Speech based text correction tool for the visually impaired

Md. Nafiz Hasan Khan[1], Md. Amit Hasan Arovi[2], Hasan Mahmud[3], Md. Kamrul Hasan[4], and Husne Ara Rubaiyeat[5]
Systems and Software Lab (SSL), Department of Computer Science and Engineering (CSE), Islamic University of Technology (IUT), Bangladesh
Computer Science, Natural Science Group[5], National University, Bangladesh
{[1]tomal04, [3]hasan, [4]hasank}@iut-dhaka.edu, [2]amit114423@outlook.com, [5]rubaiyeat@yahoo.com

***Abstract*— In this modern age of revolutionary smart devices, technology reigns supreme. Visually impaired users are a part of this modern world. They deserve to taste the beauty of this technology. However, they require assistive technology which is the concern of HCI (Human Computer Interaction). In this paper, we present a text correction tool that is entirely designed for the visually impaired. Instead of having to use traditional keyboard or mouse, they can write or edit text using speech only. The text can be read to the user using speech synthesizer. The user will be notified of different events through audio feedback and the user will be able to write text entirely using speech. Different voice commands have been designed to interact with this tool. Also, important modes can be activated using a single click on a mouse button. Since clicking either of the mouse button may not be very difficult for the visually impaired, we can use this option for flexibility. Existing speech based tools offer fast writing method, higher accuracy, but they are not optimized for the people with low or no vision. Text editing is a basic way to communicate with the computer. Providing a clear and efficient solution regarding this matter will obviously open a new door for the disabled which is in this case visual impairment. Five participants evaluated the system and the feedback was more than satisfactory.**

Keywords—Text Editor, Visually Impaired, HCI, Assistive technology, Google Speech, Speech based interaction

I. INTRODUCTION

Regular users communicate with the computer using mouse and keyboard. The keyboard is the core interaction device. Without eyesight or proper eyesight, it's impossible to communicate with the computer using these traditional input devices. There are 4 levels of visual function, according to the International Classification of Diseases -10 (Update and Revision 2006) [1]. Normal vision, moderate visual impairment, severe visual impairment, blindness. Moderate visual impairment combined with severe visual impairment are grouped under the term *"low vision"*: low vision taken together with blindness represents all visual impairment. The traditional input devices are almost useless for the visually impaired. So the visually impaired people need assistive technology in order to communicate with computers. Assistive technology [2] is an umbrella term that includes assistive, adaptive, and rehabilitative devices for people with disabilities and also includes the process used in selecting, locating, and using them. Assistive technologies for computer access and instruction includes Input and output devices, alternative access aids, modified or alternative keyboards, switches, special software, and other devices and software solutions that enable students with a disabilities to use the classroom computer for example. Text editing is a fundamental function of a computer. Everybody has the necessity of using a text editor every now and then. Visually impaired people require assistive technology such as a text editor that works completely using speech. Since they cannot see anything on a computer screen or write using our traditional keyboards, only way for them to use text editor is by speech and auditory feedback. He should be able to write using speech. A speech synthesizer should be able to read out what he/she has written. Then with a specific voice command, he should be able to replace any word or delete it. There are a few text writing methods using speech. But these require either hand gestures for text correction [3] or gaze [4]. Even though these methods are quite fast, but they are not optimized for the visually impaired. Our system is designed to allow a visually impaired to write texts using speech. The system reads back what the user has written. User can delete or replace words using voice commands. It works entirely using voice commands. We designed a few voice commands keeping in mind the constraints which we will discuss later. The commands are: "activate word replace", "activate word delete", "activate text reader", "and activate text writer", "close text editor". Here, Google speech API was used which managed to achieve very high accuracy.
The existing systems do not allow a user to edit text entirely using speech. Keeping the visually impaired people on mind, this tool was designed so that they can use this text editor with satisfaction. The remaining section of the paper contains the study of the existing works in this area, the proposed methodology, implementations, evaluation and conclusion.

II. RELATED WORKS

Use of voice modality is nothing new in the field of HCI but there are few works done to assist visual impaired for text entry and editing. SpeeG2 [3] is a smart text editor that uses speech for writing text. But it uses hand gestures for error correction. It uses Kinect to detect hand gestures. Our interface is almost like SpeeG2. It is a very fast and complete text entry tool. It provides GUI to insert, delete, and edit words. First, a user utters a sentence and the speech recognizer translates the

978-1-4673-9930-2/15/$31.00 ©2015 IEEE

spoken sentence into a sequence of words. At any time when a user speaks, the SpeeG2 GUI visualizes what the speech recognizer assumes to be the correct word sequence. Even if a user has not yet finished a sentence, partial results are shown in the GUI. When a sentence is spoken, the selection process becomes active. The user can start correcting the recognized word sequence by using the dominant hand as input modality. The hand movement is registered by a Microsoft Kinect and transformed to screen coordinates. Via the GUI the user gets continuous feedback about the speech recognition and the hand tracking. Note that the communication between the speech recognizer and the GUI has been realized via asynchronous network communication. This allows for abstraction and independent evolution of both components and depending on the scenario, our solution might be tailored with domain-specific speech recognition. Speech input and gesture-based correction can overlap and occur in parallel, providing more freedom to the user and potentially improving the performance. A user can also first speak a few sentences forming a paragraph and perform the corrections afterwards. However, visually impaired users will not be able to use the hand gestures to place a word to insert or delete or replace. Our tool provides that flexibility for the blind users. The users can select, replace, delete and get the screen read only using voice commands.

JAWS [5] is one of the most used screen readers by visually impaired people. It reads the content of the computer screen using synthesized artificial speech. JAWS provide speech and Braille output for the most popular computer applications on PC. However, it doesn't help writing or editing text. It doesn't provide a complete solution for the visually impaired.

Speech Dasher [4] supports writing text through a combination of speech and the Dasher user interface. Their work extends Dasher by visualizing the output of a speech recognizer rather than single letters. The speech-based output consists of n-best word candidates that can be selected, resulting in a higher text input rate. Based on a two-step model, the user first utters a sentence and then presses a discrete button to disable the microphone before switching to the correction phase where the recognized sentence can be modified via the zoom able Dasher interface. The Speech Dasher prototype also targets users with limited mobility and is used on a personal computer system since the correction phase can be controlled via mouse or gaze. The output of a speech recognizer offers many different possible candidates for one utterance. In Speech Dasher, only the top predictions are directly added to the interface. Excluded options and Character-based selection can be enabled by selecting a dedicated star character (*). The developers made this design choice because too many alternative choices would increase the difficulty in navigating the interface. Speech Dasher uses the Sphinx4 speech recognizer with a trigram language model. Depending on the participant, a UK or US acoustic model is applied. Three participants were used in the user study and the average text entry rate was 40 WPM compared to the 20 WPM of the original Dasher interface. The word error rate (WER) for Dasher was 1.3% while the WER for Speech Dasher was 1.8%. Note that these numbers are optimized using user-specific native speech recognition training. The performance for a non-native English speaker (i.e. German) was not as good due to the fact that the recognizer used a US acoustic model. Furthermore, the visualization was not optimal for viewing at a distance and the use of discrete start and end buttons does not map well controller-free input.
BrailleTouch is a mobile touchscreen text entry for the visually impaired [6]. Phones with multi-touch can use this system. However, users have to get accustomed to this environment. Audio feedbacks are used for notifying the users. Based on the standard Perkins Braille, BrailleTouch implements a six-key chorded braille soft keyboard.
Eleven blind participants typed for 165 twenty-minute sessions on three mobile devices: 1) BrailleTouch on a smartphone; 2) a soft braille keyboard on a touchscreen tablet; and 3) a commercial braille keyboard with physical keys. Expert blind users averaged 23.2 words per minute (wpm) on the BrailleTouch smartphone. The fastest participant, a touchscreen novice, achieved 32.1 wpm during his first session. Overall, participants were able to transfer their existing braille typing skills to a touchscreen device within an hour of practice. We report the speed for braille text entry on three mobile devices, an in depth error analysis, and the lessons learned for the design and evaluation of accessible and eyes-free soft keyboards.

Use of text-to-speech (TTS) and speech-to-text (STT) for various fruitful activities has become vital now a days. A study investigated the application of speech-to-text to assist learning in a face to face seminar [7]. A transcript was generated using speech-to-text recognition. Some participants were able to use the transcript effectively while others were not able to. Various inspiring uses of TTS and STT show how this wonderful technology can be used for the visually impaired people.

None of these existing works provide a complete solution for the text entry and correction for the visually impaired with a good level of user satisfaction. Our speech based text correction tool is flexible enough for the visually impaired. For example: if the user says "activate text writer", the tool will go to text writing mood and give voice feedback "writing mode activated". Thus the user can switch to different modes using voice. Also, there is the flexibility of using mouse for switching to different modes. For example a left mouse click will trigger the word replace mood and the user will be notified through audio feedback. Our goal was to provide a total solution to the visually impaired for text writing and correction.

III. PROPOSED METHODOLOGY

We designed our tool so it can be completely used only using speech. We used Google speech for speech recognition and speech synthesis. Basically, it works with voice commands.

A. System Description

This system has two modules. ***The recognition module*** and ***the correction module***. The recognition module handles the speech input from the user and produce desired output which is the corresponding text from the user speech. The correction module handles the edition or deletion of a word. The basic mechanism of this tool is shown in *Fig. 1*.

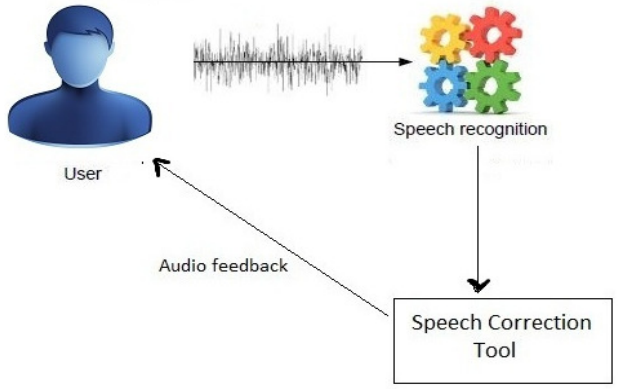

Fig. 1. Interaction with the speech tool

The functionalities can be controlled using voice commands or mouse buttons. Since mouse has only two major buttons visually impaired people may use the mouse buttons to switch between different functionalities.

The list of voice commands that we have used is given in Table 1:

TABLE 1. SPEECH BASED COMMANDS AND FUNCTIONALITIES

Commands	Functionality
Activate text writer	Activates the writing mode
Activate text reader / (Mouse Double Click)	Activates the speech synthesizer
Activate word replace / (Left Mouse Click)	Triggers the mode to replace a specified word
Activate word delete / (Right Mouse Click)	Deletes a specific word
Close text editor	Exits the text editor

For speech recognition and synthesis, we used Google Speech API.

1) Google Speech Recognition

Google speech works online. First, the voice commands are recorded. Then the recorded audio is sent to the Google server. Google returns the result with the highest probability and a set of other possible results. The language can be set for any accent. The punctuation marks can be written using speech also. The voice synthesizer can read out any text.

2) Recognition module

At first the sentence spoken by the user is recorded in a flac audio file. Then the audio data is sent to google server. Then Google processes this audio using its own algorithms. Then it returns the highest possible result along with other possible results. This process is shown in *Fig. 2*.

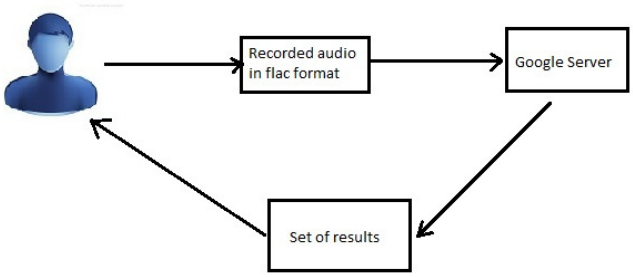

Fig. 2. Recognition module

3) Correction module

For text correction, user has to enter word replace or word deletion mode. After entering a mode, the user will be notified through the Google speech synthesizer. Then he will be asked for a specific word which should be deleted or replaced. After the user gives his choice, he will be notified through audio feedback whether it is found or not. If it is found, he will be asked which word he would like to replace it with. Then if he speaks a word, the replacing will occur successfully. The process is shown in *Fig. 3*.

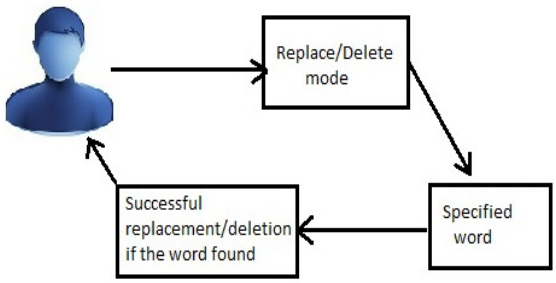

Fig. 3. Correction module

B. Implementation

This tool was implemented using Google Speech API.
For this we needed the Java Sound API jar (jsapi.jar) file and we needed java Flac Encoder to convert wav audio file to flac audio file.

The Launching class is,
Editor – this class initializes the google speech API.
Other important classes are given in Table 2.

TABLE 2: IMPORTANT CLASSES OF THE SYSTEM (PROTOTYPE) AND THEIR FUNCTIONALITIES

Microphone	Initializes microphone for recording audio
FlacEncoder	Converts wav to flac
GoogleResponse	Holds the response from google server with the confidence value
GSpeechDuplex	Allows continuous recognition
Recognizer	Submits and retrieves recognized text
Synthesizer	Reads out a text

C. *User Interfaces and Interaction*

We developed this tool using JAVA. Our focus was to provide the text correction using speech as well. This way, it can turn into a very important text editor for the visually impaired.

The system initially starts with text writing mode. User can write texts using speech and then he can hear what is written. After that he can replace any word he wishes. In *Fig. 2*, we can see how a text is replaced using this tool. First user goes to text replace mode either using a left mouse click or through the voice command "*Activate word replace*". Then he is notified through the synthesizer "*Replace mode activated. Please speak the word you would like to edit*". Then the user gives a specific word. The system responds "*No such word found*" if there is no word the user specified. If there is one or more occurrences, the system says "*word found. What would you like to replace it with?*" The user says the intended word to replace the original. The system replaces the word and gives feedback "*Word successfully replaced*". This activity is illustrated by these screenshots (*Fig. 4*).

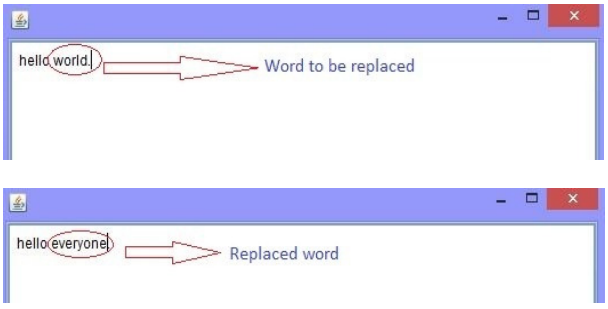

Fig. 4. Word replacement function

We can also delete any word using speech. In *Fig. 5*, we can see the snapshot of how a word is deleted. User has to enter deletion mode using right mouse button or voice command "*Activate word delete*". Then the same way he is notified and asked for a specific word. If found, the word is deleted and he is notified "*Word successfully deleted*".

Fig. 5. Word deletion

IV. EVALUATION

To evaluate the system and the proposed prototype, we have gone through five participants in Centre for Disability in Development (CDD) which is located in Savar, Dhaka, Bangladesh [8] to use the prototype and taken their feedback. We have used User Evaluation technique to evaluate our system. All users got an introduction and a maximum training period of five minutes for the prototype before the evaluation was conducted. This enabled users to get used to the speech recognition engine and how their pronunciation influences the results.

4.1 *Participants and Strategy*

Empirical evaluation [9] strategy was used to evaluate the system. In empirical evaluation we evaluate the system by the end-users. There we used dependent variables such as word error rate before and after correction. The study featured five participants (aged between 30 and 50 years). Two participants had a computer science background but nobody used voice recognition on a frequent basis and 60% of the participants have never used speech recognition. Among the participants all of them speak Bengali but can also speak English. The participants had a variety of different accents coming from different parts in Bangladesh. All tests were performed with the generic English US acoustic model of the Google Speech Recognizer. As argued in the introduction, our goal is to provide a generic text entry solution that requires no training at all.
All the voice commands are very familiar with the participants.

4.2 *Performance Measurement*

For the text editor option of our system, in each test, we recorded the time it took the participants to process the sentence starting from the point when the first sound of the sentence was uttered until the time when the entire sentence (including the full stop) was processed. To compute the text entry speed, the standard measure for a word was used [10]. A word is considered a five to eight character sequence, including spaces and punctuation. The text entry speed is computed in words per minute (WPM) [11]. In addition, the word error rate (WER) [12] was computed. The WER measure was computed as the

word-level edit distance between the stimulus text and a user's response text, divided by the number of characters in the stimulus text. For each participant, the time they spent with alternative correction methods (e.g. insertion dialogue or spelling mode) was also recorded. The goal was to observe whether users needed these correction methods and whether they contribute to the reduction of the WER. The result of users' performance is given in Table 3. When speech recognition performed poorly, the option to skip the sentence was allowed as well (without resetting the observed time).
In our solution we have used following terms to represent the process of calculating Words per minute and the word Error Rate:

Gross WPM (Words per Minute) [11] is a calculation of exactly how fast we can type with no error penalties. The gross typing speed is calculated by taking all words typed and dividing by the time it took to type the words in minutes. When calculating typing speed, a "word" we take is any five characters. For instance, "I love keyboarding, don't you?" would be counted as 6 words (30 characters / 5) and not just 5. This makes sense because typing "deinstitutionalization" obviously should count more than typing "my". Spaces, numbers, letters, and punctuation are all included, but any function keys such as Shift or Backspace are not included. This makes the number of words easy to calculate. Simply count all typed entries and divide by five to get the number of words typed. To give an example, if we type 200 characters in 1 minute, our net wpm typing speed would be (200 characters / 5) / 1 min = 40 WPM. If we type 200 characters in 30 seconds our net speed would be (200/5) / 0.5 = 80 WPM.

$$\text{Gross Word Per Minute} = \frac{A}{C} * \frac{1}{T} \quad (1)$$

Here, A= All typed entries, C= Number of characters in a word, T= Time (min).

A Net WPM [11] calculation is preferred for measuring typing speed as opposed to the Gross WPM computation since including mistakes will give a more complete picture of our true typing abilities. Gross WPM is used in the calculation of the Net WPM calculation which merits its mention.
To calculate Net WPM, we have to take our gross WPM result and subtract the amount of errors you left in per minute, also known as the error rate. To calculate error rate, simply we can divide the number of errors by the time we typed for in minutes. For example, if we typed for two minutes with a gross typing speed of 80 WPM and left in 8 mistakes, our error rate would be (8 errors / 2 minutes) = 4 errors per minute. Our net typing speed would then be (80 - 4) = 76 WPM. Note that for every mistake we have made per minute our typing speed goes down by 1 WPM.

$$\text{Net Word Per Minute} = Gross\ Word\ Per\ Minute - \frac{UE}{T} \quad (2)$$

Here, UE=Uncorrected Errors, T=Time (min)

We can have error in the text while writing and we calculated the WER while writing by:

$$\%\text{WER} = \frac{NWC}{TW} \quad (3)$$

Here, WER=Word Error Rate, NWC= Number of words which are not correct, TW= Total Words

And after correction the (%) WER is calculated by:

$$\%\text{WER} = \frac{S+D+I}{N} \quad (4)$$

Here, S=number of substitutions, D=number of deletions, I=number of insertions, N=number of words in reference

TABLE 3: AVERAGE WPM, WER BEFORE CORRECTION AND WER AFTER CORRECTION PER USER

Participants/Users	Avg. WPM	(%)WER(before correction)	(%)WER(after correction)
Participant 1	76.3	15.3	2.7
Participant 2	80.4	19.75	1.87
Participant 3	77	12.56	4.6
Participant 4	79	22.77	3.89
Participant 5	73.7	3.35	1.2

In the above table we have shown the average Word Per Minute [11], WER (before correction), and WER (after correction) for every user. And each of the participants or user gave five tests.

And we take the average by using following formulas:

$$\text{Average Word Per Minute (FEP)} = \frac{\sum_1^N F}{N} \quad (5)$$

Here, FEP= For Each Participant, F= Sum of Word Per Minute of all test, N=Total number of test

$$\%\text{WER (BC) (FEP)} = \frac{\sum_1^N P}{N} \quad (6)$$

Here, WER=Word Error Rate, BC= Before Correction, FEP= For Each Participant, P= Sum of Word Error Rate of all test, N=Total number of test

$$\%\text{WER (AC) (FEP)} = \frac{\sum_1^N P}{N} \quad (7)$$

Here, WER =Word Error Rate, AC = After Correction, FEP = For Each Participant, P = Sum of WER of all test, N=Total number of test.

V. CONCLUSION

We presented the speech based text correction tool so that the visually impaired people can use a complete text editor. Existing systems that use google speech work fine but those require the sense of vision in order for a user to use. This constraint makes those systems unusable. Our proposed tool mainly targets those users with limited to no vision. For our future work, we plan to build a complete system for the visually impaired and as the speech based text correction tool is not totally error free we will also be taking further steps to improve the strategy and method to make it totally error free. A voice keyboard can be of great flexibility for them. Also, the user will be able to search any file. He can listen to mp3 players and read pdf files. All the functionalities will be added to our system so we can provide them a complete flexible solution to communicate with the computer.

ACKNOWLEDGMENT

We thank the Centre for Disability in Development (CDD), Dhaka, Bangladesh for their support. Without their assistance it was impossible to conduct requirement gathering step. We thank CDD to actively participate in evaluating our prototype. We would also like to thank the Systems and software lab (SSL), department of Computer Science and Engineering (CSE), Islamic University of Technology (IUT), Dhaka, Bangladesh for valuable comments and guidance during the research.

REFERENCES

[1] Factsheets by World Health Organization (WHO), Visual impairment and blindness,http://www.who.int/mediacentre/factsheets/fs282/en/, last accessed on 2014-2015.

[2] Scherer, Marcia J. "Assistive technology: Matching device and consumer for successful rehabilitation," American Psychological Association, 2002.

[3] Lode Hoste and Beat Signer, "SpeeG2: ASpeech and Gesture-based Interface for Efficient Controller-free Text Entry," ICMI'13, Proceedings of the 15th ACM on International conference on multimodal interaction, Pages 213-220,Sydney, Australia, December 9-12 2013.

[4] Keith Vertanen and David J.C. MacKay,"Speech Dasher: Fast Writing using Speech and Gaze," CHI 2010, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Pages 595-598, Atlanta, Georgia, USA, April 2010.

[5] JAWS(Job Access With Speech), http://www.freedomscientific.com/Products/Blindness/JAWS last accessed on 2014-2015

[6] Caleb Southern, James Clawson, Brian Frey, Greory D. Abowd, Mario Romero, "An Evaluation of BrailleTouch: Mobile Touchscreen Text Entry for the Visually Impaired", MobileHCI'12 , San Francisco, CA, USA, September 21-24 2012.

[7] Rustam Shadiev, Wu-Yuin Hwang, Yueh-Min Huang and Chia-Ju Liu, "Investigating applications of speech-to-text recognition techonology for a face-to-face seminar to assist learning of non-native English-speaking participants," Published Online 21 January, 2015.

[8] Centre for Disability in Development (CDD), http://www.cdd.org.bd/ , last accessed on 2014-2015

[9] David N. Chin,"Empirical Evaluation of User Models and User-Adapted Systems," Umuai, The Journal of Personalization and Research, Springer, Volume 11, Issue 1, pp 181-194, March 2001

[10] MacKenzie, I. Scott, and R. William Soukoreff. "Phrase sets for evaluating text entry techniques." CHI'03 extended abstracts on Human factors in computing systems. ACM, 2003.

[11] Ahmed Sabbir Arif, Wolfgang Stuerzlinger, "Analysis of Text Entry Performance Metrics," Dept. of Computer Science & Engineering York University

[12] Iain McCowan, Darren Moore, John Dines, Daniel Gatica-Perez, Mike Flynn, Pierre Wellner, Herve Bourlard, "On the Use of Informa tion Retriev al Measures for Speech Recognition Evaluation", IDIAP research report, 04-73, March 2005.