Personalized Input: Improving Ten-Finger Touchscreen Typing through Automatic Adaptation

Leah Findlater[1,2] and Jacob O. Wobbrock[2]

[1]College of Information Studies
University of Maryland, College Park, MD
leahkf@umd.edu

[2]The Information School | DUB Group
University of Washington, Seattle, WA
wobbrock@uw.edu

ABSTRACT

Although typing on touchscreens is slower than typing on physical keyboards, touchscreens offer a critical potential advantage: they are software-based, and, as such, the keyboard layout and classification models used to interpret key presses can dynamically adapt to suit each user's typing pattern. To explore this potential, we introduce and evaluate two novel personalized keyboard interfaces, both of which adapt their underlying key-press classification models. The first keyboard also visually adapts the location of keys while the second one always maintains a visually stable rectangular layout. A three-session user evaluation showed that the keyboard with the stable rectangular layout significantly improved typing speed compared to a control condition with no personalization. Although no similar benefit was found for the keyboard that also offered visual adaptation, overall subjective response to both new touchscreen keyboards was positive. As personalized keyboards are still an emerging area of research, we also outline a design space that includes dimensions of adaptation and key-press classification features.

Author Keywords

Touchscreen text input, personalization, adaptive interfaces.

ACM Classification Keywords

H5.2 [Information interfaces and presentation]: User interfaces—*input devices and strategies*.

INTRODUCTION

Large touchscreen devices such as tablets and interactive tabletops are being adopted for an increasingly wide range of tasks. While these devices support ten-finger text input and full size QWERTY keyboards, providing rapid text entry remains a challenge [4,14,27]. The reduced tactile and kinesthetic feedback of a touchscreen not only increases the visual attention required to position hands and fingers on the keyboard but also impacts the user's knowledge of whether a key has been depressed [3]. Spurious input may also occur from hands or arms touching the surface [31], further impacting text entry.
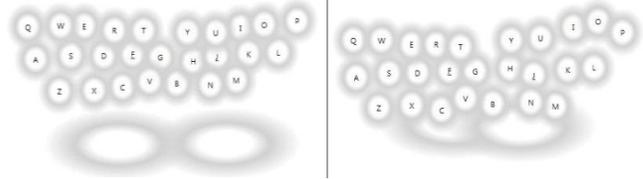
Figure 1. Personalization with a visually adaptive keyboard, showing the adapted layouts for two participants at the end of three study sessions.

Despite these challenges, touchscreens offer a major potential advantage over physical keyboards: they are software-based, and, consequently, the keyboard layout and key-press classification models can dynamically adapt to each user. In previous work, for example, we used simulations of touchscreen typing data to show that a *personalized* input model for ten-finger typing greatly improves key-press classification accuracy over a model aggregated across all users [6]. However, we did not actually build an adaptive keyboard, leaving future work to capitalize on the implications of those findings. Furthermore, other empirical exploration of personalized keyboards for ten-finger typing has been almost nonexistent, with the only user evaluation to our knowledge showing no performance benefit over a static keyboard [7].

To study the effect of keyboard adaptation on touchscreen typing performance, we designed and evaluated two novel adaptive keyboards for ten-finger typing (Figure 1). These prototypes explore two points in the design space of personalized touchscreen keyboards. While both use the same approach to personalize the underlying key-press classification model, one keyboard also *visually adapts* the location of keys while the other maintains a visually stable rectangular layout. We used an iterative approach to design the keyboards, including simulation analyses of ten-finger typing data to select and tune the final underlying key-press classification model, a J48 decision tree classifier trained on finger location and movement features.

We conducted a three-session study with 12 participants to compare performance and subjective reactions with the two adaptive keyboards and a traditional, static QWERTY layout. Our findings show that a personalized keyboard can improve typing speed over the static alternative. The personalized keyboard that did *not* visually adapt improved typing speed by 12.9% across all three sessions, rising to 15.2% for the third session; no discernible difference was found in error rates. In contrast, the personalized keyboard

that *did* visually adapt did not provide a performance benefit over the static keyboard. It was, however, considered to be the most comfortable and natural of the three keyboards.

This paper makes the following contributions. First, as keyboard adaptation is still an emerging area of research, we outline a design space for personalized touchscreen keyboards, including major dimensions of adaptation and key-press classification features. This design space provides both a rationale for the design choices that we made and highlights open areas of future work. Second, we provide two novel personalized keyboard interfaces that use more sophisticated classification models than previous work [6,7] and can accommodate a wide range of typing features. Finally, we provide empirical evidence for the benefit of such personalization, both in terms of typing performance and user experience. These findings show that the way in which keyboard adaptation is visualized can significantly impact the efficacy of and reaction to that adaptation. While our focus is on ten-finger touch-typing, our contributions are also relevant to mobile devices, where research has only begun to explore personalized text input [30].

RELATED WORK
We discuss work most relevant to personalized touchscreen keyboards in the context of our design space (next section). Here, we focus on tabletop text input, touchscreen typing, and adaptive interfaces in general.

The lack of efficient methods for entering text on interactive tabletops is well documented (*e.g.*, [4,14,31]) and may potentially detract from long-term sustained use of such devices [27,37]. As one solution, distal tactile feedback, applied not on the surface itself but on the wrist or arm, has been shown to improve typing speed [26]. The use of physical keyboards [12] and physical silicon keyboards overlaid on the surface [36] have also been proposed, highlighting the challenge of text entry on interactive tabletops. Finally, although not for tabletops *per se*, techniques to support ten-finger typing on arbitrary flat surfaces using computer vision [29] or augmented gloves [8] have been explored. The 1Line keyboard [22] has also been proposed to save visual space on smaller touchscreens.

In contrast to ten-finger typing, a large body of work exists on virtual keyboards for a *single point of input*, most often a finger or stylus. These include techniques that utilize the geometric relationship between keys to improve input accuracy [20] or enable typing without a priori determining the position of the keyboard, albeit with a high error rate [28]. Alternatives to tapping a QWERTY keyboard have also been proposed, including alternate key layouts [21,24,40] and methods to stroke between keys [19]. Though highly relevant to our work, one difference between single point of input and multiple-finger typing is highlighted by comparing two past papers on keyboard size ([25] *vs.* [33]). With stylus input, MacKenzie and Zhang [25] showed that a smaller keyboard increased errors but did not reduce

speed compared to a larger keyboard. In contrast, with multiple finger input, Sears *et al.* [33] showed that smaller keys reduce speed *and* increase errors.

Regardless of the exact keyboard layout or method used to identify key presses, language modeling is commonly used to improve input accuracy with virtual keyboards. For keyboards where a single key is overloaded with multiple letters, these models can clarify the ambiguity of key presses (*e.g.*, [8]). More generally, language modeling can be used to improve typing accuracy by reducing the chance of improbable key presses (*e.g.*, [1,2,9,10]). These approaches combine the user's key-press location with language model predictions for the next possible letter to more accurately classify which key the user was attempting to hit [9]. As discussed in the next section, work on language modeling is complementary to our research.

Our design space and, particularly, the comparison of visual and nonvisual adaptation in that design space are inspired by previous work on adaptive user interfaces. Adaptive user interfaces can introduce new challenges compared to traditional interfaces (see [5,16,17]). Those most applicable to text input include: lack of user control, unpredictability, lack of transparency of the system's decision making process, and obtrusiveness, or the degree to which the adaptations are distracting. Evaluations are also often more complicated than for non-adaptive user interfaces. For example, not only does the user need time to learn about the system, but the system needs to learn about the user.

DESIGN SPACE OF PERSONALIZED KEYBOARDS
Before describing our specific approach, we first outline a design space for touchscreen keyboard personalization. This design space is useful to introduce and define concepts that our classification models leverage as well as to map out this emerging research area. Since the focus of this paper is on adaptive (automatic) forms of personalization, we use the terms "adaptive" and "personalized" interchangeably. We also rely on a distinction made by Goodman *et al.* [9] between *touch models* and *language models* for touchscreen typing. A touch model describes how the user's finger touch is interpreted as a key press while the language model adds knowledge such as the distribution of words in a given language. These two sources of information are often combined to improve overall typing accuracy (*e.g.*, [9,10]). Our focus is primarily on improving touch models, which has received less attention of the two.

Dimensions of Adaptation
We outline three major dimensions of touch-model adaptation for touchscreen keyboards. No one dimension is entirely independent of the others, but each one may elicit different design considerations.

Dimension 1: Key-Press Classification Model
A key-press classification model classifies finger touches as specific keys, such as A, S, or D. Perhaps the simplest adaptive model is the *closest-centroid* approach, where key centroids are updated as the user types, and touches are

assigned to the key with the closest centroid. For ten-finger typing, simulation results have demonstrated the potential of a personalized closest-centroid approach to increase classification accuracy over a model aggregated across all users [6]. However, user evaluations with a ten-finger keyboard [7] and a 9-key numeric keyboard [13] have so far found no performance improvement.

Bivariate Gaussian distributions computed for each key offer a more sophisticated, probabilistic classification approach and have been used for small touchscreen devices [9,10,28] though not, to our knowledge, for ten-finger typing. Most work with bivariate Gaussian distributions has focused on models aggregated across multiple users; however, simulations by Rudchenko *et al.* [30] have demonstrated the benefit of personalized versus aggregate models. Both closest-centroid and bivariate Gaussian models only make use of the (x,y) location of the finger touch. In this paper, we explore machine learning approaches that can accommodate a larger set of features.

Dimension 2: Visual Key Layout
When the key-press classification model adapts to the user, the visual key layout may also change. Figure 2 shows an example from Go and Endo [7]: a Voronoi tessellation drawn about the adapted centroids of a closest-centroid classification model. *Not* adapting the visual layout when the underlying model changes increases the chance that the classifier's output will not match the visual key the user appears to touch. Thus, *visual adaptation* may increase the predictability of the keyboard. How users react to visual adaptation, however, has so far been inconclusive (see [7] *vs.* [13]). This issue may reflect a broader challenge with adaptive user interfaces: adaptations can demand so much attention from the user that even theoretically beneficial adaptations have a negative effect [5,17].

The issue of visual adaptation can arise with language model predictions as well. Some text input techniques increase the size of probable keys based on a language model's prediction of the next letter [1,2,10]. In this
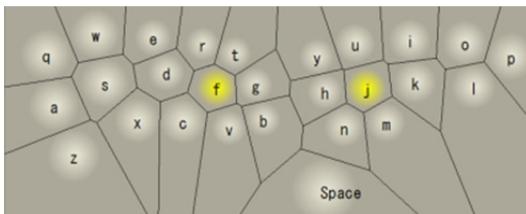

Figure 2. Visual adaptation of CATKey, showing Voronoi tessellation around adapted key centers (from [7]; copyright Springer).
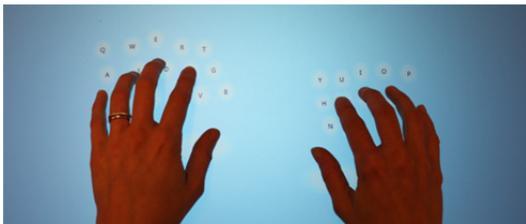
Figure 3. Adapting keyboard halves to hands in our system.

context, Gunawardana *et al.* [10] defined an *anchor* area in the center of each drawn key to improve key-press predictability. This technique, where touch points within an anchor area *always* return the anchored key, may also be useful when combined with adaptive touch models.

Dimension 3: Keyboard Positioning and Orientation
While the previous two dimensions focused on *individual* keys, the keyboard itself can also adapt more holistically to the user's hand and arm placement. This adaptation can occur when the user first places their hands on the surface: for example, the keyboard may be displayed automatically in two halves under the user's hands (see Figure 3). This type of adaptation has been previously proposed [15,32] but not evaluated. The keyboard can also adjust itself more subtly over time to account for drift as the user is typing.

Classification Features for Touchscreen Typing
The above subsections defined three dimensions of adaptation for personalized touchscreen keyboards. Here we focus on *Dimension 1*, key-press classification models, and consider what features may be most useful for classification (Figure 4). Ultimately, the set of possible features will be determined by the input hardware used (*e.g.*, a vision system may provide more features than a capacitive touchscreen). While our focus is on ten-finger typing, many features listed here are also applicable to smaller devices. Our goal is to highlight the breadth of features that exist for personalizing keyboards as well as to introduce the particular features that we later test.
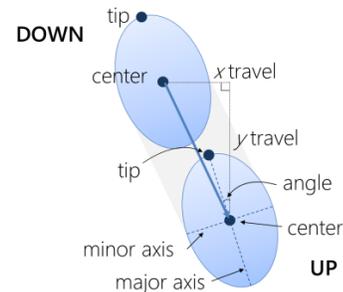

Figure 4. A subset of classification features based on a single key press event. This subset was tested in our simulations.

Touch location features are based on finger-down and finger-up locations. These features can include: (1) *finger center* (x,y) and (2) *fingertip* (x,y) locations. On large touchscreens, the keyboard can move, so these coordinates must be relative to some point(s) of reference. For adaptation to the left and right hands, *two* relative center locations can be calculated, *e.g.*, one as an offset from the F key and one as an offset from the J key (the *home* keys). Absolute values can be used for smaller screens.

Touch area features describe the size and angle of the finger on the screen; these features greatly depend on the hardware device. We used a Microsoft Surface, which provides ellipses fitted to the touch area and allows for (3) *major axis*, (4) *minor axis* and (5) *orientation*.

Movement features describe the movement of the finger

during the key-press action. Different keys may result in different movement signatures based on the finger used and how the user reaches for that key. Specific features include (6) *travel* and (7) *elapsed time* between finger down and up. Travel can be calculated as the distance in x and y directions for either the fingertip or the finger center.

Features relative to previous key presses incorporate knowledge about previous key presses. These include (8) *inter-key-press timing*, or the elapsed time between key presses, and (9) (x,y) *distance offset from previous key*. The latter may be useful if, for example, the current key-press location is affected by whether the last key was to its left or right. A simulation study by Rashid and Smith [28] showed low accuracies for a touch model based only on this feature.

Hand or arm features are based on the position of the hand and/or arm. Again, these features are highly dependent on the underlying touchscreen system in that only certain systems can provide this information.

Summary and Discussion
We have defined three dimensions of adaptation and a list of potential classification features for personalized touch models. Of course, the particular touchscreen device used will dictate which subset of features is possible and useful. In addition, any adaptive system needs to account for how the user model is trained: online, as the user types (*e.g.*, [7,13]), or offline, perhaps in an explicit training period (*e.g.*, [30]). Furthermore, if the training occurs online, the *rate* at which the adaptation occurs may impact the effectiveness of a personalized keyboard.

ITERATIVE DESIGN OF AN ADAPTIVE KEYBOARD
We took an iterative approach to design and build two personalized keyboards that were the same in all respects except for their *visual* adaptation. Here, we describe the general personalization method used, a pilot study of our early keyboard designs, and, finally, the designs that we ultimately evaluated in the full user study.

General Personalization Approach
We implemented static (non-adaptive) and personalized (adaptive) keyboard interfaces in C♯ .NET 4.0. The most basic of these interfaces, used as a control condition, was a static rectangular QWERTY layout with keys 0.9" × 0.9" (39 px × 39 px; based on the Surface's native keyboard).

For the personalized keyboards, we used Weka 3.6[1] and IKVM[2] to provide online adaptation in real-time. That is, each personalized keyboard collected training data as the user typed and, at natural break points in the user's input, this training data was used to update the classification model and possibly the visual layout of the keyboard. In our studies, model updating occurred at the end of each phrase. Key presses whose letters were subsequently backspaced were considered errors and removed from the training set.

Additionally, each key was initially seeded from 5 points sampled from a bivariate Gaussian distribution around the key's center, with x and y variance of 40 pixels and no covariance. The keyboards only began adapting after a minimum of 10 training points (key presses) were collected for every key. Based on early user feedback, we required that *all* keys achieve this minimum for consistency so that some keys did not begin adapting sooner than others (*e.g.*, E collects training data more quickly than Q). To accommodate the possibility that the user could adjust his or her typing patterns over time, the training history only stored the last 100 strikes per key.

The frequency of model-updating and the size of the training history used were determined by informal feedback from six early users and input from the research team. This process allowed us to identify a reasonable rate of adaptation for the personalized keyboards. However, determining the *optimal* rate is an open area of research and only one of many design considerations with adaptive user interfaces (see [5][17]).

Pilot Study to Select a Classifier and Feature Set
We conducted a pilot study that informed the final keyboard designs in two important ways: (1) it inspired us to reflect on the role of visual adaptation in personalized keyboards; (2) it provided realistic typing data with which to evaluate and select an appropriate machine learning classifier for the key-press classification models.

Typing Interfaces for Pilot Study
We built and tested three keyboards in the pilot study, two of which were seeded using a dataset we had previously collected on 10-finger typing patterns from 20 participants [6]. In particular, since we hoped to create novel keyboards that would allow for typing with reduced visual attention, we used data from a condition in the previous study that was meant to emulate just that: in the *no visual keyboard* condition in [6], participants typed on a blank surface.

The keyboards used in the pilot study were as follows:

- *Conventional* was a static rectangular QWERTY layout.
- *Static-Centroid* provided a static layout where the locations of keys were adjusted to the aggregate centroids of key presses across all participants from the previously collected dataset (from [6]).
- *Adaptive-Bayes* began with the same aggregate key layout and switched to a Naïve Bayes classifier after 10 training points had been observed for each key (using finger center (x, y) and fingertip (x, y) as the feature set). As with all machine learning classifiers in this paper, we used Weka 3.6. Naïve Bayes was chosen for this early keyboard implementation because it performed well in simulations we ran on the dataset from [6].

The pilot study system provided language model correction trained on an existing corpus [38]. We used a source-channel approach [9,10]: at each key press, we reduced the chance of outputting improbable letters by combining letter

---

probabilities from the touch model with those from the language model. Additionally, at the end of each word, the language model auto-corrected to the most likely full word in its dictionary if the currently displayed word was not recognized (similar to auto-correction in the Apple iOS).

Pilot Participants, Procedure and Performance Data
Four participants each volunteered for three 1.5-hour long study sessions where they used each of the three keyboards. For each session they entered a combination of pangrams and random phrases from the MacKenzie and Soukoreff phrase set [23]. Participants were asked to type quickly and accurately, and corrected errors with a right-to-left horizontal swipe (as in [6]). They typed 5 practice phrases and 40 test phrases with each keyboard during each session. Mean typing speeds and uncorrected error rates were similar across all keyboard conditions, from 27.7 to 28.2 WPM (comparable to [6]) and 0.7% to 0.9% error rate.

Discussion and Simulation Analysis of Pilot Data
Although we had predicted that the personalized keyboard (*Adaptive-Bayes*) would offer a performance benefit over the control condition, typing speeds across all three conditions were similar. We identified two potential areas for improvement. First, the visual adaptation of the keyboard was unsettling to some participants, suggesting that the role of visual adaptation was more complex than we had expected. Second, we had chosen the Naïve Bayes classifier based on simulations with our previous dataset [6], but our current keyboard interface was substantially different; for example, [6] did not provide a visible keyboard or standard typing output. As such, it seemed prudent to reassess the choice of machine learning classifier based on the newly collected pilot study data.

We ran simulations with the data from the *Conventional* keyboard condition to compare the accuracy of several machine learning classifiers on this new dataset. The simulations tested personalized key-press classification models by running 10-fold cross validations on each participant's data. We tested a number of classifiers provided in Weka, including Naïve Bayes, Support Vector Machines, and Decision Trees. J48, Weka's implementation of the C4.5 decision tree algorithm, produced consistently high classification accuracy (with all features in Table 1: 82.3% on average across all participants). In contrast, the Naïve Bayes classifier that we had used for the *Adaptive-Bayes* condition did not perform as well (58.0% for the same setup). In addition, J48 was fast for building/updating the model and classifying key presses, which is obviously critical for a real-time system such as ours. Based on these findings, we chose the J48 classifier for the full study, with a normalization filter and default classifier settings.

Final Implementation
Based on the insights from the pilot study, we implemented two personalized keyboards that were exactly the same in all respects except that one visually adapted the keyboard (*Visual-Adaptive*) while the other did not (*NonVisual-*

Adaptive). Both keyboards were initially rendered as rectangular QWERTY layouts (see Figure 6). *Visual-Adaptive* adapted the visual center of each key to be located at the centroid of all fingertip locations in that key's training history. To avoid excessive visual overlap between keys, *Visual-Adaptive* also reshuffled keys to maintain a minimum of 30 pixels (0.7") between key centers (Figure 5). For stability, reshuffling was done by maintaining the F and J "home" keys at a fixed location and translating neighboring keys away from F and J; this effect then cascaded toward outer keys.
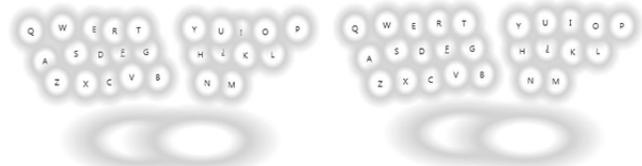
(a) Layout before reshuffling (b) Layout after reshuffling
Figure 5. Removing overlap between neighboring keys.

In contrast, *NonVisual-Adaptive* always retained a visually stable rectangular layout (Figure 6). As discussed earlier, adapting the underlying key-press model without changing the visual layout *increases* the chance that the user will press one visual key but the system will output a different letter. To mitigate this issue, *NonVisual-Adaptive* disabled the J48 classifier for one keystroke after a backspace, or if the user was typing slowly (< 1 key stroke per second). In those situations, the user is more likely to look down at the keys and target carefully, so the system simply checked which visual key boundaries contained the fingertip.

To select which subset of classification features to include in the J48 classification model, we used a wrapper-based feature selection technique [18]. That is, we incrementally added features to the model until no improvement in performance occurred. This process resulted in all features in Table 1 marked with a '*'. Additionally, we included the finger center y relative to the F and J keys for continuity, since the finger center x was already in the feature set; this inclusion did not reduce classification accuracy. With the

Feature Set	Classification Accuracy
Tip x relative to F and J keys*	53.3% (SD = 3.3)
Center x relative to F and J keys*	43.4% (SD = 6.7)
Tip y relative to F and J keys*	35.4% (SD = 2.0)
Center y relative to F and J keys*	29.7% (SD = 1.5)
Angle	28.0% (SD = 3.9)
Tip travel in x and y directions*	27.4% (SD = 1.9)
Center travel in x and y directions	24.8% (SD = 1.4)
Major axis	22.4% (SD = 1.5)
Minor axis	19.0% (SD = 1.9)
Elapsed time	14.2% (SD = 1.0)
Inter-key-press time	11.7% (SD = 2.3)
Majority classifier (right space key)	11.8%

* Used in feature set for final implementation.

Table 1. Explanatory power of classification features based on the pilot study data: mean 10-fold cross-validation results from a J48 classifier trained on individual features or pairs of related features. The majority classifier is the accuracy rate if we always classify as the most frequent key (right space key).

selected set of classification features, the J48 algorithm achieved on average a classification accuracy of 82.6% (*SD* = 3.8) across all participants.

Language model correction was not used in the full study because we observed in the pilot study that the *accuracy* of the language model—in our case perhaps unrealistically accurate—appeared to affect typing patterns. As such, we disabled the language model to control this possible interaction between the language and touch models and to isolate the effects of the touch model personalization.

Finally, to reduce the chance of inadvertent or spurious input, the user had to *activate* the keyboard by briefly placing all 10 fingers on the screen. The keyboards deactivated after 5 seconds of inactivity.

STUDY

We conducted a controlled, three-session study to compare the two personalized keyboards to a conventional static keyboard. We hypothesized that the personalized keyboards would improve typing performance and subjective experience compared to the conventional keyboard. Since previous findings are inconclusive with respect to visual adaptation [7,13], this study also allowed for an exploratory comparison of the two personalized keyboards, one of which adapted visually and one of which did not.

Method

Participants

Twelve participants were recruited through on-campus mailing lists (5 male, 7 female). Participants were volunteers and were compensated for their time. They ranged in age from 21 to 48 (*M* = 33.8). All were regular computer users and had experience with touch devices, with 10 of the 12 indicating they enter text on mobile touch devices often. In terms of physical keyboard use, one participant regularly used a natural or split keyboard, while the remaining participants used standard rectangular layouts. On a *physical* keyboard typing test administered during the study, participants typed 79.2 WPM (*SD* = 16.6) and had an uncorrected error rate of 0.2% (*SD* = 0.2).
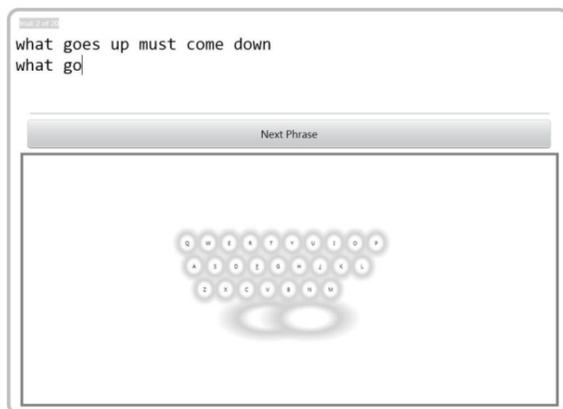

Figure 6. Experimental application showing the Conventional keyboard. Two space bar keys appear because the right and left halves of the space bar (corresponding to each thumb) were modeled separately in all keyboard interfaces.

Apparatus

We used a Microsoft Surface running custom software written in C♯ .NET 4.0. The system recorded all down and up touch events during the typing tasks. Participants sat at the Surface, which was raised ~5" off the ground with a custom-built platform to about the same height as a standard desk. The task interface, shown in Figure 6, presented phrases to be typed at the top of the screen. After typing a phrase, participants advanced by pressing the "Next Phrase" button. At the end of each condition, the system wrote all information related to the personalized key-press classification model and visual key layout to an XML file. This data was reloaded when the participant used the same keyboard condition in future sessions.

Experiment Design

The study was a 3×3 within-subjects factorial design with the following factors and levels:

- **Session:** Participants completed three sessions each.
- **Keyboard:** *Conventional*, *NonVisual-Adaptive*, and *Visual-Adaptive*. The *Conventional* layout was the same as the initial rectangular layouts for the personalized keyboards. Key presses in *Conventional* were interpreted simply by which key's visual bounds contained the fingertip. The personalized keyboards began adapting as soon as they had obtained 10 training instances for each key in Session 1.

Order of presentation for the keyboard conditions was fully counterbalanced within each session and participants were randomly assigned to orders.

Procedure

Participants completed three 1.5-hour sessions spaced at least 4 hours and at most 48 hours apart. Session 1 began with a brief introduction to the Microsoft Surface and experiment software. From the user's perspective, the adaptive keyboards behaved exactly the same as the *Conventional* keyboard until a minimum number of training examples had been recorded for each key; therefore, all participants began by typing 10 practice phrases and 20 test phrases with the *Conventional* keyboard. This data was used to seed the personalized keyboards. Participants then typed 5 practice phrases and 20 test phrases with each of the three keyboards. In Sessions 2 and 3, participants typed 5 practice phrases and 40 test phrases on each keyboard.

The typing task interspersed two pangrams with randomly selected phrases from the MacKenzie and Soukoreff phrase set [23]. To ensure we collected sufficient training data for each key in Session 1, the ratio of regular phrases to pangrams was 1:1. In Sessions 2 and 3, that ratio increased to 9:1. Subjective questionnaires were given after each keyboard condition and at the end of each session. At the end of Session 3, we interviewed participants about the keyboards. Additionally, participants completed a physical keyboard typing test (20 phrases) at the beginning of Session 1 or end of Session 2.

The main quantitative measures were speed and uncorrected error rate. Speed was calculated as WPM [39]:

$$WPM = \frac{|T| - 1}{S} \times 60 \times \frac{1}{5} \qquad (1)$$

where T is the final transcribed string and S is the elapsed time in seconds from the first to the last letter in the phrase. Uncorrected error rate was calculated as [35]:

$$uncorrected\ error\ rate = \frac{INF}{(C + INF + IF)} \qquad (2)$$

where C is the total number of correct characters, IF is the number of incorrect but fixed (backspaced) characters, and INF is the number of incorrect (but not fixed) characters. Uncorrected error rate measures the errors *remaining* in the transcribed input, as opposed to those fixed "along the way," which take time and are thus subsumed in WPM.

For WPM and uncorrected error rate, we analyzed the 20 test phrases with each keyboard in Session 1 and the 40 phrases from each of Sessions 2 and 3. Since presentation order was counterbalanced within each session, we tested for effects of *Order* on WPM within each session. Two-way ANOVAs with *Order* as a between-subjects factor and *Keyboard* as a within-subjects factor revealed no main or interaction effects of *Order* on WPM. Thus, our counterbalancing seemed effective. We used a two-way repeated measures ANOVA for WPM with both *Session* and *Keyboard* as within-subjects factors for the final analysis. For all *post hoc* pairwise comparisons, Bonferroni adjustments were used.

The main subjective measures were NASA TLX workload scales [11] and rankings of the keyboards based on ease of use, efficiency, preference, comfort, how natural the typing felt, and frustration. We report the final subjective measures collected at the end of Session 3. For workload data, we ran one-way repeated measures ANOVAs with *Keyboard* as a single factor. For the ranking data, we used Chi-square tests to evaluate whether the number of times each keyboard was top-ranked in a category differed from chance. When the expected frequency values were too low for Chi-square ($<$ 5), we used a randomization test of goodness-of-fit, which uses a Monte Carlo simulation to calculate the probability of the observed frequency values occurring by chance; it is robust against low expected values [34].

Quantitative Results
Figure 7 shows sample personalized keyboard layouts.

Speed (WPM)
As hypothesized, the keyboards affected typing speed differently; see Figure 8. A significant main effect of *Keyboard* was found on WPM ($F_{2,22}$ = 9.086, p = .001, η^2 = .452). *Post hoc* pairwise comparisons revealed that *NonVisual-Adaptive* was the fastest keyboard. That keyboard offered a 12.9% improvement overall compared to *Conventional* (p = .003), a gap which increased to 15.2% by Session 3. *NonVisual-Adaptive* was also significantly faster than *Visual-Adaptive* (p = .016), which suggests that
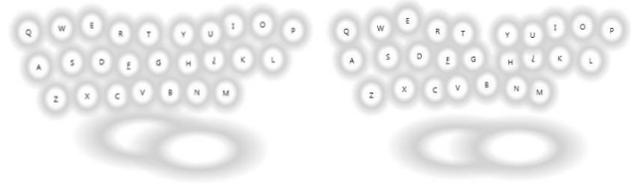
(a) Illustration of *underlying* model for NonVisual-Adaptive

(b) Actual Visual-Adaptive layout, as shown to the participant

Figure 7. Sample adaptive keyboard layouts from Participant 10 at the end of Session 3. To the user, the NonVisual-Adaptive keyboard looked exactly like the Conventional keyboard, so (a) is an illustrative visualization of the *x,y* coordinates stored in the *underlying* model. This general pattern of increased adaptation from (a) to (b) is representative of all participants.
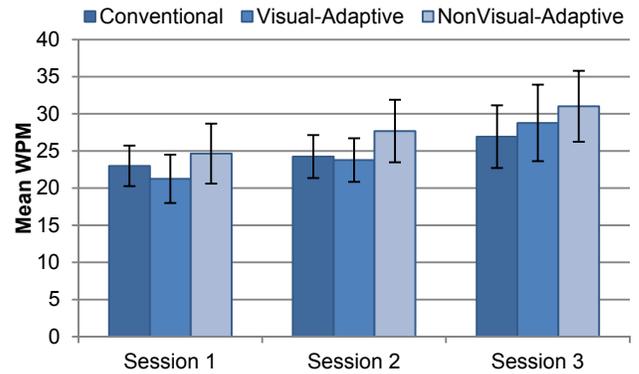

Figure 8. Mean typing speed across sessions . Higher is better. There was a general increase in speed over time, as well as an advantage of NonVisual-Adaptive compared to the other keyboards. Error bars show 95% confidence intervals.

the visual adaptation had a negative effect that counteracted the positive effects of the personalized classification model on its own. No significant difference was found between *Visual-Adaptive* and *Conventional*.

Participants also exhibited a general learning effect as the sessions progressed, shown through a main effect of *Session* on WPM ($F_{2,22}$ = 13.590, p < .001, η^2 = .553). Trends shown in Figure 8 suggest that with additional sessions, typing speeds on all conditions would increase further. No significant interaction effect was found between *Session* and *Keyboard* on WPM ($F_{4,44}$ = 1.450, p = .234, η^2 = .116), indicating all keyboards improved at about the same rate during the study.

Uncorrected Error Rate
All three keyboard conditions resulted in extremely low uncorrected error rates. The mean error rates were: *Conventional* at 0.30% (SD = 0.43%), *NonVisual-Adaptive* at 0.22% (SD = 0.26%), and *Visual-Adaptive* at 0.30% (SD = 0.49%). Because of the low error rates (29% of conditions had no uncorrected errors at all), we did not perform inferential statistics on this data. Note, however, that the means for both adaptive conditions were the same as or lower than for the Conventional keyboard. This data dispels the notion that *NonVisual-Adaptive* improved speed at the expense of increased errors.

Subjective Measures
Participants were asked to rank the three keyboards based on ease of use, efficiency, frustration, comfort, how natural the typing felt, and overall preference. Since some of these questions appeared to be capturing redundant information, we calculated Cohen's κ (a measure of categorical agreement) to group related questions. The three groups shown in Table 2 emerged, suggesting that the six questions had only captured three underlying themes. Cohen κ values between each pair of characteristics *in a group* ranged from 0.86 to 1.00; κ values between pairs *across different groups* ranged from .48 to .74. Thus, we tallied and analyzed the top keyboard votes within each group.

The majority of the time, *NonVisual-Adaptive* was rated highest in the Efficiency/Ease/Preference group, whereas *Visual-Adaptive* received the most votes in the Comfort/Naturalness group. Regardless, *Conventional* ranked poorly in both groups. Chi-square tests showed that the tally frequencies in these two groups differed significantly from chance (Efficiency/Ease/Preference: $\chi^2_{(2, N=36)}$ = 7.170, p = .028; Comfort/Naturalness: $\chi^2_{(2, N=24)}$ = 6.250, p = .044). For Least Frustration, a randomization test of goodness-of-fit using a Monte Carlo simulation with 10,000 random samples was not significant (p = .423).

NASA TLX workload ratings at the end of Session 3 were slightly higher for the *Conventional* keyboard (M = 8.3, SD = 3.6 across all measures) than for *Visual-Adaptive* (M = 7.6, SD = 3.6) and *NonVisual-Adaptive* (M = 7.1, SD = 3.1), but no significant differences were found.

Qualitative Findings
To contextualize the quantitative results, we interviewed participants at the end of the study. One potential issue with adaptive interfaces is the degree to which the adaptation requires the users' attention and distracts them from users' primary task; Jameson [17] calls this *obtrusiveness*. Eight participants remarked *negatively* on the visual adaptation in the *Visual-Adaptive* condition. This keyboard appeared to increase cognitive overhead for at least some participants because of the unusual layout; for example, P8 stated, "I felt I was constantly looking at my hands."

Interestingly, however, negative comments about the visual look of *Visual-Adaptive* keyboard were not necessarily reflective of overall preference or perceived efficiency. Four of those same 8 participants either commented that they could type quickly on *Visual-Adaptive* when *not* looking at the keyboard or ranked that condition as their

Ranking Characteristics	Conven-tional	NonVisual-Adaptive	Visual-Adaptive
Efficiency/Ease/Preference*	6	19	11
Comfort/Naturalness*	3	8	13
Least Frustration	2	6	4

Table 2. Tallies (higher is better) for top-ranked keyboard in each of three subjective groups. An asterisk (*) denotes a Chi-square test is significant at p < .05 on the distribution of tally frequencies.

preferred keyboard at the end of the study. P9 expressed this sentiment as: "[*Visual-Adaptive*]…seemed easiest this time – as long as I didn't look and see the odd spacing."

In contrast, the *NonVisual-Adaptive* keyboard was designed to be much less obtrusive than *Visual-Adaptive*, yet to still provide adaptive benefits. Highlighting this distinction, the majority of participants (7) stated they had trouble defining the difference between *Conventional* and *NonVisual-Adaptive*. For example, P5 said of *NonVisual-Adaptive*, "I felt I just got more into a flow. I didn't notice a big difference between the two. I just felt that I performed better with [*NonVisual-Adaptive*]." Likewise, P7 said she couldn't "tell what the difference is but there's something about [*NonVisual-Adaptive*] that is easier to type with." Since *NonVisual-Adaptive* and *Conventional* looked identical, these comments reflect *NonVisual-Adaptive*'s benefit in terms of the mechanics or "feel" of the typing.

Unprompted, participants also commented on the ability to type without looking frequently at the keyboard. While no one brought this up in the context of *Conventional*, 5 participants commented that either *NonVisual-Adaptive* or *Visual-Adaptive* allowed them to get into more of a "flow" than *Conventional*. P6 also compared the keyboards to the iPad: "With the iPad you see the wrong letters and it corrects in front of you [using language model correction]. But this, I know I'm hitting in the wrong place and the right letter comes up, which is nice. I'm just not used to it."

Finally, a few participants made negative comments about *how* the *Visual-Adaptive* keyboard chose to move the keys, suggesting that there may be room for improvement for this design. For example, P10 commented that although he felt he was most consistent with the *Visual-Adaptive* keyboard, the keys had migrated quite close together, causing him difficulty. Two participants also explicitly commented that *Visual-Adaptive* got easier across the three sessions, which suggests that more training (both for the user and for the system) may further improve that keyboard's usefulness.

Summary
The *NonVisual-Adaptive* keyboard provided a typing speed improvement over *Conventional*, but *Visual-Adaptive* did not. Both personalized keyboards, however, fared well in terms of subjective feedback. *NonVisual-Adaptive* was ranked as most Efficient/Easy-to-use/Preferred, while *Visual-Adaptive* was considered the most Comfortable/Natural. Qualitative feedback suggested participants recognized the performance benefits of *NonVisual-Adaptive* even though they could not necessarily recognize the source of those benefits (because it looked the same as the control condition). Error rates in all three conditions were extremely low.

DISCUSSION
Our results show that a personalized key-press classification model improves ten-finger touchscreen typing performance. Moreover, *how* this personalization is

visualized has a significant impact. We found that the visually adaptive keyboard resulted in lower performance than the non-visually adaptive keyboard, which suggests users incurred a cognitive cost from visual adaptation that counteracted the benefits of personalizing the key-press model. This theory is supported by comments from participants, which confirm previous research by Himberg *et al.* [13] examining visually adaptive 9-key numeric keypads. Despite these issues, however, the visually adaptive keyboard shows promise in that it was considered to be the most comfortable and natural of the keyboards.

While we have focused on improving the underlying touch model for ten-finger keyboards, we envision that these personalized keyboards can be combined with language models. As noted in the implementation section, we disabled language model correction for this study to isolate the impact of the touch model adaptation. However, using a source-channel approach [9,10] to combine probabilities from the personalized key-press classifier with predictions from a language model should further improve typing accuracy. An open question is whether the use of a language model will result in changes to user typing behavior such that the touch models also change—might the two models interact, and if so, how?

In addition to incorporating language models, augmenting the personalized keyboards with audio or tactile feedback is also possible. Indeed, even the distal tactile feedback that has been previously used for tabletop interaction [26] should provide a benefit here.

In creating the two personalized keyboards, we made a number of design decisions that could potentially impact performance and user satisfaction. For example, only the last 100 key presses were stored for each key and the keyboards only began adapting after 10 training points had been recorded for *every* key (our study task included pangrams to expedite this process). Future work should explore how these decisions ultimately impact the effectiveness of the adaptation. One potential area for improvement is to ensure that *over-adaptation* does not occur. Particularly with the visually adaptive keyboards, a cycle of adaptation exists between the user and the system (the system adapts to the user, which, in turn, changes the user's behavior). Identifying points of stability in the input model may reduce the potential negative effects of this adaptation cycle.

With personalized interfaces, evaluations need to be designed not only to account for training time for the user, but also training time for the system. Although we conducted a multi-session study, performance improvements from the second session to the third session indicate that learning had yet to plateau for any condition. This issue may be most pertinent to the visually adaptive condition, since it was the most unfamiliar of the keyboards. Although not statistically significant, typing speed with this keyboard appeared to improve the most

from Session 2 to Session 3 (Figure 8). A longer study would be useful for determining where performance stabilizes for all three keyboards, but particularly for the visually adaptive one.

Although we have focused on ten-finger typing, much of the design space for personalized keyboards and the study findings have implications for other modes of touchscreen typing. Two of the three major dimensions of adaptation— the underlying classification model and the visual layout of keys—should be equally applicable to mobile devices. Additionally, we tested a range of machine learning classifiers in the Weka toolkit that yielded reasonably high key-press classification accuracies, but further exploration likely will yield even better models.

CONCLUSION

We have introduced and evaluated two novel personalized keyboard interfaces that adapt their underlying key-press classification models based on an individual user's typing patterns. In addition, these keyboards explore two points in the design space of adaptive keyboards: one visually adapts while the other always maintains a visually stable rectangular layout. Results from a user evaluation showed non-visual personalization can significantly improve typing speed. However, visualizing adapted key layouts can negatively impact speed. Both adaptive keyboards received a positive subjective response and, as such, point to the further promise of this area of research. We have also presented a design space for personalized keyboards to ground our design decisions and to identify areas for future work. Although our focus has been on ten-finger touch-typing, the results and design space should also be applicable to adaptive text input techniques on small mobile devices. We believe this work is one step in a larger trend towards increased personalized interaction with devices.

REFERENCES
1. Al Faraj, K., Mojahid, M., Vigouroux, N. 2009. BigKey: A Virtual Keyboard for Mobile Devices. *Proc. HCII*, 3-10.
2. Aulagner, G., François, R., Martin, B., Michel, D., Raynal, M. 2010. Floodkey: increasing software keyboard keys by reducing needless ones without occultation. *Proc. ACS'10*, 412-417.
3. Barrett, J. 1994. Performance effects of reduced proprio-ceptive feedback on touch typists and casual users in a typing task. *Behaviour & Information Technology 13(6)*, 373-381.
4. Benko, H., Morris, M. R., Brush, A.J.B., Wilson, A.D. 2009. Insights on Interactive Tabletops: A Survey of

Researchers and Developers. *Microsoft Research Technical Report MSR-TR-2009-22*. March, 2009.

5. Findlater, L., Gajos, K. 2010. Design space and evaluation challenges of adaptive graphical user interfaces. *AI Magazine 30(4)*, 68-73.

6. Findlater, L., Wobbrock, O., Wigdor, D. 2011. Typing on flat glass: examining ten-finger expert typing patterns on touch surfaces. *Proc. CHI 2011*, 2453-2462.

7. Go, K., Endo, Y. 2007. CATKEY: Customizable and adaptable touchscreen keyboard with bubble cursor-like visual feedback. *Proc. IFIP Interact*, 493-496.

8. Goldstein, M., Book, R., Alsiö, G., Tessa, S. 1999. Non-keyboard QWERTY touch typing: a portable input interface for the mobile user. *Proc. CHI'99*, 32-39.

9. Goodman, J., Venolia, G., Steury, K., Parker, C. 2002. Lang-uage modeling for soft keyboards. *Proc. IUI '02*, 194-195.

10. Gunawardana, A., Paek, T., Meek, C. 2010. Usability guided key-target resizing for soft keyboards. *Proc. IUI '10*, 111-118.

11. Hart, S. G., Staveland, L. E. 1988. Development of Nasa Tlx (Task Load Index): Results of Empirical and Theoretical Research. *Human Mental Workload (1988)*, 139-183.

12. Hartmann, B., Morris, M. R., Benko, H., Wilson, A. D. 2009. Augmenting interactive tables with mice & keyboards. *Proc. UIST '09*, 149-152.

13. Himberg, J., Häkkilä, J., Kangas, P., Mäntyjärvi, J. 2003. On-line personalization of a touch screen based keyboard. *Proc. IUI 2003*, 77-84.

14. Hinrichs, U., Hancock, M., Collins, C., Carpendale, S. 2007. Examination of text-entry methods for tabletop displays. *Proc. Tabletop 2007*, 105-112.

15. Hirche, J., Bomark, P., Bauer, M., Solyga, P. 2008. Adaptive interface for text input on large-scale interactive surfaces. *Proc. IEEE International Workshop on Horizontal Interactive Human Computer System (TABLETOP)*, 153-156.

16. Höök, K. 2000. Steps to take before intelligent user interfaces become real. *Journal of Interacting with Computers 12(4)*, 409–426.

17. Jameson, A. 2008. Adaptive interfaces and agents. In A. Sears & J. Jacko (Eds.), The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications (2nd ed.), 433-458. CRC Press.

18. Kohavi, R., John, G.H. 1997. Wrappers for feature subset selection. *Artificial Intelligence 97*, 1-2, 273-324.

19. Kristensson, P. & Zhai, S. 2004. SHARK2: a large vocabulary shorthand writing system for pen-based computers. *Proc. UIST '04*, 43-52.

20. Kristensson, P. & Zhai, S. 2005. Relaxing stylus typing precision by geometric pattern matching. *Proc. IUI '05*, 151-158.

21. Langendorf, D.J.: Textware solution's Fitaly keyboard v1.0 easing the burden of keyboard input. *WinCELair Review*, February 1998.

22. Li, F.C.Y, Guy, R.T., Yatani, K., Truong, K.N. 2011. The 1line keyboard: a QWERTY layout in a single line. *Proc. UIST '11*, 461-470.

23. MacKenzie, I.S., Soukoreff, R. W. 2003. Phrase sets for evalua-ting text entry techniques. *Extended Abstracts CHI '03*, 754-755.

24. MacKenzie, I.S., Zhang, S. X. 1999. The design and evaluation of a high-performance soft keyboard. *Proc. CHI '99*, 25-31.

25. MacKenzie, I.S., Zhang, S.X. 2001. An empirical investigation of the novice experience with soft keyboards. *Behaviour & Information Technology 20*, 411-418.

26. McAdam, C., Brewster, S. 2009. Distal tactile feedback for text entry on tabletop computers. *Proc. BCS-HCI'09*, 504-511.

27. Morris, M.R., Lombardo, J., Wigdor, D. 2010. WeSearch: Supporting Collaborative Search and Sensemaking on a Tabletop Display. *Proc. CSCW 2010*, 401-410.

28. Rashid, D.R. and Smith, N.A. 2008. Relative keyboard input system. *Proc. IUI '08*, 397-400.

29. Roeber, H., Bacus, J., Tomasi, C. 2003. Typing in thin air: the Canesta projection keyboard—a new method of interaction with electronic devices. *Proc. CHI '03*, 712-713.

30. Rudchenko, D., Paek, T., Badger, E. 2011. Text Text Revolution: a game that improves text entry on mobile touchscreen keyboards. *Proc. Pervasive 2011*, 206-213.

31. Ryall, K., Forlines, C., Shen, C., Ringel Morris, M., Everitt, K. 2006. Experiences with and Observations of Direct-Touch Tabletops. *Proc. Tabletop 2006*, 89-96.

32. Sax, C., Lau, H., Lawrence, E. 2011. LiquidKeyboard: an ergonomic, adaptive QWERTY keyboard for touchscreens and surfaces. *Proc. ICDS 2011*, 117-122.

33. Sears, A., Revis, D., Swatski, J., Crittenden, R., Shneiderman, B. 1993. Investigating touchscreen typing: the effect of keyboard size on typing speed. *Behavour & Information Technology 12(1)*, 17-22.

34. Sokal, R. R., Rohlf, F. J. 1994. Biometry: The Principles and Practices of Statistics in Biological Research. 3rd ed. New York, NY: W. H. Freeman.

35. Soukoreff , R.W., MacKenzie, I.S. 2003. Metrics for text entry research: an evaluation of MSD and KSPC, and a new unified error metric. *Proc. CHI '03*, 113-120.

36. Weiss, M., Wagner, J., Jansen, Y., Jennings, R., Khoshabeh, R., Hollan, J. D., Borchers, J. 2009. SLAP widgets: bridging the gap between virtual and physical controls on tabletops. *Proc. CHI '09*, 481-490.

37. Wigdor, D., Penn, G., Ryall, K., Esenther, A., Shen, C. 2007. Living with a Tabletop: Analysis and Observations of Long Term Office Use of a Multi-Touch Table. *Proc. Tabletop 2007*, 60-67.

38. Wobbrock, J.O, Myers, B.A. 2006. From letters to words: efficient stroke-based word completion for trackball text entry. *Proc. Assets '06*, 2-9.

39. Wobbrock, J.O. 2007. Measures of text entry performance. In *Text Entry Systems: Mobility, Accessibility, Universality*, I. S. MacKenzie and K. Tanaka-Ishii (eds.). San Francisco: Morgan Kaufmann, 47-74.

40. Zhai, S., Smith, B.A., Hunter, M. 2002. Performance Optimization of Virtual Keyboards. *Human-Computer Interaction 17(2&3),* 229-269.