The Beauty of Errors: Patterns of Error Correction in Desktop Speech Systems

Christine A. Halverson[1], Daniel B. Horn[2], Clare-Marie Karat[3] & John Karat[3]

[1] SRI International, 333 Ravenswood Ave, Menlo Park, CA 94025, USA.
[2] University of Michigan, CREW, 701 Tappan Street, Ann Arbor, MI 48109, USA.
[3] IBM T J Watson, Research Center, 30 Saw Mill River Road, Hawthorne, NY 10532, USA.

krys@speech.sri.com, danhorn@umich.edu, {ckarat,jkarat}@us.ibm.com

Abstract: Automatic Speech Recognition (ASR) systems have improved greatly over the last three decades. However, even with 98% reported accuracy, error correction still consumes a significant portion of user effort in text creation tasks. We report on data collected during a study of three commercially available ASR systems that show how initial users of speech systems tend to fixate on a single strategy for error correction. This tendency coupled with application assumptions about how error correction features will be used, combine to make a very frustrating, and unsatisfying user experience. We observe two distinct error correction patterns: spiral depth (Oviatt & VanGent, 1996) and cascades. In contrast, users with more extensive experience learn to switch correction strategies more quickly.

Keywords: speech recognition, error correction, speech user interfaces, analysis methods.

1 Introduction

Recent announcements of speech software focus on the speed of using speech input. However, in a recent study we found that users experienced a great deal of difficulty correcting errors, and that these difficulties had a strong influence on user satisfaction (Karat et al., 1999). In our study subjects used commercially available continuous speech recognition systems to complete a set of text creation tasks. Our focus was to compare speech and keyboard as input modalities and measure user performance and satisfaction. One part of the study (Initial Use) involved 24 users who enrolled, received training, carried out practice tasks, and then completed a set of transcription and composition tasks in a single session. In the other part of the study (Extended Use), four researchers used speech recognition to carry out real work tasks over 10 sessions each with three speech recognition software products.

What stood out, during both the execution of the study and subsequent analysis, was the frequency and variety of error correction patterns that were attempted, despite generally good recognition. Attempts to correct an error often set off a *cascade* of additional errors, which then needed to be corrected. We know that poor error handling is a significant problem for the successful commercialization of recognition-based technologies (Rhyne & Wolf, 1993), so understanding the process of error correction would be of great assistance for future designs.

This paper delves deeply into users' experiences with error correction. User strategies for error correction differed between the Initial and Extended Use groups, pointing out important lessons for developing desktop speech applications. We present data about user error correction strategies gleaned primarily from the Initial Use subjects. We compare this with diagnostic data from preliminary analysis of the Extended Use subjects.

We begin by providing some background about automatic speech recognition (ASR) systems and the role of error correction in them. Next, we outline the systems we studied, as well as the experimental design and procedure. (This is reported in depth in Karat et al. (1999).) With this as foundation we discuss the differences in error correction between the keyboard and mouse interface people are used to and ASR systems. After briefly discussing how we coded errors we present our findings about the error correction patterns that users developed and compare them with preliminary data about the patterns that more experienced users display. Finally, we conclude with suggestions for the development of future ASR systems.

2   Background

The last three decades of research and development of automatic speech recognition (ASR) technology is beginning to pay off. ASR systems translate speech input into character strings or commands, and the last two years have seen the introduction of several commercial applications for dictation and navigating the desktop which rely on ASR technology for input.

While ASR technology has come a long way, there are some fundamental factors that distinguish the use of speech as an input modality. First, speech recognition technology involves errors that are fundamentally different from user errors with other input techniques (Danis & Karat, 1995). When users press keys on a keyboard, they can feel quite certain of the result. When users say words to an ASR system, they may experience system errors — errors in which the system output does not match their input — that they do not experience with other devices. Imagine how user behaviour might be different if keyboards occasionally entered a random letter whenever you typed the 'a' key. While there is ongoing development of speech recognition technology aimed at lowering error rates, we cannot expect the sort of error free system behaviour we experience with keyboards in the near future. How we go from an acoustic signal to some useful translation of the signal remains technically challenging, and error rates in the 1–5% range are the best for which anyone should hope.

Second, speech as an input modality for computers is not as 'natural' as we might like to think. Many of today's computer users have been typing for quite some time and refer to it as a 'natural experience'. In addition, it takes time and practice to develop a new form of interaction (Karat, 1995; Lai & Vergo, 1997). Speech user interfaces (SUIs) will evolve as we learn about problems users face with current designs and work to remedy them. While the systems studied in this paper represent the state-of-the-art in large vocabulary speech recognition systems, they still have areas that could stand improvement.

3   Method

3.1   Systems

We investigated three commercially available ASR systems that shipped as products in 1998. These systems are IBM ViaVoice98 Executive, Dragon Naturally Speaking Preferred 2.0, and L&H Voice Xpress Plus, referred to as IBM, Dragon and L&H below. (Philips released their Free Speech system during the course of the study and was therefore not included.) All three products share some important features. First, they all recognize continuous speech as opposed to forcing the user to speak 'discretely' with pauses between words.

Second, all are considered *modeless systems*. Each application has integrated command recognition into the dictation so that the user does not need to explicitly identify an utterance as text or command. To do this the user must learn a command grammar (a list of specific command phrases), and sometimes a keyword that is uttered to indicate what follows as a command. In general, commands must be spoken together as a phrase, without pausing between words, in order to be recognized. Otherwise, the words are treated as text.

In principle, all three systems are *speaker independent*. This means that the system should recognize text without specific training to a user's voice. However, we found the speaker independent recognition performance insufficiently accurate for the purposes of our study. To improve recognition performance, we had all users carry out the standard speaker enrolment. During enrolment, the user reads a predetermined text to the system and the system processes the users' speech to develop a speaker-specific speech model.

3.2   Experimental Procedure

Different procedures were used for the Initial Use and the Extended Use subjects in the study. Although the Initial Use study was designed to allow for statistical comparisons between the three systems, we reported on general patterns observed across the systems as they are of more general interest to the design of successful ASR systems (Karat et al., 1999). We provide a synopsis of the design and procedure here.

3.3   Initial Use

Twenty-four native English speakers, balanced for gender and with an age range from 20–55 years, participated as paid volunteers. Participants were IBM employees from a broad spectrum of occupational backgrounds.

We assigned subjects to one of three speech recognition products: IBM, Dragon, or L&H. Each subject performed two kinds of text creation tasks using each of the input modalities. The order of modality was varied, as was the task order within each modality. All text was created with the assigned product's dictation application. Similar to Windows95 WordPad they provided basic editing functions but did not include advanced functions such as spelling or grammar checking.

Immediately before the speech tasks all participants had a training session with the experimenter. This session was standardized across the three systems to cover basic areas such as text entry and correction. Each subject dictated a body of text supplied by the experimenter, composed a brief document, learned how to correct mistakes, and was given free time to explore the functions of the system. During the training session, each subject was shown how to make corrections both during and after dictation. Subjects were also taught the different ways to make corrections, including text selection by voice, re-dictation, and use of application specific correction dialogues. Subjects were not trained for keyboard-mouse text creation tasks.

3.4   Extended Use

In contrast, the subjects in the Extended Use study were the four co-authors of this paper. Each subject used each of the three speech recognition products for ten sessions of approximately one-hour duration; for 30 sessions across the products. During seven of the ten sessions the subjects used speech recognition software to carry out actual work related correspondence. The first, sixth and tenth sessions were used for benchmark tasks. Similar to Initial Use subjects, we completed one transcription task and one email composition task. We expected our performance to improve so the tasks were considerably longer than in the Initial Use experiment to prevent ceiling effects. All sessions were videotaped. In addition, after completing at least 20 sessions, subjects completed the same set of transcription tasks used in the Initial Use study. We limit the presentation of the results of the Extended Use phase of the study to some general comparisons with the Initial Use data.

3.5   Analysis

For the Initial Use sessions, we performed a detailed analysis of the videotapes of the text creation tasks. This included coding of all of the pertinent actions carried out by subjects in the study. A taxonomy of approximately 100 codes was constructed. Over 6500 individual events were coded for 12 subjects covering speech and keyboard text creation tasks (Halverson et al., 1999). Coding included misrecognitions of commands, along with a range of usability and system problems. We paid particular attention to the interplay of text entry and correction segments during a task, as well as strategies used to make corrections. Because of the extensive time required to do this, we completed the this detailed analysis for 12 of the 24 subjects in the Initial Use phase of the study (four randomly selected subjects from each of the three systems, maintaining gender balance). Our findings here are based on detailed data from those 12 subjects. Additionally, we report selected data from the four subjects in the Extended Use phase. Data from each of the three speech recognition systems are collapsed together.

4   Results

Elsewhere we compared task performance by modality (Karat et al., 1999). For this study, we introduced a measure that allowed us to compare speech and keyboard input in terms of entry time. This measure, corrected words per minute (CWPM) is the number of words in the final document divided by the time the subject took to enter the text and make corrections, and is equivalent to typing speed reported as WPM. Speech input took significantly longer than keyboard and mouse when corrections were factored into throughput. Table 1 summarizes the data for the transcription tasks by juxtaposing speech and keyboard measures for Initial Use as well as comparisons to Extended Use performance.

Transcription	Initial Use		Extended Use
	Speech	Keyboard & mouse	Speech
CWPM	13.6	32.5	25.1
Time (min)	7.52	2.64	3.10

Table 1: Mean corrected words per minute and time per task by entry modality and task type ($n = 12$).

After 20 sessions Extended Use subjects' mean task time is beginning to approach the task time for keyboard and mouse input. In addition, the measure of corrected words per minute is also nearing the rate associated with keyboard and mouse interaction.

We wanted to take a closer look at why. This improvement, albeit after 20 hours of experience, led us to look more closely at the effort going into error correction. We began by defining correction episodes as an effort to correct one or more words through actions that 1) identified the error, and 2) corrected it. Thus, if a subject selected one or more words using a single select action and retyped or re-dictated their correction, we scored this as a single correction episode. Each action taken during an episode counted as a step. In the previous example, the episode consists of two steps — the select action and the re-dictation.

Identifying episodes and their steps allowed us to quantify and compare the number of episodes between the modalities and between Initial and Extended Use. Table 2 summarizes the data for transcription tasks.

Transcription	Initial Use		Extended Use
	Speech	Keyboard & mouse	Speech
# Episodes	11.3	8.4	8.8
Steps/Episode	7.3	2.2	3.5

Table 2: Mean number of correction episodes and steps per task by entry modality ($n = 12$ for Initial Use, $n = 4$ for Extended Use).

Surprisingly, we found that the number of episodes was not significantly different between Initial Use and Extended Use. However, the number of steps per episode *was* significantly reduced for Extended Use subjects. Our impression was that the pattern of what happened during the correction episode was was different.

To confirm this we further segmented the data into correction 'primitives'. A primitive is defined as whatever is necessary to identify the error, get to it, and act on it. The difference from our definition of an episode is that the steps in a correction primitive are not necessarily successful, thus requiring multiple primitives per episode. In the example we gave above one episode did equal one primitive. That is, 1) the select both moved to and selected the error, and 2) the re-dictate acted on it. Because the re-dictate was successful, this was the end of the correction. However, in most cases this was not so.

4.1   Inside a Correction Episode

In general, there are three types of errors. First is when the user means to do one thing and physically does the wrong thing. This is a *direct* error. On a keyboard this is a typo — actually pressing one sequence of keys when another was intended. In speech one can mis-speak or stutter. In both cases, there is immediate

physical feedback that an error has been made. The second phenomenon that can be classified as a direct error is when you change your mind. When you first write something you may intend to say it one way; on rereading you decide to say it another way. While no real error has been made, we call this an *intent* error. For the rest of this paper, we will not distinguish this kind of error from other direct errors, precisely because we are more interested in how users recover from errors.

The third type of error is one that appears in speech but not keyboard modality. Speech induced errors are fundamentally different than keyboard errors because they are produced by the operation of the system not the user. Thus, we call this an *indirect* error. When the speech engine mis-recognizes what the user says the output is a valid word. This means that tools designed for keyboard and mouse systems, like spell checkers, will not 'catch the error', because the word is properly spelled. This also means that errors are difficult to detect during proofreading.

Even with stellar speech engine accuracy this kind of error will occur. For example, 95% accuracy means there will be 5 errors per 100 words (on average). One hundred words is roughly the length of a paragraph, and for an 8 page paper, like this one with approximately 65 paragraphs, that's 325 errors that need to be corrected. Once you begin to create additional errors during the correction process this can become truly overwhelming.

4.2   Techniques and Strategies

During tasks in the keyboard modality users predominately used a technique of deleting text (usually by backspacing over it) and retyping (73% of all corrections). The alternate technique was to select the word to be corrected and type over it (27%). Performance in the speech conditions was more varied, as it included options not available in keyboard conditions. While there are parallels to keyboard methods of error correction in ASR systems there are also important differences.

An important distinction is the many ways the user can get rid of a prior action. Systems have some combination of two or three commands that handle "undo what I just said" and "undo my last action". For example, SCRATCH THAT and UNDO. (For the remainder of the paper we will indicate a command by using all caps.) Some subtle issues arise in speech input because what the user last said may be dictated text or a command. In most cases SCRATCH THAT will cover both. In the case of just dictated text SCRATCH THAT will delete the last

Basic Stages	Keyboard/	Speech Examples		
in a correction	Mouse	2 step episode	3 step episode	5 step episode
episode	example	BEST CASE	MOST COMMON	Correction Dialog
Locate/move to	Use mouse to move and click to relocate		MOVE LEFT ONE WORD	MOVE LEFT ONE WORD
Select so can operate	Keyboard select	SELECT *Kiss*	SELECT THAT	SELECT THAT
Operate on-- i.e. Correct	Retype over	*Keep*	*Keep*	CORRECT THAT
				SELECT three
				CLICK OK

Figure 1: Actions are represented in regular type. Spoken is in bold, command are all capitals and text is in italics. This is an example of changing the phrase "Kiss the dog" to the intended "keep the dog".

word or phrase said. If the user had selected text and was correcting it, SCRATCH THAT will delete the new text and restore the previously selected text, still selected. In some cases if the user just performed a command, such as a formatting command, SCRATCH THAT will also undo the formatting. On the other hand, UNDO is the method of choice if you've just done an unintended command, for example one that deletes your entire document, and you want to recover quickly. Because the exact command varies across products, we combined these into one code: SCRUNDO, short for scratch plus undo.

Each application provides several different ways to correct errors. During training we taught users five techniques and had them practice each one. These included:

1. undoing the immediately previous action and continuing dictation (SCRUNDO);

2. selecting a word by voice and re-dictating the word (correcting by *re-dictation*);

3. selecting a word by voice, opening the error correction dialogue; and

   (a) picking from an alternates list of words (not supported by L&H);

   (b) spelling the word out loud; and

   (c) typing in the correction in the dialogue box.

We told them they could use any of these techniques at any time, as well as use keyboard and mouse to correct if desired. We did not suggest any particular technique was better than another, nor did we suggest any strategies. (By strategy, we mean both what techniques they use to correct an error and when they use them.)

There are three main patterns of *when* an error is corrected. *Inline* refers to correcting an error immediately or almost as soon as it happens. For example, typographical errors (typos) are often corrected by immediately backspacing and retyping the information. In contrast, some people do not make corrections as they go along. Rather they wait until they have all their text on the page and then take a *post-entry* or *proofreading* pass (or several) and make corrections. Finally, these strategies are not mutually exclusive so a mixed strategy, of in-line correction and then a proofreading pass, is possible.

While early speech recognition products varied in the strategies of error correction strategies that they encouraged for users, the systems in the current study all accommodate in-line correction and post-entry correction equally well. Earlier, discrete speech systems, such as IBM's VoiceType, encouraged users (in documentation and online help) to dictate first and then switch to correction mode, while Dragon Dictate encouraged users to make corrections immediately after an error was dictated. These strategies were encouraged to accommodate system designs and limitations, and not because of a user driven reason.

4.3 What We Expected and What We Saw

Ideally, correction should be a straightforward process. What we expect to see is a pattern of detecting the error, locating and selecting the word to be corrected, and then making the correction. Given the variety of correction methods in speech this could be at best a 2-step process (select, re-dictate). This is possible because the command SELECT <word> locates the word and highlights it. (All three products have an algorithm that cycles through selections if there is more than one in the document, although each product makes different choices about which direction to search from the current cursor position.) Using the

correction dialogue, it could be a 4-step process, or at most a 5-step process (move, select, open correction dialogue, choose from list, close dialogue). (Figure 1 summarizes some common patterns.)

While using the correction dialogue may take more steps, it can be the least error prone because the word does not need to be re-dictated. With the correct information in the alternates list the user can select the word, open the correction dialogue, and pick the word — replacing the selected word and closing the dialogue — in just three steps. Because of this efficiency we might expect to see the correction dialogue used frequently. Instead, we see it invoked in only 8% of all correction primitives.

Instead, the most common correction technique is to re-dictate the incorrect word. Initial Use subjects chose this technique 40% of the time. This makes sense when you consider two things. First, it is a two step process: Select the word and re-dictate. Second, if there is a problem the user can just undo the last dictation and try again. This is a lot like typing where users have gotten used to typing, immediately backspacing and retyping. However, there is a significant problem with this technique. One problem is that the accuracy for re-dictation is considerably lower than for initial dictation (47% averaged across all systems, vs. reported accuracy of 95% or better).

There is another problem with correction: when speech is used, there is still a chance of error — not only with re-dictation, but also with mis-recognition of commands. In fact, 22% of the correction primitives were about undoing the effect of incorrect commands. This is the tip of the iceberg however. When this occurs we often see several errors, expanding a correction episode to as many as twenty-five steps with five or more correction primitives. In short, we see a *cascade* of errors.

While it might have been faster, none of the subjects analysed switched to an all keyboard strategy for correction. (Only one subject reliably integrated keyboard and mouse with speech during correction.)

In fact, subjects tended to stay in the speech modality much longer during a correction than we expected or was efficient. In later comments they reported that they knew there must be a better way to do it, and figured that integrating speech with keyboard would be more efficient, but they had no idea how to go about it. They may have been biased to correcting with speech because they knew the experiment was to evaluate speech systems. Nonetheless, their tenacity was despite experimenters' instructions to use keyboard and mouse whenever they wished.

In sum, we see two distinct patterns. First, Initial Use subjects' fixated on re-dictation, in spite of reduced recognition accuracy. This pattern is what Oviatt & VanGent (1996) refer to as *spiral depths*: the number of times a subject continues to re-dictate the same word, despite incorrect recognition. Second, across all systems commands were sometimes mis-recognized as either dictation or other commands. During error correction, these mis-recognitions caused a new error that must be corrected before the original error can be dealt with. The result is a *cascade* of errors where apparent user frustration increased with the depth.

These two patterns — cascades and spirals — and how users handle them, are behind the difference in the number of steps in a correction episode of Initial Use and Extended Use.

4.4   Spiral Depths of Re-dictation

Oviatt & van Gent designed their experiment to understand the potential benefits of multi-modal input for error resolution. To this end, they simulated the equivalent of serial mis-recognitions to a spiral depth of 6. They found, as we did, that "on the first repetition following an error, an initial *no-switch strategy*, is evident in which users tend to repeat the same lexical content within the same mode" (p.207). On subsequent repetitions they found an increasing likelihood to switch either modality or lexical expression or both.

In our study, a little over 50% of the time we saw subjects continue to re-dictate to a spiral depth of 3. One quarter of the time they continued to a depth of 4. Slightly less than one quarter of the time they gave up after level two. We rarely saw changing to a synonym in the composition tasks, which is the only place they could have changed the lexical expression.

Unlike Oviatt & van Gent we did not see an increasing likelihood to switch modalities with the depth. Correcting with the keyboard, was equally as likely as successfully re-dictating the correction or a switch to the correction dialogue (which usually maintains modality). Almost as likely was giving up on the error completely. While a switch in modality did result in correcting the error, opening the correction dialogue usually meant more frustration.

Unfortunately, the correction dialogues appear to be designed with the assumption that they will be invoked first not last. Thus, by the time the user invokes a correction dialogue *the speech engine data has been lost or discarded and the alternates list is empty!* One subject explicitly reselected the word each time rather than undoing the previous action. Her

reward was finding the alternates list still full of words. However, the majority of subjects used one of the SCRUNDO commands, and as a result find it empty. (One of the products does not provide an alternates list in the correction dialogue, so subjects invoked the dialogue only in order to spell.)

4.5   Cascades of Errors

Five times more often than isolated spirals we observed cascades. One quarter of these contained embedded spirals, within which other errors regularly intervened. In these spirals, the picture was different than with the isolated spirals. Just under half of these ended in correction dialogues, with a quarter ending in successful re-dictations and another 20% were corrected on the keyboard. (Less than 1% gave up without correcting.) Correcting phrases sometimes meant two embedded spirals in a cascade.

What predominantly characterizes cascades is what we might think of as "commands gone wrong". These may be because the speech engine mis-recognized a command — either as another command or as dictation — or because the subject used the wrong command or said a command in the wrong way. In each of these cases an additional error may be created that must also be corrected. Error cascades seem to have 'depth' similar to spirals, but are in fact different. To see this we need to look at a specific error.

Take the following case. The subject is trying to correct the composition he has just dictated. (We have diagrammed this in Figure 2. The top line is a running commentary of the 'plot' of the correction. In the diagram, moving across the page indicates length while moving down the page indicates levels.)

He begins by moving into position. The first command is successful, but the second command is mis-recognized and inserted in the text as dictation. His immediate response is an almost involuntary *back*, quickly cut off. This mis-speak is also inserted in the text and creates a second error. Then, trying to correct that error, the command is again mis-recognized and inserted in the text as dictation. He now has three levels separating him from the actual correction.

To reset the level he must correct each of these errors. Because these products all have multiple levels of undo, he uses the "undo what I just said" command (symbolized here as SCRUNDO) repeatedly to undo the effects of each problem.

Now he again tries to move ahead in the document, and again his command is mis-recognized as dictation. Again using undo, he is finally successful at the move and dictates the word he wants to insert.

This example shows the difficulty subjects get into because of command mis-recognitions. While this example is many levels deep, most cascades are only one level down but many more long. Figure 2 is only 11 steps long, while twice that number was not uncommon.

What we also see is evidence of subjects' lack of knowledge about the commands and lack of facility using them. Command mis-recognitions are dramatic because they have unexpected effects, while command misuse often has no discernible effect at all. Nonetheless, the effort, and steps required for the subject to recover from these errors is just as significant.

5   Discussion

While we only present a detailed example from one subject, these patterns are pervasive across the 12 subjects analysed. Initial Use subjects have a lot to contend with — spirals, cascades, and cascades with embedded spirals. They create these with help from the design of the speech products — and their frustration shows.

This is in contrast to the observed behaviour of the Extended Use subjects. While we have not completed an in-depth video analysis like that for Initial Use we do have diagnostic evidence of very different behaviour. After 20 sessions, all four Extended Use subjects have learned two things. They cut off re-dictation spirals at level 2 and they cut cascades very quickly. They do both of these by switching to different techniques more rapidly.

In the case of mis-recognized commands, the Extended Use subjects switch modalities, using their facility with the keyboard and mouse to manipulate text, make selections, and close dialogue boxes. These two strategies appear to account for the significant decrease in the number of steps in a correction episode between Initial and Extended Use subjects.

In sum, the increasing speed and facility of more experienced users appears related to the patterns of error correction experienced by first time users. While designers of ASR systems have developed correction aids, like the correction dialogue with its alternates list, users' patterns of use often make it unusable. Instead, novice users predominantly use one method of error correction — staying with the speech modality and re-dictating the same word three or four times before switching techniques. This arises because of reduced recognition accuracy with re-dictation. The result is a spiral. In addition, we saw a pattern of cascading errors due to problems with using commands. The primary culprit was mis-recognized
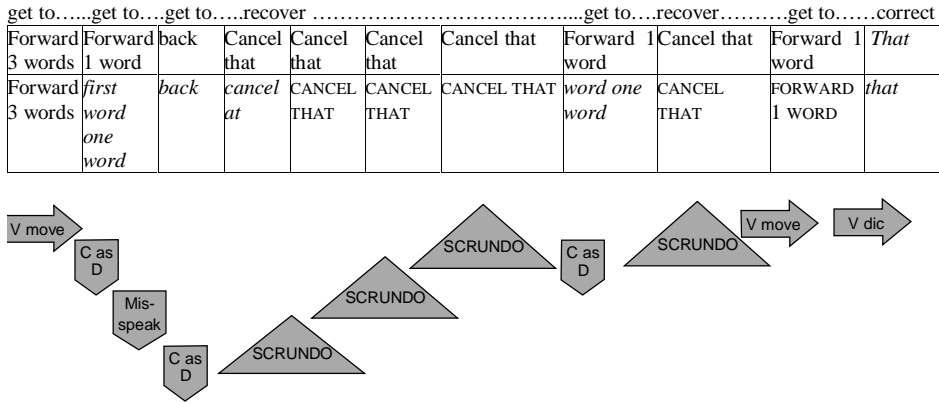
get to…...get to….get to…...recover …………………………........get to….recover……….get to……correct

Forward 3 words	Forward 1 word	back	Cancel that	Cancel that	Cancel that	Cancel that	Forward 1 word	Cancel that	Forward 1 word	*That*
Forward 3 words	*first word one word*	*back*	*cancel at*	CANCEL THAT	CANCEL THAT	CANCEL THAT	*word one word*	CANCEL THAT	FORWARD 1 WORD	*that*

Figure 2: Diagram of cascade, showing levels, and length. 'V' is the abbreviation for voice, while 'C as D' represents command mis-recognized as dictation. The first line is what the subject is trying to do, the second line is what he says, while the third line is what is recognized. Recognized commands are in caps. Text inserted into the document is in italic.

commands, although not knowing the right command, or being able to say it properly was also a factor.

Based on the observed errors there are some design recommendations we can suggest. First, increase recognition accuracy for re-dictation. Second, recognize novice users' tendency to use the SCRATCH THAT (or comparable) command. In current ASR systems this appears to discard information saved from the speech engine that is necessary to populate the alternates list in the correction dialogue. Third, recognize that novices' tendency to fixate on one technique may mean they stick with a strategy past its optimum. We saw this despite explicit training on a variety of techniques. This tendency contributes to the spiral and cascade error patterns that mark Initial Use error correction.

Learning to do things differently takes a significant amount of time. The snapshot we have of the Extended Use subjects after 20 hours shows that these lessons are being learned. We will be looking in more depth at Extend Use sessions to see exactly how long it takes to learn to switch strategies quickly.

Acknowledgements

References

Danis, C. & Karat, J. (1995), Technology-Driven-Design of Speech Recognition Systems, *in* G. Olson & S. Schuon (eds.), *Proceedings of the Symposium on Designing Interactive Systems: Processes, Practices, Methods and Techniques (DIS'95)*, ACM Press, pp.17–24.

Halverson, C., Horn, D., Karat, C., & Karat, J. (1999), Understanding What Speech Recognition Can't: A Language to Understand Your Users Behaviour, *in Proceedings of AVIOS'99: Speech Developers Conference*, American Voice Input/Output Society, pp.267–76.

Karat, C., Halverson, C., Horn, D. & Karat, J. (1999), Patterns of Entry and Correction in Large Vocabulary Continuous Speech Recognition Systems, *in* M. G. Williams, M. W. Altom, K. Ehrlich & W. Newman (eds.), *Proceedings of CHI'99: Human Factors in Computing Systems*, ACM Press, pp.568–75.

Karat, J. (1995), Scenario Use in the Design of a Speech Recognition System, *in* J. M. Carroll (ed.), *Scenario-Based Design: Envisioning Work and Technology in System Development*, John Wiley & Sons, pp.109–34.

Lai, J. & Vergo, J. (1997), MedSpeak: Report Creation with Continuous Speech Recognition, *in* S. Pemberton (ed.), *Proceedings of CHI'97: Human Factors in Computing Systems*, ACM Press, pp.431–38.

Oviatt, S. & VanGent, R. (1996), Error Resolution During Multimodal Human–Computer Interaction, *in* T. Bunnell & W. Idsardi (eds.), *Proceedings of the Fourth International Conference on Spoken Language Processing*, University of Delaware and AI DuPont Institute, New York, pp.204–7.

Rhyne, J. & Wolf, C. (1993), Recognition-Based User Interfaces, *in* H. R. Hartson & D. Hix (eds.), *Advances in Human–Computer Interaction*, Vol. 4, Ablex, pp.191–250.

Author Index

Keyword Index