Force9: Force-assisted Miniature Keyboard on Smart Wearables

Lik-Hang LEE
Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong;
Center for Ubiquitous Computing, The University of Oulu, Finland

Ngo-Yan YEUNG
Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong

Tristan BRAUD
Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong

Tong LI
Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong

Xiang SU
Department of Computer Science, The University of Helsinki, Finland;
Center for Ubiquitous Computing, The University of Oulu, Finland

Pan HUI
Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong;
Department of Computer Science, The University of Helsinki, Finland

ABSTRACT

Smartwatches and other wearables are characterized by small-scale touchscreens that complicate the interaction with content. In this paper, we present Force9, the first optimized miniature keyboard leveraging force-sensitive touchscreens on wrist-worn computers. Force9 enables character selection in an ambiguous layout by analyzing the trade-off between interaction space and the easiness of force-assisted interaction. We argue that dividing the screen's pressure range into three contiguous force levels is sufficient to differentiate characters for fast and accurate text input. Our pilot study captures and calibrates the ability of users to perform force-assisted touches on miniature-sized keys on touchscreen devices. We then optimize the keyboard layout considering the goodness of character pairs (with regards to the selected English corpus) under the force-based configuration and the users' familiarity with the QWERTY layout. We finally evaluate the performance of the trimetric optimized Force9 layout, and achieve an average of 10.18 WPM by the end of the final session. Compared to the other state-of-the-art approaches, Force9 allows for single-gesture character selection without addendum sensors.

CCS CONCEPTS

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

KEYWORDS

force-assisted interaction; text entry; smartwatches; optimization.

ICMI '20, October 25–29, 2020, Virtual event, Netherlands

ACM Reference Format:
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 (ICMI '20), October 25–29, 2020, Virtual event, Netherlands.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3382507.3418827

1 INTRODUCTION

In recent years, a large number of smartwatches have been launched onto the market. Originally designed for quick interactions with the phone environment, smartwatches' user-accepted usages are more limited due to insufficient interaction methods. Common usages include message notification, biometric information collection, user health status monitoring, as well as location positioning and city navigation. However, text entry for message input on smartwatches is usually restricted to predefined texts and emojis for one-click replies. The rising popularity of smartwatches calls for more efficient and user-friendly input methods.

The soft QWERTY keyboard and its local variants are the *de facto* standard on smartphones. Indeed, the input interface of mobile devices is highly influenced by physical keyboards, where the QWERTY layout is prevalent. This is especially true when a sizable input interface is available and one keypad corresponds to a discrete character, number or symbol. By nature, the standard QWERTY layout is designed for unambiguous, physical keyboards, and migrated *as-is* to the spacious touchscreen interfaces of smartphones. However, migrating the layouts to smartwatches makes text input challenging due to the touchscreen's limited size [1].

Alternative text-input methods should be considered for the emerging size-constrained interfaces of smartwatches. Current researches on text entry for smartwatches consider consecutive gestures, including multiple taps [2], swipes [3], shift pointing [4], panning on the movable keyboard [5], combining the thumb and index finger to create gesture combination [6], as well as additional sensors worn on the index and middle fingers [7]. However, all of these systems require the users to perform a minimum of two consecutive gestures or a continuous set of gestures to search the

Figure 1: System Implementation: typing the phrase 'FORCE KEYBOARD' illustrated by the following 5 characters 'F' 'R' 'C' 'E' 'Y', chosen on the ambiguous keys through three levels of force exertion.

characters in the QWERTY layout. To reduce the gesture complexity, optical sensors are worn on the index finger and middle finger to distinguish the characters in a 2-level ambiguous keyboard [7], where the index finger maps to the first character while the middle finger maps to the alternative character inside the same key. However, addendum sensors require additional preparation times for putting on the sensor, on top of constantly carrying two additional pieces of hardware. In contrast, we address these issues by proposing a system based on single-step gestures without addendum sensors for text entry on miniature keyboards.

In this paper, we present a force-assisted text input technique for size-constrained smart wearables. In order to perform character selection, we explore the alternative modality of force-assisted input and leverage the force-sensitive touchscreens available on smartphones and smartwatches. We arrange 26 characters of the Roman alphabet ($a-z$) and white space ("_") into a 9-key ambiguous layout. Each key features three characters, composing an ambiguous keyboard layout. Users select the target character among the three characters in a key by exerting a mapped amount of force (see Figure 3b). We divide the force spectrum into three ranges: *light tap*, *shallow force touch*, and *deep force touch* for distinguishing the characters within the same key in our optimized layout. Figure 1 shows the procedure of typing the word phrase 'FORCE KEYBOARD' on our prototypical implementation on an *Apple Watch Series 3*. Three characters are arranged within a key. The applied force level distinguishes the target character. For instance, the first character in the word phrase, 'F', is located in the key of 'OKF'. The user applies a *deep force touch* to select this character, with an exerted force of more than 0.9915N. In addition, swipe gestures maps to the '*Enter*' and '*Delete*' keys, e.g., on a left-handed watch, the swipe gestures trigger from the upper right corner: a swipe from right to left along the upper edge maps to '*Delete*'; and a swipe along the right edge from the upper corner to the lower corner leads to '*Enter*'.

Our contribution is threefold: (1) We design a miniature force-assisted ambiguous keyboard for the constrained interface of smartwatches. (2) We solve the optimization problem for character placement within our ambiguous keyboard, and propose an intuitive, yet efficient, layout for force touch interaction. This paper presents the first optimized force-assisted ambiguous keyboard layout for the *index finger-to-forearm interaction*. We design this solution by combining the three following objectives: ease of force-touch, familiarity with the QWERTY layout, and goodness of the character pair configuration. (3) We evaluate the proposed solution (Trimetric optimized Force9) through a user experiment with 10 participants. Users achieve a text entry speed of 10.8 Word-per-Minutes (WPM) with error rates of below 3 % in the final testing session.

The paper is organized as follows. After discussing the related work in Section 2, we validate our initial intuition and explain our design choices in Section 3. We then present our layout optimization in Section 4. Accordingly, we evaluate the performance of the proposed layouts (Section 5), and discuss our findings in Section 6.

2 RELATED WORK

We discuss the most relevant works in this section: text entry with tiny touchscreens, force dimension, and layout optimization.

2.1 Text Entry on Tiny Touchscreen Interfaces

As keys on miniature QWERTY keyboards on smartwatches are hard to choose precisely, several works group the characters into multiple partitions for easier selection. Zoomboard (9.3 WPM) [2] requires the user to zoom into a region of interest on a QWERTY keyboard before selecting the target character. Other similar zoom-based miniature QWERTY keyboards such as CallOut and ZShift [5] achieve similar performance, up to 9.10 WPM. DriftBoard (9.74 WPM) [4], a panning-based interaction technique, controls cursors on the miniature QWERTY keyboard. Splitboard (14 WPM) [1], Swipeboard (10.7 WPM) [3], HoldBoard (9.10 WPM) [6], and SwipeKey [8] divide the keyboard into a hierarchy of which multiple partitions contain several characters. However, two or more consecutive gestures are required to complete one character selection, as a first gesture performs a downward search in the top-to-down traversal. Other techniques (up to 24 WPM) seek a single tap on an ambiguous key containing several characters. Disambiguation of characters on the same key can be achieved by statistical decoding [9], gesture recognition [10], predictive input [11], identification of fingers using additional on-finger sensors [7], or finger tap classification by machine learning [12]. In comparison, Force9 enables single-gestures inputs without additional sensors or complex computations.

2.2 Force-assisted Text Input

Prior works prove the human ability to exert force in discrete levels (6 levels [13], and up to 10 levels [14] with sufficient feedback) and continuous spectrums [15]. However, there are few works on force-assisted text input for tiny touchscreen interfaces [16]. Only two prior works apply force-sensitive interfaces on mobile devices. Hsiu et al.(12.4 WPM) [17] propose an ambiguous QWERTY keyboard on smartwatches in which every key contains two characters. The two characters in the same key are distinguished by two discrete levels of force exerted on the screen. A force-assisted scanning ambiguous keyboard (4.2 – 11 WPM) [18] requires a thumb-sized interface to exert force and select characters. Other variants include the thumb interaction within a palm area (6.47 WPM) [19] and the desktop scenario (33 WPM) [20]. In contrast, we are interested in

the feasibility and optimization of an ambiguous keyboard with multiple force levels.

2.3 Optimization of Keyboard Layout

Numerous studies propose multi-metric optimizations for keyboard layouts on smartphones. Bi et al. [21] consider the long-term efficiency and ease of visual search for soft keyboards. Dunlop et al. [22] strive to balance the keypad size with the prediction problem on the semi-ambiguous keyboards of mobile phones. In another work [23], they present a triple metric optimization for keyboard layouts considering speed, familiarity, and spell-checking ambiguity. Other works [24, 25] find a Pareto frontier between multiple criteria such as gesture clarity, gesture speed, the similarity to the QWERTY keyboard, and comfort level. Our solution shares the first two criteria proposed by the prior works. However, we introduce the multiple force levels on a tiny touchscreen as a third criterion to our optimization. To the best of our knowledge, this paper is the first work to consider the easiness of force exertion in the evaluation of the keyboard arrangement (optimization) problem for *index finger-to-forearm interaction* on miniaturized keyboards.

3 A PILOT STUDY AND DESIGN CHOICES

The keyboard layout, that we define as the required number of keys and the available characters within a key, is subject to the user's ability to control the exerted pressure. Prior works [13, 14] investigate the human ability to control the force level on a continuous spectrum with visual clues. A recent work by Yeo et al. [26] studies the accuracy of force-assisted gestures on small touchscreens such as hold-and-release, and twist-and-pan gestures. In this paper, we consider text input, which involves swift and repetitive force-assisted taps on the keys. We regard force-assisted clicks on the keys as quick and discrete actions.

We examine the user's ability to distinguish the force levels on a touchscreen. We recruited ten participants from the local university campus (ages ranging from 19 to 29, all right-handed). Half of them owned a pressure-sensitive smartphone, but none had prior experience in pressure-assisted interaction. We implemented the experimental system on an *iPhone 7 plus* equipped with a pressure-sensitive touchscreen (5.5"). The touchscreen can detect pressures ranging continuously from 0 to 3.3 N, corresponding to 0 to 6.66 units in Swift programming. The range is large enough to be split into several discrete levels. The system records the accuracy of the force-assisted clicks. A crossbar serves as visual clues regarding the target level spanning. Users perform Quick Release [27]to complete the force-assisted clicks. The zone confirmation is estimated by the averaged force level of the last 240 ms before the finger is released [28]. We used square-shaped keys [29] sized of 5.5 * 5.5 mm [8] to maximize the tap speed and minimize the error rate.

The pilot study aims to evaluate the accuracy of users in distinguishing several discrete force levels. We focused on the number of errors when performing force-assisted taps. We considered four force level configurations: 2-level, 3-level, 4-level, and 5-level. The 2-level, 3-level, and 4-level configurations feature two force ranges (0N – 0.45N and 0.45N – 3.3N), three force ranges (0N – 0.45N, 0.45N – 0.90N and 0.90N – 3.3N), and four force ranges (0N – 0.45N, 0.45N – 0.90N, 0.90N – 1.35N, 1.35 – 3.3N), respectively. For the

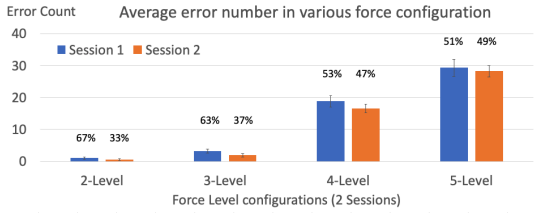

Figure 2: Average error number of Force-assisted clicks.

5-level configuration, each level spans over 0.45N as follows: 0N – 0.45N, 0.45N – 0.9N, ... 1.8N – 3.3N. We held two sessions on two successive days. In each session, we asked the participants to press on each level 10 times in continuous order, for instance, 1-2-3-4-5-1-2-3-4-5 for the 5-level configuration. In the briefing, the users had 10 minutes to familiarize themselves with the interface. Overall, this experiment consists of 2,800 trials (140 force-assisted clicks x 2 sessions x 10 participants).

Figure 2 shows the average number of errors per session for the four configurations. The error bars show the standard deviation. The average accuracy (and total error / total trial) for 2-level, 3-level, 4-level, and 5-level are respectively 96.25% (15/400), 91.50% (51/600), 55.88% (353/800), 42.50% (575/1000). The decreasing accuracy from 2-level to 5-level aligns with the finding in prior work [17, 30]. The accuracy drops dramatically for 4-level and 5-level, with errors representing more than half of the inputs. One-way ANOVA shows that the force level configuration directly impacts the number of errors of force click ($F_{3,36}$ = 309.61, p < .001). Bonferroni and Holm multiple pairwise comparisons show that the 2-level and 3-level configurations feature an insignificant difference in the number of errors (Bonferroni p = 0.5995), while other comparisons display the significant effect of the force level configuration (p < .001). Participants had trouble distinguishing two or more levels between the two ends of the force span. In contrast, the majority of users (8 out of 10) easily applied a force-assisted click to the two ends of the force span while feeling confident with the middle level of 3-level configuration. This indicates that the 2-level and 3-level configurations are more accurate than 4-level and 5-level. We also notice a slight improvement in each configuration during the second session as users got accustomed to the task.

Regarding the choice between 2-level and 3-level, we first consider the constrained space on the touchscreen of the Apple Watch. Each key occupies 30.25 mm^2 (5.5 mm x 5.5mm), regarded as the minimal size for comfortable key selection [8]. The 2-level, 3-level, 4-level, and 5-level configurations respectively need 14, 9, 7, 6, and 5 keys to accommodate the 27 characters, corresponding to 28.09%, 18.06%, 14.04%, 12.04%, and 10.00% of the Apple Watch's screen. The 4-level and 5-level configurations show a low improvement in space-saving compared to the 3-level configuration, while their accuracy is not appropriate for text input tasks. The 2-level configuration needs 55.56% more space than the 3-level configuration, with the 3-level configuration showing comparable accuracy to the 2-level configuration. Therefore, the 3-level configuration represents a compromise between accuracy and space-saving. By using the 3-level configuration, we reduce the size of the keyboard to 16.5

mm * 16.5mm. Considering the implementation environment on an Apple watch (42.0 mm * 35.9 mm), only 18.05% of the screen is occupied. In contrast, ZoomBoard[2], SwipeBoard [3], ZShift and CallOut [5], and SpiltBoard [1] take from 50% to 75% of the screen area.

Based on the pilot study, one ambiguous key maps to three characters in the 3-level configuration. The user selects the character by tapping with different forces defined as follows: *light tap*, *shallow force touch* and *deep force touch*. We ask a follow-up question to understand the users' force disambiguation on miniature keys independently from the optimized layout influence. All ten participants ranked the easiness of selecting the target force level in descending order as follows: (i) *light tap* (the 1st force level), (ii) *deep force touch* (The 3rd force level), and (iii) *shallow force touch* (The 2nd force level). The users reflected that the *light tap* is the usual interaction on smartphones touchscreens. The *deep force touch* was easy to reach because users exerted maximum force to complete the character input. The *shallow force touch* was regarded as the hardest one as accurate force between the two thresholds is required. Accordingly, this paper considers three **3-level and 9-key layouts**, as follows.

Alphabetical Multi-tap : The alphabetical layout (Figure 7c) is commonly available in the feature phones [31], which serves as our *baseline* condition. Multi-taps on the key have been employed to disambiguate three characters with the same key. That is, the alphabetical multi-tap layout requires the user to perform multiple taps for the 2nd- and 3rd-level characters. The text entry rate of the alphabetical multi-tap layout has been measured within the range between 5.33 and 10.53 Words-per-Minute (WPM) for novices and experts [32]. However, prior work was conducted on larger oval-shaped physical keys. Moreover, prior works suggest that text entry on touchscreens are more error-prone than physical keypads [33, 34]. Thus, we re-evaluate the alphabetical multi-tap layout on the miniature keys on a small-sized touchscreen.

Alphabetical Force9: The T9 layout and the alphabetically ordered layout are commonly applied for testing new modalities of text entries [10][11]. Users know the character order instinctively [35], which leads to performance improvement [36] and better usability [37] for novices. The alphabetical Force9 layout (Figure 7b) is identical to the T9 layout. However, the key disambiguation of this layout is accomplished by selecting the appropriate force level mapped to the three characters in one key. Force-assisted disambiguation enables the user to use a single force-assisted tap to select the 2nd- and 3rd-level characters within one discrete tap. For example, the word phrase 'COOL' needs 12 taps with alphabetical multi-tap layout and 4 taps with force-assisted T9 layout (4 *deep force touch*) if no typing error happens. The situation worsens with the number of characters in the word phrase, for instance, '*FORCEFULLY*' requires 27 taps with alphabetical multi-tap layout and 10 taps with a force-assisted T9 layout (1 *light tap*; 1 *shallow force touch*; and 8 *deep force touch*) if no typing errors happen. Specifically, the easiness of text entry and text entry speed are influenced by the Key Stroke Per Character (KSPC) [38]. For instance, Zoomboard (2.15 KSPC), and Spiltboard (1.85 KSPC) show no significant difference with alphabetical multi-tap (2.03 KSPC). Thus, we derive a force-assisted experimental condition based on the T9 layout.

Trimetric Optimized Force9: This layout (Figure 7d) employs the same disambiguation methods as the Alphabetical Force9 but

Figure 3: Interaction on ambiguous keyboards under a 3-level force spectrum.

the character arrangement in this layout is optimized by three metrics. For instance, the first character in the word phrase, 'F', is located in the key of 'OKF' (Figure 3), where the user applies a *deep force touch* to select the character, with an exerted force of the maximum level in the spectrum. The prior works [21, 22] demonstrate that the optimization of keyboard layout can improve text entry performance, and our optimization model (Section 4) introduces a metric to enhance the easiness of force-assisted disambiguation.

4 OPTIMIZATION OF KEYBOARD LAYOUT

The alphabetically ordered keyboard does not consider the constraints of the force-assisted configuration. We formulate our optimization problem and propose the corresponding optimized keyboard layout.

4.1 Maximizing the goodness of character pair

The goodness of a character pair in a candidate solution keyboard $k \in K$ is subject to two key factors: (i) the finger movement distance and (ii) the ambiguity for auto-correction.

We first address the finger movement distance. The time to input a character on the soft keyboard can be divided into two temporal factors – the time for finger movement from one key to another and the time for targeting on the character key. The nearer and bigger a key is, the quicker the character input. For our keyboard configuration design, we design all keys to be of identical size (5.5 mm * 5.5mm [8]).

We compute the frequencies $a_{i,j}$ of key pairs by building the Bigram of character pairs in the selected English corpus [39]. Let α={a,b,c,...,x,y,z,_} be the alphabet we consider, $Bigram_{i,j}$ refers to the probability of the finger moving from character i to character j. We build a character pairing table of size 727 (27*27), $Bigram_{i,j}$, by normalizing the frequencies as $Bigram_{i,j} = a_{i,j} / \sum_{\forall i,j \in \alpha} a_{i,j}$. Zhai et al. [40] indicate that the source of the corpus does not significantly influence the optimization of the keyboard layout. Our analysis shows that the most common character pair is E_, where _ means white space, with 102,736,597,698 occurrences in our corpus. The probability of finger movement between E to _ is 0.0354. The next most frequent character pairs are ER ($P_{ER} = 0.0311$), S_ ($P_{S\_} = 0.0291$), TH ($P_{IN} = 0.0220$), ES ($P_{ES} = 0.0199$), and IT ($P_{IT} = 0.0190$), while the least probable character pairing was QJ with a 0.0000005 probability. Next, we normalize the probability to a Bigram score, $Bi_{i,j}$ defined as follows: $Bi_{i,j} = \frac{100 * Bigram_{i,j}}{max(Bigram_{i,j})}$. For instance, the most common character pair E_ and ER will respectively get 100.00 and 88.10 points.

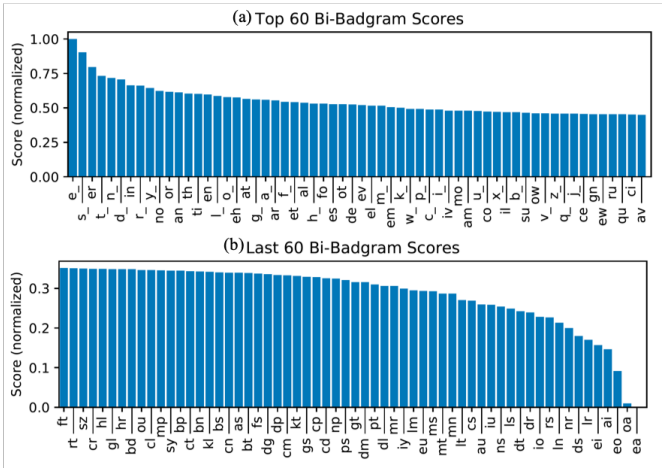

Figure 4: (a) Top 60 bi-badgram; and (b) last 60 bi-badgram.

Tapping on a soft keyboard is error-prone especially with small keys on a constrained interface [8]. Automatic correction algorithms alleviate the issue but they have limitations on checking valid words. For example, the words 'hit' and 'hat' are both valid and the ambiguity issue rises if the characters 'A' and 'I' are arranged together [23]. To address this issue, we build the table of *badgrams* [23] showing the 676 (26*26) possible character pairs. As the white space character is used to separate words, we exclude it from the Badgram. We scan all words of identical length in the corpus and count the Badgram frequency of the character pair, $b_{i,j}$, by checking whether a character substitution in a word leads to another valid word. We then normalize the frequencies as probabilities $Badgram_{i,j} = b_{i,j}/\sum_{\forall i,j \in \alpha} b_{i,j}$. Our analysis gives the following top Badgrams: AE with $P_{AE} = 0.0175$, AO with $P_{AO} = 0.0146$, AI with $P_{AI} = 0.0125$, EO with $P_{EO} = 0.0121$, EI with $P_{EI} = 0.0118$. Having A and E as a neighbouring pair leads to many ambiguous words such as *and* instead of *end*, *ha* instead of *he*, *bat* instead of *bet*, and so on. Finally, a normalization of the probability of Badgram results in the Badgram score, $Bad_{i,j} = 100 * Badgram_{i,j}/\max(Badgram_{i,j})$.

Finally, we obtain the satisfaction score of the character pairs, $CP_{i,j} = Bi_{i,j} - Bad_{i,j}$ [22]. For combinations including the white space, $CP_{i,j} = Bi_{i,j}$ We aim to find the most common character pairs which are not likely to trigger ambiguity with the auto-correction software. Figure 4a shows the most preferred 60 bi-badgram and their satisfaction scores, according to the computed Bigrams and Badgrams. The 10 most preferred character pairs of E_, S_, ER, T_, N_, D_, IN, R_, and Y_ are ranked in the top 10. Figure 4b shows the least preferred 60 ranked bi-badgram and their satisfaction scores. The 10 least preferred character pair are AV, CI, QU, RU, EW, GN, CE, J_, Q_, and Z_. The goodness of the character pair configuration, M_{pair}, can be computed by summing the satisfaction score if the characters are neighbors, as follows.

$$O_1 : M_{pair} = \sum_{\forall i,j \in \alpha} \begin{cases} CP_{i,j} & \text{if char i \& j are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

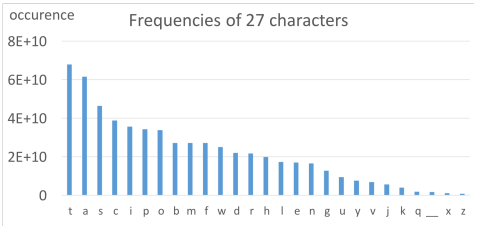

Figure 5: Frequencies of characters in the English corpus including white space '_'.

Prior keyboard optimizations define adjacent keys by their 2D geometric proximity. Force9 introduces a third dimension driven by the force level division. We regard two characters within a key as adjacent to each other if the associated force levels are continuous. In the alphabetical Force9 layout, character 'a' is adjacent to 'b' but not 'c' and character 'b' is adjacent to both 'a' and 'c'. Regarding geometric proximity, we consider three separate planes corresponding to the three force levels. For instance, 'a' is geometrically adjacent to 'd', 'j', and 'm', while 'h' is geometrically adjacent to 'e', 'n', and 'q'. We normalize the M_{pair} into M_{pair_n}, ranging from 0 to 1 by $M_{pair_n} = M_{pair}/1.1 * M_{pair\_best}$, M_{pair_best} being the highest scored keyboard and 1.1 an offset index to facilitate the optimal search.

4.2 Maximizing the familiarity with QWERTY layout

Most computer and smartphone users are accustomed to the QWERTY keyboard layout. Previous studies show that users learn and adopt QWERTY-like layouts faster than alphabetical layouts [41]. Among physical and virtual keyboards, the trapezoidal-shaped QWERTY configuration consists of 10, 9 and 8 characters on the first, second and third row. We make minor modifications to fit all the characters in the 9-keypad configuration. We compute the familiarity between the candidate keyboard and the 'starter layout' (Figure 6d) by rating the summed Euclidean distance of all the characters $i \in \alpha$ between the geometrical center of a given key in the candidate solution k_i to the center of the corresponding character position in the 'starter layout' and s_i, as the reference of home positions. The character positioned furthest away from their home position in the candidate solution will be punished. We evaluate the summed distance of a candidate solution, k_{dist} in the candidate solution set K as: $k_{dist} = \sum_{\forall i \in \alpha} Distance(k_i, s_i)^2$. We obtain the normalized familiarity score in the below. The lesser the distance generated from the character alternation, the higher the familiarity in this metric.

$$O_2 : M_{familiarity} = 1 - \frac{k_{dist}}{\max_{\forall k \in K}(k_{dist})}$$

4.3 Maximizing the easiness of force keypad interaction

Taking into account the user feedback in the pilot study, we arrange the three characters within a given key as follows: *light tap* – most preferred, *shallow force touch* – least preferred, and *deep force touch*.

Character layout	O_1	O_2	O_3	Sum
Alphabetical (Fig. 7b)	0.824	0.265	0.296	1.385
QWERTY (Fig. 6d)	0.763	1.000	0.370	2.133
Mono-OPT (O_1) (Fig. 6a)	0.909	0.3062	0.852	2.067
Bi-OPT (O_1,O_3) (Fig. 6b)	0.899	0.286	1.000	2.184
Tri-OPT (O_1–O_3) (Fig. 6c)	0.834	0.548	1.000	2.381

Table 1: Optimization results by genetic algorithm: O_1 – O_3 are the metrics and Sum are the sum of equal weighting from 0.00 to 3.00, where higher scores mean better objective functions, or vice versa.

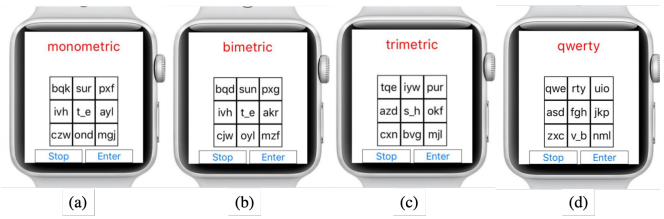

Figure 6: Layout on smartwatches: (a) Mono-metric (O_1) optimized layout; (b) Bimetric (O_1 + O_3) layout; (c) Trimetric (O_1 + O_2 + O_3) optimized layout; (d) QWERTY layout.

We assign the most frequent character tier to the *light tap* force level. The second most frequent character tier occupies the *deep force touch* force level. Finally, the least frequent character tier belongs to the *shallow force touch* level as users found it to be the most difficult to access. We rank the characters $CF_i \forall i, j \in \alpha$ in descending order and pair the three tiers with three force levels, based on the 27 character frequency analysis in our corpus: $Tier_1$ [t, a, s, c, i, p, o, b, m] – *light tap*, $Tier_2$ [f, w, d, r, h, l, e, n, g] – *deep force touch*, $Tier_3$ [u, y, v, j, k, q, '_', x, z] – *sallow force touch*. Our intuition is analogous to a prior work [42] that modifies the multi-press T9 keyboard by character frequency. We compute the ease of use of our force keyboard by the summed weighted score of character-tier matches. We define a reward scheme to give a score to those configurations if the characters in the candidate solution are assigned to the matched tiers. As the weighted score is summed by the CF_i (no normalization).

$$O_3 : M_{force} = \sum_{\forall i \in \alpha} \begin{cases} CF_i & \text{if character i in the matched tier} \\ 0 & \text{otherwise} \end{cases}$$

4.4 Optimized Force9 Layout

Table 1 shows the optimization results under a Force9 configuration. We intend to choose a layout that considers all the metrics, and generate the keyboards by progressively adding criteria, as follows. The keyboard on the third row introduces the goodness of character pair as top-priorities. The layout on the fourth row considers both the goodness of character pair and the ease of force keypad interaction. The trimetric optimized layout (final row) considers all the metrics and scores the highest among all the candidate keyboards. The results show that keyboard layouts can maintain both the goodness of character pair and ease of force keypad interaction. The bi-metric layout reduces the goodness of character pair by 1.1% compared to the mono-metric layout (Figure 6). However, the ease of force keypad interaction increases by 17.37%. Similarly, the trimetric layout shows a reduction of 8% in the goodness of character pair, which is a satisfactory trade-off for a 51.38% improvement in the familiarity with the QWERTY layout and a 17.37% improvement to the ease of force interaction. Figure 6c shows the optimized Force9 layout. This layout scores 71.9% higher than the alphabetical layout and 11.6% higher than the QWERTY layout.

In summary, the alphabetical constrained layout neglects the familiarity with the QWERTY-like keyboard (O_2) and the ease of force keypad interaction (O_3), while the QWERTY layout does not satisfy the ease of force keypad interaction. In comparison, the trimetric optimized layout achieves a comparable score (2.381) to the QWERTY layout (2.067) with a drastic improvement in force keypad interaction.

5 EVALUATION

To investigate how the users perform with the three keyboard layouts, we conduct a **Within-Subjects** evaluation, as follows.

5.1 Participants, Keyboard and Apparatus

We recruited another 10 participants from our university campus (Age: 18 – 31). None of them owned a pressure-sensitive smartphone nor had prior experience with one.

Apple Watch is the only force-sensitive smartwatch currently available on the consumer market. The experiment was performed on a smartphone due to current limitations in smartwatches. We were able to implement a demonstration application; however, due to API restrictions, this implementation leads to constant back-and-forths between the force-sensitive keyboard and the parent interface after a short period of time. In long task assessment scenarios, it is inappropriate to ask the users to switch between the interfaces after every phrase. Due to similar issues, smartphones are usually employed in the literature [17, 43]. Figure 7a shows the testing environment, and Figure 7b and 7c illustrate the application interface of traditional alphabetical multi-tap and force-assisted T9 respectively. We arranged the characters in alphabetical order with '_' as the final character in the 3-level configuration.

The experimental system on an *iPhone 7 plus* device applies the force span ranges, as follows. In the force-assisted settings (Alphabetical and Trimetric Optimized Force9), Quick Release [27] is employed to select the characters in the keys. The key confirmation is computed by the averaged value of force level in the last 240 ms once the finger is released [28]. Before the beginning of the study, we further calibrated the *user-independent* cut-off points for the 3-level configuration to improve the ease of force-click. The force span ranges were as follows: 1st level (*light tap*): 0.0000 to 0.3718N, 2nd level (*shallow force touch*): 0.3719 to 0.9914N, and 3rd level (*deep force touch*): 0.9915N up to 3.300N (max). The visual cues dynamically shows the character mapping with one out of three force levels in the chosen ambiguous key. In the multi-tap setting, the participants performed multiple taps to select the characters in the keys in a regular T9 fashion. We applied a time-out of 400 ms to recognize the final tap on consecutive character switch [44], with

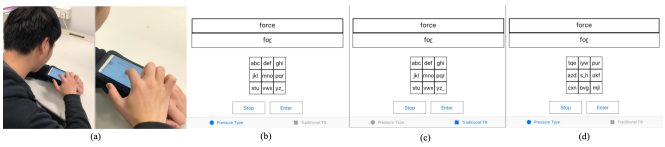

Figure 7: Testing Environment for Trimetric Optimized Force9 a) the sitting posture of a participant; A closer look at the layout on the smartphone touchscreen – b) Alphabetical Force9, c) alphabetical multi-tap, and d) Trimetric OPT Force9.

visual cues showing the currently tapped character. In all three layouts, the keys were 5.5 mm x 5.5mm [8], and the swipe gestures for the '*Enter*' and '*Delete*' keys were disabled.

5.2 Procedures

The participants had five minutes to familiarize themselves with three keyboard layouts: Alphabetical Multi-tap, alphabetical Force9 (Figure 7b) and trimetric optimized Force9 (Figure 6c). The participants were sitting during the entire experiment and typed on the smartphone in a stable and silent environment. We allowed the users to type with their index finger, and instructed them to type as fast and accurately as possible. We counterbalanced three keyboard conditions to alleviate the carry-over effect: 1) Alphabetical Multi-tap, 2) Alphabetical Force9, and 3) Trimetric Optimized (OPT) Force9. For each condition, we ran a total of five sessions, one per day for five days. Each of them requiring the user to type 15 word phrases from the MacKenzie's phrase set [45]. 1-minute breaks were done when necessary. Three conditions correspond to a total of 2,250 words (3 conditions x 15 word-phrases x 5 sessions x 10 participants). After the sessions on the first day and the fifth day, we asked the participants to complete the NASA TLX questionnaire about the Force9 layouts.

5.3 User Performance

Text entry rate: Figure 8 shows the character-level text entry rate (Word-per-Minute, WPM) with the trimetric optimized layout. The error bars represent the standard deviation. Two-way repeated measures ANOVA demonstrates the significant effect of **Layout** ($F_{2,135}$ = 25.68, p < .001) and **Session** ($F_{4,135}$ = 33.44, p < .001) to the *text entry rate*, with the existence of **Interaction** between **Layout** and **Session** ($F_{8,135}$ = 6.08, p < .001). The statistical significance in **Session** indicates a learning effect on the new layout. Participants achieved 7.43 WPM (σ = 1.86) on average with the trimetric optimized layout over the five sessions. The average text entry rate increased to 10.18 WPM (σ = 1.27) on the fifth day from 5.78 WPM (σ = 0.64) on the first day, showing a 76.01% improvement in speed. In contrast, the participants on the first day achieved an average of 5.61 WPM (σ = 0.99) and 6.45 WPM (σ = 1.08) for the alphabetical multi-tap and alphabetical Force9 layouts. Therefore, our results show that the initial performance of participants with the trimetric optimized layout was only 89.58% of the alphabetical layout. The participants with Trimetric optimized layout ($\bar{M}$ = 8.22 WPM, σ = 1.05) took four days to compensate up to 97.64% of the alphabetical Force9 layout ($\bar{M}$ = 8.99, σ = 0.84). On the fifth day, the performance of the trimetric optimized layout surpassed the alphabetical Force9 layout by 20.93% ($\bar{M}$ = 8.41 WPM, σ = 0.96) and the alphabetical multi-tap layout by 46.96% ($\bar{M}$ = 6.92 WPM, σ = 0.80). 4 out of 10 participants achieved 11.19 – 11.75 WPM in the fifth session, indicating that some fast-learning participants can achieve even higher text entry rates. The trimetric optimized Force9 layout outperforms the alphabetical Force9 layout that neglects the familiarity with the QWERTY-like keyboard and the easiness of the force keypad interaction. In addition, the two Force9 layouts significantly outperform the alphabetical multi-tap layout because of the difficulties in repetitive targeting of miniaturized keys. We also observe that the participants sometime over-tap on the key for the 2nd- and 3rd-level characters. Accordingly, the participants need to tap further to make a correction, for example, character 'B' is over-tapped as 'C' and the user needs two additional taps to revert to 'B', which deteriorates the text entry rate.

Error rate: Figure 9 shows the Uncorrected Error Rate (UER) of the trimetric optimized layout, where the error bars represent the standard deviation values. Two-way repeated measures ANOVA reveals statistical significance in the effect of **Layout** ($F_{2,135}$ = 47.10, p < .001) and **Session** ($F_{4,135}$ = 30.48, p < .001) to the *Error rate*, in addition to the existence of **Interaction** between **Layout** and **Session** ($F_{8,135}$ = 8.43, p < .001). Trimetric optimized Force9 achieved a mean UER of 3.98% (σ = 0.0140) throughout the five sessions. In comparison, the alphabetical layout achieved the averaged UER values of 6.35% (σ = 0.0180) for the force-assisted condition and 7.30% (σ = 0.0175) for the multi-tap condition. Alphabetical multi-tap layout is more erroneous than trimetric optimized layout due to the difficulty of targeting the miniaturized keys. Compared to the alphabetical Force9 layout, trimetric optimized Force9 layout considers the neighbouring positions of character pairs and allocates the most frequently used characters at the two ends of the force spectrum to improve the character easiness and the overall text entry performance.

The UER in the trimetric optimized Force9 condition improves significantly from 5.57% (σ = 0.0179) on the first day to 2.89% (σ = 0.0125) on the fifth day. On the first day, the users' unfamiliarity with the new layout leads to the initial high error rate. The errors are classified into two categories: (1) The user mistypes on the adjacent keys. (2) The user mistakenly selects a neighboring character inside an ambiguous key due to unintended force exertion.

On the first day, 60.30% of erroneous characters on the trimetric optimized layout are caused by the first type of error (Spatial error), while 39.70% of errors are caused by the second type of error (Force error). On the fifth day, the overall error rate drops and the percentage of the first type of error decreases to 32.31%. The users are more familiar with the trimetric optimized layout and hence the amount of mistyping on the adjacent keys decreases. On the fifth day, 67.69% of errors belong to the second type. The overall error rate improves because the absolute number of the second type error decreases slightly from 17 (1st day) to 15 (5th day) ($\bar{M}$ = 14.84, σ = 1.90). The participants show an improvement in controlling the force exertion for the new input modality. The average rate of the second type of error is 1.97% (74 out of 3750 characters) across the 5-day session. In the Trimetric Optimized Force9 layout, the most frequent characters are assigned to the two ends of the

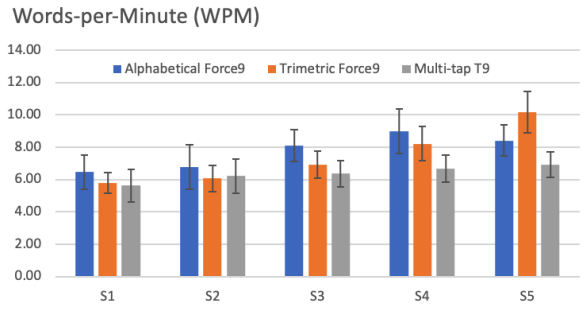

Figure 8: Mean text entry speed over 5-day sessions.

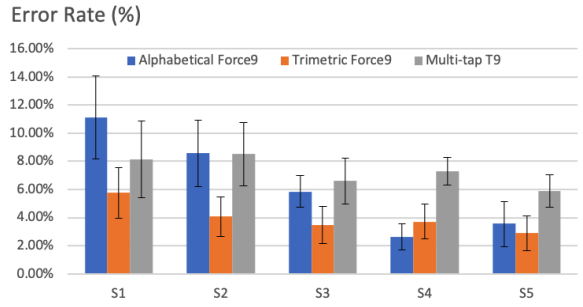

Figure 9: Mean uncorrected error rate over 5-day sessions.

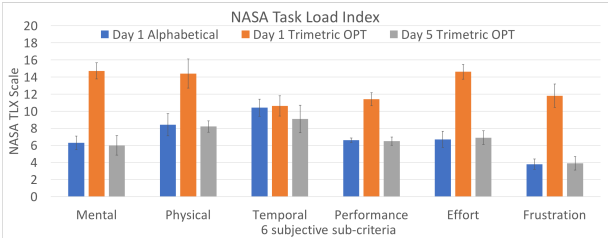

Figure 10: User Taskloads in NASA TLX.

3-level force span, i.e. '*light tap*' and '*deep force touch*', while the least probable characters located at the middle position ('*shallow force touch*') inside the keys, where '*light tap*', '*shallow force touch*' and '*deep force touch*' respectively correspond to 63.09%, 6.54%, and 30.37% of the character occurrences (Figure 5).

NASA Task Load Index Our NASA TLX questionnaire aims to investigate the user loading to the unfamiliar layouts, so we record the alphabetical layout on the first day and keep track with the trimetric optimized layout on the first and fifth days. Figure 10 shows the results of NASA TLX [46]. Student's T-tests between each session with the alphabetical layout and the trimetric optimized layout on the first day show a significantly higher mental load ($p < .001$), physical load ($p < .001$), performance ($p < .001$), effort ($p < .001$), and frustration ($p < .001$) but no significant difference in the temporal factor ($p = 0.841$). T-tests yielded no significant differences in all aspects between the alphabetical layout on the

Figure 11: Miniaturized Force9 vs a coin of 1-cent USD.

first day and the trimetric optimized layout on the fifth day ($p < .001$). We conclude that the participant's perceived load significantly decreased over the 5-day sessions. To sum up, the user perception on the trimetric keyboard improved during the 5-day period. As the alphabetical order is ingrained in the participants' memory, the participants prefer alphabetical order over the trimetric optimized layout. After practicing the trimetric optimized layout over the 5-day sessions, their typing performance enhanced significantly and hence improved user perception and had higher user acceptance results.

6 DISCUSSION

Alternative Disambiguation Strategy: One prior work has stated that the modality of force is non-deterministic and transient [28]. Instead of designing probabilistic algorithms to manage such transient modality, we demonstrate an alternative solution by computing a layout to reduce the likelihood of which the users will encounter the error-prone selection in the ambiguous keyboard. The optimization enables users to achieve more accurate character selection on the ambiguous keys through a single force-assisted gesture, reducing from 6.35% UER for alphabetical layout to 3.98% UER for the trimetric optimized layout (Section 5). Additionally, other existing works relying on consecutive gestures for one character input could introduce burden in the text entry tasks. That is, the users have to memorize the association between the swipes/taps and the characters, such as SwipeBoard [3] (13.30% error rate). In contrast, the users can understand our layout at a glance, although learning a new layout is less favorable. Our layout also encourages the users to perform one discrete press, knowing that the visual occlusion on the miniaturized keys [47] can degrade the user performance [48]. That is, every step in the multi-gesture interaction could possibly introduce errors, especially when the users' fingers cause visual occlusion, for instance, multi-taps for zooming-in the keyboard and selecting a key in Zoomboard [2] (14% error rate). It is worth mentioning that our solution does not install any addendum sensors on the finger position for key disambiguation [7].

Force-assisted Text Entry: The average character-level entry rate of trimetric optimized Force9 reaches 10.18 WPM during the final trial. We acknowledge that other latest text entry solutions designated for smartwatches demonstrate a faster text entry rate than our approach, such as Zoomboard [2] (9.3 WPM, 8th trial) ZShift [5] (5.4 – 9.1 WPM, depending on key sizes), DriftBoard [4] (9.7 WPM, 10th trial), HoldBoard (10.24 WPM, 12nd trial) [6], SwipeKey [8] (11.0 WPM, 6th trial), and SplitBoard [1] (19.58 WPM, 2-hour training). The majority of these works employ deterministic gestures

(e.g. taps, swipes, panning) on the touchscreens of highly familiar layouts such as standard QWERTY (Zoomboard, ZShift, DriftBoard, HoldBoard, and SplitBoard) and alphabetical (SwipeKey) layouts. It is important to note that Force9 occupies a unique space of input modality with transient force, demonstrating the feasibility of 3-level ambiguous keys within a 9-key layout. Notable text entry systems leveraging force-assisted disambiguation with *index finger-to-forearm* interaction are as follows: ForceBoard (12.47 WPM, 5th session) [17] containing 2-level ambiguous keys and 15-key standard QWERTY layout; and an ambiguous (1-line) scanning keyboard (4.2 – 11 WPM, without/with word prediction).

Design Implications for Smart Wearables: Force-sensitive layouts can efficiently shrink the on-screen keyboard size, proportionally to the number of force levels employed. Considering the limit of the human ability of force disambiguation, a 2-level force-assisted layout and the 3-level force-assisted layout can respectively reduce the keyboard size by a half to two-thirds of the original size. As shown in Figure 11, achieving the size as small as a one-cent USD coin. Also, a rising number of smart wearables are being launched on the market. Such wearables include smart rings and smart wristbands, which present a screen real estate even smaller than smartwatches. Force9 can serve as a promising text entry approach for these size-constrained smart wearables, where only a tiny area of the force-sensitive touch interface is available.

Limitations and future works: The current user study shows that users can become familiar with the trimetric optimized layout in 5 days and recover full performance on the fifth day compared to the alphabetical layout. Our work has the following limitations: a) form-factor difference between smartphones and smartwatches; b) limited proofs of mobility (e.g. walking postures) and the user performance with smart watches in the wild; c) a potential difference in the user text entry rate and accuracy with alternative force confirmation techniques; and d) impact of arm postures to the user performances. For future work, we will design a longer period for user studies to investigate the ultimate performance of trimetric optimized Force9 and explore the user behavior with the sub-optimized keyboard layouts, for instance, Bimetric vs Trimetric layout. Our optimizer computes several keyboard layouts of comparable score in O_1 and O_3 but displays large score variations in O_2. An interesting future research direction would be to explore the user behavior with these various keyboard layouts, for instance, Bimetric vs Trimetric layout.

7 CONCLUSION

Force9 serves as an alternative approach to performing text input for the tiny touchscreen of smartwatches, by leveraging the force-sensitive touchscreens of smart wearable, significantly reducing the on-screen keyboard's size (2.7 cm^2.). We believe our work presents an optimized keyboard layout considering the force dimension, and enhances the understanding of text entry on 9-keypad ambiguous layouts. The 3-level layout balances the on-screen size (18.05% of the screen) and the user's ability to distinguish the force exertion exerted (91.5% accuracy). We then optimized the keyboard layout for our 3-level configuration under three equally weighted metrics. Accordingly, we compared the performance of force disambiguation and multi-tap disambiguation under the T9-like layout, in which our participants achieved 10.18 WPM on the 5th day. The prominent feature of the miniaturize-sized layout enables Force9 to work on the common wearables such as the spectacle frame of smartglasses, smart jewellery and finger-worn computers.

ACKNOWLEDGEMENTS

The authors would like to acknowledge 5G-VIIMA and REBOOT Finland IoT Factory projects funded by Business Finland, 5GEAR project and the 6G Flagship project funded by the Academy of Finland (Decision No. 318927), and project 16214817 from the Research Grants Council of Hong Kong. We would also like to thank the reviewers for their generous suggestions.

REFERENCES

[1] Jonggi Hong, Seongkook Heo, Poika Isokoski, and Geehyuk Lee. Splitboard: A simple split soft keyboard for wristwatch-sized touch screens. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1233–1236, New York, NY, USA, 2015. ACM.

[2] Stephen Oney, Chris Harrison, Amy Ogan, and Jason Wiese. Zoomboard: A diminutive qwerty soft keyboard using iterative zooming for ultra-small devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 2799–2802, New York, NY, USA, 2013. ACM.

[3] Xiang 'Anthony' Chen, Tovi Grossman, and George Fitzmaurice. Swipeboard: A text entry technique for ultra-small interfaces that supports novice to expert transitions. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 615–620, New York, NY, USA, 2014. ACM.

[4] Tomoki Shibata, Daniel Afergan, Danielle Kong, Beste F. Yuksel, I. Scott MacKenzie, and Robert J.K. Jacob. Driftboard: A panning-based text entry technique for ultra-small touchscreens. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 575–582, New York, NY, USA, 2016. ACM.

[5] Luis A. Leiva, Alireza Sahami, Alejandro Catala, Niels Henze, and Albrecht Schmidt. Text entry on tiny qwerty soft keyboards. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 669–678, New York, NY, USA, 2015. ACM.

[6] Sunggeun Ahn, Seongkook Heo, and Geehyuk Lee. Typing on a smartwatch for smart glasses. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces*, ISS '17, pages 201–209, New York, NY, USA, 2017. ACM.

[7] Aakar Gupta and Ravin Balakrishnan. Dualkey: Miniature screen text entry via finger identification. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 59–70, New York, NY, USA, 2016. ACM.

[8] Yuan-Fu Shao, Masatoshi Chang-Ogimoto, Reinhard Pointner, Yu-Chih Lin, Chen-Ting Wu, and Mike Chen. Swipekey: A swipe-based keyboard design for smartwatches. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '16, pages 60–71, New York, NY, USA, 2016. ACM.

[9] Mitchell Gordon, Tom Ouyang, and Shumin Zhai. Watchwriter: Tap and gesture typing on a smartwatch miniature keyboard with statistical decoding. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 3817–3821, New York, NY, USA, 2016. ACM.

[10] Aske Mottelson, Christoffer Larsen, Mikkel Lyderik, Paul Strohmeier, and Jarrod Knibbe. Invisiboard: Maximizing display and input space with a full screen text entry method for smartwatches. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '16, pages 53–59, New York, NY, USA, 2016. ACM.

[11] Pui Chung Wong, Kening Zhu, and Hongbo Fu. fingert9: Leveraging thumb-to-finger interaction for one-handed text entry on smartwatches. In *SIGGRAPH Asia 2017 Mobile Graphics & Interactive Applications*, SA '17, pages 9:1–9:3, New York, NY, USA, 2017. ACM.

[12] Hyunjae Gil, DoYoung Lee, Seunggyu Im, and Ian Oakley. Tritap: Identifying finger touches on smartwatches. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 3879–3890, New York, NY, USA, 2017. ACM.

[13] Sachi Mizobuchi, Shinya Terasaki, Turo Keski-Jaskari, Jari Nousiainen, Matti Ryynanen, and Miika Silfverberg. Making an impression: Force-controlled pen input for handheld devices. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1661–1664, New York, NY, USA, 2005. ACM.

[14] Graham Wilson, Craig Stewart, and Stephen A. Brewster. Pressure-based menu selection for mobile devices. In *Proceedings of the 12th International Conference*

on Human Computer Interaction with Mobile Devices and Services, MobileHCI '10, pages 181–190, New York, NY, USA, 2010. ACM.

[15] Yui-Pan Yau, Lik Hang Lee, Zheng Li, Tristan Braud, Yi-Hsuan Ho, and Pan Hui. How subtle can it get? a trimodal study of ring-sized interfaces for one-handed drone control. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 4(2), June 2020.

[16] L.H. Lee, Y. Zhu, Y.P. Yau, T. Braud, X. Su, and P. Hui. One-thumb text acquisition on force-assisted miniature interfaces for mobile headsets. In *2020 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10, 2020.

[17] Min-Chieh Hsiu, Da-Yuan Huang, Chi An Chen, Yu-Chih Lin, Yi-ping Hung, De-Nian Yang, and Mike Chen. Forceboard: Using force as input technique on size-limited soft keyboard. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, MobileHCI '16, pages 599–604, New York, NY, USA, 2016. ACM.

[18] Mingyuan Zhong, Chun Yu, Qian Wang, Xuhai Xu, and Yuanchun Shi. Forceboard: Subtle text entry leveraging pressure. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 528:1–528:10, New York, NY, USA, 2018. ACM.

[19] Lik Hang Lee, Kit Yung Lam, Tong Li, Tristan Braud, Xiang Su, and Pan Hui. Quadmetric optimized thumb-to-finger interaction for force assisted one-handed text entry on mobile headsets. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 3(3), September 2019.

[20] Xin Yi, Chen Wang, Xiaojun Bi, and Yuanchun Shi. 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*, CHI '20, page 1–13, New York, NY, USA, 2020. Association for Computing Machinery.

[21] Xiaojun Bi, Barton A. Smith, and Shumin Zhai. Quasi-qwerty soft keyboard optimization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 283–286, New York, NY, USA, 2010. ACM.

[22] Mark D. Dunlop, Naveen Durga, Sunil Motaparti, Prima Dona, and Varun Medapuram. Qwerth: An optimized semi-ambiguous keyboard design. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services Companion*, MobileHCI '12, pages 23–28, New York, NY, USA, 2012. ACM.

[23] Mark Dunlop and John Levine. Multidimensional pareto optimization of touchscreen keyboards for speed, familiarity and improved spell checking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 2669–2678, New York, NY, USA, 2012. ACM.

[24] Brian A. Smith, Xiaojun Bi, and Shumin Zhai. Optimizing touchscreen keyboards for gesture typing. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 3365–3374, New York, NY, USA, 2015. ACM.

[25] Jun Gong, Zheer Xu, Qifan Guo, Teddy Seyed, Xiang 'Anthony' Chen, Xiaojun Bi, and Xing-Dong Yang. Wristext: One-handed text entry on smartwatch using wrist gestures. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 181:1–181:14, New York, NY, USA, 2018. ACM.

[26] Hui-Shyong Yeo, Juyoung Lee, Andrea Bianchi, and Aaron Quigley. Watchmi: Pressure touch, twist and pan gesture input on unmodified smartwatches. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '16, pages 394–399, New York, NY, USA, 2016. ACM.

[27] Sachi Mizobuchi, Shinya Terasaki, Turo Keski-Jaskari, Jari Nousiainen, Matti Ryynanen, and Miika Silfverberg. Making an impression: Force-controlled pen input for handheld devices. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1661–1664, New York, NY, USA, 2005. ACM.

[28] Christian Corsten, Simon Voelker, and Jan Borchers. Release, don't wait! reliable force input confirmation with quick release. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces*, ISS '17, page 246–251, New York, NY, USA, 2017. Association for Computing Machinery.

[29] Seungyon Lee and Shumin Zhai. The performance of touch screen soft buttons. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 309–318, New York, NY, USA, 2009. ACM.

[30] Jibin Yin, Xiangshi Ren, and Shumin Zhai. Pen pressure control in trajectory-based interaction. 29:137–148, 03 2010.

[31] Dale L. GroverMartin T. KingClifford A. Kushler. Reduced keyboard disambiguating computer, May 7 1995. US Patent 5818437A.

[32] Christina L. James and Kelly M. Reischel. Text input for mobile devices: Comparing model prediction to actual performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '01, pages 365–371, New York, NY, USA, 2001. ACM.

[33] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. Evaluation of a new error prevention technique for mobile touchscreen text entry. In *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration*, OzCHI '13, pages 397–400, New York, NY, USA, 2013. ACM.

[34] James Clawson, Kent Lyons, Alex Rudnick, Robert A. Iannucci, Jr., and Thad Starner. Automatic whiteout++: Correcting mini-qwerty typing errors using keypress timing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 573–582, New York, NY, USA, 2008. ACM.

[35] L. H. Lee, K.Y. Lam, Y.P. Yau, T. Braud, and P. Hui. Hibey: Hide the keyboard in augmented reality. In *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom*, pages 1–10, 2019.

[36] Shumin Zhai and Barton A Smith. Alphabetically biased virtual keyboards are easier to use: Layout does matter. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '01, pages 321–322, New York, NY, USA, 2001. ACM.

[37] Jun Gong and Peter Tarasewich. Alphabetically constrained keypad designs for text entry on mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 211–220, New York, NY, USA, 2005. ACM.

[38] I. Scott MacKenzie. Kspc (keystrokes per character) as a characteristic of text entry techniques. In *Proceedings of the 4th International Symposium on Mobile Human-Computer Interaction*, Mobile HCI '02, pages 195–210, London, UK, UK, 2002. Springer-Verlag.

[39] T. Segaran and J. Hammerbacher. *Beautiful Data: The Stories Behind Elegant Data Solutions*. Theory in practice. O'Reilly Media, 2009.

[40] Shumin Zhai, Michael Hunter, and Barton A. Smith. The metropolis keyboard - an exploration of quantitative techniques for virtual keyboard design. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, pages 119–128, New York, NY, USA, 2000. ACM.

[41] Sunyu Hwang and Geehyuk Lee. Qwerty-like 3x4 keypad layouts for mobile phone. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1479–1482, New York, NY, USA, 2005. ACM.

[42] Hokyoung Ryu and Katrina Cruz. Letterease: Improving text entry on a handheld device via letter reassignment. In *Proceedings of the 17th Australia Conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future*, OZCHI '05, pages 1–10, Narrabundah, Australia, Australia, 2005. Computer-Human Interaction Special Interest Group (CHISIG) of Australia.

[43] Hiroki Kurosawa, Daisuke Sakamoto, and Tetsuo Ono. Myotilt: A target selection method for smartwatches using the tilting operation and electromyography. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '18, New York, NY, USA, 2018. Association for Computing Machinery.

[44] Lee Butts and Andy Cockburn. An evaluation of mobile phone text input methods. In *Proceedings of the Third Australasian Conference on User Interfaces - Volume 7*, AUIC '02, pages 55–59, Darlinghurst, Australia, Australia, 2002. Australian Computer Society, Inc.

[45] I. Scott MacKenzie and R. William Soukoreff. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, pages 754–755, New York, NY, USA, 2003. ACM.

[46] NASA AMES Research Center Human Performance Research Group. Nasa task load index (tlx), 1999.

[47] Katie A. Siek, Yvonne Rogers, and Kay H. Connelly. Fat finger worries: How older and younger users physically interact with pdas. In Maria Francesca Costabile and Fabio Paternò, editors, *Human-Computer Interaction - INTERACT 2005*, pages 267–280, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[48] Xiaojun Bi, Yang Li, and Shumin Zhai. Ffitts law: Modeling finger touch with fitts' law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, page 1363–1372, New York, NY, USA, 2013. Association for Computing Machinery.