by voice or keyboard. Two different ways of verifying and correcting the spoken input were examined. Extensive timing analyses were performed to determine which aspects of the interface were critical to speedy completion of the task. The results show that speech is preferable for strings that require more than a few keystrokes. The results emphasize the need for fast and accurate speech recognition, but also demonstrate how error correction and input validation are crucial for an effective speech interface.

1. Introduction

Although significant advances have been made in speech recognition system performance in recent years [lee89h], very few application programs use speech input. This discrepancy is based on the fallacy of equating speech recognition performance with the usability of a spoken language application. Clearly, the accuracy of the speech recognition module is a key factor in the usability of a spoken language system. This accuracy is usually measured as word recognition accuracy when the system is tested by a set of speakers reading sentences from a test corpus. The recognition accuracy is analyzed and determines the speech recognizer's performance. But there are other interface issues involved, which are frequently overlooked. The most intuitive interface issue is related to response time. It has been shown that the amount of delay introduced by the system significantly affects the characteristics of a task as well as human performance [grossberg76, rudnicky90b]. Less intuitive interface issues concern the control of the interaction. When does the system listen to the speaker and when should it ignore speech as extraneous? How can the system best signal to the speaker that it is ready to listen? How can a user verify that the system understood the utterance correctly? How does the user correct any recognition errors quickly and efficiently? These are only some of the for which there are no answers, yet.

While other researchers have found speech to be the best communication mode in human-human problem solving [chapanis81], results from evaluations of computer speech recognizers point in the opposite direction [martin80, morrisonetal84], with few, contrived, exceptions [poock82]. The community has become aware that speech applications need more than good recognition to function adequately [nye82, belldigits, leggettandwilliams84], but no solutions are offered.

Our objectives in this paper are to clarify some of the tradeoffs involved when given the option of using either speech or typing as an interface to an application program. We deliberately chose the simplest possible task to avoid confusing task-related cognitive factors with the inherent advantages and disadvantage of the interface modes.

2. The Experiments

Two experiments were conducted at Carnegie Mellon to contrast the input of numeric data through speech with data entry through a conventional keyboard. The experiments were essentially identical, only the method of stimulus presentation was changed. Both experiments required the subjects to enter three lists of 66 digit strings into the computer, using three different data entry modes. In experiment A, the digit

strings were presented on the screen, just above the area where either the speech recognition result or the typed input was displayed. In experiment B, the subjects had to read the digit strings from a sheet of paper placed next to the keyboard and monitor. We will refer to experiment A as the '*screen*' experiment and to experiment B as the '*paper*' experiment throughout this report.

There were 3 lists of 66 digit strings to be entered. Each data set contained exactly 11 randomly generated digit strings of length 1, 3, 5, 7, 9, and 11. The first six digit strings included one string of each length and were identical for all data sets. These first six digit strings were considered training data and removed from the transcripts before data analysis. The order of presentation for the remaining digit strings in each data set was randomized at the onset of the experiment and remained constant throughout the study.

Three data entry modes were included in the experiment.

- In the first mode **speech only**, subjects could only use speech to enter a digit string. They read the digit string out loud into a head-mounted, close-talking microphone. The speech recognizer would then analyze the speech signal and display the recognition. When the result of the speech recognizer for the digit string was not correct, the subject was instructed to repeat the digit string into the microphone. This procedure was repeated until the number displayed as the recognition was correct. If the displayed recognition result was correct, the subject would then say the word "OK" or "ENTER". The speech system would then store the number that was entered and the subject could proceed to the next number on his/her list.
- In the **speech with keyboard correction** mode, the subject would again read the digit string into the system. If the recognition was not correct, the subject was instructed to use the keyboard to enter the correct digit string terminated by a carriage return. If the recognition string was correct, or after the keyboard correction was performed, the subject again hit the enter key to store the number in the system.
- In the **keyboard only** mode, subjects typed in the digit string, which was then also displayed for confirmation and correction. If they had miskeyed the string they could correct it again using the keyboard. Once the correct digit string was displayed on the screen, subjects would hit the enter key to store the number in the system and proceed to the next number.

Each subject entered the different lists using each of the different input modes (speech only, speech with typed correction and typing only). Both experiments were replicated 3x3 Latin Square designs [myers72].

2.1. Subjects

Eighteen (18) subjects were recruited at Carnegie Mellon for an experiment in speech recognition. All subjects claimed to be casual typists. Nine (9) subjects were used in experiment A (on-screen presentation of each stimulus) and 9 subjects were used in experiment B (where the list of digit strings was presented on paper).

2.2. Apparatus

Subjects were seated in front of a SUN-3/280 computer workstation running MACH/UNIX. The keyboard for this workstation does not have a numeric keypad and all numbers had to be typed in using keys on the top row of the keyboard. The workstation was connected to the SPHINX speech recognition system [lee88, rudnicky90a]. SPHINX is a large-vocabulary, speaker-independent, continuous speech recognition system developed at Carnegie Mellon. SPHINX has reported some of the highest recognition accuracies in the speech community and includes special hardware for fast recognition times. The speech recognition vocabulary consisted of the words ZERO through NINE, OH, ENTER and OK. The grammar allowed either an arbitrary length digit string to be spoken or the words OK or ENTER. When a spoken digit string was recognized, the system displayed the result using numerals (i.e. ZERO, OH => 0; ONE => 1; TWO => 2; ... NINE => 9). Typed input was displayed without alteration on the same line as the spoken input.

The workstation was running a dedicated program to control the experiment. No other servers or processes were running. The program that controlled the experiment also recorded a log of time stamps and actions.

3. The Results

The data are presented in very abbreviated highlight form. A complete report on these experiments is available as a technical report.

3.1. Transaction Error Rate

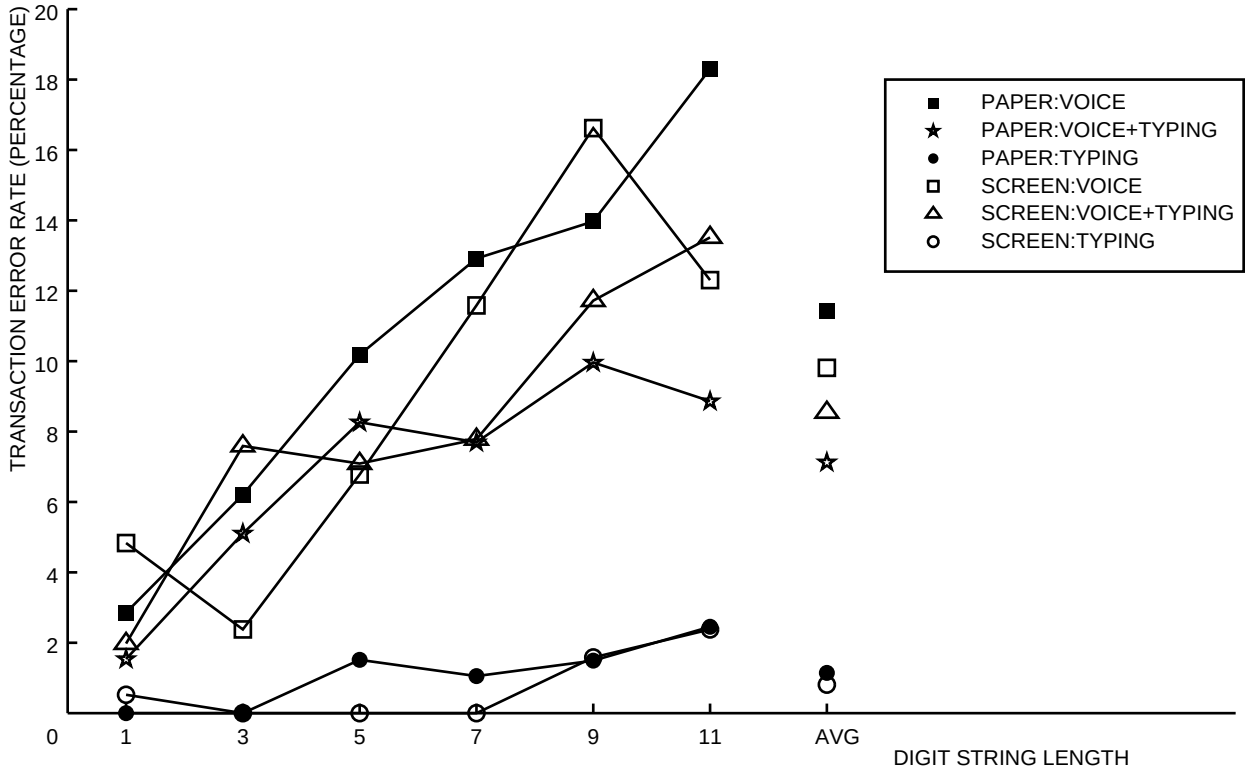
The 'transaction accuracy' of each mode was defined as the number of transactions with the system that were absolutely necessary to enter all numbers under perfect conditions divided by the total number of transactions that were actually made by the subjects. A transaction is counted each time the user speaks an utterance into the system or presses the enter key. Transaction accuracy combines recognition errors made by the speech recognizer with misspoken utterances and incorrectly spoken or typed digits. We report the transaction accuracy in terms of the transaction error rate, where the transaction accuracy percentage is subtracted from 100 percent.

The average transaction error rate in the paper experiment was 11.5 percent for the voice-only condition, 7.2 percent for the voice with typing condition and 1.2 percent for the typing condition. In the screen experiment, the voice error rate reached 9.8 percent, voice with typing was at 8.6 percent and typing

alone resulted in 0.9 percent error rate. Transaction error rate as a function of string length is plotted for both experiments in Figure 3-1.

Accuracy decreases with string length, more so for speech than for typing, the latter reflecting the cumulative effect of word errors.

Figure 3-1: Transaction error rate in both experiments



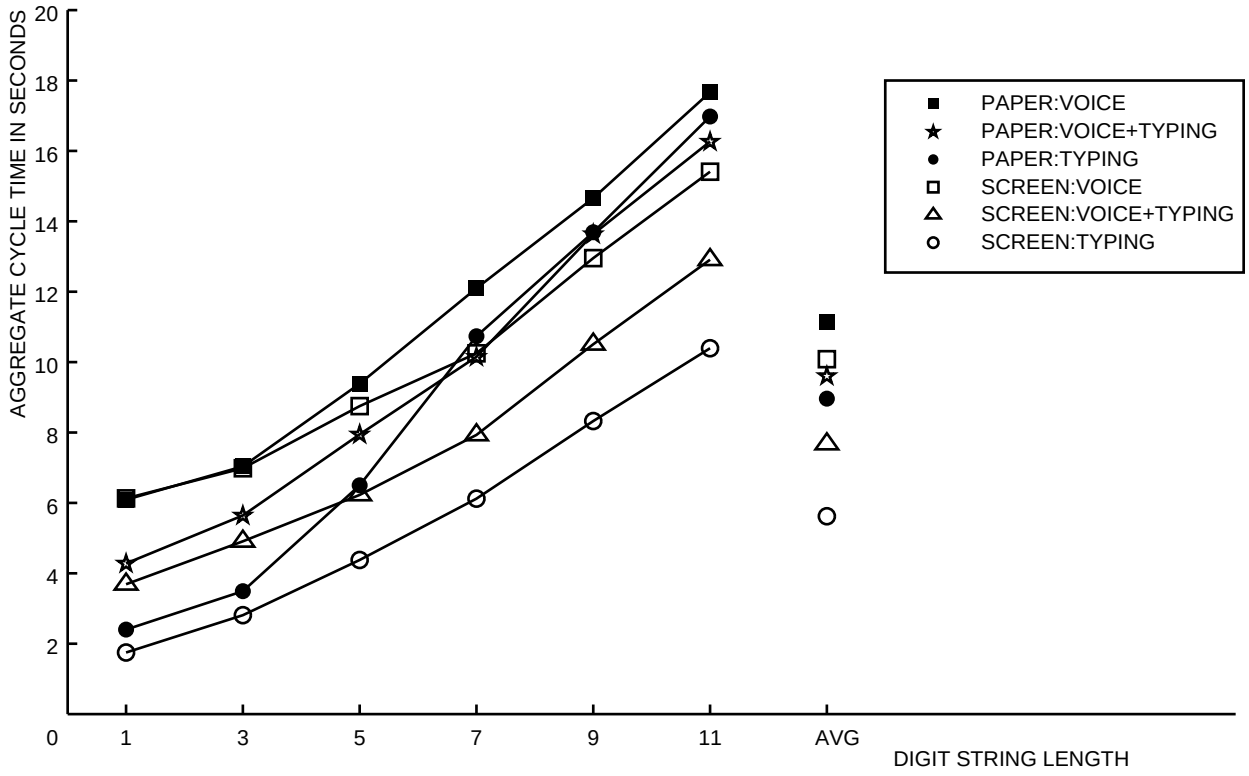
The transaction error rate is plotted for 3 input modes: voice only, voice with typing and typing only. Both the paper experiment and the screen experiment data are plotted separately for each condition and digit string length. AVG denotes the overall average for a condition.

3.2. Aggregate cycle time

To determine the efficiency of data entry under the different conditions, we measured the total time a subject needed to enter a number correctly. This aggregate cycle time includes the time elapsed before the subject began speaking after the system had displayed its prompt, the time required to produce the utterance or to type the digit string and (for speech) any system recognition time until the recognition

result was displayed. The number computed is the result of adding these times for each initial recognition attempt, each correction attempt and the final confirmation cycle. Thus this time reflects the average time to enter a digit string *correctly*, including all correction and verification time.

Figure 3-2: The aggregate cycle time to input one number correctly for both experiments



.

The aggregate cycle time is shown for each input condition (voice, voice with typing and typing) for both experiments. The plot includes each digit string length plotted separately; AVG denotes the overall average for a condition.

In the **paper** experiment, subjects averaged 11.1 seconds to enter a digit string using voice only, 9.6 seconds using voice with typing and 8.9 seconds using typing alone. In the **screen** experiment the voice mode allowed subjects to average one complete digit string entry every 10.0 seconds, voice with typing resulted in one digit string entered every 7.6 seconds and typing allowed one digit string every 5.6 seconds. This comparison is also shown in Figure 3-2, plotted as a function of digit length.

To better understand the differences between the conditions, we analyzed the aggregate cycle time in

terms of its component times. We factored out the time for the initial attempt to enter the digit string, the time for the correction cycles and the time necessary for the confirmation cycle.

The aggregate cycle time data analyzed by components for the paper experiment are shown in Figure 3-3, which includes the times for different digit string lengths. The overall averages for the paper experiment can be summarized as follows:

In the voice condition, subjects averaged 5.20 seconds for the initial digit string entry, 1.86 seconds for all corrections to the string and 4.06 seconds for the confirmation transaction. In the voice with typed correction mode, we measured an average of 5.32 seconds for the initial digit string entry, 1.70 seconds for all corrections to the string and 2.57 seconds for the confirmation transaction. In the typing condition, the initial digit string entry transaction lasted for an average of 6.23 seconds, while all corrections to the string counted for an average of .31 seconds and the confirmation cycle required 2.41 seconds.

For the **screen** experiment the aggregate cycle time component data is shown in Figure 3-4. We can summarize the overall averages for the screen experiment as follows:

In the voice condition, the screen experiment averaged 5.14 seconds for the initial digit string entry, 1.51 seconds for any corrections to the string and 3.42 seconds for the confirmation transaction.

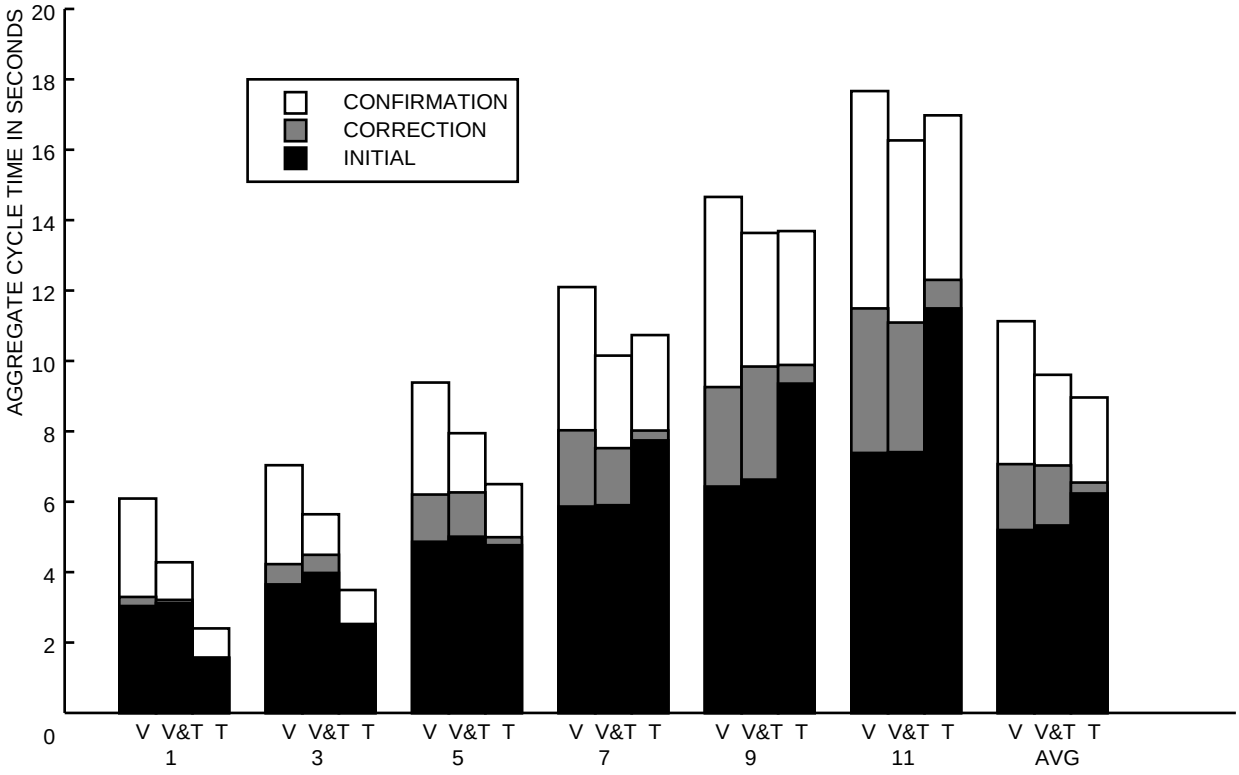
In the voice with typed correction condition, the initial digit string entry transaction took 4.69 seconds, the corrections for the string averaged 1.53 seconds and the confirmation transaction required 1.45 seconds. Finally, in the typing condition subjects needed 4.51 seconds for the initial digit string entry, 0.16 seconds on the average for all corrections to the string and 0.94 seconds for the confirmation transaction.

3.3. Input Duration time

Since the aggregate cycle times reflect system delays that were much longer for the voice conditions than for the typing condition, we also measured simple input time. That is, the time required by the subjects to type in the digit string or to speak the utterance when they tried to enter a digit string for the first time. This time is a reflection of the actual typing speed or the speech rate, and ignores all influences of reaction time or system processing delays.

In the paper experiment, we found that speaking the average digit string time took 2.49 seconds in the voice condition, 2.45 seconds in the voice with typing condition while typing time was 4.35 seconds from

Figure 3-3: The aggregate cycle time broken down by components for the paper experiment



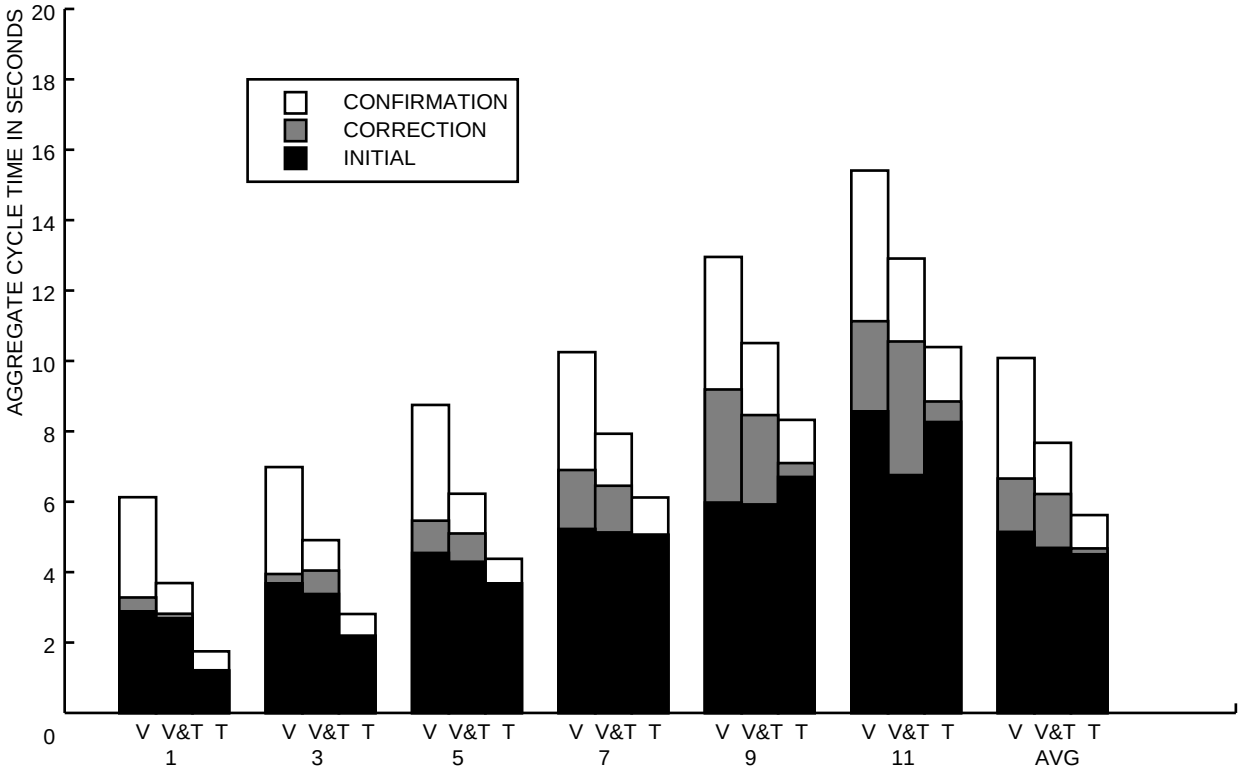
The components are composed of the initial attempt to enter a string, any corrections and the final confirmation that the string is correct. The modes are abbreviated as V=Voice, V&T=Voice with typing, T=Typing. Each digit string length is plotted separately; AVG denotes the overall averages for a condition.

the first to the last keystroke. In the screen experiment, the voice condition averaged 2.42 seconds from the beginning of the first word to the end of the utterance. The voice with keyboard correction mode required 2.44 seconds of speech, while typing lasted 3.12 seconds. Figure 3-5 shows the comparison between the typing and speaking rate for both experiments, plotted by string length.

4. Discussion

Before we draw conclusions from the results, we should note that these experiments were biased against speech recognition. One bias was introduced with speech recognition equipment that worked much slower than real time. When subjects had to wait several seconds for a response, their attention wandered and they were more likely to produce utterances that were not task related. We must also

Figure 3-4: The aggregate cycle time broken down by components for the screen experiment

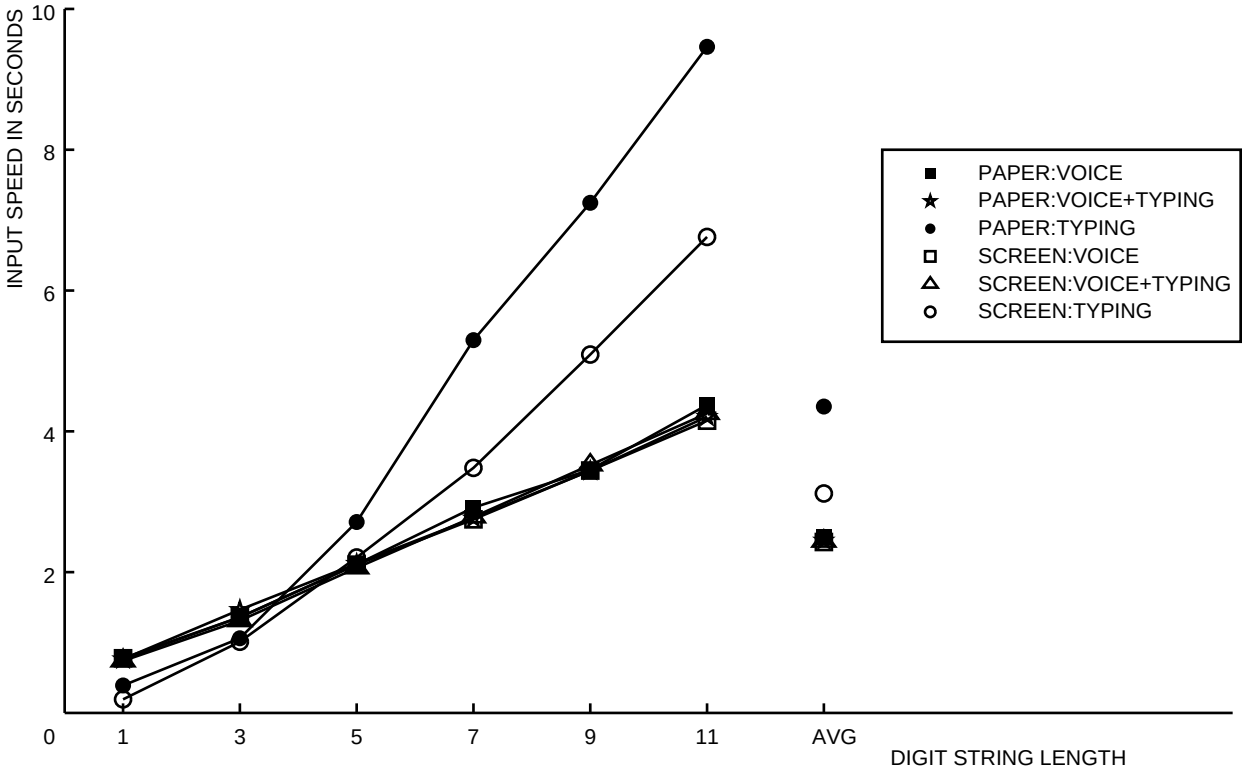


The components are composed of the ini<tial attempt to enter a string, any corrections and the final confirmation that the string is correct. The modes are abbreviated as V=Voice, V&T=Voice with typing, T=Typing. Each digit string length is plotted separately; AVG denotes the overall averages for a condition.

assume that their response-time profile is somewhat different, most likely slower than it would be otherwise. In future experiments, a speech interface with better hardware is likely to perform better than in these baseline comparisons.

Another bias came from the use of digits as the basic data unit of the task. Each digit is equivalent to one monosyllabic spoken word (except "seven" and "zero") or one typed character. In most tasks, except those concerned exclusively with alphabets and digits, we find that a monosyllabic word is more equivalent to four or five typed characters. Thus, in other kinds of tasks, the advantage of speech over typing may be more significant because of a greater typing effort involved.

Figure 3-5: The raw speech and typing rates for both experiments



The speech or typing rate measures only the time required to say or type the string, excluding all reaction time. The rates are plotted for all 3 conditions in both experiments. Data are plotted by string length as well as overall averages.

4.1. Utterance Accuracy

The utterance accuracy results show that speech requires many more interactions to complete the task than typing. This is in part due to the inadequate performance of the speech recognizer involved, which was not well suited to the digit recognition task. A better recognition system has been described by [digits89]. Speech had a strong disadvantage, especially for longer strings that needed many corrections. Even though it is not novel to assert the need for higher accuracy speech recognition, these numbers provide a reference for comparison with future, higher accuracy spoken language systems.

4.2. Aggregate Cycle Time

The basic comparisons in these experiments concern the time to enter a number correctly, including all corrections and confirmations that are required. This time was measured as wall clock time, which therefore also included system overhead time. System processing time was much longer for the speech conditions. The speech recognizer we used has reported recognition speeds of 1.5 times real time, plus several hundred milliseconds extra delay for cleanup and reinitialization of the hardware between utterances.

As our results show, speech is almost comparable to typing for the longer digit strings, but typing has a clear advantage for shorter digit strings. The cycle times for the screen experiment were quite a bit faster than those for the paper experiment. This can be attributed to the close proximity of the stimulus and the system display in the same area of the screen. In the paper experiment, the typing condition was slower than speaking, especially for longer digit strings. We attribute this effect to looking at the digit string on the paper and then looking back at the keyboard to type it in. The longer strings require more alternations of looking at the string and typing a part of it, then looking again, etc. Reading the strings was conceptually simpler. There was no need to change the eye position until the complete final result was displayed, which only occurred once after the complete digit string was read.

Considering the components of the aggregate cycle time in Figures 3-3 and 3-4, we find relatively fast initial entry times for speech, comparable to or better than the equivalent time for typing. The speech mode loses the race due to correction time. Typing accuracy avoids almost any correction, and speech loses most of its ground. In the confirmation transaction, typing is again very fast, but speech is about a constant amount slower. In the paper experiment, confirmation times also increase with string length, indicating the extra effort involved to verify long strings. One lesson that becomes clear from these data is the need to obtain better accuracy and response time for speech input. We especially need to have faster correction mechanisms, and ideally, a better system would totally avoid the need for multiple corrections in the voice-only conditions.

Effective speech interface design requires that it be possible to correct or bypass the speech modality. The effectiveness of this is shown in the improved throughput observed for the voice+keyboard condition. More generally, appropriate error-correction facilities need to be provided.

4.3. Input Duration Time

The input duration times measure the typing speed and the speech rate. These times give a lower bound on what can be done by casual users. Note, however, that these times were obtained using a standard keyboard, not numeric keypads and that they are not characteristic for expert touch typists.

Speech input is fast. This is evident if we compare average speech rate, which is estimated at about 200 words per minute with typing; even good typists cannot normally achieve this rate of input [boffandlincoln89]. Our data confirm these findings.

The input duration times in our experiments also show that real time response and accurate speech recognition are essential if a clear advantage is to be shown for speech. The average difference between pronouncing a digit string and typing one was less than 2 seconds in both experiments. Thus, if the speech recognizer has more than a 2 second delay or if the recognizer has a significant error rate (as it did in our experiments) or the interface introduces other artificial delays, speech would cease to be a desirable communication mode.

The results showed that the raw typing rate in the paper experiment was much slower than typing rate for the screen experiment. This difference can only be attributed to the extra load imposed on the users when they divide their attention between the keyboard, the screen and the paper containing the data to be entered. If a task has these characteristics, sometimes more vaguely described as 'eyes-busy' characteristics, then speech would be a preferable input channel for data entry. In our experiment even a relatively small increase in the work load for the eyes substantially changed the performance in the typing rate. Other, more demanding tasks can be expected to degrade performance in the typing mode even more.

5. Summary

With these experiments, we have shown how speech compares with typing for the entry of digit string tasks. Real word tasks, requiring more keystrokes per syllable, would demonstrate the effectiveness of speech much better.

Depending on the task, as demonstrated by our comparisons of screen vs paper presentation, speech can have tremendous advantages for casual users. The paper task required a certain visual effort, because the subject was looking back and forth between the paper containing the input data, the

keyboard and the screen result. The more a task requires visual monitoring of input (or most other kinds of cognitive distractions), the more preferable speech will become as an input medium. Of course, the vocabulary of the task must lie within the range of the speech recognizers that are available.

The screen experiment demonstrates that speech can provide an advantage despite adverse circumstances. Even when the subject has all relevant task information present in a small visual area of the screen, speech still helps out by eliminating the time spent locating keys on the keyboard. Speech allows the user achieve a cleaner separation of modalities and allows data input functions to be localized in a single channel, thus eliminating the interference produced by having to share the visual channel.

In tasks that require no visual monitoring, have very short words (e.g. digits) or when using skilled typists, speech will not demonstrate an advantage. This is particularly true when data is entered from specific, customized devices such as a numeric keypad or a specialized typewriter.

The key to building improved spoken language applications lies in better speech recognition speed and accuracy, as well as effective strategies for correcting errors and confirming correct recognitions. Improving recognition accuracy and speed lies in the domain of chip designers and speech researchers. The challenge to spoken language interface builders is to find effective strategies for managing a communication channel that is prone to errors and often requires input validation.

References

Table of Contents

List of Figures