A Model for Detecting and Locating Behaviour Changes in Mobile Touch Targeting Sequences

Daniel Buschek
LMU Munich
Amalienstr. 17, 80333 Munich, Germany
daniel.buschek@ifi.lmu.de

ABSTRACT

Touch offset models capture users' targeting behaviour patterns across the screen. We present and evaluate the first extension of these models to explicitly address *behaviour changes*. We focus on user changes in particular: Given only a series of touch/target locations (x, y), our model detects 1) if the user has changed therein, and if so, 2) at which touch. We evaluate our model on smartphone targeting and typing data from the lab ($N = 28$) and field ($N = 30$). The results show that our model can exploit touch targeting sequences to reveal user changes. Our model outperforms existing non-sequence touch offset models and does not require training data. We discuss the model's limitations and ideas for further improvement. We conclude with recommendations for its integration into future touch biometric systems.

Author Keywords

Change point model; touch biometrics; regression; computational interaction

INTRODUCTION

A well-known aspect of mobile touch input is its inaccuracy: Users may not hit their target exactly, for example due to occlusion [4, 27], limited reach [5], finger orientation [17] and alignment [18], movement [15], and encumbrance [22].

Touch offset models [6, 16, 29] are a computational tool to address this problem: They use touch data to learn users' 2D targeting error patterns across the screen. The models can then shift future touches, improving touch accuracy. These models have also been used to recognise users based on their targeting patterns [10] and to infer hand posture [11] and finger [6].

However, none of these models considered touch *sequences*: 1) Touch corrections were applied per single touch [16, 29]. 2) Recognition of users and hand postures aggregated evidence over multiple touches [6, 10, 11]. Hence, it remains unexplored how to utilise structures within offset sequences, for example *changes* in the underlying behaviour patterns.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
IUI'18, March 7–11, 2018, Tokyo, Japan
© 2018 Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-4945-1/18/03...$15.00
https://doi.org/10.1145/3172944.3172952

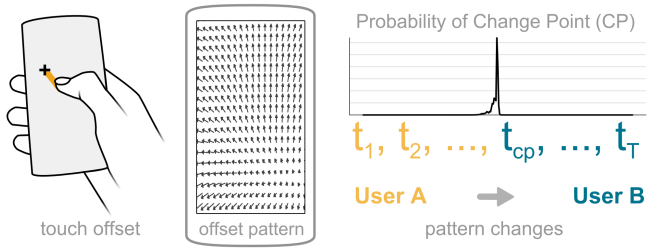

Figure 1. Left: Touch targeting errors (offsets) are measured between touch and target location. Centre: Offset models use regression on such touch targeting data to learn users' individual targeting behaviour patterns across the screen. Right: We extend this approach to touch sequences with behaviour changes, in particular changing users. In this example of a sequence of T touches $t_1, t_2, \ldots, t_T$, the user changes after *cp* touches. Our model exploits the resulting change in offset patterns (i.e. regression parameters) to 1) detect if a touch sequence contains a user change, and to 2) locate at which touch that change occurs.

Touch targeting is often sequential behaviour, for example when typing or navigating between or within apps. Hence, we argue that there is a gap between the actually sequential user behaviour and the existing offset models' non-sequential approach. This motivates our investigation of the first touch offset model that treats touches in sequences (Figure 1). Our model is thus able to address a new use case for utilising touch offsets: Exploiting behavioural (in)consistencies in targeting sequences to detect and locate changes in user behaviour over time. In particular, here we address the case of changing users.

Detecting user changes has great practical relevance, for example for informing interface adaptation and personalisation, and behavioural biometric systems that protect users' personal data and devices. Touch targeting is the most fundamental and common interaction on mobile devices today; yet it typically yields rather limited data (2D touch/target locations). Thus, detecting user changes purely based on targeting behaviour presents an important and challenging problem.

We contribute 1) a *novel touch offset model*, the first to extend existing offset models from single touches to touch sequences with change points; and 2) *detailed evaluations* of this model on touch targeting data in the lab ($N = 28$) and typing data in the wild ($N = 30$).

The results show that our model can exploit touch targeting sequences to reveal user changes. We discuss this as a building block for touch biometric systems, considering limitations and integration opportunities.

RELATED WORK

In mobile touch input, touch-to-target distances ("offsets") depend on the GUI target's location on the screen. Touch offset models aim to capture and utilise these patterns. Here, we relate our work to this line of research.

Henze et al. [16] collected touch targeting data with a smartphone game. They fitted fifth-order polynomial functions to correct offsets, reducing error rate by 7.79 %. These functions predicted *x*/*y* offsets based on touch *x*/*y*, respectively. Later, a user-specific "2D" offset model by Buschek et al. [11] used quadratic polynomials and considered both touch x and y to predict offsets for both dimensions. Our approach embeds such polynomials. In contrast to the related work, our model explicitly considers that the polynomial's parameters might change over time; this is the foundation for detecting changes.

Beyond linear models, Weir et al. [29] used Gaussian Process (GP) regression [23] for offset modelling to improve touch accuracy. Other work also included motion data into such models [20, 21]. Weir et al. [29] found that offset patterns are highly user-specific, and Musić and Murray-Smith [19] pointed out their speed-specific nature when tapping while walking. Thus, later work employed these GP offset models to personalise mobile messaging in context [9], and to identify and authenticate users [10]. This research motivates the use of touch offsets for touch biometrics in general, and in particular our investigation of an offset model that detects and locates user changes. The related work [10] also provides a baseline approach for comparison (see our evaluation).

All existing offset models needed recorded touch and target locations as training data. In particular, related work used 60 to 400 touches [6, 10, 11, 29]. Weir et al. [28] showed that this can be reduced, using a sparse model, namely a Relevance Vector Machine (RVM) [26]. Since our model looks for changes, not known behaviour, it does not require training data beyond the analysed sequence. Nevertheless, this related work motivates our investigation of detecting user changes in sequences of different lengths, showing that longer sequences (i.e. more touches) better reveal changes.

In summary, existing touch offset models employ regression to learn a mapping from sensed to intended location, using training data as a set, not a sequence. In contrast, our model considers touches as sequences of consistent targeting patterns (modelled with linear regression) with potential pattern changes (i.e. changing regression parameters). This novel approach to touch offset modelling enables our model to utilise touch targeting sequences to detect and locate behaviour changes.

APPLICATION CONTEXT

We highlight that our contribution in this paper is not a practical application, but a novel model and its detailed evaluation with a discussion of limitations and opportunities. Nevertheless, as a motivation for this line of research, here we provide examples of possible future applications for our model.

Automated guest mode: A smartphone could use our model as part of a system that detects user changes to automatically enter a "guest mode" for temporary device sharing. That mode could enable phone calls but block viewing private photos.

Detecting misuse: Consider that a user leaves behind an unlocked phone in a bar. A stranger might grab it to browse private data. The device could analyse the sequence of recent touches to detect if the user has changed. It could then lock itself or raise alarm, for example by sending an email with its GPS location to the legitimate owner or a trusted third party.

Personalisation on shared devices: Another use case is a shared device, for example a "family tablet". Different users might use it for online shoppping, without explicitly switching user accounts. It is thus not clear which user has seen which products and should be shown which recommendations in the future. To solve this problem, the device could detect user changes based on touch behaviour to automatically open a prompt asking for the new current user. It might also recognise the new user automatically upon detecting a change when combined with a user identification model. Moreover, detecting user changes in touch sequences also helps to assign parts of the browsing history to different users post hoc.

Note that we consider our model a *building block* among others – many practical applications benefit from combining it with other data and methods, as described in our discussion.

CHANGE POINT MODELLING FOR TOUCH SEQUENCES

Here, we first introduce the (linear) touch offset models that we then integrate into our change point model.

Overview: Touch Targeting Offset Models

We describe a touch $\mathbf{t}$ at screen location x,y as a vector $\mathbf{t} = (x,y)^T \in \mathbb{R}^2$. Abstractly, an offset model is a function $f(\mathbf{t})$ that maps a touch $\mathbf{t}$ to an offset $\mathbf{o} = (o_x, o_y)$. For training, this offset is typically measured between the touch location $\mathbf{t}$ and the target location (e.g. the target button's centre). In applications, adding the model's predicted offset to the touch yields a corrected touch location $\mathbf{t'} = (t_x + o_x, t_y + o_y)^T$. If the model works well, this correction shifts the touch towards the user's intended target, improving touch accuracy [6, 11, 29]. The model developed in this paper detects behaviour changes based on these offset patterns, instead of correcting touches.

Linear Touch Offset Models with Basis Functions

The linear touch offset models described in related work [6, 11] follow the general linear model, in matrix form defined as:

$$\mathbf{o} = \mathbf{X}\mathbf{w} + \varepsilon, \tag{1}$$

where $\mathbf{X}$ is the $T \times d$ design matrix with one row per touch (e.g. in an interaction sequence of T touches) and one column per basis function (explained below). Moreover, $\mathbf{w}$ contains the model's parameters, and $\mathbf{o}$ is a $T \times 2$ vector containing the observed offsets for both screen dimensions for all T touches. Finally, ε is Gaussian noise. See the related work for more details on parameters and how to estimate them from data [6].

The model uses basis functions Φ to transform the raw input touches. For example, the quadratic model described in related work [6, 11] uses $\Phi(\mathbf{t}) = (1, x, y, x^2, y^2)^T \in \mathbb{R}^5$. In general, basis functions allow the linear model to account for non-linear relationships between touch locations and offsets.

Extending Offset Models to Sequences & Change Points

We adapt the change point model from Rasmussen [24], which uses Bayesian analysis and the general linear model. We decided to build upon this model, since it allows us to integrate the linear offset model described above with few adaptations. The result is the first touch offset model that analyses touches as sequences to detect behaviour changes.

Rationale

Intuitively, the model as described below "tries out" all possible splits of a touch targeting sequence into two parts, expecting each part to show different but consistent user behaviour. A change point then seems more likely for those splits of the observed data which match this expectation.

Formal Definition

We describe the parts relevant to our changes and use of the model. More background can be found in related work [24].

First, to be able to treat our two-dimensional output $\mathbf{o} \in \mathbb{R}^{T\times 2}$ with one-dimensional regression, we unroll it into a long vector $\hat{\mathbf{o}} = (o_{x1}, ..., i_{xT}, o_{y1}, ...o_{yT})^T \in \mathbb{R}^{2T\times 1}$ (as in [29]).

We define a matrix $\mathbf{G}_t \in \mathbb{R}^{T\times 2d}$ per timestep t (the "splits"):

$$\mathbf{G}_t = \begin{bmatrix} \mathbf{X}[1,...,t;1,..,d] & \mathbf{0}_{t,d} \\ \mathbf{0}_{T-t,d} & \mathbf{X}[t+1,...,T;1,..,d] \end{bmatrix} \quad (2)$$

Therein, $\mathbf{0}_{n,m}$ is a zero-filled matrix of dimension $n \times m$, and the $\mathbf{X}[...;...]$ denote submatrices of $\mathbf{X}$ which consist of rows and columns with the given lists of indices.

Next, we stack up these matrices $\mathbf{G}_t$ to $\hat{\mathbf{G}}_t \in \mathbb{R}^{2T\times 4d}$ to match the way we handle the 2D output with $\hat{\mathbf{o}}$:

$$\hat{\mathbf{G}}_t = \begin{bmatrix} \mathbf{G}_t & \mathbf{0}_{T,2d} \\ \mathbf{0}_{T,2d} & \mathbf{G}_t \end{bmatrix} \quad (3)$$

Using these matrices $\hat{\mathbf{G}}_t$, we define ρ_t as an intermediate step:

$$\rho_t = |\hat{\mathbf{G}}_t^T \hat{\mathbf{G}}_t|^{-1/2} [\hat{\mathbf{o}}^T \hat{\mathbf{o}} - \hat{\mathbf{o}}^T \hat{\mathbf{G}}_t (\hat{\mathbf{G}}_t^T \hat{\mathbf{G}}_t)^{-1} \hat{\mathbf{G}}_t^T \hat{\mathbf{o}}]^{-(T-d)/2} \quad (4)$$

Finally, normalisation of these ρ_t yields the probability $p(c_t|\hat{\mathbf{o}})$ of a change point c_t happening at time t:

$$p(c_t|\hat{\mathbf{o}}) = \rho_t \left(\sum_{s=1}^{T-1} \rho_s \right)^{-1} \quad (5)$$

Computing $p(c_t|\hat{\mathbf{o}})$ for all $t = 1, ..., T-1$ yields the posterior. For each timestep, we now have a probability describing how likely we think it is that the change happens there. The prior implied in the transformations leading to the equations above is uniform [24]: Before seeing the data, the model considers each time step to be equally (un)likely as a change point.

Model Limitations

The model in this specific form has two limitations: First, it only expects a single change point in a sequence. We focus on this case in our analyses to evaluate the fundamental model. However, as described in related work [24], the formal approach can be extended to multiple change points, in our case by modifying Equation 2 (e.g. to split into three parts). Even in its current form, the *detection* task (i.e. change in sequence or not?) works with multiple change points, as long as at least one of them is a clear enough change to our model.

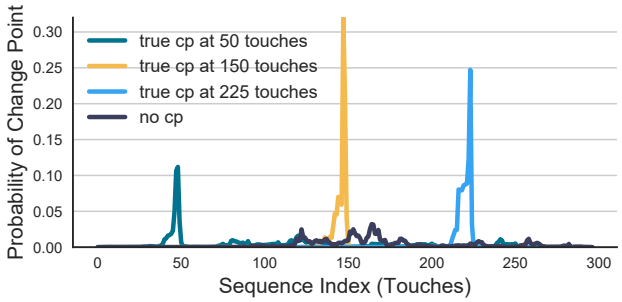

Figure 2. Posterior distributions for four example sequences, three with change points (at 50, 150, 225 touches), and one without a true change point. The plots show the probability of a change point at each touch in the sequence, as estimated by our model. Note how the probability mass concentrates around true change points and how the model remains "undecided" for the sequence with no true change point.

Second, the model formulation originally analyses sequences post hoc. For example, a touch biometric system could use it to detect user changes in touch data after a usage session. However, this approach can be turned into an online version by processing chunks of touches (e.g. the last N touches, updated at each touch). We return to these limitations in our discussion.

TOUCH SEQUENCE AND CHANGE POINT ANALYSES

Model Application I: Detecting Change Points

Intuitively, we decide that a change point is present in a touch sequence if our model's posterior $p(c_t|\hat{\mathbf{o}})$ shows a clear "peak" (i.e. low posterior entropy). In contrast, if the model cannot find a clear change point the posterior will be "flat" (i.e. high entropy). Figure 2 shows examples for each of these cases.

Implementing this approach, we decide if a change point is present or not based on the posterior entropy H:

$$H = -\sum_{t=1}^{T-1} p(c_t|\hat{\mathbf{o}}) \log p(c_t|\hat{\mathbf{o}}) \quad (6)$$

In practice, the actual decision can be made, for example, by comparing the entropy against a threshold.

Model Application II: Locating Change Points

A second task is to estimate *where* a change point is located within a sequence. Our model's posterior distribution also enables such estimates. For example, we can estimate the time step of the change point with the posterior mean location l_{mean}:

$$l_{\text{mean}} = \sum_{t=1}^{T-1} p(c_t|\hat{\mathbf{o}}) t, \quad (7)$$

We could also use the maximum's location $l_{\max}$:

$$l_{\max} = \arg\max_t p(c_t|\hat{\mathbf{o}}) \quad (8)$$

We compare both approaches in our analyses. Our results show that l_{mean} yields more accurate estimates.

STUDY I - MOBILE TOUCH TARGETING IN THE LAB

We evaluate the described model on a published smartphone touch targeting dataset [6], collected in the lab. Per participant ($N = 28$), the dataset contains two sessions (a week apart), each with touches at 400 target locations per combination of the two independent variables: *hand posture* (right thumb, device in right hand; right index finger, device in left hand) and *target type* – crosshair, key (4×7 mm), app icon (9×9 mm), and full width button (height 9 mm).

Preprocessing

Following related work [6, 10], we removed outliers that result from participants hitting the screen by accident (e.g. accidental double tap). These touches are clearly too far away from the target to be intended targeting actions. We removed touches which are further away from their target than the task's mean offset length plus three standard deviations (1.4 % of the data).

Evaluation Procedure

The data contains touches from 28 users. Hence, to extensively test our model, we can create sequences with and without user changes as follows: For each pair of users u_i, u_j, we concatenate n_i touches from u_i with n_j touches from u_j to form one sequence (of length $n_i + n_j$) that has a user change (if $i \neq j$) or no such change (if $i = j$). Note that this concatenation introduces no artefacts (e.g. unusually long time gap) since the sequence is still just a list of touch and target locations.

We then analyse these concatenated sequences with our model. Ideally, the model would predict a change for all sequences with touches from two different users (i.e. $i \neq j$), and no change for all sequences with touches from a single user (i.e. $i = j$).

Moreover, by varying n_i and n_j, we can 1) shift the change point's location within the sequence, and 2) evaluate the model on sequences of different lengths overall.

Performance Measures

To measure model performance, we use the two standard measures receiver-operating-characteristic area-under-curve (ROC AUC) and equal error rate (EER) [14]: In our case, the ROC curve is obtained by varying the decision threshold applied to the model's "score" (i.e. entropy, Equation 6). The curve plots false positives (missing to detect a change) against true positives (rightfully detecting a change). A high area under this curve (AUC) means that cases with and without changes can be distinguished well. The maximum AUC score (1) indicates perfect distinction. Moreover, EER is the point on the ROC curve closest to the line "false positive rate equals false negative rate". An EER of $X\%$ means that in $X\%$ of cases the system would incorrectly detect a change, while in $X\%$ of actual changes it would fail to detect them.

Results – Detecting User Changes

Table 1 shows the results of detecting user changes with three different variations of our model (linear, quadratic, hyperbolic basis functions). The models analysed sequences of 300 touches with one hand posture and target type, with change points located at 150 touches (i.e. at 50 % of the sequence).

Target type	Hand posture	Change point detection accuracy									
		linear Φ		quadr. Φ		hyperb. Φ		rel. work		combined	
		AUC	EER	AUC	EER	AUC	EER	AUC	EER	AUC	EER
cross	thumb	0.96	12	0.97	8	0.96	10	0.85	23	0.97	8
	index	0.92	17	0.92	16	0.92	16	0.79	29	0.94	13
key	thumb	0.94	15	0.96	15	0.95	13	0.86	23	0.97	8
	index	0.92	15	0.91	17	0.92	16	0.83	23	0.93	15
app	thumb	0.88	19	0.88	17	0.88	15	0.80	29	0.91	13
	index	0.89	21	0.91	19	0.91	17	0.78	31	0.93	15
fill	thumb	0.72	33	0.70	35	0.68	33	0.49	48	0.64	40
	index	0.76	35	0.74	33	0.74	33	0.53	46	0.73	29

Table 1. Evaluation of change point detection for different targets, hand postures, and offset models (i.e. basis functions Φ), with sequences of 300 touches and a true change point at 150 touches. The second to last column shows the results using the approach from related work [10], the last column combines it with our approach.

Target type	Hand posture	Change point detection accuracy					
		true cp at 25%		true cp at 50%		true cp at 75%	
		AUC	EER	AUC	EER	AUC	EER
cross	thumb	0.91	16	0.96	10	0.91	17
	index	0.85	23	0.92	16	0.85	23
key	thumb	0.90	19	0.95	13	0.89	21
	index	0.86	20	0.92	16	0.85	21
app	thumb	0.82	24	0.88	15	0.83	24
	index	0.81	27	0.91	17	0.81	26
fill	thumb	0.61	40	0.68	33	0.64	37
	index	0.68	37	0.74	33	0.68	37

Table 2. Evaluation of change point detection for different locations of the change point within the analysed sequence (for sequences of length 300 and the hyperbolic offset model). The results show that the model performs better if a sufficient number of touches is observed for behaviour both *before* and *after* the change point.

Target Types and Hand Postures

Behaviour changes were clearer for interactions with smaller targets. The large *fill* targets in particular hide user-specific targeting behaviour, explained by their large area that allows for less consistent behaviour [10]. Changes were also clearer for thumb touches than index finger ones. Both findings are in line with related work [10], which analysed individuality of offsets touch by touch (not as sequences with change points).

Model Variations

Comparing variations of our model, the one with hyperbolic basis functions performs best overall. It beats the quadratic one only by a slight margin. This suggests that the complexity of offset patterns is sufficiently captured by quadratic/hyperbolic terms regarding the detection of changes.

Change Point Location in the Sequence

If a change occurs near the start or end of a sequence, either behaviour before or after the change is only observed for few touches. Hence, we are interested in evaluating how the model reacts to change points in different places.

Table 2 shows results for change points at 25 %, 50 %, and 75 % of the whole sequence. These results show that change points are easier to find with our model when a sufficient

number of touches is observed for behaviour before and after them. This is to be expected: Behaviour changes are clearer if consistent behaviour can be observed around them, which only becomes evident with a certain number of touches. Nevertheless, visual inspection and high AUC values indicate that the model's posterior provides considerable evidence for change points even if they happen near the start/end of a sequence.

Sequence Length

We also analysed how evident change points appear to our model in sequences of different lengths. Figure 3 (left) shows the results: Changes are harder to spot in shorter sequences. In contrast, more data gives our model longer subsequences of consistent behaviour, making it easier to detect a change.

Comparison to Related Work / Non-Sequence Offset Models

Table 1 further shows results with the only previously published approach for distinguishing users purely based on touch offset patterns [10]. That approach uses non-sequence touch offset models [29] to evaluate and aggregate the likelihood of a specific user in contrast to others over all touches. Hence, it needs training data from the user that it seeks to distinguish from others; plus data from others.

Following the related work [10], for each evaluation case (see procedure above), we trained that model on data from u_i vs data from all others but u_j – using data from one study session, then testing on sequences concatenated from data of u_i and u_j from the other session. This is a recommended evaluation scheme from behavioural biometrics [8, 10]; further details on this model and approach can be found in said related work.

In contrast, our model does not require any data beyond the sequence at hand. Moreover, instead of aggregating single touches, it considers the sequence as such, that is, largely consistent behaviour over time, possibly changing at some point. Figure 3 shows that our sequence model is able to exploit more touches (EERs decrease per touch), while aggregation largely stagnates at higher EERs after a certain number of touches. The black regression lines in Figure 3 show this (EER in % falls -0.07 per touch with our model vs -0.02 with the non-sequence model).

Based on the results in Table 1, our model improves EERs by $\approx 40\,\%$ relative to the previously proposed approach for distinguishing users purely with touch offset patterns.

Model Combination

Finally, as the last column in Table 1 shows, it is possible to combine the "scores" obtained by both models (ours + related work [10]) to improve user change detection. We combined the models' outputs as a weighted sum, with weights informed by cross validation on one randomly chosen user's data. Intuitively, this combined model performs better since it utilises both indications of *known* behaviour and *change* of said behaviour: 1) The other model [10] trained as described above assesses if the behaviour matches to that of the expected user (u_i in our evaluation scheme, see above); and 2) our model assesses if the given sequence contains a behaviour change.

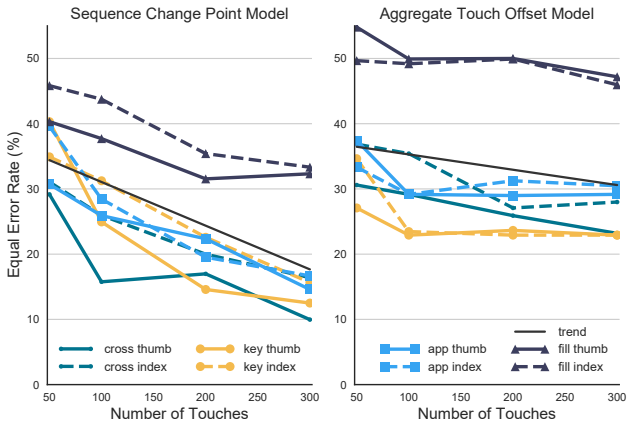

Figure 3. Change point detection for sequences of different lengths (true change point at 50 %). For our approach (left), more data leads to better accuracy (i.e. lower EER). In comparison, the previously proposed approach (right, [10]) remains at a higher error rate.

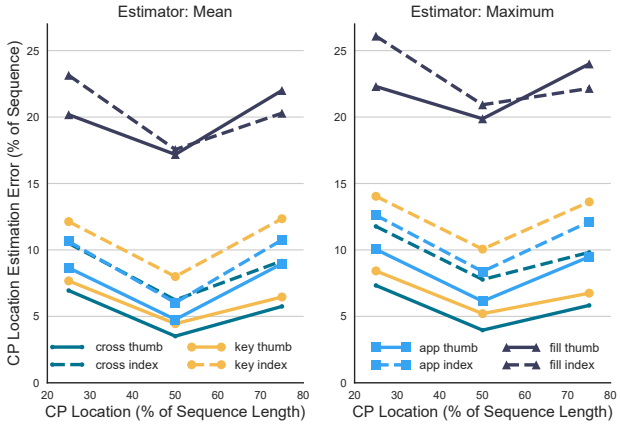

Figure 4. Change point location estimation (hyperbolic Φ, sequence with 300 touches). We measure the difference in percent of sequence length between predicted location and true location. We compare two estimators (left: mean, Equation 7; right: max, Equation 8) and three locations of the change point (x-axes: 25 %, 50 %, 75 % of sequence).

Results – Locating User Changes

We are also interested in detecting *where* a user change happens within a touch sequence. Our model inherently yields an estimate of this change point location (Equations 7 and 8). We again follow the described evaluation procedure, now only looking at cases with change points. We measure the distance between the estimated change point location and the true location, in % of the length of the whole sequence.

Target Types and Hand Postures

In line with results for detecting changes, estimating their locations is easier for thumb input than index finger touches (compare full vs dotted lines in Figure 4). Moreover, *fill* targets stand out with much higher estimation errors ($\approx$17.5 %), again explained by less consistent user behaviour for such large targets. In contrast, the model located changes for the smaller targets more accurately, with errors of 3.51 % to 8.00 %. For the tested sequence length, this means that changes are located within about 10 to 24 touches around the true point. To put this into context, in a typing sequence, for example, a user change might thus be located within up to 2 to 5 words.

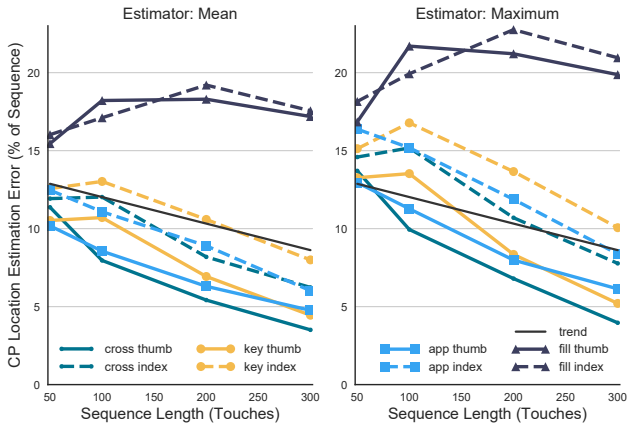

Figure 5. Change point location estimation for sequences of different lengths. The true change point is at 50 % of the sequence. The mean approach (Equation 7, left) is more accurate than the maximum approach (Equation 8, right). Overall, changes are more accurately located as more touches are observed.

Comparison of Change Point Location Estimators
Figure 4 (left vs right) shows two different approaches of estimating the change point location from our model's posterior: mean (Equation 7) and maximum (Equation 8). The mean estimator was more accurate for all targets and postures.

Change Point Location in the Sequence
Figure 4 shows results for changes at 25 %, 50 %, and 75 % of the sequence. In line with results for detecting changes, estimating change locations is easier when a sufficient number of touches is observed for behaviour before and after them.

Sequence Length
We also compared location estimation for sequences of different lengths. Figure 5 shows the results: Overall, change point estimation is more accurate relative to the length of the sequence as more touches are observed. An exception are the *fill* targets, again likely due to the highly inconsistent targeting behaviour evoked by their large area. Comparing both estimators, the mean approach is consistently better than the maximum approach for all tested sequence lengths. The trends are similar (error in % sequence length about -0.02 per touch).

STUDY II - MOBILE TOUCH TYPING IN THE WILD

We complement our lab analyses with evaluations on touch sequences collected from users' own everyday typing. This promises results of high ecological validity. In particular, an in-the-wild study allows us to observe targeting behaviour from a variety of everyday contexts for each user. This study is part of a larger research project (see [7] for further analyses); here we focus on targeting behaviour.

Apparatus

We modified Google's Android Open Source Project (AOSP) Keyboard[1] to record keyboard touch data. To protect users' privacy, the data was filtered on the device: In particular, we used a "trigram filter", which recorded touch locations only for three subsequent touches with a random chance of 10 %. After each recording, there was a minimum gap of at least one character. These parameters were tested in a pre-study to ensure that the recorded data would not reveal entered text. Typing in special text entry fields (e.g. name, address, password) was never recorded. Moreover, we added a "private mode" button to the keyboard with which participants could pause recording at any time.

We assessed hand posture via an experience sampling method (ESM): Our keyboard displayed a set of posture pictograms and asked participants to select their current hand posture. This posture selection overlay was shown when opening the keyboard, but no more than once every ten minutes.

Participants

We recruited 30 participants (15 female) via a newsletter and social media. Their mean age was 24 years (18-33); one was left-handed. Most were students and proficient touch keyboard users. They received a € 15 gift card as compensation.

Procedure

We deployed our app in a three-week field study on participants' own smartphones. App, study and examples of recorded data were explained in an initial meeting. In line with our university's regulations, people signed an informed consent form. Demographics were recorded with a questionnaire.

Participants then used their phone as usual. We did not instruct them, for example, to use certain hand postures (but we assessed postures via ESM, see apparatus).

After three weeks, we informed participants about the end of the study and asked them to uninstall the app. They also provided feedback in a post-study questionnaire. No data was recorded beyond the study.

Study Limitations

Samples in this study and the lab dataset are biased towards students and thus young people. Touch behaviour might be different for other user groups.

Moreover, our data might contain typing from non-participants (e.g. device sharing). However, all participants owned personal unshared devices. We also instructed them to use the "private mode" if they should need to share their device temporarily.

Since hand posture was assessed via ESM, it might not be perfectly accurate. A Likert question in the post-study questionnaire asked about this (*"I always selected the posture I was actually using"*). All but one person (strongly) agreed with this statement.

Finally, we only observed targeting behaviour while typing, not for tapping in general (e.g. app selection on home screen).

Preprocessing and Evaluation Procedure

In total, we recorded 204,685 touches with screen locations and the locations of their corresponding target keys. People used their own devices; related work [2, 13] pointed out that in this case touch locations need to be normalised for analyses across screen sizes. Hence, we normalised them using the recorded keyboard widths and heights.

[1] https://android.googlesource.com/platform/packages/inputmethods/LatinIME/, *accessed 7th October 2017.*

Hand posture	N	Change point detection accuracy									
		linear Φ		quadr. Φ		hyperb. Φ		rel. work		combined	
		AUC	EER	AUC	EER	AUC	EER	AUC	EER	AUC	EER
all	30	0.91	17	0.92	17	0.85	21	0.86	36	0.92	16
two thumbs	19	0.90	18	0.90	17	0.86	20	0.71	33	0.93	15
right thumb	12	0.88	16	0.88	20	0.84	23	0.64	38	0.87	19

Table 3. Change point detection on touch targeting sequences from typing in the wild (sequences of length 300 with true change point at 50 %).

Hand posture	N	Change point detection accuracy					
		true cp at 25%		true cp at 50%		true cp at 75%	
		AUC	EER	AUC	EER	AUC	EER
all	30	0.86	22	0.92	17	0.86	22
two thumbs	19	0.85	23	0.90	17	0.85	22
right thumb	12	0.81	26	0.88	20	0.81	26

Table 4. Evaluation of change point detection in the wild for different locations of the change point within the analysed sequence (for sequences of length 300 and the quadratic offset model).

For the following analyses, we used the same evaluation procedure as in the first study. However, we have many more touches here than in the lab. To allow for comparisons to the first study, we subsample shorter sequences of the same lengths as in study one (up to 300 touches). To avoid introducing randomness by this procedure, we follow the common ten-fold cross-validation scheme, that is, repeating all analyses on different subsequences and reporting the averaged results.

Results

To avoid redundancy with our report of the lab study, here we focus on differences, comparisons and additional insights.

Model Variations

The model with quadratic basis functions performed best overall (Table 3). It outperformed the linear one only slightly. This suggests that the complexity of offset patterns for typing in the wild is sufficiently captured by linear/quadratic terms regarding the detection of user changes. In contrast to the lab, the hyperbolic model performed worse. One explanation is that this model overfits on noisy everyday data, whereas the simpler linear/quadratic models are more robust.

Hand Postures

Table 3 shows results for two postures, namely typing with both thumbs (i.e. two handed use) or with the right one only (i.e. one handed use). The number of participants per posture differs, since not everyone used every posture. Other postures (e.g. index finger) occurred as well, but were used too rarely and/or by too few people for a meaningful analysis here.

Direct comparisons are difficult due to the different numbers of participants. Nevertheless, the results suggest the tendency that typing with both thumbs revealed user changes better than typing with the right thumb. This is in line with related work on password typing biometrics [8], which found in a lab study that users are easier to distinguish by typing behaviour with both thumbs compared to one.

Change Point Location and Sequence Lengths

As in the first study, we evaluated different locations of the change point (Table 4) as well as different sequence lengths.

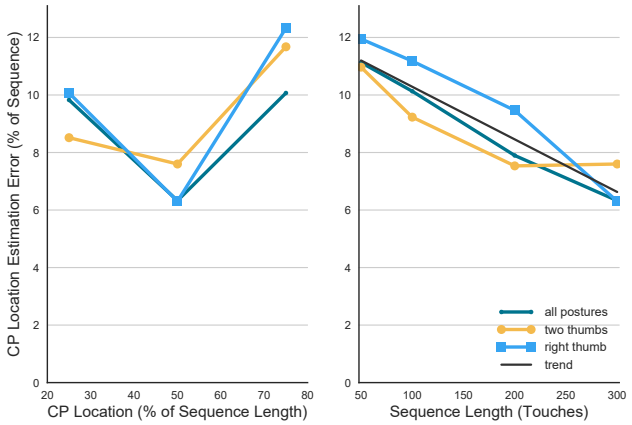

Figure 6. Change point location estimation on touch targeting sequences from typing in the wild. Left: Results for different locations of the change point within the sequence (of length 300 touches). Right: Results for sequences of different lengths (true change point at 50 %).

The results confirmed the findings from the lab study: Behaviour changes are clearer 1) if consistent behaviour can be observed both before and after them for a sufficient number of touches, and 2) for longer sequences overall. The impact on AUCs and EERs was comparable to the first study.

Performance in the Lab vs in the Wild

Table 1 and Table 3 show overall comparable change point detection accuracy for key targets with our model. In contrast, the non-sequence touch offset model [10] performs a lot worse (EER 23 in the lab vs 36 in the wild). We explain this with varying contexts of typing in daily life (e.g. on the go vs at home). Relying on training data is challenging if that data was collected in a different context. In contrast, our model analyses the given sequence without training on previous recordings, rendering it more robust to changing everyday situations.

Model Combination

The combined model still overall performed best (Table 3), but the advantage was less distinct than in the lab; the linear version of our model performed better than the combined model for typing with the right thumb. This is explained by the overall worse performance of the non-sequence model.

Locating User Changes

For locating the change point (Figure 6), this study overall confirmed the findings from the lab. Changes are more accurately located by our model 1) when a sufficient number of touches is observed before and after them; and 2) as more touches are observed (error in % sequence length about -0.02 per touch).

DISCUSSION

Improving Offset Features for Touch Biometrics

Our analyses show that mobile touch targeting sequences reveal information about user changes. However, are the observed results "good enough" to consider the model useful for future research and (biometric) applications? We conclude that the answer is yes, as explained below.

First, our model presents several advances compared to prior work: 1) It outperforms the existing non-sequential method for

exploiting touch offset patterns to detect changing users [10]. Comparing lab vs wild, it also better handles noisy real-world data. 2) Moreover, our model requires no training data. 3) Our model can be combined with the existing one [10] to improve user change detection accuracy overall. 4) Our EERs are also lower than those obtained for detecting user changes (between enrolment and authentication, i.e. using training data) with touch offset features on typing data in the lab [8] (our EER for typing 17-26 % vs 30 % in the related work). 5) Finally, to the best of our knowledge, we are the first to present a model for *locating* change points in sequences of touch offsets.

Second, regarding observed absolute EERs, we highlight our focus on touch offsets only: While this paper contributes a novel offset model, practical applications interested in detecting changing users can take into account more information. Related work showed that combining different behavioural biometrics improves accuracy [12]. This was also shown for the particular case of combining touch offsets with other touch features [8, 25]. Hence, related work demonstrates that improvements in utilising touch offsets, as achieved here, lead to overall better touch biometric systems.

We thus conclude that our model presents a useful novel building block for touch biometric systems that enables them to better exploit touch targeting (offset) sequences.

Modifying and Extending the Model

Online Change Point Detection

Some use cases require online change point detection. While our model analyses sequences post hoc, it can be integrated into an online version, for example, by processing the last N touches, updated per touch. We also plan to investigate dedicated online change point models (e.g. [1]) in future work.

Handling Multiple Change Points

Our model as described here assumes one change point. This is suitable for some applications, such as locking a phone or sending a warning email to its legitimate owner if a user change has been detected in the last touch sequence.

Other applications might involve (post hoc) analyses of interaction sequences that contain multiple user changes. The model formulation can be extended directly to multiple changes (see [24]). Even without modifications, *detecting* a change is possible if at least one change point is evident enough to our model. Thus, our model might be used in a "divide-and-conquer" scheme that (recursively) subdivides the sequence at a detected change to check the subsequences for further changes. We will investigate such extensions in future work.

Handling Short (Sub)Sequences

Our analyses revealed that user changes are harder to detect and locate if the model has little data to work with. This is the case for 1) short touch sequences and 2) short subsequences, that is, the user changes near the start/end of the sequence, such that only few touches are available for one of the users. To improve the model for such cases, it could be combined with an approach that uses recorded training data, as demonstrated with our combined model (also see next subsection).

Combining Change Detection and User Recognition

Our model detects user changes. Another task is to *recognise* users. For example, a touch biometric system might assign different parts of a finished browsing session on a shared device to specific users to inform future personal shopping recommendations based on who visited which website/product (see application context section). Such a task benefits from combining our model with a method to match the obtained subsequences (i.e. split at change points) to known users. One combination candidate is the model from prior work [10], which we already successfully combined with our model to improve the detection of changes itself (see Table 1 and 3).

Our model itself could also be used to help recognise a known user: Concatenate a previously recorded touch sequence with new touches. If our model detects no change point, this suggests that the new data comes from the same user as the old one. In contrast, if a change point is detected, this indicates that the new data comes from a different user. We plan to explore this idea further in future work, comparing it to other anomaly detection methods (see e.g. [3, 8]).

Causes of Changes in Mobile Touch Targeting Behaviour

We focussed on behaviour changes due to user changes, since 1) this extends related work on the user-specific nature of touch targeting behaviour [10, 29], and 2) promises to be useful as a building block for touch behaviour biometrics. Nevertheless, there are other reasons for changes in touch targeting behaviour, such as changing hand postures, starting to walk, and so on. Our model itself is flexible and could be applied for detecting other changes. An interesting challenge is to distinguish causes (e.g. new user or new hand posture?). To address this question our model could be combined with posture detectors [15, 30] and further information (e.g. application context, other sensor data).

Our model already reveals insights into the relative impact of context and user changes: Our typing data sequences contain touches from multiple postures ("all" postures row in Table 3) and likely from multiple contexts in the wild (e.g. at home vs on the go), yet the high AUCs indicate that our model still picked up user changes. This indicates that user changes are more pronounced in typing sequences than changing postures or situations – a promising result for touch biometrics.

Implementing and Integrating the Model in Applications

Our model has several favourable properties that facilitate easy implementation and integration: 1) It requires no training data. 2) It is lightweight (our implementation has about 30 lines of Python code). 3) The model runs fast and can utilise highly efficient matrix routines, since it can be written almost entirely in vectorised/matrix form.

CONCLUSION

Touch offset models capture 2D targeting behaviour patterns to improve touch accuracy or to recognise users and hand postures [6, 10, 16, 29]. We presented the first extension of these models to explicitly consider touches as *sequences*, enabling our model to assess *changes* in user behaviour. In particular, our model detects and locates user changes based on touch

targeting offsets. This is useful, for example, to detect misuse, to manage multiple user accounts, or to inform user-specific personalisation and recommendation systems.

We evaluated our model on smartphone targeting and typing data from the lab ($N = 28$) and field ($N = 30$). Depending on hand posture, target, sequence length and change point location, it detects user changes with a best equal error rate of 8 %, and locates them with a best error of 3.5 % sequence length. Our model outperforms the existing non-sequence touch offset models for detecting user changes and does not require training data. We demonstrated that it can be combined with existing models to further improve accuracy. Our model also provides a posterior distribution, useful for locating user changes within sequences and for combining it with other probabilistic information (e.g. from other sensors).

Hence, our model allows researchers and applications interested in changes in users' mobile interaction behaviour to better exploit touch targeting (offset) sequences. By releasing our model and data, we thus hope to provide a useful novel building block for touch biometrics.

PROJECT RESOURCES

Our models and data are available on the project's website: http://www.medien.ifi.lmu.de/touch-change

ACKNOWLEDGEMENTS

We thank Benjamin Bisinger for his help with the field study. Work on this project was partially funded by the Bavarian State Ministry of Education, Science and the Arts in the framework of the Centre Digitisation.Bavaria (ZD.B). This research was supported by the Deutsche Forschungsgemeinschaft (DFG), Grant No. AL 1899/2-1.

REFERENCES

1. Ryan Prescott Adams and David J. C. Mackay. 2007. Bayesian Online Changepoint Detection. *arXiv.org* (2007), 7. DOI:http://dx.doi.org/arXiv:0710.3742v1
2. Naif Alotaibi, Emmanuel Pascal Bruno, Michael Coakley, Alexander Gazarov, Stephen Winard, Filip Witkowski, Alecia Copeland, Peter Nebauer, Christopher Keene, and Joshua Williams. 2014. Text Input Biometric System Design for Handheld Devices. In *Proceedings of Student-Faculty Research Day, Pace University*. 1–8. https://pdfs.semanticscholar.org/6bbf/41e3a8b1ceb7ec9cdd4bb2797dd5ebb7a1ff.pdf
3. M. Antal and L. Z. SzabÃş. 2015. An Evaluation of One-Class and Two-Class Classification Algorithms for Keystroke Dynamics Authentication on Mobile Devices. In *20th International Conference on Control Systems and Computer Science*. 343–350. DOI: http://dx.doi.org/10.1109/CSCS.2015.16
4. Patrick Baudisch and Gerry Chu. 2009. Back-of-device Interaction Allows Creating Very Small Touch Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 1923–1932. DOI: http://dx.doi.org/10.1145/1518701.1518995
5. Joanna Bergstrom-Lehtovirta and Antti Oulasvirta. 2014. Modeling the Functional Area of the Thumb on Mobile Touchscreen Surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1991–2000. DOI: http://dx.doi.org/10.1145/2556288.2557354
6. Daniel Buschek and Florian Alt. 2015. TouchML: A Machine Learning Toolkit for Modelling Spatial Touch Targeting Behaviour. In *Proceedings of the 20th International Conference on Intelligent User Interfaces (IUI '15)*. ACM, New York, NY, USA, 110–114. DOI: http://dx.doi.org/10.1145/2678025.2701381
7. Daniel Buschek, Benjamin Bisinger, and Florian Alt. 2018. ResearchIME: A Mobile Keyboard Application for Studying Free Typing Behaviour in the Wild. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA. DOI: http://dx.doi.org/10.1145/3173574.317382
8. Daniel Buschek, Alexander De Luca, and Florian Alt. 2015a. Improving Accuracy, Applicability and Usability of Keystroke Biometrics on Mobile Touchscreen Devices. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 1393–1402. DOI: http://dx.doi.org/10.1145/2702123.2702252
9. Daniel Buschek, Alexander De Luca, and Florian Alt. 2015b. There is More to Typing Than Speed: Expressive Mobile Touch Keyboards via Dynamic Font Personalisation. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '15)*. ACM, New York, NY, USA, 125–130. DOI: http://dx.doi.org/10.1145/2785830.2785844
10. Daniel Buschek, Alexander De Luca, and Florian Alt. 2016. Evaluating the Influence of Targets and Hand Postures on Touch-based Behavioural Biometrics. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1349–1361. DOI: http://dx.doi.org/10.1145/2858036.2858165
11. Daniel Buschek, Simon Rogers, and Roderick Murray-Smith. 2013. User-specific Touch Models in a Cross-device Context. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '13)*. ACM, New York, NY, USA, 382–391. DOI:http://dx.doi.org/10.1145/2493190.2493206
12. Heather Crawford, Karen Renaud, and Tim Storer. 2013. A framework for continuous, transparent mobile device authentication. *Computers & Security* 39, Part B (2013), 127 – 136.
13. Benjamin Draffin, Jiang Zhu, and Joy Zhang. 2014. *KeySens: Passive User Authentication through Micro-behavior Modeling of Soft Keyboard Interaction*.

Springer International Publishing, Cham, 184–201. DOI: http://dx.doi.org/10.1007/978-3-319-05452-0_14

14. Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern Recognition Letters* 27, 8 (2006), 861 – 874. DOI: http://dx.doi.org/10.1016/j.patrec.2005.10.010
15. Mayank Goel, Jacob Wobbrock, and Shwetak Patel. 2012. GripSense: Using Built-in Sensors to Detect Hand Posture and Pressure on Commodity Mobile Phones. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 545–554. DOI: http://dx.doi.org/10.1145/2380116.2380184
16. Niels Henze, Enrico Rukzio, and Susanne Boll. 2011. 100,000,000 Taps: Analysis and Improvement of Touch Performance in the Large. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI '11)*. ACM, New York, NY, USA, 133–142. DOI:http://dx.doi.org/10.1145/2037373.2037395
17. Christian Holz and Patrick Baudisch. 2010. The Generalized Perceived Input Point Model and How to Double Touch Accuracy by Extracting Fingerprints. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 581–590. DOI: http://dx.doi.org/10.1145/1753326.1753413
18. Christian Holz and Patrick Baudisch. 2011. Understanding Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2501–2510. DOI: http://dx.doi.org/10.1145/1978942.1979308
19. Josip Musić and Roderick Murray-Smith. 2016. Nomadic Input on Mobile Devices: The Influence of Touch Input Technique and Walking Speed on Performance and Offset Modeling. *Human-Computer Interaction* 31, 5 (2016), 420–471. DOI: http://dx.doi.org/10.1080/07370024.2015.1071195
20. Josip Musić, Daryl Weir, Roderick Murray-Smith, and Simon Rogers. 2016. Modelling and correcting for the impact of the gait cycle on touch screen typing accuracy. *mUX: The Journal of Mobile User Experience* 5, 1 (19 Apr 2016), 1. DOI: http://dx.doi.org/10.1186/s13678-016-0002-3
21. Matei Negulescu and Joanna McGrenere. 2015. Grip Change As an Information Side Channel for Mobile Touch Interaction. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 1519–1522. DOI: http://dx.doi.org/10.1145/2702123.2702185
22. Albert Ng, Michelle Annett, Paul Dietz, Anoop Gupta, and Walter F. Bischof. 2014. In the Blink of an Eye: Investigating Latency Perception During Stylus Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1103–1112. DOI: http://dx.doi.org/10.1145/2556288.2557037
23. Carl Edward Rasmussen and Christopher K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. The MIT Press. http://www.ncbi.nlm.nih.gov/pubmed/15112367
24. Peter Rasmussen. 2001. Bayesian Estimation of change points using the general linear model. *Water Resources Research* 37, 11 (2001), 2723–2731. DOI: http://dx.doi.org/10.1029/2001WR000311
25. Pin Shen Teh, Ning Zhang, Andrew Beng Jin Teoh, and Ke Chen. 2016. A survey on touch dynamics authentication in mobile devices. *Computers and Security* 59 (2016), 210–235. DOI: http://dx.doi.org/10.1016/j.cose.2016.03.003
26. Michael E. Tipping. 2001. Sparse Bayesian Learning and the Relevance Vector Machine. *Journal of Machine Learning Research* 1 (2001), 211–244. http://www.ai.mit.edu/projects/jmlr/papers/volume1/tipping01a/abstract.html
27. Daniel Vogel and Patrick Baudisch. 2007. Shift: A Technique for Operating Pen-based Interfaces Using Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. ACM, New York, NY, USA, 657–666. DOI: http://dx.doi.org/10.1145/1240624.1240727
28. Daryl Weir, Daniel Buschek, and Simon Rogers. 2013. Sparse Selection of Training Data for Touch Correction Systems. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '13)*. ACM, New York, NY, USA, 404–407. DOI: http://dx.doi.org/10.1145/2493190.2493241
29. Daryl Weir, Simon Rogers, Roderick Murray-Smith, and Markus Löchtefeld. 2012. A User-specific Machine Learning Approach for Improving Touch Accuracy on Mobile Devices. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 465–476. DOI: http://dx.doi.org/10.1145/2380116.2380175
30. Ying Yin, Tom Yu Ouyang, Kurt Partridge, and Shumin Zhai. 2013. Making Touchscreen Keyboards Adaptive to Keys, Hand Postures, and Individuals: A Hierarchical Spatial Backoff Model Approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 2775–2784. DOI: http://dx.doi.org/10.1145/2470654.2481384