GazeButton: Enhancing Buttons with Eye Gaze Interactions

Sheikh Rivu
Bundeswehr University Munich, Germany
sheikh.rivu@unibw.de

Yasmeen Abdrabou
Bundeswehr University Munich, Germany
German University in Cairo, Egypt
yasmeen.essam@unibw.de

Thomas Mayer
LMU Munich, Germany
thomas.r.mayer@web.de

Ken Pfeuffer
Bundeswehr University Munich, Germany
ken.pfeuffer@unibw.de

Florian Alt
Bundeswehr University Munich, Germany
florian.alt@unibw.de

ABSTRACT

The button is an element of a user interface to trigger an action, traditionally using click or touch. We introduce GazeButton, a novel concept extending the default button mode with advanced gaze-based interactions. During normal interaction, users can utilise this button as a universal hub for gaze-based UI shortcuts. The advantages are: 1) easy to integrate in existing UIs, 2) complementary, as users choose either gaze or manual interaction, 3) straightforward, as all features are located in one button, and 4) one button to interact with the whole screen. We explore GazeButtons for a custom-made text reading, writing, and editing tool on a multitouch tablet device. For example, this allows the text cursor position to be set as users look at the position and tap on the GazeButton, avoiding costly physical movement. Or, users can simply gaze over a part of the text that should be selected, while holding the GazeButton. We present a design space, specific application examples, and point to future button designs that become highly expressive by unifying the user's visual and manual input.

CCS CONCEPTS

• **Human-centered computing** → **Human computer interaction (HCI)**; **Interaction techniques**; *Text input*;

KEYWORDS

Interaction Modality, Text Input, Touch and Gaze.

ACM Reference Format:
Sheikh Rivu, Yasmeen Abdrabou, Thomas Mayer, Ken Pfeuffer, and Florian Alt. 2019. GazeButton: Enhancing Buttons with Eye Gaze Interactions. In *Communication by Gaze Interaction (COGAIN @ ETRA'19), June 25–28, 2019, Denver , CO, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3317956.3318154

COGAIN @ ETRA'19, June 25–28, 2019, Denver , CO, USA

ACM ISBN 978-1-4503-6728-8/19/06…$15.00
https://doi.org/10.1145/3317956.3318154

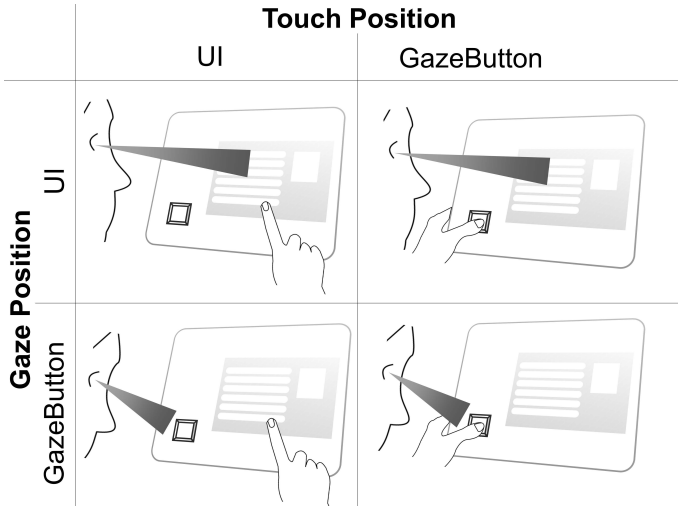

Figure 1: GazeButton augments default buttons (top left) with three additional input states. These can be selected via gaze, adding new actions.

1 INTRODUCTION

Since the introduction of direct manipulation by Shneiderman [Shneiderman 1983], the common computer interface is based on a 2D GUI. A fundamental element to enable interaction with a GUI is a *button*. Whether on a desktop computer or a mobile touch device, the semantics of a button remain the same: users can select the button with manual input to trigger an atomic action. We explore how eye gaze can enhance this basic concept of a button.

Advances in eye tracking technology enable complementary eye trackers integrated into the default display or attached as peripheral devices [Khamis et al. 2018a; Tobii 2019]. Research on gaze input explored a plethora of interaction techniques and device types [Esteves et al. 2015; Jacob 1990; Sibert and Jacob 2000; Stellmach and Dachselt 2012; Zhai et al. 1999]. These methods have been investigated isolated as generic selection methods or as alternatives to the default manual input. Recent work explored the integration of gaze and touch as interaction techniques on large displays [Turner et al. 2015]. However, it is not entirely clear how the user's gaze can work in unity with the manual input as well as how gaze can be integrated in long-established UI concepts.

We present *GazeButton* – a novel UI element that extends the default button concept with users' visual attention. The idea is to simplify the many manual interactions users perform through unifying them into a single button. Whereas the default button enables an atomic action, adding gaze allows to make it more expressive by taking into account where the user looks while issuing the manual input [Pfeuffer et al. 2014]. The advantages for UI design are:

- Expressiveness: GazeButton is one UI element that covers a variety of actions the user can perform, making it a highly expressive hub from which the user can shortcut interactions at any time.
- Complementarity: As only one new UI element is introduced, the existing user interface remains largely unchanged. Hence the user can at any time choose to either use the GazeButton or the legacy way of interaction.
- Integration: As we propose a single UI element, it is easy to integrate it into existing UIs to benefit from gaze interaction.
- Reach: Users can look anywhere on the screen while triggering an action from the button.

From an input-theoretic perspective, GazeButton provides three input states in addition to the default interaction state (Figure 1). Here the top left state shows a typical user interaction without the use of gaze: users look somewhere on the screen, and also touch on it. At the top right, the user presses the GazeButton while looking at any object in the UI. For example, if the user is looking at a text field, the button's action is to place the text cursor at the visual position. However, if the user is looking at a key on the keyboard, the user would type that key. In the bottom left scenario, users can look at the button while touching a target in the UI. Hence, users can touch a target, but apply a different mode (such as shift) to it. Lastly, users can look and touch the same button (bottom right) to trigger another distinct action, e.g., to provide a system-level menu. In this paper, we explore these conceptual dimensions in the context of text writing and editing on a touchscreen tablet.

We contribute, first, a concept and design recommendation which can be used in the UI design process of applications to enhance eye gaze interactions. Second, we present an input state diagram which explores the combinations in which touch and gaze can be implemented in UIs to enhance the user's experience. Third, we illustrate application examples which show the utility of GazeButton. The focus of our work is to introduce novel interaction techniques for users to be carried out with minimal effort and as shortcuts to common actions.

2 RELATED WORK

In this section we review prior work. In particular, we summarize work that investigates mobile UI design and eye-tracking based interaction, as well as eye typing.

2.1 Mobile UI

Researchers have focused on the mobile touchscreen UI to improve the usability by accounting for better, ergonomic use. The grip and reach issue, i.e. that the hand holding the tablet is limited in interaction, has been extensively studied [Bergstrom-Lehtovirta and Oulasvirta 2014; Odell and Chandrasekaran 2012; Trudeau et al. 2013; Wolf and Henze 2014; Wolf et al. 2015].

Various methods were proposed to improve this and allow unimanual tablet interaction. For example, relevant UI elements can be automatically moved to the expected grip area [Wagner et al. 2012] and grip sensing can be integrated for flexible holding and UI adaptation [Cheng et al. 2013; Hinckley et al. 2016]. Pfeuffer et al. proposed gaze and touch interaction to enable whole-screen reachability, i.e. the free thumb touches redirected to the gaze position [Pfeuffer and Gellersen 2016]. However, in their design it is difficult to understand for users how this interaction works, as no dedicated UI element is displayed. In principle users may learn how it works and thus would not necessitate a button. However it is desired to minimise learning as much as possible, considering how touchscreen devices are used normally.

Pfeuffer et al. designed 'thumb buttons', that is buttons explicitly placed near the expected grip position of a tablet [Pfeuffer et al. 2017]. Though not employing gaze, these buttons enhance a stylus' functionality by providing various mode-switching features.

In this paper, our idea is to utilise such "thumb buttons" without a pen, but in combination gaze functionality. This provides users with clear visual feedback of the additional gaze interaction.

2.2 Gaze Interaction

Bolt demonstrates the potential of gaze for future applications [Bolt 1981]. A particular focus in this work is on how interactions are being initiated. The usefulness of eye movements have also been studied by [Jacob 1990] where the author not only diagnoses the barrier to using eye movement as a medium but also implements gaze-based interaction techniques. We learn from the reported experiences and observations. Performance of eye gaze as an input medium has been studied by [Sibert and Jacob 2000; Zhai et al. 1999], showing that it can be faster than manual input devices. Bednarik et al. investigate gaze with buttons in 'gaze-augmented interaction', where specific button UI elements are highlighted the longer users look at it [Bednarik et al. 2009]. As observed by [Stellmach and Dachselt 2012], gaze-based interaction techniques are not only effective and more natural but also a highly adaptive method for handheld devices. We take inspiration from these by adapting eye movements as a natural input medium to our application.

2.3 Mobile Gaze Interaction

The use of gaze-based interaction techniques has also been demonstrated for various hand held devices and is not limited to desktops. For example, [Drewes et al. 2007] discuss the potential of eye tracking in mobile devices through a user study. [Khamis et al. 2018a] presents a holistic view on the past, present and future of eye tracking on hand held devices. Their work not only describes the advancements that took place in research but also new opportunities and challenges requiring further research. We also get insight from [Bulling and Gellersen 2010], discussing the aspects of emerging research on eye tracking on mobile devices.

Gaze and touch interactions are investigated by [Stellmach and Dachselt 2012] through a user study, looking at how gaze can be used as an interaction technique for mobile devices to zoom and pan while browsing images on a large screen to highlight important information. The aim was to get user insight on using gaze and if they would enjoy gaze assisted applications for interaction. Turner

et al. applied gaze and touch to the context of transferring content between a local touchscreen and a remote display [Turner et al. 2015, 2014, 2011]. Pfeuffer et al. explored this multimodal UI on tablet and desktop touchscreens [Pfeuffer et al. 2014, 2015], showing that gaze has a high potential to provide interaction benefits when using eye gaze input to extend the default touch actions, e.g., by avoiding occlusion issues when using default touch input.

Various platforms have been proposed for mobile devices to enable gaze-based applications. For example, [Hohlfeld et al. 2015] applied computer vision-based gaze tracking in mobile scenarios. Using computer vision on tablets for gaze tracking has also been explored by [Wood and Bulling 2014]. Their work presents a prototype that achieves robust and near-realtime gaze estimation.

We take inspiration from these observations which state that gaze-based interaction is attractive for people using hand held devices and work to enhance experience in the context of text editing applications in a multi-touch tablet.

2.4 Eye Typing

Gaze typing was first introduced in the 1970s as an application to be used by disabled people [Majaranta and Räihä 2002]. Since then, research have been done in this area to make eye-typing as fast and accurate as possible. Eye-typing has also found its way in authentication applications such as interaction with public displays [Khamis et al. 2018b] and Virtual Reality (VR) [Rajanna and Hansen 2018]. An increasing number of research focuses on testing different typing speeds. Dasher [Tuisku et al. 2008] is one of the fastest eye-typing applications today. It is operated by continuous gaze pointing gestures through the letters. EyeSwipe is another example of eye typing. It works by scanning the gaze line [Kurauchi et al. 2016]. In addition, dwell-based gaze interaction has also found its way to eye-typing in many applications such as pEyeWrite which used a pie menu shape as a text pad [Huckauf and Urbina 2008]. GazeTheKey uses dwell time as entry method [Sengupta et al. 2017].

As can be seen from the previous applications, the focus has been on various input methodologies and different application areas. In contrast, we focus on a different perspective. In this paper we extend touch typing by gaze interaction.

3 CONCEPT GAZEBUTTON

3.1 Design Considerations

In contrast to work aimed at designing two-thumb text entry on touchscreen devices [Oulasvirta et al. 2013], we enhance text-based applications using gaze by providing more functionality while keeping the UI simple. Simplification also leads to less effort in using the hand during the use of the tablet. The introduced GazeButton can be manipulated in various ways to enhance users' experience. For simplicity and for making interaction intuitive, the following points should be considered in the concept design of applications.

3.1.1 Where we Look. Gaze, as sensed by the eye tracker, provides a 2D position on the screen, to infer *where* we look. Based on where a user looks, different features can be enabled. With respect to the GazeButton, a designer has the option to implement different features depending on whether the user looks at the GazeButton, or elsewhere within the application.

3.1.2 Where we Touch. Similar to gaze, touch gestures can enable various features as well, hence making it important to consider *where*. On the most basic level, the application distinguishes touch to be sensed by GazeButton or elsewhere within the application area. The button, again, allows a user to either touch in other areas or touch the button for more options.

3.1.3 Where we Look and Touch. Taken together, the two modalities can be represented in four variations when considering the spatial relation to the GazeButton. Initially the default touch input is enabled for a button. When users touch the GazeButton, the system considers where the user is looking. There are two distinct ways: 1) the user is also looking at the button – then a secondary level menu is brought up, like a Windows start button; 2) the user is looking somewhere else on the screen – in this case, a different action related to what the user is looking at is initiated.

An issue may be that users are less precise to touch the GazeButton when they are looking somewhere else in the UI. A potential solution may be to increase the selection space of the button the further users are looking away. Other approaches are simply using a larger button (at the expense of UI space) or using a tactile button. Alternatively, other input handling methods can allow users to better understand where they touch [Serim and Jacucci 2016].

3.1.4 Gestures we use. We can also take advantage of touch gestures users can perform on the button to increase the expressiveness of the GazeButton. The gestures are straightforward movements, such as drag, hold or tap. Tapping is normally mapped to a default action (i.e., selecting what you can see), while holding can be mapped to tasks that involve duration such as selecting an area. Lastly, movement in basic directions can be mapped to secondary functions that can aid the user's task, such as mode switching.

3.1.5 Areas we look at. Since our main two features used for interaction are touch and gaze, the combination of where we look and touch provides us with various functionality. Where the user looks actively during the use of the application leads to different results. The gazing area can be subdivided into three parts: over keyboard, over text, over GazeButton.

The dimensions are illustrated in Figure 1. As can be seen in the four images,we represent the button and the target UI. The figures represents the combination of interaction techniques a person can use. Depending on the purpose, the user can look at the button and touch elsewhere or touch the button and gaze elsewhere. We also divide the total operational area where the user can gaze or touch into separate parts. The first area is the button itself. In addition, we have the soft keyboard and the text area.

3.2 System Input Interpretation

At the system level, input events are provided from the sensors of the input devices. Here, the position is most relevant. We describe this for the example of a text writing and editing tool. The input state diagram in Figure 2 shows the how different inputs may lead to a different action. However, from the user's view, the user always interacts on what they are looking at, making it easy to understand what happens when. The diagram provides an overview on how the input events of gaze and touch are interpreted by the system.

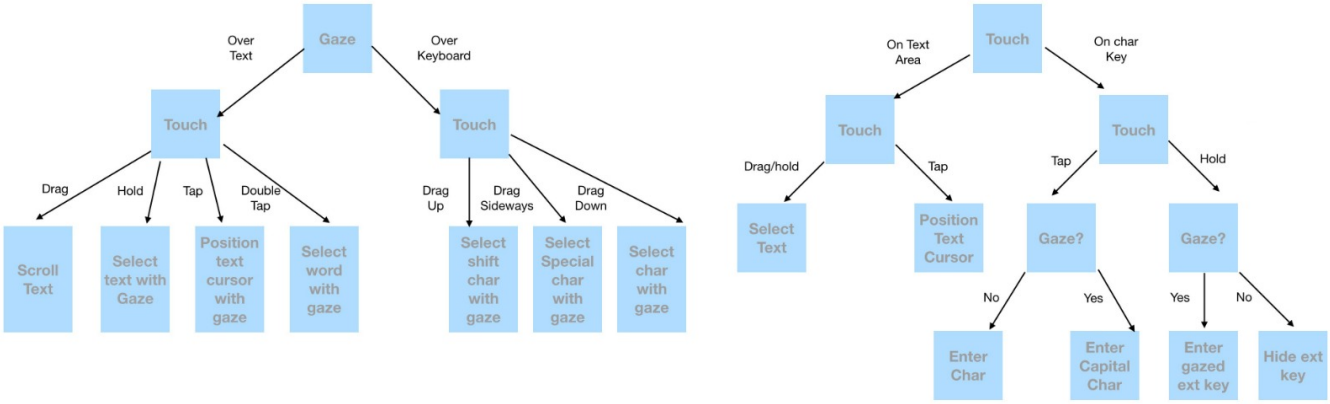

Figure 2: Input state diagram showing how the the prototype examples map to features with the GazeButton.

The first part on the left illustrates how input events are interpreted from receiving an event to providing an appropriate action. Applications would not need to implement all of them. Our goal here is to provide some examples to demonstrate possibilities.

This model is representing the mapping of the top right part of Figure 1. This represents the combination when a user is performing a touch gesture on the button. Based on where the user is gazing, different functionality is provided to the user. If the user gazes over the text, then one can perform gestures such as "drag", "hold", "tap" and "double tap" on the button to perform operations like selecting text with gaze or position the text cursor with gaze. On the other hand, if the user is gazing over the keyboard, then performing gestures such as drag up, down or sideways on the button can allow the user to select special characters or shift characters with gaze.

The second model illustrates two combinations. The first combination is the one which allows a user to gaze at the area and also touch the area which maps the top left part of Figure 1. The available options for touching are either on the text area or the keyboard. If the user touches the text area, as illustrated in the model, then the user can either "drag" or "tap" to select text and position the text cursor. On the other hand, if the user touches a character key, then the user can either "hold" or "tap" for further options. Provided that a user holds the touch in the area and gazes at a key, then a character is entered. This is an example of the combination of gazing and touching over an area. However, if the user does not gaze after performing "hold" over an area, then a capital character is entered, illustrating the example of gazing at the button while touching the area (cf. bottom left part of Figure 1).

The demonstrated implementations enable quick interaction with the application while requiring only minimal movement. Since all interactions are enabled with just one button, interactions involve little effort for the user.

4 TEXT APPLICATION EXAMPLES

In the following, we demonstrate the GazeButton concept in the context of a text writing and editing application on the tablet. Here, we introduce several variations of interaction techniques. The button is located at the left side of the tablet, close to where the left hand would have a firm grip. For right handers, the system could provide an option to switch or use grip-sensing technology [Cheng et al. 2013; Hinckley et al. 2016]. As our focus is on illustrating our concept, we use a larger size for the text font to try the interactions first without any influence from potential eye-tracking accuracy challenges [Khamis et al. 2018a].

The text application consists of basic UI components: a text field at the top and a keyboard at the bottom with the design being inspired by the interfaces of current keyboard designs of tablets. The GazeButton is located at the bottom left, as it can be easily reached by the fingers while holding the device. Thus, users can utilise the tablet with the GazeButton functionality, using one-handed interaction. We present our examples based on the conceptual categories as illustrated in Figure 1.

4.1 Implementation

The application is implemented using a Surface Pro 3 tablet with a Tobii 4C eye tracker (90 Hz). The tablet has an i3 core with 1,5 GHz and a 4 GB RAM. The tracker is placed at the bottom of the tablet's screen, oriented in landscape mode. The software is implemented in Java with Processing. The simultaneous interaction using touch and gaze is done by interpreting the occurring input events in a way as detailed in the following examples. The text characters are drawn each as a separate object to enable us to set the cursor between them. We used a font that has a ratio of screen width divided by 10 to be large enough to interact with gaze as accurately as possible. The keyboard layout is similar to the normal keyboard, using the same character positions and dimensions. This is done to add the extra features for the buttons and to enable both touch and gaze. Lastly we also added gaze data smoothing. We use a moving average over the last four gaze samples, to ease gaze selections in addition to the smoothing that comes with the Tobii's software by default.

4.2 Look at UI, Touch Button

4.2.1 Over Text. The following examples deal with interactions that users can perform when they use the button and their visual focus lies within the text field where users entered text before. This is in accordance with the top right part of Figure 1. Depending on the touch gesture performed on the GazeButton, users can employ four different features of the application.

First, most simply, users can perform a drag gesture which will lead to scrolling the text up and down. This allows users to quickly navigate the written text, instead of the necessity of moving the hand upwards to the text field. Second, users can perform selection of text with gaze by performing a hold gesture on the GazeButton. This functionality is again to help the user perform selections easily. Third, a tap gesture on the button enables the user to position the text cursor with gaze which makes it effortless on the part of the user from the point of view of typing. An illustration of this design implementation is provided in Figure 3. As can be seen in the diagram, while performing a tap on the button in combination with the gaze on the text area successfully changes the cursor position in the text. Fourth, users can apply double taps on the button while selecting a word with gaze. The implementation of this design is highlighted in Figure 4: performing a double tap allows a word to be selected in the gaze direction over the text area.

Figure 3: Change text cursor position with gaze and touch. The user looks at where the cursor should be and then touch/tap the GazeButton.

These features all allow a user to quickly perform add-on features to typing with minimal hand movement on a tablet. The interactions are easy, given the location of the button on the application while a gaze can be performed easily on the tablet area without any physical movement.

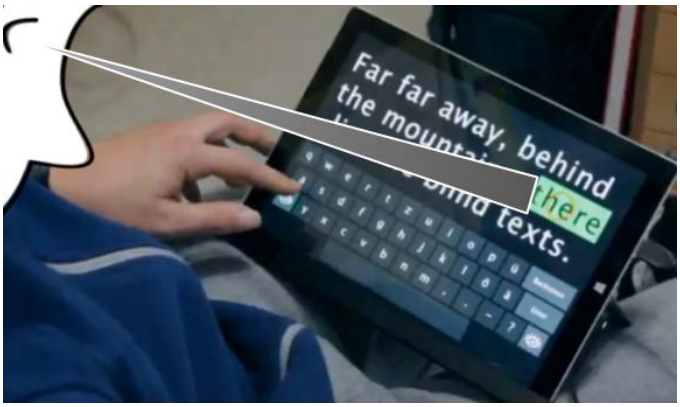

Figure 4: Highlighting/Selecting a word with gaze can be done by looking at the word and double-tapping the GazeButton.

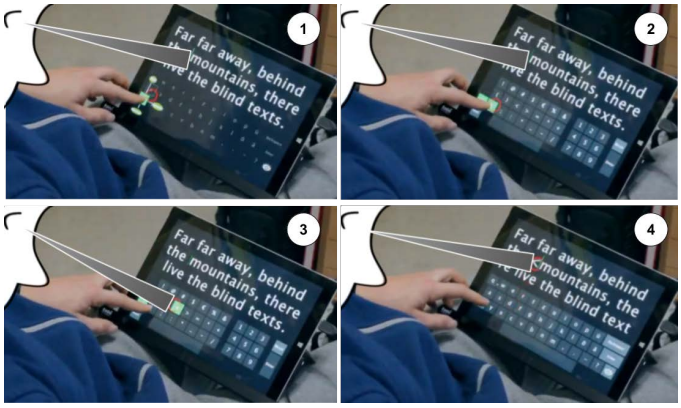

Figure 5: Selecting special character with gaze. First (1), the user has to select where he will enter then click on the GazeBuuton. Then (2), to enter a special character the user will slide right to change the keyboard mode. After that (3), the user can choose which character to enter and finally (4), the character is typed in the text.

4.2.2 Over Keyboard. The following examples, which are also in accordance with the top right part of Figure 1, refer to interactions which can be performed by the user when they touch the button while their visual focus is on the virtual keyboard. Similar to previous examples which are based on the touch gesture and gaze, three different features of the application can be employed here also. First, the user can select shift character with gaze with a gesture of dragging up by touching the button. Second, dragging sideways allows the user to select special character with gaze. An example is provided in Figure 5, where a user is seen dragging the button sideways. This drag motion allows the user to select a special character from the virtual keyboard which gets inserted in the text area in the position of the cursor. Thirdly, any character can be selected with gaze with the gesture of dragging down.

These designs, similar to all other implementation demonstrated above, help keeping hand movement to a minimum while enhancing user experience. It also highlights how changing the gaze area allows designers to implement novel techniques featuring various functionality, thus maximising design efficiency.

4.3 Look at UI, Touch UI

This subsection illustrates examples corresponding to the top left part of Figure 1, which infuse interactions that users can perform when both their visual focus and touch target lies within the UI. Depending on the gestures, users can trigger four features of the application. In line with the theme of our work, these features have been designed to provide interaction with minimal movements. The advantage is that all features can easily be merged into a UI.

If the user touches the text area and performs either a drag or hold gesture, then either text selection is enabled and the other feature is to position text cursor using the tap gesture. Another alternative is to touch the character key which gives the user the option to either hold or tap and perform gazing. Depending on whether the user performs a gaze operation by tapping, the user can either the enter extension key or hide extension key (Figure 6).

Figure 6: Selecting extension keys with gaze. On the left, the user presses on the desired key to get its extension. On the right, the user gaze on the desired extension letter.

4.4 Look at Button, Touch UI

We have implemented one feature in our application for the use case when the visual focus of the user is at the button while touching the UI (corresponding to the bottom left part of Figure 1). When the user touches the character key, and applies the hold gesture with gaze option, then a capital character is inserted in the text area as shown in Figure 7.

Figure 7: Selecting upper case character with gaze. On the left, the user first has to gaze at the GazeButton to change the keyboard to the upper case mode. Then on the right, the user touches the desired key.

5 DISCUSSION

GazeButton has many applications and can be useful in different areas for the design of user interfaces. GazeButton allows for experimenting with novel and expressive ways of interaction that eye-tracking technology can enable. It is only one button that can be used at any time, where the user gets their most common interactions done with a short touch gesture. We also provided a description on how one can integrate the button in an existing touchscreen UI.

The idea of the GazeButton can be extended to other form factors and application areas. It can be integrated in websites as a universal button to provide extra information or to do search on the selected or gazed area. It can be used as an indirect handle in drawing applications as a shortcut to menus, e.g., to change the colour, font, or size. In desktop applications, we can consider doing different mouse gestures over it like cross horizontally, cross vertically, or X sign. On mobile devices, they can be used, e.g., in case of single hand usage on phones with larger screens than the thumb can reach. In some situations it can be tedious with a phone, when one cannot reach parts of the phone display with their thumb, so that the user hand needs to be engaged, e.g., selecting an app, clicking a button, positioning the text cursor in a chat message, or clicking a web link.

Overall, it has the potential to enhance accessibility on large scree and shorten interaction time. To better take account of what the user is looking at during touch actions, GazeButton provides a simple touchpad-like hub. In this context, the techniques we demonstrated are only a small subset of the possible interactions. Yet we believe it enhances users' experience and usability of tablets.

As next step we plan to study the interaction of users with the system. We will compare it to the default touch input with two hands and whether users report issues or benefits from the system. It is not entirely clear if people will be able to use the gaze part as we use a lot of visual feedback (e.g. arrows appear when GazeButton is pressed). In addition, we will consider techniques enhancing gaze precision to allow interaction with smaller fonts.

6 CONCLUSION

GazeButton is a novel button concept that extends the default button mode with advanced gaze-based interactions. In this paper we explained the concept of GazeButtons and how it can be used by touch or gaze. We discussed and reflected on its four dimensions, expressiveness, complementarity, integration, and reach. We created a prototype ad demonstrated how the idea can be used. We also came up with four possible dimensions with their usage. We provided examples of how GazeButtons can be used as a text editing tool on tablets for three of the four dimensions which are look and touch on the text area/keyboard, look at the GazeButton and touch the text area/keyboard, look at the text area and touch the GazeButton and finally, look and touch the GazeButton. We showed different examples, such as cursor positioning, text highlighting, and keyboard switching. We also discussed how GazeButton can be used on desktop PCs and mobile phones. Finally, we sketched alternative use cases beyond text editing.

In the next steps, the concept will be evaluated by users to test its usability and learnability. In addition, we will address difficulties that occur as a result of the interaction modality. Finally, we will compare between different font sizes versus speed and accuracy of the interaction modality.

REFERENCES

Roman Bednarik, Tersia Gowases, and Markku Tukiainen. 2009. Gaze interaction enhances problem solving : Effects of dwell-time based , gaze-augmented , and mouse interaction on problem-solving strategies and user experience.

Joanna Bergstrom-Lehtovirta and Antti Oulasvirta. 2014. Modeling the Functional Area of the Thumb on Mobile Touchscreen Surfaces. In *CHI '14*. ACM, 1991–2000. https://doi.org/10.1145/2556288.2557354

Richard A. Bolt. 1981. Gaze-orchestrated Dynamic Windows. *SIGGRAPH Comput. Graph.* 15, 3 (Aug. 1981), 109–119. https://doi.org/10.1145/965161.806796

Andreas Bulling and Hans Gellersen. 2010. Toward mobile eye-based human-computer interaction. *IEEE Pervasive Computing* 4 (2010), 8–12.

Lung-Pan Cheng, Hsiang-Sheng Liang, Che-Yang Wu, and Mike Y. Chen. 2013. iGrasp: Grasp-based Adaptive Keyboard for Mobile Devices. In *CHI '13*. ACM, 3037–3046. https://doi.org/10.1145/2470654.2481422

Heiko Drewes, Alexander De Luca, and Albrecht Schmidt. 2007. Eye-gaze Interaction for Mobile Phones. In *Proceedings of the 4th International Conference on Mobile Technology, Applications, and Systems and the 1st International Symposium on Computer Human Interaction in Mobile Technology (Mobility '07)*. ACM, New York, NY, USA, 364–371. https://doi.org/10.1145/1378063.1378122

Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2015. Orbits: Gaze Interaction for Smart Watches Using Smooth Pursuit Eye Movements. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 457–466. https://doi.org/10.1145/2807442.2807499

Ken Hinckley, Seongkook Heo, Michel Pahud, Christian Holz, Hrvoje Benko, Abigail Sellen, Richard Banks, Kenton OâĂŹHara, Gavin Smyth, and Bill Buxton. 2016. Pre-Touch Sensing for Mobile Interaction. In *CHI '16*. ACM, to appear.

Oliver Hohlfeld, André Pomp, Jó Ágila Bitsch Link, and Dennis Guse. 2015. On the applicability of computer vision based gaze tracking in mobile scenarios. In

Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services. ACM, 427–434.

Anke Huckauf and Mario H. Urbina. 2008. Gazing with pEYEs: Towards a Universal Input for Various Applications. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications (ETRA '08)*. ACM, New York, NY, USA, 51–54. https://doi.org/10.1145/1344471.1344483

Robert J. K. Jacob. 1990. What You Look at is What You Get: Eye Movement-based Interaction Techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*. ACM, New York, NY, USA, 11–18. https://doi.org/10.1145/97243.97246

Mohamed Khamis, Florian Alt, and Andreas Bulling. 2018a. The Past, Present, and Future of Gaze-enabled Handheld Mobile Devices: Survey and Lessons Learned. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '18)*. ACM, New York, NY, USA, Article 38, 17 pages. https://doi.org/10.1145/3229434.3229452

Mohamed Khamis, Ludwig Trotter, Ville Mäkelä, Emanuel von Zezschwitz, Jens Le, Andreas Bulling, and Florian Alt. 2018b. CueAuth: Comparing Touch, Mid-Air Gestures, and Gaze for Cue-based Authentication on Situated Displays. *IMWUT* 2, 4 (2018), 174:1–174:22. https://dl.acm.org/citation.cfm?id=3287052

Andrew Kurauchi, Wenxin Feng, Ajjen Joshi, Carlos Morimoto, and Margrit Betke. 2016. EyeSwipe: Dwell-free text entry using gaze paths. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 1952–1956.

Päivi Majaranta and Kari-Jouko Räihä. 2002. Twenty years of eye typing: systems and design issues. In *Proceedings of the 2002 symposium on Eye tracking research & applications*. ACM, 15–22.

Dan Odell and Vasudha Chandrasekaran. 2012. Enabling comfortable thumb interaction in tablet computers: A windows 8 case study. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 56. SAGE Publications, 1907–1911.

Antti Oulasvirta, Anna Reichel, Wenbin Li, Yan Zhang, Myroslav Bachynskyi, Keith Vertanen, and Per Ola Kristensson. 2013. Improving Two-thumb Text Entry on Touchscreen Devices. In *CHI '13*. ACM, 2765–2774. https://doi.org/10.1145/2470654.2481383

Ken Pfeuffer, Jason Alexander, Ming Ki Chong, and Hans Gellersen. 2014. Gaze-touch: Combining Gaze with Multi-touch for Interaction on the Same Surface. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 509–518. https://doi.org/10.1145/2642918.2647397

Ken Pfeuffer, Jason Alexander, Ming Ki Chong, Yanxia Zhang, and Hans Gellersen. 2015. Gaze-Shifting: Direct-Indirect Input with Pen and Touch Modulated by Gaze. In *UIST '15*. ACM, 373–383.

Ken Pfeuffer and Hans Gellersen. 2016. Gaze and Touch Interaction on Tablets. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 301–311. https://doi.org/10.1145/2984511.2984514

Ken Pfeuffer, Ken Hinckley, Michel Pahud, and Bill Buxton. 2017. Thumb + Pen Interaction on Tablets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 3254–3266. https://doi.org/10.1145/3025453.3025567

Vijay Rajanna and John Paulin Hansen. 2018. Gaze Typing in Virtual Reality: Impact of Keyboard Design, Selection Method, and Motion. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. ACM, New York, NY, USA, Article 15, 10 pages. https://doi.org/10.1145/3204493.3204541

Korok Sengupta, Raphael Menges, Chandan Kumar, and Steffen Staab. 2017. Gazethekey: Interactive keys to integrate word predictions for gaze-based text entry. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces Companion*. ACM, 121–124.

Baris Serim and Giulio Jacucci. 2016. Pointing While Looking Elsewhere: Designing for Varying Degrees of Visual Guidance During Manual Input. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5789–5800. https://doi.org/10.1145/2858036.2858480

Ben Shneiderman. 1983. Direct Manipulation: A Step Beyond Programming Languages. *Computer* 16, 8 (Aug. 1983), 57–69. https://doi.org/10.1109/MC.1983.1654471

Linda E. Sibert and Robert J. K. Jacob. 2000. Evaluation of eye gaze interaction. In *CHI*. ACM, 281–288.

Sophie Stellmach and Raimund Dachselt. 2012. Look & Touch: Gaze-supported Target Acquisition. In *CHI '12*. ACM, 2981–2990. https://doi.org/10.1145/2207676.2208709

Tobii. 2019. Tobii EyeX. In *https://gaming.tobii.com/products/peripherals/*.

Matthieu B Trudeau, Paul J Catalano, Devin L Jindrich, and Jack T Dennerlein. 2013. Tablet keyboard configuration affects performance, discomfort and task difficulty for thumb typing in a two-handed grip. *PloS one* 8, 6 (2013), e67525.

Outi Tuisku, Päivi Majaranta, Poika Isokoski, and Kari-Jouko Räihä. 2008. Now Dasher! Dash away!: longitudinal study of fast text entry by Eye Gaze. In *Proceedings of the 2008 symposium on Eye tracking research & applications*. ACM, 19–26.

Jayson Turner, Jason Alexander, Andreas Bulling, and Hans Gellersen. 2015. Gaze+RST: Integrating Gaze and Multitouch for Remote Rotate-Scale-Translate Tasks. In *CHI '15*. ACM, 4179–4188. https://doi.org/10.1145/2702123.2702355

Jayson Turner, Andreas Bulling, Jason Alexander, and Hans Gellersen. 2014. Cross-device Gaze-supported Point-to-point Content Transfer. In *ETRA'14*. ACM, 19–26.

Jayson Turner, Andreas Bulling, and Hans Gellersen. 2011. Combining Gaze with Manual Interaction to Extend Physical Reach. In *PETMEI '11*. ACM, 33–36. https://doi.org/10.1145/2029956.2029966

Julie Wagner, Stéphane Huot, and Wendy Mackay. 2012. BiTouch and BiPad: Designing Bimanual Interaction for Hand-held Tablets. In *CHI '12*. ACM, 2317–2326. https://doi.org/10.1145/2207676.2208391

Katrin Wolf and Niels Henze. 2014. Comparing Pointing Techniques for Grasping Hands on Tablets. In *MobileHCI '14*. ACM, 53–62. https://doi.org/10.1145/2628363.2628371

Katrin Wolf, Markus Schneider, John Mercouris, and Christopher-Eyk Hrabia. 2015. Biomechanics of Front and Back-of-Tablet Pointing with Grasping Hands. *Int. J. Mob. Hum. Comput. Interact.* 7, 2 (April 2015), 43–64. https://doi.org/10.4018/ijmhci.2015040103

Erroll Wood and Andreas Bulling. 2014. Eyetab: Model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications*. ACM, 207–210.

Shumin Zhai, Carlos Morimoto, and Steven Ihde. 1999. Manual and gaze input cascaded (MAGIC) pointing. In *CHI'99*. ACM, 246–253.