WalkType: Using Accelerometer Data to Accommodate Situational Impairments in Mobile Touch Screen Text Entry

Mayank Goel[1], Leah Findlater[2,3] and Jacob O. Wobbrock[3]

[1]Computer Science & Engineering
DUB Group
University of Washington
Seattle, WA 98195 USA
mayank@cs.washington.edu

[2]College of Information Studies
University of Maryland
College Park, MD 20742 USA
leahkf@umd.edu

[3]The Information School
DUB Group
University of Washington
Seattle, WA 98195 USA
wobbrock@uw.edu

ABSTRACT

The lack of tactile feedback on touch screens makes typing difficult, a challenge exacerbated when situational impairments like walking vibration and divided attention arise in mobile settings. We introduce *WalkType*, an adaptive text entry system that leverages the mobile device's built-in tri-axis accelerometer to compensate for extraneous movement while walking. WalkType's classification model uses the displacement and acceleration of the device, and inference about the user's footsteps. Additionally, WalkType models finger-touch location and finger distance traveled on the screen, features that increase overall accuracy regardless of movement. The final model was built on typing data collected from 16 participants. In a study comparing WalkType to a control condition, WalkType reduced uncorrected errors by 45.2% and increased typing speed by 12.9% for walking participants.

Author Keywords: Touch screen; text entry; adaptive; situational impairments; mobile; virtual keyboard; walking.

ACM Classification Keywords: H.5.2. Information interfaces and presentation: User Interfaces—*input devices and strategies*.

General Terms: Human factors, design, experimentation.

INTRODUCTION

Touch screen devices have become the dominant platform for mobile computing; however, the lack of tactile feedback on these devices requires a high level of visual attention to select targets accurately. Input is particularly challenging when the user is in motion [25,26,29], a state that can be thought of as causing *situational impairments* [35]. Situational impairments may be caused by a variety of factors including vibration, divided attention, diverted gaze, body motion, awkward postures, cold temperatures, clothing, rainwater, glare, uneven terrain, ambient noise, or encumbering baggage. The challenges of situational impairments are exacerbated for mobile text entry on virtual keyboards because of the many repeated targeting actions that take place in quick succession. Researchers have explored various techniques to accommodate some situational impairments, like walking versus stationary interaction [29,36] and, to a lesser extent, adaptive techniques to *automatically* meet such needs [21,42]. Despite these advances, techniques to improve interaction in the presence of situational impairments are relatively unexplored, particularly for text entry.

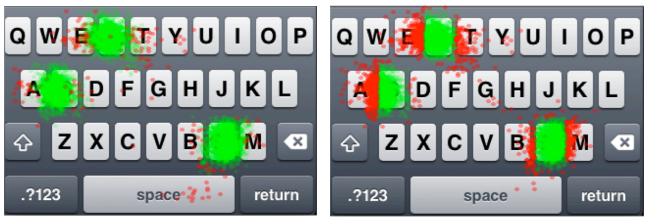

Figure 1. Visualizaton of key presses with WalkType (*left*) and without WalkType (*right*) collected while users were walking during model-building, showing touch points for "N", "R" and "S". Correct key-press classifications are in green; errors are in red. In our study, WalkType (*left*) corrected 90.1% of the errors.

In this paper, we present *WalkType* (Figure 1), a system that uses a touch screen device's built-in accelerometer to increase text entry accuracy while the user is walking. Taking inspiration from image stabilization techniques of cameras to remove motion blur (*e.g.*, [5]), WalkType compensates for imprecise input by incorporating multiple features computed from the accelerometer data: displacement, acceleration, and inference about the user's movement. Additionally, WalkType uses tap location and the finger travel distance during taps to improve the user's text entry accuracy, features that increase overall accuracy regardless of movement. Previous work on adaptive text entry has largely focused on adjusting key-press probabilities based on language models [7,13,14]. To our knowledge, taking into account the user's motion through accelerometer or other sensor data has not been explored.

To design and build WalkType, we first collected touch screen typing data from 16 participants. Based on this data, we built two key-press classification models that incorporate accelerometer information. In simulations, these models improved classification accuracy over a control condition from 72.8% to 94.6% and 95.7%,

CHI 2012, May 5-10, 2012, Austin, TX, USA.

respectively. While some improvement occurred for both walking and sitting, walking received an additional 4.8% increase in accuracy. The final WalkType prototype is a composite of these two models and a third more basic model that together achieved an accuracy of 97.3%.

We evaluated the final WalkType system in a controlled study with 16 participants, 5 of whom had also participated in our model-building study. Our findings show that WalkType improves typing performance for both sitting and walking, but that the benefits are greatest for walking. WalkType improved text entry speed compared to a control condition from 28.3 to 31.1 words per minute (WPM). Uncorrected error rate [37] also improved, particularly for walking, where average error rate dropped from 10.5% to 5.8% with WalkType. Finally, WalkType was highly preferred by participants, who recognized its performance benefits despite there being no visual difference between WalkType and the control interface.

The main contribution of this paper is a demonstration that accelerometer data *can* be used to improve typing performance on mobile devices when the user is situationally-impaired due to walking. This contribution comes in three parts: (1) an exploration of key-press classification models for improving mobile touch screen typing; (2) WalkType itself and its component models that incorporate accelerometer data; and (3) an evaluation of WalkType, showing that it significantly improves user typing performance over a standard touch screen keyboard, particularly while the user is walking.

RELATED WORK

We draw motivation from prior research exploring solutions to make interaction with devices easier while walking and while experiencing situational impairments. In addition, we draw on work on screen content stabilization.

Situational Impairments and Walking User Interfaces

The importance of accounting for the context in which a mobile device is used has been stressed by numerous researchers (*e.g.,* [18]). Walking has been shown to affect both input and output with mobile devices. Mizobuchi *et al.* [29] evaluated how increasing size of target buttons could improve text entry performance while walking. Lin *et al.* [25] studied the effect of walking on stylus tapping, and found that performance decreased while walking. Yesilada *et al.* [44] demonstrated that the number of errors made by an unimpaired user on a mobile device was similar to a motor-impaired desktop user. Consequently, existing techniques for motor-impaired users may be useful in accommodating situational impairments on mobile devices. Kane *et al.* [20] proposed an auto-correction system to help motor-impaired typists. Mobile devices can also impact the user's ability to read information. For example, walking has been shown to have a negative effect on both text legibility [30] and reading comprehension [1].

To address the foregoing challenges, techniques have been proposed to bridge the gap between stationary and walking interaction. For example, Brewster *et al.* [4] used audio feedback to improve touch screen interaction while standing and walking. Bragdon *et al.* [3] evaluated touch screen gestures in mobile environments, and established that gestures starting on the screen border as a reference point are not "significantly affected by the environment." Taking an adaptive approach, Kane *et al.* [21] coined the term *walking user interfaces (WUIs)* and evaluated a method to automatically enlarge soft buttons when users are walking versus stationary. Yamabe and Takahashi [42] used accelerometer information to automatically adapt the size of fonts and images while walking. Yatani and Truong [43] investigated how two-handed chorded keyboard input could improving using a stylus-based PDA while walking.

Touch Screen Text Entry and Adaptive Keyboards

A plethora of touch screen text entry techniques have been developed for both finger and stylus input. Prior work [28] provides a full review. Approaches to improve text entry performance with QWERTY keyboard layouts have been proposed, for example, using geometric pattern matching [23] and gestures [22]. Of particular relevance to our work are approaches that combine *language model* predictions with probabilities from a *touch model* (*e.g.*, [13,14]) to improve overall input accuracy. Language model predictions for the next letter to be typed have been used to resize keys, either visibly [7] or invisibly, that is, without showing the adaptation to the user [14]. (Apple's iPhone uses this approach.) Gunawardana *et al.* [14] ran a simulation study demonstrating the usefulness of key *anchoring* when the language model predictions are invisible. That is, regardless of the changing predicted letter probabilities, a center (anchor) area on the visible key always returns that letter, ensuring that a "direct hit" on the key by a user's finger provides a predictable result. We use this approach in WalkType.

As with WalkType, others have proposed techniques that use models of key-press distributions built on aggregate typing data collected from users. Most commonly, bivariate Gaussian distributions have been used to model individual keys [13,14,33]. A small number of projects have introduced models that adapt to individual typing patterns. A simulation study by Rudchenko *et al.* [33] showed that a *personalized* key-press model using bivariate Gaussian distributions for each key improved performance over an aggregate model. Adapting the location of keys based on the centroid of the user's previous key presses has also been studied for larger devices [12,16], but no performance benefits have been found in user evaluations.

Accelerometer-Based Input

Researchers have leveraged accelerometers for tasks ranging from scrolling and changing screen orientation [17], to mapping the speed of the cursor on a mobile device to the inferred degree of tilt [38]. Novel text entry

techniques using an accelerometer as the primary source of input have also been proposed [19,31,34,39]. In general, these techniques have relatively high error rates and are meant for specialized contexts where touch screen keyboards may not be available. TiltType [31] and TiltText [39], for example, use combinations of tilting and button-pressing for entering letters. Our system, in contrast, uses accelerometer data as an additional source of information to improve typing on a standard QWERTY layout.

Screen Content Stabilization

WalkType derives motivation from the image stabilization techniques broadly found in digital cameras. There has been extensive research in countering user motion in image and video capture. Researchers have worked to remove effects of camera shake for images [8,24]. Similar techniques have also been used by researchers to stabilize contents of a screen. Behringer [2] addressed the problem of shaking displays in moving vehicles by dynamically shifting screen content. Similarly, *NoShake* [32] utilizes the accelerometer of a smartphone to perform content stabilization by dynamically moving screen content.

Finally, of particular relevance to walking-based situational impairments, Crossan *et al.* [6] leveraged an accelerometer to analyze in which phase of a user's gait he or she more comfortable interacting with a stylus-based PDA, and which areas of the screen are more error-prone. Although their approach is relevant to our work, Crossan *et al*. studied simpler target selection tasks than for keyboarding.

THE DESIGN OF *WALKTYPE*

WalkType uses multiple sources of information to classify a user's finger-touches as key-presses. Among these sources is data from the device's built-in accelerometer, which is used to account for extraneous movement while the user is walking. In this section, we outline WalkType and the process taken to build it, including a study to collect training data from 16 participants. In the next section, we describe a controlled study of the final WalkType system.

Model-Building

Along with accelerometer data, WalkType uses tap locations and tap travel distance to better predict the intended key. The Weka machine learning toolkit[1] was used to generate two J4.8 Decision Tree models with pruning confidence set to Weka's default (0.25). For classification, the first model used time-domain accelerometer data between taps and the second model used the pattern of accelerometer data generated from the three axes due to the phone's motion while walking. The final WalkType system combined output from both of these models along with a simple Euclidian model. Our analysis showed that this composite model performed better than individual models.

For clarity, we use the term *Euclidian model* throughout this paper to refer to a simple key-press classification model that takes as input the (x, y) coordinate of a finger-touch and returns the letter whose corresponding key's visual bounds contain those coordinates.

The models were built based on typing data collected from 16 participants (10 males, 6 females) who each volunteered for a 45-minute study session. All participants self-rated as expert computer users and intermediate to expert touch screen smartphone users. They were between 21 and 35 years of age (M = 28.69, SD = 4.48).

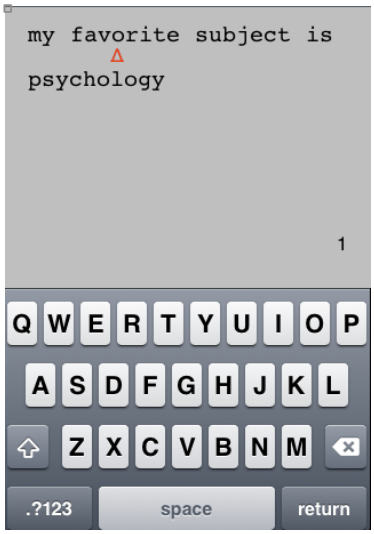

Figure 2. In *WalkType Collect*, the user was only given feedback about whether they pressed a key or not. The red triangular cursor moved forward after every key-press.

We built a custom data collection application, *WalkType Collect*, for the iPhone 3GS that records the device's movement using the on-device low-noise tri-axis accelerometer. We wanted to elicit natural typing patterns and did not want participants to be overly concerned with the accuracy of their input. Thus, we followed the approaches of Gunawardana *et al.* [14] and Findlater *et al.* [9] and created Collect's keyboard in such a way that it only gave the user feedback *that a tap had occurred*, but not where it occurred or what key had been hit. To convey this feedback, a small cursor moved under the phrase as the user typed. Figure 2 shows an example. If the user realized that they were off by a character or two while typing, they could swipe from right to left anywhere on the screen to delete one tap at a time. Participants were instructed to try to delete their tap when they knew they had made an obvious mistake or when they felt they were off by a character or two. We requested participants not to go back through the whole phrase in order to correct a supposed error. Participants were asked to enter 50 phrases in 2 postures, sitting and walking, while holding the device with both hands and typing with both thumbs. The order of postures was counterbalanced and participants were randomly assigned to orders. Short phrases of English text from MacKenzie and Soukoreff's phrase set [27] were used. Apart from these, every fifth phrase was a randomly

[1] http://www.cs.waikato.ac.nz/ml/weka

selected pangram from a list of 35 pangrams to ensure sufficient data for all letters of the alphabet.

The lack of tap-location feedback meant that users made mistakes while entering text, which added noise to our data. Thus, outliers were removed during post-processing by eliminating all taps that landed outside the Euclidean bounds of the intended key or its immediate neighbors. Figure 3 shows filtered data for one participant for the "H" key. About 2.5% taps were filtered out in this process.

Figure 3. Tap filtering for letter "H". Taps not on the intended key or its immediate neighbors are filtered out (in red).

The logs from Collect contained tap-start and tap-end locations, amount of travel while tapping, the time interval between taps, the intended key, and temporal accelerometer data. The touch-screen-based features (*tap location, tap travel* and *time elapsed* between taps) form the *base set* of classification features used in both models described in the next two subsections. We chose to include tap travel and time elapsed in this set based on observations we made while developing Collect. For tap travel, we observed that, at times, the tap-start and tap-end locations were not the same, yielding a potential feature to increase classification accuracy. For time elapsed, we observed that typing speed appeared to impact the user's input accuracy: the tendency to type the wrong key was relatively low when typing slowly compared to more quickly.

Displacement and Acceleration Model

We hypothesized that one of the major reasons for inaccuracy in typing while walking is the general movement of the phone and its displacement from a relatively stable location with respect to the user. Based on this hypothesis, the Displacement and Acceleration Model improves tap accuracy by incorporating acceleration features in all three axes, and magnitude and direction of displacement in the z-axis. To calculate these features, the data from the smartphone's on-device accelerometer was first passed through a low-pass filter to remove noise. This model also includes the base set of features.

To calculate the acceleration features, the filtered accelerometer data was resampled to 10 samples between two consecutive taps. We selected this sampling rate as it gave reasonable resolution and did not overly increase the number of attributes for the classifier. These 10 samples of (x, y, z) values constitute 30 features for the model. When dealing with accelerometer data, it is often necessary to compensate for gravitational pull on the three axes. We found this compensation unnecessary because phone orientation stays relatively constant while typing.

For the displacement magnitude and direction features in the z-axis, we first subtracted the mean acceleration from the filtered data and then double-integrated it using the cumulative sum. The direction in which the phone moved in the z-axis was also calculated. To do so, we compared the device's instantaneous acceleration with the moving mean acceleration of the device. If the instantaneous acceleration was less than the mean, we inferred that the device was moving forward. Otherwise, it was moving backward.

We conducted a 10-fold cross-validation on the WalkType Collect data to evaluate the Displacement and Acceleration Model. The model improved classification accuracy on average from 72.8% (for the Euclidian model) to 94.6%. This is a significant increase in overall accuracy (paired-samples t-test: $t_{15} = 22.23$, $p < .001$). To evaluate the benefit of the accelerometer data, we also tested this model after removing all accelerometer features. Classification accuracy dropped to 90.8% on average, which was a significant decrease (paired-samples t-test: $t_{15} = 12.95$, $p < .001$). See Figure 4 for a comparison of all models, including the Displacement and Acceleration Model.

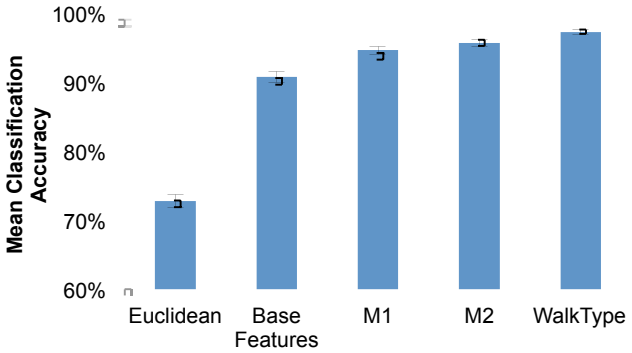

Figure 4. Classification accuracy with different models. M1 is the Displacement and Acceleration Model and M2 is the Walking Pattern Model. The difference in performance of Base Features and WalkType illustrates the benefit of the accelerometer data. Error bars show standard error.

Walking Pattern Model

In analyzing the Collect data, we observed that the phone oscillated in a largely repeatable pattern while a user was walking and typing. Figure 5 shows one such instance of the pattern in the z-axis. The Walktype Pattern Model leverages the on-device accelerometer to obtain this pattern in all three axes. In addition to the base set of classification features, it incorporates four new features per axis. Crossan *et al.* [6] observed a similar pattern and analyzed how it could be used to detect phases where the user is more comfortable performing target selection with a stylus. We use different techniques as a user's interaction with a device while typing is very different while walking.

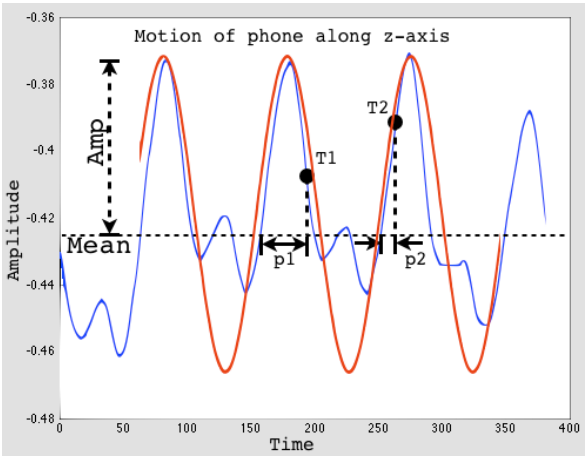

Figure 5. Motion of the phone along *z*-axis when the user is walking. T1 and T2 are two instances of taps; p1 and p2 are the elapsed times since the signal crossed the mean. The waveform in red is part of the dominant frequency wave.

To make the model adaptive to different walking speeds, we calculated the dominant frequency (*e.g.*, red wave in Figure 5) of the user's motion and its mean amplitude from all three axes. This gives us a proxy for detecting changes in the user's speed and intensity of movement. To calculate the instantaneous dominant frequency, we took the Fast Fourier Transform (FFT) of the accelerometer signal and found the frequency with the maximum amplitude. This *frequency* and *amplitude* constitute the first two features. For the third feature, the *direction* of the last mean crossing before the current tap gives a measure of the direction in which the device is moving. Finally, to pinpoint where in the pattern a tap event occurs, we use the *elapsed time* since the accelerometer signal crossed the mean value of the signal, as demonstrated in Figure 5.

These features in the *x*-axis are particularly useful in detecting the user's footstep pattern. We observed that when users' feet hit the ground, their taps tended to shift slightly towards the center of the keyboard. We also observed that a shift to the left was more common when the left foot hit the ground, and a shift to the right was more common when the right foot hit (Figure 6). When we do the analysis shown in Figure 5 on the *x*-axis data, we can detect which foot strikes the ground. If the current *x*-axis data is less than the mean, then the user's left foot has landed, and vice-versa for the right foot. Because the effect of the foot-strike on the user's accuracy would attenuate over time, we also calculated the time since the last foot hit the ground. This calculation was performed in exactly the same way as for the *z*-axis (Figure 5).

We provide the classifier with the tap location on the screen, the direction in which the phone is going in *y*- and *z*-axes, and the last foot that struck the ground. We also provide three temporal components denoting time since the last change in direction in the three axes. On 10-fold cross-validation with the Collect data, the Walking Pattern Model outperformed the Displacement and Acceleration Model with a mean classification accuracy of 95.7% compared to 94.6%, a significant difference ($t_{15} = 5.11$, $p < .001$).

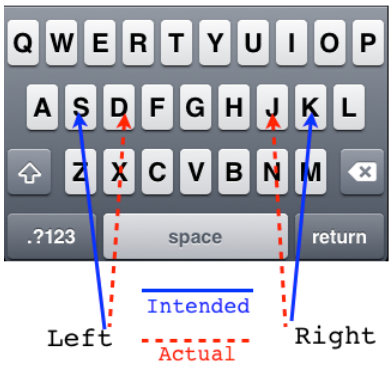

Figure 6. The Walking Pattern Model takes into account error in typing when the user's left or right foot just strikes the ground.

Combined WalkType Model

The Combined WalkType Model is a composite of the three sub-models: the Displacement and Acceleration Model, the Walking Pattern Model, and the Euclidean Model. A majority voting approach is used, whereby for each finger-touch, the key selected by at least two of the three internal models is output to the text stream. When all three models disagree, the Walking Pattern Model prevails, since it model performed the best in isolation on the WalkType Collect data. Figure 7 shows a block diagram detailing the approach. The Euclidian Model is included because, although classification accuracy was high for both of the accelerometer-based models, some keys become major sources of errors as they got masked by adjacent keys. An example confusion matrix is shown in Figure 8. Here a more frequently occurring key dominates adjacent keys, for example, "A" dominates "S". To counter this problem we combined the two models along with the Euclidean model. As mentioned earlier, the Euclidean model selects the key containing tap location. Although simple and non-adaptive, this model increases the probability of less-frequently occurring keys like "W" being correctly classified.

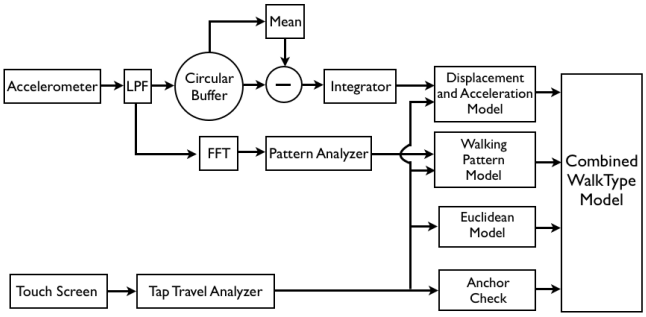

Figure 7. Block diagram of major components of WalkType's Model Building phase.

The mean classification accuracy of the Combined WalkType Model is 97%, significantly higher than both the Displacement and Acceleration Model and the Walking Pattern Model (paired two-tailed *t*-tests, respectively: $t_{15} = 6.51$, $p < .001$; $t_{15} = 5.94$, $p < .001$).

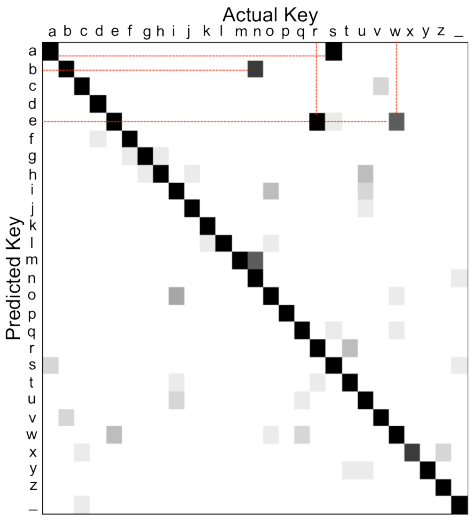

Figure 8. Frequently occurring keys dominate adjacent keys. 's' is dominated by 'a', 'r' & 'w' are dominated by 'e'.

The Combined Walktype model also incorporates key-target anchoring [14]. After incorporating the three models, there were still taps that, although clearly landing in a key's center, did not get classified correctly. Gunawardana *et al.* [14] addressed this issue in the context of adjusting key-press probabilities based on a language model. Their work demonstrated that anchoring some part of the key increases overall typing accuracy. We thus define an anchor area in the middle of each visual key; a tap within that anchor area bypasses the classification models and instead returns the visual key. We reserved the central 20% along the *x*-axis and 50% along the *y*-axis of each key as the anchor area. Introducing anchors further increased the overall accuracy of the WalkType Combined model to 97.28% (significant compared to without anchors: $t_{15} = 5.193$, $p < .001$).

This final model was used in *WalkType*. Referring back to Figure 4 shows the cumulative improvement in accuracy for each component within this final model. In a verification step, we confirmed that the accelerometer features are, indeed, a critical component of WalkType. Removing these features drops classification accuracy by 6.8% ($SD = 2.6$) on average across participants.

WalkType Online

All simulations in the previous section were run offline with Weka. The final step in the process of creating WalkType was to port the model to an iPhone 3G for online use and evaluation. To do so, we first ported the classification models generated by Weka into the iPhone and then optimized and synchronized their operations to work in real-time. The online version of WalkType first filters the accelerometer data, and then sends it to the Combined WalkType Model for classification.

EVALUATION

While the model-building phase demonstrated the potential of WalkType to improve key-press classification accuracy, the goal of our controlled evaluation was to see whether our simulation results would transfer to a real typing task. In particular, we sought to see whether WalkType would lessen the text entry performance degradation incurred by walking compared to sitting.

Participants

Sixteen participants (8 male, 8 female) ranging in age from 21 to 40 years ($M = 29.7$, $SD = 5.7$) were recruited. Five of these participants also participated in the model-building study. All participants had more than 10 years of experience with computers and self-rated as intermediate to expert computer users. Fourteen participants were near-expert touch screen smartphone users with approximately 2-3 years' worth of use. Two participants did not own touch screen smartphones and had little experience with them.

Apparatus

Participants used our custom experiment software on an Apple iPhone 3GS that has a 3.5-inch capacitive screen with 480 × 380 pixels. The application was developed using Objective-C. It recorded all of the users' screen interactions as well as movement data from the accelerometer.

Figure 9 shows a screenshot. The presented phrase appears atop the screen, wrapping onto a second line if necessary. Entered text, in dark blue for contrast, appears immediately below. When the user types a key, keystroke feedback is displayed at the top of the keyboard exactly above the position where the user tapped. Because WalkType uses finger-travel and elapsed time as features, the decision of which key was pressed occurs when the user lifts his or her finger. Accordingly, we modified the key-press feedback to occur when the finger is lifted, as opposed to when the user touches the screen, as occurs for the built-in iPhone keyboard. In the bottom-right corner of the text area, the current trial number is displayed, with a trial being equal to one phrase. Finally, the backspace functionality for the keyboard was also modified: a swipe anywhere on the screen from right-to-left backspaced one character.

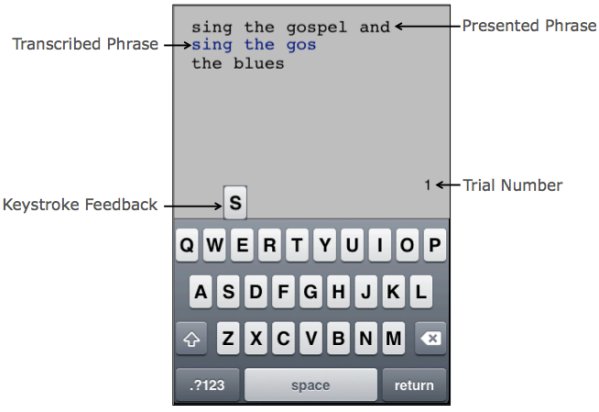

Figure 9. Testing interface showing the presented phrase, transcribed phrase, trial number and keystroke feedback.

Procedure

The procedure was designed to fit in a single 45-minute session. Each session began with an introduction to the

tasks and experiment software. Participants were asked to familiarize themselves with the application and to ask any clarifying questions. This learning phase lasted approximately 5 minutes, until the user was comfortable with the system and had completed at least 5 phrases without assistance. For the walking tasks, we followed an approach similar to Kane *et al.* [21], wherein participants followed a human pacesetter, remaining within 3-5 feet. To simulate a routine and unconstrained environment, the pacesetter ensured that the walking speed was consistent at about 1.07 m/s (3.51 feet/s), a pace that comfortably accommodates a wide range of age and abilities [10]. The walking tasks were performed in a relatively quiet corridor of a university building.

For each condition, participants completed 30 test phrases, 24 of which were randomly selected from the MacKenzie and Soukoreff phrase set [27]. Apart from these, every fifth phrase was a randomly selected pangram to cover all letters in the alphabet. Participants were asked to hold the phone in both hands and type with their thumbs. We asked participants to type quickly and accurately, and to fix errors unless those errors were noticed "far behind" their current point of entry. Finally, participants were requested to complete each trial without failing to keep walking and were offered a rest period at the end of each trial.

Design & Analysis

The study was a within-subjects 2×2 factorial design. The factors and levels were:

- ***Interface*****:** *WalkType* and *Control*. Both interfaces used the same experiment software but the underlying key-press classification models were different. WalkType used the final combined model from the previous section, while Control was non-adaptive, using only the Euclidian model.
- ***Posture*****:** *Walking* and *Sitting*.

Presentation of the interfaces was counterbalanced. Within each interface, postures were also counterbalanced. With 30 trials (test phrases) in each condition, participants performed 2×2×30=120 trials each, for a total of 1920 trials in the study. Overall, we collected 57,663 key presses from 16 participants.

The main measures were speed, calculated as words per minute (WPM), and uncorrected error rate, following Soukoreff and MacKenzie [37]. Uncorrected errors represent those errors left in the transcribed text. Corrected errors, which are errors made during entry, are of less interest, as such errors slow WPM and are thus subsumed by it. Also, to evaluate whether participants could perceive any difference between the two systems, at the end of the session we asked them to rate which one of the two systems they preferred and why. To guard against any bias, participants were not initially made aware of which keyboard was Control and which was WalkType.

To calculate WPM and uncorrected error rate, we used *StreamAnalyzer* [41]. Statistical analyses were done using SPSS 19. We tested for effects of presentation order on the main measure of typing speed using a 3-way ANOVA with *presentation order* of the interfaces as a between-subjects factor and *Interface* and *Posture* as within-subjects factors. No main effect of presentation order was found, indicating that overall counterbalancing was effective. However, there was an asymmetric skill transfer: a significant interaction occurred between presentation order and *Interface* on typing speed ($F_{1,14}$ = 5.569, p = .032, η^2 = .288). The control condition benefited more when it followed WalkType than vice versa. Since this asymmetric skill transfer only injures WalkType's performance relative to the control condition, we are confident that our comparisons are still trustworthy. We present results from a 2-way repeated measures ANOVA with *Interface* and *Posture* as within-subjects factors.

For uncorrected error rate, we used the nonparametric Aligned Rank Transform [15,40] with *Interface* and *Posture* as within-subjects factors. We used this nonparametric procedure because uncorrected error rate is highly skewed toward zero and violates normality. All pairwise comparisons were protected against Type I error using a Bonferroni adjustment.

RESULTS

Speed (WPM)

Speed results are shown in Figure 10. Overall, WalkType improved typing speed regardless of whether the user was sitting or walking: on average 31.1 WPM (*SD* = 10.7) compared to 28.3 WPM (*SD* = 9.6) in the control condition. This difference was significant, as seen in a main effect of *Interface* on typing speed ($F_{1,15}$ = 6.777, p = .020, η^2 = .311). No significant main effect of *Posture* was found.

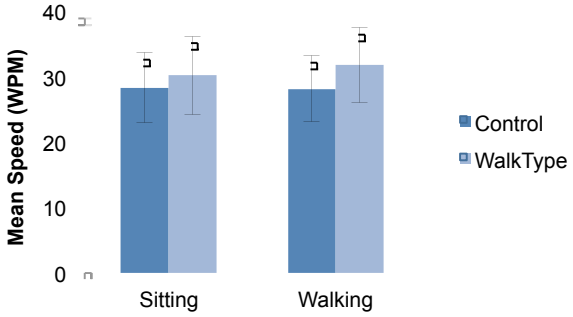

Figure 10. WalkType resulted in higher typing speeds than the control condition, particularly while participants were walking. Error bars are 95% confidence intervals.

We had expected WalkType to improve performance more for walking than for sitting, which would be seen through an interaction of *Interface* × *Posture* on typing speed. This interaction was only a trend ($F_{1,15}$ = 3.485, p = .082, η^2 = .189). Based on our hypotheses, however, we conducted pairwise comparisons of the two interfaces within each

level of *Posture*. (See Games [11] for justification of pairwise comparisons on trend-level effects.)

Pairwise comparisons showed that participants benefited more using WalkType while walking than while sitting. For walking, WalkType improved typing speed by 12.9% compared to the control interface, which was a significant difference (p = .002). For sitting, in contrast, no significant difference was found between the two interfaces (p = .166).

Uncorrected Error Rate

Figure 11 shows mean error rates per condition. Mirroring the speed results, participants exhibited a marked improvement in error rate while using WalkType. A main effect was found for *Interface* on error rate, indicating that, overall, WalkType significantly reduced errors compared to the control condition ($F_{1,15}$ = 22.339, p < .001, η^2 = .598). *Posture* also significantly affected error rate, with walking resulting in increased errors compared to sitting ($F_{1,15}$ = 9.316, p = .008, η^2 = .383).

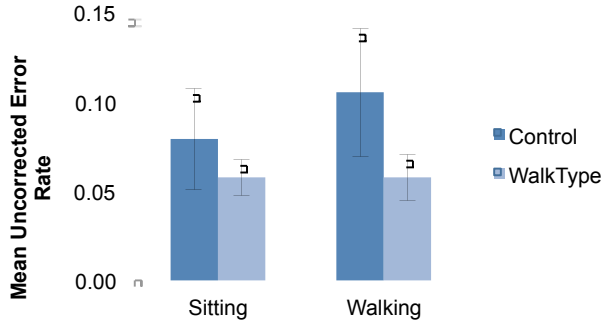

Figure 11. WalkType resulted in lower error rates than Control, especially for walking. Error bars are 95% confidence intervals.

Of perhaps more interest is how WalkType impacted errors compared to the control condition while walking. A significant *Interface* × *Posture* interaction for error rate showed that the effectiveness of the interfaces differed by posture ($F_{1,15}$ = 13.139, p = .002, η^2 = .472). Pairwise comparisons revealed that WalkType was particularly effective at accommodating the situational impairments introduced by movement. Compared to the control condition, error rates decreased from 10.5% to 5.8% with WalkType, which was a significant difference (p = .004). No significant difference in error rate was found between the two interfaces for sitting.

Preference

At the end of the study we asked participants which of the two interfaces they preferred. Preferences reflected performance results, with 14 out of 16 participants choosing WalkType over Control ($\chi^2_{(1,N=16)}$ = 7.56, p = .006).

DISCUSSION

Our goal was to develop an adaptive soft keyboard that leverages accelerometer data to compensate for the situational impairments introduced while walking. WalkType successfully improved both typing speed and error rates, particularly for users walking. On average, WalkType improved typing speed by 12.9% and reduced uncorrected error rates by 45.2% while participants were walking. Although there were no visual differences between the control interface and WalkType, participants perceived the performance benefit of WalkType and overwhelmingly preferred it to the control condition.

The Combined WalkType Model uses a majority voting approach between the three models. We also tried to combine these models into one single model with one decision tree, but the performance of the system decreased considerably because the number of features increased and became unwieldy.

The WalkType models were built using data collected both while participants were walking and while they were sitting. While experimenting with different models, we observed that using only the training data from walking further increased classification accuracy for walking, but *decreased* accuracy for sitting. Based on this finding, we combined the two datasets for the final WalkType model. However, detecting whether a user is walking or sitting and dynamically switching between different models—one trained on walking data and one trained on sitting data—would likely provide a further performance benefit.

Our analysis showed that the major source of incorrect classification was confusion between adjacent keys in the same row. We created a confusion matrix on the basis of our classification results and found that 72.8% of incorrect classifications were of adjacent keys. Further examination into whether misclassifications occurred more towards the left or right sides of the keyboard showed that misclassifications occurred about evenly in this regard. The split of same-row errors was 48.4% to the right of the intended key and 51.6% to the left. This predominance of same-row confusion shows that it was easier for users to reliably hit in the vertical direction compared to the horizontal direction. That users could do this was somewhat unexpected. While walking, most vibrations are along the *y*-axis, *i.e.*, the phone moves backward and forward more relative to the user. We anticipated, incorrectly, that this would lead to more inter-row misclassifications. (Thankfully, WalkType is data-driven, and thus it did not suffer for our misconception.)

Most researchers examining effects of walking on user interfaces do not use a pacesetter as we did, but either ask participants to "walk normally" or employ a treadmill [1,30,32]. Kane *et al.* [21] also used the pacesetter approach to make sure that participants walked at near constant speeds, while enabling them to be "off the treadmill" and in a natural context. Some of the features we have incorporated, such as the dominant frequency of the user's walking pattern and its amplitude, may be useful in adapting to different walking speeds. Additional testing is needed to determine how well WalkType performs with varying walking speeds and whether additional training data is needed to model that context.

During the model-building phase, we collected typing data in a condition where users were given feedback only as to whether they had hit a key, but not whether they hit the *correct* key. In comparison, in the final user study, participants saw which letters they had entered, which could have led them to more accurate typing behavior. This leads to an interesting conjecture—that training the system on more realistic data, like we collected in our evaluation, could further improve performance.

Finally, our study concentrates on the scenario where users hold the phone in both hands and type with their thumbs. It will also be interesting to explore how our models perform on data collected with users holding the phone in one hand and typing with the index finger of the other hand.

FUTURE WORK

The text entry improvements obtained by leveraging the accelerometer suggest a similar approach may be useful for mobile interaction more broadly. Stabilizing input with techniques similar to those proposed in our work, and making the device aware of users' movement patterns and gait may, for example, be useful in accounting for vibrations when users are trying to press a button or move a slider control.

Additional sensors may also be useful in further improving WalkType's classification model. We used only data from the accelerometer to infer device position and rotation. The obvious extension is to evaluate whether the built-in gyroscope would be better suited for inferring rotation.

Personalization is also a promising area of future work for WalkType. Our explorations of the WalkType training data indicate that personalized models, where the system is trained only on one user's data, may increase classification accuracy over models generated from data from all participants. A prior simulation study [33] on mobile touch screen typing data also supports this conjecture.

From our model-building data we observed that users were regularly off by a character when they tried to tap the keys on the borders of the screen, like "A", "Q", SPACE, *etc*. At times, participants' taps landed on the bezel of the device instead of on its keys. Our signal processing filtered out the vibrations caused by tapping of the screen while walking. We believe if we separate vibrations from walking and tapping, then there is potential to leverage the tapping vibrations to detect taps on the bezel and change the system's behavior accordingly, *e.g.*, by entering the most likely intended letter.

CONCLUSION

Using mobile devices in a variety of environments can lead to situational impairment due to, among other sources, vibration and divided attention. We have introduced WalkType, an adaptive system for mobile touch screen devices that leverages the on-device accelerometer to compensate for vibrations and extraneous movements caused by walking. We performed two studies with 16 participants each, first to collect the data for WalkType's model, and then to evaluate the generated models. WalkType increases users' typing speeds from 28.3 WPM to 31.3 WPM, and also reduces the number of uncorrected errors from 10.5% to 5.8% while participants are walking. While WalkType focused on touch screen text entry, its approach may be useful for mobile input in general.

ACKNOWLEDGMENTS

We thank Gaetano Borriello and Jon Froehlich. This work was supported in part by the National Science Foundation (NSF) under grant IIS-0811063. Any opinions, findings, conclusions or recommendations expressed in this work are those of the authors and do not necessarily reflect those of the National Science Foundation.

REFERENCES

1. Barnard, L., Yi, J.S., Jacko, J.A. and Sears, A. (2007). Capturing the effects of context on human performance in mobile computing systems. *Personal Ubiquitous Computing, 11* (2), 81-96.
2. Behringer, R. (2001). Stabilization of a Display in a Moving Environment. *Proc. Annual Federal Laboratory Symposium on Advanced Displays and Interactive Displays*, 131-134.
3. Bragdon, E., Nelson, E., Li, Y. and Hinckley, K. (2011). Experimental analysis of touch-screen gesture designs in mobile environments. *Proc. CHI '11*. New York: ACM Press, 403-412.
4. Brewster, S. (2002). Overcoming the lack of screen space on mobile computers. *Personal and Ubiquitous Computing, 6* (3), 188-205.
5. Canon Inc. What is Optical Image Stabilizer? Retrieved January 9, 2012 from http://www.canon.com/bctv/faq/optis.html
6. Crossan, A., Murray-Smith, R., Brewster, S., Kelly, J. and Musizza, B. (2005). Gait phase effects in mobile interaction. *Extended Abstracts CHI'05*. New York: ACM Press, 1312-1315.
7. Faraj, K.A., Mojahid, M., and Vigouroux, N. (2009). BigKey: A Virtual Keyboard for Mobile Devices. *Proc. HCI'09 Part III*. Berlin: Springer-Verlag, 3-10.
8. Fergus, R., Singh, B., Hertzmann, A., Roweis, S.T. and Freeman, W.T. (2006). Removing camera shake from a single photograph. *Proc. TOG'06*. New York: ACM Press, 787-794.
9. Findlater, L., Wobbrock, J.O. and Wigdor, D. (2011). Typing on Flat Glass: Examining Ten-finger Expert Typing Patterns on Touch Surfaces. *Proc. CHI'11*. New York: ACM Press, 2453-2462.
10. Fitzpatrick, K., M. A. Brewer, and S. M. Turner. (1982). Another Look at Pedestrian Walking Speed. *Journal of the Transportation Research Board, No. 1982*, 21-29.
11. Games, P.A. (1971). Multiple Comparisons of Means. *American Educational Research Journal*, 531–565.
12. Go, K., and Endo, Y. (2007). CATKey: customizable and adaptable touchscreen keyboard with bubble cursor-like visual feedback. *Proc. INTERACT'07*. Berlin: Springer-Verlag, 493-496.

13. Goodman, J., Venolia, G., Steury, J. and Parket, C. (2002). Language modeling for soft keyboards. *Proc. IUI'02.* New York: ACM Press, 194-195.
14. Gunawardana, A., Paek, T. and Meek, C. (2010). Usability Guided Key-Target Resizing for Soft Keyboards. *Proc. IUI'10.* New York: ACM Press*,* 111-118.
15. Higgins, J.J. and Tashtoush, S. (1994). An aligned rank transform test for interaction. *Nonlinear World 1* (2), 201-211
16. Himberg, J., Häkkilä, P., Kangas, P. and Mäntyjärvi, J. (2003). On-line personalization of a touch screen based keyboard. *Proc. IUI'03.* New York: ACM Press*,* 77-84.
17. Hinckley, K., Pierce, J., Sinclair, M. and Horvitz, E. (2000). Sensing techniques for mobile interaction. *Proc. UIST '00.* New York: ACM Press, 91-100.
18. Johnson, P. (1998). Usability and mobility: Interactions on the move. *Proc. Workshop on HCI with Mobile Devices.* Glasgow, Scotland, GIST Technical Report G98-1.
19. Jones, E., Alexander, J., Andreou,A., Irani, P. and Subramanian, S. (2010). GesText: accelerometer-based gestural text-entry systems. *Proc. CHI '10.* New York: ACM Press*,* 2173-2182.
20. Kane, S.K., Wobbrock, J.O., Harniss, M. and Johnson, K.L. (2008). TrueKeys: Identifying and correcting typing errors for people with motor impairments. *Proc. IUI '08.* New York: ACM Press, 349-352.
21. Kane, S.K., Wobbrock, J.O. and Smith, I.E. (2008). Getting Off the Treadmill: Evaluating Walking User Interfaces for Mobile Devices in Public Spaces. *Proc. MobileHCI'08.* New York: ACM Press*,* 109-118.
22. Kristensson, P. and Zhai, S. (2004). Shark: A large vocabulary shorthand writing system for pen-based computers. *Proc. UIST '04*. New York: ACM Press*,* 43-52.
23. Kristensson, P. and Zhai, S. (2005). Relaxing stylus typing precision by geometric pattern matching. *Proc. IUI '05.* New York: ACM Press*,* 151-158.
24. Kundur, D. and Hatzinakos, D. (1996). Blind image deconvolution. *IEEE Signal Processing Magazine 13* (6), 43-64.
25. Lin, M., Goldman, R., Price, K.J., Sears, A. and Jacko, J. (2007). How do people tap when walking? An empirical investigation of nomadic data entry. *International Journal of Human-Computer Studies 65* (9), 759-769.
26. MacKay, B., Dearman, D., Inkpen, K. and Watters, C. (2005). Walk 'n Scroll: A Comparison of Software-based Navigation Techniques for Different Levels of Mobility. *Proc. MobileHCI'05.* New York: ACM Press*,* 183-190.
27. MacKenzie, I.S. and Soukoreff, R.W. (2003). Phrase sets for evaluating text entry techniques. *Extended Abstracts CHI'03.* New York: ACM Press*,* 754-755.
28. MacKenzie, I.S., Soukoreff, R.W. (2002). Text Entry for Mobile Computing: Models and Methods, Theory and Practice. *Human-Computer Interaction 17* (2-3), 147-198.
29. Mizobuchi, S., Chignell, M. and Newton, D. (2005). Mobile Text Entry: Relationship between Walking Speed and Text Input Task Difficulty. *Proc. MobileHCI'05.* New York: ACM Press, 122-128.
30. Mustonen, T., Olkkonen, M. and Hakkinen, J. (2004). Examining Mobile Phone Text Legibility while Walking. *Proc. CHI'04.* New York: ACM Press, 1243-1246.
31. Partridge, K., Chatterjee, S., Sazawal, V., Borriello, G. and Want, R. (2002). TiltType: accelerometer-supported text entry for very small devices. *Proc. UIST'02.* New York: ACM Press, 201-204.
32. Rahmati, A., Shepard, C. and Zhong, L. (2009). Noshake: Content stabilization for shaking screens of mobile devices. *Proc. PerCom'2009.* Washington DC: IEEE, 1-6.
33. Rudchenko, D., Paek, T. and Badger, E. (2011). Text Text Revolution: A Game That Improves Text Entry on Mobile Touchscreen Keyboards. *Proc. Pervasive'11.* Berlin: Springer-Verlag*,* 206-213.
34. Sazawal, V., Want, R., and Borriello, G. (2002). The Unigesture Approach. *Proc. MobileHCI '02*. New York: ACM Press, 256-270.
35. Sears, A., Lin, M., Jacko, J. and Xiao, Y. (2003). When computers fade: Pervasive computing and situationally-induced impairments and disabilities. *Proc. HCI'03.* Marwah: Lawrence Erlbaum Associates, 1298-1302.
36. Shildbach, B. and Rukzio, E. (2010). Investigating selection and reading performance on a mobile phone while walking. *Proc. MobileHCI'10.* New York: ACM Press*,* 93-102.
37. Soukoreff, R.W. and MacKenzie, I.S. (2003). Metrics for text entry research: An evaluation of MSD and KSPC, and a new unified error metric. *Proc. CHI'03.* New York: ACM Press, 113-120.
38. Weberg, L., Brange, T. and Hansson, Å. W. (2001). A piece of butter on the PDA display. *Extended Abstracts CHI'01.* New York: ACM Press, 435-436.
39. Wigdor, D. and Balakrishnan, R. (2003). TiltText: using tilt for text input to mobile phones. *Proc. UIST'03*. New York: ACM Press, 81-90.
40. Wobbrock, J.O., Findlater, L., Gergle, D., and Higgins, J.J. (2011). The Aligned Rank Transform for nonparametric factorial analyses using only ANOVA procedures. *Proc. CHI'11.* New York: ACM Press, 143-146.
41. Wobbrock, J.O. and Myers, B.A. (2006). Analyzing the input stream for character-level errors in unconstrained text entry evaluations. *ACM Transactions on Computer-Human Interaction 13* (4)*,* 458-489.
42. Yamabe, T. and Takahashi, K. (2007). Experiments in Mobile User Interface Adaptation for Walking Users. *Proc. IPC'07.* Los Alamitos, California: IEEE Computer Society, 280-284.
43. Yatani, K. and Truong, K.N. (2009). An Evaluation of Stylus-based Text Entry Methods on Handheld Devices Studied in Different Mobility States. *Pervasive and Mobile Computing 5* (5)*,* 496-506.
44. Yesilada, Y., Harper, S., Chen, T. and Trewin, S. (2010). Small-device users situationally impaired by input. *Computers in Human Behavior* 26 (3), 427-435.