Voice Typing: A New Speech Interaction Model for Dictation on Touchscreen Devices

Anuj Kumar[†‡], Tim Paek[†], Bongshin Lee[†]

[†]Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
{timpaek, bongshin}@microsoft.com

[‡]Human-Computer Interaction Institute,
Carnegie Mellon University
Pittsburgh, PA 15213, USA
anujk1@cs.cmu.edu

ABSTRACT

Dictation using speech recognition could potentially serve as an efficient input method for touchscreen devices. However, dictation systems today follow a mentally disruptive speech interaction model: users must first formulate utterances and then produce them, as they would with a voice recorder. Because utterances do not get transcribed until users have finished speaking, the entire output appears and users must break their train of thought to verify and correct it. In this paper, we introduce Voice Typing, a new speech interaction model where users' utterances are transcribed as they produce them to enable real-time error identification. For fast correction, users leverage a marking menu using touch gestures. Voice Typing aspires to create an experience akin to having a secretary type for you, while you monitor and correct the text. In a user study where participants composed emails using both Voice Typing and traditional dictation, they not only reported lower cognitive demand for Voice Typing but also exhibited 29% relative reduction of user corrections. Overall, they also preferred Voice Typing.

Author Keywords

Speech recognition; dictation; multimodal; error correction; speech user interfaces

ACM Classification Keywords

H.5.2 [Information Interfaces and Presentation]: User Interfaces – Voice I/O;

General Terms

Design; Human Factors

INTRODUCTION

Touchscreen devices such as smartphones, slates, and tabletops often utilize soft keyboards for input. However, typing can be challenging due to lack of haptic feedback [12] and other ergonomic issues such as the "fat finger problem" [10]. Automatic dictation using speech recognition could serve as a natural and efficient mode of input, offering several *potential* advantages. First, speech throughput is reported to be at least three times faster than typing on a hardware QWERTY keyboard [3]. Second, compared to other text input methods, such as handwriting or typing, speech has the greatest flexibility in terms of screen size. Finally, as touchscreen devices proliferate throughout the world, speech input is (thus far) widely considered the only plausible modality for the 800 million or so non-literate population [26].

However, realizing the potential of dictation critically depends on having reasonable speech recognition performance and an intuitive user interface for quickly correcting errors. With respect to performance, if users are required to edit one out of every three words, which is roughly the purported Word Error Rate (WER) of speaker-independent (i.e., not adapted), spontaneous conversation, no matter how facile the editing experience may be, users will quickly abandon speech for other modalities. Fortunately, with personalization techniques such as MLLR and MAP acoustic adaptation [8,15] as well as language model adaptation [5], WER can be reduced to levels lower than 10%, which is at least usable. Note that all commercially released dictation products recommend and perform acoustic adaptation, sometimes even without the user knowing (e.g., dictation on Microsoft Windows 7 OS).

With respect to quickly correcting errors, the editing experience on most dictation systems leaves much to be desired. The speech interaction model follows a *voice recorder* metaphor where users must first formulate what they want to say in utterances, and then produce them, as they would with a voice recorder. These utterances do not get transcribed until users have finished speaking (as indicated by a pause or via push-to-talk), at which point the entire output appears at once after a few seconds of delay. This is so that the recognizer can have as much context as possible to improve decoding. In other words, real-time presentation of output is sacrificed for accuracy. Users must then break their train of thought, verify the output verbatim and correct errors. This process can be mentally disruptive, time-consuming, and frustrating. Indeed, users typically

CHI'12, May 5–10, 2012, Austin, Texas, USA.

spend only 25-30% of their time actually dictating. The rest of the time is spent on identifying and editing transcription errors [13,18].

In this paper, we introduce Voice Typing, a new speech interaction model where users' utterances are transcribed as they produce them to enable real-time error identification. For fast correction, users leverage a gesture-based marking menu that provides multiple ways of editing text. Voice Typing allows users to influence the decoding by facilitating immediate correction of the real-time output. The metaphor for Voice Typing is that of a secretary typing for you as you monitor and quickly edit the text using the touchscreen.

The focus of this paper is on improving speech interaction models when speech serves as the primary input modality. We present two contributions. First, we elaborate on Voice Typing, describing how it works and what motivated the design of its interaction model. Second, we describe the results of a user study evaluating the efficacy of Voice Typing in comparison to traditional dictation on an email composition task. We report both quantitative and qualitative measures, and discuss what changes can be made to Voice Typing to further enhance the user experience and improve performance. Our results show that by preventing decoding errors from propagating through immediate user feedback, Voice Typing can achieve a lower user correction error rate than traditional dictation.

VOICE TYPING

As mentioned previously, the speech interaction model of Voice Typing follows the metaphor of a secretary typing for you while you monitor and correct the text. Real-time monitoring is important because it modulates the speed at which users produce utterances. Just in the way that you would not continue speaking if the secretary was lagging behind on your utterances, users of Voice Typing naturally adjust their speaking rate to reflect the speed and accuracy of the recognizer. Indeed, in an exploratory design study we conducted where participants dictated through a "noisy microphone" to a confederate (i.e., an experimenter), we found that as the confederate reduced typing speed (to intentionally imply uncertainty about what was heard), participants also slowed their speaking rate. For the prototype we developed, the transcription speed was such that users produced utterances in chunks of 2-4 words (see the Prototype section for more details). In other words, for real-time feedback, in Voice Typing, users are not restricted to speaking one word-at-a-time with a brief pause in between, as in discrete recognition [24], nor required to wait for long after speaking full utterances, like in traditional dictation. Instead, users can speak in small chunks that match their thought process. Ideally, if the recognizer could transcribe as quickly as users could produce speech with perfect accuracy, the Voice Typing experience would be more like dictating to a professional stenographer. However, with current state-of-the-art recognition where corrections are required due to misrecognitions, Voice Typing is more akin to dictating to a secretary or a fast-typing friend.

We now elucidate the motivation for Voice Typing and the technical challenges required to realize its full potential. We also describe the prototype we implemented for touchscreen devices.

Motivation

Voice Typing is motivated by both cognitive and technical considerations. From a cognitive standpoint, human-computer interaction researchers have long known that providing real-time feedback for user actions not only facilitates learning of the user interface but also leads to greater satisfaction [21]. With respect to natural language, psycholinguists have noted that as speakers in a conversation communicate, listeners frequently provide real-time feedback of understanding in the form of back-channels, such as head nods and "uh-huh" [6]. Indeed, research suggests that language processing is incremental i.e. it usually proceeds one word at a time, and not one utterance or sentence at a time [2,31,32], as evidenced by eye movements during comprehension. Real-time feedback for text generation is also consistent with the way most users type on a keyboard. Once users become accustomed with the keyboard layout, they typically monitor their words and correct mistakes in real-time. In this way, the interaction model for Voice Typing is already quite familiar to users.

From a technical standpoint, Voice Typing is motivated by the observation that dictation errors frequently stem from incorrect segmentations (though to date we are unaware of any published breakdown of errors). Consider the classic example of speech recognition failure: "It's hard to wreck a nice beach" for the utterance "It's hard to recognize speech." In this example, the recognizer has incorrectly segmented "recognize" for "wreck a nice" due to attaching the phoneme /s/ to "nice" instead of "speech." Because having to monitor and correct text while speaking generally induces people to speak in small chunks of words, users are more likely to pause where segmentations should occur. In other words, in the example above, users are more likely to utter "It's hard <pause> to recognize <pause> speech," which provides the recognizer with useful segmentation information. In the Experiment section, we assess whether Voice Typing actually results in fewer corrections.

Voice Typing has yet another potential technical advantage. Since there is an assumption that users are monitoring and correcting mistakes as they go along, it is possible to treat previously reviewed text as both language model context for subsequent recognitions and supervised training data for acoustic [8] and language model adaptation [5]. With respect to the former, real-time correction prevents errors from propagating to subsequent recognitions. With respect to the latter, Voice Typing enables online adaptation with

acoustic and language data that has been manually labeled by the user. There is no better training data for personalization than that supervised by the end user.

Prototype

While the technical advantages of Voice Typing are appealing, implementing a large vocabulary continuous speech recognition (LVCSR) system that is designed from scratch for Voice Typing is no small feat and will likely take years to fully realize (see the Related Work section). At a high level, decoding for LVCSR systems typically proceeds as follows (see [22] for a review). As soon as the recognizer detects human speech it processes the incoming audio into acoustic signal features which are then mapped to likely sound units (e.g., phonemes). These sound units are further mapped to likely words and the recognizer connects these words together into a large lattice or graph. Finally, when the recognizer detects that the utterance has ended, it finds the optimal path through the lattice using a dynamic programming algorithm or Viterbi [23]. The optimal path yields the most likely sequence of words (i.e., the recognition result). In short, current LVCSR systems do not return a recognition result until the utterance has finished. If users are encouraged to produce utterances that constitute full sentences, they will have to wait until the recognizer has detected the end of an utterance before receiving the transcribed text all at once. This is of course the speech interaction model for traditional dictation.

In order to support the speech interaction model of Voice Typing, the recognizer would have to return the optimal path through the lattice created thus so far. This can be done through *recognition hypotheses*, which most speech APIs expose. Unfortunately, for reasons that go beyond the scope of this paper, recognition hypotheses tend to be of poor quality. Indeed, in building a prototype, we explored leveraging recognition hypotheses but abandoned the idea due to low accuracy. Instead, we decided to use LVCSR decoding as is, but with one modification. Part of the way in which the recognizer detects the end of an utterance is by looking for silence of a particular length. Typically, this is defaulted to 1-2 seconds. We changed this parameter to 0 milliseconds. The effect was that whenever users paused for just a second, the recognizer would immediately return a recognition result. Note that the second of delay is due to other processing the recognizer performs.

To further facilitate the experience of real-time transcription, we coupled this modification with two interaction design choices. First, instead of displaying the recognition result all at once, we decided to display each word one by one, left to right, as if a secretary had just typed the text. Second, knowing the speed at which the recognizer could return results and keep up with user utterances, we trained users to speak in chunks of 2-4 words.

Providing real-time transcriptions so that users can monitor and identify errors is only the first aspect of Voice Typing. The second is correcting the errors in a fast and efficient manner on touchscreen devices. To achieve this goal, we leveraged a marking menu that provides multiple ways of editing text. Marking menus allow users to specify a menu choice in two ways, either by invoking a radial menu, or by making a straight mark in the direction of the desired menu item [14]. In Voice Typing, users invoke the marking menu by touching the word they desire to edit. Once they learn what choices are available on the marking menu, users can simply gesture in the direction of the desired choice. In this way, marking menus enables both the selection and editing of the desired word, and provides a path for novice users to become expert users. Figure 1(a) displays the marking menu we developed for Voice Typing. If users pick the bottom option, as shown in Figure 1(b) they receive a list of alternate word candidates for the selected word, which is often called an *n-best list* in the speech community. The list also contains an option for the selected word with the first letter capitalized. If they pick the left option, they can delete the word. If they pick the top option, as shown in Figure

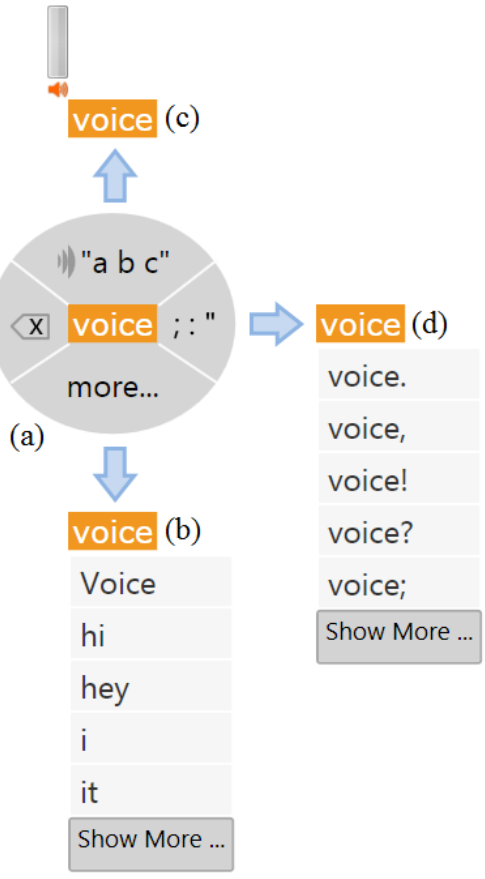


Figure 1. Screenshots of (a) the Voice Typing marking menu, (b) list of alternate candidates for a selected word, including the word with capitalized first letter, (c) re-speak mode with volume indicator, and (d) list of punctuation choices.

1(c) they can re-speak the word or spell it letter by letter. Note that with this option they can also speak multiple words. Finally, if they pick the right option, as shown in Figure 1(d) they can add punctuation to the selected word. We decided to include this option because many users find it cumbersome and unnatural to speak punctuation words like "comma" and "period." Having a separate punctuation option frees users from having to think about formatting while they are gathering their thoughts into utterances.

It is important to note that Voice Typing could easily leverage the mouse or keyboard for correction, not just gestures on a touchscreen. For this paper, however, we decided to focus on marking menus for touchscreen devices for two reasons. First, a growing number of applications on touchscreen devices now offer dictation (e.g., Apple's Siri which uses Nuance Dragon [20], Android Speech-to-Text [28], Windows Phone SMS dictation [19], Vlingo Virtual Assistant [33], etc.). Second, touchscreen devices provide a unique opportunity to utilize touch-based gestures for immediate user feedback, which is critical for the speech interaction model of Voice Typing. In the user study below, our comparison of marking menus to regular menus reflects the correction method employed by almost all of these new dictation applications.

RELATED WORK

A wide variety of input methods have been developed to expedite text entry on touchscreen devices. Some of these methods are similar to speech recognition in that they utilize a noisy channel framework for decoding the original input signal. Besides the obvious example of handwriting recognition and prediction of complex script for languages such as Chinese, a soft keyboard can dynamically adjust the target regions of its keys based on decoding the intended touch point [10]. The language model utilized by speech recognition to estimate the likelihood of a word given its previous words appears in almost all predictive text entry methods, from T9 [9] to shape writing techniques [34] such as SWYPE [30].

Beyond touchscreen input methods that are similar to speech recognition, a few researchers have explored how to obtain more accurate recognition hypotheses from the word lattice so that they can be presented in real-time. Fink *et al.* [7] found that providing more right context (i.e., more acoustic information) could improve accuracy. Likewise, Baumann *et al.* [4] showed that increasing the language model weight of words in the lattice could improve accuracy. Selfridge *et al.* [25] took both of these ideas further and proposed an algorithm that looked for paths in the lattice that either terminated in an end-of-sentence (as deemed by the language model), or converged to a single node. This improved the stability of hypotheses by 33% and increased accuracy by 21%. Note that we have not yet tried to incorporate any of these findings, but consider this part of our future work.

With respect to the user experience of obtaining real-time recognition results, Aist *et al.* [1] presented users with pre-recorded messages and recognition results that appeared either all at once or in an incremental fashion. Users overwhelmingly preferred the latter. Skantze and Schlangen [27] conducted a similar study where users recited a list of numbers. Again, users preferred to review the numbers in an incremental fashion. All of this prior research justifies the Voice Typing speech interaction model. To our knowledge, the user study we describe in the next section represents the first attempt to compare incremental, real-time transcription with traditional dictation on a spontaneous language generation task using LVCSR decoding.

The Voice Typing gesture-based marking menu is related to research in multimodal correction of speech recognition errors. In Martin *et al.* [17], preliminary recognition results were stored temporarily in a buffer which users could interactively edit by spoken dialogue or by mouse. Users could delete single words or the whole buffer, re-speak the utterance, or select words from an n-best list. Suhm *et al.* [29] proposed switching to pen-based interaction for certain types of corrections. Besides advocating spelling in lieu of re-speaking, they created a set of pen gestures such as crossing-out words to delete them. Finally, commercially available dictation products for touchscreen devices, such as the iPhone Dragon Dictation application [20], also support simple touch-based editing. To date, none of these products utilize a marking menu.

USER STUDY

In order to assess the correction efficacy and usability of Voice Typing in comparison to traditional dictation, we conducted a controlled experiment in which participants engaged in an email composition task. For the email content, participants were provided with a structure they could fill out themselves. For example, "Write an email to your friend Michelle recommending a restaurant you like. Suggest a plate she should order and why she will like it." Because dictation entails spontaneous language generation, we chose this task to reflect how end users might actually use Voice Typing.

Experimental Design

We conducted a 2x2 within-subjects factorial design experiment with two independent variables: *Speech Interaction Model* (Dictation vs. Voice Typing) and *Error Correction Method* (Marking Menu vs. Regular). In Regular Error Correction, all of the Marking Menu options were made available to participants as follows. If users tapped a word, the interface would display an n-best list of word alternates. If they performed press-and-hold on the word, that invoked the re-speak or spelling option. For deleting words, we provided "Backspace" and "Delete" buttons at the bottom of the text area. Placing the cursor between words, users could delete the word to the left using

"Backspace" and the word to the right using "Delete." Users could also insert text anywhere the cursor was located by performing press-and-hold on an empty area.

The order of presentation of *Speech Interaction Model* and *Error Correction Method* was counter-balanced. We collected both quantitative and qualitative measures. With respect to quantitative measures, we measured rate of correction and the types of corrections made. With respect to the qualitative measures, we utilized the *NASA task load index (NASA-TLX)* [11] because it is widely used to estimate perceived workload assessment. It is divided into six different questions: mental demand, physical demand, temporal demand, performance, effort, and frustration. For our experiment, we used the software version of NASA-TLX, which contains 20 divisions, each division corresponding to 5 task load points. Responses were measured on a continuous 100-point scale. We also collected qualitative judgments via a post-experiment questionnaire that asked participants to rank order each of the four experimental conditions (Dictation Marking Menu, Voice Typing Marking Menu, Dictation Regular and Voice Typing Regular) in terms of preference. The rank order questions were similar to NASA-TLX so that we could accurately capture all the dimensions of the workload assessment. Finally, we collected open-ended comments to better understand participants' preference judgments.

Software and Hardware

We developed the Voice Typing and Dictation *Speech Interaction Models* using the Windows 7 LVCSR dictation engine. As mentioned before, for Voice Typing, we modified the silence parameter for end segmentation via the Microsoft System.Speech managed API. In order to control speech accuracy across the four experimental conditions, we turned off the (default) MLLR acoustic adaptation. Both types of *Error Correction Methods* were implemented using the Windows 7 Touch API and Windows Presentation Foundation (WPF). We conducted the experiment on a HP EliteBook 2740p Multi-Touch Tablet with dual core 2.67 GHz i7 processor and 4 GB of RAM.

Participants

We recruited 24 participants (12 males and 12 females), all of whom were native English speakers. Participants came from a wide variety of occupational backgrounds (e.g., finance, car mechanics, student, housewife, etc.). None of the participants used dictation via speech recognition on a regular basis. The age of the participants ranged from 20 to 50 years old (M = 35.13) with roughly equal numbers of participants in each decade.

Procedure

In total, each experimental session lasted 2 hours, which included training the LVCSR recognizer, composing two practice and three experimental emails per experimental condition, and filling out NASA-TLX and post-experiment questionnaires. To train the LVCSR recognizer, at the start of the each session, participants enrolled in the Windows 7 Speech Recognition Training Wizard, which performs MLLR acoustic adaptation [8] on 20 sentences, about 10 minutes of speaking time. We did this because we found that without training, recognition results were so inaccurate that users became frustrated regardless *Speech Interaction Model* and *Error Correction Method*.

During the training phase for each of the four experimental conditions, the experimenter walked through the interaction and error correction style using two practice emails. In the first practice email, the experimenter demonstrated how the different *Speech Interaction Models* worked, and then performed the various editing options available for the appropriate *Error Correction Method* (i.e., re-speak, spelling, alternates, delete, insert, etc.). Using these options, if the participant was unable to correct an error even after three retries, they were asked to mark it as incorrect. Once participants felt comfortable with the user interface, they practiced composing a second email on their own with the experimenter's supervision. Thereafter, the training phase was over and users composed 3 more emails. At the end of each experimental condition, participants filled out the NASA-TLX questionnaire. At the end of the experiment, they filled out the rank order questionnaire and wrote open-ended comments.

RESULTS

Quantitative

In order to compare the accuracy of Voice Typing to Dictation, we computed a metric called *User Correction Error Rate* (UCER), modeled after Word Error Rate (WER), a widely used metric in the speech research community. In WER, the recognized word sequence is compared to the actual spoken word sequence using Levenshtein's distance [16], which computes the minimal number of string edit operations–substitution (S), insertion (I), and deletion (D)–necessary to convert one string to another. Thereafter, WER is computed as: WER = $(S + I + D) / N$, where N is the total number of words in the true, spoken word sequence.

In our case, measuring WER was not possible for two reasons. First, we did not have the true transcript of the word sequence – that is, we did not know what the user had actually intended to compose. Second, users often improvised after seeing the output and adjusted their utterance formulation, presumably because the recognized text still captured their intent. Moreover, we believe that although WER accurately captures the percentage of mistakes that the recognizer has made, it does not tell us much about the amount of effort that users expended to correct the recognition output, at least to a point where the text was acceptable. The latter, we believe, is an important metric for acceptance of any dictation user interface. Thus, we computed UCER as:

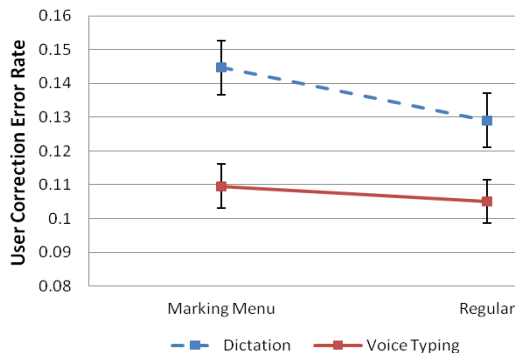


Figure 2. User correction error rate for all four conditions. Blue data points are for traditional Dictation, red data points for Voice Typing.

$$UCER = \frac{Sub + Ins + Del + Uncorrected}{Num}$$

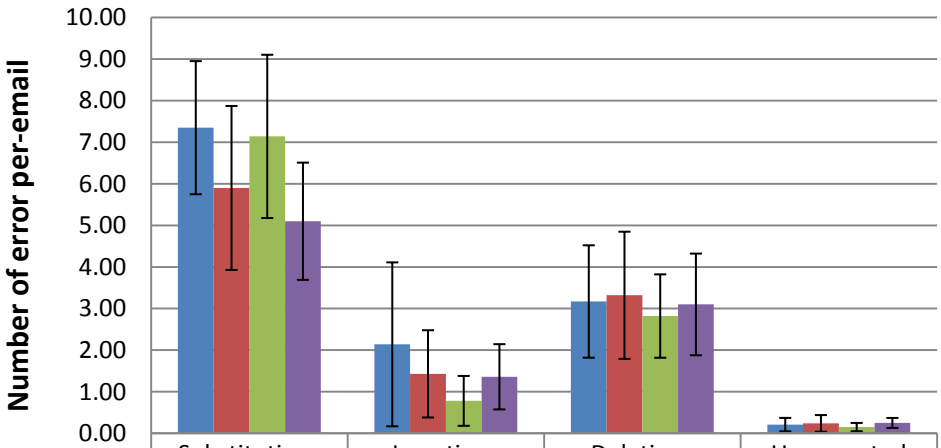
where *Sub* is the number of substitutions the user made using the 'Respeak' mode (both spelling and re-speaking the entire word) or via using the alternates, *Ins* is the number of word insertions, *Del* is the number of word deletions, *Uncorrected* is the number of words that user identified as incorrect but did not correct due to difficulties in error correction (see the Procedure Section), and *Num* is the number of words in the final text that was submitted by the user.

While the UCER metric captures the amount of effort users expended to correct mistakes, it does not include the errors that were left "unidentified and uncorrected" by the user. For example, there were occasions when words were recognized as plural, instead of singular, but were left unchanged. This may be because they were unidentified or because the user did not feel that the grammatically incorrect text affected the intended meaning. In any case, these cases were rare and more importantly, they were equally distributed across the experimental conditions.

In terms of UCER, a repeated measures ANOVA yielded a significant main effect for the *Speech Interaction Model* ($F(1,46) = 4.15$, $p < 0.05$), where Voice Typing (M = 0.10, SD = 0.01) was significantly lower than Dictation (M = 0.14, SD = 0.01). Looking at Figure 2, it may seem as if Marking Menu had slightly higher UCER than Regular, but we did not find any main effect for *Error Correction Method*, nor did we find an interaction effect.

In terms of types of corrections, as described previously, UCER consists of four different types of corrections. We wanted to further tease apart the types of corrections that were significantly different across experimental conditions to understand what led to lower UCER in Voice Typing than Dictation. Figure 3 plots average errors per-email for all four conditions. For substitutions, We obtained a significant main effect for *Error Correction Method* ($F(1,46) = 5.9$, $p < 0.05$), where Marking Menu had significantly higher substitutions (M = 7.24, SD = 0.5) than Regular (M = 5.50, SD = 0.5). For insertions, deletions, and identified but uncorrected errors, we did not find any significant effects.

To contrast the amount of time users had to wait to see the



	Substitutions	Insertions	Deletions	Uncorrected
Dictation, Marking Menu	7.35	2.14	3.17	0.21
Dictation, Regular	5.90	1.43	3.32	0.24
Voice Typing, Marking Menu	7.14	0.78	2.82	0.15
Voice Typing, Regular	5.10	1.36	3.10	0.25

Figure 3. Average number of substitutions, insertions, deletions made by the user in order to correct an email for each of the four experimental conditions, and the number of words left uncorrected.

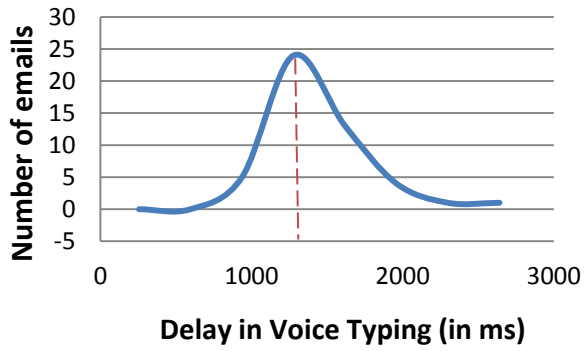


Figure 4. Frequency distribution of the system response times across emails in Voice Typing condition. For most emails the delays were within one standard deviation (0.3 seconds) of the average (1.27 seconds), and all the emails were with two standard deviations from the average.

recognition result, although this was not a dependent variable, we measured the delay from the time the recognizer detected speech (Time (speech)) to when the actual transcription (Time (text)) was shown in the user interface:

$$Delay = \frac{\sum_{all\ emails} Time\ (text) - Time\ (speech)}{Total\ Number\ of\ Emails}$$

As shown in Figure 4, the average waiting time for Voice Typing was 1.27 seconds, which of course was significantly lower than that of Dictation, 12.41 seconds. Although the delay in dictation seems ten times large, we should note that this includes the time that the user took to speak the entire utterance, as well as the delay time.

For Voice Typing, we were interested to see if the average delay varied across emails either due to the users' speaking style or other acoustic differences that might have led to a difference in user experience. Surprisingly, all the delays were within two standard deviations from the average, with 37 (out of 48 emails for Voice Typing) having delays within one standard deviation from the average i.e., between 1.27 ± 0.3 seconds, as shown in Figure 4.

Qualitative

Voice Typing vs. Dictation

A repeated measure ANOVA on the NASA TLX data for mental demand yielded a significant main effect for *Speech Interaction Model* ($F(1, 46) = 4.47$, $p = 0.03$), where Voice Typing (M = 30.90, SD = 19.16) had lower mental demand than Dictation (M = 39.48, SD = 20.85). Furthermore, we found a significant main effect *Speech Interaction Model* on effort and frustration ($F(1,46) = 4.03$, $p = 0.04$ and $F(1,46) = 4.02$, $p = .05$, respectively). In both cases, Voice Typing displayed significantly lower effort and frustration than Dictation.

On the rank order questionnaire, overall 18 out of 24 participants indicated a preference for Voice Typing over Dictation (χ^2 (1) = 7.42, $p < 0.01$). Furthermore, 18 participants ranked Voice Typing as having less effort than Dictation (χ^2 (1) = 3.99, $p < 0.05$), and 17 ranked Voice Typing as having less frustration (χ^2 (1) = 9.53, $p < 0.05$). As indicated, all of the above rankings were statistically significant by Wilcoxon tests.

Open-ended comments highlighted the motivation for the speech interaction model of Voice Typing. As one participant put it, *"It [Voice Typing] was better because you did not have to worry about finding mistakes later on. You could see the interaction [output] as you say; thereby reassuring you that it was working fine."* On the other hand, not all participants agreed. One participant explained: *"I preferred Dictation, because in Voice Typing, if one word was off as I was speaking, it would distract me."*

Marking Menu vs. Regular

On the NASA-TLX data, we found a significant main effect for *Error Correction Method* on physical demand ($F(1,46) = 4.43$, $p = 0.03$), where Marking Menu (M = 19.50, SD = 20.56) had significant lower physical demand than Regular (M = 28.13, SD = 19.44).

On the rank order questionnaires, 21 out of 24 participants overall preferred Marking Menu to Regular (χ^2 (1) = 25.86, $p < 0.01$). Furthermore, 21 participants ranked Marking Menu as having less mental demand than Regular (χ^2 (1) = 22.3, $p < 0.01$) and 21 ranked Marking Menu as having less physical demand (χ^2 (1) = 22.3, $p < 0.01$). As indicated, all of the above rankings were statistically significant by Wilcoxon tests.

With respect to open-ended comments, participants seemed to appreciate the menu discovery aspect of marking menus. As one participant put it, *"It [Marking Menu] was great for a beginner. It was easier mentally to see the circle with choices and not have to concern myself with where to select my [error correction] choices from."* Another participant highlighted how Marking Menu allowed both selection and correction at the same time: "*It [Marking Menu] seemed to involve less action.*" Again, not everyone agreed with 3 participants claiming that they found Marking Menu to be challenging to learn. In order to understand why these 3 participants felt that way, we went through their video recordings. It turned out that these participants had larger fingers than most, and had difficulties selecting and swiping (up, down, or left) words, particularly single letter words like "a."

DISCUSSION AND FUTURE WORK

In this paper, we introduced Voice Typing, a new speech interaction model where users' utterances are transcribed as they produce them to enable real-time error identification. Our experimental results indicated that Voice Typing had significantly fewer corrections than Dictation even though the acoustic and language models for the recognizer were the same for both *Speech Interaction Models*. A plausible

explanation is that when users correct transcriptions in real-time, this prevents errors from propagating. Users also felt that Voice Typing exhibited less mental demand, perceived effort, and frustration than Dictation.

With respect to *Error Correction Methods*, while there was no significant difference in the overall user correction error rate between the two methods, we found that users used substitutions (re-speak, spell, or alternates) significantly more in Marking Menu than Regular, the current style of error correction in the dictation interfaces today. One plausible explanation for this is that since users found gestures to be less mentally and physically demanding, they preferred to substitute the word rather than leave it uncorrected. This is evidenced by the trend that people left more words uncorrected in the Regular method than Marking Menu. Another plausible explanation is that the current interface required edit operations at a word level even if successive words were incorrectly recognized. This could have potentially led to more substitutions than needed. We plan to explore phrase-level edit operations as future work.

Despite the above positive findings for Voice Typing, we do not believe that the Dictation interaction model should be completely dismissed. There is merit in using a *voice recorder* style of interaction when the context demands a "hands-free, eyes-free" interaction, such as driving. Surely, in these cases, Voice Typing would not be a feasible approach. Also, in other cases such as highly-populated public spots, we imagine that typing on a soft keyboard might still be preferred for privacy.

In terms of future work, as discussed in the Motivation section, LVCSR decoding currently is not well suited for Voice Typing. While speech researchers continue to improve recognition accuracy by building better underlying algorithms and using larger datasets, our findings suggest that it may be time to rethink the speech interaction model of dictation and consider changing the decoding process to support accurate, real-time feedback from ground up. This is where we plan to focus our efforts next.

REFERENCES

1. Aist, G., Allen, J., Campana, E., Gallo, C., Stoness, S., Swift, M., and Tanenhaus, M.K. Incremental understanding in human-computer dialogue and experimental evidence for advantages over non-incremental methods. *Proc. DECALOG 2007*, (2007), 149-154.
2. Altmann, G. and Kamide, Y. Incremental interpretation at verbs: restricting the domain of subsequent reference. *Cognition 73*, 3 (1999), 247-264.
3. Basapur, S., Xu, S., Ahlenius, M., and Lee, Y.S. User expectations from dictation on mobile devices. *Proc. HCI 2007*, Springer-Verlag (2007), 217-225.
4. Baumann, T., Atterer, M., and Schlangen, D. Assessing and improving the performance of speech recognition for incremental systems. *Proc. HLT-NAACL 2009*, (2009), 380-388.
5. Bellegarda, J. Statistical language model adaptation: Review and perspectives. *Speech Communication 42*, (2004), 93-108.
6. Clark, H.H. and Brennan, S.E. Grounding in communication. *Perspectives on Socially Shared Cognition*, (1991), 127-149.
7. Fink, G., Schillo, C., Kummert, F., and Sagerer, G. Incremental speech recognition for multimodal interfaces. *Proc. IECON 1998*, (1998), 2012-2017.
8. Gales, M.J.F. Maximum Likelihood Linear Transformations for HMM-Based Speech Recognition. *Computer Speech and Language 12*, 2 (1998), 75-98.
9. Grover, D.L., King, M.T., and Kushler, C.A. Patent No. US5818437, Reduced keyboard disambiguating computer. Tegic Communications, Inc., Seattle (1998).
10. Gunawardana, A., Paek, T., and Meek, C. Usability guided key-target resizing for soft keyboards. *Proc. IUI 2010*, ACM Press (2010), 111-118.
11. Hart, S.G. Nasa-Task Load Index (Nasa-TLX); 20 Years Later. *Proc. Human Factors and Ergonomics Society Annual Meeting*, (2006), 904-908.
12. Hoggan, E., Brewster, S.A., and Johnston, J. Investigating the effectiveness of tactile feedback for mobile touchscreens. *Proc. CHI 2008*, ACM Press (2008), 1573-1582.
13. Karat, C.M., Halverson, C., Karat, J., and Horn, D. Patterns of entry and correction in large vocabulary continuous speech recognition systems. *Proc. CHI 1999*, ACM Press (1999), 568-575.
14. Kurtenbach, G. and Buxton, W. User learning and performance with marking menus. *Proc. CHI 1994*, ACM Press (1994), 258-264.
15. Lee, C.-H. and Gauvain, J.-L. MAP estimation of continuous density HMM: Theory and Applications. *Proc. DARPA Speech & Nat. Lang Workshop*, (1992), 185-190.
16. Levenshtein V.I. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady 10*, 8 (1966), 707-710.
17. Martin, T.B. and Welch, J.R. Practical speech recognizers and some performance effectiveness parameters. *Trends in Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ, USA, 1980.
18. MacKenzie, I.S. and Soukoreff, R.W. Text entry for mobile computing: Models and methods, theory and practice. *Human-Computer Interaction 17*, (2002), 147-198.

19. Mobile Speech Recognition | Voice Recognition | Windows Phone 7. http://www.microsoft.com/windowsphone/en-us/howto/wp7/basics/use-speech-on-my-phone.aspx.
20. Nuance – Dragon Dictation: iPhone – Dragon Dictation for iPad™, iPhone™ and iPod touch™ is an easy-to-use voice recognition application. http://www.nuance.com/for-business/by-product/dragon-dictation-iphone/index.htm.
21. Payne, S. Mental Models in Human-Computer Interaction. *The Human-Computer Interaction Handbook*. Lawrence Erlbaum, Mahwah, NJ, USA, 2009.
22. Rabiner, L. and Juang, B.H. *Fundamentals of Speech Recognition*. Prentice Hall, Upper Saddle River, NJ, USA, 1993.
23. Rabiner L.R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE 77*, 2 (1989), 257-286.
24. Rosen, L.D. A Review of Speech Recognition Software. The National Psychologist, 6(5), (1997), 28-29.
25. Selfridge, E., Arizmendi, I., Heeman, P., and Williams, J. Stability and accuracy in incremental speech recognition. *Proc. SIGDIAL 2011*, (2011), 110-119.
26. Sherwani, J. and Rosenfeld, R. The case for speech and language technologies for developing regions. *Proc. Human-Computer Interaction for Community and International Development Workshop*, (2008).
27. Skantze, G. and Schlangen, D. (2009). Incremental dialogue processing in a micro-domain. *Proc. EACL 2009*, (2009), 745-753.
28. Steele, J., To, N. The Android Developer's Cookbook: Building Applications With the Android SDK. *Addison-Wesley Professional*, 1st Edition, October 2010.
29. Suhm, B., Myers, B., and Waibel, A. Multi-Modal Error Correction for Speech User Interfaces. *ACM TOCHI 8*, 1 (2001), 60-98.
30. SWYPE | Type Fast, Swype Faster. http://www.swype.com.
31. Tanenhaus, M.K., Spivey-Knowlton, M.J., Eberhard, K.M., and Sedivy, J.C. Integration of visual and linguistic information in spoken language comprehension. *Science 268*, 5217 (1995), 1632-1634.
32. Traxler, M.J., Bybee, M.D., and Pickering, M.J. Influence of connectives on language comprehension: Eye-tracking evidence for incremental interpretation. *The Quarterly Journal of Experimental Psychology: Section A 50*, 3 (1997), 481-497.
33. Voice to Text Application Powered by Intelligent Voice Recognition | Vlingo. http://www.vlingo.com.
34. Zhai, S. and Kristensson, P.O. Shorthand Writing on Stylus Keyboard. *Proc. CHI 2003*, ACM Press (2003), 97-104.