Improving the Input Accuracy of Touchscreens using Deep Learning

Abinaya Kumar
University of Stuttgart
Stuttgart, Germany
abinayakumar.ak@gmail.com

Aishwarya Radjesh
University of Stuttgart
Stuttgart, Germany
aishwaryaradjesh@gmail.com

Sven Mayer
University of Stuttgart
Stuttgart, Germany
info@sven-mayer.com

Huy Viet Le
University of Stuttgart
Stuttgart, Germany
mail@huyle.de

ABSTRACT

Touchscreens combine input and output in a single interface. While this enables an intuitive interaction and dynamic user interfaces, the fat-finger problem and the resulting occlusions still impact the input accuracy. Previous work presented approaches to improve the touch accuracy by involving visual features on the top side of fingers, as well as static compensation functions. While the former is not applicable on recent mobile devices as the top side of a finger cannot be tracked, compensation functions do not take properties such as finger angle into account. In this work, we present a data-driven approach to estimate the 2D touch position on commodity mutual capacitive touchscreens which increases the touch accuracy by 23.0 % over recently implemented approaches.

CHI'19 Extended Abstracts, May 4–9, 2019, Glasgow, Scotland, UK

ACM ISBN 978-1-4503-5971-9/19/05.
https://doi.org/10.1145/3290607.3312928

CCS CONCEPTS

• **Human-centered computing** → **Interaction techniques**; *User studies*; *Laboratory experiments*; *Touch screens*.

KEYWORDS

Touchscreen; touch input; targeting; input accuracy; capacitive image; smartphone; deep learning.

ACM Reference Format:
Abinaya Kumar, Aishwarya Radjesh, Sven Mayer, and Huy Viet Le. 2019. Improving the Input Accuracy of Touchscreens using Deep Learning. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts (CHI'19 Extended Abstracts), May 4–9, 2019, Glasgow, Scotland, UK*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3290607.3312928

INTRODUCTION AND RELATED WORK

Virtually all mobile phones use touchscreens as the main interface. Touchscreens combine input and output in a single surface which brings a wide range of advantages. Not only do touchscreens enable manufacturers to build compact and robust devices; they also enable dynamic user interfaces (UI) based on the concept of direct touch. Similar to physical objects, users can interact with elements of the UI by simply touching, moving, or rotating them. This enables an intuitive interaction for the user while the full front side of a mobile device is usable for both input as well as output.

Over 10 years, smartphones exclusively use mutual capacitive touchscreens due to their responsiveness, durability, and support for multi-touch. On a technical basis, these touchscreens consist of spatially separated electrodes in two layers which are arranged as rows and columns. The controller first measures the change of coupling capacitance between two orthogonal electrodes to obtain low-resolution finger imprints [2], and then translates the imprints into two-dimensional coordinates which often correspond to the centroid of the contact area [6].

While the translation is kept simple, users can neither see nor feel whether they "touched" the desired target (*i.e.* assess whether the 2D coordinate is within the activation area of a target) as the finger tip occludes the area below. Indeed, Holz and Baudisch [5, 6] found that users target based on visual features located on the top/along their fingers (*e.g.*, nail bottom and center of finger nail) in contrast to the contact area's centroid which is occluded. This mismatch between the user's mental model and the touch device's translation process results in error offsets of 4 mm around the target. While using the finger's visual features to reconstruct the user's mental model reduces the error offset to 1.6 mm, this is not applicable to recent mobile devices as the finger's upper side is not trackable.

Improving the touch accuracy is important as precise touch input increases the user's performance (less corrections are required) and reduces errors (*e.g.*, activating unintended functions). Henze *et* *al.* [3, 4] observed users' touch behavior within games featuring typing and targeting tasks, and found that the touch input is systematically skewed on average. By deriving a static compensation function, they showed that touch offsets can be reduced by 7.8 % for targeting and 9.1 % for typing. Recent on-screen keyboards such as SwiftKey follow a similar approach and monitor user-based offset patterns to automatically compensate for touch errors based on input modeling[1].

[1]SwiftKey Typing Heatmap: https://blog.swiftkey.com/access-swiftkey-heatmap/

A large body of previous work presented active approaches to compensate touch errors produced by occlusions. Shift [16] displays occluded screen content in a call-out above the finger and is recently used in most iOS devices for precise text cursor placement. A similar approach by Roudaut *et al.* [14] provides a magnified popup which shows the occluded content and enables users to perform fine-grained input. Similarly, previous work also investigated the use of offset cursors which extend the thumb by a line [7]. Instead of performing input by touching a target, previous work [8, 15] proposed to move the screen content below a target instead of the target itself. By using input beyond the touchscreen, previous work also proposed input on the rear to avoid occlusions [1, 10, 12].

The presented approaches require either external tracking mechanisms to track features on the top/along the side of the fingers, are based on static compensation functions, or require additional actions to achieve precise touch input. We present a first step towards increasing the touch accuracy based solely on the raw data of mutual capacitive touchscreens and deep learning. In particular, we present a model which translate the raw capacitive data into a 2D touch coordinate with an average error offset of 2.35 *mm* which already represents an improvement of 23.0 % over recent touch controllers. We further share our data set to enable future work to reproduce and improve our approach.

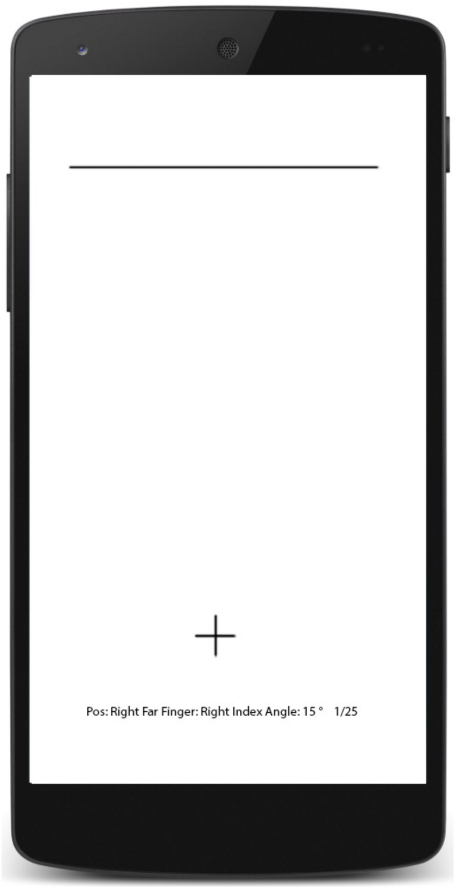

Figure 1: Screenshot of the study application.

DATA COLLECTION

We conducted a study to collect *capacitive images* (*i.e.* the raw data of a capacitive touchscreen) of touches and their ground truth position.

Tasks & Study Design

To collect *capacitive images* labeled with the respective touch position, we developed a target selection task in which participants target cross hairs on the screen. Thereby, we instructed participants to aim for the intersection within the cross hair which we consider to be the ground truth position of a touch.

Based on a previous study by Holz and Baudisch [6], we used a $2 \times 4 \times 4$ within-subjects design with the independent variables being the Finger (thumb and index finger), HeadPosition (right near, right far, front, below head), and the finger's pitch Angle (15°, 25°, 45°, 60°). We used the same HeadPositions as Holz and Baudisch which needs to be considered due to the head parallax. The finger's pitch angle was considered since it affects the area of finger imprint. We randomized the order of the conditions. Each condition consists of 25 targets.

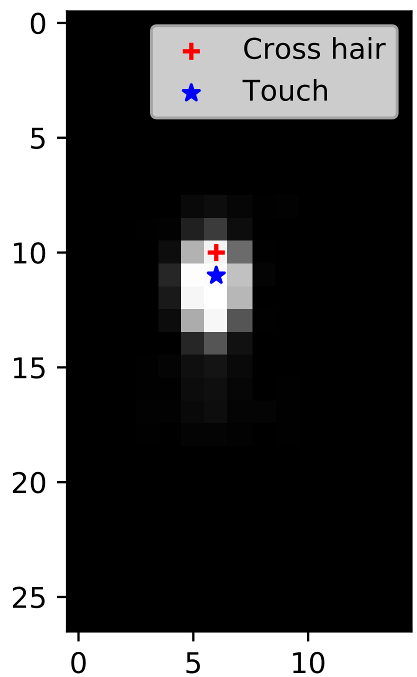

Figure 2: Demonstrating the offset between the desired touch position (i.e. center of cross hair) and the registered touch.

Apparatus

We used an LG Nexus 5 with a modified kernel as described in previous work [9, 11, 13] to access the 15×27 8-bit capacitive image of the Synaptics ClearPad 3350 touch sensor. An exemplary image of the raw capacitive data is shown in Figure 2, whereas each image pixel corresponds to a $4.1\,mm \times 4.1\,mm$ square on the $4.95''$ touchscreen. The pixel values represent the differences in electrical capacitance (in pF) between the baseline measurement and the current measurement. We developed an application which represents a target selection task to collect the ground truth position of touches. The application is depicted in Figure 1, and shows a cross hair which represents the target to touch. All capacitive images were logged in the background.

Participants & Procedure

We recruited 12 participants (8 male and 4 female) with an average age of 23.9 ($SD = 1.4$) from our university's volunteer pool. After we obtained informed consent, we seated participants in front of a table on which the device was flatly placed according to the condition. We used a protractor to ensure that participants applied the correct pitch angle prior to each condition. While stabilizers or markers would be more accurate, previous work [6] have discussed that these distract participants and thus distort the data. We instructed participants to touch the cross hair as accurate as possible (*i.e.*, at the intersection of the vertical and horizontal line). To capture a sufficient amount of capacitive images per target, we further instructed participants to hold the finger for two seconds on the touchscreen until the displayed progress bar is filled. After removing the finger, a new target was displayed. The study took around 60 minutes per participant.

MODELING

We present our data set and describe two steps towards a model to improve the touch accuracy. This includes cleaning and preparing the data set, and training convolutional neural networks (CNNs), the state-of-the-art for image data, to estimate the intended touch position based on a capacitive image.

Dataset & Preprocessing

In total, we collected 392,292 capacitive images during the study. We filtered empty images in which no touches were performed, as well as erroneous images in which more than one finger was touching the screen to avoid wrong labels. In particular, we performed a blob detection using OpenCV to determine whether a single touch happened as expected. The blob detection omitted all blobs that were not larger than one pixel of the image ($4.1\,mm \times 4.1\,mm$) as these can be considered as noise of the capacitive touchscreen. Further, we removed all images in which the difference between cross hair and touch is larger than $300\,px$ (*e.g.*, caused by unintended touches). In total, our data set consists

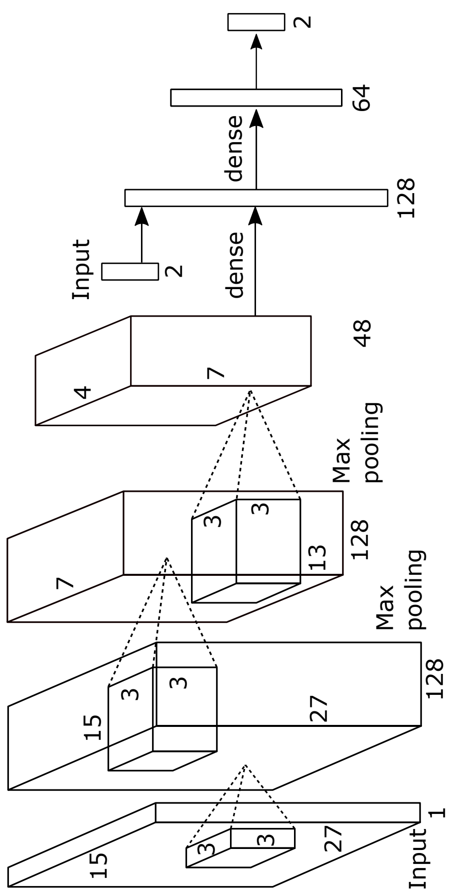

Figure 3: Architecture of the convolutional neural network used for estimating the 2D position of a touch.

of 345, 973 valid capacitive images which we used for model development. We split the dataset into a training (75 %) and a test set (25 %). The data was split participant-wise so that data of the same participant did not occur in both sets (*i.e.*, we used data of 8 participants to train and 4 for testing).

Convolutional Neural Network

We implemented a CNN using *Keras* 2.2.4 based on the *TensorFlow* backend. We performed a grid search to determine the most suitable hyperparameters. If we do not report a hyperparameter in the following, we applied the standard value (*e.g.*, optimizer settings) as reported in Keras' documentation.

Our final CNN architecture is shown in Figure 3. The input consists of capacitive images with 27×15 pixels normalized to a range between 0 and 1 which goes through two convolution and pooling layers as well as a flattening layer. In addition, we further provide an initial estimated touch position represented by the centroid of the touch blob. We determined the centroid using OpenCV's blob detection which could be easily re-implemented on mobile devices. Our initial estimated 2D position is added to the fully connected layer in addition the features from the convolution and pooling layers.

The output consists of 2 values (x, y) that represent the estimated 2D touch position relative to the upper left corner of the display in px. We trained the CNN using an RMSprop optimizer with a batch size of 64. We experimented with different learning rates and found that an initial learning rate of .001 leads to the best performance. To prevent overfitting, we used a 0.5 dropout after each convolution and pooling layer. While we experimented with L2 Regularization, it did not improve the overall performance in our experiments. We initialized the network weights using the Xavier initialization scheme. We used the root mean squared error (RMSE) as the loss function.

Our CNN achieved an average error offset of $41.23\,px$ ($SD = 22.96\,px$) for the test set which equals to $2.35\,mm$ ($SD = 1.31\,mm$) based on a screen resolution of $1920 \times 1080\,px$ on a $4.95''$ display ($0.0571\,mm$ per pixel). In comparison, the controller of our test device (LG Nexus 5) achieved an average error offset of $50.7\,px$ ($SD = 24.97\,px$) which equals to $2.89\,mm$ ($SD = 1.43\,mm$). This results in an improvement of 23.0 % with our CNN over the standard touch controller.

DISCUSSION AND CONCLUSION

We presented a data-driven approach to improve the touch input accuracy on mutual capacitive touchscreens. Our approach can be readily deployed on commodity touch input devices and further considers properties of the touch which static compensation functions (*e.g.*, [3, 4]) cannot consider. In contrast to static compensation functions, which were shown to improve the touch accuracy by up to 9.1 %, our CNN improved the touch accuracy by 23.0 % without any additional sensors for tracking. As our model could act as a replacement for the recent translation of touch surfaces into 2D coordinates, we assume that it can improve the touch accuracy on current touch input devices at almost no costs.

We presented our model, under which the error offsets drop to 2.35 *mm* from 2.89 *mm*, which involves different finger postures sensed by mutual capacitive touchscreens to approximate how users conceptualize touch input. Based on our publicly released dataset and model[2], future work could evaluate our touch translation approach in a (long-term) study as well as more scenarios such as while walking. Moreover, future work could further improve our approach using touchscreens which provide touch images in a higher resolution (*e.g.*, with infrared sensing such as on the SUR40). As different users apply different targeting strategies, user-based models could be trained on the device after a short calibration phase with the task used in our study.

[2]https://github.com/interactionlab/improving-touch-accuracy

Acknowledgements: This work was financially supported by the German Research Foundation (DFG) within Cluster of Excellence in Simulation Technology (EXC 310/2) at the University of Stuttgart.

REFERENCES

[1] Patrick Baudisch and Gerry Chu. 2009. Back-of-device Interaction Allows Creating Very Small Touch Devices. In Proc. of *CHI '09*. https://doi.org/10.1145/1518701.1518995
[2] Li Du. 2016. An Overview of Mobile Capacitive Touch Technologies Trends. *arXiv preprint arXiv:1612.08227* (2016).
[3] Niels Henze, Enrico Rukzio, and Susanne Boll. 2011. 100,000,000 Taps: Analysis and Improvement of Touch Performance in the Large. In Proc. of *MobileHCI '11*. https://doi.org/10.1145/2037373.2037395
[4] Niels Henze, Enrico Rukzio, and Susanne Boll. 2012. Observational and Experimental Investigation of Typing Behaviour Using Virtual Keyboards for Mobile Devices. In Proc. of *CHI '12*. https://doi.org/10.1145/2207676.2208658
[5] Christian Holz and Patrick Baudisch. 2010. The Generalized Perceived Input Point Model and How to Double Touch Accuracy by Extracting Fingerprints. In Proc. of *CHI '10*. https://doi.org/10.1145/1753326.1753413
[6] Christian Holz and Patrick Baudisch. 2011. Understanding Touch. In Proc. of *CHI '11*. https://doi.org/10.1145/1978942.1979308
[7] Sunjun Kim, Jihyun Yu, and Geehyuk Lee. 2012. Interaction Techniques for Unreachable Objects on the Touchscreen. In Proc. of *OzCHI '12*. https://doi.org/10.1145/2414536.2414585
[8] Huy Viet Le, Patrick Bader, Thomas Kosch, and Niels Henze. 2016. Investigating Screen Shifting Techniques to Improve One-Handed Smartphone Usage. In Proc. of *NordiCHI '16*. https://doi.org/10.1145/2971485.2971562
[9] Huy Viet Le, Thomas Kosch, Patrick Bader, Sven Mayer, and Niels Henze. 2018. PalmTouch: Using the Palm as an Additional Input Modality on Commodity Smartphones. In Proc. of *CHI '18*. https://doi.org/10.1145/3173574.3173934
[10] Huy Viet Le, Sven Mayer, Patrick Bader, Frank Bastian, and Niels Henze. 2017. Interaction Methods and Use Cases for a Full-Touch Sensing Smartphone. In Proc. of *CHI EA '17*. https://doi.org/10.1145/3027063.3053196
[11] Huy Viet Le, Sven Mayer, Patrick Bader, and Niels Henze. 2017. A Smartphone Prototype for Touch Interaction on the Whole Device Surface. In Proc. of *MobileHCI EA '17*. https://doi.org/10.1145/3098279.3122143
[12] Huy Viet Le, Sven Mayer, and Niels Henze. 2018. InfiniTouch: Finger-Aware Interaction on Fully Touch Sensitive Smartphones. In Proc. of *UIST '18*. https://doi.org/10.1145/3242587.3242605
[13] Sven Mayer, Huy Viet Le, and Niels Henze. 2017. Estimating the Finger Orientation on Capacitive Touchscreens Using Convolutional Neural Networks. In Proc. of *ISS '17*. https://doi.org/10.1145/3132272.3134130
[14] Anne Roudaut, Stéphane Huot, and Eric Lecolinet. 2008. TapTap and MagStick: Improving One-handed Target Acquisition on Small Touch-screens. In Proc. of *AVI '08*. https://doi.org/10.1145/1385569.1385594
[15] Kenji Suzuki, Kazumasa Okabe, Ryuuki Sakamoto, and Daisuke Sakamoto. 2016. Fix and Slide: Caret Navigation with Movable Background. In Proc. of *MobileHCI '16*. https://doi.org/10.1145/2935334.2935357
[16] Daniel Vogel and Patrick Baudisch. 2007. Shift: A Technique for Operating Pen-based Interfaces Using Touch. In Proc. of *CHI '07*. https://doi.org/10.1145/1240624.1240727