UNIVERSITÄT KOBLENZ · LANDAU
Faculty 4: Computer Science

WeST People and Knowledge Networks
Institute for Web Science and Technologies

Hands-free Text Editing using Voice and Gaze

Master's Thesis

in partial fulfillment of the requirements for
the degree of Master of Science (M.Sc.)
in Web Science

submitted by
Sabin Bhattarai

First supervisor:	Prof. Dr. Steffen Staab Institute for Web Science and Technologies
Second supervisor:	Korok Sengupta Institute for Web Science and Technologies

Koblenz, December 2019

Statement

I hereby certify that this thesis has been composed by me and is based on my own work, that I did not use any further resources than specified – in particular no references unmentioned in the reference section – and that I did not submit this thesis to another examination before. The paper submission is identical to the submitted electronic version.

	Yes	No
I agree to have this thesis published in the library.	☐	☐
I agree to have this thesis published on the Web.	☐	☐
The thesis text is available under a Creative Commons License (CC BY-SA 4.0).	☐	☐
The source code is available under a GNU General Public License (GPLv3).	☐	☐
The collected data is available under a Creative Commons License (CC BY-SA 4.0).	☐	☐

. . . . . .

(Place, Date) (Signature)

Note

- If you would like us to contact you for the graduation ceremony,

  please provide your personal E-mail address: . . . . . .

- If you would like us to send you an invite to join the WeST Alumni

  and Members group on LinkedIn, please provide your LinkedIn ID : . . . . . .

Acknowledgement

I would like to express my deepest gratitude to my thesis advisors Prof. Dr. Steffen Staab and Korok Sengupta for their commendable supervision and motivation throughout the course of my Masters thesis. Korok Sengupta has been a constant presence from the beginning and has provided the support and guidance I required from planning the experiments to implementation and writing of the final paper. He has been inspirational and has been continuously pushing for my success in every step of the way. The wisdom and knowledge I received from my advisors has played a big role in completion of my thesis. This project has been a great learning opportunity for me.

I am also grateful to the participants who patiently sat through the experiment and showed exceptional calmness and composure.

My family has also been my support system, without whom this success would not have been possible. Thank you.

Contents

1 Introduction

The experience of voice-based interactions in a hands-free computing environment has gained prominence with a new generation of voice interaction systems. Cloud services like Zoho Docs[1], Google Docs [2], and Microsoft Office 365 [3] have integrated speech recognition tools [11] for document creation and editing tasks. Another modality in a hands-free environment is human gaze - which too has been successfully investigated and used for text entry [18] and navigation[13]. Such hands-free modalities can be powerful for users who are unable to use their hands to operate physical input devices like keyboard and mouse. There has also been increased interest in developing and optimizing completely hands-free technologies [23] to provide accessible systems for the disabled. Despite ongoing research efforts and steady improvements in hands-free technologies, there are still limitations to its efficiency. In a hands-free environment, forming an error-free document is slower using speech only modality because of accuracy challenges that are unfaced by traditional input devices. Identifying and correcting errors within the created text were shown to account for a major amount of delay [21]. Card *et al.*[7] also reports that up to one-fourth of a user's time was spent on error correction. Poor error handling is, therefore, a significant problem that requires improvement to optimize hands-free technology for the disabled.

Editing errors involve identification of an error, navigating to the location of the error and then applying corrective measures to remove that error. Different approaches have been implemented to improve error navigation and resolution. At present, speech-based navigation within textual documents has used a target-based approach or a direction-based approach [42]. Direction-based navigation involves users giving a series of commands like "Select paragraph 3", "Go to Line 2", then "Select a word" to navigate to the erroneous word. However, constructing a valid direction-based command can be complex and time-consuming for the user. In contrast, target-based navigation involves giving a single command to highlight the target word, for example, *"Select [target word] "*. Target-based navigation also has its limitations. One difficulty will arise if the target word appears on the screen multiple times requiring further steps for navigation. So, an alternative hands-free tool that can act as a pointing device to specifically target the error is necessary. While gaze, as a hands-free modality has been used to navigate around web browsers, [30] we investigate the use of gaze modality to navigate to a transcription error and evaluate if gaze facilitates in selecting the transcription error faster than speech navigation in a hands-free environment.

Although different research [3,10,12,18] has focused on improving the text entry system to prevent an error from occurring, very little work has been done on the error correction process in a hands-free environment. Different techniques like deep recurrent neural network [16], using contextual information [2], pruning long short-term memory (LSTM) model [17] reduces the error, but these techniques focus on error prevention and not error correction. We,

[1] `https://www.zoho.com/docs/`
[2] `https://www.google.co.uk/docs/about/`
[3] `https://www.office.com/`

therefore, research on error correction process of locating the error and correcting it when each of the error prevention technique fails.

After locating the error, the correction process may involve re-speaking the word, spelling out each character of the word or choosing from a list of alternative words [45]. However, as an error in transcribed words often involves misrecognition of the word by similar sounding words, re-speaking the word would increase the chance of repeating the same error. [45] also showed that re-speaking had lower accuracy in comparison to spelling out the word. But spelling out each character can again be time-consuming.

1.1 Thesis statement

We investigate if the use of hands-free multimodal approach, where Gaze provides spatial context and voice provides an intention to user's needs, can improve the overall text editing process. We will design a web platform that uses gaze input and voice input to navigate to and edit errors. A summative and formative evaluation of our multimodal approach for text editing would be performed. To understand the robustness of these hands-free editing methods, an evaluation will be done in a controlled experimental environment that will facilitate in understanding the impact of different noise and light conditions. The thesis statement for our work is:

"Hands-free text editing using multimodal approach (Voice and Gaze) can improve the text editing process than using unimodal approach (Voice only)"

In this research, voice-based editing was used as a baseline for our experiments and we designed a system that facilitates voice-based editing that is currently most popular in the scientific literature. When developing error correction using voice only as a unimodal approach, we extended the idea of navigation via mapping discussed by Schalkwyk and Christian [41, 8] which is considered faster compared to target-based navigation by Sears *et al* [42].

Our navigation via mapping technique will assign numbers to every word in a text area (sequentially in a left-to-right, top-to-bottom manner), and the user speaks that number to locate to the word as discussed in our design investigation. We therefore, analyzed our baseline: traditional voice-based editing approach with our proposed system for text editing using a multimodal approach i.e. using gaze plus voice.

As we know the editing task involves identifying the error, locating to the error and correcting the error, we chose to build a system that provides an interface where the user can follow two steps during the error correction process. First, when an error word is selected, the user can select from the suggestion list or re-spell each character in the word. Multiple errors was corrected in multiple iterations, for example, two errors within the transcribed text would have the first error corrected followed by the second error corrected. Such system

would allow us to perform different statistical analysis under different experimental scenarios for proving the validity of our thesis statement.

1.2 Thesis contribution

We present novel system to compare and analyse the unimodal approach of error correction against the multimodal approach. The integration of gaze based interaction technique with voice only approach and using it to design a robust system for better error correction is part of the thesis contribution.

Integration of Gaze and Voice based input: When performing transcription using speech recognition tool we require asynchronous voice based input so as to provide user friendly experimental setup. Similarly, using gaze alongside voice to collect experimental data need to be integrated well for accurate analysis. Therefore, as part of the thesis contribution, we introduce pilot studies and feedback sessions for better system design.

Design and implementation for gaze-based system: We identify key design challenges to build a system that is user friendly while supporting effective gaze-based interaction. We formulate the design architecture and feedback analysis to solve some of the challenges to facilitate better hands-free editing system.

2 Background

2.1 Voice based interaction

Speech recognition errors and improper handling of them is regarded as one of the main weakness in limiting the performance in speech-based applications [48, 39]. Sears *et al.*[42] suggests that 66% of the time is spent in correcting errors with only 33% of time used in transcribing. Karat *et al.* [21] also provides a detailed analysis of how users productivity gain of using speech dictation is mislaid during error correction. It, therefore, becomes important to reduce the time spent in correcting errors by having a better mechanism for handling wrongly transcribed words.

The concept of interactive correction was introduced by Martin *et al.*[28] for errors caused during speech recognition. They suggested that the results obtained during recognition would be stored in a buffer and interactive editing of buffer would occur by deleting a single word, deleting the entire buffer, or by re-speaking. Robbe *et al.*[38] believes that the most intuitive interactive correction method is by re-speaking the error word supporting *Briton et al.* [6] who claims re-speaking as preferred repair mechanism in human to human dialogue. However, human misunderstandings can be corrected after two or three repeats, but speech recognition errors can go into a loop of unpredictability. This would mean multiple repeats before a resolution is obtained. Such lengthy attempts can be time-consuming, frustrating and can result at the end of an interaction [37]. [1, 34] suggest a second interactive correction method as an alternate to re-speaking i.e. selecting correct words from the list of alternative words. They, however, fail to discuss the disadvantage of such an approach when the list of alternative words does not contain the correct word for replacement.

Error correction in speech recognition platforms show two phases in interactive correction[3]: first, an error must be identified and located; then it can be corrected. Locating point of error was accomplished by using speech-based navigation only[26]. Navigation was done using a direction-based and target-based approach. Direction based navigation used commands such as (e.g., “Move left” followed by “Stop”) i.e. the mouse would move continuously left until “Stop” command. However, this can cause the cursor to overshoot the target needing further attempts to locate the target. Alternatively, target-based navigation used predefined discrete commands like ’select word’ to locate the point of error but has its own limitation when words appear multiple times in the document.

De Mauro *et al.* [10] discussed the design of a voice-controlled mouse aimed at simplifying the direction-based commands. Their approach shortened the direction-based commands like *’Move Left’* by mapping it to using simple vowels ’A’ i.e. uttering ‘A’ would have continuous movement of mouse towards left. Each vowel was mapped to commands while controlling the mouse movements. Whilst this method attempts to reduce the time to say lengthy commands, users need to spend time familiarizing with the mapping of commands.

Many efforts in research have been made in voice-controlled navigation within the document[8, 26, 10, 31, 42, 41] to efficiently locate to the word through navigational voice commands. In voice based research, speech recognition tool transcribe the voice to text (Figure 1)[4] which is then mapped to set of predefined commands for navigation. Voice controlled navigation can be classified as continuous, direction-based, target-based and navigation via mapping. However, each navigational approach comes with challenges. Continuous navigation technique presented by [26, 10, 31] lacked the ability to fluidly and continuously obtain movements without having to repeat the commands. Similarly, complexity in constructing valid direction-based commands alongside longer navigation sequences created harder navigation within documents. Although target-based navigation in locating error is efficient compared to that of direction-based navigation, recognition error when executing commands can be an issue. Therefore, an increase in command statements can be error-prone.

Figure 1: Speech recognition tool used for transcribing text to detect commands for navigation.

Furthermore, navigation via mapping discussed by [41, 8] is considered faster compared to either continuous, target-based or direction-based navigation. One such mapping technique is using numbers that allow each word within a text to be assigned a number (sequentially in a top-to-bottom, left-to-right manner), and the user speaks that number to locate to the word. In our experiment when analyzing voice-based editing and performing navigational tasks, we will implement mapping by number technique.

[4] https://ul.gpii.net/content/speech-recognition

2.2 Gazed-based interaction

In a hands-free environment, gaze can be considered as a useful modality in performing actions such as pointing or selection. Gaze interaction requires an eye tracking device which calculates the focal point by deducing the positions of the eyelid. When a user looks at a certain point on the screen, the position selection is triggered depending on the dwell time concept. Dwell time is a predefined time out where a selection gets activated.

Eye tracking devices have been used as a pointing device to select targets (Figure 2) [5] and have been tested using the ISO multi-directional tapping task [BB12] with satisfactory performance as compared to a mouse input device. The first gaze-enabled pointing technique was presented by Zhai *et al.*[49]. They presented an acceleration technique in which a gaze cursor is warped to the vicinity of the target region the user was looking at. They reported that while the speed advantage was not obvious over mouse pointing, almost all users subjectively felt faster with the pointing technique by gaze. As the cursor movement distance is shortened using gaze as per Fitts law[14], this method would reduce the overall time of navigation. Although they used gaze to get to the area of interest, the specific pointing to the desired item was done using a mouse. This highlights the main limitation of gaze-based navigation which is fast but lacks accuracy.

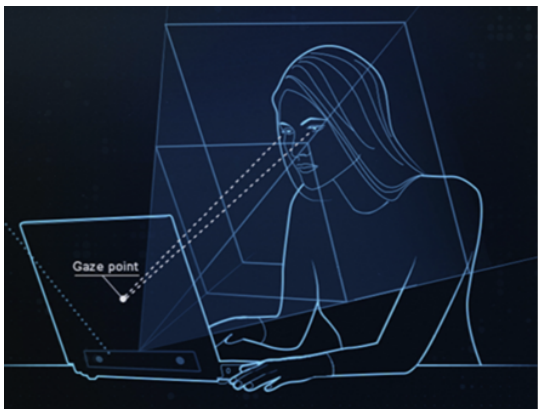


Figure 2: Eye tracker used for gaze point on screen

Jacob *et al.*[19] laid the foundation for gaze-based interaction by introducing dwell-based activation, gaze-based hot-spots, and context-awareness. Gaze-based interaction was used for selecting objects, moving an object, scrolling text and pointing menu commands. This work, however, focused on sufficiently large targets and less accuracy was observed when dealing with smaller targets. [24] also used the dwell-based technique for pointing and selection. Visual feedback was necessary from the user to activate or deactivate the gaze to the target location. The user looking away before the dwell period would mean deactivation. The

[5] https://help.tobii.com/hc/en-us/articles/115003827934-Get-started

accuracy of eye tracker was maintained by using the zooming technique which also required activation using dwell-based gazing. Similarly, another dwell was necessary for selecting the target after the zoom. This meant delays due to a number of discrete steps involved.

2.3 Multimodal (Gaze + Voice) editing

Prior multimodal research from Oviatt *et al.*[35, 36] combining pen-based gesture and voice, Mantravadi *et al.* [27] combining speech and gaze modalities have shown the importance of multimodality over singular modality. Oviatt *et al.*[35, 36] outlines multimodal solutions (combination of more than two input modalities, for example, pen-and-speech or mouse-and-speech) as a superior method compared to speech only solutions for a spatial navigation task. Similarly, Mantravadi *et al.* [27] conducted experiments on multimodal interactions using voice and gaze together. Using the multimodal approach in context of predicting user's menu selection from a large of screen displayed options, it was shown that the use of speech and gaze together had improved efficiency than modality acting alone.

Jacob *et al.*[20] defined an interface that identifies and provides a solution to errors posed by the imperfect recognition of user input. They discuss and introduce benefits of using multi-modalities for error correction whereby switching between modalities during repeated errors can be helpful. Oviatt *et al.*[37] performed a simulation study using multimodal error correction. They suggested that users switched between modalities naturally to avoid repeating errors. Another study by Oviatt *et al.*[37] analyzed different GUI based interfaces that used speech input and pen input modalities. This study presented that by carrying out multimodal interaction during correcting error, the overall task completion time was improved.

McNair *et al.*[29] took a multimodal approach in error correction with an explicit focus on speech-based selection and interactive correction methods. Selection method involved target-based navigation using mouse and interactive correction was done by re-speaking or by selecting an alternative word returned by the recognizer for the original utterance. This multimodal correction approach taken by McNair assumed that the correct word would be included in the list of alternative words. This assumption had a severe limitation because the correct word may be far down the list or be missing from the suggestion list.

Similarly, Suhm *et al.*[45] explored further into multimodal approaches for correcting recognition errors using keyboard, mouse, stylus, and speech. It was observed that using speech only for correcting error was slower compared to using multimodal techniques. One important step in correcting error involved locating the error. The multimodal technique used in locating the point of error was done by using touchscreen-based navigation or mouse. Likewise, Danis *et al.*[9] developed an editor using speech but used the mouse to accomplish cursor movements.

Several researchers have explored speech-based navigation in combination with the mouse, keyboard and or touch but very few have considered complete hands-free environment. An experiment was done by Miniotas *et al.*[32] that takes a multimodal approach for selecting a target using gaze and voice. This work presents a dwell-based activation of a target while highlighting the area around the region of the target in different colors. This allows the user to verbally choose a color if the gaze misidentifies the target. This approach makes use of gaze and speech modality to try improving the accuracy of target selection. Sengupta *et al.*[43] have also demonstrated the use of the multimodal framework in hands-free web browsing and found improved performance for example multimodal browser performed 70% better in link selection activity in comparison to unimodal approach.

Bourguet *et al.*[5] suggests that no perfect error correction mechanism has yet been designed and claims that an appropriate mechanism focusing on speed is to be required in the context of error correction. More importantly, the paper claims that little work was done for correcting errors using hands-free modalities. In a hands-free environment, as we cannot make use of a mouse, keyboard, and touch as a solution to correcting recognition error, there must be a comprehensive study on alternate hands-free modalities. We, therefore, plan to create a fully integrated platform for error correction that is equipped with eyes and voice commands to help non-able-bodied individuals.

3 Research Problem

In a hands-free environment, there has been little work optimizing the error correction process when speech recognition tools wrongly transcribe words. Existing solutions so far all have relied mostly on the mouse, keyboard, and or touch for error correction invalidating the focus we have on the hands-free environment. Similarly, hands-free techniques used while correcting wrongly transcribed words mostly used singular modality (for e.g., Speech only). Some multimodal approaches taken were slow and showed poor performance as discussed in previous chapter.

We therefore design a novel approach for hands-free voice based text entry where error correction is done using voice and gaze. For this multimodal approach we implement two version i) dwell-time selection ii) voice command selection. Dwell-time selection allows user to select the error word using gaze for 1 second dwell time. The selected word is then edited using voice commands. Dwell-time selection could face midas touch problem i.e. erroneous word selection if dwelled to any word for 1 second. We therefore introduce command selection approach, where dwelling and saying "Select" command selects the word.

The aim of our master thesis would, therefore, be to investigate the following research questions:

- **RQ1**: How can we integrate gaze with voice-based text entry for error corrections.
- **RQ2**: Does the multimodal approach for error correction using speech and gaze improve the performance of hands-free text editing in comparison to unimodal approach?

4 Methodology

In this chapter, we will outline the generic challenges faced for our research work and describe different approaches to design and investigate the solution. We will also discuss different technical aspects of the implementation to support our research problem. As our objective of the research is to analyse multimodal and unimodal approach for error correction, we required a system that supports use of both voice and gaze seamlessly.

4.1 Generic challenges

Different challenges were faced to identify and implement the correct and robust system for our experiment. The use of voice and gaze for error correction had to integrated well and be usable under different experimental setups. The usability, speed and error rate were among the challenges that needed to be addressed for our research work.

4.1.1 Usability

A better and usable method for error correction using voice only and using gaze with voice had to be designed. Different experimental tasks meant different usability criteria under different environment that would improve on the traditional approach of correcting errors. The integration between voice and gaze had to be uniform across the system for better user acceptance. It was therefore important for users to feel the ease of use and learnability when using our system.

4.1.2 Speed

Our system required the total elapsed time for one task cycle – the first step to the last step-including idle time to be minimum. The completion rate without unnecessary steps and no idle time between steps was important to realise. The interaction time between the system and voice input as well as gaze input had to be instant. It was therefore a challenge to build a smooth-running system by reducing the time required to perform the task and increase the output achieved.

4.1.3 Error Rate

One important aspect in our experiment was to have accurate eye tracking device and voice recognition tool. The accurate implementation of asynchronous voice input alongside gaze input was a challenge to overcome within our system. From calibration of eye tracker to gaze point click on the screen, to voice integration had to be accurate and error free. Similarly, the differentiation in gaze click against the normal gazing on the screen was a challenge to address.

4.2 Design Investigation

As per our research objective we planned to implement two experimental scenarios by leveraging the use of gaze and voice. For our experimental scenarios i) Read and Correct Task ii) Image description task using unimodal approach (Voice only) and multimodal approach (using Voice and Gaze), we performed pilot study to better design our system. This allowed us to further plan what voice commands were better suited for our experiment and how gaze could be integrated with voice.

4.2.1 Pilot Study I:

In order to make use of voice for error correction in transcribed texts, we chose to perform a study in investigating the design challenges that Speech API would have. For this we chose Google Docs which was a popular voice-based text entry system. As Google Docs had built-in error correction mechanism for wrongly transcribed texts, we were able to obtain feedbacks to better understand the advantages and disadvantages of the system.

Three university graduates and two bachelor's degree students (3 male, 2 female, ages 20-29) volunteered for our study. Although, each individual was familiar and have used Google docs, they were not aware of the speech-based transcription feature to construct text. Each participant was therefore explained the purpose of the study and shown the working of the speech-based transcription in Google docs. They were then given the opportunity to form a text through speech and were asked to fix the errors if present. It was advised to remember and use the inbuilt commands for error correction.

The participants were then asked to share their experience when correcting transcription error and point out the problems they faced using the built-in commands in Google Docs. The feedbacks given were taken into account to better design our system using voice only approach. The feedbacks were based on the following challenges the participants faced.

1. Many commands had to be remembered and getting used to the commands was time consuming.
2. When transcribed text had two occurrences of the same word, the selection of the desired word using the command "Select [word]" would choose the last word in the sentence. For example, if the sentence was *"The quick brown fox jumps over the lazy dog"* and the participant wanted to select the first word "The", saying the command "Select [The]" would select the last word instead. Hence multiple words became hard to edit.
3. Effort had to be put to navigate to locate the word for correction.

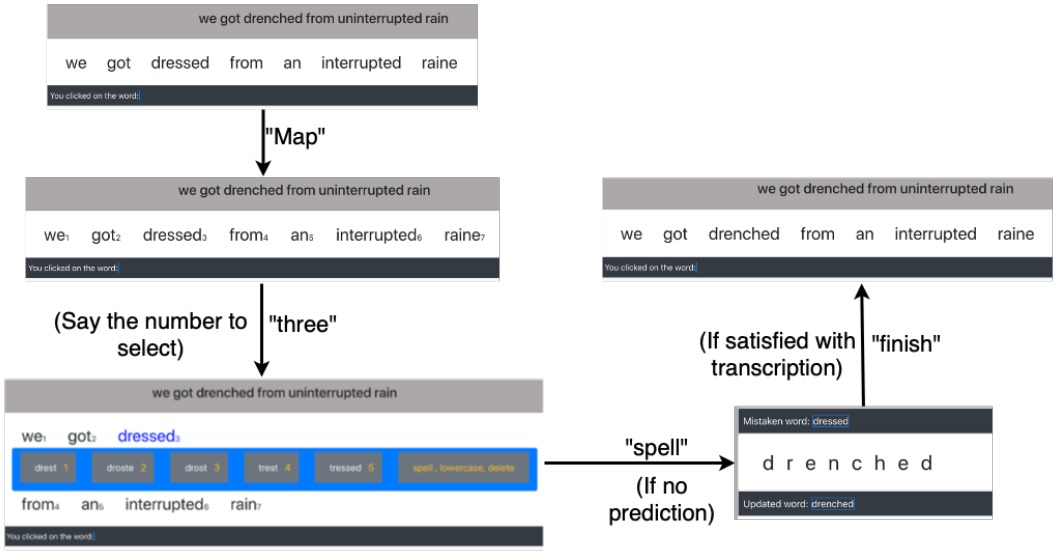
Voice-only approach

We discussed the challenges from the first pilot study and designed an initial implementation for the voice-only approach by introducing the "Map" command. Giving "Map" command during error correction would mark each transcribed word with a number next to it. Participant then gives a command which has to be a "[Number]" for example "2". This would select the second word in the transcribed text allowing further correction for the selected word. Such approach eliminated the issue of locating the error word within the transcribed sentence and also solved the issue of selecting the word when multiple occurrences occurred. Alongside "Map" command, the voice only error correction system we designed also offered list of predictions for a selected word and four additional options.

1. Delete – Giving "Delete" command would remove the currently selected word from the transcription text.
2. Spell – As was often the case where re-speaking the word for correction of a word would make same error, "Spell" command was introduced. It allowed participant to replace the word by a new word that is spelled.
3. Cancel – in order for the user to cancel the current selected word which could have been by error, "Cancel" command was introduced.
4. Case change - given a word with uppercase it was possible to change it all to lowercase using "lowercase" command. Similarly, when the word was lowercase and the user wanted to change the first letter in the word to uppercase, "uppercase" command was used.

Similarly, when using the "Spell" command, participants were taken to spell mode where they were given the flexibility to use "Map" command for letter level correction. Letter level correction allowed to overcome transcription errors caused due to homophones, diction or ambient noise. The detailed workflow of the Voice only approach can be seen in Figure 3.

(a) Workflow of Voice-only approach using available predictions



(b) Workflow of Voice-only approach using "SPELL" mode

Figure 3: Voice-only edit method using “Map” functionality.

4.2.2 Pilot Study II:

The second pilot study was done after our initial implementation for Voice-only approach taking account the initial challenges. Same participants were called upon to give in their feedback for the second pilot study. The participants gave the following feedback after having used our voice-based approach for correcting transcription errors.

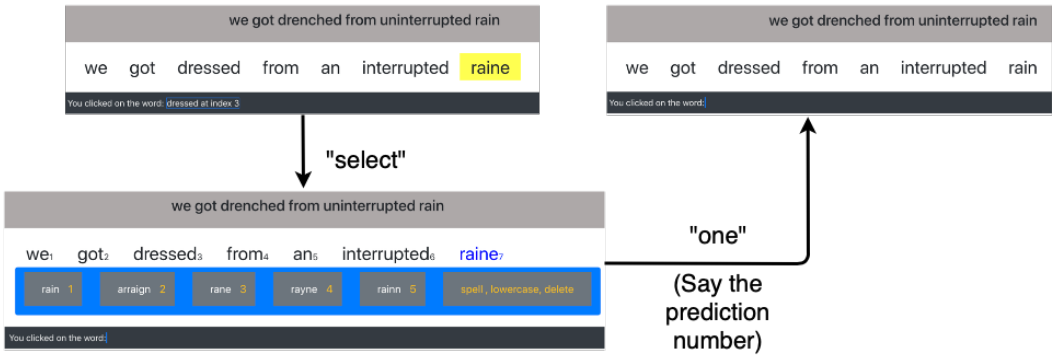
1. Participant were able to navigate to the wrong word quickly compared to when using voice commands in Google Docs.
2. Predictions helped user to correct errors faster.
3. Spell mode as additional mode of correction was very helpful.
4. Having to use "Map" command repetitively for correcting errors was uncomfortable.

Augment Voice with Gaze based approach (TaG)

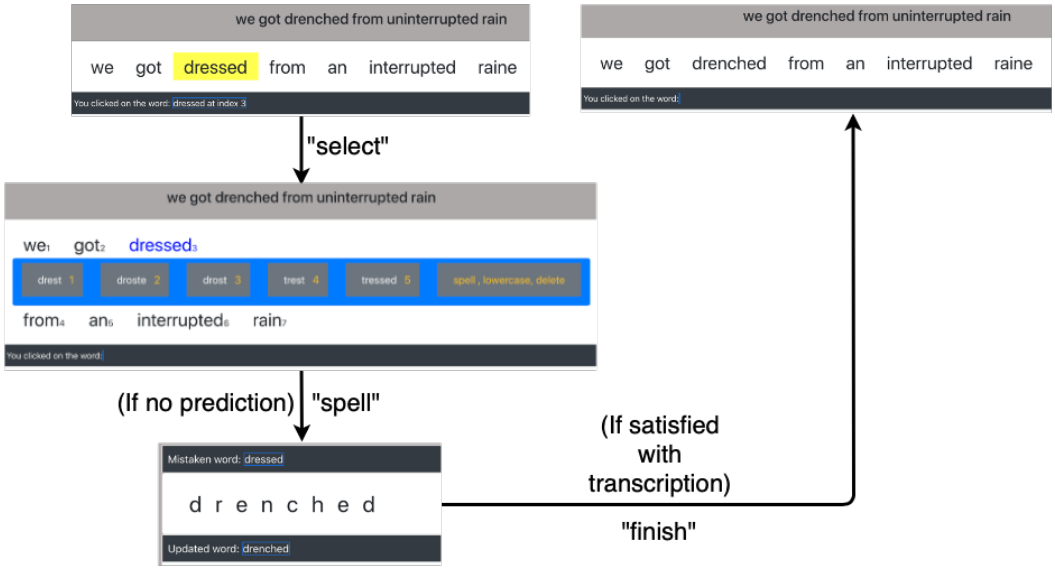
After our first pilot study we introduced voice only approach to correct transcription errors with the map-based command. From the feedback we were able to observe that there was an improvement in navigation while correcting errors. There was however a question raised by participants about an additional step to select the word i.e. needing to first give "Map" command and then again speak the number to identify the error word. Also, the repetitive use of "Map" command was a concern raised during the pilot study I. Hence, to remove the additional step we introduced Gaze alongside Voice only approach (TaG) as a multimodal error correction technique. The implementation introduced selection of the error word by looking at the word for a specific dwell time. This would then select the word eliminating the process involved in voice only approach (unimodal approach) i.e. saying "Map" followed by a number.

The dwell based approach had a drawback of Midas Touch [47]. Users would find themselves clicking on the word that they did not intend to click during error correction. We therefore wanted to examine additional method within our gaze and voice based editing approach. For our TaG method, we studied D-TaG (Dwell TaG) and Command TaG (C-TaG).

I. Dwell TaG (D-TaG) - We design a setup where participants will look at the word for certain time (dwell time of 1 seconds) which triggers the word selection. The selection of word prompts prediction to be displayed for the word. Although this eliminates the use of "Map" command and saying "Number", this process introduces a risk of Midas Touch. The workflow shown in Figure 4shows the usage of spell mode and prediction mode with the dwell time of 1 second.

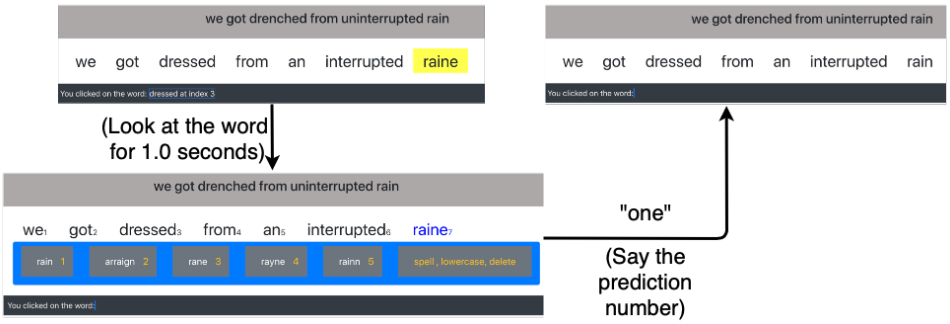


(a) Workflow of Dwell TaG approach using available predictions

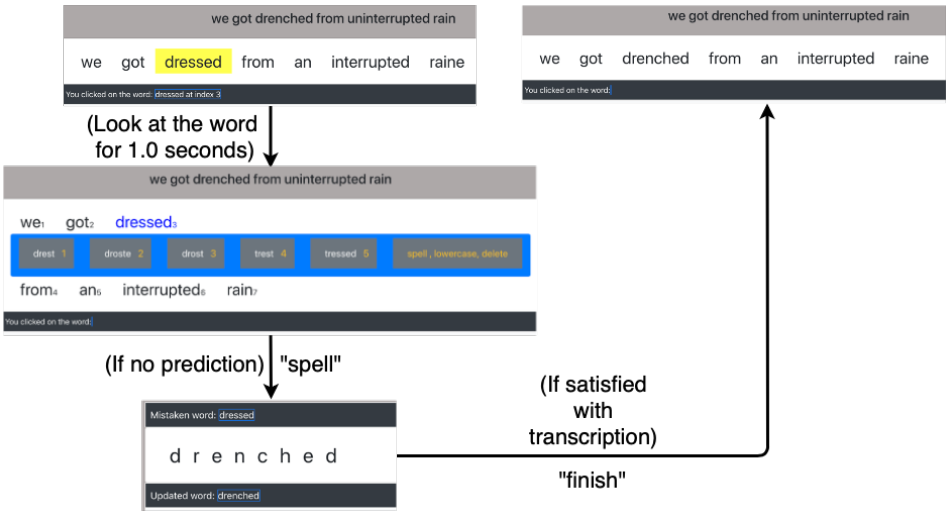


(b) Workflow of Dwell TaG approach using "SPELL" mode

Figure 4: Dwell based Dwell TaG workflow depicting the “dwelling” approach which needs no verbal commands like “map” or “select” for selecting the erroneous word.



(a) Workflow of Command TaG approach using available predictions



(b) Workflow of Command TaG approach using "SPELL" mode

Figure 5: Command TaG workflow showing the use of “select” to confirm the selection of an incorrect word highlighted by gaze.

II. Command TaG - We introduce command TaG(C-TaG) setup to eliminate the Midas touch problem. The idea is to look at the incorrect word in a similar way as Dwell TaG followed by saying a command "Select". This means the inadvertent triggering of word after 1 seconds of dwelling as in Dwell TaG is eliminated. Although it solves the Midas touch problem, we face the problem of additional step in selecting the incorrect word. The workflow for Command TaG is shown in Figure 5.

After our design investigation we implemented three ways of correcting errors across two experimental scenarios. The Voice only map based unimodal approach, Dwell TaG and Command TaG as multimodal approach. From our pilot study I and II we considered each feedbacks and implemented a robust system for further analysis. Let us now look at the our approach for design implementation and the technical implementation.

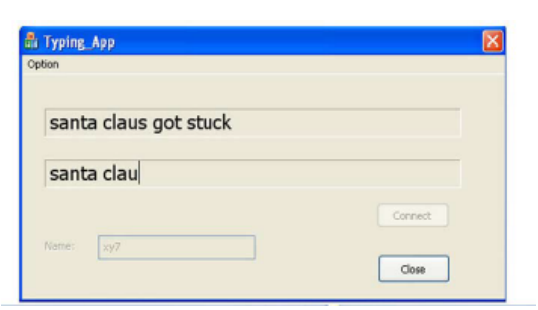
4.3 Our Approach

The task is to build a web platform that facilitates the correction of errors in a hands-free environment for text input. The error correction will be done in a hands-free environment with the help of two modalities (i.e. eyes and voice). We investigate our hypothesis by comparing the performance of voice-based text editing (acting as a baseline) against multimodal text editing. This would allow us to investigate our research objective i.e. observe if the multimodal approach is better than the unimodal approach in text editing.

Two experimental scenarios are developed to investigate error correction in a hands-free environment as described below:

4.3.1 Experiment Scenario I – Read and Correct Task

In this task, participants transcribe given sentences from a list of the predefined set of sentences or phrases using a speech recognition tool. The design and experimental approach described by Ruan *et al.*[40] and Lyons *et al.*[25] will be used where a text to transcribe is given at the top and the result of speech recognition tool is given below as shown in Figure 6. The error that occurs within the transcribed text will then have to be corrected by the user using a unimodal and multimodal approach.



(a) Experimental setup for text entry by hand and mouth by Lyons *et al.*[25]

(b) Experimental setup showing transcribe text on top by Ruan *et al.*[40]

Figure 6: Experimental setup for analysing text entry system through read and correct approach

Initially, the candidate can be under two system setups i.e. using the unimodal approach with voice only or using a multimodal approach with voice and gaze enabled. The system allows for selecting between the two setups, enabling voice only or enabling voice plus gaze.

If in a voice-only mode for this part of read and correct task, the user will be given a valid and robust reference text. The text is obtained from a combination of multiple datasets as described in dataset section to which he/she has to transcribe using the speech recognition

tool. The transcribed text will be presented to the user underneath the original text. Now the user has to initiate a voice command with the "Next" keyword suggesting the transcribed text is correct. Each possible voice commands for our system is shown in Table 1.

	Description	**Actions**	**Commands**
User Input	User willing to transcribe text	- Start Listening	- "Start"
		- Stop Listening	- "Stop"
		- Reset Phrase	- "Reset"
		- Next phrase	- "Next"
Edit Mode	Change from typing mode to edit mode	- Activate Edit Mode	- "Edit"
Mapping each word	When edit mode is on	- Map each word to a number	- "Map"
Selecting mapped word	When words are mapped to numbers	- Select a word	- "Select *<number>*"

Table 1: Voice based commands

If, however, the error is present, the user will use a voice-based command to locate to the point of error. Once located, the list of a word appears underneath the error word with a list of alternate suggestions as shown in Figure 7. The suggestions of words would be given by the system after analyzing and considering homophones, synonyms, phonemes, language models. Datamuse and WORDS API's discussed in section 4.3 will be used to obtain suggestion words from each of the entities: homophones, synonyms, phonemes, and language models.

In addition, the system will make use a multimodal approach using both voice and gaze. The distinction from voice-only approach would be that gaze would be used to locate to the point of error. Each selection would also be performed using gaze in combination with voice commands.

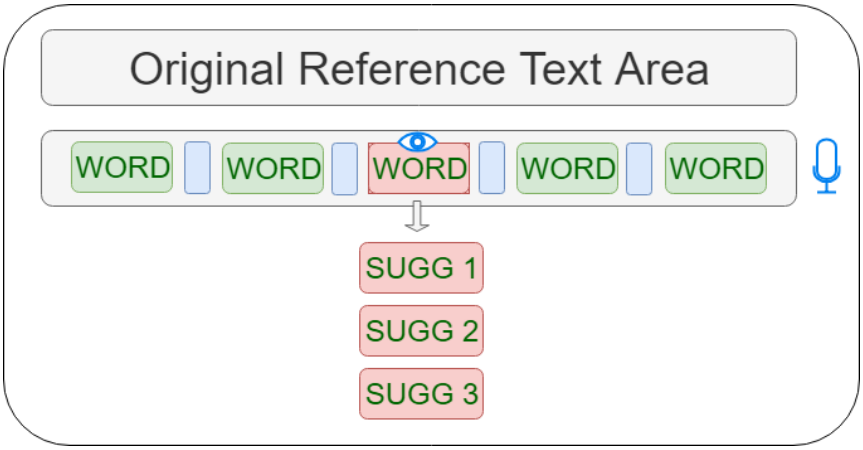


Figure 7: Experimental setup with original reference text

4.3.2 Experiment Scenario II – Image Description Task

Dunlop *et al.* [12] argues that evaluating text entry and text editing requires free-form text entry that is not based on tradition transcription/copy tasks. They report that the approach of a fixed phrase copying provides internal consistency but lacks representativeness in natural text entry systems. We take an approach described by Dunlop *et al.* [12] which involves image description task. The idea involves providing an image and asking users to describe the image within a fixed amount of time. The flexibility for users to conceptually form a text into a large scrolling text field allows us to create a realistic scenario for our experiment.

Thus for our image description experiment, we will have a slightly different user interface whereby the user creates a document of his choice depending on the image provided. The user will have to construct a text describing the image provided as explained in section 3.1 of our approach. Here the user will have no prior knowledge of image and will be instructed to follow certain conditions set by the system. Dunlop *et al.* [12] outlines the similar approach of image description task. They discuss the approach of facilitating user to freely compose text creating a realistic scenario. The experimental setup we designed takes a similar method of showing the image at the top and allowing the user to transcribe text below the image as shown in figure 8.

In addition, we intend to design commands as can be seen in Table 1 for interactive correction using eye and voice in hands-free editing scenarios discussed above. With gaze, we would use an eye-tracking device to locate the point of error and simulate the movement of the cursor. The commands in Table 1 for voice would be used in combination with gaze for interactive correction of error. This will help us to analyze both unimodal approaches of correcting errors as well as multimodal hands-free editing.

The speech recognition and transcription process will follow in a similar manner as for

the first experimental scenario. An interactive correction will also occur in the same manner as explained in the workflow section in implementation chapter. Again, the evaluation will be done for block completion time, error correction time and uncorrected errors.

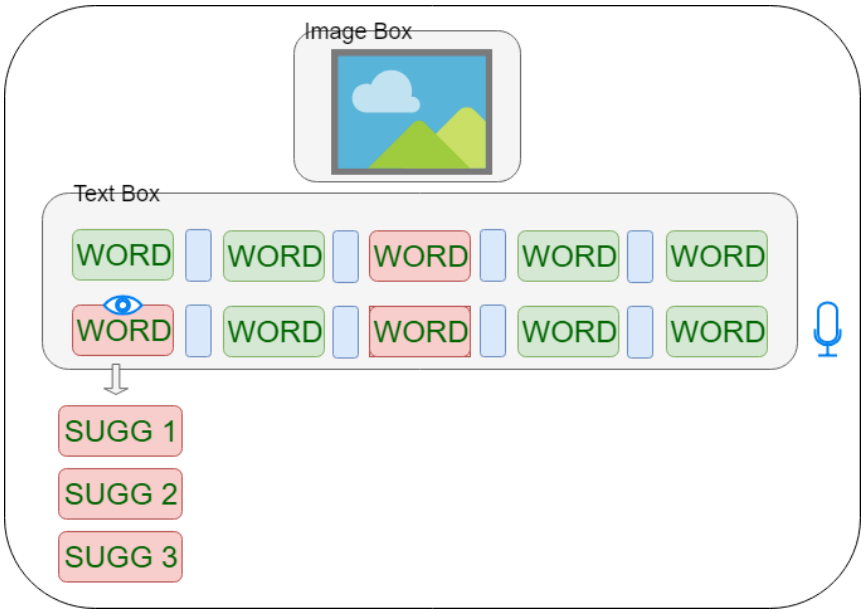


Figure 8: Experimental setup for image description task

4.4 System Implementation

We performed technical implementation on the planned experimental setups discussed during our design investigation. The programming language best fit for our system was JavaScript language and underlying framework was ReactJS in a NodeJS platform.

JavaScript was used to build an interface for hands-free interactive correction process. It was also preferred language choice for making API calls within chrome browser for using speech recognition tool. Integrating with third party wrappers for eye tracking device was done in NodeJS platform which was built on Chrome's JavaScript runtime. The actions and functionalities for interactive correction process was done using gaze or voice command or both. The intuitive design was built using CSS and ReactJS framework for JavaScript. We therefore analysed the workflow for our entire error correction process and discussed each technical aspect required for designing our system.

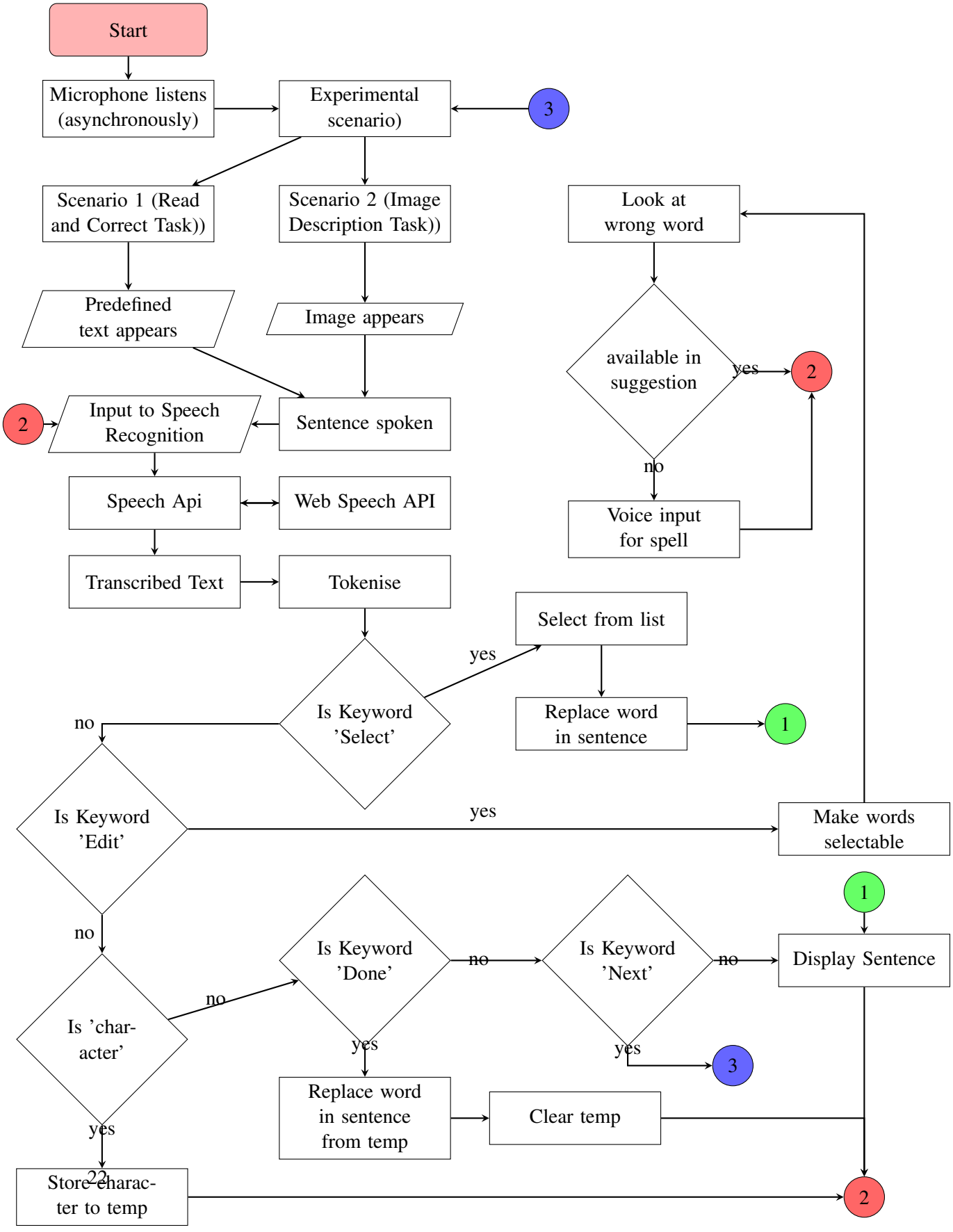
4.4.1 Workflow

Our multimodal error correction process includes speech recognizer for continuous speech and spelled letters recognition. Eye tracker device captures eye movement which was used within the interface. The flowchart below shows our multimodal error correction process.

In interacting with our platform, the user will be assigned to perform the task in one of our two experimental scenarios i.e. 1) Read and correct task and 2) Image description task. For each of the experimental scenario, the user provides a primary input using voice to the microphone that is listening asynchronously. The speech gets sent to Google's web API returning the automatically interpreted text which is displayed on the screen. All voice input will be displayed onto the screen until predefined command such as "edit", "next", "select" [refer to list of commands in chapter 3] etc is detected. As the user observes the text in the screen, he/she will now be able to realize if transcription error has occurred. In case of an error, the user can now provide voice input to the asynchronously listening speech recognition tool for "Edit" command ("Is keyword Edit" in the flowchart). As the system detects the command "Edit", interactive error correction mode gets activated to recover the error.

For interactive error correction, the exact location of the error within the inputted text is to be determined. This is done using gaze modality where the user looks at the wrong word and the system identifies the location of an error. After the error is identified and located, the user will be given a list of the suggested word for replacement. If the list contains the intended word for correction, the user uses voice to select the word and gives the command [Number] to select the word. If, however, the list does not contain the intended word, the user will have an option to spell out each character for the misrecognized word ("Voice input for a spell" shown in the flowchart below). Finally, the repaired context gets updated and will be displayed to the user.

Furthermore, once the editing task is complete and the user sends the "Finish" command, the edit mode changes to text entry mode. If now the error correction is complete and transcribed text is accepted (i.e. with command "Next"), no repair is done, and the next set of tests is to be carried out. This process continues until the user completes all task assigned.


Start
Microphone listens (asynchronously)
Experimental scenario)
3
Scenario 1 (Read and Correct Task))
Scenario 2 (Image Description Task))
Look at wrong word
Predefined text appears
Image appears
available in suggestion
yes
2
no
2
Input to Speech Recognition
Sentence spoken
Speech Api
Web Speech API
Voice input for spell
Transcribed Text
Tokenise
Select from list
yes
Is Keyword 'Select'
Replace word in sentence
1
no
Is Keyword 'Edit'
yes
Make words selectable
1
no
Is Keyword 'Done'
no
Is Keyword 'Next'
no
Display Sentence
Is 'char-acter'
no
yes
yes
3
Replace word in sentence from temp
Clear temp
yes
Store charac-ter to temp
2

4.4.2 Software architecture

The experiment in this paper compares the use of Voice only, Gaze plus voice (Dwell TaG and Command TaG) to correct transcription errors. As shown in the workflow, read and correct task uses one interface layout while image description task uses modified interface layout. Both interfaces however do not differ much, and the underlying correction mechanism remains the same. Therefore, we took a generic approach on implementing the web application following one architectural design.
Our design approach will follow the architecture shown in Figure 9

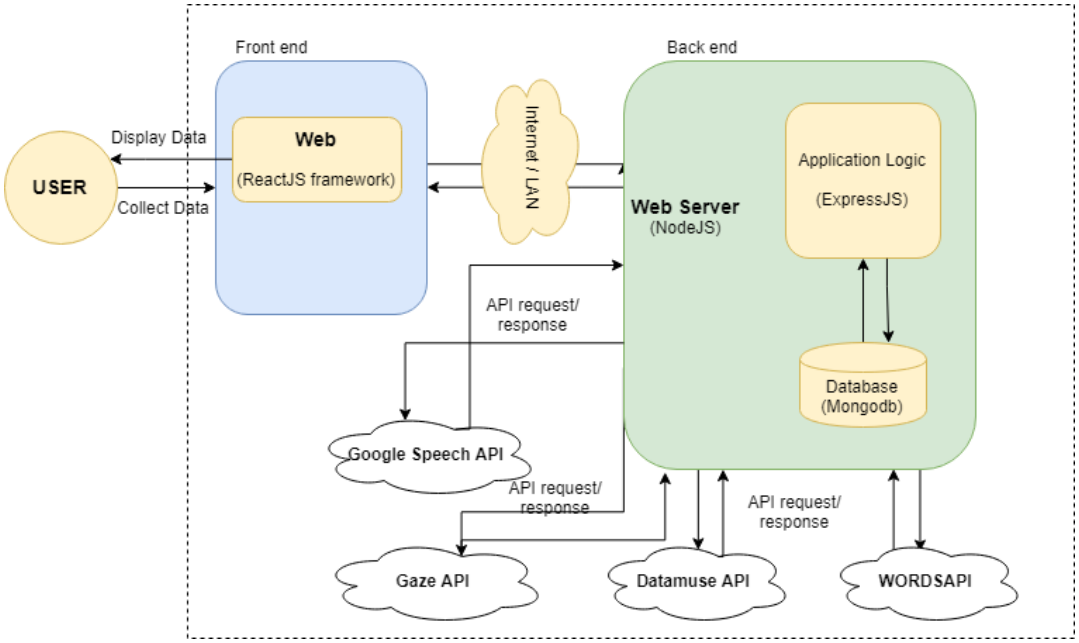


Figure 9: Architectural design for the web.

Web Application
We will use **Node.js** which provides a runtime JavaScript execution engine for building our web application. Similarly, **Express.js** being a Node.js web application framework providing various HTTP utility methods, we make use of it extensively in our project for third-party API calls.

As we have a back-end service up and running through the node and express implementation. We need a user interface for interaction with the client. For this we use **React.js** which is a JavaScript library for building user interfaces.

API's used in suggestion list Once the error word is located we will send that word to API's such as datamuseAPI and WORDSAPI to obtain a list of alternate words depending on the queries we set. For example, we retrieve words for similar sounding words or synonyms, or rhyming words. We obtain a list of alternative suggestion list depending on the intersection of results obtained from each API's. We will then analyze if it improves the editing process.

- Datamuse API: Allows word-finding for a given query where queries can be constrained with meaning, spelling, sound, and vocabulary.
- WORDSAPI: Allows us to retrieve details of a word including the similar sounding words, synonyms, rhyming words, and different variations in pronunciations.

4.4.3 Speech recognition component

As in Figure 10, we can see the folder structure for the implementation of our system. We can realize a common *SpeechRecognition* class which is shared as component across all tasks i.e. image description task and read correct task. Each task has its own JavaScript class which extends *SpeechRecognition* base class for enabling voice input. Similarly, user interface design is under *homePage* folder which implements multimodality and voice only.

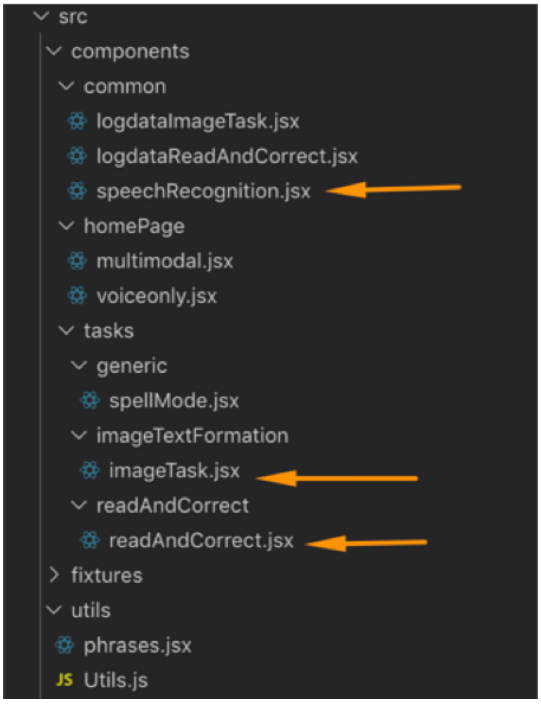


Figure 10: Folder structure for web application in ReactJS framework

In Figure 11, we can see how SpeechRecognition functionality is implemented using the browsers inbuilt webkitSpeechRecognition tool. It also a returns a wrapper around the component which enables another component such as *imageTask* and *readAndCorrect* to be rendered.

Although incorporating voice into our experiments was done using the inbuilt browser speech recognition tool, integrating with gaze required further work. Let us now discuss our approach to combining gaze with voice into our experiments.

```
export default function SpeechRecognition(options) {
  const SpeechRecognitionInner = function(WrappedComponent) {
    const BrowserSpeechRecognition =
      typeof window !== "undefined" &&
      (window.SpeechRecognition ||
        window.webkitSpeechRecognition ||
        window.mozSpeechRecognition ||
        window.msSpeechRecognition ||
        window.oSpeechRecognition);
    const recognition = BrowserSpeechRecognition
      ? new BrowserSpeechRecognition()
      : null;
    const browserSupportsSpeechRecognition = recognition !== null;
    let listening;
    if (
      !browserSupportsSpeechRecognition ||
      (options && options.autoStart === false)
    ) {
      listening = false;
    } else {
      recognition.start();
      listening = true;
    }

    return class SpeechRecognitionContainer extends Component {
      constructor(props) {
        super(props);

        if (browserSupportsSpeechRecognition) {
          recognition.continuous = options.continuous !== false;
          recognition.interimResults = true;
          recognition.onresult = this.updateTranscript.bind(this);
          recognition.onend = this.onRecognitionDisconnect.bind(this);
        }
      }
    }
  }
```

Figure 11: Code snippet showing the implementation of voice as input using chromiums inbuilt speech recognition tool

4.4.4 Gaze interaction

As our experiments required gaze interaction with our system, we required to firstly track users' eyes and secondly to allow eyes to interact with the system. Our approach was to track users' eyes and as per eye movement we would bind eye point with mouse cursor. This meant moving eyes in the screen would move mouse cursor in the screen. The interaction had to feel more real and thus we chose Tobii 4C eye tracking device for effortless gaze point tracking as per user's instincts and intentions. The illuminators and sensors used for calculating gaze point from Tobii 4C allowed us to manipulate mouse movement as per eye movement.

We got Tobii 4C to track users' eyes on the computer screen. The mouse cursor on the computer screen had to be linked with the Tobii 4C gaze locator which was not inbuilt with the device software. This meant we had to bind the mouse movement with the gaze device so that we could allow users to perform mouse movement as per our experimental requirement. For this reason, we chose Project IRIS SDK which was a C++ wrapper that did the binding between the Tobii 4C eye tracker and the mouse cursor on the computer. The following gives the architectural sense of the design used.

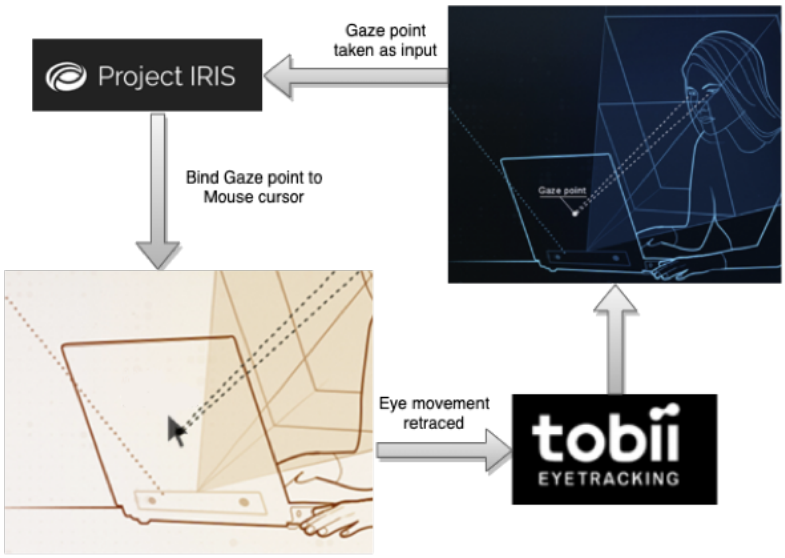


Figure 12: Bind gaze point to mouse cursor using project IRIS tool

Although we were able to bind gaze point to the mouse cursor of the computer screen, we required to further manipulate our application in order to click the mouse cursor via our web applications. The problem we faced is that web applications cannot directly perform a mouse click outside of its domain i.e. application running on chrome browser due to security issues. Therefore, for us to trigger a mouse click outside of web browser and directly to our computer operating system we had to create a NodeJS application that would leverage slack and Auto-hotkey tool to trigger a mouse click event. The slack integration needed is discussed below.

4.4.5 Slack integration

As can be seen in Figure 13, we create a NodeJS application which creates an instance on Slack RTM client that connects to the Slack application via an API token obtained from Slack messaging service. To obtain the API token one requires to create a channel in slack which has a binding to one of the listeners registered by Slack RTM client instance. This means that when our NodeJS application receives a slack message, it triggers a subprocess to run an Auto-Hotkey executable which in turn triggers a mouse click.

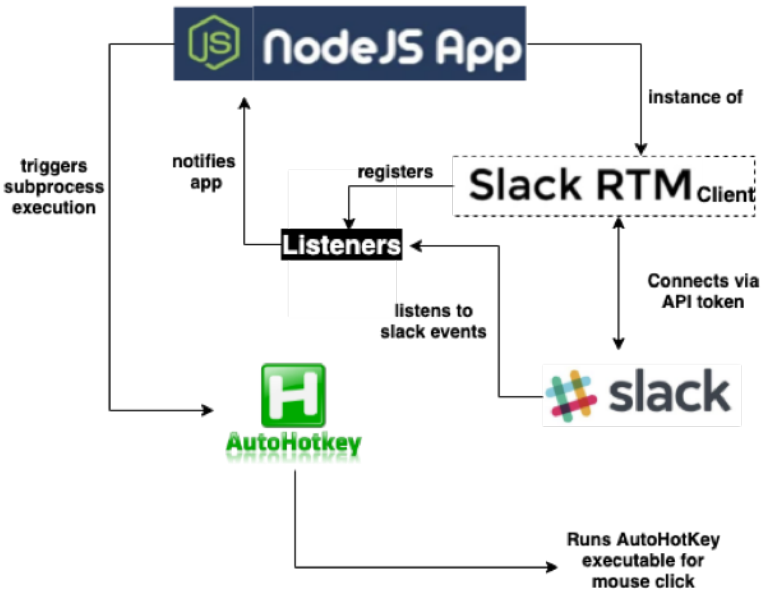


Figure 13: Bind gaze point to mouse cursor using project IRIS tool

5 Evaluation

We evaluated a system built for text editing under two setups i) unimodal approach on error correction using voice only ii) multimodal approach on error correction using voice and gaze together. The study involved two tasks performed under controlled environment to evaluate error correction system using Dwell TaG, Command TaG and Voice only approach. In this section we discuss the participants involved, apparatus used, procedure for our experiment, and dataset used.

5.1 Experimental setup

The experiment took place in the research laboratory at the University of Koblenz-Landau, Campus Koblenz. As per the experimental setup, participants were asked to sit in front of a 24-inch monitor which had Tobii Eye Tracker attached at the bottom. An adjustable chair was provided to adjust their positioning at which point eye tracker was calibrated per user. Similarly, a microphone was positioned on the side of the monitor for tracking voice as shown in Figure 14.

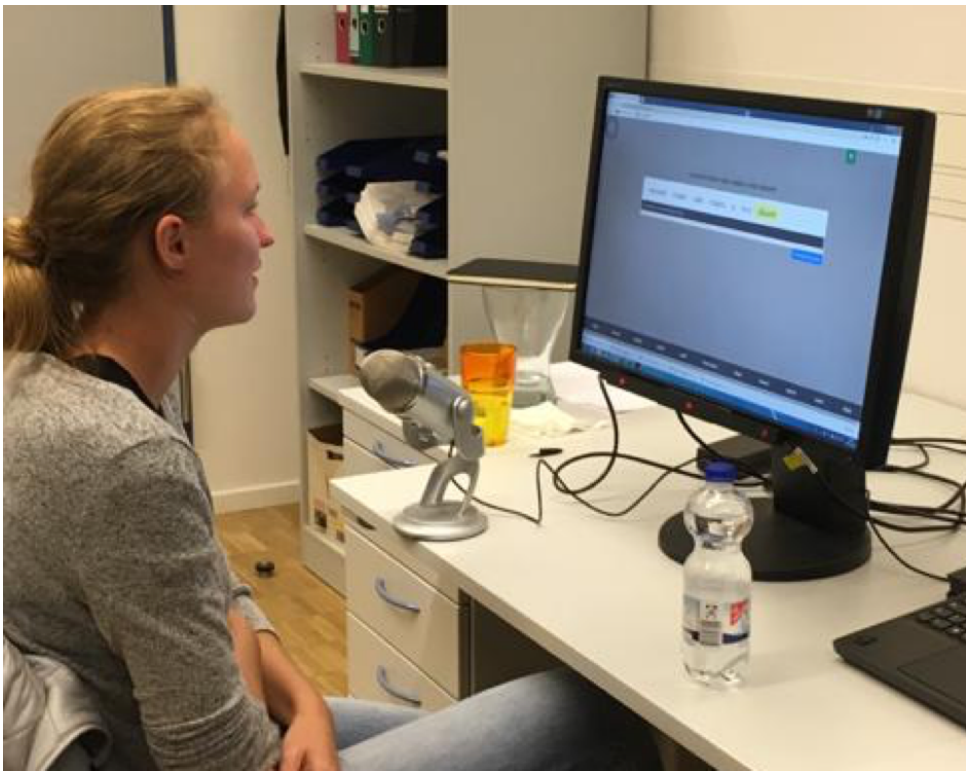


Figure 14: Experimental setup showing participant using microphone for Voice input and Tobii Eye Tracker 4C for Gaze input to correct errors in a controlled environment.

5.2 Participants

A total of 10 participants performed two different tasks i) Read and Correct Task ii) Image description task under three experimental setup i) Voice only ii) Command TaG and, iii) Voice only on different days. Each participant performed the error correction experiment according to the Latin Square Ordering scale. Using this scale, we had participant allotted to different slots in randomized order to reduce any bias that may have arose during our experiment.

All participants volunteering for our study were either university graduates or degree students (6 male, 4 females, ages 20- 32) . Each participant had good competency in English and were therefore familiar with the words used in sentences for transcription. Four participants had prior experience with using eye tracking device but were not familiar with using voice and gaze together.

5.3 Apparatus

For gaze input experiments, we used Tobii Eye Tracker 4C [6] shown in Figure 15 to track and collect gaze data. For voice input experiments, we used Google chromes inbuilt webkit speech recognition tool[7]. All sessions within the experiment was recorded using the Open Broadcaster Software (OBS)[8] for further data evaluation. The experiments were conducted in a controlled environment with minimal disturbance and controlled ambient light as shown in Figure14. The system was built using Javascript and React JS framework for collecting user performance data and was stored in a *.csv* file.

Figure 15: Tobii Eye Tracker 4C

[6] https://gaming.tobii.com/tobii-eye-tracker-4c/
[7] https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API/Using_the_Web_Speech_API
[8] https://obsproject.com/

5.4 Procedure

As per the experiment procedure, participants were asked to sign the consent form prior to the experiment followed by the detailed explanation of the study. For each of two tasks, participants were shown how the system works and were asked to position themselves in front of the screen so as to perform eye tracking calibration. Calibration for eye tracking was done using Tobii 4C calibration software which used six calibration points. After calibration, participant performed training on the system for the first two blocks followed by the real experiment on remaining blocks. Participants were made aware that for each of the task, their gaze data was collected for evaluation purpose even when gaze was not required for certain experiment. Breaks were given in between, and calibration was done again in case participant had to move from their position. After each block we downloaded the recorded data in a *.csv* file. When participant finished specified task, they were given to complete NASA TLX questionnaire and SUS questionnaire.

For each of two tasks, 10 participants participated across three edit methods i) Voice only ii) Dwell TaG and, iii) Command TaG. For Read and Correct task, 12 blocks of which 2 were training blocks were to be completed. Each block consisted of 5 sentences to be transcribed and corrected. The sentences presented (discussed in dataset section below) to the participant had to be transcribed as shown in Figure16 and if any error, it had to be corrected.

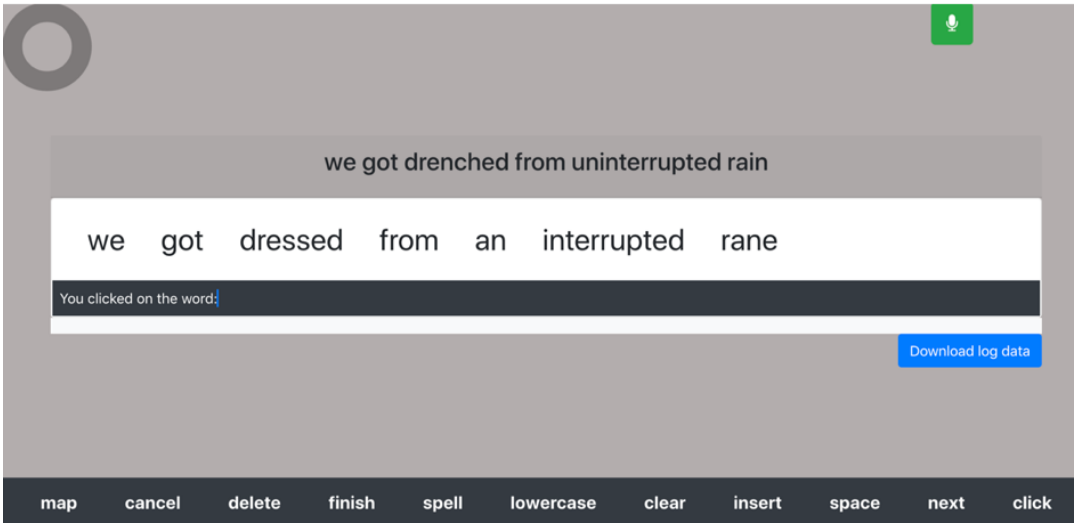


Figure 16: Read and Correct Task: Participants transcribe the given text and correct when error is present.

Similarly, for Image Description task, we had 5 blocks each with three images but only two had to be selected, described and corrected as shown in Figure 17. The dependent variables were block completion time in seconds, error correction time (spell and suggestion mode) in seconds and uncorrected errors (count).

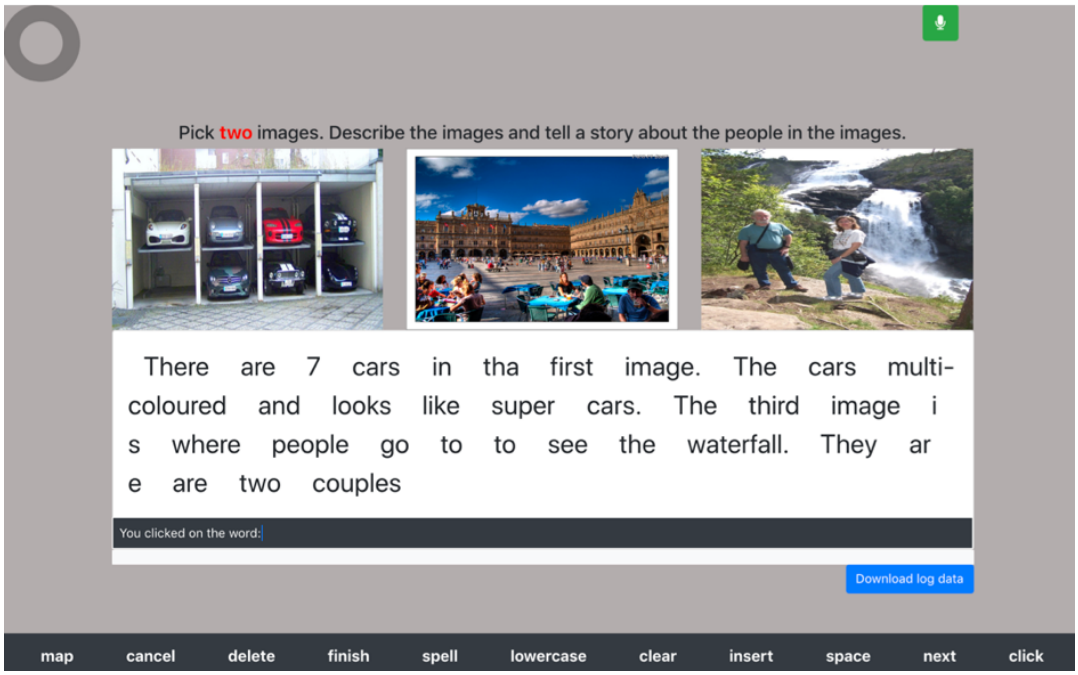


Figure 17: Read and Correct Task: Participants transcribe the given text and correct when error is present.

5.5 Dataset

During the study, we will make use of three different established datasets for each of our experimental scenarios 1) Read and Correct task [15] and 2) Image Description task[22].

For our experimental scenario 1, we will use Acoustic-Phonetic Continuous Speech dataset by Garofolo *et al.*[15] as it has been specifically designed for speech to text entry experiments. Acoustic-Phonetic Continuous Speech Corpus contains recordings of phonetically-balanced prompted English speech. Phrases from such text corpus would be useful for our experiment as we observe misrecognition from speech recognition tools due to phonetic complexity in the phrase.

Similarly, for experimental scenario 2 - Image Description Task, we will use the image task set used by Dunlop *et al.* [12] in their research.

5.6 Quantitative Evaluation

To evaluate the performance of the handsfree editing system using voice and gaze, we discussed empirical measures of text editing performance [44]. Since transcribing of text alongside editing of text will be character based, the measures were focused on characters. We evaluated i) Block completion Time ii) Uncorrected errors iii) Error Correction Time under Spell and Suggestion mode as a measure of effectiveness and efficiency measures for error correction. In this section we will therefore discuss each measurement metrics, their technical implementation and the observed results in two experimental scenarios i) Read and Correction Task ii) Image description Task with three edit methods i) Voice only ii) Dwell TaG and, iii) Command TaG.

5.6.1 Read and Correction Task

For this task of reading the given text and transcribing it, participants were required to correct the error present during transcription. Twelve blocks with each block with 5 sentences were part of the experiment. Each measurement metric for quantitative evaluation was considered when using different edit approaches. Overall, Dwell TaG and Command TaG showed low average error correction time and a smaller number of uncorrected errors compared to unimodal voice only approach. For our experimental data the first two blocks were used for training and has not been used for evaluation.

5.6.1.1 Block completion time

We calculate the block completion time for overall time taken to have transcribed and corrected 5 sentences. For each sentence we log the "Start" time and the "End Task" time. Thus, we subtract the "End Task" time against the "Start" time to evaluate the sentence completion time. We then sum time for all five sentences to obtain the block completion time. We have a total of 12 Blocks for which 2 are for training. Block completion time was evaluated across all three method of error correction i) Voice only ii) Dwell TaG iii) Command TaG as discussed.

Before evaluating the result let us look at the technical implementation required. As per implementation we first obtained the time in string format which was then parsed to obtain the time difference in seconds. The Figure 18 code snippet shows the implementation in JavaScript.

```
function helperTimeDifferenceInMilliseconds(strTime, endTime) {
    if (strTime && endTime) {
        let hmsm = strTime
        let hmsm2 = endTime
        let a = hmsm.split(':'); // split it at the colons
        let b = hmsm2.split(':');

        let milliseconds = ((b[0] - a[0]) * 60 * 60 * 1000) +
                           ((b[1] - a[1]) * 60 * 1000) + ((b[2]
                              - a[2]) * 1000) + (b[3] - a[3])
        return milliseconds / 1000
    } else {
        return 0;
    }
}

// Given array of start time and end time, calculates duration for each value in the array
function calculateDuration(startArr, endArr) {
    let array1 = startArr
    let array2 = endArr

    let durationArr = array1.map(function (num, idx) {
        return helperTimeDifferenceInMilliseconds(num, array2[idx]);
    });
    return durationArr
}
```

Figure 18: code snippet for calculating the difference between the start and end time

With the generic function implemented to calculate block completion time, we were able to obtain statistical data for further analysis.

Measurement of block completion time was done to check the complexity in task completion across multimodal and unimodal approach of correcting errors. With Shapiro-Wilk test[33] showing that our data was normally distributed, we performed ANOVA test to check for statistical significance. ANOVA showed no statistical significance at ($F_{2,27}$ = 3.11, p = .061) for block completion time.

Additionally, average block completion time using Voice-only was 354.41 seconds, using Dwell TaG was 294.96 seconds and Command TaG was 315.95 seconds. This shows that Dwell TaG and Command TaG (multimodal approach) performed better than Voice only (unimodal approach). From Figure 19 except for block 10 and block 12, Command TaG and Dwell TaG always has lower median value for block completion time than Voice only approach. Voice only has the max block completion time of 840 seconds while Command and Dwell TaG reached maximum of 700 seconds.

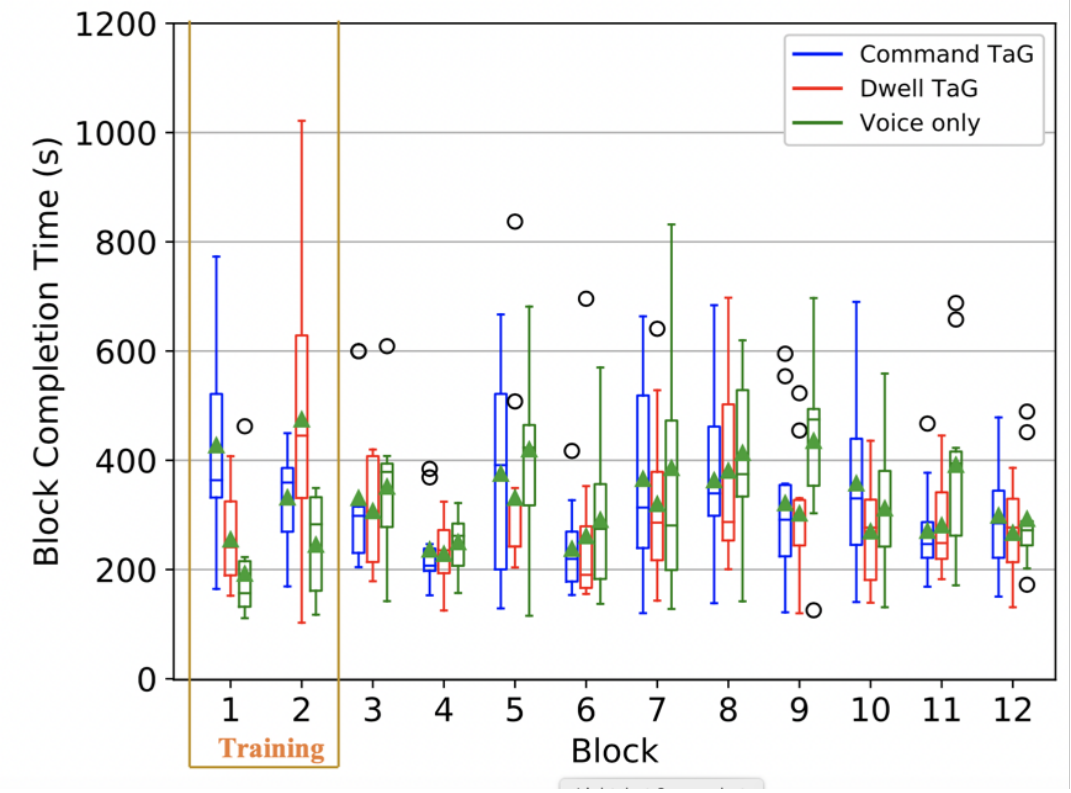


Figure 19: Average block completion time for Dwell TaG, Command TaG and Voice only input methods

5.6.1.2 Error correction time

We evaluate the time taken to replace the error word with the correct word across the transcribed sentence. We start the time when the error word is selected until the error word is replaced. Same function was used as described in Block completion time to calculate the duration of the error correction.

Shapiro-Wilk test was taken for our sample data which showed the data was normally distributed. ANOVA was then performed on the total error correction time for all the input methods. The result was statistically significant across 10 Blocks ($F_{2,33} = 31.97$, $p < .001$).

As can be seen in Figure 20, time taken for correcting errors using Dwell TaG is faster compared to Command TaG and Voice only. Evaluating 10 blocks gave an average error correction time of 8.11 seconds, 11.34 seconds and 14.9 seconds for Dwell TaG, Command TaG and Voice only respectively. For each block, Voice only approach took longer time to correct errors with the maximum of 27 seconds. Dwell TaG performed better than Command

or Voice with error correction time as low as 4 seconds.

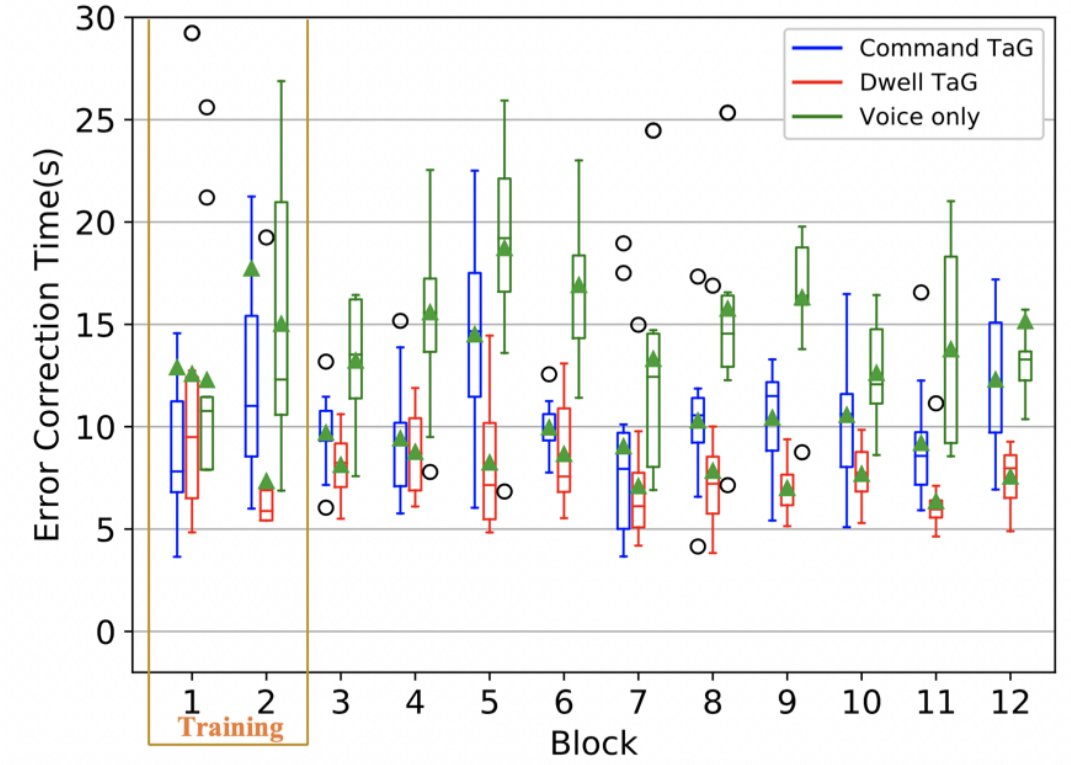


Figure 20: Box and whisker plot showing the error correction time for Dwell TaG, Command TaG and Voice only

Total time taken to correct errors was further broken down into two modes of correcting transcription errors. 1) Spell mode: time taken to correct error by spelling out each character 2) Suggestion mode: time taken to select the correct word from the list of suggestion words. Error correction time when participant used each of the correction mode was analysed.

5.6.1.2.1 Suggestion mode We measure time taken to select the word from the suggestion list to have the error word replaced. It also consisted of case changes i.e. lowercase and uppercase. If in suggestion mode, we log how many times selection from the list was done throughout the sentence correction process by setting a counter which was used for calculation.

We evaluated data obtained while suggestion mode was used i.e. when transcription error was observed, participant selected the correct word from the list of suggestion list. It was observed that this approach was a faster process in correcting errors where participant only spent 6.66 seconds in average for correcting errors mainly because the correct word within

suggestion list relied mostly on better prediction model. We observed that participant spent less time in suggestion mode for correcting transcript errors with Dwell TaG contributing 6.16s, Command TaG with 5.20s and Voice with 8.63s. This suggests that suggestion mode was faster than spell mode approach but at the same time indicates multimodal approach being better than unimodal approach as shown in Figure 21.

Shapiro-Wilk test was taken for our sample data which showed our data was normally distributed. ANOVA was then performed on the total error correction time for Dwell TaG, Command TaG and Voice only approaches. This result was statistically significant across 10 Blocks ($F_{2,27} = 86.13$, $p < .001$).

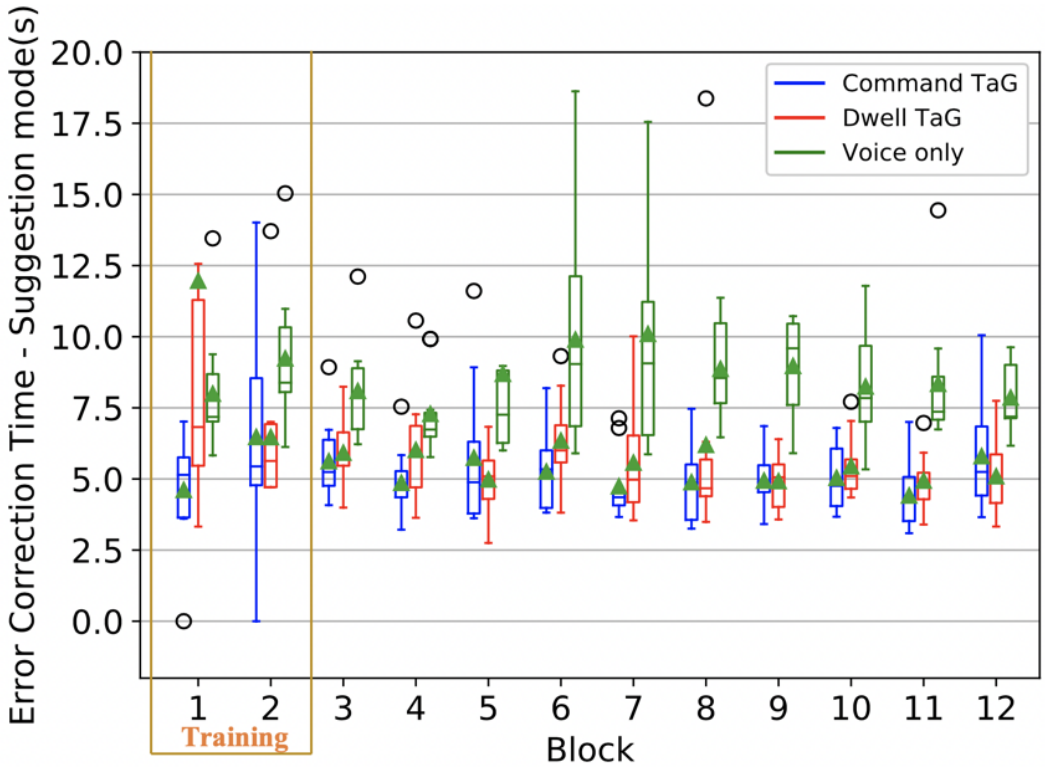


Figure 21: Error correction time for Dwell TaG, Command TaG and Voice only when using Suggestion mode

5.6.1.2.2 Spell mode In case the suggestion list does not have the correct word, we activate spell mode. We measure the time taken to complete spelling out each character representing the correct word. Similarly, if in spell mode, we set counter to evaluate the number of times the spell mode was activated. The distinction of these editing modes is through voice commands i.e. command "Spell" would trigger spell mode.

We presented error correction time under multimodal and unimodal approaches when participants chose spell mode in correcting transcription errors. We observed spell mode took longer time to finish compared to suggestion mode with average error correction time of 19.46s using Command TaG, 12.81s using Dwell TaG and 23.36s using Voice only. The Shapiro-Wilk test was taken for our sample data which showed our data was normally distributed. ANOVA was then performed on the total error correction time for Dwell TaG, Command TaG and Voice only approaches. This result was statistically significant across 10 Blocks ($F_{2,27} = 34.88$, $p < .001$).

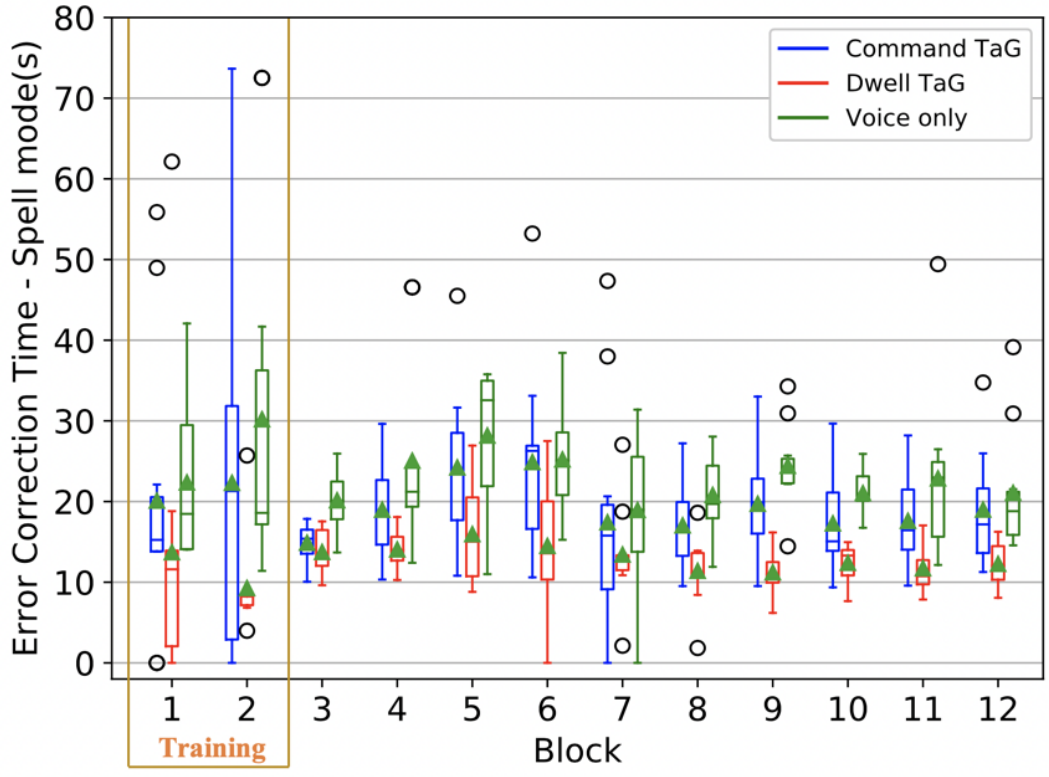


Figure 22: Error correction time for Dwell TaG, Command TaG and Voice only when using Spell mode

In Figure 22 we can see that the median value for command TaG and Dwell TaG was always lower across all blocks compared to that with Voice only. Although five blocks for Command TaG shows maximum value higher than Voice only, upper quartile range for Command TaG is lower than upper quartile of Voice only. Q3 for Dwell TaG is always lower than both Voice only and Command TaG telling us multimodal approach took relatively less correction time when in spell mode. Figure 22 also suggests our hypothesis of multimodal approach (Dwell TaG, Command TaG) being better compared to unimodal (Voice only) approach for correcting transcription errors with unimodal approach (Voice only) taking maximum of 38 seconds.

5.6.1.3 Uncorrected Error

The uncorrected errors will be used as a mechanism to measure the performance of editing. We compared a reference text to the transcribed text and evaluated 3 entities.

- Substitutions(S) i.e. the number of replacements
- Deletions (D) i.e. the number of removals
- Insertions (I) i.e. the number of additions

With these entities calculated, we could formulate Levenshtein distance.

Levenshtein Distance

The Levenshtein distance is a string metric used in determining the difference in two sequences[46]. As an example, one can obtain the Levenshtein distance between two words which evaluates how many single-character edits (insertions, deletions or substitutions) have to be done for a word to change into the original word.

Mathematically, the Levenshtein distance between two strings a and b (of length |a| and |b| respectively) is given by

$$lev_{a,b}(|a|, |b|)$$

where

$$lev_{a,b}(i,j) = \begin{cases} max(i,j), & \text{if } min(i,j) = 0, \\ min \begin{cases} lev_{a,b}(i-1,j) + 1 \\ lev_{a,b}(i,j-1) + 1 \\ lev_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)}, \end{cases} & \text{otherwise,} \end{cases}$$

$$where 1_{(a_i \neq b_j)}$$

is the indicator function equal to 0 when

$$a_i = b_j$$

and equal to 1 otherwise, and

$$lev_{a,b}(i, j)$$

is the distance between the first i characters of a and the first j characters of b.

JavaScript language was used in a NodeJS development platform to create a generic function that calculates Levenshtein distance between two sentences. The code snippet shown in Figure 23 uses the above-mentioned mathematical approach for the calculation.

```
// Compute the edit distance between the two given strings
getEditDistance = function (originalSetence, lastSentence) {
    if (originalSetence.length == 0) return lastSentence.length;
    if (lastSentence.length == 0) return originalSetence.length;

    let resultMatrix = [];

    // increment along the first column of each row
    let i;
    for (i = 0; i <= lastSentence.length; i++) {
        resultMatrix[i] = [i];
    }

    // increment each column in the first row
    let j;
    for (j = 0; j <= originalSetence.length; j++) {
        resultMatrix[0][j] = j;
    }

    // Fill in the rest of the resultMatrix
    for (i = 1; i <= lastSentence.length; i++) {
        for (j = 1; j <= originalSetence.length; j++) {
            if (lastSentence.charAt(i - 1) == originalSetence.charAt(j - 1)) {
                resultMatrix[i][j] = resultMatrix[i - 1][j - 1];
            } else {
                resultMatrix[i][j] = Math.min(resultMatrix[i - 1][j - 1] + 1, // substitution
                    Math.min(resultMatrix[i][j - 1] + 1, // insertion
                        resultMatrix[i - 1][j] + 1)); // deletion
            }
        }
    }

    return resultMatrix[b.length][originalSetence.length];
};

```

Figure 23: Code snippet for calculating the Levenshtein distance

We used Levenshtein distance between the original sentence and the final sentence after error correction to evaluate the uncorrected errors for a task. As an example, we can see the data in Figure 24 that was collected from one of the participants and Levenshtein distance between the original sentence and the first transcription was evaluated.

	OriginalTranscription	FirstTranscription	Levenshtein (Original - First)
1	destroy everyfile related to my audits	destroy every file relented to my orders	7
2	they enjoy it when I audition	they enjoy it when I edition	2
3	employee layoffs coincided with the company reorganization	Employee layoffs conceded with the company reorganisation	4
4	rob sat at the pond and sketched the stray geese	Roxette at the pond and sketchy the stray geese	8
5	the hallway opens into a huge chamber	the hallway opens into a huge chambered	2
1	the paper boy bought two apples and three ices	the paperboy bought 2 apls and 3 Isis	14

Figure 24: Data showing Levenshtein distance between original and first transcribed text

Uncorrected errors include characters that were missed or wrongly interpreted during the transcription of a given sentence. The Levenshtein distance as described in section 4.3.2 was measured. We evaluated the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other to determine the uncorrected errors across 10 blocks for all 10 participants. We observed the grand mean of 4.49, 6.24, 7.13 for uncorrected errors using Dwell TaG, Command TaG and Voice only approach respectively.

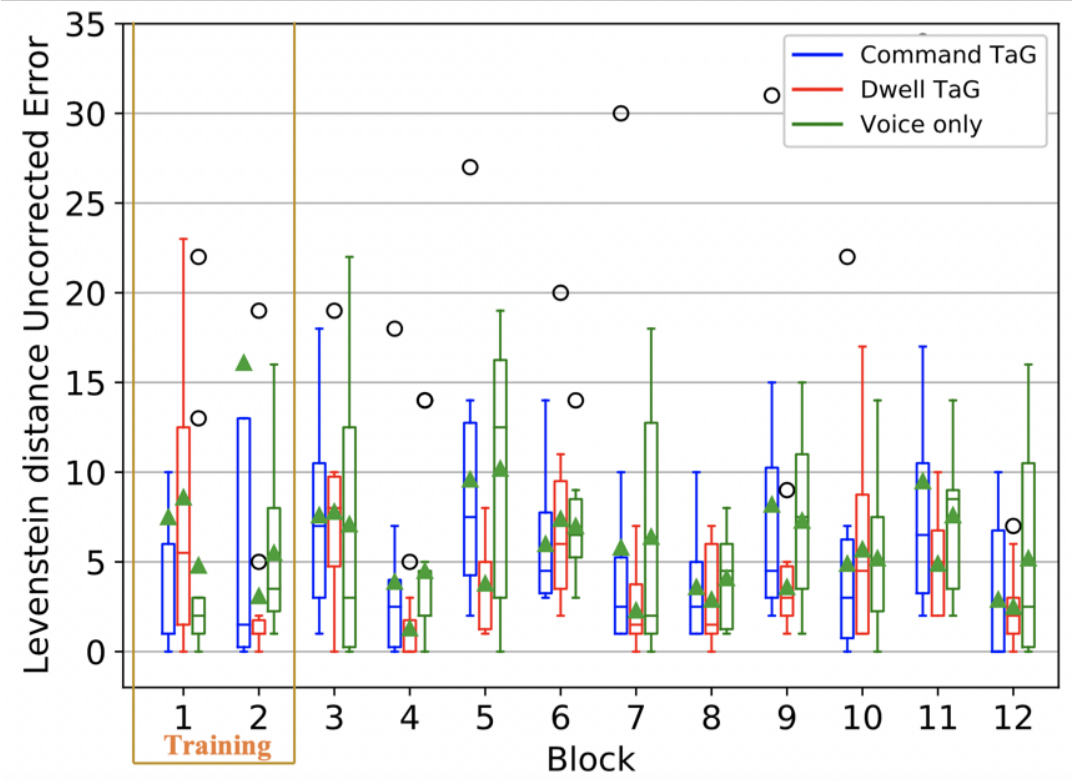


Figure 25: Uncorrected errors for Dwell TaG, Command TaG and Voice only

Although the ANOVA test at ($F_{2,27}$ = 3.23, p = 0.06) had no statistically significant difference in uncorrected errors, we observed multimodal approach has relatively less uncorrected errors overall as shown in Figure 25. The median value across all blocks for dwell TaG was always lower compared to Voice only approach. Similarly, uncorrected errors for Command TaG except for block 11 was lower than for Voice only approach. The result showed multimodal approach performed better than unimodal approach.

5.6.2 Image Description Task

A realistic scenario of text creation and editing was done within image description task where participants described the images presented to them. Similar to Read and Correct task, we performed quantitative measures i) Block Completion Time ii) Error Correction to evaluate each of editing approach: Voice only, Dwell TaG and Command TaG. Similar to previous task, first two blocks were for training and were removed as part of the analysis.

5.6.2.1 Block completion time

Shapiro-Wilk test showed that our data was normally distributed. We performed ANOVA test on image description task which showed no statistical significance at ($F_{2,27}$ = 2.60, p = 0.009) for block completion time.

Average block completion time using Voice only was 141.96 seconds, using Dwell TaG was 113.15 seconds and Command TaG was 97.49 seconds. This shows that Dwell TaG and Command TaG (multimodal approach) performed better than Voice only (unimodal approach). From the box plot in Figure 26, Command TaG and Voice TaG always has lower median value for block completion time than Voice only approach.

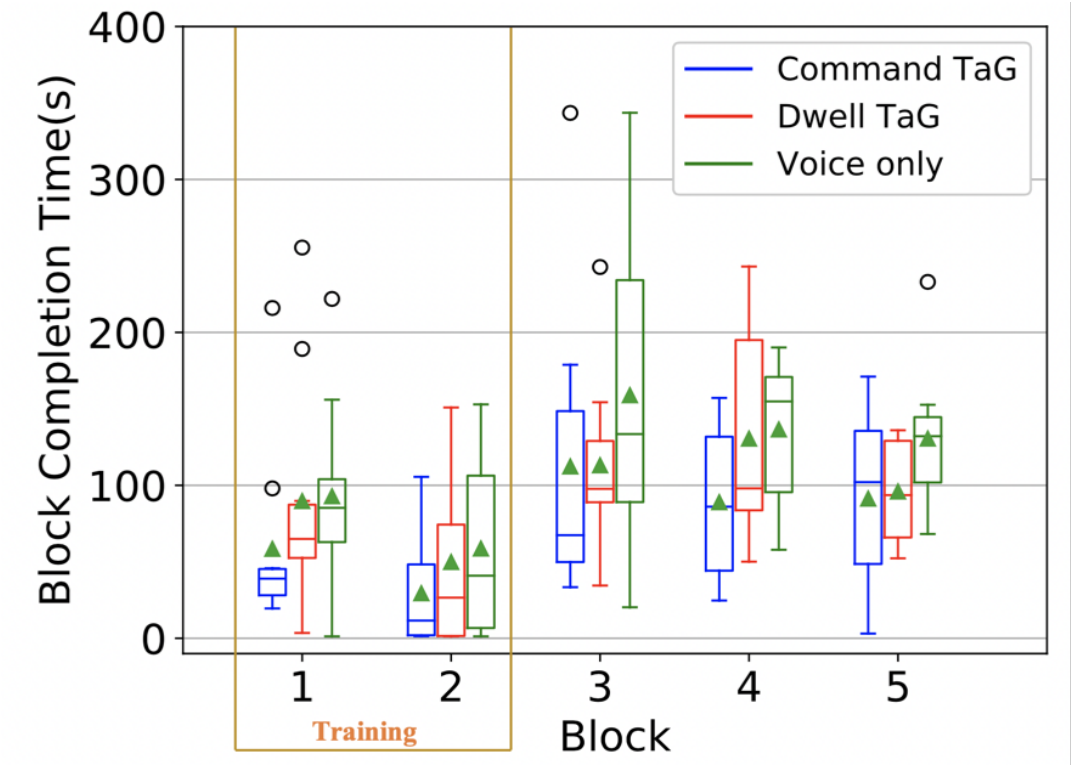


Figure 26: Average block completion time for Dwell TaG, Command TaG and Voice only input methods for Image Description Task

5.6.2.2 Error correction time

We evaluate the error correction time for image description task across Voice only, Dwell TaG and Command TaG. Shapiro-Wilk test showed that our data was normally distributed for total error correction. We performed ANOVA test on image description task which showed statistical significance at ($F_{2,12}$ = 9.96, p = 0.001) for error correction time. We can observe that multimodal approach takes shorter error correction time compared to that of unimodal approach as shown in the box plot in Figure 27 where Voice only approach reaches the max error correction time of 230 seconds for block 3. Also median for multimodal approach across all block is less compared to unimodal approach.

An average error correction for Voice only was 70.06 seconds compared to 31.37s seconds and 42.30 seconds for Command TaG and Dwell TaG respectively.

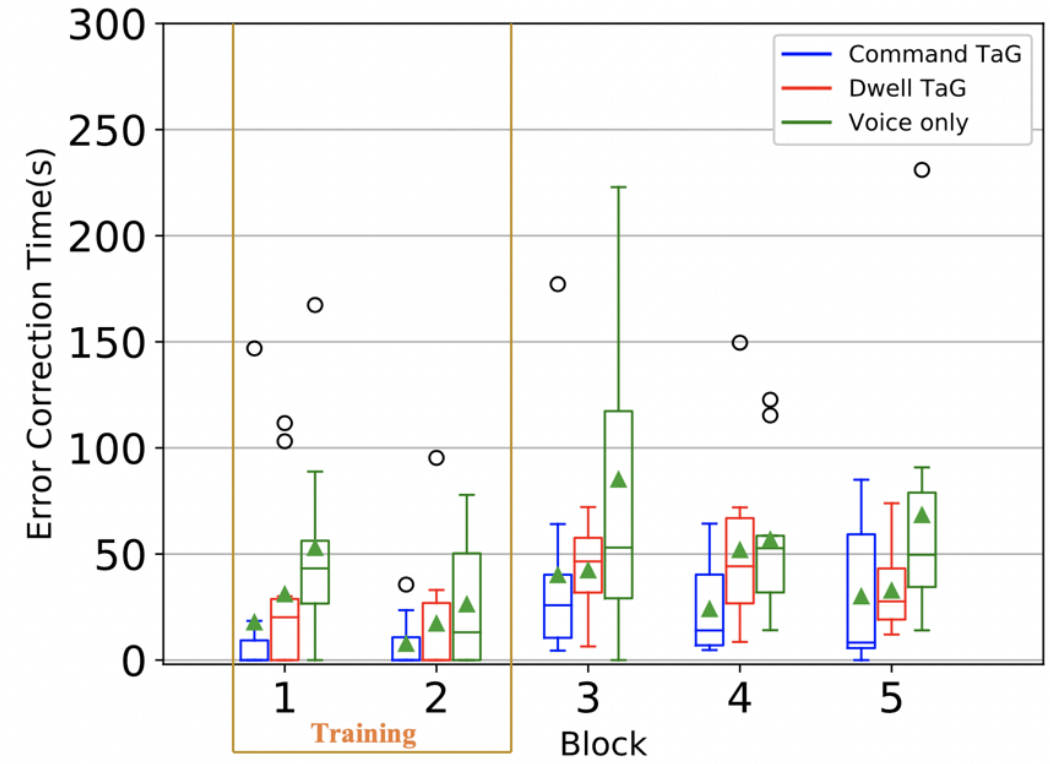


Figure 27: Error correction time for Dwell TaG, Command TaG and Voice only input methods for Image Description Task

Total time taken to correct errors was further broken down into two modes of correcting transcription errors. 1) Spell mode and 2) Suggestion mode. Shapiro-Wilk test showed that our data was normally distributed for error correction in spell mode and error correction in suggestion mode

5.6.2.2.1 Spell mode We observed statistical significance at ($F_{2,12} = 8.72, p = 0.004$). It was also clear that unimodal (Voice only) approach had higher error correction time across all blocks with average error correction time of 35.25 seconds compared to 17.54 for Command TaG and 12.99 seconds for Dwell TaG as per Figure 28. We can also see Block 2 has 0 error correction time for Command TaG and Dwell TaG which is because participant felt confident with the first training and didnot want to go through another set of training session. We can however see that participant did training for Voice only for both blocks suggesting they needed time to getting used to the voice commands.

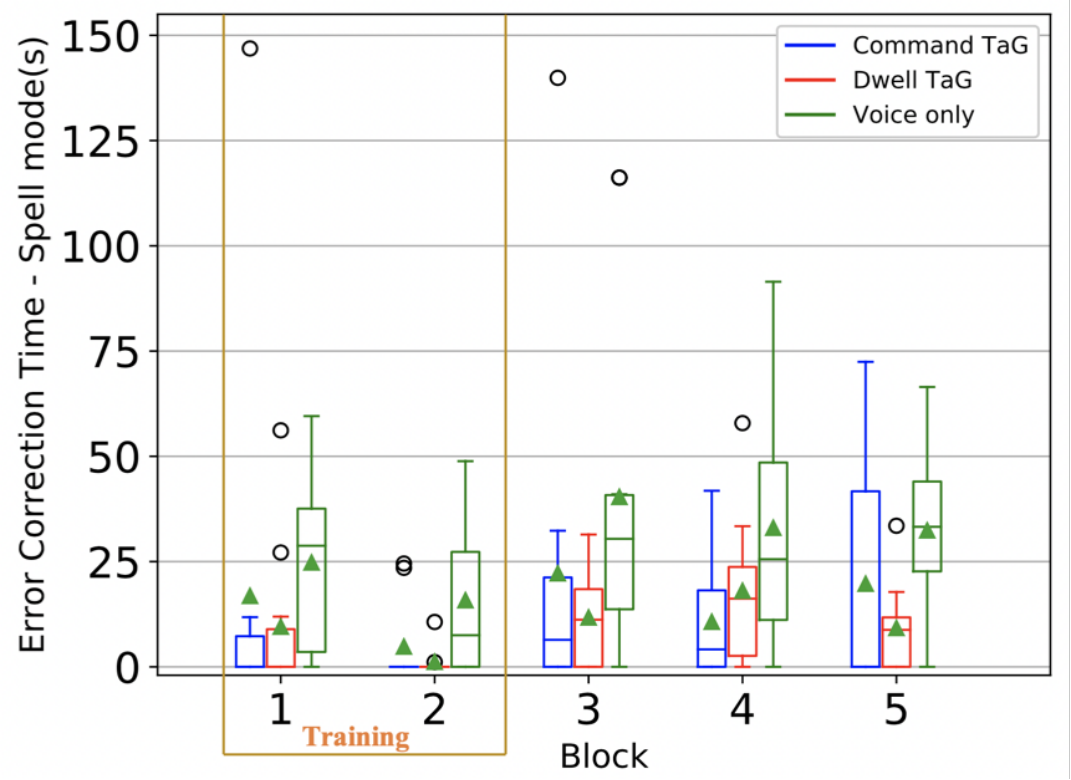


Figure 28: Error correction time for Dwell TaG, Command TaG and Voice only when using Spell mode

5.6.2.2.2 Suggestion mode There was a statistical significance at ($F_{2,12}$ = 6.96, p = 0.002) for error correction time when using suggestion mode only. The Dwell TaG, Command TaG and Voice only approach had average error correction time of 29.31seconds, 12.83 seconds and 34.81 seconds respectively. Also, the box plot in Figure 29 shows the error correction time in suggestion mode reaching as high of 120 seconds for voice only approach while the Dwell and Command TaG reached a high of 60. We also observed that the voice only approach had lower upper quartile(Q3) compared to Dwell TaG. This was however the case only for Block 4.

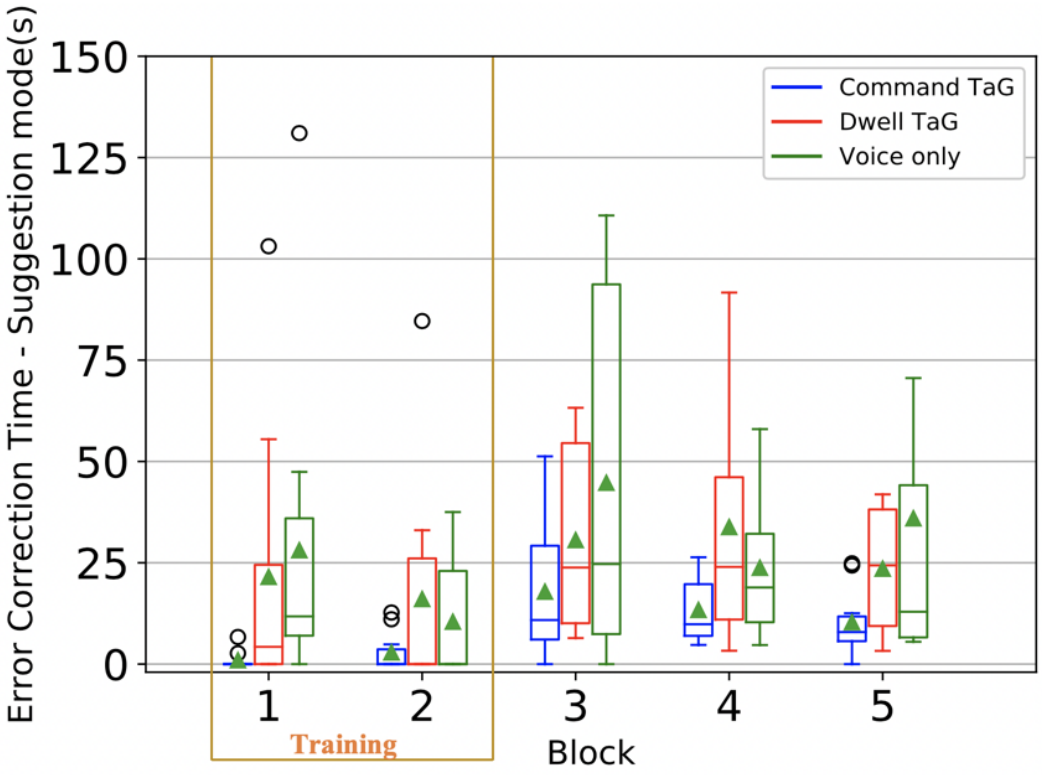


Figure 29: Error correction time for Dwell TaG, Command TaG and Voice only when using Suggestion mode

5.7 Qualitative Evaluation

In addition to comparing the usability of the multimodal interface with respect to efficiency, effectiveness and performance, a new research experiment analysing the qualitative evaluation was done. We evaluated the subjective workload using NASA-TLX and System Usability Scale (SUS) score across each experimental tasks i) Read and correction task ii) Image description task

After our final experimental study on using Dwell TaG, Command TaG and Voice only for read and correction task and image description task, qualitative evaluation was analysed depending on the feedback from the participants. The evaluation was based on preference, comfort, speed, accuracy, usage intention and overall experience.

Preference: Every participant was asked to make a ranking as per their preference for the three correction methods. It was indicative that 77.7% went with Dwell TaG as the most preferred choice for correcting transcription errors followed by Voice and Command TaG.

Comfort: Voice only approach was favoured among the three. Although the correction time was larger for Voice only, the issue of Midas touch problem (accidental selection on non-erroneous words) encountered during the correction process using Dwell Tag was reported painful. Similar issue while within Command TaG was stated where participant having to dwell on a word while giving the "Select" command was uncomfortable.

Speed and Accuracy: When asked about the temporal demand on completing the task, most participant indicated the Dwell TaG as the desired method despite uncomfortable false triggering. Furthermore, dwelling on a word and confirming selection by speaking "Select" command for Command TaG method was complained by most participants. The accuracy lacked in Command TaG compared to Dwell TaG which upon getting familiar was appreciated by most participants. Voice only approach also had shortcomings when commands were not recognised correctly.

Overall Dwell TaG was the most preferred choice of usage by participants. Voice was ranked as second choice compared to Command TaG. The overall experience outlined the multimodal approach being convenient compared to unimodal approach. Few participants expressed fatigue from using the command when it was not recognised correctly. Some concern was raised when using Command TaG as dwelling while saying "Select" command was tedious. Participants however gave positive review on using Dwell TaG outlining some unease due to Midas touch problem.

5.7.1 NASA-TLX

National Aeronautics and Space Administration-Task Load Index (NASA-TLX) is one of the most widely used tools for subjective workload. As often a task's workload can exceed the individual's ability, NASA-TLX would provide a reference for us to figure out the participants mental workload for the system we designed. For that reason, we make use of NASA-TLS's six subscales to measure various aspects of mental workload as the combination of these categories provides accurate rating of task mental workload.

The six subscales consider the "subjective importance" and "magnitude" with respect to task. Accordingly we have 3 broad categories of scales 1) Task-Related measuring the objective demands of the task (Physical, Mental and Temporal Demands) 2)Behaviour-Related which reflects an individual's subjective evaluation of the task (Own performance and Effort) and, 3) Subject-Related which includes the psychological impact on the individual (Frustration).

NASA TLX was evaluated to understand the workload of the task through mental, physical and temporal demands alongside effort, performance and frustration level. Figure 30 below compares the multimodal and unimodal approaches i.e. Dwell TaG, Command TaG and Voice only. We observed the ratings given by 10 participants across three different approaches with 6 questions about the error correction task. The measurement was scaled between 1 to 5 with higher rating giving the worst behaviour.

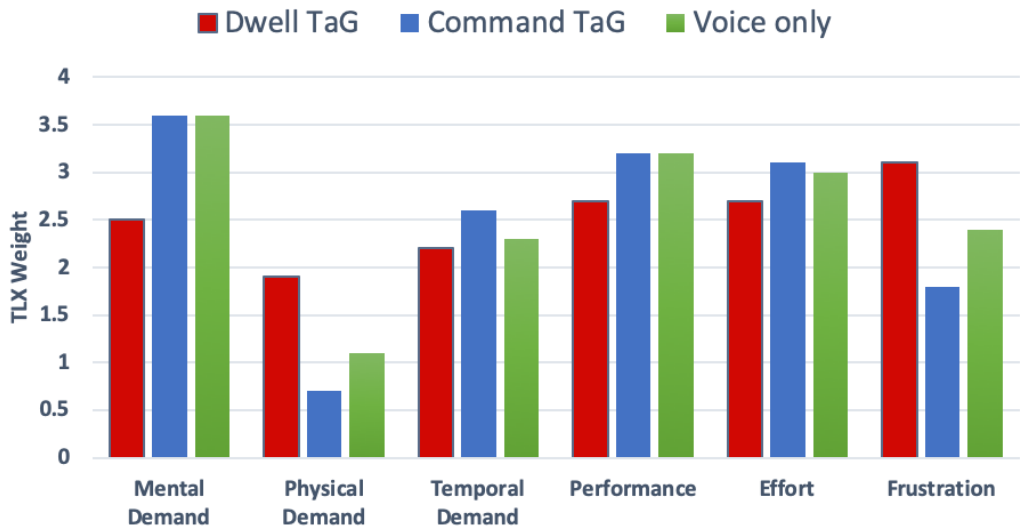


Figure 30: Bar chart showing the NASA TLX for Dwell TaG, Command TaG and Voice only

As can be seen in Figure 30, mental demand was higher for Command and Voice only with an Average TLX weight of 3.6. Dwell was considered to have low mental demand with

2.7 compared to Command and Voice. Physical demand across all three approach was below 2.0 which was expected considering not much physical activity required for each task. Furthermore, each participant felt a similar amount of pressure due to the pace at which task elements occurred, resulting in 2.2, 2.3 and 2.6 across Dwell TaG, Voice and Command TaG respectively. By comparing the cognition demands we can substantiate our initial claim of using Dwell TaG for correcting transcription errors being less demanding compared to Command TaG or Voice approach.

Additionally, the last three columns outlined the emotion about doing the task. Although all three approach of correcting errors took a lot of effort from participants, Dwell TaG required less effort of three with a TLX weight of 2.7. However, Dwell Tag seem to be the frustrating approach among three with 3.2 compared to 2.4 and 1.8 for Voice and Command TaG respectively. In terms of performance, every user felt they were performing well and were satisfied with their approach in correcting the transcription errors.

5.7.2 System Usability Scale (SUS) Score

The System Usability Scale (SUS) provides a mechanisim to measure the usability of the system. We present 10 usability questions to candidates who categorises into one of

- Strongly Disagree
- Disagree
- Neutral
- Agree
- Strongly Agree

For each of this scale we provide a numerical representation i.e. 1 to 5 points respectively. Next step would then to calculate the SUS score as following

- X = Sum of the points for all odd-numbered questions – 5
- Y = 25 – Sum of the points for all even-numbered questions
- SUS Score = (X + Y) x 2.5

The question we presented in odd-numbered ranking had positive tone, hence if the response was strongly agree, we gave the maximum point. Vice versa, for even-numbered questions which were negative toned, minimum point was given if response was strongly agree. The result was then multiplied by 2.5 to ensure maximum point of 10 for each question.

We then evaluated the results for each of error correction approach across all tasks and compared with the standard SUS score metric shown in Figure 31 in analysing the results.

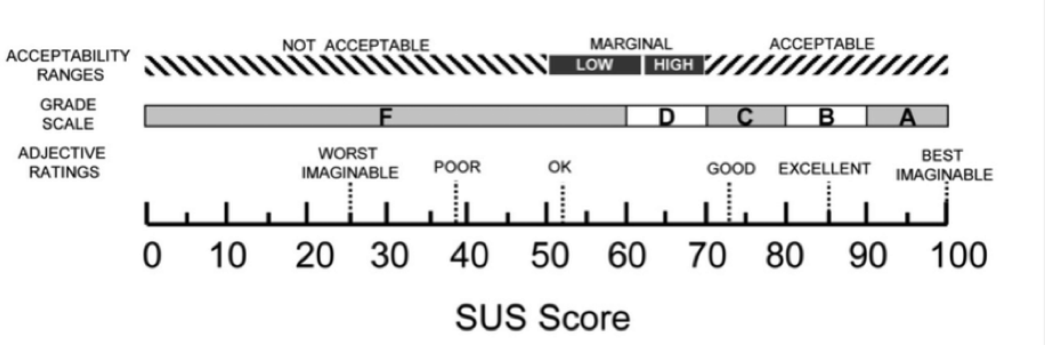


Figure 31: SUS score metric [4]

Thus, with the help of SUS score, we were able to tell the usability performance in terms of the systems effectiveness, efficiency and overall ease of use.

According to the algorithm of SUS score, usage of Dwell TaG got 70, Voice got 74.25 and Command TaG got 66.25 as can be seen in Figure 32. According to the adjective ratings for SUS score described in section 4.3.2, Voice only and Dwell TaG usage was "Good" compared to Command TaG which were rated "Ok". Also, in terms of grade scale, Command TaG fell under the D category while Voice and Dwell TaG got C.

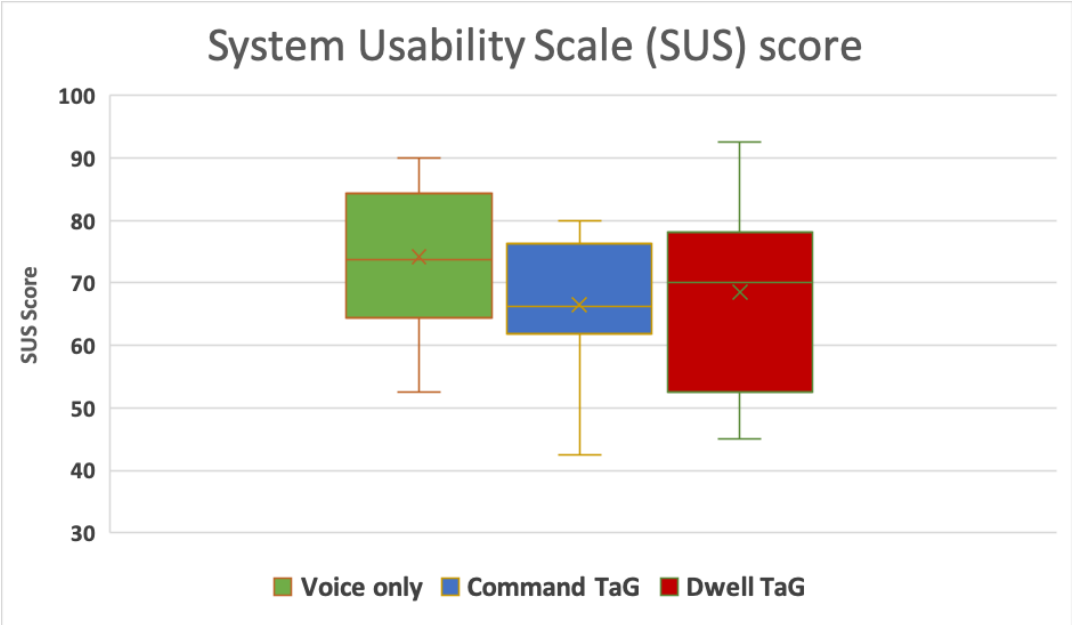


Figure 32: SUS score for Dwell TaG, Command TaG and Voice only approach given by 10 participants. Y-axis is the SUS score, ranging from 0 to 100

6 Discussion

It was evident from the evaluation chapter that multimodal approach (voice and gaze) of error correction was favoured over unimodal approach (voice only). From two experimental tasks i) Read and correct task and ii) Image description task, we saw block completion time for Dwell TaG was better than Voice only. Similarly, error correction time was less for Dwell TaG compared to Command TaG and Voice only. It became clear that an additional step of giving "Map" command to select the error word took longer time when using Voice only approach. As we can realise that when user makes an attempt to correct an error in voice only approach, the first step is to look at the error and say the command "Map". Since the user looks at the error word first, with Dwell TaG, we already select the word reducing the time to say further commands for selection.

Furthermore, the qualitative measures taken to evaluate different edit methods also showed multimodal being more intuitive than unimodal approach. When evaluating mental workload, participants gave a positive ranking to multimodal approach compared to unimodal across objective demands, individual's subjective evaluation and psychological impact.

Also, in terms of comfort, Dwell TaG was considered better than Command TaG. The possible reason is the stress to gaze at the word and maintain the dwelling while saying "Select" command. Voice only was however considered simple and comfortable despite the longer error correction time. This was because the gaze suffered either the Midas touch problem for Dwell TaG or the glitter effect for Command TaG while voice was less error prone.

Interestingly, participants chose Voice only approach against Dwell TaG and Command TaG as per the preference. This was a valid feedback as there were no time constraint on completion of task, and despite voice taking longer time to correct errors, participants felt more comfortable with it. The reason could be that participant were less familiar in using eye tracking devices over voice recognition tools.

7 Conclusion

In this research, we studied error correction techniques using voice based input as a unimodal approach and the integration of voice and gaze as multimodal approach. Throughout our thesis work, we devised different experimental tasks using different edit techniques. We analysed the existing solutions as well as our proposed solution to deduce our hypothesis - if multimodal approach in correcting errors would perform better than unimodal approach.

We developed a web application which supported voice only input as well as gaze input. In the beginning we conducted pilot studies to obtain feedbacks from users for creating a robust and usable system. We first analysed the existing voice based editing tools from Google Docs and obtained feedback on the challenges it faced. Further research was done, and a robust system was designed eliminating the challenges faced by existing solutions. Second feedback session was conducted for our system to further enhance the capability of our system. Thus, a system that was usable, reliable, well-functioning and effective was developed.

Our system supported different experimental setup whereby we were able to test and analyse different approaches for error correction. First scenario was to analyse voice only approach. Second scenario was to introduce gaze together with voice for multimodal approach in correcting errors. The second scenario TaG (Talk and Gaze) had two versions: Dwell TaG (D-TaG) and Command TaG(C-TaG). Thus, the edit methods using Voice only, Dwell TaG and Command TaG for correcting transcription errors were analysed. Two tasks were proposed i) Read and Correct task and ii) Image description task. Data collected for each of these tasks evaluated the system performance whereby we were able to perform quantitative measures and qualitative measures.

Block completion time, Uncorrected Errors and Error correction time were the quantitative measures which allowed us to evaluate the system under different edit methods. For each tasks, Dwell TaG and Command TaG performed better than Voice only approach. Dwell TaG also showed least error correction time, higher correction rate and higher usability scores. It therefore was safe to conclude that multimodal approach performed better in most cases and was preferred approach on error correction compared to unimodal approach. Although qualitative measures from NASA TLX and SUS score did not show multimodal approach to be distinctly better than unimodal approach, we could argue that multimodal approach was a preferred choice with lower cognitive load.

8 Future Work

Our study had some limitations which could be improved in future for larger user acceptance and better evaluation of unimodal approach against multimodal approach. The limitations we observed were during speech recognition, calibration of eye gaze, prediction list, familiarity with the system and shorter training sessions. We also propose better investigation on complex edits and natural error correction for future research.

When using existing speech recognition tool, we realised transcription issues. There were cases where phonetically similar words were wrongly transcribed for example “ices” as “isis”, “related” as “relented”. This became a problem when voice commands were wrongly recognised as this would not execute the command expected leaving participants frustrated. For example, when a voice command “Map” was given, speech recognition tool recognised it as “My app” resulting in massive downtime in error correction. We believe with better training and also maintaining some sort of key value pair for commands and voice in dictionary would result in effective system in future work.

Calibration when using gaze device is another sector which could be improved. We had some candidates who were discouraged when gaze was not detected correctly, and re-calibration was needed per session. It was also reported that calibration software had a drift for taller candidates i.e. their gaze point was slightly above than expected. In future work, we could also improve the device used as well enhance the software used for calibration.

During our experiment, error correction was done in two modes, suggestion mode and spell mode. When an error appeared and was selected for correction, users were prompted with list of suggestions. The suggestion list consisted of five words which were obtained from prediction algorithm from third party API’s. This was however not accurate and left many participants annoyed as they now had to go to spell mode to spell out each character for correction. Therefore, as a future work, we could extend the prediction model to better predict the result.

Familiarity with the system and shorter training session could be improved in future for collecting better user data. It was mentioned by most participant that the voice commands they had to remember could have been improved if more training sessions were available. In future we can organise longer training sessions, so participants get used to the eye tracking as well as be familiar with the voice commands.

Similarly, as an extension to our work, we could now analyse further multimodal approach in correcting errors. We could introduce touch, head gestures etc to observe different error correction technique and compare them to find the best solution.

References

[1] Ainsworth, W. A. and Pratt, S. [1992], 'Feedback strategies for error correction in speech recognition systems', *International Journal of Man-Machine Studies* **36**(6), 833–842.

[2] Aleksic, P., Ghodsi, M., Michaely, A., Allauzen, C., Hall, K., Roark, B., Rybach, D. and Moreno, P. [2015], Bringing contextual information to google speech recognition, *in* 'Sixteenth Annual Conference of the International Speech Communication Association'.

[3] Baber, C. and Hone, K. S. [1993], 'Modelling error recovery and repair in automatic speech recognition', *International Journal of Man-Machine Studies* **39**(3), 495–515.

[4] Bangor, A., Kortum, P. and Miller, J. [2009], 'Determining what individual sus scores mean: Adding an adjective rating scale', *J. Usability Studies* **4**(3), 114–123.
URL: *http://dl.acm.org/citation.cfm?id=2835587.2835589*

[5] Bourguet, M.-L. [2006], 'Towards a taxonomy of error-handling strategies in recognition-based multi-modal human–computer interfaces', *Signal Processing* **86**(12), 3625–3643.

[6] Brinton, B., Fujiki, M. and Sonnenberg, E. A. [1988], 'Responses to requests for clarification by linguistically normal and language-impaired children in conversation', *Journal of Speech and Hearing Disorders* **53**(4), 383–391.

[7] Card, S. K., Moran, T. P. and Newell, A. [1980], 'The keystroke-level model for user performance time with interactive systems', *Communications of the ACM* **23**(7), 396–410.

[8] Christian, K., Kules, B., Shneiderman, B. and Youssef, A. [2000], A comparison of voice controlled and mouse controlled web browsing, *in* 'Proceedings of the fourth international ACM conference on Assistive technologies', ACM, pp. 72–79.

[9] Danis, C., Comerford, L., Janke, E., Davies, K., De Vries, J. and Bertrand, A. [1994], Storywriter: A speech oriented editor, *in* 'Conference companion on Human factors in computing systems', ACM, pp. 277–278.

[10] De Mauro, C., Gori, M., Maggini, M. and Martinelli, E. [2001], Easy access to graphical interfaces by voice mouse, Technical report, Technical report, Università di Siena. Available from the author at: maggini ....

[11] Douglas, H. R. [1999], 'Method and apparatus for editing documents through voice recognition'. US Patent 5,875,429.

[12] Dunlop, M., Nicol, E., Komninos, A., Dona, P. and Durga, N. [2016], Measuring inviscid text entry using image description tasks, *in* 'CHI'16 Workshop on Inviscid

Text Entry and Beyond', San Jose, CA.
URL: *http://www.textentry.org/chi2016/9%20-%20Dunlop%20-%20Image%20Description%20Tasks.pdf*

[13] Farrell, S. and Zhai, S. [2005], 'System and method for selectively expanding or contracting a portion of a display using eye-gaze tracking'. US Patent App. 10/648,120.

[14] Fitts, P. M. [1954], 'The information capacity of the human motor system in controlling the amplitude of movement.', *Journal of experimental psychology* **47**(6), 381.

[15] Garofolo, J. S. [1993], 'Timit acoustic phonetic continuous speech corpus', *Linguistic Data Consortium, 1993* .

[16] Graves, A., Mohamed, A.-r. and Hinton, G. [2013], Speech recognition with deep recurrent neural networks, *in* 'Acoustics, speech and signal processing (icassp), 2013 ieee international conference on', IEEE, pp. 6645–6649.

[17] Han, S., Kang, J., Mao, H., Hu, Y., Li, X., Li, Y., Xie, D., Luo, H., Yao, S., Wang, Y. et al. [2017], Ese: Efficient speech recognition engine with sparse lstm on fpga, *in* 'Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays', ACM, pp. 75–84.

[18] Hansen, J. P., Hansen, D. W. and Johansen, A. S. [2001], Bringing gaze-based interaction back to basics., *in* 'HCI', Citeseer, pp. 325–329.

[19] Jacob, R. J. [1991], 'The use of eye movements in human-computer interaction techniques: what you look at is what you get', *ACM Transactions on Information Systems (TOIS)* **9**(2), 152–169.

[20] Jacob, R. J. [1993], 'Eye movement-based human-computer interaction techniques: Toward non-command interfaces', *Advances in human-computer interaction* **4**, 151–190.

[21] Karat, C.-M., Halverson, C., Horn, D. and Karat, J. [1999], Patterns of entry and correction in large vocabulary continuous speech recognition systems, *in* 'Proceedings of the SIGCHI conference on Human Factors in Computing Systems', ACM, pp. 568–575.

[22] Klimt, B. and Yang, Y. [2004], Introducing the enron corpus., *in* 'CEAS'.

[23] Kumar, S. and Dorairangaswamy, M. [n.d.], 'Hands-free pc control for users with disabilities of their hands'.

[24] Lankford, C. [2000], Effective eye-gaze input into windows, *in* 'Proceedings of the 2000 symposium on Eye tracking research & applications', ACM, pp. 23–27.

[25] Lyons, M. J., Chan, C.-H. and Tetsutani, N. [2004], Mouthtype: Text entry by hand and mouth, *in* 'CHI'04 Extended Abstracts on Human Factors in Computing Systems', ACM, pp. 1383–1386.

[26] Manaris, B. and Harkreader, A. [1998], Suitekeys: a speech understanding interface for the motor-control challenged, *in* 'Proceedings of the third international ACM conference on Assistive technologies', ACM, pp. 108–115.

[27] Mantravadi, C. S. [2009], Adaptive multimodal integration of speech and gaze, PhD thesis, Rutgers University-Graduate School-New Brunswick.

[28] Martin, T. [1980], 'Practical speech reco-gniers and some performance effectiveness parame-ters', *Trends in speech recognition* pp. 24–38.

[29] McNair, A. E. and Waibel, A. [1994], Improving recognizer acceptance through robust, natural speech repair, *in* 'Third International Conference on Spoken Language Processing'.

[30] Menges, R., Kumar, C., Müller, D. and Sengupta, K. [2017], Gazetheweb: A gaze-controlled web browser, *in* 'Proceedings of the 14th Web for All Conference on The Future of Accessible Work', ACM, p. 25.

[31] Mihara, Y., Shibayama, E. and Takahashi, S. [2005], The migratory cursor: accurate speech-based cursor movement by moving multiple ghost cursors using non-verbal vocalizations, *in* 'Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility', ACM, pp. 76–83.

[32] Miniotas, D., Špakov, O., Tugoy, I. and MacKenzie, I. S. [2006], Speech-augmented eye gaze interaction with small closely spaced targets, *in* 'Proceedings of the 2006 symposium on Eye tracking research & applications', ACM, pp. 67–72.

[33] Mudholkar, G. S., Srivastava, D. K. and Thomas Lin, C. [1995], 'Some p-variate adaptations of the shapiro-wilk test of normality', *Communications in Statistics-Theory and Methods* **24**(4), 953–985.

[34] Murray, A., Frankish, C. and Jones, D. [2014], 'Data-entry by voice: Facilitating correction of misrecognitions', *Interactive speech technology* pp. 137–144.

[35] Oviatt, S. [1997], 'Multimodal interactive maps: Designing for human performance', *Human-computer interaction* **12**(1), 93–129.

[36] Oviatt, S., Cohen, P., Wu, L., Duncan, L., Suhm, B., Bers, J., Holzman, T., Winograd, T., Landay, J., Larson, J. et al. [2000], 'Designing the user interface for multimodal speech and pen-based gesture applications: state-of-the-art systems and future research directions', *Human-computer interaction* **15**(4), 263–322.

[37] Oviatt, S. and VanGent, R. [1996], Error resolution during multimodal human-computer interaction, *in* 'Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on', Vol. 1, IEEE, pp. 204–207.

[38] Robbe, S., Carbonell, N. and Valot, C. [1997], Towards usable multimodal command languages: Definition and ergonomic assessment of constraints on users' spontaneous speech and gestures, *in* 'Fifth European Conference on Speech Communication and Technology'.

[39] Roe, D. B., Wilpon, J. G. et al. [1994], *Voice communication between humans and machines*, National Academies Press.

[40] Ruan, S., Wobbrock, J. O., Liou, K., Ng, A. and Landay, J. [2016], 'Speech is 3x faster than typing for english and mandarin text entry on mobile devices', *arXiv preprint arXiv:1608.07323* .

[41] Schalkwyk, J., Beeferman, D., Beaufays, F., Byrne, B., Chelba, C., Cohen, M., Kamvar, M. and Strope, B. [2010], "your word is my command": Google search by voice: a case study, *in* 'Advances in speech recognition', Springer, pp. 61–90.

[42] Sears, A., Feng, J., Oseitutu, K. and Karat, C.-M. [2003], 'Hands-free, speech-based navigation during dictation: difficulties, consequences, and solutions', *Human-computer interaction* **18**(3), 229–257.

[43] Sengupta, K., Ke, M., Menges, R., Kumar, C. and Staab, S. [2018], Hands-free web browsing: enriching the user experience with gaze and voice modality, *in* 'Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications', ACM, p. 88.

[44] Soukoreff, R. W. and MacKenzie, I. S. [2003], Metrics for text entry research: an evaluation of msd and kspc, and a new unified error metric, *in* 'Proceedings of the SIGCHI conference on Human factors in computing systems', ACM, pp. 113–120.

[45] Suhm, B., Myers, B. and Waibel, A. [2001], 'Multimodal error correction for speech user interfaces', *ACM transactions on computer-human interaction (TOCHI)* **8**(1), 60–98.

[46] Trekhleb [n.d.], 'trekhleb/javascript-algorithms'.
URL: *https://github.com/trekhleb/javascript-algorithms/tree/master/src/algorithms/string/levenshtein-distance*

[47] Velichkovsky, B., Sprenger, A. and Unema, P. [1997], Towards gaze-mediated interaction: Collecting solutions of the "midas touch problem", *in* 'Human-Computer Interaction INTERACT'97', Springer, pp. 509–516.

[48] Yankelovich, N., Levow, G.-A. and Marx, M. [1995], Designing speechacts: Issues in speech user interfaces, *in* 'Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press/Addison-Wesley Publishing Co., pp. 369–376.

[49] Zhai, S., Morimoto, C. and Ihde, S. [1999], Manual and gaze input cascaded (magic) pointing, *in* 'Proceedings of the SIGCHI conference on Human Factors in Computing Systems', ACM, pp. 246–253.