T-Force: Exploring the Use of Typing Force for Three State Virtual Keyboards

Shariff AM Faleel*
amshamoh@student.ubc.ca
University of British Columbia - Okanagan
Kelowna, Canada

Yishuo Liu
liuyishuo@huawei.com
Huawei HMI Lab
Shenzhen, China

Roya A Cody
roya.cody1@huawei.com
Huawei HMI Lab
Toronto, Canada

Bradley Rey
bradley.rey@ubc.ca
University of British Columbia - Okanagan
Kelowna, Canada

Linghao Du
linghao.du@huawei.com
Huawei HMI Lab
Toronto, Canada

Jiangyue Yu*
jiangyue.yu@mail.utoronto.ca
University of Toronto
Toronto, Canada

Da-Yuan Huang
dayuan.huang@outlook.com
Huawei HMI Lab
Toronto, Canada

Pourang Irani
pourang.irani@ubc.ca
University of British Columbia - Okanagan
Kelowna, Canada

Wei Li
wei.li.crc@huawei.com
Huawei HMI Lab
Toronto, Canada

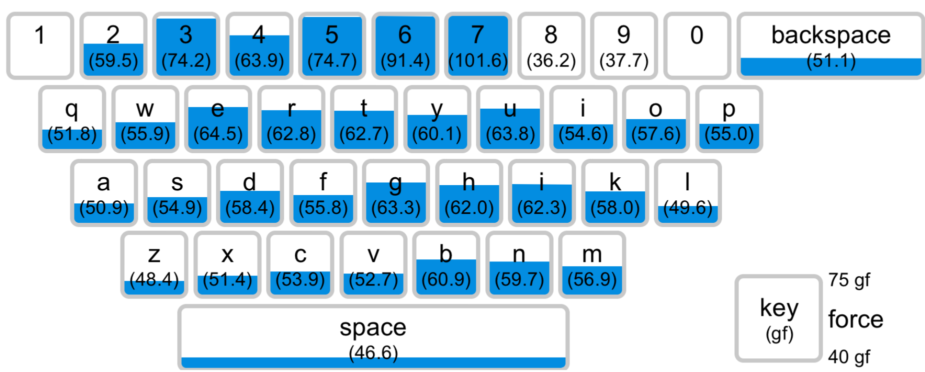


Figure 1: Mean maximum forces applied to keys across participants, captured from a three-state, force sensitive, virtual keyboard while 10-finger typing. Our analysis shows that using a force threshold function (i.e., T-Force) allows us to propose methods for classifying touch events between keypresses and non-keypresses. Furthermore, accommodating the variations seen, we can further improve the utilization of force in classification.

ABSTRACT

Three state virtual keyboards which differentiate contact events between released, touched, and pressed states have the potential to improve overall typing experience and reduce the gap between virtual keyboards and physical keyboards. Incorporating force sensitivity, three-state virtual keyboards can utilize a force threshold to better classify a contact event. However, our limited knowledge of how force plays a role during typing on virtual keyboards limits further progress. Through a series of studies we observe that using a uniform threshold is not an optimal approach. Furthermore, the force being applied while typing varies significantly across the keys and among participants. As such, we propose three different approaches to further improve the uniform threshold. We show that a carefully selected non-uniform threshold function could be sufficient in delineating typing events on a three-state keyboard. Finally, we conclude our work with lessons learned, suggestion for future improvements, and comparisons with current methods available.

*Work done during intership at Huawei HMI Lab, Toronto, Canada.

CHI '23, April 23–28, 2023, Hamburg, Germany

ACM ISBN 978-1-4503-9421-5/23/04…$15.00
https://doi.org/10.1145/3544548.3580915

CCS CONCEPTS

• **Human-centered computing** → *Empirical studies in HCI*; **Keyboards**; *Touch screens*; *User centered design.*

KEYWORDS

three-state Virtual keyboard, force sensitive touch interactions, 10-finger typing

ACM Reference Format:

Shariff AM Faleel, Yishuo Liu, Roya A Cody, Bradley Rey, Linghao Du, Jiangyue Yu, Da-Yuan Huang, Pourang Irani, and Wei Li. 2023. T-Force: Exploring the Use of Typing Force for Three State Virtual Keyboards. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3544548.3580915

1 INTRODUCTION

The widespread adoption of mobile devices immediately brought rise to the use of virtual keyboards. To accommodate a reduction in performance over dedicated physical keyboard peripherals, research has explored many methods to enhance text-entry on touch surfaces. In particular, studies have explored layout [28, 46], algorithmic enhancements [13, 29], interaction modalities [31], comfort [1, 35], optimizations [52] and improved feedback [44]. One key advantage enabled by virtual keyboards has been to leverage their software-defined nature to suit each user's specific typing preferences and profile [11, 26, 52, 54]. Now, with the possibility of force-sensitive touch screens, such as in Apple's 3D touch interactions, recent attention has shifted to exploring the use of force for keyboard input [57–59].

The lack of physical landmarks on virtual keyboards makes it difficult for users to build haptic typing memory [25, 42]. Furthermore, virtual keyboards which primarily operate under two states (i.e., whether or not a key has been touched and released) do not allow users to rest their hands or fingers on the screen [26], making this medium tedious for lengthier text-entry tasks [2, 35]. In fact, resting on the keyboard has been found to be critical for efficient 10-finger typing [6, 7, 47]. Despite this, the majority of prior work focuses on improving the two-state keyboard model mostly due to the hardware limitations. Recent works that explore force-sensitive touch surfaces to overcome these challenges use force as an input for statistical decoding along with a mechanism to filter out accidental or irrelevant touch events [14, 57]. It is worth noting that these approaches use force as one of many features within data-driven models to improve typing performance on virtual keyboards. However, they do not explore characteristics of force behaviour and its effects on three-state functionality. This insight can largely improve performance, in terms of typing efficiency and user experience, and provide a more simple and robust means for discerning *touch* and *press* states simply through force alone.

A three-state keyboard interface allows for two key benefits: 1) Users are afforded the ability to rest their hands and fingers on the touch surface, similar to common interactions with physical keyboards; 2) The system would be able to provide feedback to the user on events other than the keypress itself (e.g., feedback on key gaps and resting fingers along the home row). This additional feedback is not possible with two-state virtual keyboards [14, 26], where most, if not all, contact events are treated as input events. To achieve these benefits and an efficient 10-finger typing experience on a three-state virtual keyboard, we first need further insight into force characteristics across the keyboard while resting and typing. Specifically, we focus on understanding and exploring the impact of using force data for the purpose of classifying touched and pressed states. As such, we focus on two research questions within this work: **RQ1:** What are the force characteristics present when interacting on a virtual keyboard? **RQ2:** Can we use force values to effectively distinguish between touched and pressed states [26], allowing users to rest without accidentally triggering keystrokes and type without missing keystrokes. Answers to these questions combined with novel approaches to improving virtual keyboards such as data-driven models [4, 8, 12, 49] and feedback mechanisms [22, 38, 42, 44] could potentially further bridge the gap between physical and virtual keyboards.

We explore these research questions through a sequence of user studies. In two initial data collection studies (N=10 and N=12) we observe how users rest and type on a flat touch-sensitive keyboard. From the collected data, we first derive a flat force threshold. The flat threshold is used to classify a contact event on the touch surface as either *touched* or *pressed*, where a keystroke is only triggered in the pressed state past the force threshold. In a third study (N=12), we collect typing data while utilizing the flat force threshold. Our analysis of the forces exerted show that there is an overlap between resting and typing forces and having a flat threshold can lead to false positives (incorrectly classified as pressed) and false negatives (incorrectly classified as touched). We also observe that there is a significant variation in the force being applied across different regions of the keyboard and between participants. As such, we derive three alternative approaches to having a flat threshold that can potentially improve the balance between false positives and false negatives when typing on a virtual keyboard: (1) Personalized threshold (PT), (2) Non-uniform threshold (NUT), and (3) Dynamic threshold (DT). In a final consequent study (N=12), we explore these new force threshold functions within a typing task, where we observe the new functions individually having an impact on the number of false negatives as well as improving the user experience over a constant threshold function.

Our contributions are threefold. **C1:** We present three user studies exploring force characteristics when resting and typing on a virtual keyboard. Through detailed analysis of captured data, we show that a threshold function based only on force could be sufficient to define a three-state virtual keyboard. We further demonstrate that the force exerted by typing on a virtual keyboard is not uniform across the keyboard or among participants. **C2:** We propose three approaches for modifying the threshold function to account for such variation and conduct a fourth user study for validation. **C3:** We conclude with discussion of lessons learned through our approach, comparisons to current methods, and how future work can further improve on our approaches to further enhance the user experience. Overall, our results aim to provide insight into how future three-state virtual keyboards with force-sensitive surfaces can be designed.

2 RELATED WORK

We review work on methods for improving 10-finger virtual keyboard typing performance and on three-state virtual keyboards with a focus on the addition of force. Studies have explored 10-finger virtual keyboards as they become more prevalent. These keyboards are often co-located on the same screen as the display itself [25], such as on tablets [28] or table tops [8, 52]. Odell et al. [44] observe co-location of text output and keyboard as a factor that influences typing performance. However, it is not obvious if this translates to non co-located scenarios. Additionally, there exists a large body of work focused on typing on smaller screens (e.g., for smartphones [18, 37] and smartwatches [16, 30]). While not directly applicable to our work, we discuss these where relevant.

2.1 Improving virtual keyboard performance

The lack of tactile or localized haptic feedback on most touch surfaces, negatively impacts the typing performance on virtual keyboards, causing hand drift [8, 33, 52] and finger touch misalignment on a straight row [12, 27, 57]. A large body of work on improving the performance of virtual keyboards has been built around such observations. Broadly, these can be separated into three major research directions:

1) Modelling Typing Behaviour. Due to a lack of tactile feedback [27], modelling typing behaviour attempts to reduce errors. Such, model-based approaches leverage the advantage of the virtual keyboard being software-defined. They use a range of different techniques. Modelling touch locations [11, 27] and finger movement [4, 49] attempt accommodate the inaccuracies resulting a touch location not matching the expected location. Statistical decoding [57] attempts to address this by predicting what the user would have meant to type and fixing errors in the output text. These approaches also include personalizing the model to fit the users [11, 52] to further reduce errors.

2) Feedback Mechanisms. Visual [19, 23, 51], auditory [23, 38], haptic [22, 38] and tactile [42, 44] feedback mechanisms have been studied and incorporated within virtual keyboards. With 10-finger typing, haptic and tactile feedback have gained more attention as they have been shown to have a larger impact on typing performance [6, 44]. Most of these approaches on providing feedback assume a two-state model of the touch surface, where a contact is either in a pressed or released state. As a result, the focus has been on providing feedback during touch- and release-based keystrokes. We note, additional information can also be encoded, such as how close to the edge of a key a keystroke takes place [17, 45]. However, this is not the case with touch typing on regular physical keyboards, which allow for a resting state [26], and as such a pressed state. Therefore, feedback alone is likely not sufficient to delineate resting and pressing on a three-state, force sensitive, keyboard

3) Resting on virtual keyboards. The desire, and also the need, to rest one's fingers on the keyboard has been studied with gesture typing on mobile devices [56, 60] and for different text-entry mechanisms [14, 26, 59]. However, this resting state has not been extensively studied in the context of 10-finger typing on virtual keyboards [14, 26]. As soft-keyboards proliferate the many devices we use (e.g., smartphones, tablets, tabletops, and in recent years laptops such as the Lenovo Yogabook), effectively addressing the ability to rest the hand while typing requires a shift in how we view touch-based virtual keyboards. To better enable resting, our work looks towards incorporating force to enable a third *pressed* state on virtual keyboards.

2.2 Three-state virtual keyboards and the use of force

Despite the literature on physical keyboards emphasizing the importance of being able to feel the keys and key-presses [6, 47], as well as to match with currently available consumer devices, many of the approaches to improve performance on virtual keyboards assume the two-state model. One such approach considered to improve two-state virtual keyboards is to disable touch input in the region where the palms are resting [12, 52, 57]. While this alleviates some of the issues that stem from constantly hovering over a touch-enabled surface, high variability in finger touch location [12, 52, 57] and fatigue when typing for long periods [50] remain.

Intuitively, the presence of a third state, combined with the dynamic nature of a virtual keyboard, can help reduce the gap between virtual keyboards and their physical counterparts. Once a virtual keyboard can effectively include a third resting state, related approaches on simulating textures [10], and localized haptic feedback [21, 24, 48] on flat surfaces can be utilized. One notable work on three-state virtual keyboards is TapBoard [26]. Within their prototype, a resting state is simulated by constraining the temporal dimension of a touch event. While this approach allows for the addition of more dimensions to the typing experience, it is still limited in terms of providing a third-state for events such as rollover, resting over a modifier key (e.g., shift), or pressing down while resting the finger without tapping. An alternative solution, which we explore in-depth, is to use a force-sensitive touch surface which can provide a more robust resting state. While current touchscreens do not embed force sensing, as seen in past devices, we can expect methods that beneficially leverage such an auxiliary channel to influence the inclusion of force sensing in future devices.

Many previous studies have made observations on the force or pressure being exerted when typing on virtual keyboards [12, 20, 57], though they do not explicitly incorporate a resting state. This resting state is crucial for when not actively typing [26] or when positioning fingers correctly [12]. Kim et al. [20] observed that the amount of pressure on a virtual keyboard is significantly lower when compared to physical keyboards. Yi et al. [57] and Gu et al. [14] successfully use pressure data as an input dimension to a statistical model for typing. Furthermore, Yi et al. [57] and Findlater et al. [12] observed that during continuous typing contact is rarely made with the virtual keyboard, unless performing a keystroke. This observation is also echoed by Kim et. al [26] in which they assume a keystroke is inherently a binary input in the form of a tap. We believe this stems from not having a robust interface that allows users to rest their fingers, combined with the bias of being acclimated to touch surfaces where any contact is treated as an input event. While data-driven approaches have shown promise in providing this functionality [14, 57] they don't provide insight into how different variables influence typing. We believe that gaining insight into how force as a specific variable can influence the classification process will help make these models more robust and simple.

3 METHODOLOGY

3.1 Study Platform

We developed a platform to collect force profile data and for use within our studies. The platform, built in Python, contains three primary components: force data collection, finger-to-key mapping, and the interface for typing tasks.

Force data collection. We used the Sensel Morph[1] to collect force data while typing. It has a 230 mm by 130 mm touch surface, with 185×105 sensor elements operating at 120 Hz. A keypress is simulated by our study platform when pressure is detected on a region corresponding to a given key. At any given time the Sensel Morph API provides data on readings from all sensor elements, which include the following for each contact event: estimated area and centroid of contact region, the total force in the contact region, and state of the contact (beginning of a contact, continuation of a contact event, and end of a contact event). The keyboard layout used in all studies is identical to the Lenovo Yogabook layout (see Figure 3).

Finger to key mapping. To observe the finger locations for key mapping we used MediaPipe[2] and OpenCV[3]. The mapping process is carried out through post-processing of captured videos. The video feed is captured using a standard web camera pointing down at the keyboard (the camera on the top of the center display in Figure 2). To calculate the finger-to-key mapping, a projection is calculated from the four edges of the Sensel Morph's touch surface in the video to a rectangular space. The fingertip locations from the MediaPipe are then projected to this rectangular space which is then mapped to the array output from the Sensel Morph's API. The finger-to-key mapping is calculated by getting the location of the projected fingertip that is closest to a given contact's centroid and used for calculating which finger triggers the corresponding contact.

Typing interface. The transcription typing interface presented to the participants can be seen in Figure 4 and was present on the vertical display in Figure 2. The interface shows the text to be transcribed on the top row and feedback from user input on the bottom row. Based on the task, the feedback and text shown could be modified. We had participants transcribe phrases from MacKenzie and Soukoreff [41], presented in random order during our studies.

3.2 Force function

To effectively characterize the force when typing, to distinguish between the *touched* and *pressed* states, we utilize the following function:

$$g(f) = s \in \{touched, pressed\} \tag{1}$$

Where f is the total force exerted for a given contact event, and s is the state of the contact. We refer to this function as *T-Force*. For a two-state virtual keyboard, the function g would be set to always return *pressed*.

[1]https://morph.sensel.com
[2]https://mediapipe.dev
[3]https://opencv.org

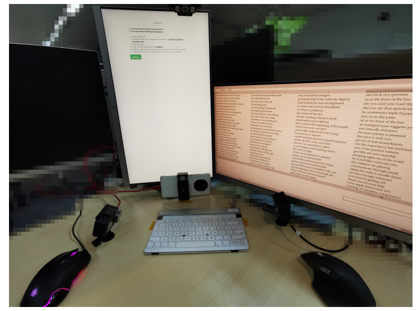


Figure 2: The study platform setup with the study web interface displayed in the center (adjusted to match each participant's eye-level). The camera on the top of this display was used for capturing finger locations. The display on the right contains the phrases used for the user study as described in Section 6.

Figure 3: The Sensel Morph used, with the printed layout attached for all studies. Highlighted with red ovals are the locations of the physical markers (used only when collecting resting data in Section 4.1) placed on the "f" and "j" keys as well as near the top of the force sensor.

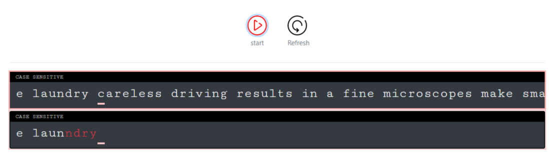


Figure 4: The user interface that was shown on a blank web page and seen during the typing tasks. The upper row shows the text to be typed. The lower row shows the participant's input. Depending on the task, the input shown is modified. In this instance, any errors are highlighted in red (used in Section 5.1 and Section 6.1).

4 DERIVING FORCE THRESHOLD

The purpose of the following two data collection studies is to explore the force characteristics first of resting fingers and palms and second of typing on the keyboard to derive a force threshold.

Though not the primary focus, during these studies we also consider additional factors (i.e., bias from constraining how the users may interact with force-sensitive touch surfaces) which we believe may influence the force exerted on the virtual keyboard. We aim to observe the maximum force exerted of a contact event, and we hypothesize there will be a significant variation in the force being exerted during typing. With force characteristics captured, we conclude this section with a derived constant force threshold for three-state virtual keyboards.

4.1 Resting forces and behavior

4.1.1 Design. We collected data across two conditions: (1) *wrist-on*, with the wrist on the Sensel Morph where index fingers rest on the top markers in Figure 3 and (2) *wrist-off*, with the wrist resting outside the force sensitive surface where index fingers rest on the bottom markers in Figure 3. We consider these two conditions separately to assess if there is a subconscious bias to exert less force when the wrists are not resting on the interaction surface. A within-subjects study design was utilized with the conditions being counterbalanced across participants. Within each condition, two trials were conducted for a total of four trials per participant.

4.1.2 Participants. We recruited 10 participants, with ages ranging from 21 to 31 ($M = 26, SD = 3.4$, 1 female). All participants were right-handed.

4.1.3 Procedure. Throughout, and before explaining the study procedure, participants are asked to be seated at the study platform. At the beginning of each condition a practice session, following the procedure mentioned below, was conducted using a physical keyboard. This was to practice the procedure to be experienced as well as to normalize one's self with their natural action of resting on a keyboard. Each of the two trials in each condition started with participants' hands completely off the force sensor. A trial was conducted by prompting the participant to place both hands on the keyboard with their index fingers on the markers fixed to the Sensel Morph for consistency (please see Figure 3) and rest as they would before typing on a regular keyboard. After five seconds, participants were instructed to move their right hand to a dummy keyboard placed on the right side of the force sensitive surface (simply used as an anchor point to rest away from the surface) and to rest for another five seconds. Then they were instructed to move their right hand back to the force sensitive surface. After another five seconds, a similar process was repeated for the left hand. Each participant spent five to ten minutes completing the entire study.

4.1.4 Results. To remove the noise resulting from accidental touches, only contact events that lasted more than 1 *second* were considered. This filtering resulted in 870 contact events. In the wrist-on condition, the wrist resting forces were excluded by removing the contacts that are closer to bottom left and right corners. The excluded regions can be seen in Figure 5 highlighted by rectangles, where 139 contact events were removed. Note that the wrist data should be interpreted with care, given the anatomy of the hand and total contact surface of the wrist, the force surface reads the wrist resting event as multiple contact events. This filtering resulted in a remaining 731 contact events. The effect of the two conditions on the log of the maximum force of a contact event, averaged by participant and condition, was analyzed with repeated measures ANOVA following testing for normality. The results show no significant difference between the wrist-on ($M = 19.8gf, SD = 13.05gf$) and wrist-off ($M = 22.72gf, SD = 12.61gf$) conditions ($F(1, 9) = 2.04, p = 0.19$).

4.2 Typing forces and behavior

4.2.1 Design. In the second data collection and observation study, participants were asked to type on the force sensitive surface. Here, the platform was set up to function as a two state keyboard across two conditions: (1) Unrestricted; this is similar to the unrestricted condition used by Findlater et. al [12]. In this condition, the force sensitive surface was left blank without a keyboard layout. As such, any contact event is considered a valid keystroke, and the typing interface shows only asterisk feedback. This condition allows us to capture the most natural typing behaviour of participants as it provides minimal constraints. (2) Restricted; in this condition, the printed keyboard layout was present on the force sensitive surface. Text feedback was provided on the transcription typing interface, with incorrectly typed letters being shown in red. This condition allows us to capture how participants type when they have to be more precise with the keystrokes, including correcting mistakes while typing. Again, a within-subjects design was utilized with the two conditions being counterbalanced across participants. Within each condition, three trials were conducted for a total of six trials per participant.

4.2.2 Participants. For this study, we recruited 12 participants, with ages ranging from 21 to 31 ($M = 26, SD = 3.4$, 2 female). All participants were right-handed.

4.2.3 Procedure. Participants were asked to be seated throughout the study. To familiarize themselves, participants performed a transcription typing task with the study platform on a physical keyboard. After taking a break, they started the trials. Throughout, we instructed participants to type as fast and as accurately as possible. Here, each trial lasted 65 seconds for an average session length of 10 to 15 minutes.

4.2.4 Results. We collected a total of 12517 contact events (4614 with the restricted and 7903 with the unrestricted condition). The contact events follow a common trend of raising to a peak value and gradually decreasing until the contact event ends. Also notable is that some keystrokes reach the peak value within the first frame of the contact, i.e, they reach the peak force value within 8ms. For statistical analysis, average maximum force of contact events for each participant for each condition was used. Since the data does not violate the normality constraint, repeated measures ANOVA was used to test the effect of the conditions used (restricted/unrestricted). Results shows no significant effect ($F(1, 11) = 0.355, p = 0.56$) between the restricted ($M = 62.1gf, SD = 31.8gf$) and unrestricted ($M = 36.26gf, SD = 21.9gf$) conditions.

4.3 Constant force threshold for a three-state keyboard

Figure 6 shows a summary of the forces observed from the data collected within our studies. We define the T-Force function in

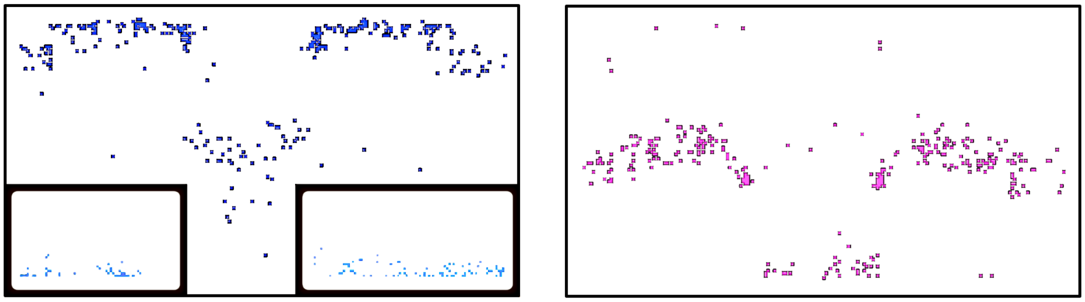


Figure 5: Participant contact points for the wrist-on (left) and wrist-off (right) conditions. In the wrist-on condition, the classified wrist contacts are highlighted by the rectangles in the bottom left and right corners.

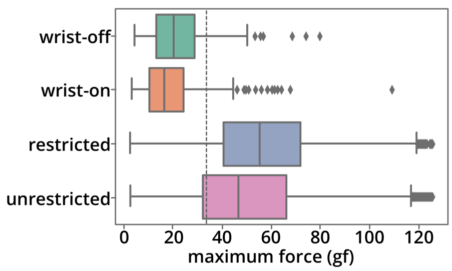


Figure 6: Box plot of the maximum force results from the two data collection studies split by the respective conditions used in both studies. The dashed line references the mean wrist force collected from the study in Section 4.1.

Equation 1 with a threshold T_f:

$$g(f) = \begin{cases} pressed & f > T_f, \\ touched & f \leq T_f \end{cases} \tag{2}$$

To define a constant force threshold. we use a linear Support Vector Machine (SVM) to estimate T_f using the maximum force data collected. To do this, we first removed the outliers (< 3 SD) from the labeled resting (Section 4.1) and typing data (Section 4.2). We utilized only two labels as no significant differences were found between the respective conditions in each study (i.e., wrist-off compared to wrist-on for the resting data and restricted compared to unrestricted for the typing data). Due to the data imbalance, we trained SVM models by randomly sampling 500 data points from each class (i.e., resting and typing). We trained 300 models, of which the top 100 performing models ($M = 0.82, SD = 0.008$) were used to estimate the threshold. The training data was then used for selection as we were interested in models that best fit the trained data. Since there is only one input feature, the hyperplane from the SVM models would be an approximation to T_f. These values are then averaged to obtain the final value for T_f, which is $31gF$ ($SD = 3.7gF$).

5 TYPING WITH A CONSTANT FORCE THRESHOLD

To understand how the T-Force function 2 performs and can be improved, we run another typing study to collect data with a three state virtual keyboard. In place of using the derived $31gF$, in this study, we use a lower value of $20gF$ for the T-Force function 2. Due to an overlap between the resting and typing data, a limitation of prior SVM-based estimated threshold is that the recall of the keystroke events from the typing data is 81%. That is, one in every five keystrokes would have been misclassified as "touched" states with the estimated threshold when using the T-Force function 2. Ideally, a three-state virtual keyboard will minimize the false positives and false negatives at the same time. Thus, we use a lowered threshold to observe keystrokes that would have resulted in false negatives and to also reduce the number of typing errors which could lead to frustration.

5.1 User study

5.1.1 Design. For this study we consider two tasks. First is a transcription typing task, where a participant types a random sequence of phrases similar to the second data collection study (Section 4.2). Second is the memory typing task, similar to the memorization tasks used by Varcholik et al. [53] and Gu et al. [14]. This task allows us to study a typing experience that is increasingly similar to one experienced in typical daily routines. To type from memory, we use the first sentence of the Gettysburg address. We expect there to be a difference in the force patterns between these two tasks. Note, that within this study we are not measuring typing performance, still only collecting force data for confirmation and further improvement of our force threshold function.

5.1.2 Participants. For this study, we recruited 12 participants, ages ranging from 21 to 31 ($M = 26.6, SD = 3.6$, 2 female). All participants were right-handed.

5.1.3 Procedure. To familiarize participants with the study setup, we first asked them to perform one warm-up trial with a physical keyboard. Throughout the procedure, we instructed participants to type the phrases shown on screen as accurately and quickly as possible. Throughout this study, the feedback provided was in plain text without highlighting any errors. First, the participants completed the transcription typing task across three sixty-five-second trials. While typing, the participants were shown a prompt to pause for 5 seconds after 30 seconds of continuous typing. This intermittent pause was introduced to observe the effect of pausing and resuming while typing (e.g., while thinking about what to type next when composing an email). Second, before administering the memorization task, the participants were given the phrase they were expected to memorize and type. To further familiarize themselves with the phrase, we provided them with three regular transcription typing tasks with the memorization phrase as the target phrase. This practice stage was not included in the analysis. Then, participants conducted three trials, with a maximum of 65 seconds for each trial, where they had to type the memorization phrase. The participants were allowed to take breaks between any of the trials. Sessions lasted between 30 to 40 minutes.

5.1.4 Results. We collected a total of 18953 contact events, of which 15098 were keystroke events. To determine a keystroke, we combine data from contact location on the Sensel Morph, finger tracking, and finally incorporate the forces captured. Since the data violated normality constraints, we conduct Kruskal-Wallis tests throughout.

We analyze the mean maximum force of all contact data as a dependent variable across the two task conditions of the study (transcription/memorization), the finger association for each keystroke (Figure 7b), and the 12 participants (Figure 7a). We observe finger association ($H = 52.3, df = 9, p < 0.01$) and participant ($H = 72.08, df = 11, p < 0.01$) to have a main effect on the mean maximum force of the contact events, while the two conditions did not ($H = 2.21, df = 1, p = 0.13$). Dunn's test with Bonferroni correction with the finger association shows that all significant differences are between the pairwise combinations between either the index, middle or ring finger combined with either the pinky finger or thumb. Dunn's test with Bonferroni correction with participants doesn't show any larger trends.

To gain insight into how the impact of using different fingers and their movements (e.g., flexion/extension of fingers) can be useful to improve the T-Force Function 2, we group the contact events based on the key under the location of the keystroke for analysis. We consider two factors: row of the key (Figure 7c), and column of the key (Figure 7d). The row of a key is one of either the numeric, top (row starting with "qwer.."), middle (row starting with "asdf.."), bottom (row starting with "zxcv.."), or the space-bar row. The columns were based on the common mapping of fingers to keys [4], which has a total of 9 groups (see Figure 8). Note, that though this grouping is referred to as *column* the space-bar in this case is considered as a separate region. The dependent variable in all cases was the maximum force applied during a contact event. We run two Kruskal-Wallis tests for each group to test their effect on the maximum force of each keystroke, averaged over the respective groups and participants. We observe a main effect for both factors. Post-hoc analysis with Dunn's test with Bonferroni correction show that with the rows ($H = 33.34, df = 8, p < 0.01$), the space-bar row has a significantly lower (p<0.01) maximum force compared to the other rows. A similar outcome can be observed with the results of the columns ($H = 20.39, df = 4, p < 0.01$), where the space-bar region is significantly lower (p<0.01) than all other columns. Among the rows, the only other significant difference is between the top row and the middle row (p=0.03). Among the columns, the results show no significant difference between combinations of columns 3, 4, 5, and 6 (p>0.05). Whereas differences are observed among the other pairs.

In the data collected, of the total contact time with the force surface, 24.6% of that time is from non-keystroke contact events. This includes unintentional touches, false negatives, and other typing behaviour related touch events such as rollover. To further expand on our results, we also analyze false negatives. We did not analyze false positives as the data collected does not allow us to directly observe this type of error (i.e., keystrokes that were false positives and errors due to general typing behavior cannot be distinguished). We further analyzed two types of false negatives. First, we include keystrokes below the derived T_f ($31gf$), which would have been registered as non-keystroke contact events. Second, the contact events below the used T_f ($20gf$) for the study, which should have been a keystroke which is extracted by filtering contact events that do not trigger a keystroke event, and are immediately followed by a keystroke which is a deletion error or the letter that would have been triggered as a keystroke. From the first type of false negatives, we have 1190 contact events (6.2%). From the second type of false negatives, we get 489 contact events (2.5%). Figure 10 shows all false negatives as a ratio to total keystrokes recorded in each respective group.

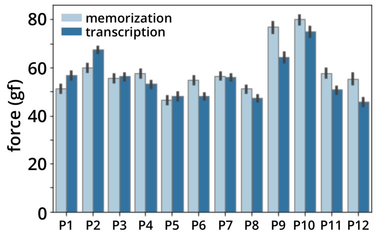
Correlation analysis between this ratio and the mean maximum force for the three factors show that there is a significant negative correlation between them for columns (r(7)=-0.85, p=.003) and participants (r(10)=-0.73, p=.005), but not for the rows (r(3)=-0.58, p=.29). Another observation that we make is the accuracy of using the derived T_f: the percentage of keystrokes whose maximum force applied is above the derived T_f of 31 gf is 91% compared to the 81% previously calculated. This could imply that participants adapt to requiring to exceed a threshold by typing harder. This also requires the analysis of the false negatives to be considered with care.

5.2 Improved T-Force functions

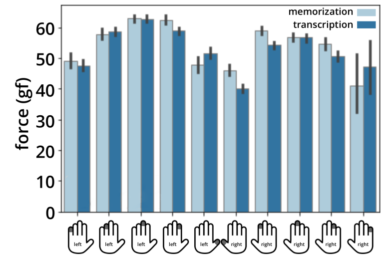
Results above inspired three different approaches for improving the T-Force Function 2:

(1) Personalized threshold: Motivated by the observation that different participants use different force levels, a simple approach is to set thresholds that are personalized to the participant. Utilizing individually captured *wrist-off* (Section 4.1) and *restricted* (Section 4.2) data, we followed the same procedure to that in Section 4.3 to derive a personalized threshold.

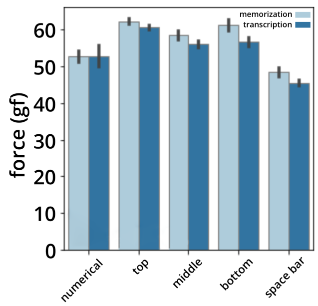
(2) Non-uniform threshold: As seen from the data grouped by columns and rows, different regions of the keyboard lead to keystrokes with different levels of force applied. Here, the threshold we utilize would differ based on the location of the keystroke on the keyboard. This could be as granular as having a threshold for each individual key (see Figure 1). A consideration with the grouping considered in the above analysis is the interaction between the two groupings. On average the spacebar has the lowest mean maximum force for both row and column, which also corresponds to the thumb's mean maximum force as seen in Figure 7b. Similarly, in regards to the numeric row, a large portion of the keystrokes are for the backspace (86%). In addition, we observe a larger variation between the character keys in column groupings and a significant correlation with the false negatives. Hence, for the non-uniform threshold, we chose to use the column groupings.



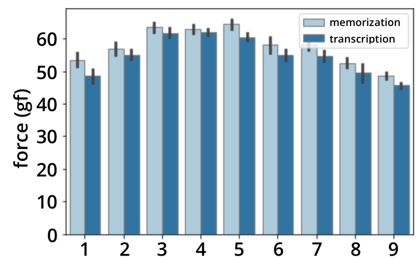
(a) Mean maximum force of keystrokes grouped by participants.



(b) Mean maximum force of keystrokes grouped by finger detected for the keystroke.



(c) Mean maximum force of keystrokes grouped by rows.



(d) Mean maximum force of keystrokes grouped by columns.

Figure 7: Mean maximum force grouped across dimensions from the user study with the three-state keyboard with the constant force threshold. Black lines denote 95% confidence intervals.

Figure 8: The *column* mapping of keys, used within the analysis.

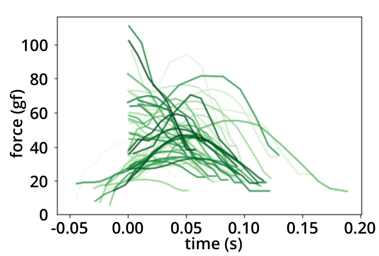


Figure 9: Force profile of 100 randomly sampled keystrokes. Please note, t=0 is when the keystroke was registered by the platform.

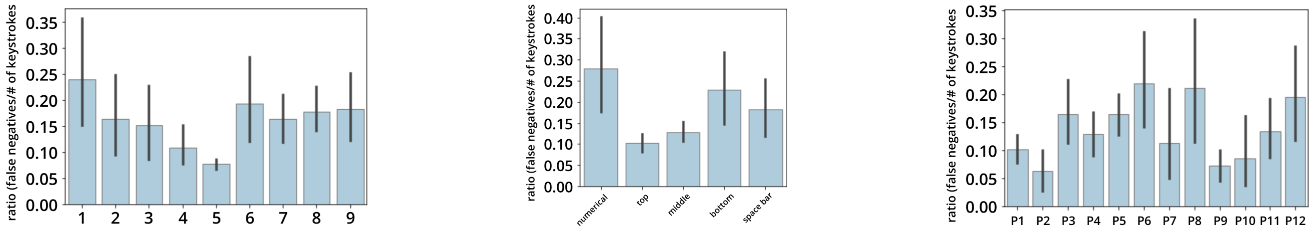
To derive the non-uniform threshold, we use the relative changes between the columns. We use the derived T_f of $31gF$ as the maximum threshold (for column 3) and the T_f used earlier ($20gF$) as the minimum threshold (for the space-bar column (column 9)), with the other columns thresholds set to the relative difference between the columns as seen in Figure 7d.

(3) Dynamic threshold. A dynamic approach allows for different thresholds to be used across contexts. An initial hypothesis, rejected by the data, was that the force being applied would differ based on the context of what is being typed. We tested the following scenarios: (1) Continuously typing for a period, expecting the force to lower over time. (2) Successive characters typed by the same finger. (3) Force applied after typing a space or correcting an error.

We chose to base the threshold on the force being applied over time, as seen in Figure 9. We observed that most keystrokes were taps, where the mean total contact time of a tap was 133 ms, with the median being 103 ms. Additionally, two force profiles for keystrokes were seen. First, a gradual increase and then decrease. Second, a quick peak within one frame (< 8 ms) and reducing from then on. Of the keystrokes data collected, 71% of them were triggered within the first frame following the second force profile discussed. Of the keystrokes that are below $31gf$, 37% of them are triggered in the first frame, where the mean total contact time ($0.09s$) of a contact event is less than the overall mean.

Motivated by these observations, and the approach used in Tapboard [26], we define the T-Force for dynamic threshold as follows:

We use $31gf$ for T_f, $20gf$ for $T_{f\_secondary}$ and $0.15s$ for $t_{secondary}$ which is the ~95% percentile of the total contact time distribution of keystrokes below $31gf$.



(a) Grouped by columns **(b) Grouped by rows** **(c) Grouped by participants**

Figure 10: False negatives as a ratio between the false negatives and the total number of keystrokes detected across dimensions during the user study with the three-state keyboard with a constant force threshold. Black lines denote 95% confidence intervals.

Algorithm 1: Function $g(.)$ with dynamic threshold

if $f > T_f$ **then**
  return *pressed*;
else if *contact ended & total contact time* $< t_{secondary}$ & $f_{max} > T_{f\_secondary}$ **then**
  return *pressed*;
else
  return *touched*;

6 TYPING WITH MODIFIED THRESHOLD FUNCTIONS

6.1 User study

We conducted a final user study to explore the modified threshold functions, false negatives and positives, and additional resting behaviour under increasingly natural typing scenarios. The modified functions are studied individually to assess how the variations perform in comparison to the uniform thresholds.

6.1.1 Design. Five conditions were tested, the three approaches defined in Section 5.2 and two baselines with uniform thresholds, *uniform-low* with the previously derived T_f (i.e., $31gF$) and *uniform-high* to mirror the typical force required on physical keyboards (i.e., $50gF$). A within-subjects study design was utilized with the conditions being balanced through latin square design across participants. We collected 3 trials per participant per condition.

6.1.2 Participants. We recruited 12 participants, with ages ranging from 21 to 31 ($M = 26.6, SD = 3.6$, 2 female). All participants were right-handed.

6.1.3 Procedure. Similar to previous studies, participants used the study platform. First, the participant was asked to perform a transcription typing task for 30 seconds on a physical keyboard as a warmup. Following this, the *wrist-off* condition from the resting data collection study (Section 4.1) and the *restricted* condition from the typing data collection study (Section 4.2) were each presented three times, alternating between them. The data collected from this stage was then used to calculate a personalized threshold as discussed prior. This stage also functioned as a baseline for typing on a virtual keyboard. Figure 12 shows the thresholds obtained for the personalized condition of each participant.

Inspired from work by Kim et. al [26] and Varcholik et al. [53], the main task used for the study was designed to resemble real-life typing and is as follows. In place of transcription typing, as seen in previous studies, we generated a 4x25 grid of 100 sampled phrases [41], which were presented to the participants on a separate display (see Figure 2). Within a trial, twenty phrases were sampled randomly. For each phrase, participants were shown an instruction to type the content of the cell containing the phrase on the grid. The participants were expected to find the phrase on the gird, type it accurately, and then press Enter. If the phrase entered had any errors, highlighted in red during typing, the trial would not proceed until errors were corrected. When the phrase was entered correctly, a popup was shown instructing the participant to wait to start the next trial while resting their hands on the virtual keyboard. Additionally, the popup showed a textbox which printed out any character if it was accidentally triggered during the resting period. This was added for participants to better understand the resting forces on the virtual keyboard throughout.

After typing the 20 phrases in a trial, 5 additional phrases with the words "asdf", "jkl;", "fdsa" and ";lkj" were presented to be typed. We refer to these phrases as homerow phrases. As the participants did not have to move their fingers around the virtual keyboard to perform a keystroke on the homerow phrases, we consider these as controls for false positives during typing. For the purpose of comparison, we assume the chances of an erroneous input being a result of typing errors is significantly less when typing the homerow phrases.

We note that there was no time limit to complete the trials. After each condition (i.e., after conducting three trials of 20 phrases each for a condition), participants were asked to complete the NASA TLX questionnaire and provide subjective feedback on their typing experience. Each session lasted between 70 to 90 minutes in total.

6.1.4 Results. To understand the impact these non-uniform threshold approaches have on false negatives, we extract the ratio between false negatives and keystrokes from the typing data as done in Section 5.1.4; Figure 11 shows the ratio of false negatives to total keystrokes grouped by condition. Overall, we observe that the non-uniform thresholds have a lower number of false negatives. As the data violated normality constraints we used a Kruskal-Wallis test comparing the false negatives across conditions, where it shows a main effect ($H = 14.0, df = 4, p = .007$). Pairwise comparison

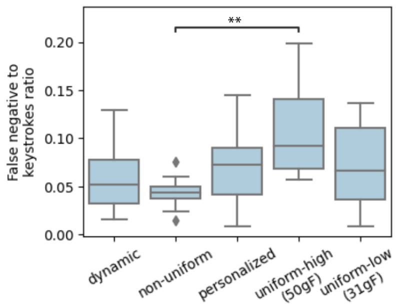


Figure 11: False negatives as a ratio between number of false negatives and total number of keystrokes from typing data grouped by the tested conditions. Significant pairs are denoted by ** where $p \leq 0.01$.

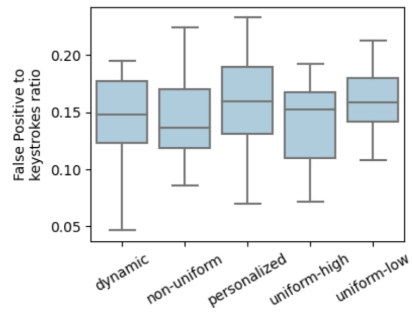


Figure 13: Number of errors during typing the "homerow" phrases as a ratio between the errors and the total number of keystrokes. The number of errors are treated as an estimate to the false positives. Has no pair-wise significance.

using Dunn's test with Bonferroni corrections shows only the non-uniform threshold condition ($M = .043, SD = .016$) is significantly different from the uniform-high condition ($M = .106, SD = .047$). Lastly, we calculate the number of keystrokes that would have been classified as "touched" events with the 31gf threshold; 468 keystrokes with the dynamic threshold approach, 481 with the non-uniform threshold approach and 318 with the personalized threshold approach.

Similar to the observations from the data collected in Section 5.1.4, here also we observe contact events that are not keystrokes. In this data set, the percentage of total contact time that is from non-keystroke contact events, during study periods not including resting, is 73.3% compared to 24.6% from the data collected in Section 5.1.4. This increase in the percentage of time could have been influenced by the design of the study itself; resting behaviour could have been encouraged due to participants resting their hands during the rest periods between trials. Nonetheless, during the trials, participants were inclined to keep contact with the virtual keyboard, even when not typing, which was not required. As an example, while searching for a phrase in-between moments of typing participants often kept their hands rested on the keyboard.

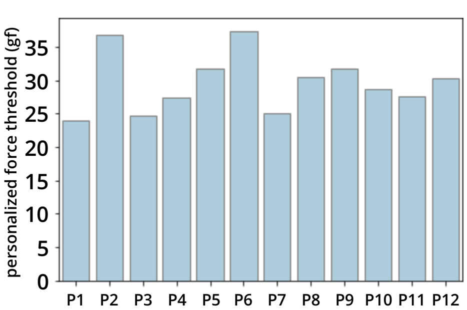


Figure 12: Personalized force threshold was calculated during the study for each participant.

We analyze the typing data from the homerow phrases to observe the impact these approaches have on false positives. The errors made during this task, as a ratio between the number of errors and total keystrokes, averaged across participants can be seen in Figure 13. We use the error as an estimate of the false positives, where we assume the errors from the uniform-high condition can act as a baseline due to the minimal number of false positives in this condition. As the data violated the normality constraint, we conducted a Kruskal-Wallis test. Results showed no significant effect among the conditions ($H = 2.49, df = 4, p = 0.64$). Although, we do observe that the dynamic threshold and non-uniform threshold approaches have a lower error rate compared to the uniform-low threshold and the personalized threshold.

We also analyze the data from the resting periods for false positives. We count the number of times a character was entered while participants were resting between phrases. Figure 14 shows the summary of these false positives as an average for each resting interval. As the data violated the normality constraint, we conducted a Kruskal-Wallis test. Results showed a main effect ($H = 15.1, df = 4, p = 0.004$). Pair wise comparison using Dunn's test with Bonferroni corrections shows the dynamic ($M = 1.02SD = 0.67, H = -2.9, p = 0.03$) and personalized threshold ($M = 1.3, SD = 1.05, H = -3.6, p = 0.002$) conditions are significantly different from the uniform-high condition ($M = 0.08, SD = 0.15$).

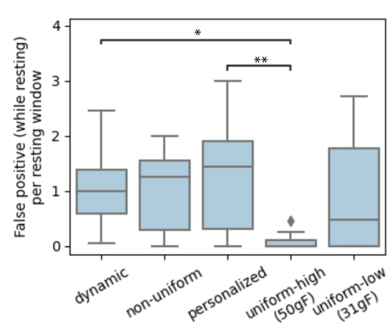


Figure 14: Average number of false triggers during each resting period. Significant pairs denoted by * where $p \leq 0.05$ and by ** where $p \leq 0.01$.

For completeness, and as an initial observation, we also analyzed the average words per minute (WPM) in this study (see Figure

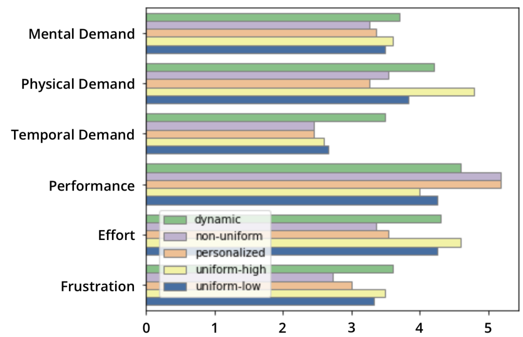


Figure 15: Summary of NASA TLX scores comparing the five study conditions.

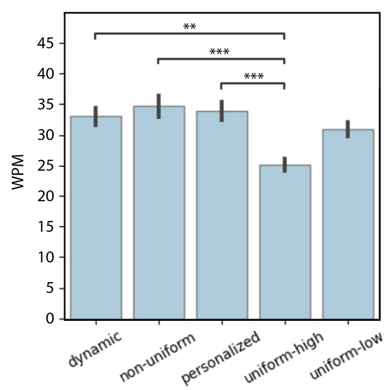


Figure 16: Mean WPM for each study condition. Black lines denote 95% confidence intervals. Significant pairs denoted by ** where $p \leq 0.01$ and by *** where $p \leq 0.001$.

16). As the log of the WPM satisfied the normality constraint, repeated measures ANOVA was used. Results show a significant effect ($F(4, 44) = 8.36, p < 0.01$). Post-hoc analysis with Turkey HSD shows a similar trend to that of false negatives and false positives, the uniform-high condition ($M = 24.9, SD = 7.8$) has a lower WPM compared to the other conditions, significantly different from dynamic threshold ($p < 0.01$), non-uniform threshold ($p < 0.001$) and personalized threshold ($p < 0.001$). Though not significant, dynamic threshold ($M = 32.8, SD = 12.1$), non-uniform threshold ($M = 34.5, SD = 14.1$) and personalized threshold ($M = 33.8, SD = 12.4$) on average have better WPM than the uniform-low condition ($M = 30.8, SD = 9.9$).

From the NASA TLX workload rating (see Figure 15) we observe the personalized threshold approach and the non-uniform threshold based approaches perform slightly better than other conditions in all measures. Although we observe no significant effect. The dynamic threshold approach performs comparable to the uniform or worse on some measures. Based on the subjective feedback at the end of each condition, the participants found the dynamic approach the most confusing, due to its perceived unpredictability. To this effect, participants noted that the non-uniform and personalized approaches were easier, however, were prone to false positives which is reflected in the quantitative results above.

7 DISCUSSION

We present lessons learned regarding the use of force on a three-state virtual keyboard, a discussion surrounding potential improvements to T-Force, and compare our results to that of earlier work. Throughout, we touch upon limitations and future work. We restate, the primary contribution we make is the investigation and accompanying analysis towards the use of force in defining a threshold function for three-state virtual keyboards. As such, a more comprehensive comparison with other three-state virtual keyboard techniques, along with longitudinal typing performance and ablation studies concerning how the different formulations for T-Force can be combined, is left for future work.

7.1 Lessons learned

7.1.1 The role of force in a three-state keyboard. Through the data collected, we uncover many characteristics towards the use of force in defining a functional three-state keyboard. When incorporating force within a virtual keyboard, we can enable people with the ability to rest their fingers, similar to the action on a physical keyboard, without fear of registering a keystroke upon releasing a touch. It is worth noting that mechanical keyboard switches offer actuation forces generally around the 50gf level. Yet, on virtual keyboards, the force exerted can be much lower, which also corroborates results by Kim et al. [20] and the 31gf value found in Section 4.3. While resting forces are relatively consistent, the force exerted by people when typing is however not uniform across a virtual keyboard (Figure 1). The difference in the maximum force exerted on the keyboard surface varies the most across columns, but also across rows. In particular, regions pertaining to the pinky fingers and thumbs see significantly less force being applied. Furthermore, and importantly, different users have different levels of average maximum force exertions and even individual keystroke force profiles. Thus, a simple constant threshold function may not be the optimal solution for incorporating force.

7.1.2 Force threshold functions. The primary goal of the T-Force Function 1 is to allow the users of a virtual keyboard to have a "touched" state, while also ensuring that keystrokes are not missed. In defining a constant threshold function, the data suggests that there is a large overlap between force exerted when resting and typing. An obvious solution to allow resting without the fear of triggering a keystroke is to increase the threshold. As seen in the results with our uniform-high condition, this approach would allow one to rest on the keyboard without the worry of triggering a key, however, comes at the cost of having a higher number of false negatives and being less comfortable for typing. By simply modifying the constant threshold function to consider the variations in forces we found throughout our studies, improvements can be afforded. We propose three different formulations for the threshold function that leverages these variations.

The results from the user study in Section 6.1 show that these approaches can decrease false negatives, though they are not efficient in reducing the false positives, which is a limitation of the approaches discussed. This is also reflected by the subjective feedback received from the participants. Particularly with the non-uniform threshold, participants felt that it was easier to type with the pinky fingers where they have to apply less force but tend to miss-trigger while resting. While the dynamic threshold approach is seen less favorably by the participants, we observe that the false positives with this approach are lower compared to the other two approaches. One drawback participants highlighted with the dynamic threshold approach is that they could not clearly understand when the key would be triggered. Similar feedback was provided by some participants for the non-uniform threshold approach. This could potentially be offset by adequate feedback mechanisms, however, remains to be studied. Importantly, the subjective feedback towards the non-uniform and personalized threshold approaches shows participants rate it to be less frustrating and more comfortable on average. We believe that with further enhanced threshold functions, utilizing feedback and results captured in this work, the overall experience and typing performance on virtual keyboards could yet improve. As future work, we intend to explore the choice of maximum and minimum values for the non-uniform force threshold method and create increasingly personal force threshold functions.

As limitation, results from the analysis across our studies should be interpreted with care due to the smaller number of participants. An example in our final study, instead of fully counterbalancing across the five conditions (which would require 120 participants), we instead used a balanced Latin square design. This yields ten orderings for the five conditions [3] and offers a compromise between validity and practicality. However, since this approach can only mitigate the risk of carryover and ordering effects to a certain degree, we explicitly chose to include all data from the twelve participants we recruited. Additionally, we acknowledge that our participant diversity could be improved and that an equal balance is something we strive for. As gender seems to play a marginal role with regard to touch characteristics [5], such differences reinforce the need to individualize beyond gender or even age, for example. As a result, our findings suggest the need to personalize force thresholds remains to provide beneficial insight into the design of force-sensing virtual keyboards.

7.2 Improving T-force

Our approach towards the definition of a threshold function stems from the perspective that false positives and false negatives should both be eliminated. As we have observed, not only is there a large overlap on the force exerted between resting and typing, the threshold functions are not always able to successfully filter them out. This is also seen in other recent work with three-state virtual keyboards [14]. Another perspective to consider in optimizing the threshold function is to find a trade-off between the false positives and false negatives. That is, to explore the following question: are users more tolerant to false positives or false negatives when using three-state keyboards? The threshold could be adjusted accordingly to minimize the overall frustration of the user.

A presumable solution to improve the threshold function would be a combination of the three different approaches; where the maximum and minimum values used in the non-uniform threshold approach are scaled and each column follows a dynamic approach, both based on personalized data. In this work, we have studied these approaches individually to assess how accounting for different variations in force exerted would impact a threshold function's performance. Selection of the optimal parameters, that minimize both false positives and false negatives, for this combined threshold function and the impact it would have on the different metrics explored are non-trivial. Future work should explore these elements of using a threshold function to further enable three-state functionality on virtual keyboards. Another approach that we have not considered here is the thresholds changing over time as a person types. This can even further personalize the virtual keyboard to account for each user's unique force profiles. As seen in more recent work on personalized interaction [55], utilizing an individual's data, maybe an increasingly optimal and viable approach.

7.3 Comparison to previous work

While we reserve direct comparison with different approaches for three-state virtual keyboards for future work, here we discuss our findings in comparison to previous work.

7.3.1 Resting behaviour and differentiating contact events. Throughout our typing studies, participants looked to rest directly on the Sensel Morph; similarly, Tapboard [26] and TypeBoard [14] reported the trend of participants resting on the keyboard when they are made familiar with the capability. This highlights a natural action, similar to the use of physical keyboards, that should be supported. Furthermore, while not the focus of this paper, we also observed that participants did not rest their fingers in a straight line similar to findings in previous studies [12, 27, 57]. These resting behaviours can potentially be impactful in defining keystrokes versus non-keystrokes on three-state virtual keyboards.

In contrast to earlier work [12, 57], our results do not support the claim that all contacts are keystrokes. Instead, a large portion of these consists of resting the fingers on the screen, presumably for comfort. While this would include accidental touches, it can be argued that it is also a result of other typing behaviors commonly seen in 10-finger touch-typing on physical keyboards [7, 9]. We intend to further investigate this in future work. Another aspect that we believe might influence the resting behaviour is the study design itself. When considering real-life typing applications, users seldom continuously type like they would in a transcription typing task. While transcription typing tasks are important when comparing the efficiency of different text input methods, it may not capture all aspects of typing. Such aspects include small pauses to think or refer to something or the use of shortcuts and hotkeys. Other methods such as memorization and copy tasks [53] and real-world tasks where participants annotate the input after completing the task [14] could help in capturing such typing characteristics.

7.3.2 Incorporating time. Looking towards time as a means for differentiating touch events, Kim et al. [26] argue for a trade-off between resting and typing by defining a temporal threshold. However, one key limitation to their approach is that a keystroke is always triggered at the end of a touch event. Furthermore, as seen in our analysis, keystrokes are triggered within a much smaller time window (see Figure 9) than the threshold they utilized (i.e., 300 *ms*). While they propose this approach as an alternative on non-force sensitive virtual keyboards, time could still beneficially augment the threshold functions within our work. For example, time could be used as another layer of filtering, similar to the dynamic threshold approach.

7.3.3 Machine learning models for three-state keyboards. With a similar goal to our work, Gu et al. [14] use data provided by a Sensel Morph to train a machine learning model for classifying intentional (i.e, contact resulting in a "pressed" state) from unintentional (i.e., contact resulting in a "touched" state) events. Our results complement their findings on using a force-sensitive surface to define a three-state virtual keyboard, emphasising the value in investigating the impact of factors, and specifically force, on model classification. Our force threshold function proposed could assist in simplifying models required. For example, T-Force does not make the assumption that only one finger from a hand is in contact with the surface when performing a keystroke, and does not suffer from any time delays since the keystroke would be registered as soon as the force exceeds the threshold. It is also possible to combine other input modalities for data-driven models such as with touch location [8, 12], finger motion [4, 49] and gaze [15]. Analyzing characteristics of these individually, and in combination with force would further inform future data-driven models to be more robust and efficient.

7.3.4 Typing speeds. Words per minute (WPM) as a metric for typing speed was captured within our final study as a preliminary assessment of the efficiency of our T-Force methods. Due to the lack of tactile feedback afforded when compared with physical keyboards, flat surface keyboards do not provide comparable WPM rates [43]. Consequently, prior work on flat surface and on-screen typing show WPM rates between 20 and 35 WPM without augmentation [14, 36, 39, 43], 40 WPM when augmented with haptics [39], 30 WPM for altered layouts [34], and between 35 and 41 WPM when force is considered [14, 57][4]. Our results show already comparable typing speeds to these previously proposed methods, with our non-uniform condition reporting 34.5 WPM. Taking the knowledge gained from our results within this work, we can now move forward in optimizing the use of force for flat surface typing. Future work can then begin to study typing efficiency through appropriate study methodologies, specifically over many blocks as typing is a learned behaviour which betters over time [40], where we can expect improved WPM rates from those found within this work. Comparing with other text entry platforms, we note that flat surface typing typically affords lower typing speeds. Physical keyboards offer an average of 51 WPM [7]. These speeds offer a target WPM for flat surface typing which we strive to achieve in the future.

8 CONCLUSION

In this paper we explore the use of a force sensitive touch surface to define three state virtual keyboards for 10-finger typing. More specifically, we focus on capturing force characteristics during resting and typing that allow us to distinguish between "touched" and "pressed" states. Furthermore, we focus on minimizing miss recognition while also allowing to rest on the virtual keyboard without accidentally triggering a keystroke. Initially we define a constant force threshold function, where any contact event that exceeds the threshold is considered to be in a "pressed" state. In order to derive this threshold we collect resting and typing data across two data collection studies. In a third study, we observe that the constant force threshold is not an optimal solution. However, we gain knowledge regarding variations of force being used across the keyboard, participants, and fingers. As such, we propose three different force threshold approaches that leverage these variations, which we compare with constant thresholds in a final user study. Though not significant, the results show improvements both quantitatively and qualitatively. Throughout, we shed light on how force based three-state virtual keyboards can be designed and conclude with discussions on lessons learned, improvements that can be made, and comparisons with prior work.

[4]Please note we have used a conversion of 1 word per minute to 1.3 Chinese characters per minute [32] where needed.

REFERENCES

[1] Thomas Bekken Aschim, Julie Lidahl Gjerstad, Lars Vidar Lien, Rukaiya Tahsin, and Frode Eika Sandnes. 2019. Are Split Tablet Keyboards Better? A Study of Soft Keyboard Layout and Hand Posture. *Lecture Notes in Computer Science* (2019), 647–655. https://doi.org/10.1007/978-3-030-29387-1_37

[2] Myroslav Bachynskyi, Gregorio Palmas, Antti Oulasvirta, Jürgen Steimle, and Tino Weinkauf. 2015. Performance and Ergonomics of Touch Surfaces. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems.* https://doi.org/10.1145/2702123.2702607

[3] James V. Bradley. 1958. Complete Counterbalancing of Immediate Sequential Effects in a Latin Square Design. *J. Amer. Statist. Assoc.* 53, 282 (1958), 525–528. https://doi.org/10.1080/01621459.1958.10501456 arXiv:https://www.tandfonline.com/doi/pdf/10.1080/01621459.1958.10501456

[4] Daewoong Choi, Hyeonjoong Cho, and Joono Cheong. 2015. Improving Virtual Keyboards When All Finger Positions Are Known. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology.* https://doi.org/10.1145/2807442.2807491

[5] Amrish O. Chourasia, Douglas A. Wiegmann, Karen B. Chen, Curtis B. Irwin, and Mary E. Sesto. 2013. Effect of Sitting or Standing on Touch Screen Performance and Touch Characteristics. *Human Factors* 55, 4 (2013), 789–802. https://doi.org/10.1177/0018720812470843 arXiv:https://doi.org/10.1177/0018720812470843 PMID: 23964418.

[6] Matthew J. C. Crump and Gordon D. Logan. 2010. Warning: This Keyboard Will Deconstruct- the Role of the Keyboard in Skilled Typewriting. *Psychonomic Bulletin & Review* 17, 3 (2010), 394–399. https://doi.org/10.3758/pbr.17.3.394

[7] Vivek Dhakal, Anna Maria Feit, Per Ola Kristensson, and Antti Oulasvirta. 2018. Observations on Typing from 136 Million Keystrokes. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems.* https://doi.org/10.1145/3173574.3174220

[8] Jörg Edelmann, Philipp Mock, Andreas Schilling, Peter Gerjets, Wolfgang Rosenstiel, and Wolfgang Straßer. 2012. Towards the keyboard of oz. In *Proceedings of the 2012 ACM international conference on Interactive tabletops and surfaces - ITS '12.* https://doi.org/10.1145/2396636.2396662

[9] Anna Maria Feit, Daryl Weir, and Antti Oulasvirta. 2016. How We Type. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems.* https://doi.org/10.1145/2858036.2858233

[10] Tamara Fiedler and Yasemin Vardar. 2019. A Novel Texture Rendering Approach for Electrostatic Displays. In *International Workshop on Haptic and Audio Interaction Design - HAID2019.* Lille, France. https://hal.archives-ouvertes.fr/hal-02011782

[11] Leah Findlater and Jacob Wobbrock. 2012. Personalized input. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12.* https://doi.org/10.1145/2207676.2208520

[12] Leah Findlater, Jacob O. Wobbrock, and Daniel Wigdor. 2011. Typing on flat glass. In *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11.* https://doi.org/10.1145/1978942.1979301

[13] Mayank Goel, Leah Findlater, and Jacob Wobbrock. 2012. WalkType: Using Accelerometer Data to Accomodate Situational Impairments in Mobile Touch Screen Text Entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12).* Association for Computing Machinery, New York, NY, USA, 2687–2696. https://doi.org/10.1145/2207676.2208662

[14] Yizheng Gu, Chun Yu, Xuanzhong Chen, ZHUOJUN LI, and Yuanchun Shi. 2021. TypeBoard: Identifying Unintentional Touch on Pressure-Sensitive Touchscreen Keyboards. In *The 34th Annual ACM Symposium on User Interface Software and Technology.* nil. https://doi.org/10.1145/3472749.3474770

[15] Tatsuhito Hasegawa and Tatsuya Hatakenaka. 2019. Touch-Typing Detection Using Eyewear: Toward Realizing a New Interaction for Typing Applications. *Sensors* 19, 9 (2019). https://doi.org/10.3390/s19092022

[16] Min-Chieh Hsiu, Da-Yuan Huang, Chi An Chen, Yu-Chih Lin, Yi ping Hung, De-Nian Yang, and Mike Chen. 2016. ForceBoard. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. nil. https://doi.org/10.1145/2957265.2961827

[17] Seokhee Jeon, Hongchae Lee, Jiyoung Jung, and Jin Ryong Kim. 2018. User-Adaptive Key Click Vibration on Virtual Keyboard. *Mobile Information Systems* 2018 (2018), 1–12. https://doi.org/10.1155/2018/6126140

[18] Xinhui Jiang, Yang Li, Jussi P.P. Jokinen, Viet Ba Hirvola, Antti Oulasvirta, and Xiangshi Ren. 2020. How We Type: Eye and Finger Movement Strategies in Mobile Typing. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/3313831.3376711

[19] Huhn Kim, Seungyoun Yi, and So-Yeon Yoon. 2019. Exploring touch feedback display of virtual keyboards for reduced eye movements. *Displays* 56 (2019), 38–48. https://doi.org/10.1016/j.displa.2018.11.004

[20] Jeong Ho Kim, Lovenoor Aulck, Michael C Bartha, Christy A Harper, and Peter W Johnson. 2012. Are There Differences in Force Exposures and Typing Productivity Between Touchscreen and Conventional Keyboard? *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 56, 1 (2012), 1104–1108. https://doi.org/10.1177/1071181312561240

[21] Jin Ryong Kim, Xiaowei Dai, Xiang Cao, Carl Picciotto, Desney Tan, and Hong Z. Tan. 2012. *A Masking Study of Key-Click Feedback Signals on a Virtual Keyboard*. Springer Berlin Heidelberg, 247–257. https://doi.org/10.1007/978-3-642-31401-8_23

[22] Jin Ryong Kim and Hong Z. Tan. 2014. *Haptic Feedback Intensity Affects Touch Typing Performance on a Flat Keyboard*. Springer Berlin Heidelberg, 369–375. https://doi.org/10.1007/978-3-662-44193-0_46

[23] Jin Ryong Kim and Hong Z. Tan. 2014. A study of touch typing performance with keyclick feedback. In *2014 IEEE Haptics Symposium (HAPTICS)*. https://doi.org/10.1109/haptics.2014.6775459

[24] Jin Ryong Kim and Hong Z. Tan. 2015. Effect of information content in sensory feedback on typing performance using a flat keyboard. In *2015 IEEE World Haptics Conference (WHC)*. https://doi.org/10.1109/whc.2015.7177718

[25] Sunjun Kim and Geehyuk Lee. 2012. Typing on a Touch Surface: Effect of Feedback with Horizontal Touch Keyboard and Vertical Display Setup. In *The 10th Asia Pacific Conference on Computer Human Interaction*. Human Centered Design Organization and ACM, 525–530.

[26] Sunjun Kim, Jeongmin Son, Geehyuk Lee, Hwan Kim, and Woohun Lee. 2013. TapBoard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/2470654.2470733

[27] Ue-Hwan Kim, Sahng-Min Yoo, and Jong-Hwan Kim. 2019. I-Keyboard: Fully Imaginary Keyboard on Touch Devices Empowered By Deep Neural Decoder. *CoRR* (2019). arXiv:1907.13285 [cs.HC] http://arxiv.org/abs/1907.13285v1

[28] Yuki Kuno and Buntarou Shizuki. 2017. *Meyboard: A QWERTY-Based Soft Keyboard for Touch-Typing on Tablets*. Springer International Publishing, 193–207. https://doi.org/10.1007/978-3-319-58077-7_16

[29] Sunghyuk Kwon, Donghun Lee, and Min K. Chung. 2009. Effect of Key Size and Activation Area on the Performance of a Regional Error Correction Method in a Touch-Screen Qwerty Keyboard. *International Journal of Industrial Ergonomics* 39, 5 (2009), 888–893. https://doi.org/10.1016/j.ergon.2009.02.013

[30] Lik Hang Lee, Ngo Yan Yeung, Tristan Braud, Tong Li, Xiang Su, and Pan Hui. 2020. Force9: Force-assisted Miniature Keyboard on Smart Wearables. In *Proceedings of the 2020 International Conference on Multimodal Interaction*. nil. https://doi.org/10.1145/3382507.3418827

[31] Blaine Lewis, Greg d'Eon, Andy Cockburn, and Daniel Vogel. 2020. KeyMap: Improving Keyboard Shortcut Vocabulary Using Norman's Mapping. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3313831.3376483

[32] Changshuan Li. 2010. Coping strategies for fast delivery in simultaneous interpretation. *The Journal of Specialised Translation* 13 (2010), 19–25.

[33] Frank Chun Yat Li, Leah Findlater, and Khai N. Truong. 2013. Effects of Hand Drift While Typing on Touchscreens. In *Proceedings of Graphics Interface 2013* (Regina, Sascatchewan, Canada) *(GI '13)*. Canadian Information Processing Society, CAN, 95–98.

[34] Frank Chun Yat Li, Richard T. Guy, Koji Yatani, and Khai N. Truong. 2011. The 1line Keyboard: A QWERTY Layout in a Single Line. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 461–470. https://doi.org/10.1145/2047196.2047257

[35] Ming-I Brandon Lin, Ruei-Hong Hong, and Yu-Ping Huang. 2020. Influence of Virtual Keyboard Design and Usage Posture on Typing Performance and Muscle Activity During Tablet Interaction. *Ergonomics* 63, 10 (2020), 1312–1328. https://doi.org/10.1080/00140139.2020.1778097

[36] Ming-I Brandon Lin, Ruei-Hong Hong, and Yu-Ping Huang. 2020. Influence of virtual keyboard design and usage posture on typing performance and muscle activity during tablet interaction. *Ergonomics* 63, 10 (2020), 1312–1328. https://doi.org/10.1080/00140139.2020.1778097 arXiv:https://doi.org/10.1080/00140139.2020.1778097 PMID: 32496886.

[37] Yiqin Lu, Chun Yu, Xin Yi, Yuanchun Shi, and Shengdong Zhao. 2017. Blindtype. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 2 (2017), 1–24. https://doi.org/10.1145/3090083

[38] Zhaoyuan Ma, Darren Edge, Leah Findlater, and Hong Z. Tan. 2015. Haptic keyclick feedback improves typing speed and reduces typing errors on a flat keyboard. In *2015 IEEE World Haptics Conference (WHC)*. https://doi.org/10.1109/whc.2015.7177717

[39] Zhaoyuan Ma, Darren Edge, Leah Findlater, and Hong Z. Tan. 2015. Haptic keyclick feedback improves typing speed and reduces typing errors on a flat keyboard. In *2015 IEEE World Haptics Conference (WHC)*. 220–227. https://doi.org/10.1109/WHC.2015.7177717

[40] I. Scott MacKenzie and R. William Soukoreff. 2002. Text Entry for Mobile Computing: Models and Methods, Theory and Practice. *Human–Computer Interaction* 17, 2-3 (2002), 147–198. https://doi.org/10.1080/07370024.2002.9667313 arXiv:https://www.tandfonline.com/doi/pdf/10.1080/07370024.2002.9667313

[41] I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase sets for evaluating text entry techniques. In *CHI '03 extended abstracts on Human factors in computing systems - CHI '03*. https://doi.org/10.1145/765891.765971

[42] Dan Odell. 2015. On-screen keyboards. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. https://doi.org/10.1145/2785830.2785848

[43] Dan Odell. 2015. On-Screen Keyboards: Does the Presence of Feedback or Tactile Landmarks Improve Typing Performance?. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark) *(MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 131–136. https://doi.org/10.1145/2785830.2785848

[44] Dan Odell and Eric Faggin. 2014. The Typing Performance and Preference Costs of Reducing Tactile Feedback and Tactile Landmarks in Tablet Keyboards. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 58, 1 (2014), 1790–1794. https://doi.org/10.1177/1541931214581373

[45] Sarangi P. Parikh and Joel M. Esposito. 2012. Negative Feedback for Small Capacitive Touchscreen Interfaces: a Usability Study for Data Entry Tasks. *IEEE Transactions on Haptics* 5, 1 (2012), 39–47. https://doi.org/10.1109/toh.2011.71

[46] Radosław Puka, Piotr Łebkowski, and Jerzy Duda. 2021. One Row Keyboard: The Concept of Designing a Common Layout for Physical and Virtual Keyboards. *Electronics* 10, 6 (2021). https://doi.org/10.3390/electronics10060663

[47] Ely Rabin and Andrew M. Gordon. 2004. Tactile Feedback Contributes To Consistency of Finger Movements During Typing. *Experimental Brain Research* 155, 3 (2004), 362–369. https://doi.org/10.1007/s00221-003-1736-6

[48] Gregory Reardon, Nikolas Kastor, Yitian Shao, and Yon Visell. 2020. Elastowave: Localized Tactile Feedback in a Soft Haptic Interface via Focused Elastic Waves. In *2020 IEEE Haptics Symposium (HAPTICS)*. https://doi.org/10.1109/haptics45997.2020.ras.hap20.25.aa4d97aa

[49] Mark Richardson, Matt Durasoff, and Robert Wang. 2020. Decoding Surface Touch Typing from Hand-Tracking. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. https://doi.org/10.1145/3379337.3415816

[50] Dominik Schmidt, Florian Block, and Hans Gellersen. 2009. *A Comparison of Direct and Indirect Multi-touch Input for Large Surfaces*. Springer Berlin Heidelberg, 582–594. https://doi.org/10.1007/978-3-642-03655-2_65

[51] Andrew Sears, Julie A. Jacko, Josey Chu, and Francisco Moro. 2001. The Role of Visual Search in the Design of Effective Soft Keyboards. *Behaviour & Information Technology* 20, 3 (2001), 159–166. https://doi.org/10.1080/01449290110049790

[52] Weinan Shi, Chun Yu, Xin Yi, Zhen Li, and Yuanchun Shi. 2018. Toast. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 1–23. https://doi.org/10.1145/3191765

[53] Paul D. Varcholik, Joseph J. LaViola, and Charles E. Hughes. 2012. Establishing a Baseline for Text Entry for a Multi-Touch Virtual Keyboard. *International Journal of Human-Computer Studies* 70, 10 (2012), 657–672. https://doi.org/10.1016/j.ijhcs.2012.05.007

[54] Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. 2014. Uncertain text entry on mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. nil. https://doi.org/10.1145/2556288.2557412

[55] Pui Chung Wong, Kening Zhu, Xing-Dong Yang, and Hongbo Fu. 2020. Exploring Eyes-Free Bezel-Initiated Swipe on Round Smartwatches. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3313831.3376393

[56] Zhican Yang, Chun Yu, Xin Yi, and Yuanchun Shi. 2019. Investigating Gesture Typing for Indirect Touch. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 3 (2019), 1–22. https://doi.org/10.1145/3351275

[57] Xin Yi, Chen Wang, Xiaojun Bi, and Yuanchun Shi. 2020. PalmBoard: Leveraging Implicit Touch Pressure in Statistical Decoding for Indirect Text Entry. In

Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems. https://doi.org/10.1145/3313831.3376441

[58] Xiao Ying Zhao, Ian A Spraggs, David A Pakula, and Tang Y Tan. 2021. 3d touch. US Patent App. 16/708,350.

[59] Mingyuan Zhong, Chun Yu, Qian Wang, Xuhai Xu, and Yuanchun Shi. 2018. ForceBoard. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. nil. https://doi.org/10.1145/3173574.3174102

[60] Suwen Zhu, Jingjie Zheng, Shumin Zhai, and Xiaojun Bi. 2019. i'sFree. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. nil. https://doi.org/10.1145/3290605.3300678