University of Stuttgart
Germany

Hand-and-Finger-Awareness for Mobile Touch Interaction using Deep Learning

Huy Viet Le

Universität Stuttgart

Hand-and-Finger-Awareness for Mobile Touch Interaction using Deep Learning

Von der Fakultät für Informatik, Elektrotechnik und Informationstechnik
und dem Stuttgart Research Centre for Simulation Technology
der Universität Stuttgart zur Erlangung der Würde eines Doktors der
Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

Huy Viet Le

aus Nürtingen

Hauptberichter:	Prof. Dr. Niels Henze
Mitberichter:	Prof. Antti Oulasvirta
Mitprüfer:	Jun.-Prof. Dr. Michael Sedlmair

Tag der mündlichen Prüfung: 28.03.2019

Institut für Visualisierung und Interaktive Systeme
der Universität Stuttgart

2019

Zusammenfassung

Mobilgeräte wie Smartphones und Tablets haben mittlerweile Desktop Computer für ein breites Spektrum an Aufgaben abgelöst. Nahezu jedes Smartphone besitzt einen berührungsempfindlichen Bildschirm (Touchscreen), welches die Ein- und Ausgabe zu einer Schnittstelle kombiniert und dadurch eine intuitive Interaktion ermöglicht. Mit dem Erfolg sind Anwendungen, die zuvor nur für Desktop Computer verfügbar waren, nun auch für Mobilgeräte verfügbar. Dieser Wandel steigerte die Mobilität von Computern und erlaubt es Nutzern dadurch Anwendungen auch unterwegs zu verwenden.

Trotz des Erfolgs von Touchscreens sind traditionelle Eingabegeräte, wie Tastatur und Maus, aufgrund ihrer Eingabemöglichkeiten immer noch überlegen. Eine Maus besitzt mehrere Tasten, mit denen verschiedene Funktionen an derselben Zeigerposition aktiviert werden können. Zudem besitzt eine Tastatur mehrere Hilfstasten, mit denen die Funktionalität anderer Tasten vervielfacht werden. Im Gegensatz dazu beschränken sich die Eingabemöglichkeiten von Touchscreens auf zweidimensionale Koordinaten der Berührung. Dies bringt einige Herausforderungen mit sich, die die Benutzerfreundlichkeit beeinträchtigen. Unter anderem sind Möglichkeiten zur Umsetzung von Kurzbefehlen eingeschränkt, was Sheidermans goldene Regeln für das Interface Design widerspricht. Zudem wird meist nur ein Finger für Eingabe verwendet, was die Interaktion verlangsamt. Weitere Herausforderungen, wie das Fat-Finger Problem und die limitierte Erreichbarkeit

auf großen Geräten, tragen mit Unbequemlichkeiten bei. Neue berührungsbasierte Interaktionsmethoden werden benötigt, um die Eingabemöglichkeiten auf Touchscreens zu erweitern und die Eingabe mit mehreren Fingern, wie es bei traditionellen Eingabegeräten üblich ist, zu ermöglichen.

In dieser Arbeit wird untersucht, wie es einzelnen Fingern und Teile der Hand ermöglicht werden kann, Eingaben auf einem mobilen Gerät zu tätigen und zwischen deren Eingaben zu unterscheiden. Dieses Konzept wird als "*Hand-und-Finger-bewusste*" Interaktion bezeichnet. Durch die Erkennung von Hand und Finger können einzelnen Fingern und Teile der Hand verschiedene Funktionen zugewiesen werden, was die Eingabemöglichkeit erweitert und viele Herausforderungen der Touch Interaktion löst. Des Weiteren ermöglicht die Anwendung des Konzepts der "*Hand-und-Finger-bewussten*" Interaktion auf die komplette Geräteoberfläche nun auch die Verwendung der hinteren Finger zur Eingabe, die bisher nur das Gerät hielten. Dies addressiert weitere Herausforderungen der Touch Interaktion und bietet viele Möglichkeiten zur Realisierung von Kurzbefehlen.

Diese Dissertation enthält die Ergebnisse aus zwölf Studien, welche sich auf die Design Aspekte, die technische Realisierbarkeit und die Benutzerfreundlichkeit der "*Hand-und-Finger-bewussten*" Interaktion fokussieren. Im ersten Schritt wird die Ergonomie und das Verhalten der Hand untersucht, um die Entwicklung neuer Interaktionstechniken zu inspirieren. Anschließend wird erforscht, wie gut einzelne Finger und Teile der Hand mit Hilfe von Deep Learning Techniken und Rohdaten von kapazitiven Sensoren erkannt werden können. Dabei wird sowohl ein einzelner kapazitiver Bildschirm, als auch ein Gerät, das rundum Berührungen registriert, verwendet. Basierend darauf präsentieren wir vier Studien, die sich damit befassen Kurzbefehle von Computer-Tastaturen auf mobile Geräte zu bringen, um die Benutzerfreundlichkeit von Textverarbeitung auf Mobilgeräten zu verbessern. Wir folgen dabei dem angepassten benutzerzentriertem Designprozess für die Anwendung von Deep Learning.

Der Kernbeitrag dieser Dissertation erstreckt sich von tieferen Einsichten zur Interaktion mit verschiedenen Fingern und Teilen der Hand, über einen technischen Beitrag zur Identifikation der Berührungsquelle mit Hilfe von Deep Learning Techniken, bis hin zu Ansätzen zur Lösung der Herausforderungen mobiler Berührungseingabe.

Abstract

Mobile devices such as smartphones and tablets have replaced desktop computers for a wide range of everyday tasks. Virtually every smartphone incorporates a touchscreen which enables an intuitive interaction through a combination of input and output in a single interface. Due to the success of touch input, a wide range of applications became available for mobile devices which were previously exclusive to desktop computers. This transition increased the mobility of computing devices and enables users to access important applications even while on the move.

Despite the success of touchscreens, traditional input devices such as keyboard and mouse are still superior due to their rich input capabilities. For instance, computer mice offer multiple buttons for different functions at the same cursor position while hardware keyboards provide modifier keys which augment the functionality of every other key. In contrast, touch input is limited to the two-dimensional location of touches sensed on the display. The limited input capabilities slow down the interaction and pose a number of challenges which affect the usability. Among others, shortcuts can merely be provided which affects experienced users and contradicts Shneiderman's golden rules for interface design. Moreover, the use of mostly one finger for input slows down the interaction while further challenges such as the fat-finger problem and limited reachability add additional inconveniences. Although the input capabilities are sufficient for simple applications, more complex everyday tasks which require intensive input, such

as text editing, are still not widely adopted yet. Novel touch-based interaction techniques are needed to extend the touch input capabilities and enable multiple fingers and even parts of the hand to perform input similar to traditional input devices.

This thesis examines how individual fingers and other parts of the hand can be recognized and used for touch input. We refer to this concept as *hand-and-finger-awareness* for mobile touch interaction. By identifying the source of input, different functions and action modifiers can be assigned to individual fingers and parts of the hand. We show that this concept increases the touch input capabilities and solves a number of touch input challenges. In addition, by applying the concept of *hand-and-finger-awareness* to input on the whole device surface, previously unused fingers on the back are now able to perform input and augment touches on the front side. This further addresses well-known challenges in touch interaction and provides a wide range of possibilities to realize shortcuts.

We present twelve user studies which focus on the design aspects, technical feasibility, and the usability of *hand-and-finger-awareness* for mobile touch interaction. In a first step, we investigate the hand ergonomics and behavior during smartphone use to inform the design of novel interaction techniques. Afterward, we examine the feasibility of applying deep learning techniques to identify individual fingers and other hand parts based on the raw data of a single capacitive touchscreen as well as of a fully touch sensitive mobile device. Based on these findings, we present a series of studies which focus on bringing shortcuts from hardware keyboards to a fully touch sensitive device to improve mobile text editing. Thereby, we follow a user-centered design process adapted for the application of deep learning.

The contribution of this thesis ranges from insights on the use of different fingers and parts of the hand for interaction, through technical contributions for the identification of the touch source using deep learning, to solutions for addressing limitations of mobile touch input.

Acknowledgements

Over the past three years, I had one of the best times of my life working together with a number of amazing colleagues and friends who inspired me a lot. Without their support, this work would never have been possible.

First and foremost, I would like to thank my supervisor **Niels Henze** who inspired my work and always supported me in the best possible ways to achieve my goals. Without his support, I would have never came this far. I further thank my committee **Antti Oulasvirta**, **Michael Sedlmair**, and **Stefan Wagner** for the great and inspiring discussions. Discussions with **Syn Schmitt** in the SimTech milestone presentation, and a number of student peers and mentors in doctoral consortia at international conferences further shaped my thesis. I would also like to thank **Albrecht Schmidt** for all his great support which even goes beyond research. Moreover, I thank **Andreas Bulling** for the opportunity to stay another five months to finalize my thesis.

Before my time as a PhD student, I had the great honor to meet a number of awesome people who introduced me into the world of Human-Computer Interaction research. I thank **Alireza Sahami Shirazi** for his outstanding supervision during my bachelor's thesis. His inspiration and recommendations played a huge role in getting me into HCI research. I further thank **Tilman Dingler** for his exceptional support and organization which provided me with the opportunity to write my master's thesis at the Lancaster University. During my time in Lancas-

ter, I had a great and memorable time working with **Corina Sas**, **Nigel Davies**, and **Sarah Clinch**. I further thank **Mateusz Mikusz** who helped me finding an accommodation and ensured that everything was fine.

I had the great pleasure to work with amazingly helpful and skilled colleagues who shaped my time as a PhD student. I thank my incredible office mates **Dominik Weber**, **Hyunyoung Kim**, and **Nitesh Goyal** for all the inspiring discussions and for bearing the time with me while I typed on my mechanical keyboard. I am further thankful for all the collaborations which taught me how to write papers, build prototypes, and supervise students. In particular, I thank **Sven Mayer** for sharing his research experiences and for all the great work together which resulted in a lot of publications. I further thank **Patrick Bader** for sharing his endless knowledge on hardware prototyping and algorithms. I also thank **Francisco Kiss** for helping me with his extensive knowledge in electrical engineering and soldering skills. I am further thankful to **Katrin Wolf** for inspiring me a lot with her experiences in mobile interaction, and **Lewis Chuang** for the valuable collaboration.

A PhD is not only work but also a lot of fun. I thank **Jakob Karolus** and **Thomas Kosch** for the great and adventurous road trips through the US. I further thank the rest of the awesome hcilab group in Stuttgart who made every day a really enjoyable day: **Alexandra Voit**, **Bastian Pfleging**, **Céline Coutrix**, **Lars Lischke**, **Mariam Hassib**, **Matthias Hoppe**, **Mauro Avila**, **Miriam Greis Norman Pohl**, **Pascal Knierim**, **Passant El.Agroudy**, **Paweł W. Woźniak**, **Rufat Rzayev**, **Romina Poguntke**, **Stefan Schneegaß**, **Thomas Kubitza**, **Tonja Machulla**, **Valentin Schwind**, and **Yomna Abdelrahman**. A special thanks goes to **Anja Mebus**, **Eugenia Komnik** and **Murielle Naud-Barthelmeß** for all their support and the administrative work that keeps the lab running smoothly.

It was also a pleasure to work with awesome student assistants who supported me in conducting studies, recruiting participants, and transcribing interviews. This thesis would have not been possible without the support of **Jamie Ullerich**, **Jonas Vogelsang**, **Max Weiß**, and **Henrike Weingärtner** - thank you!

Last but not least, I would like to thank my family for their unconditional support - my father **Hung Son Le** and mother **Thi Bich Lien Luu** for raising me to be the person I am today, for inspiring and making it possible for me to get

the education I wanted, and to making it possible for me to explore technology. I thank my sister **Bich Ngoc Le** for being there for me and supporting me in all possible ways. Further, I thank all my friends for their emotional support and patience that they showed me on my way to the PhD.

Thank you!

Table of Contents

1 Introduction **15**
1.1 Research Questions 18
1.2 Methodology 19
1.2.1 Limitations of the User-Centered Design Process 19
1.2.2 Limitations of Common Deep Learning Processes 21
1.2.3 User-Centered Design Process for Deep Learning 22
1.3 Research Context 25
1.4 Thesis Outline 27

2 Background and Related Work **29**
2.1 Background 29
2.1.1 History and Development of Touch Interaction 30
2.1.2 Capacitive Touch Sensing 33
2.2 Related Work 35
2.2.1 Hand Ergonomics for Mobile Touch Interaction 35
2.2.2 Novel Touch-Based Interaction Methods 39
2.2.3 Interacting with Smartphones Beyond the Touchscreen 47
2.3 Summary 50

3 Hand Ergonomics for Mobile Touch Interaction **53**
3.1 Interaction Beyond the Touchscreen 54
3.1.1 Reachability of Input Controls 55
3.1.2 Unintended Inputs 56
3.2 Study I: Range and Comfortable Area of Fingers 57
3.2.1 Study Design 57
3.2.2 Apparatus 58
3.2.3 Procedure 59
3.2.4 Participants 60
3.2.5 Data Preprocessing 61
3.2.6 Results 64
3.2.7 Discussion 69
3.3 Study II: Investigating Unintended Inputs 71
3.3.1 Study Design 71
3.3.2 Apparatus 73
3.3.3 Tasks and Procedure 74
3.3.4 Participants 75
3.3.5 Data Preprocessing 76
3.3.6 Results 77
3.3.7 Discussion 87
3.4 General Discussion 90
3.4.1 Summary 90
3.4.2 Design Implications 91

4 Hand-and-Finger-Awareness on Mobile Touchscreens **93**
4.1 Identifying the Source of Touch 94
4.1.1 The Palm as an Additional Input Modality 94
4.1.2 Investigating the Feasiblity of Finger Identification 99
4.2 Input Technique I: Palm as an Additional Input Modality (*PalmTouch*) . . 101
4.2.1 Data Collection Study 101
4.2.2 Model Development 104
4.2.3 Evaluation 108
4.3 Input Technique II: Finger Identification 117
4.3.1 Data Collection Study 117

4.3.2 Model Development . . . 121
4.3.3 Evaluation . . . 128
4.4 General Discussion . . . 135
4.4.1 Summary . . . 136
4.4.2 Lessons Learned . . . 136
4.4.3 Data Sets . . . 138

5 Hand-and-Finger-Awareness on Full-Touch Mobile Devices 139
5.1 *InfiniTouch*: Finger-Aware Input on Full-Touch Smartphones . . . 140
5.1.1 Full-Touch Smartphone Prototype . . . 140
5.1.2 Ground Truth Data Collection . . . 144
5.1.3 Finger Identification Model . . . 147
5.1.4 Validation . . . 150
5.1.5 Mobile Implementation and Sample Applications . . . 152
5.1.6 Discussion and Limitations . . . 156
5.2 Exploring Interaction Methods and Use Cases . . . 160
5.2.1 Interviews . . . 160
5.2.2 Results . . . 162
5.2.3 Discussion . . . 166
5.3 General Discussion . . . 167
5.3.1 Summary . . . 167
5.3.2 Lessons Learned . . . 168

6 Improving Shortcuts for Text Editing 171
6.1 Text Editing on Mobile Devices . . . 172
6.1.1 Study Overview . . . 173
6.2 Study I: Shortcuts on Hardware Keyboards . . . 174
6.2.1 Apparatus . . . 174
6.2.2 Procedure and Participants . . . 174
6.2.3 Log Analysis: Shortcuts on Hardware Keyboards . . . 175
6.2.4 Interviews: Hardware and Touchscreen Keyboards . . . 176
6.2.5 Discussion . . . 180
6.3 Study II: Gesture Elicitation . . . 181
6.3.1 Referents . . . 181

6.3.2 Apparatus and Procedure 182
6.3.3 Participants 183
6.3.4 Results 183
6.3.5 Gesture Set for Shortcuts in Text-Heavy Activities 188
6.3.6 Discussion 189
6.4 Study III: Implementing the Gesture Set on a Full-Touch Smartphone . . 190
6.4.1 Apparatus 190
6.4.2 Participants 190
6.4.3 Procedure and Study Design 191
6.4.4 Modeling 191
6.4.5 Mobile Implementation 195
6.5 Study IV: Evaluation of Shortcut Gestures 196
6.5.1 Study Procedure and Design 196
6.5.2 Apparatus 198
6.5.3 Participants 199
6.5.4 Results 199
6.5.5 Discussion 207
6.6 General Discussion 209
6.6.1 Summary 209
6.6.2 Lessons Learned 210

7 Conclusion and Future Work **211**
7.1 Summary of Research Contributions 212
7.2 Future Work 215

Bibliography **217**

List of Acronyms **256**

1

Introduction

Over two billion people own a mobile device such as a smartphone or a tablet [285]. With their mobility and increasing processing capability, mobile devices replaced personal computers and laptops for the majority of everyday computing tasks. Millions of downloads on mobile app stores show that applications such as email clients, web browsers, calendars, and even editors for various media have become viable alternatives to their desktop counterparts. While mobile phones started with arrays of hardware buttons and a small display, recent smartphones incorporate a touchscreen that combines input and output in a single interface. This enables users to directly touch elements of the user interface (UI) and interact with them intuitively similar to physical objects.

With touchscreens, smartphones can be designed as compact and self-contained mobile devices which leverage the whole front side for input as well as output. As a consequence, a wide range of applications previously designed for computers with keyboard and mouse are now also offering touch-based UIs. This transition increases the mobility of computing devices and enables users to use their device even while on the move. However, keyboards and mice as input devices are still superior to touch input since they provide more input capabilities. The difference is noticeable especially for complex tasks which require high precision (*e.g.* pla-

cing the caret in a text) and repetitive actions for which shortcuts are commonly used (*e.g.* copy and paste). Limited input capabilities slow down the interaction and lead to a lack of shortcuts which are fundamental for experienced users as described by Shneiderman's golden rules for interface design [209].

In contrast to touchscreens, a computer mouse offers multiple buttons which enable users to activate different functions at the same cursor position. Similarly, hardware keyboards offer modifier keys (*e.g.*, `Ctrl`, `Alt`, and `Shift`) which add additional dimensions to every other key. Touchscreens, however, translate a touch on the display into a two-dimensional coordinate which is mapped to the UI. While direct manipulation is powerful, the input's expressiveness is limited to single coordinates despite the sheer amount of additional information that a smartphone could provide about a touch. With *3D Touch*[1], Apple showed that touch input can be purposefully extended by a pressure modality based on a proprietary technology involving an additional sensing layer. While this is the prime commercial example, the touch input vocabulary on commodity smartphones can also be extended without additional sensors beyond the touchscreen. In particular, the raw data of capacitive touchscreens was used for estimating the touch contact size [24], shape [182], and the orientation of a finger on the display [156, 198, 265]. These interaction techniques generally leverage properties beyond touch coordinates to provide additional input dimensions. However, mapping functions to specific finger postures increases the likelihood of unintended activations since a finger is now controlling multiple modalities simultaneously.

One solution to lower the likelihood of unintended activations is to identify the touching finger or part of the hand to avoid interference with the main finger for interaction (*e.g.* the thumb). Previous work [38, 63, 82] identified parts of the finger (*e.g.* knuckle) or individual fingers to use the touch source as an additional input modality. However, the number of fingers that can touch the display during the prevalent single-handed grip [109, 110, 176, 178] is limited while additional wearable sensors [74, 75, 152] are required for an accurate finger identification. Differentiating between inputs of multiple fingers and hand parts while enabling them to interact with the device would profoundly extend the touch input capabilities. This would make smartphones more suitable for tasks

[1] `https://developer.apple.com/ios/3d-touch/`

which require complex inputs and help to solve common touch input limitations such as the fat-finger problem [16, 217], reachability issues [20, 133], and the lack of shortcuts. Without requiring immobile and inconvenient wearable sensors, or a second hand which is not always available, smartphones could become an even more viable and mobile replacement for personal computers and laptops.

One step towards this vision was presented by previous work on Back-of-Device (BoD) interaction (*e.g.* [16, 39, 46, 133, 197, 250, 269]). With the input space extended to the rear, fingers that previously held the device are now able to perform input. However, previous work treated the touch-sensitive rear as an additional input space but not as an opportunity to enable individual fingers to perform specific input. Generally, only grip patterns were considered [33, 35, 36], while touch-sensitive areas were limited so that only the index finger can perform BoD input [10, 46, 133]. Consequently, the input space was extended but individual fingers and hand parts are still not usable as different input modalities.

Touch inputs from individual hand parts and fingers need to be recognized and differentiated to use them as unique input modalities. In particular, the raw data of capacitive sensors (such as from recent touchscreens) contain enough signal which could be used to infer the source of a touch. With deep learning, robust and lightweight models could be developed which identify hand parts and fingers on nowadays' smartphones. This concept profoundly extends the mobile touch input vocabulary and will be referred to as *hand-and-finger-aware* interaction.

Before this concept can be used on commodity smartphones, a wide range of challenges need to be addressed. First, designing *hand-and-finger-aware* interactions with a focus on usability requires an understanding of the behavior and ergonomics of individual fingers while holding smartphones. There is no previous work which analyzes the reachable areas for each finger, nor the areas in which fingers typically move and reside. Second, the technical feasibility of identifying individual hand parts and fingers needs to be investigated. There is no system yet which identifies fingers and hand parts with accuracies usable for realistic everyday scenarios based on the raw data of commodity capacitive touch sensing technologies. Third, we also need to evaluate the concept of *hand-and-finger-awareness* with potential users to gather feedback. This enables to improve the concept to a level which is ready for the mass-market.

1.1 Research Questions

In this thesis, we explore the concept of *hand-and-finger-aware* interaction for mobile devices. To inform the design and development of *hand-and-finger-aware* interaction methods, we present an exploration of six high-level research questions (RQs). The RQs are presented in Table 1.1.

An important basis to design input on the whole device surface is the analysis of finger movements which do not require a grip change. Since a grip change leads to a loss of grip stability and could lead to dropping the device, we need to understand the range which individual fingers can cover and the areas in which they can comfortably move (RQ1). In addition to explicit movements, we further need to understand micro-movements which fingers perform while interacting with the device. An understanding is vital to minimize unintended inputs generated by these movements (RQ2).

We use the raw data of capacitive sensors to identify hand parts and fingers based on deep learning. Before this approach can be leveraged for *hand-and-finger-aware* interaction, we need to investigate its feasibility and usability. We investigate the identification of hand parts and fingers using the raw data of a single capacitive touchscreen, *i.e.* on today's commodity smartphones (RQ3). We further examine the feasibility of identifying individual fingers on fully touch sensitive smartphones (RQ4). This would enable the fingers on the rear to perform input, while the grip can be reconstructed for further interaction techniques.

After understanding the ergonomics and behavior of all fingers while holding and interacting with smartphones, we evaluate *hand-and-finger-aware* interaction for common use cases. This helps to understand how this concept can be leveraged to further improve mobile interaction. Since touch input on recent mobile devices poses a number of limitations, we investigate how we could address them on a fully touch sensitive smartphone. This includes an elicitation of the limitations and potential solutions proposed by experienced interaction designers (RQ5). Finally, we focus text editing as a specific use case which the interaction designers identified as important but still inconvenient due to the limited input capabilities. In particular, we investigate the design and implementation of shortcuts on fully touch sensitive smartphones to improve text editing (RQ6).

Research Question	No.	Chapter
I. Hand Ergonomics for Mobile Touch Interaction		
How can we design Back-of-Device input controls to consider the reachability of fingers in a single-handed grip?	(RQ1)	Chapter 3
How can we design Back-of-Device input controls to minimize unintended inputs?	(RQ2)	Chapter 3
II. Identifying Fingers and Hand Parts		
How can we differentiate between individual fingers or hand parts on a capacitive touchscreen?	(RQ3)	Chapter 4
How can we estimate the position of individual fingers and identify them on a fully touch sensitive smartphone?	(RQ4)	Chapter 5
III. Improving Mobile Touch Interaction		
Which typical touch input limitations could be solved with a fully touch sensitive smartphone?	(RQ5)	Chapter 5
How can we design and use shortcuts on a fully touch sensitive smartphone to improve text editing?	(RQ6)	Chapter 6

Table 1.1: Summary of research questions addressed in this thesis.

1.2 Methodology

Designing, developing, and evaluating novel interaction techniques is one of the major topics in human-computer interaction (HCI). The goal of an interaction technique is to provide users with a way to accomplish tasks based on a combination of hardware and software elements.

1.2.1 Limitations of the User-Centered Design Process

Previous work in HCI presented novel interaction techniques based on the user-centered design (UCD) process [102] as shown in Figure 1.1. The UCD process outlines four phases throughout an iterative design and development cycle to develop interactive systems with a focus on usability. The process consists of phases for understanding the context of use, specifying the user requirements, and developing a solution (*i.e.*, implementing a working prototype) which is evaluated against the requirements. Each cycle represents an iteration towards a

solution which matches the users' context and satisfies all of the relevant needs (*e.g.*, increasing the usability to a level which satisfies relevant users). The UCD process focuses on the concept of the solution itself, assuming that specified user requirements can be unambiguously translated into a working prototype. Indeed, previous work commonly identified the need and requirements of an interaction technique and prototyped them using hand-crafted algorithms which range from simple value comparisons [152], thresholding [24, 74], and transfer functions [39] through computer vision techniques [93, 96] to kinematic models [23, 202].

With the advent of deep learning, complex relationships and patterns (*e.g.*, in sensor data) can be learned from large amounts of data. Due to the increased availability of computing power and open-source frameworks (*e.g.*, TensorFlow[1], Keras[2], PyTorch[3]), deep learning became a powerful tool for HCI researchers to develop solutions which are robust, lightweight enough to run on mobile devices, and do not even require domain knowledge (*e.g.*, for a particular sensor and its noise). In addition, major parts of the prototypes can be reused even in market-ready versions of the system by reusing the data for model development or retraining the model for similar sensors. Prominent examples include object recognition in image data which even outperform humans [87, 88, 218].

Despite the powerful modeling capabilities, deep learning produces black box models which can hardly be understood by humans. Due to the lack of knowledge about a deep learning model's internal workings, the model needs to be trained, tested, and validated with potential users within multiple iterations until it achieves the desired result. In contrast, the UCD process describes the design of a solution in a single step without involving potential users, an evaluation of its usability in a subsequent step, and a full refinement in a further iteration. Due to the huge effort required for developing a deep learning model (*i.e.* gathering a data set and multiple iterations of model development), the UCD process needs to be refined in order to incorporate iterative developments and tests of a model, as well as evaluating the model's usability within the whole interactive system. In particular, the designing solution step needs to incorporate the modeling cycle of a deep learning process and connect it to the usability aspects of the UCD.

[1] TensorFlow: `https://www.tensorflow.org/`
[2] Keras: `https://keras.io/`
[3] PyTorch: `https://pytorch.org/`

1.2.2 Limitations of Common Deep Learning Processes

A typical process for developing and evaluating deep learning models consists of four phases: gathering a representative data set (*e.g.*, through a data collection study or using already existing ones), preparing the data (*e.g.*, exploring, cleaning, and normalizing), training and testing the model, as well as validating its generalizability on previously unseen data. Thereby, training and testing are often repeated in multiple iterations to find the most suitable hyperparameters that lead to the lowest model error on the test set based on trial-and-error and grid search [101] approaches. A final model validation with previously unseen data then assesses whether the chosen hyperparameters were overfitting to the test set.

For this process, the deep learning community often use a training-test-validation split [42] (*i.e.*, training and test set for model development, and the validation set for a one-time validation of the model) to develop and validate a model's performance. However, software metrics alone (*i.e.*, accuracies and error rates to describe how well the model generalizes to unseen data) do not describe the usability of a system which is the main focus of the UCD process. Instead of software metrics, factors such as the effect of inference errors on the usability (*i.e.* how well is the perceived usability for a given use case and how impactful are errors?), the model stability (*i.e.* how noisy are the estimations over time for none to small variations?), and the usefulness of the investigated system should be considered. As systems are used by a wide range of users and in different scenarios, the validation also needs to assess whether the model can generalize beyond the (specific and/or abstract) tasks used in a data collection study. Moreover, while previous work considered accuracies above 80 % to be sufficient [113], sufficiency depends on the use case (*i.e.* whether the action's consequence is recoverable and how much the consequence affects the user) which can only be evaluated in studies through user feedback.

In summary, a typical process for deep learning describes the iterative nature of developing and evaluating black box models, but does not consider the usability of the model and thus of the final system. To apply deep learning techniques in HCI, we need to refine and combine the UCD process with typical deep learning processes to consider both the iterative development and evaluation of models, as well as their usability within the final system.

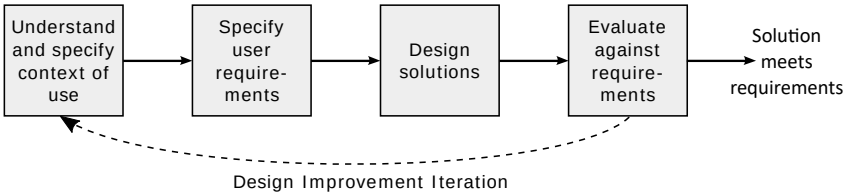


Figure 1.1: The user-centered design process as described in ISO 9241-210 [102].

1.2.3 User-Centered Design Process for Deep Learning

We present the user-centered design process for deep learning (UCDDL) which combines the UCD process with steps required for deep learning and is depicted in Figure 1.2. The UCDDL consists of five phases, whereas the first two phases are identical to the traditional UCD process and focus on understanding users as well as specifying requirements. The next three phases focus on developing a prototype based on deep learning and evaluating the system based on the factors described above. In the following, we describe the UCDDL which we apply throughout this thesis.

1. Understand and specify the context of use. This phase is about identifying users who will use the system, their tasks, and under which conditions they will use it (*e.g.*, technical and ergonomic constraints). This step could consist of user studies to understand the context of use, or based on findings from previous work.

2. Specify user requirements. Based on the identified context, application scenarios and prototype requirements need to be specified. Based on these requirements, the solution will be developed and evaluated against.

3. Collect data based on user requirements. Training a deep learning model requires a representative and large enough data set as the ground truth. Gathering this data set in the context of a user study involves the design and development of an apparatus which runs mockup tasks to cover all expected interactions.

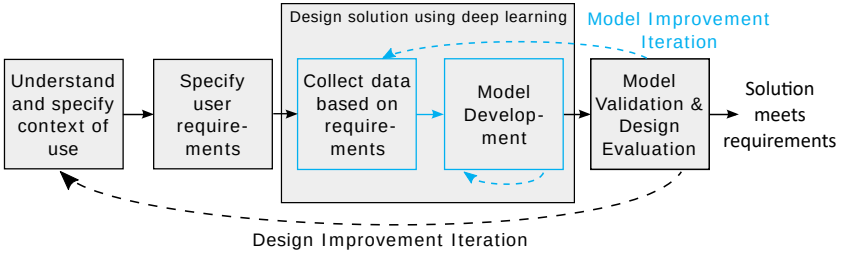


Figure 1.2: Adapted user-centered design process for deep learning in the context of interactive systems in HCI.

Instructing potential users to perform certain tasks even enables the apparatus to automatically label each collected sample. This assumes that the experimenter is carefully observing whether participants actually perform the requested input correctly (*e.g.*, when instructing participants to touch with a certain finger, it can be assumed that the captured data samples represent the instructed finger). The user study needs to be conducted with a representative set of potential users which cover all relevant factors to collect a sufficient amount of data for model training.

The data set is the foundation of the developed system and needs to be refined (*i.e.*, extended with more variance by adding users and tasks to cover the specified requirements) in case the final system does not generalize to new users and tasks which were specified in the requirements. In this case, another data collection study needs to be conducted whereas the resulting new data set needs to be combined with the already existing data set. In addition, the data collected in the evaluation phase (see Phase 5) could also be used to extend the existing data set.

4. Model development. Based on the data set, this phase applies deep learning to develop a model which is used by the system. Prior to the actual model training, the data set often needs to be cleaned (*e.g.*, removing empty or potentially erroneous samples for which the label correctness cannot be ensured) or augmented in case producing the desired amount of data is not feasible (*e.g.* adding altered samples such as by rotating the input or adding artificial sensor noise). Further, we first explore the data set with techniques such as visual inspection, descriptive

and inferential statistics (*e.g.* finding correlations), as well as applying basic machine learning models such as linear regression and SVMs using simple feature extraction. This step provides an overview of the data set and helps choosing the optimal model and hyperparameters in later steps. In case only very few samples could be collected (*e.g.*, due to a high effort for collecting or labeling), these basic models represent a viable solution.

After data preprocessing and exploration, the data set needs to be split into a training and test set to avoid the same samples being "seen" during training and testing. Since the same user could generate highly similar data, the data set should further be split by participants (instead of by samples as commonly applied). Previous work commonly used a rate of 80 %:20 % for a training-test split, and a 70 %:20 %:10 % for a training-test-validation split. While the deep learning community commonly use a training-test-validation split to detect overfitting to the test set due to hyperparameter tuning, the UCDDL process replaces the validation set with a user study in the next phase. This has two advantages: First, the full data set can be used to train the model and test it based on the test set. Second, the user study in the next phase can gather a validation set with new participants which are usually larger than 10 % of the data set. More importantly, the model's usability (and also the accuracy) can be evaluated in a realistic scenario based on feedback from potential users. This is not possible with a training-test-validation split which focuses only on the modeling aspect.

The goal of the training process is to achieve the highest accuracy on the test set. The model is then deployed in the respective system (*e.g.* a mobile device in this thesis) for the evaluation in the next phase.

5. Model Validation and Design Evaluation. This phase evaluates the system as a whole with participants who did not participate in the data collection study (Phase 3). The evaluation focuses on three aspects: (1) a model validation to achieve the same results as the commonly used training-test-validation approach (combined with training and test of the previous phase), (2) evaluating the model usability (and optionally also the model error) in a realistic but controlled scenario

to focus on individual aspects, and (3) evaluating the system within a common use case (as specified in Phase 2) to assess the usefulness of the system and the perceived usability of the model in a uncontrolled scenario.

The model validation replaces the validation set based on similar tasks as used in the data collection study. In particular, data is collected with the same tasks which, at the same time, can also be used to introduce participants into the system. This prepares them for the usability evaluation within realistic scenarios which consists of a set of tasks that resemble a realistic use case. This set of tasks is designed to be controlled enough to enable a focus on individual aspects of the system (*e.g.*, recognition accuracy and usability of certain classes of the model). For instance, a set of tasks could be designed in a pre-defined order so that model predictions can be compared with the order to determine the accuracy. To focus on the perceived usability, tasks could also be designed to expect only one type of input (*i.e.* one class). This enables to evaluate false positives for a certain class while collecting qualitative feedback from the participants about the used class. More complex outputs, such as regression, could employ additional sensors such as high-precision motion capture systems as ground truth. For the usability evaluation of the full system, participants use the prototype to solve tasks in a fully functional environment (*e.g.*, an application designed for a certain use case, or even well-known applications). This step is less controlled and focuses on the system's usability and usefulness. This results in qualitative feedback and quantitative measures such as the task completion time or success rate.

In summary, the evaluation in the UCDDL covers the model validation as well as the usability aspect as described in the UCD process.

1.3 Research Context

The research leading to this thesis was carried out over the course of three years (2016 – 2018) in the Socio-Cognitive Systems group at the Institute for Visualization and Interactive Systems. It was additionally part of a project funded in the Cluster of Excellence in Simulation Technology (SimTech) at the University of Stuttgart. The presented research was inspired by collaborations, publications, and discussions with many experts from within and outside the field of HCI.

Cluster of Excellence in Simulation Technology SimTech is an interdisciplinary research association with more than 200 scientists from virtually all faculties of the University of Stuttgart. A major part of the research was conducted in the project network "PN7 - Reflexion and Contextualisation"[1]. The research presented in this thesis underwent an examination in the form of a mid-term presentation accompanied by Prof. Dr. Syn Schmitt from the Institute of Sports and Exercise Science. Moreover, intermediate research results were presented at the annual SimTech Status Seminar.

University of Stuttgart The research presented in this thesis was inspired by collaborations with colleagues from the University of Stuttgart. With the scientific expertise and technical knowledge from Patrick Bader, Thomas Kosch, and Sven Mayer we published six publications which are all in the scope of this thesis [123–125, 130, 132, 136]. Moreover, the collaborations resulted into further publications with relevant topics but beyond the scope of this thesis [117, 128, 133, 155, 156, 158–160] and tutorials on "Machine Learning for HCI" organized at national as well as international conferences [126, 134, 157]. Amongst others, online magazines and communities such as Arduino[2], hackster.io[3], and open-electronics.org[4] reported on our prototypes presented in this work.

The research was further inspired by discussions with a broad range of student peers and senior researchers at the doctoral consortium at the International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI 2016) [122] and the ACM CHI Conference on Human Factors in Computing Systems (CHI 2018) [121]. In addition, collaborations with Patrick Bader, Passant El.Agroudy, Tilman Dingler, Valentin Schwind, Alexandra Voit, and Dominik Weber resulted in publications beyond the scope of this thesis [11, 12, 48, 89, 131, 137, 247].

[1] http://www.simtech.uni-stuttgart.de/en/research/networks/7/

[2] http://blog.arduino.cc/2018/10/19/infinitouch-interact-with-both-sides-of-your-smartphone/

[3] http://blog.hackster.io/dual-sided-smartphone-interaction-with-infinitouch-6362c4181fa2

[4] http://www.open-electronics.org/infinitouch-is-the-first-fully-touch-sensitive-smartphone/

External Collaborations Further research beyond the scope of this thesis was conducted with external collaborators. This includes Katrin Wolf from the Hamburg University of Applied Sciences [129], Lewis Chuang from the Max Planck Institute for Biological Cybernetics [160], Sarah Clinch, Nigel Davies, and Corina Sas from the Lancaster University [131], as well as Agon Bexheti, Marc Langheinrich, and Evangelos Niforatos from the Università della Svizzera italiana [48].

1.4 Thesis Outline

This thesis consists of seven chapters, the bibliography, and the appendix. We present the results and evaluations of 12 empirical studies, an extensive review of related work, as well as a discussion and summary of the findings in the conclusion chapter. We structure the work as follows:

Chapter 1 - Introduction motivates the research in this thesis and gives an overview about the research questions and the author's contributions. We further present the user-centered design process for deep learning which we follow throughout this thesis.

Chapter 2 - Background provides an overview of the history of touch interaction, an explanation of capacitive touch sensing, as well as an extensive review of touch-based interaction techniques on mobile devices and beyond.

Chapter 3 - Hand Ergonomics for Mobile Touch Interaction describes the results of two studies investigating the behavior and ergonomic constraints of finger while holding a mobile device.

Chapter 4 - Hand-and-Finger-Awareness on Mobile Touchscreens presents two models that use the raw data of capacitive touchscreens to recognize the source of touch, and their evaluations within realistic use cases.

Chapter 5 - Hand-and-Finger-Awareness on Full-Touch Mobile Devices develops a smartphone prototype with touch sensing on the whole device surface and shows how fingers can be identified. Further, we discuss how full-touch smartphones can solve recent touch input limitations.

Chapter 6 - Improving Shortcuts for Text Editing applies the findings from the previous chapters and presents four studies which cover all steps from understanding shortcut use on keyboards, a gesture elicitation study, a data collection study to train a gesture recognizer using deep learning, and finally an evaluation study.

Chapter 7 - Conclusion and Future Work discusses the findings from the previous chapters, summarizes them, and provides directions for further research.

2

Background and Related Work

While touchscreens enable intuitive interactions, keyboards and mice as input devices are still superior to touch input as they provide more input capabilities by enabling the use of multiple fingers. In this thesis, we explore novel touch-based interaction techniques which differentiate between individual fingers and hand parts to solve limitations of recent mobile touch interaction. To understand the technologies used in this thesis, this chapter provides an introduction to touch-based interaction as well as its history and technical background. We further review previous work in the domain of extending touch interaction and present recent challenges of mobile touch interaction which we address in this thesis.

2.1 Background

Touchscreens are ubiquitous in our modern world. According to statista [285], over 2.5 billion people own a smartphone with a touchscreen as the main interface. People use smartphones for tasks which were previously exclusive to stationary computers and in a wide range of scenarios such as while sitting, walking, encumbered, or even during other tasks. The combination of input and output in

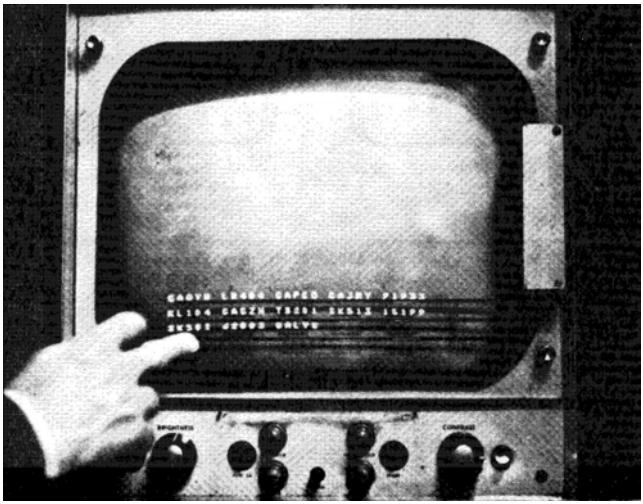


Figure 2.1: The first touchscreen as developed by E.A. Johnson. Image taken from [106].

a single interface enable intuitive interaction through direct touch. Moreover, touchscreens enable manufacturers to build compact and robust devices which use nearly the whole front surface for input and output.

2.1.1 History and Development of Touch Interaction

The first finger-based touchscreen was invented in 1965 by E.A. Johnson [105] who described a workable mechanism for developing a touchscreen. As with most consumer devices nowadays, the presented prototype used capacitive sensing. The inventor envisioned the invention to be used for air-traffic-control, such as facilitating selections of call signs, flights, and executive actions [106, 184]. Figure 2.1 shows the display configuration for the touch interface. Five years later, Samuel Hurst and his research group at the University of Kentucky developed the first resistive touchscreen in 1970. In contrast to capacitive sensing methods as invented by E.A. Johnson, resistive touchscreens were more durable back then, not expensive to produce, and operation is not restricted to conductive objects such as human skin or conductive pens. Nowadays, resistive touch sensing can be found

mostly in public areas such as restaurants, factories, and hospitals. In 1972, the first widely deployed touchscreen based on infrared light was developed [55], and was deployed in schools throughout the united states. This technology employed fingers interrupting light beams that ran parallel to the display surface.

In 1982, Nimish Mehta [162] developed the first multi-touch device which used a frosted-glass panel with a camera behind it so that it could detect action which are recognizable through black spots showing up on the screen. Gestures similar to today's pinch-to-zoom or manipulation through dragging were first presented in a system by Krueger *et al.* [116]. Although the system was vision-based and thus is not suitable for touch interaction, many of the presented gestures could be readily ported to a two-dimensional space for touchscreens. One year later, the first commercial PC with a touchscreen (Hewlett Packard HP-150[1]) was released. The touchscreen is based on infrared sensing but was not well perceived at that time as graphical user interfaces were not widely used. In 1984, Bob Boie presented the first transparent multi-touch screen which used a transparent capacitive array of touch sensors on top of a CRT screen. Similarly, Lee *et al.* [138] developed a touch tablet in 1985 that can sense an arbitrary number of simultaneous touch inputs based on capacitive sensing. Using the compression of the overlaying insulator, the tablet is further capable of sensing the touch pressure. Recent iPhones incorporate this input modality under the name *Force Touch*.

In 1993, the Simon Personal Communicator from IBM and BellSouth (see Figure 2.2) was released, which was the first mobile phone with a touchscreen. Its resistive touchscreen enabled features such as e-mail clients, a calendar, address book, a calculator, and a pen-based sketchpad. In the same year, Apple Computer released the MessagePad 100, a personal digital assistant (PDA) that can be controlled with a stylus but without a call functionality. The success of PDAs continued with the Palm Pilot by Palm Computing as the handwriting recognition worked better for the users. However, in contrast to smartphones nowadays, all these devices require the use of a stylus.

In 1999, FingerWorks, Inc. released consumer products such as the TouchStream and the iGesture Pad that can be operated with finger inputs and ge-

[1] `http://www.hp.com/hpinfo/abouthp/histnfacts/museum/personalsystems/0031/`

Figure 2.2: Simon Personal Communicator, the first smartphone with a touchscreen by IBM and BellSouth. Image taken from arstechnica[1].

stures. The company was eventually acquired by Apple Inc. to contribute to the development of the iPhone's touchscreen and the Apple's Multi-Touch trackpad. Based on the work by Jun Rekimoto [194], Sony introduced a first flat input surface in 2002 that provides two-dimensional images of the changes in the electric field. This technology is known as mutual capacitive sensing and the electric field changes represent low-resolution shapes of conductive objects touching the sensor. In contrast to camera-based approaches, all elements are integrated into a flat touch panel which enables the integration into mobile devices. Touchscreens incorporated in smartphones nowadays are based on this technology.

In the subsequent years, new touch-based technologies were introduced but these are not employed on smartphones due to space constraints. For example, Jeff Han introduced multitouch sensing through frustrated total internal reflection (FTIR) which is based on infrared (IR) LEDs and an IR camera below the touch surface to sense touch input. This enables building high-resolution touchscreens and is less expensive than other technologies. In 2008, the Microsoft Surface 1.0, a table-based touchscreen, was released that integrated a PC and five near-infrared cameras to sense fingers and objects placed on the display. Three years later, the second version of the Microsoft Surface (now called Microsoft PixelSense)

was released that is based on Samsung's SUR40 technology. This technology represents a 40-inch interactive touch display in which pixels can also sense objects above it. This enables to build a less bulky tabletop without cameras below the display and generates a $960 \times 540\,px$ touch image that can be used for object tracking.

2.1.2 Capacitive Touch Sensing

Since the invention of the first touchscreen, a wide range of further touch sensing technologies have been presented. While many of these approaches provide a higher touch sensing resolution and expressiveness compared to the earlier invented capacitive and resistive touchscreens, they are less suitable for mobile devices due to their immobile setup. Amongst others, these technologies include frustrated total internal reflection [77], surface acoustic waves [142], camera-based touch sensing (*e.g.* RGB [225], depth [252]), infrared touch sensing [2], and inductive touch sensing [43].

Due to their compact size, robustness, and responsiveness, capacitive touchscreens are widely used in mobile devices nowadays. In particular, mobile devices use projected capacitive touchscreens which sense touches with a higher resolution than surface capacitance which is often used on larger surfaces with four electrodes at each corner. Figure 2.3 sketches the functional principle of a mutual capacitive touchscreen. Mutual capacitance is one of the two types of the projected capacitance principle and is commonly used in recent mobile devices [15]. The touch sensor itself consists of three layers; an *electrode pattern layer* in the middle which is responsible for the actual touch sensing and two *protective layers*. The touch sensor with all of its layers is transparent and placed on top of the display unit such as a liquid crystal display (LCD). The *electrode pattern layer* is connected to a touch controller and consists of conductive wires made out of indium tin oxide (ITO) which is transparent and sketched on the bottom left of Figure 2.3.

The controller measures the change of coupling capacitance between two orthogonal electrodes, *i.e.* intersections of row and column pairs [50]. These measurements result in a low-resolution finger imprint which is shown on the bottom right of Figure 2.3 and referred to as a *capacitive image* [73, 99, 136, 156].

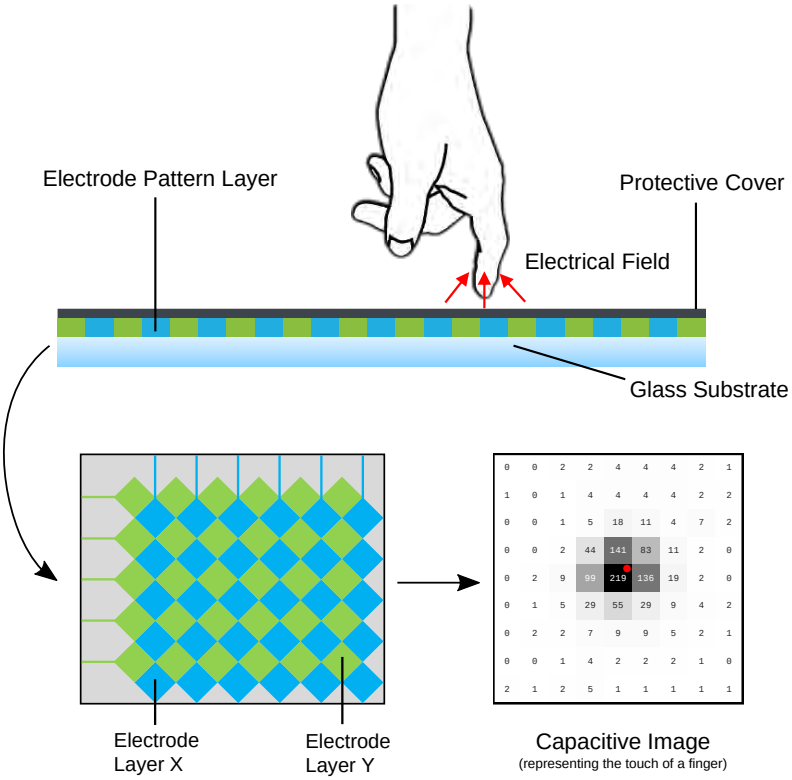


Figure 2.3: Components of a mutual capacitive touchscreen and the resulting capacitive image. Figure adapted and extended based on `http://www.eizo.com/library/basics/basic_understanding_of_touch_panel/`.

Capacitive touchscreens of commodity smartphones comprise around 400 to 600 electrodes (*e.g.*, 15×27 electrodes with each being $4.1 \times 4.1\,mm$ on an LG Nexus 5). The touch controller translates the measurements into a 2D coordinate which is then provided to the operating system (indicated as a red dot in the Figure).

While touch interaction on recent mobile devices is based solely on the 2D coordinate of a touch (*i.e.* the red dot), the remaining information about a touch

is omitted. In this thesis, we present a number of approaches which uses the capacitive images of commodity mutual capacitive touchscreens in mobile devices to infer the source of a touch such as different fingers and hand parts.

2.2 Related Work

Related work presented a wide range of novel interaction techniques to extend the touch input vocabulary on mobile devices. Following the structure of this thesis, we first describe the ergonomics and physical limitations of the hand for interaction with mobile devices. Secondly, we describe interaction methods that improve and extend the interaction with a touchscreen (on the front side) on mobile devices. Lastly, we go one step further and review related work that presents novel interaction methods based on touch input beyond the front touchscreen (*e.g.*, the back and edges of a device).

2.2.1 Hand Ergonomics for Mobile Touch Interaction

In contrast to stationary input devices such as a hardware keyboard and mouse, users usually hold and interact with mobile devices simultaneously. This poses a wide range of challenges. When using a smartphone in the prevalent single-handed grip [54, 109, 110, 176], the same hand is used for holding and interacting with the device. This limits the fingers' range and generates unintended inputs due to the continuous contact with the device. In the following, we review previous work on ergonomics of the hand when holding a smartphone and supportive finger movements which users perform during interaction.

Placement, Movement, and Range of Fingers

To inform the design of novel interaction methods on mobile devices, an understanding of finger placement, movement and their ranges is vital. A wide range of

heuristics have been proposed by designers over the years[1,2,3,4,5]. Previous work further investigated the range of the thumb to inform the design of mobile user interfaces [20]. Since BoD and edge input became more relevant in recent years, all other fingers need to be investigated to inform the design of fully hand-and-finger-aware interaction methods. While previous work showed where fingers are typically placed when holding a smartphone [276], there is no work that studied the areas reachable by all fingers on mobile devices. This thesis contributes to this research area by studying finger placements, ranges and reachable areas of all fingers on mobile devices.

An important basis to inform the placement of on-screen interaction elements and on-device input controls is the analysis of areas on the device that can be reached by the fingers. Bergstrom-Lehtovirta and Oulasvirta [20] modeled the thumb's range on smartphones to inform the placement of user interface elements for one-handed interaction. To predict the thumb's range, the model mainly involves the user's hand size and the position of the index finger which is assumed to be straight (adducted). For the predicted range of the thumb, they introduced the term *functional area* which is adapted from earlier work in kinesiology and biomechanics. In these fields, possible postures and movements of the hand are called *functional space* [118]. Thumb behavior was further investigated by Trudeau *et al.* [231] who modeled the motor performance in different flexion states. Park *et al.* [189] described the impact of touch key sizes on the thumb's touch accuracy while Xiong *et al.* [268] found that the thumb develops fatigue rapidly when tapping on smaller targets.

Besides the thumb, previous work investigated the index finger during smartphone interaction. Yoo *et al.* [276] conducted a qualitative study to determine the comfortable zone of the index finger on the back of the device. This was done without moving the finger and by asking users during the study. From a biomechanical perspective, Lee *et al.* [139] investigated the practicality of different strokes

[1] https://www.uxmatters.com/mt/archives/2013/02/how-do-users-really-hold-mobile-devices.php

[2] http://blog.usabilla.com/designing-thumbs-thumb-zone/

[3] http://scotthurff.com/posts/facebook-paper-gestures

[4] https://www.smashingmagazine.com/2016/09/the-thumb-zone-designing-for-mobile-users/

[5] https://medium.com/@konsav/-55aba8ed3859

for BoD interaction. Similarly, prior work found that using the index finger for target selection on the BoD leads to a lower error rate than using the thumb for direct touch [143, 256]. Wobbrock *et al.* [256] showed that both the thumb on the front and index finger on the BoD perform similarly well in a Fitts' law task. Wolf *et al.* [260] found that BoD gestures are performed significantly different than front gestures. Corsten *et al.* [40, 41] used BoD landmarks and showed that the rear position of the index finger could be accurately transferred to the thumb by pinching both fingers.

Since different grips can be used as an input modality [254], a wide range of prior work sought an understanding of how users hold the phone while using it. Eardly *et al*. [53, 54] explored hand grip changes during smartphone interaction to propose use cases for adaptive user interfaces. They showed that the device size and target distance affects how much users tilt and rotate the device to reach targets on the touchscreen. Moreover, they investigated the effect of body posture (*e.g.*, while standing, sitting, and lying down) on the hand grip, and showed that most grip movements were done while lying down followed by sitting and finally standing [52].

Previous work in biomechanics looked into different properties of the hand. Napier *et al.* [175] investigated two movement patterns for grasping objects which they call *precision grip* and *power grip*. People holding objects with the *power grip* use their partly flexed fingers and the palm to apply pressure on an object. Sancho-Bru *et al.* [205] developed a 3D biomechanical hand model for *power grips* and used it to simulate grasps on a cylinder. However, as smartphones are not necessarily held in a power grip, this model cannot be applied to smartphone interaction. Kuo *et al.* [118] investigated the functional workspace of the thumb by tracking unconstrained motion. This is the space on the hand which is reachable by the thumb. Brook *et al.* [26] introduced a biomechanical model of index finger dynamics which enables the simulation of pinch and rotation movements. As holding a smartphone and interacting with the touchscreen introduces additional constraints to all fingers, these results cannot be applied to model the hand grip and ergonomics.

Supportive Finger Movements

Although users intend to move only the thumb to perform single-handed input on a front touchscreen, they unconsciously perform a wide range of further "*supportive*" movements. These movements maintain the balance and grip on the device, increase the reachability of the thumb on the display (*e.g.*, through tilting [34] and grip shifts [53, 54]), or are unavoidable due to the limited movement independence of fingers (*e.g.*, moving one finger also moves other fingers [76]). An important basis to design BoD input controls that take unintended input into account is the analysis of *supportive micro-movements* during common smartphone tasks.

Tilting the device is one type of *supportive micro-movements* which is used to increase the thumb's reachability on the display. Previous work found that users tilt the device towards their thumb to reach farther distanced targets (*e.g.*, at the top left corner) and away from their thumb to reach targets at the bottom right corner [34, 54]. Eardley *et al.* [52–54] referred to all movements which increase the reachability as "grip shifts" and explored them for different device sizes and tasks. Based on video recordings with manually identified key points and accelerometer values, they quantified the number of grip shifts during common smartphone tasks. They found that more grip shifts occurred with increasing device sizes while the amount of tilt and rotation varied with grip types and phone sizes. Moreover, they showed that the body posture (*e.g.*, sitting and standing) affects the device movements, suggesting that device sizes and body postures need to be considered for exploring *supportive micro-movements*. While these findings explain the device movements, no previous work investigated the actual finger movements which could generate unintended input on the device surface.

The limited independence of finger movements causes another type of *supportive micro-movements*. Previous work in biomechanics found that even when asked to move just one finger, humans usually also produce motion in other fingers [76]. The limited independence of the fingers is due to biomechanical interconnections such as connected soft tissues [242] and motor units [207]. Moreover, Trudeau *et al.* [231] found that the thumb's motor performance varies by the direction and device size during single-handed smartphone use while the motor performance is generally greater for two-handed grips [230]. While

Sancho-Bru [205] presented a biomechanical model of the hand for the power grip [175], an application thereof is not possible for an investigation of *supportive micro-movements* as smartphones are not used solely in a power grip.

One chapter of this thesis contributes to the understanding of *supportive micro-movements* by studying how fingers on the rear move while interacting with the front side.

2.2.2 Novel Touch-Based Interaction Methods

Recent touchscreens are designed to register two-dimensional locations of touches. These locations are provided to the application layer of the operating system to enable interaction with the user interface. Besides the two-dimensional location of touches, a wide range of touch properties are available that can be used to increase the input vocabulary of touch interaction. Well-known examples from recent operating systems are the long-press that leverages the dwell time and gestures that are based on subsequent touch locations. While these additions are beneficial, they require additional execution time. Moreover, the touch input vocabulary is still limited when compared to other input devices such as hardware keyboards or computer mouses. In the following, we describe related work that improves touch input using data from touchscreens and their mobile device.

Extending Touch Interaction on Mobile Touchscreens

Previous work presented a wide range of approaches to extend the touch input vocabulary on mobile touch-based devices. In the following, we describe two common approaches that do not require additional sensors beyond a touchscreen. This includes approaches that are (1) based solely on two-dimensional touch locations available on all touchscreen technologies, and (2) based on the raw data of capacitive touchscreens representing low-resolution fingerprints of touches.

Using the Two-Dimensional Location of Touches Approaches to extend the touch input vocabulary based on only the two-dimensional location of touch inputs can readily be deployed on any touch-based mobile device. Since all touchscreens already provide the two-dimensional location of touches, no additional information and sensors are required.

Single taps are mostly used for selection-based interaction such as selecting an action assigned to a button. Gestures play an important role in making user interfaces more intuitive (*e.g.*, moving objects by dragging them) and in providing shortcuts for a faster access to frequently used functions (*e.g.*, launching applications [190], searching [141]). A gesture can be performed by moving the finger while in contact with the touchscreen. This generates a trajectory of two-dimensional locations of touches that are then interpreted as gestures by the system. Previous work in HCI invested a sheer amount of effort to improve gesture-based interfaces, such as through methodologies for gesture design [237, 238, 255, 256], simple gesture recognizers for fast prototyping purposes [5, 235, 257], improving gesture memorability [173, 277], and through design guidelines for gesture designs [4, 278]. However, gestures have the disadvantage that they require additional execution time as well as enough screen space for the execution. Moreover, a comprehensive set of gestures would lead to conflicts (*e.g.*, unintended activations) and the accuracy of gesture recognizers would decrease due to ambiguity errors.

Previous work proposed a wide range of interaction methods to enrich touch interaction beyond gesture shapes and types. Amongst others, a gesture starting from the device's bezel can be distinguished from a gesture starting on the touchscreen itself. This differentiation was used in previous work to provide shortcuts to the clipboard [200] and to improve one-handed interaction by offering reachability features [112].

Moreover, researchers implemented simple heuristics to use the finger orientation as an input dimension. Roudaut *et al.* [202] presented MicroRolls, a micro-gesture that extends the touch input vocabulary by rolling (*i.e.*, changing pitch and roll angle of the finger) the finger on the touchscreen. Since touchscreens translate touch contact areas to two-dimensional locations based on the area's centroid [23, 98, 202], a trajectory of two-dimensional locations is generated through the changing contact area induced by finger rolling. MicroRolls uses this trajectory to recognize rolling movements with accuracies of over 95 %. However, this interaction techniques cannot be used during a drag action since the

segmentation of the gesture requires down and up events. Thus, Bonnet *et al.* [23] presented ThumbRock which improves MicroRolls by additionally using the size of the contact area as reported by Apple iOS.

Using the Raw Data of Capacitive Touchscreens Nowadays, the majority of touchscreens incorporated in mobile devices are based on mutual capacitive sensing. Taking the measurements of all electrodes of the touchscreen, a two-dimensional image (referred to as *capacitive image* [73, 99, 136, 156]) can be retrieved as shown in Section 2.1.2. Previous work predominantly used an LG Nexus 5 since its touch controller (Synaptics ClearPad 3350) provides a debugging bridge to access the 8-bit capacitive images with a resolution of $27 \times 15\,px$ at $6.24\,ppi$. While capacitive images can be used to recognize body parts for authentication purposes [73, 99], previous work also used the resulting area for interaction methods. Amongst others, Oakley *et al.* [182] used the area of touches on smartwatches to provide shortcuts to pre-defined functions. Similarly, Boring *et al.* [24] used the size of the contact area to enable one-handed zooming and panning.

To extend the touch input performed with fingers, researchers developed machine learning models that infer additional properties based on the capacitive images. Amongst others, machine learning models can be used to estimate the pitch[1] and yaw[2] angle of a finger touching the display [156, 265]. In contrast to the approach on the tabletop [244], machine learning was necessary as no high-resolution contact area is available. Moreover, Gil *et al.* [63] used basic machine learning techniques to identify fingers touching the display. However, they showed that a usable accuracy can only be achieved with exaggerated poses on smartwatches so that each finger touched with a distinct angle. Recent Huawei devices incorporate *KnuckleSense*, an additional input modality that differentiates between touches made by fingers and knuckles. This technology is based on FingerSense, a proprietary technology by Qeexo[3] of which no technical details are publicly available.

[1] Pitch angle: Angle between the finger and the horizontal touch surface.
[2] Yaw angle: Angle between the finger and the vertical axis.
[3] `http://qeexo.com/fingersense/`

Extending Touch Interaction through Additional Sensors

Previous work and smartphone manufacturers used additional built-in sensors to augment touch input. Amongst others, this includes sensors to measure the applied force, microphone recordings, inertial measurement units (IMUs), and pre-touch sensing. Moreover, we give an overview of external sensors that were used in previous work to extend touch input.

Force and Pressure Pressure input offers an additional input dimension for mobile touch interaction. Since interaction can be performed without moving the finger, this input dimension benefits user interfaces on small displays and situations in which finger movements are not desirable. Using the force applied on the touchscreen of a mobile device was first used by Miyaki and Rekimoto [167] to extend the touch input vocabulary. Based on force sensitive resistors between the device and a back cover, they measured the changing pressure levels to prototype one-handed zooming on mobile devices. Stewart *et al.* [223] investigated the characteristics of pressure input on mobile devices and found that a linear mapping of force to value worked the best for users. Researchers further used the shear force, the force tangential to the display's surface, to extend pressure input. Amongst others, Harrison and Hudson [80] developed a touchscreen prototype that uses the shear force for interaction while Heo and Lee [90] augment touch gestures by sensing normal and tangential forces on a touchscreen. Beyond the touchscreen, force can also be used for twisting the device as an input technique [68, 69, 119].

With the iPhone 6s, Apple introduced the pressure input dimension under the name *Force Touch*. Based on force sensors below the touchscreen or a series of electrodes on the screen curvature (Apple Watch), they used the additional input dimension to enable users to perform secondary actions such as opening a context menu or peeking into files. To estimate the force of a touch without additional sensors, Heo and Lee [91] used the built-in accelerometer and position data of the touchscreen.

Acoustics The sound resulting from an object's impact on the touchscreen can be used to differentiate between the source of input. By attaching a medical stethoscope to the back of a smartphone, Harrison *et al.* [82] showed the feasibility

of differentiating between different parts of the finger (*e.g.*, pad, tip, nail, or knuckle) as well as objects (*e.g.*, stylus). Lopes *et al.* [145] used a similar approach and augmented touch interaction based on a contact microphone to sense vibrations. With this, they showed that different hand placements on the touch surface (e.g., tap with a finger tip, knock, slap with the flat hand, and a punch) can be reliably recognized. Similarly, Paradiso *et al.* [188] used four contact piezoelectric pickups at the corners of a window to differentiate between taps and knocks.

In general, approaches based on acoustic sensing have shown to be reliable to identify the source of touch. However, since microphones are required to continuously capture the acoustics, these approaches are prone to errors in noisy situations. Thus, they are not suitable for interaction on mobile devices such as smartphones and tablets.

Physical Device Movement A wide range of previous work combined touch input with the built-in accelerometer of mobile devices. Hinckley *et al.* [92] introduced the terminology of *touch-enhanced motion techniques* which combine information of a touch and explicit device movements sensed by the IMU. For example, a touch and a subsequent tilt sensed by the accelerometer can be used to implement one-handed zooming while holding an item on the touchscreen followed by shaking the device can be used to offer a shortcut to delete files. Similar gestures were explored especially for interaction with wall displays using a mobile phone. Hassan *et al.* [86] introduced the *Chucking* gesture in which users tap and hold an icon on the touchscreen, followed by a toss measured by the accelerometer to transfer the file to the wall display. To transfer items between public displays using a mobile phone, Boring *et al.* [25] proposed a similar gesture in which users hold an object on the touchscreen and move the mobile devices between displays. Researchers also used the built-in accelerometer to enhance text entry on mobile devices. This includes the use of the device orientation to resolve ambiguity on a T9 keyboard [249] and the improvement of one-handed gestural text input on large mobile devices [273].

In contrast, *motion-enhanced touch techniques* combine touch input and the implicit changes of the accelerometer values to infer touch properties. For

example, a soft tap can be differentiated from a hard tap through the impact of the touch. Going one step further, Seipp and Devlin [213] used touch position and accelerometer values to develop a classifier that determines whether users are using the device in a one-handed grip with the thumb or in a two-handed grip with the index finger. With this, they achieved an accuracy of 82.6 %. Similarly, Goel *et al.* [66] used the touch input and device rotation to infer the hand posture (i.e., left/right thumb, index finger) with an accuracy of 87 %. By attaching a wearable IMU to the user's wrist, Wilkinson *et al.* [251] inferred the roll and pitch angle of the finger, and the force of touches described by the acceleration data.

Proximity Touch Sensing Marquardt *et al.* [150] proposed the *continuous interaction space*, which was among the first models that describe the continuity between hover and on-screen touches. They proposed a number of use cases that enables users to combine touch and hover gestures anywhere in the space and naturally move between them. Amongst others, this includes raycasting gestures to extend reachability, receiving hints through hovering over UI elements [37], and avoiding occlusion by continuing direct touch actions in the space above. Spindler *et al.* [222] further proposed to divide the interaction above the tabletop into multiple layers while Grossman [71] explored hover interaction for 3D interaction.

Hover information can also be used to predict future touch locations. Xia *et al.* [264] developed a prediction model to reduce the touch latency of up to 128 *ms*. To avoid the fat-finger problem, Yang *et al.* [270] used a touch prediction to expand the target as the finger approaches. Similarly, Hinckley *et al.* [93] explored hover interaction on mobile devices and proposed to blend in or hide UI components depending on whether a finger is approaching or withdrawn (*e.g.*, play button in a video player appears when the finger is approaching). Since a finger can also be sensed above the display, Rogers *et al.* [198] developed a model for estimating the finger orientation based on sensing the whole finger on and above a touchscreen.

Previous work presented different approaches to enable proximity touch sensing. The SmartSkin prototype presented by Rekimoto [194] calculates the distance between hand and surface by using capacitive sensing and a mesh-shaped

antenna. Annett et al. [3] presented Medusa which is a multi-touch tabletop with 138 proximity sensors to detect users around and above the touchscreen. On the commercial side, devices such as the Samsung Galaxy S4 and the Sony Xperia Sola combine mutual capacitance (for multi-touch sensing on the touchscreen), and self-capacitance (generates a stronger signal but only senses a single finger) to enable hover interaction[1].

Fiducial Markers and Capacitive Coupling A large body of work coupled external sensors and devices with touchscreens to extend the touch input vocabulary. The focus lies especially on identifying the object touching the display, such as different fingers, users, and items.

A common approach to identify objects on the touchscreen is to use fiducial markers. These markers assign a unique ID to an object through a uniquely patterned tag in the form of stickers [108, 195], NFC tags [240, 241], RFID tags [183], unique shapes [85], and through rigid bodies of conductive areas attached to objects ("capacitance tags") [194]. While these approaches are only suitable for objects due to the attachment of tags, previous work investigated the use of capacitive coupling (*i.e.*, placing an electrode between object and the ground to change the electric field measured by the touchscreen) to reliably identify users [243] and authenticate them with each touch [100]. Similarly, DiamondTouch [47] identifies users based on an electric connection to the chair they are sitting on while Harrison *et al.* [81] used Swept Frequency Capacitive Sensing (SFCS) which measures the impedance of a user to the environment across a range of AC frequencies. Using the same technology, Sato *et al.* [206] turned conductive objects to touch-sensitive surfaces that can differentiate between different grips (*e.g.*, touch, pinch, and grasp on a door knob).

Active Sensors To identify different fingers on the display, previous work used a wide range of different sensors. Approaches that achieved high accuracies include the use of IR sensors [74, 75] and vibration sensors [152] mounted on different fingers. Further approaches include electromyography [19], gloves [149]

[1] `https://www.theverge.com/2012/3/14/2871193/sony-xperia-sola-floating-touch-hover-event-screen-technology`

and RFID tags attached to the fingernail [239]. To avoid instrumenting users with sensors, previous work also used a combination of cameras attached to a mobile device and computer vision to identify fingers [245, 284]. For example, Zheng *et al.* [284] used the built-in webcam of laptops to identify fingers and hands on the keyboard. Using depth cameras such as the Microsoft Kinect provides additional depth information for finger identification. Amongst others, these were used by Murugappan [172] and Wilson [252] to implement touch sensors. The Leap Motion[1] is a sensor device that uses proprietary algorithms to provide a hand model with an average accuracy of $0.7\,mm$ [248]. Colley and Häkkilä [38] used a Leap Motion next to a smartphone to evaluate finger-aware interaction. While these are all promising approaches, they are not yet integrated into mass-market devices since wearable sensors are limiting the mobility while sensors attached to the device (e.g., cameras) are increasing the device size.

Extending Touch Interaction on Tabletops

Previous work presented a wide range of novel interaction methods based on images of touches provided by touchscreens. Researchers predominantly focused on tabletops that provide high-resolution images of touches [8, 56, 62] through technologies such as infrared cameras below the touch surface or frustrated total internal reflection [77]. The Microsoft PixelSense is a common example and provide high-resolution images with a resolution of $960 \times 540\,px$ (24 ppi). This enabled a wide range of novel interaction methods including the development of widgets triggered by hand contact postures [154], using the forearm to access menus [114], using the contact shape to extend touch input [18, 30], and gestures imitating the use of common physical tools (*e.g.*, whiteboard eraser, eraser, camera, magnifying glass) to leverage familiarity [83]. The latter was commercialized by Qeexo as TouchTools[2].

Based on a rear-projected multi-touch table with a large fiber optic plate as the screen, Holz and Baudisch developed a touchscreen that senses fingerprints for authentication [96]. This is possible due to a diffuse light transmission while the touchscreen has a specular light reflection. Other approaches for user

[1] `https://www.leapmotion.com/`
[2] `http://qeexo.com/touchtools/`

Type \ Position	Front side	Back side	Top side	Bottom side	Left side	Right side
Touch	Fingerprint scanner Secondary screen [j] Hardware buttons (*e.g.*, back, home)	Fingerprint scanner BoD Touch [a,j] [16, 46] Heart rate sensor [f] [151] BoD touchscreen [m]	-	-	-	Edged display [b]
Buttons	Hardware keyboard [b] Home/Menu button [c] Back/Recent button [c]	BoD Button [d] Volume button [l]	Power button [e]	-	Volume buttons Bixby button [f]	Power button Volume buttons Shutter button [g]
Slide	-	-	-	-	Silent switch [e]	-
Pressure	Force Touch [272]	-	-	-	Side pressure [h] [59, 221]	
Scrolling	Trackball [i]	LensGesture [266]	-	-	-	Scrolling wheel [b]
Tapping	-	BoD taps [197]	Edge taps [161]			
Misc	Front camera Front speaker Light sensor Distance sensor Notification LED	Back camera Back speaker Torchlight E-ink display [k]	Microphone Audio port USB port [g]	Microphone Speaker USB port Audio port	-	-

[a] OPPO N1, [b] RIM BlackBerry 8707h, [c] HTC Tattoo, [d] LG G-Series, [e] iPhone 5, [f] Samsung Galaxy S8, [g] Nokia Lumia 840, [h] HTC U11, [i] Nexus One, [j] LG X, [k] YotaPhone 2, [l] Asus Zenfone, [m] Meizu Pro 7.

Table 2.1: Types of interaction controls beyond the touchscreen that are presented in prior work and in recent or past smartphones. While some are not intended for interaction initially (*e.g.*, camera), these sensors could still be used for interaction in the future, *e.g.* [266].

identification on tabletops are based on users' back of the hand captured by a top-mounted camera [193], by their hand geometry [22, 208], their shoes based on a camera below the table [196], through personal devices [1], tagged gloves [148], finger orientations [45, 280], IR light pulses [163, 199], and through capacitive coupling [47, 243].

2.2.3 Interacting with Smartphones Beyond the Touchscreen

Since users are holding the smartphone during the interaction, the touchscreen on the front is not the only surface that could be used for input. Previous work and smartphone manufacturers presented a wide range of input mechanisms beyond the touchscreen on the device surface. While the power and volume buttons are integral parts of a smartphone nowadays, we describe further input mechanisms in the following.

On-Device Input Controls

Previous work and manufacturers presented a broad range of input controls for smartphones of which we provide an overview in Table 2.1. We categorized them by their location on the device, and by the expected type of input.

Current smartphones such as the iPhone 7 and Samsung Galaxy S8 incorporate fingerprint sensors below the touchscreen or on the back of the device. These are mainly used for authentication purposes but can also recognize directional swipes that act as shortcuts for functions such as switching or launching applications. Previous work envisioned different functions that can be triggered using a fingerprint sensor [185]. Due to a small number of devices that support any form of interaction on the rear, researchers presented different ways to use built-in sensors for enabling BoD interaction, including the accelerometer [140, 197] to recognize taps and the back camera to enable swipe gestures [266]. Previous work also presented a number of smartphone prototypes that enable touch input on the whole device surface, including the front, back and the edges [127, 132, 168]. This enables a wide range of use cases which includes touch-based authentication on the rear side to prevent shoulder surfing [46], improving the reachability during one-handed smartphone interaction [133], 3D object manipulation [10, 214], performing user-defined gesture input [215] and addressing the fat-finger problem [16]. Recently, Corsten *et al.* [39] extended BoD touch input with a pressure modality by attaching two iPhones back-to-back.

Before HTC recently introduced Edge Sense, pressure as an input modality on the sides of the device have been studied in previous work [59, 95, 221, 253] to activate pre-defined functions. Legacy devices such as the Nexus One and HTC Desire S provide mechanical or optical trackballs below the display for selecting items as this is difficult on small displays due to the fat-finger problem [16]. As screens were getting larger, trackballs became redundant and were removed. Similarly, legacy BlackBerry devices incorporated a scrolling wheel on the right side to enable scrolling.

For years, smartphones featured a number of button controls. Amongst others, this includes a power button, the volume buttons, as well as hardware buttons such as the back, home and recent buttons on Android devices. As a shortcut to change the silent state, recent devices such as the iPhone 7 and OnePlus 5

feature a hardware switch to immediately mute or unmute the device. Moreover, the Samsung Galaxy S8 introduced an additional button on the left side of the device as a shortcut to the device assistant while other devices incorporate a dedicated camera button. Since a large number of hardware buttons clutter the device, previous work used the built-in accelerometer to detect taps on the side of the device [161].

Back-of-Device Prototyping Approaches

The simplest and most common approach for BoD interaction is to attach two smartphones back-to-back [39, 46, 214, 261, 262]. However, this approach increases the device thickness which negatively affects the hand grip and interaction [133, 226]. This is detrimental for studies that observe the hand behavior during BoD interaction, and could lead to muscle strain. To avoid altering the device's form factor, researchers built custom devices that resemble smartphones [10, 33, 228]. However, these approaches mostly lack the support of an established operating system so that integrating novel interactions into common applications becomes tough. As a middle ground, researchers use small additional sensors that barely change the device's form factor. These include 3D-printed back cover replacements to attach a resistive touch panel [133], and custom flexible PCBs with 24 [168, 169] and 64 [33] square electrodes. However, neither the panel size nor the resolution is sufficient to enable precise finger-aware interactions such as gestures and absolute input on par with state-of-the-art touchscreens.

Beyond capacitive sensing, researchers proposed the use of inaudible sound signals [174, 203, 246], high-frequency AC signals [283], electric field tomography [282], conductive ink sensors [67], the smartphone's camera [263, 266], and other built-in sensors such as IMUs and microphones [70, 201, 279]. While these approaches do not increase device thickness substantially, their raw data lack details for precise interactions or inferring the touching finger or hand part. While using flexible PCBs as presented in previous work is a promising approach, the resolution is not sufficient. Further, previous work used proprietary technologies so that other researchers cannot reproduce the prototype to investigate interactions on such devices. There is no previous work that presents a reproducible (*i.e.*, uses commodity hardware) full-touch smartphone prototype.

2.3 Summary

In this chapter, we discussed the background and previous work on mobile touch interaction. We started this chapter with the history of touch interaction and background of capacitive touch sensing, which forms the foundation of the technical parts of this work. Moreover, we reviewed previous work with a focus on extending mobile touch interaction by hand-and-finger-awareness.

Following the structure of this thesis, we first reviewed previous work on hand ergonomics for mobile touch interaction. Previous work investigated the range of the thumb for single-handed touch interaction to inform the design of touch-based user interfaces. However, there is neither work that does the same for all other fingers nor the area in which fingers can move without a grip change. An understanding thereof is vital to inform the design of fully hand-and-finger-aware input methods especially on fully touch sensitive smartphones. We investigate the areas which can be reached by all fingers without a grip change and their maximum range by addressing RQ1. In addition to the reachability aspect, the fingers on the back move unintentionally, amongst others, to maintain a stable grip [52, 54], increase the thumb's range [34, 54], or as a consequence of the limited independence between the finger movements [76]. These movements cause unintended inputs on fully touch sensitive smartphones which frustrate users and renders all BoD input techniques ineffective. Ideally, BoD input controls need to be placed so that they are reachable without a grip change but also in a way which minimizes unintended input. This requires an investigation of *supportive micro-movements*, their properties, as well as the areas in which they occur. We address this with RQ2.

In the second part of the related work, we reviewed different approaches to extend the touch input vocabulary. A wide range of approaches use different sensors to infer additional properties of a touch. For example, this includes the finger orientation, pressure, shear force, size of the touch area, as well as identifying the finger or part of the hand which performed the touch. However, the presented approaches have practical disadvantages which affect the usability, convenience, and mobility. Input techniques which infer additional properties of a touch (*e.g.*, finger orientation, pressure, or size of touch area) extends the

input vocabulary and its expressiveness. However, they also pose limitations since specific finger postures may now trigger unwanted actions. In contrast, input techniques which differentiate between the source of input (*e.g.*, identifying individual fingers and hand parts) do not interfere with the main finger for interaction and thus do not have these limitations. While previous work [63, 82] presented approaches to identify fingers and hand parts, they are either based on immobile and inconvenient technologies (e.g., wearable sensors [152], cameras [38, 284], stethoscope [82]) or are not accurate enough for interaction [63]. We address RQ3 by presenting a novel deep learning based approach which uses the raw data of a capacitive touchscreen on commodity smartphones to differentiate between touches of different sources.

In the third part of the related work, we presented an overview of already existing on-device input controls either incorporated on commodity smartphones or presented in related work. Since a part of the presented input techniques in this thesis is based on a fully touch sensitive smartphone, we further reviewed previous work which presented approaches to implement BoD input. These approaches are mostly based on stacking devices or attaching additional touchpads, which increases the size of the device significantly and thus affects the usual hand grip and the usability. Addressing RQ4, we present a fully touch sensitive smartphone prototype with the size of a normal smartphone and which identifies the finger performing input on the whole device surface.

From previous work, we identified main challenges of recent touch input which includes the lack of expressiveness and shortcuts, the limited reachability, and the lack of input precision which is also known as the fat-finger problem [217]. Addressing RQ5, we interview experienced interaction designers to elicit novel ways to solve the challenges of touch input based on fully touch sensitive smartphones and the concept of hand-and-finger-awareness. With text editing being inconvenient on mobile devices due to limited input capabilities and precision, RQ6 focuses on this specific use case with a series of studies representing all steps of the UCDDL to design and implement text editing shortcuts on fully touch sensitive smartphones.

3

Hand Ergonomics for Mobile Touch Interaction

By combining input and output in a single interface, touchscreens enable users to interact with devices which are merely larger than the touchscreen itself. Although they already existed since 1965 [105], it was not until the release of the iPhone in 2007 that touchscreens became the main interfaces of mobile devices. Since then, numerous design refinements and a large body of research on understanding the human factors of mobile interaction have been carried out to inform the design of touch-based interfaces which became intuitive and usable as they are nowadays.

Similarly, the first mobile phone prototype with BoD input was presented in 2003 [94] with a wave of follow-up work being published in the years after (*e.g.* [10, 16, 39, 46, 133, 168, 169, 197, 224, 250, 269]). Despite a wide range of exciting use cases and the technology for simple BoD input being readily available, the concept of enabling all other fingers to interact on the back side is still not widely adopted yet. Most of the previous work on BoD input focused on novel use cases and interaction techniques while the human factors such as ergonomic constraints and unintended inputs are not investigated yet.

Without understanding how hands behave while performing BoD input, we cannot design usable BoD input. Since users prefer to use mobile devices in a single-handed grip [54, 109, 110, 176], the same hand is used for holding and interacting with the device. Although input on the back and edges more than doubles the surface for input, the effectively usable surface is constrained by how far fingers can reach while holding a device. Moreover, since finger movements are shown to have limited independence [76] while being occupied with maintaining a stable grip, another challenge is to understand and avoid the unintended input that they generate while the thumb performs input on the front. With the research presented in this chapter, we contribute with design implications which helps designing usable BoD input in the prevalent single-handed grip.

Parts of this chapter are based on the following publication:

H. V. Le, S. Mayer, P. Bader, and N. Henze. "Fingers' Range and Comfortable Area for One-Handed Smartphone Interaction Beyond the Touchscreen." In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI'18. New York, NY, USA: ACM, 2018. DOI: 10.1145/3173574.3173605[a]

H. V. Le, S. Mayer, and N. Henze. "Investigating Unintended Inputs for One-Handed Touch Interaction Beyond the Touchscreen." In: *Currently under review.*

[a] Video Preview: https://www.youtube.com/watch?v=Kzp-s07mIbo

3.1 Interaction Beyond the Touchscreen

Almost every smartphone incorporates additional input controls beyond the touchscreen. Physical buttons provide shortcuts to change the device volume and power state while fingerprint scanners enable authentication as well as simple BoD gestures. These input controls extend touchscreen input and are accessible even without looking at the device. Recent smartphones offer an increasing number of further input controls such as dedicated buttons for the device assistant, silent switches, and even secondary touchscreens on the rear (*e.g.*, Meizu Pro 7 and YotaPhone). Previous work explored further promising use cases and showed

that BoD input can provide additional shortcuts [185, 197, 212, 215], increase the thumb's reachability on the touchscreen [133, 276], and even solve touch input's limitations such as the fat-finger problem [16, 250].

Additional input controls beyond the touchscreen clearly improve mobile interaction but also pose new challenges. When using a smartphone in the prevalent single-handed grip [54, 109, 110, 176], the same hand is used for holding and interacting with the device. This limits the fingers' range and generates unintended inputs due to the continuous contact with the device.

3.1.1 Reachability of Input Controls

With the increasing size of recent smartphones, it becomes difficult to reach input controls distributed across the whole device surface in the prevalent single-handed grip. Often, users are required to stretch the thumb in an uncomfortable way or change the grip in which they hold the device. These movements are detrimental as they not only cause muscle strains but also lead to dropping the device. Researchers and manufacturers explored a number of approaches to replace thumb stretching and grip changes with more subtle actions to increase reachability. For example, Apple introduced the *Reachability* feature which enables users to shift down the screen content by half its size with a double tap on the home button. Similarly, researchers presented techniques based on BoD gestures [133, 271] and shortcuts [34, 112] to move the screen content to a more reachable position. However, these approaches require additional actions from the users which also affects the usability and work only for touchscreens.

Another branch of research focused on understanding the ergonomic constraints of the hand while holding a smartphone. The findings then enable to design user interfaces which preemptively avoid grip changes. An important basis to design input controls for one-handed interaction is the analysis of finger movements that do not require a grip change. While Bergstrom-Lehtovirta and Oulasvirta [20] modeled the thumb's maximum range, there is neither previous work that does the same for all other fingers nor the area in which fingers can move without a grip change. Yoo *et al.* [276] explored the area in which the index finger can rest comfortably, but on a qualitative basis without moving the finger. However, the movement of fingers provides important implications for the design

of on-device input controls. Specifically, they reveal the areas that users can reach without losing grip stability and the maximum range coverable by fingers without grip changes. Despite their relevance for the design of one-handed interaction, no previous work explored the areas and maximum ranges that all fingers can reach without grip changes using freeform tasks (RQ1).

3.1.2 Unintended Inputs

Since the same hand is used for holding and interacting with the device, the fingers on the back often perform *supportive micro-movements* while the thumb is performing input on the front side. With *supportive micro-movements*, we refer to finger movements on the back and sides which support the thumb in performing input, such as by maintaining a stable grip [52, 54], increasing the thumb's input precision and range [34, 54], or as a consequence of the limited independence between the finger movements [76]. Especially with input controls on the rear or even fully touch sensitive smartphones, *supportive micro-movements* cause unintended inputs which frustrate users and lead to embarrassing mistakes.

Avoiding unintended inputs is vital for the usability of BoD interaction. An important basis to minimize unintended inputs is an understanding of *supportive micro-movements* which occur while holding the device and interacting with it. Previous work analyzed *supportive micro-movements* by quantifying the number of grip shifts on different smartphones through video observations [53, 54] and built-in motion sensors [34, 52, 54]. However, we have no understanding of how fingers move and unintended inputs that they would generate on the rear. To help designers minimizing unintended inputs for BoD interaction, we need to understand the areas in which *supportive micro-movements* occur, as well as the behavior of fingers within these areas on different device sizes and usage scenarios (RQ2). In conjunction with our findings on the reachability of all fingers, this would enable designers to design BoD input controls which consider unintended inputs as well as reachability during single-handed use.

3.2 Study I: Range and Comfortable Area of Fingers

We use a quantitative approach to empirically study the areas that can be reached without changing the hand grip and losing grip stability (*comfortable area*), and the range that can be covered with no grip change and stretched fingers (*maximum range*) while holding the smartphone one-handed in portrait mode. In the following study, participants performed two tasks to explore the *comfortable area* and the *maximum range* on four smartphones with different sizes. We recorded all finger movements using a high-precision motion capture system. Based on the results, we derive four generalizable design implications for the placement of on-device input controls that are suitable for one-handed interaction. These can increase the usability especially in scenarios where one hand is occupied.

3.2.1 Study Design

The study has two independent variables, PHONE and FINGER. For PHONE, we used four smartphones in different sizes (see Table 3.1 and Figure 3.1). For FINGER, we used all five fingers of the right hand. This results in a 4×5 within-subject design. We counterbalanced PHONE using a Balanced Latin square and used a random order for FINGER. For each condition, participants performed two independent tasks to explore the *comfortable area* and to determine the *maximum range*. During these tasks, they were seated in front of the motion capture system (see Figure 3.2b) on a chair without armrests. We did not instruct participants to use specific hand grips as this would influence the participant's usual hand grip and thus the generalizability of the study results.

Device	Abbr.	Height	Width	Depth	Area	%
Samsung Galaxy S3 mini	S3	12.16	6.30	0.99	76.61	100.0
Samsung Galaxy S4	S4	13.70	7.00	0.79	95.90	125.2
OnePlus One	OPO	15.29	7.59	0.89	116.05	151.5
Motorola Nexus 6	N6	15.93	8.30	1.01	132.22	172.6

Table 3.1: Sizes of smartphones (in *cm*) used in both studies. The front and back surface area are shown in in cm^2 (width $\times$ height). The percentage column shows the increase in area starting from the S3.

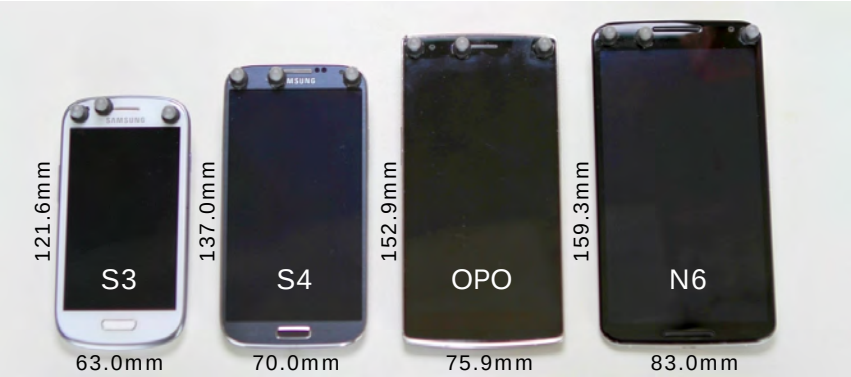


Figure 3.1: Smartphones used in both studies: Samsung Galaxy S3 Mini, Samsung Galaxy S4, OnePlus One, and Motorola Nexus 6.

3.2.2 Apparatus

Table 3.1 shows the four phones that were used. We specifically selected these devices to get a steady increase in device width as this dimension has a noticeable influence on the grip. In the remaining work, we will use the following abbreviations for the devices: *S3*, *S4*, *OPO* and *N6*. The *OPO* and *N6* are representative for recent large flagship smartphones (*e.g.*, Samsung S8 Plus, One Plus 5 or iPhone 7 Plus; on average 154% of the *S3*'s area) while the *S4* and *OPO* are representative for their standard versions (*e.g.*, Samsung S8, OnePlus X, or iPhone 7; on average 126% of the *S3*). The *S3* and *S4* are representative for small devices such as the iPhone SE, LG Nexus 5, or Sony Xperia Compact (on average 109% of the *S3*). While laser-cut device mockups could have been an alternative, we used real devices out-of-the-box to keep the participant's hand grip as realistic as possible. Due to a neglectable difference in device thickness (SD=1.0mm), different device shapes (*e.g.*, edges and corners) should not affect the grip and finger movements as the edges are clamped between fingers and palm.

To record finger motions with sub-millimeter accuracy, we used an *OptiTrack* motion capture system with eight cameras (*OptiTrack Prime 13W* capturing at 240fps). The cameras were firmly mounted to an aluminum profile structure as shown in Figure 3.2b. To enable these infrared cameras to record the finger

(a) Marker Placements **(b) Motion Capture Setup**

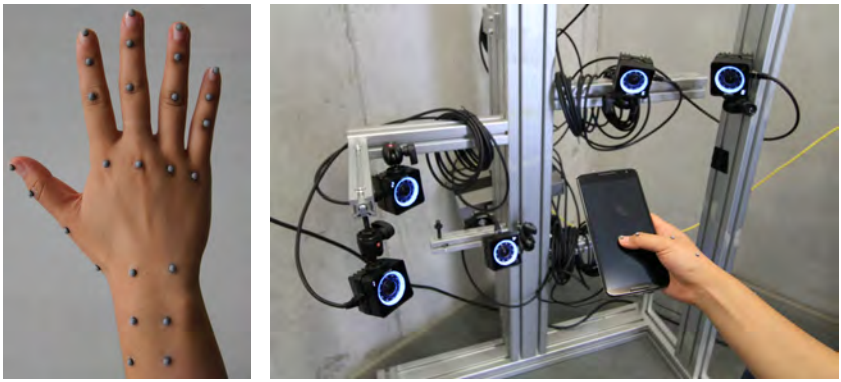


Figure 3.2: Study setup: (a) motion capture system consisting of 8 *OptiTrack Prime 13W* , (b) reflective markers on a participant's hand, and (c) a participant exploring the comfortable area of the thumb on a Nexus 6 in front of the motion capture system.

movements, we attached 26 skin adhesive markers (4 *mm* hemispheres) on all joints of the hand similar to Feit *et al.* [58] as shown in Figure 3.2a. Additionally, we attached four markers on the top part of each smartphone which enables us to track the phones in six degrees of freedom (DoF).

3.2.3 Procedure

After participants signed the consent form, we collected demographic data using a questionnaire and measured their hand size and finger lengths. We then proceeded to attach 26 skin adhesive markers on their right hand to enable motion tracking. We handed out an instruction sheet explaining the procedure of the study and the two tasks which should be performed. The instruction sheet further explains three criteria that participants should fulfill while performing the tasks. This includes (1) holding the device one-handed (2) in portrait mode, and (3) not moving any finger except the one that the experimenter asks to move. We further gave participants a demonstration of the required movements and asked them to do it tentatively to ensure that everything is fully understood.

After handing a smartphone to the participants, we asked them to loosen and move their fingers on the device surface and hold the device as they would usually do afterward. To avoid influencing their usual grip after the experimenter hands out the device, we further asked participants to perform movements as if they would unlock the device with unlocking patterns to start using the device. In the first task, we collected data about the *comfortable area* of each finger. This is the area that one can reach without changing the hand grip and without losing the stable and firm grip through, *e.g.*, overstretching. We instructed participants to freely move the specified finger and cover all areas on the device that they can reach without changing the initial grip, losing grip stability or overstretching fingers to a degree which leads to straining the muscles. We further hinted that different finger flexion degrees should be probed to fill out the explored area and that they should continue exploring beyond the device surface (*e.g.*, beyond the top edge) if fingers can reach it comfortably.

In the second task, we investigated the finger's *maximum range*. This is the range that one can reach with a stretched finger while not changing the initial grip (*i.e.*, not moving any other finger). We instructed participants to keep the specified finger fully extended while performing an arc motion (*i.e.*, abduction and adduction) as far as possible without moving any other finger. Both tasks were repeated for all five fingers whereas the finger order was randomized. We decided to not randomize the task order as exploring the comfortable area involves free (and thus influenceable) movements in contrast to exploring the maximum finger range. While we gave participants 60 seconds to fully explore the comfortable area, the maximum finger range was explored for 30 seconds as there are fewer DoF to explore. The experimenter monitored the markers throughout the study to ensure that only one finger was moving while all others were not. The study including optional breaks took 40 minutes on average.

3.2.4 Participants

We recruited 16 participants (7 female) through our university mailing list. Participants were between 19 and 30 years old ($M = 23.5$, $SD = 3.5$). All participants were right-handed with hand sizes between 163 *mm* and 219 *mm* ($M = 184.1$ *mm*, $SD = 17.1$) measured from the tip of the middle finger to the wrist crease. Our

collected data comprise samples from the 5th and 95th percentile of the anthropometric data reported in prior work [191]. Thus, the sample can be considered as representative. We reimbursed the participants with 10 EUR.

3.2.5 Data Preprocessing

The goal of the preprocessing step is to assign unique identifiers to the markers and convert them from 3D to 2D space (*i.e.*, front side for thumb markers, rear side for all others).

Labeling and Cleaning Data

We labeled all markers using semi-automatic labeling provided by OptiTrack's *Motive:Body* software. We used the *Fragment/Spike* option (Max Spike$= 5\,mm/frame$; Max Gap$= 10 frames$) which followed the trajectory until a gap or a spike in marker movement was found. These settings were chosen to prevent marker swaps in the trajectory. We removed all frames in which the phone's rigid body was not tracked due to technical issues. These issues can occur as each of the four markers of the rigid body need to be captured by at least three cameras to be reconstructed. We further applied a heuristics to detect erroneous rigid body tracking by assuming that the phone was not held in uncommon poses (*e.g.*, up-side-down, flipped). In total, we removed 2.1% of all recorded frames.

Generating 2D Heatmaps

To transform recorded 3D movements onto 2D planes (front and back side), we transformed each hand marker from the global coordinate system into the device's coordinate system and projected them onto the device surfaces. Movements on the device surfaces are represented by heatmaps with a raster size of $1\,mm \times 1\,mm$. Due to a fixed duration and capture rate during the tasks, the number of data points on the heatmaps represents the frequency in which the respective locations were covered by the finger. We validated the transformation by sampling five frames per participant which we manually checked for correctness.

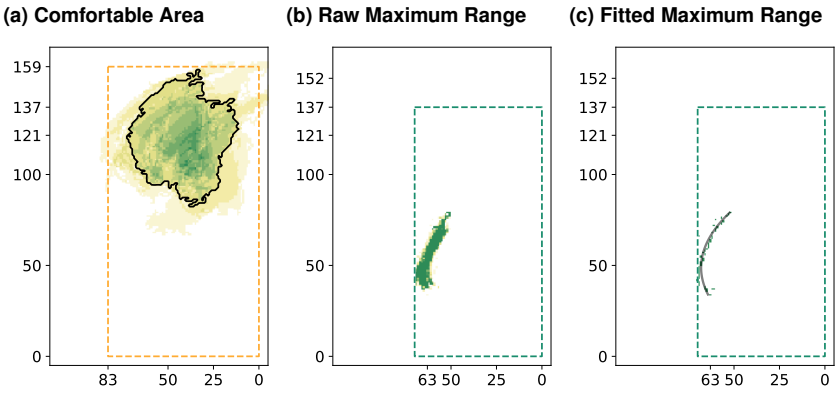


Figure 3.3: (a) Comfortable area of the index finger on an N6. Black contour shows the area explored by at least 25% of participants; (b) raw recording of the maximum range task of an index finger on the S4; (c) maximum range after preprocessing and curve fitting (black curve).

Determining the Comfortable Area

We used the markers placed on the fingertips to determine the comfortable area for interaction. We first filtered noise in each heatmap by removing all data points with a sum less than 10 in a 5×5 neighborhood (*i.e.*, all spots explored less than $41.6 ms$ at $240 fps$). Using dilation and erosion on a binary version of the heatmap, we then filled little gaps within the comfortable area. Since heatmaps are now binary, the results for each participant were added up to retrieve a heatmap representing all explored spots normalized over participants (see Figure 3.3a). To remove spots that are only reachable to a small number of participants due to an outstanding hand size or convenient grip, we removed all spots which are not explored by at least 25% of all participants to exclude outliers.

Determining the Maximum Range

We applied the same noise removal procedure as described above. We then retrieved the farthest data points into each direction starting from the bottom

right corner of the device and omitted all other points. This removes accidental touches or touches with a finger that was not fully stretched (see Figure 3.3b and Figure 3.3c). Using the farthest points, we fitted a quadratic function to describe the finger's maximum range. Bergstrom-Lehtovirta and Oulasvirta [20] showed that the thumb's maximum range can be described by quadratic functions (reported average $R^2 = .958$). We will show that this is also possible for all other fingers with a high R^2. To reproduce their approach, we fitted the same quadratic function $f_{a,h,k}$ to the filtered data also using non-linear regression and a least-squares approach. In contrast to their study, our participants were free to hold the phone in any grip they were used to. As a specific grip could not be assumed, we had to include the rotation of the phone in the hand into the fitting process. We therefore introduced a rotation matrix R_α resulting in the function $g_{a,h,k,\alpha}$ as shown in Equation (3.3):

$$f_{a,h,k}(x) = a(x+h)^2 + k \tag{3.1}$$

$$R_\alpha = \begin{pmatrix} cos(\alpha) & -sin(\alpha) \\ sin(\alpha) & cos(\alpha) \end{pmatrix} \tag{3.2}$$

$$g_{a,h,k,\alpha}(x) = R_\alpha \cdot \begin{pmatrix} x \\ f_{a,h,k}(x) \end{pmatrix} \tag{3.3}$$

The corresponding error function e which we used to find the parameters a, h, k and α is:

$$e_{a,h,k,\alpha}(p) = f_{a,h,k}(r_x) - r_y \text{ with } r = R_\alpha^{-1} \cdot p \tag{3.4}$$

The range of g in which finger movements are restricted in abduction and adduction movement is then obtained from the minimum and maximum value in x direction of the filtered data after rotating by R_α. To finally retrieve the average maximum range of each finger over all participants, we calculated the mean function over x for all $g_{a,h,k,\alpha}(x)$ of each participant.

3.2.6 Results

To facilitate the notation, we will use the abbreviations F_0 to F_4 for the thumb to the little finger respectively. To report values for each finger at once, we use square brackets containing the values starting with F_0 (*e.g.*, [F_0 F_1 F_2 F_3 F_4]). We mapped the origin $(0,0)$ of all figures to the bottom right corner of the smartphone.

Comfortable Area

Figure 3.5 depicts the contour of the comfortable area for all fingers. The color of the contours denotes the device while dashed lines represent the size of the respective device.

Area size Table 3.2 shows the size of the comfortable area for each finger on the four devices. Between the devices, there is a linear growth of the comfortable area with increasing device sizes for the index and the middle finger. We performed a Pearson's correlation test to test for a significant correlation between the device's diagonal length and the size of the comfortable area. We found a significant correlation for the index and the middle finger (r = [−.303 .975 .985 .311 .699], p = [.697 .025 .015 .689 .301]). This correlation can be described as a linear behavior with an average fitness of R^2 = [.09 .95 .97 .10 .49].

Area positions The dots in Figure 3.5 represent the area's centroid position averaged over all participants. Attached whiskers represent the standard deviation. The centroids are gradually shifting towards the upper left edge with increasing

	S3	S4	OPO	N6	Mean	SD
Thumb - F_0	33.6	41.9	35.2	35.0	36.4	3.3
Index Finger - F_1	30.8	37.3	48.9	47.6	41.1	7.5
Middle Finger -F_2	24.6	28.5	35.3	36.7	31.3	5.0
Ring Finger - F_3	15.8	11.7	18.0	22.0	16.9	3.7
Little Finger -F_4	16.9	16.0	20.5	23.7	19.3	3.1
BoD Union (F_1 - F_4)	79.7	88.6	109.6	106.7	96.2	14.4

Table 3.2: Comfortable areas in cm^2 for all fingers on four devices.

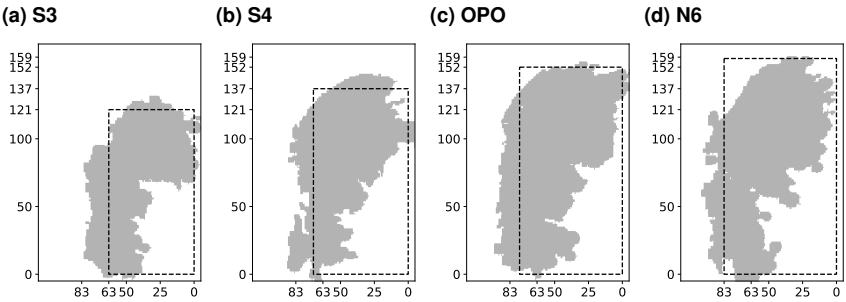


Figure 3.4: Union of comfortable areas of all fingers on the BoD.

sizes of the smartphone. This shift can be described by a linear function with a fitness of R^2 = [.67 .94 .92 .89 .66] for all five fingers. This suggests that the index, middle and ring finger are linearly shifting towards the left edge with increasing device sizes. Pearson's correlation test revealed a correlation between the device's diagonal and a gradual shift of the index, ring and little finger towards the top left corner (r = [.818 .974 .962 .956 .983], p = [.182 .026 .038 .044 .017]).

Union of BoD comfortable areas We show the union of the comfortable areas for the back side in Figure 3.4. Hereby, they show that 68.8 % of the *S3* can be reached without changing the grip or losing stability, while this is the case for 67.3 % for the *S4*, 73.4 % for the *OPO* and 67.7 % for the *N6*.

Maximum Finger Range

The bold quadratic curves in Figure 3.6 describe the maximum range reachable by each finger averaged over all participants. The dotted curves represent the standard deviations from the mean curve in bold. Bergstrom-Lehtovirta and Oulasvirta [20] showed in prior work that the thumb's maximum range can be described by quadratic curves (reported average $R^2 = .958$). They tested this by fitting a quadratic curve into each thumb trajectory made by participants. We performed the same test for the finger range heatmap for our participants to test whether the maximum range of other fingers can be described by quadratic curves.

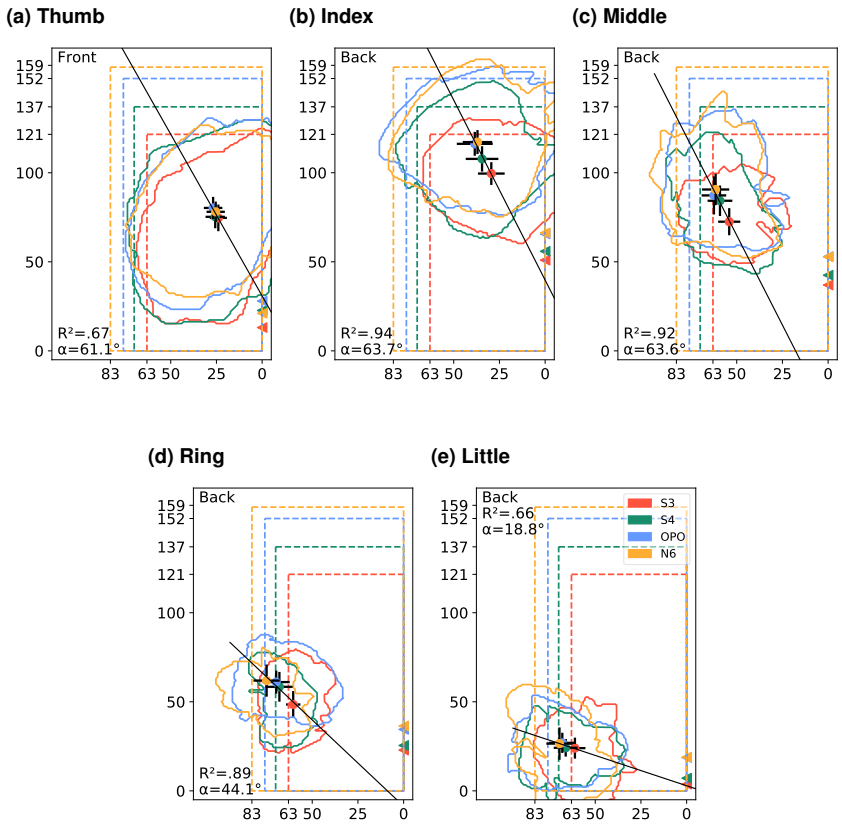


Figure 3.5: Contours of the comfortable areas averaged over participants for all fingers. Dots indicate the area's centroid with whiskers indicating the standard deviation. Black line visualizes the areas' shift with angle α towards the upper left corner with increasing device sizes. Triangles on the right show the average y-position of the respective finger's MCP joint and thus describing the grip. Device sizes are indicated by dashed lines and ticks in mm. Movements of the thumb took place on the front side while all other movements were on the back side.

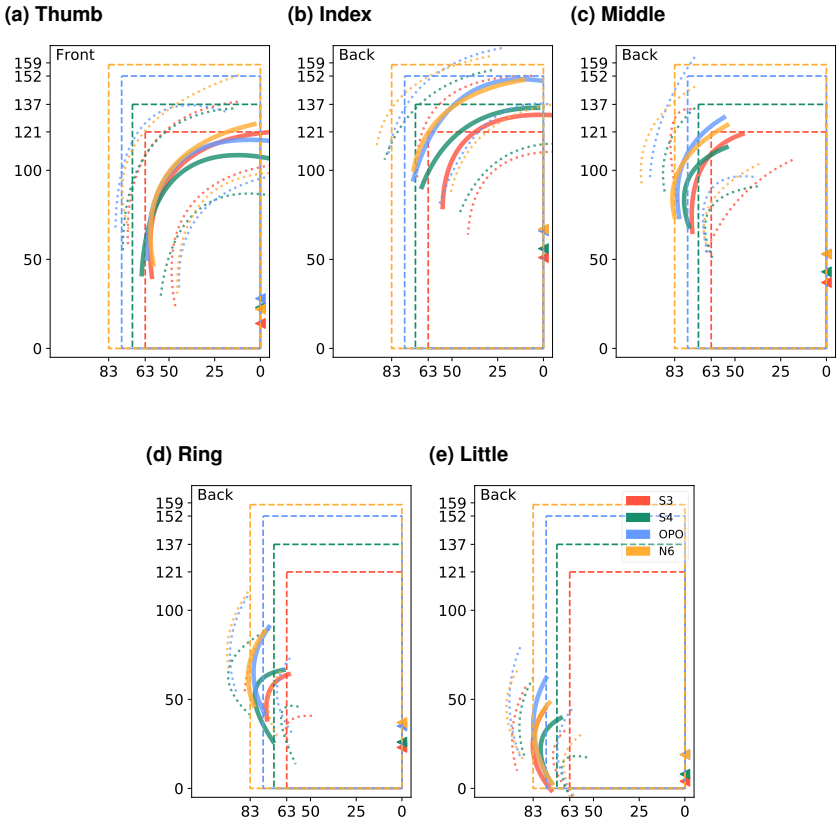


Figure 3.6: These figures show the maximum range for all fingers on four different devices when fully stretched. Bold quadratic curves represent the mean range, while the dotted curves show the ranges one standard deviation further from the mean. Triangles on the right show the average y-position of the respective finger's MCP joint. Device sizes are indicated by dashed lines and ticks in mm. Movements of the thumb took place on the front side while all other movements were on the back side.

Our test yielded an average fitness of R^2 = [.91 .96 .99 .93 .96] indicating that the maximum range of other fingers can also be described through quadratic curves.

Effect of Grip and Hand Sizes

We investigated the effect of hand sizes on the common comfortable area by dividing the data into three balanced sets: *Small hands* ($< 17.5\,cm$), *medium hands* (between $17.5\,cm$ and $20.0\,cm$), and *large hands* ($> 20.0\,cm$).

The way a user holds the device influences the position of the comfortable area and the maximum range (see Figure 3.7). For the four device sizes, we observed the index finger's metacarpophalangeal joint (MCP) which is the joint between the finger and the hand bones. The y-position of this joint indicates the position along the height of the device and thus how high users held the phone, starting from the bottom edge of the phone. Their y-positions are depicted as triangles in Figure 3.7. A one-way ANOVA reveals a significant difference in the y-position of the index finger's *MCP* between four different device sizes, $F_{3,271} = 23.518$, $p < .001$. Bonferroni post hoc tests revealed significant differences between *S3* and *S4* ($p = .002$), *S3* and *OPO* ($p < .001$), *S3* and *N6* ($p < .001$), *S4* and *OPO* ($p = .004$) as well as *S4* and *N6* ($p < .001$).

Figure 3.7a shows the comfortable areas of participants with small hands. Even within a hand size group, the positions of the comfortable areas can be different. Thus, we calculated the average variance (a) between the centroids within groups of hand sizes and (b) between the centroids of all participants as a measure for the spreadness of the comfortable areas. We tested whether there is a significant difference between the average variances of these two groups. A Welch's t-test revealed that the variance for group a (M = 14.8, SD = 8.1) was significantly different from group b (M = 16.2, SD = 8.3), $t_{545.82} = 2.055$, $p = .040$. This shows that the variance between the centroids of the comfortable area can be decreased by $1.4\,mm$ on average when splitted into hand size groups.

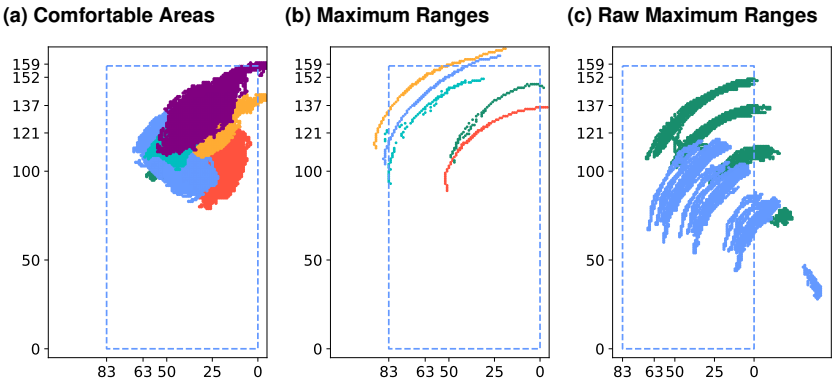


Figure 3.7: For the N6 and the index finger, (a) a set of comfortable areas of participants with small hands, (b) a set of preprocessed maximum ranges of participants with medium hands, and (c) raw maximum ranges for all index finger joints of P5 and P15 with different grips (blue: little finger supporting device's bottom; green: little finger grasping the edge).

3.2.7 Discussion

To help designing one-handed BoD and edge interaction, we conducted a study to record finger movements on smartphones using a high-precision motion capture system. We focused on the area that is reachable without grip changes and losing grip stability (*comfortable area*) and the range that is reachable without a grip change (*maximum range*) for all five fingers on four different smartphone sizes.

The results show that the upper half of the device is comfortably reachable by the index and middle finger (see Figure 3.4). This conforms with findings from previous work [133, 215, 256], and the placement of fingerprint sensors on recent commercial devices (*e.g.*, Google Pixel). The ring and little finger can reach the lower left quarter of the device while the lower right quarter is covered by the palm or parts of the fingers close to the palm (*i.e.*, *proximal phalanges*). Thus, the lower left quarter is not reachable by any finger without a grip change. We further showed that the comfortable areas of the index and middle finger are larger than the counterparts of the ring and little finger. This indicates that both ring and

little fingers are less flexible when grasping the device. While the ring finger can only be moved individually to a lesser extent [76], the little finger is required to support the grip from the bottom side or stabilizing on the left side.

With increasing device sizes, we found that the comfortable areas of the index and middle fingers significantly increase. This conforms with the observation of higher flexibility described above as these fingers can fully explore the increasing rear surface. We also observed that the hand grip, indicated by the positions of the metacarpophalangeal joints (MCPs), move towards the top with increasing device sizes. A possible explanation for this shift is that users try to balance the device's vertical center of gravity by moving the grip towards the top with increasing device height. The shift in hand grip, in turn, affects the centroids of the comfortable areas that shift towards the top left corner of the device. Similarly, the shift of the comfortable area towards the left side can be explained by the balancing of the horizontal center of gravity.

Extending previous work by Bergstrom-Lehtovirta and Oulasvirta [20], we showed that quadratic functions combined with a rotation also enable to describe the maximum range of all fingers with an average $R^2 = .95$. This rotation is necessary as users hold the device in slightly different angles. Conforming the comfortable areas, the maximum ranges show that the upper left corner is not reachable without a grip change. With increasing device sizes, the maximum range moves towards the top left corner similar to the centroids of the comfortable areas. This is caused by the hand grip's shift towards the top edge. Still, the gap between the maximum range and the upper left corner of the device also increases for larger devices and cannot be reached without a grip change.

We investigated and reported the average maximum range of all fingers and the comfortable areas that are reachable by at least 25% of all participants (to exclude outliers). These help smartphone designers to find suitable locations for additional input controls that can be operated in a single-handed grip by a wide range of users. We also reported the variance for groups of participants with different hand sizes and grips. This variance decreases significantly when looking at different groups of hand sizes separately. Since smartphones are not produced for specific hand sizes, we presented the variance as an outlook to future work as an analysis for different groups would go beyond the scope of this thesis.

3.3 Study II: Investigating Unintended Inputs

We use a quantitative approach to empirically study *supportive micro-movements* during common smartphone tasks and scenarios. While participants perform common smartphone tasks (*i.e.*, reading, writing, and abstract touch gestures) in typical scenarios (*i.e.*, sitting and walking) on four different smartphones, we recorded the movement of all fingers with a high-precision motion capture system.

Fingers performing *supportive micro-movements* or holding the phone touch certain spots on the back surface which could produce unintended inputs on BoD inputs controls. In the following, we refer to areas in which *supportive micro-movements* (and thus possibly unintended inputs) occur as the *grip areas*. We adapted the approach described in the previous section (see Section 3.2) to first identify the *grip areas*. Based on the *grip areas*, we then derive the *safe areas* which are the ideal locations to place BoD input controls. *Safe areas* are a subset of the comfortable areas which are outside of the *grip areas*. Thus, the *safe areas* represent a subset of the comfortable areas in which no *supportive micro-movements* and thus unintended inputs occur. We further use the same set of devices to investigate the effect of device size on the amount of *supportive micro-movements*. This helps to find suitable device sizes especially for single-handed interaction with fully touch sensitive smartphones.

In addition to the findings of the previous study which help designers to consider reachability, the findings of the following study help to avoid unintended inputs.

3.3.1 Study Design

We conducted a study to analyze *supportive micro-movements*. Thereby, we focused on the size and position of *grip areas*, the amount of finger movements within these areas, and the length of typical trajectories of *supportive micro-movements* on differently sized smartphones to inform the design of BoD inputs. We adapted the apparatus and processing pipeline from our previous study (see Section 3.2) to find *comfortable areas* which are not covered by *grip areas*. Moreover, we used a NASA-TLX questionnaire [84] and five 7-point Likert scale questions to assess the perceived workload and usability for each smartphone.

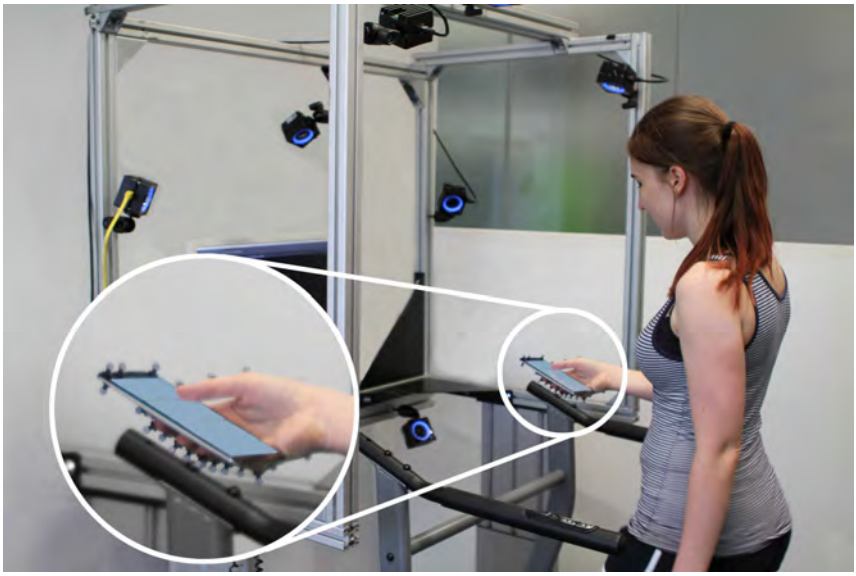


Figure 3.8: A participant interacting with a smartphone while the hand is being tracked by a motion capture system.

This data helps understanding the perceived effort caused by the *supportive micro-movements* which are required for performing single-handed input with the thumb from the user's perspective. We conducted the study in a sitting and walking scenario as previous work showed significant effects of walking (*e.g.* hand oscillations) on smartphone interaction [21, 52, 178].

We used a within-subjects design with SCENARIO and PHONE as the two independent variables. SCENARIO consists of *sitting* on a chair to simulate a still scenario, and *walking* on a treadmill to simulate a mobile scenario with hand oscillations but still enable motion tracking. For each SCENARIO, we used four different smartphones sizes for PHONE. We alternated the order of the SCENARIO for each participant and counterbalanced PHONE with a Balanced Latin Square. In each condition, participants performed three tasks which replicate realistic use cases: *reading* a text, *writing* messages, and performing *abstract input* gestures. These tasks were balanced through a randomized order.

Figure 3.9: Placement of the reflective markers ($6.4\,mm$ spheres) on the right hand for enabling motion tracking.

3.3.2 Apparatus

We used the same set of smartphones as in the previous study in Section 3.2 which are shown in Figure 3.1 and Table 3.1. These devices were selected due to a steady increase in device width which influences the grip the most [133, 226]. From small to large, we used a Samsung Galaxy S3 mini (*S3*), Samsung Galaxy S4 (*S4*), OnePlus One (*OPO*), and a Motorola Nexus 6 (*N6*).

We used an *OptiTrack* motion capture system with eight cameras (*OptiTrack Prime 13W*, $240\,fps$) to record finger movements with sub-millimeter accuracy. The cameras were firmly mounted to an aluminum profile rack as shown in Figure 3.8. We attached 25 reflective markers ($6.4\,mm$ spherical markers) on all joints of the hand as in the previous study (see Section 3.2, and Figure 3.9). In addition, we attached four markers as a rigid body at the top of each smartphone for tracking it with six DoF as shown in Figure 3.1. Participants were sitting on a

chair without an armrest in the *sitting* SCENARIO and walked on a treadmill in the *walking* SCENARIO (see Figure 3.8). Participants walked with 3 *km/h* which is the preferred walking speed for interaction with mobile devices [21].

We developed a custom application to replicate realistic writing and reading tasks which also enables us to log all events. Moreover, we used Fitts' Law tasks (will be described in detail in Section 4.2.1) to cover common touch gestures and induce grip shifts. Our application instructs participants to perform different tasks and logs timestamps for each touch event so that we can synchronize them with *OptiTrack*'s motion data. Figure 3.10 shows screenshots of the respective tasks.

3.3.3 Tasks and Procedure

For each PHONE and SCENARIO, participants performed three tasks. In the *writing* task, participants transcribed excerpts of MacKenzie's phrase set [146] which simulates a text messaging application (see Figure 3.10a). In the *reading* task, participants read and scrolled through text passages adapted from an English learning book [192] (see Figure 3.10b) for two minutes and then answered three comprehension questions which motivated them to focus on reading. With an *abstract input* task, we cover common touch gestures while inducing grip shifts. The task consists of three gestures: *dragging*, in which participants dragged a tile into a target shape with both being randomly placed within a 2×3 grid spanned across the whole screen (see Figure 3.10c); *tapping*, in which they tapped on a target (appeared at a random location) and held it for one second; and *scrolling*, in which they scrolled vertical and horizontal bars into a target shape. Each gesture was repeated 12 times in a randomized order. After each task, participants filled in a NASA-TLX questionnaire [84] and answered five 7-point Likert scale questions to assess the perceived workload and usability for each smartphone.

After obtaining informed consent, we collected demographic data and measured the participants' hand size. We attached the skin adhesive markers on their right hand to enable motion tracking. We explained the tasks and asked the participants to perform them on trial to ensure that everything was fully understood. While they held the devices in a single-handed grip, we did not instruct them to use specific grips as this would influence the generalizability of the study.

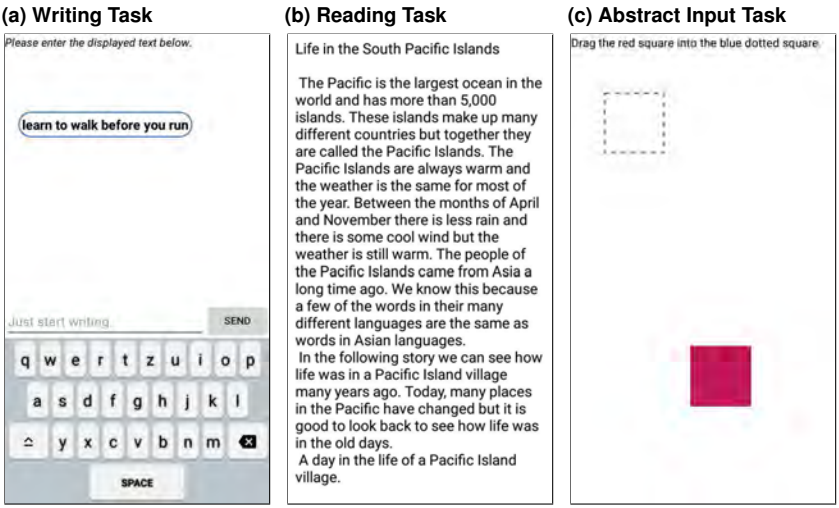


Figure 3.10: Screenshots of the implemented tasks: (a) shows the writing task, (b) shows the reading task, and (c) shows the dragging gesture of the abstract input tasks.

Moreover, for the *writing* task, we instructed them to type as if they would text friends instead of artificially being as precise as possible. Including briefing, optional breaks, and attaching markers, the study took around 90 minutes.

3.3.4 Participants

We recruited 16 participants (9 female, 7 male) between the ages of 18 and 27 (M = 22.9, SD = 2.4). All participants were right-handed. The average hand size was measured from the wrist crease to the middle fingertip and ranged from 17.5 cm to 22.0 cm (M = 19.4 cm, SD = 1.4 cm). Our collected data comprise samples from the 5th and 95th percentile of the anthropometric data [191]. Thus, the sample can be considered as representative. Participants were reimbursed with 10 EUR.

3.3.5 Data Preprocessing

We preprocessed the 3D motion data into 2D heatmaps representing movements on the front and back side of the devices. We adapted the processing pipeline as described in the previous section (see Section 3.2.5) to achieve comparability to the *comfortable areas*. In short, we applied the following preprocessing steps:

1. *Labeling motion data*: We labeled markers using semi-automatic labeling as provided by OptiTrack's *Motive* software. To avoid marker swapping, we used a Max Spike of $4\,mm/frame$ and a Max Gap of $5\,frames$. We did not use any reconstruction and smoothing approaches to avoid generating artificial marker positions. In total, we labeled 17,158,404 frames (*i.e.*, 19.9 hours of motion capturing).

2. *Transforming global to local coordinate system*: We transformed each hand marker from the global coordinate system into the phone's coordinate system and projected them onto the device surfaces. The pivot point is located at the top right corner on the front side. We validated the transformation by sampling five random frames per participant and manually checked them for correctness. While fingers in common grips (*i.e.*, in which the device's rear faces the floor) touch the rear surface to balance and hold the device, rare cases could occur in which the fingers hover over the device's rear such as when holding the phone orthogonal to the floor. In contrast to rear touchscreens or finger painting, our approach enables to also consider finger movements which are slightly hovering during the study.

3. *Cleaning data*: We removed all frames in which the rigid body was not tracked due to occlusion or being out of the tracking grid. To avoid erroneous rigid body tracking (*e.g.*, marker swaps), we assumed that the phone was never held in uncommon orientations (*e.g.*, up-side-down, flipped). With this heuristics, we removed 2.1 % of all recorded frames.

4. *Generating 2D heat maps and determining grip areas*: We generated 2D heat maps representing the *grip areas* with a raster size of $1 \times 1\,mm$ by projecting the markers onto the back and front plane. To remove noise caused by potential markers swaps, we removed all data points with a sum

less than 10 in a 5×5 neighborhood (*i.e.*, all spots touched less than $41.6\,ms$ at $240\,fps$). Using dilation and erosion on a binary version of the heat map, we then filled small gaps within the *grip areas*. The union of the binary heat maps of all participants and tasks finally represent the total *grip areas* per finger and device. In contrast to the previous processing pipeline (see Section 3.2.5), we did not remove outliers (*i.e.*, all spots not touched by at least 25 % of the participants) to cover all areas in which unintended inputs could occur instead of common *grip areas*.

5. *Determining average activity and trajectory lengths*: We represent the finger activity by their average movement speed in cm/s. Thereby, we calculated the movement speed between each subsequent frame and averaged them over all three tasks. We represent the average trajectory length in total travel distance (cm) of a BoD finger while the thumb moves towards the display and performs an input gesture. To determine start and end of single input trajectories, we used the timestamps of the *abstract input tasks* which are separated with short pauses in between. We removed noise caused by potential marker occlusions or swaps by filtering the X and Y coordinates for outliers with a $M \pm 3SD$ filter. The filter removed $0.27\,\%$ of the data.

3.3.6 Results

We present the *grip areas*, finger movement activities, lengths of finger trajectories, and perceived workload and usability for each device. We abbreviate fingers with F_0 to F_4 (*i.e.*, thumb to the little finger) and use square brackets to report values for all fingers [F_0 F_1 F_2 F_3 F_4] and devices [$S3$ $S4$ OPO $N6$]. We mapped the origin $(0,0)$ of all figures to the bottom right device corner as participants used their right hand. While we report the *grip areas* of the thumb for comparison, all ANOVAs are conducted without the thumb as a level for FINGER since we are focusing on unintended BoD inputs. All conducted Tukey post hoc tests are Bonferroni corrected. We corrected the DoFs using Greenhouse-Geisser in case the assumption of sphericity had been violated.

Grip Areas

Figures 3.11 and 3.12 shows the grip areas for all fingers and devices in the *sitting* and *walking* scenario across all three tasks. The colors of the contours represent the device, and the dashed lines represent the size of the respective device. In the following, we describe the characteristics of these areas.

Area Size Table 3.3 shows the size of the *grip areas* for each finger, PHONE, and SCENARIO in cm^2. A Pearson's correlation test revealed significant correlations between the device's diagonal length and the size of the *grip area* in the *sitting* SCENARIO for all fingers (r = [.971 .983 .983 .977 .986], p = [.029 .017 .017 .022 .014]). This correlation can be described as a linear behavior with an average fitness of R^2 = [.94 .97 .97 .95 .97]. For the *walking* SCENARIO, we could not find significant correlations between the device's diagonal length and the size of the *grip area* for all fingers (r = [.947 .605 .398 .445 .685] , p = [.053 .395 .602 .555 .315]).

A three-way RM-ANOVA revealed significant main effects for FINGER ($F_{2.60,38.95} = 4.41$, $p = .012$), PHONE ($F_{1.29,19.31} = 6.404$, $p = .015$), and SCENARIO ($F_{1,15} = .7.02$, p = .018) on the *grip area*. We found neither significant

Scenario	Finger	S3	S4	OPO	N6	Mean	SD
sitting	Thumb (F_0)	46.54	58.76	69.19	86.41	65.22	14.62
	Index (F_1)	14.43	17.23	23.12	26.82	20.4	4.86
	Middle (F_2)	12.67	15.1	19.28	22.62	17.42	3.82
	Ring (F_3)	7.4	13.88	15.96	20.15	14.35	4.6
	Little (F_4)	12.37	17.76	21.93	27.44	19.88	5.53
	$\cup$BoD (F_{1-4})	40.13	51.06	65.39	74.72	57.83	15.3
walking	Thumb (F_0)	50.01	60.92	68.79	88.26	67.0	13.97
	Index (F_1)	21.48	15.25	20.8	32.94	22.62	6.43
	Middle (F_2)	20.81	14.27	15.19	30.19	20.11	6.33
	Ring (F_3)	18.35	12.26	12.62	30.51	18.43	7.38
	Little (F_4)	21.88	18.05	22.39	41.09	25.85	8.96
	$\cup$BoD (F_{1-4})	61.13	51.48	62.4	101.05	69.02	21.91

Table 3.3: Grip areas in cm^2 for all fingers and scenarios on four devices. $\cup$BoD represents the union of the grip areas on the rear.

two-way interactions nor three-way interactions between the factors (p > .05, each). Tukey post hoc tests revealed significant differences between OPO and N6 (p = .036) and between S4 and N6 (p = .007). A further Tukey post hoc test did not reveal significant differences between the fingers.

Due to significant main effects of SCENARIO and since we are interested in differences within the scenarios, we conducted two further two-way RM-ANOVAs on the *sitting* and *walking* subset. For the *sitting* SCENARIO, we found significant main effects for PHONE ($F_{3,45} = 9.26$, $p < .001$) but not for FINGER ($F_{3,45} = 4.41$, $p = .153$) and no two-way interactions between FINGER × PHONE ($F_{2.95,44.19} = .616$, $p < .605$). A Tukey post hoc test revealed significant differences between the S3 and N6, between S4 and N6, and between the OPO and N6 ($p < .001$, each) but not for the other combinations (p > .05). For the *walking* SCENARIO, we found main effects for FINGER ($F_{1.89,28.30} = 3.83$, $p = .002$), PHONE ($F_{1.16,17.37} = 3.97$, $p < .001$), and a two-way interaction effect between FINGER × PHONE ($F_{2.84,42.6} = 2.18$, $p = .003$). A Tukey post hoc test revealed significant differences between the S3 and N6, between S4 and N6, and between the OPO and N6 ($p < .001$, each) but not for the other combinations (p > .05).

Area Position The dots in Figures 3.11 and 3.12 represent the area's centroid position averaged over all participants while whiskers represent the standard deviation.

For the *sitting* SCENARIO, the shift of the centroids towards the upper side can be described by a linear function with a fitness of R^2 = [.77 .89 .04 .66 .04] for all five fingers. Pearson's correlation test revealed no correlation between the device's diagonal and a gradual shift of all fingers towards the top left corner (r = [.877 −.945 −.187 .811 .194], p = [.123 .055 .813 .189 .806]). For walking this shift can be described by a linear function with a fitness of R^2 = [.88 .77 .89 .85 .71] for all five fingers. Pearson's correlation test revealed no correlation between the device's diagonal and a gradual shift of all fingers towards the top left corner (r = [.94 −.875 .942 .921 .841], p = [.806 .06 .125 .058 .079]).

Safe Areas The dark gray areas in Figure 3.13 represent the total *grip area* on the back of the device. The light gray areas represent the total *comfortable area*

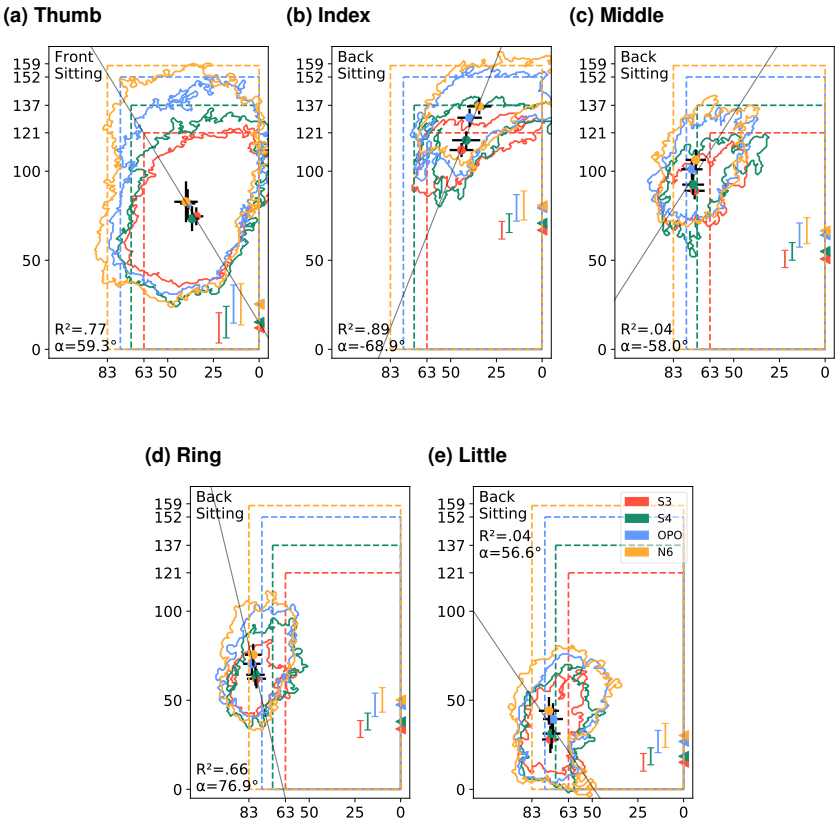


Figure 3.11: These figures show the grip areas for all fingers on four different devices (S3, S4, OPO, N6) and in the SITTING scenario. Dots indicate the area's centroid with whiskers indicating the standard deviation. Black lines visualize the areas' shift with angle α towards the upper left corner with increasing device sizes. Triangles and lines on the right show the average y-position of the respective finger's MCP joint and thus describing the grip. Device sizes are indicated by dashed lines and ticks in mm. Movements of the thumb took place on the front side while all other movements were on the back side.

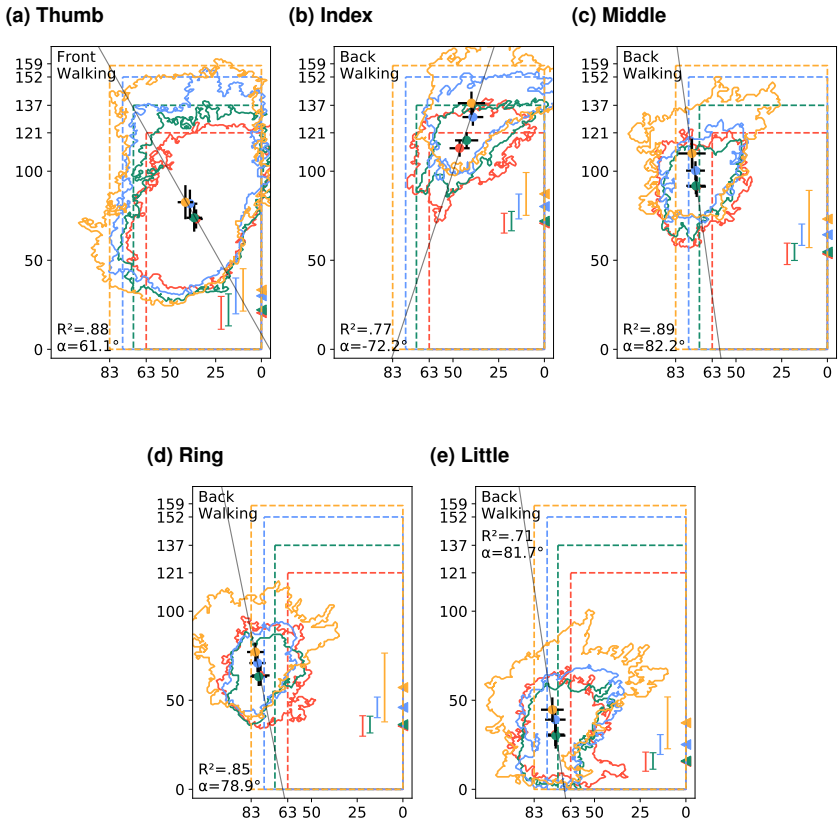


Figure 3.12: These figures show the grip areas for all fingers on four different devices (S3, S4, OPO, N6) and in the WALKING scenario. Dots indicate the area's centroid with whiskers indicating the standard deviation. Black lines visualize the areas' shift with angle α towards the upper left corner with increasing device sizes. Triangles and lines on the right show the average y-position of the respective finger's MCP joint and thus describing the grip. Device sizes are indicated by dashed lines and ticks in mm. Movements of the thumb took place on the front side while all other movements were on the back side.

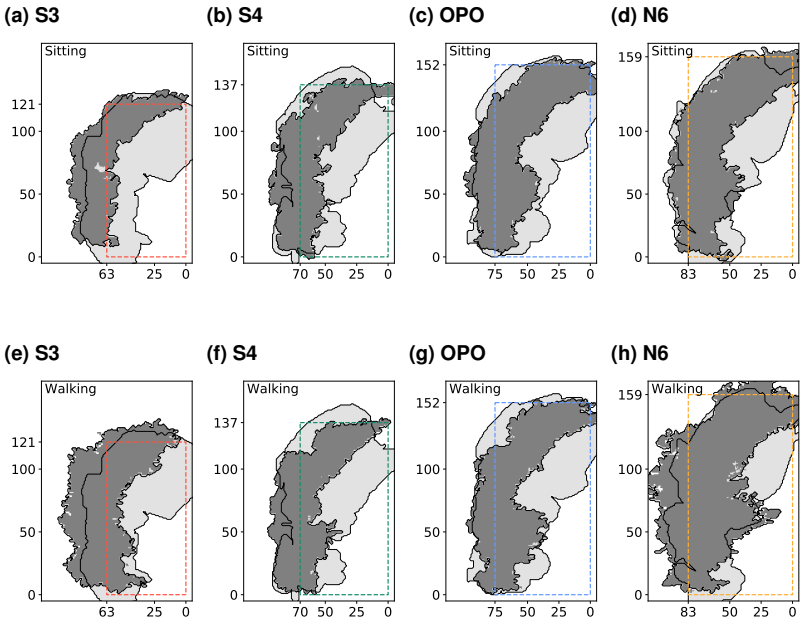


Figure 3.13: The dark gray areas represent the union of the back fingers' *grip area* ($F_1 - F_4$). The areas in light gray show the *comfortable areas* for the BoD fingers as shown in Section 3.2.6. We refer to the subsets of the *comfortable areas*, which are not covered by the *grip area*, as the *safe areas*. The axes denote *mm* starting from the bottom right corner.

as reported in Section 3.2.6. With both areas overlapping, the remaining light gray areas represent the area which is comfortably reachable while no *supportive micro-movements* occurred within these areas. We refer to these areas as the *safe areas*. The *safe areas* correspond to [60.3 48.4 40.9 35.9]% of the *comfortable area* during *sitting* and [45.4 48.1 43.4 25.6]% during *walking*.

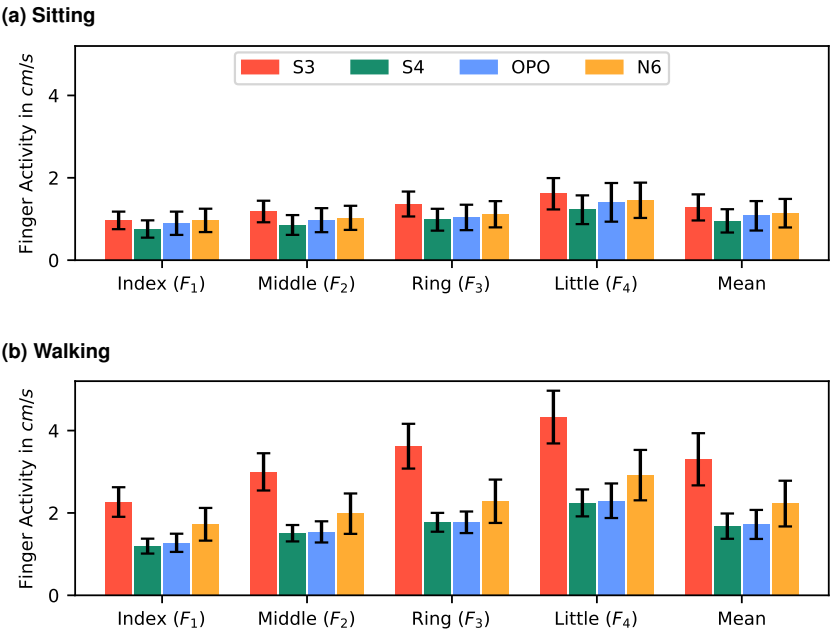


Figure 3.14: The average finger activity across all tasks in cm/s. Error bars represent the standard deviations.

Finger Movement Activity

Figure 3.14 depicts the movement activity for all fingers on the back of all devices. A three-way RM-ANOVA revealed significant main effects for FINGER ($F_{1.69,25.39} = 136.205$, $p < .001$), PHONE ($F_{3,45} = 46.25$, $p < .001$), SCENARIO ($F_{1,15} = 412.274$, $p < .001$), as well as for all two-way interactions ($p < .001$, each) and three-way interactions ($F_{3.33,49.99} = 9.45$, $p < .001$). Due to significant main effects in SCENARIO and since we are interested in differences within the scenarios, we conducted two further two-way RM-ANOVAs on the *sitting* and *walking* subset.

For the *sitting* SCENARIO, we found significant main effects for FINGER ($F_{1.69,25.39} = 54.67$, $p < .001$), PHONE ($F_{3,45} = 5.02$, $p < .001$), as well as a two-

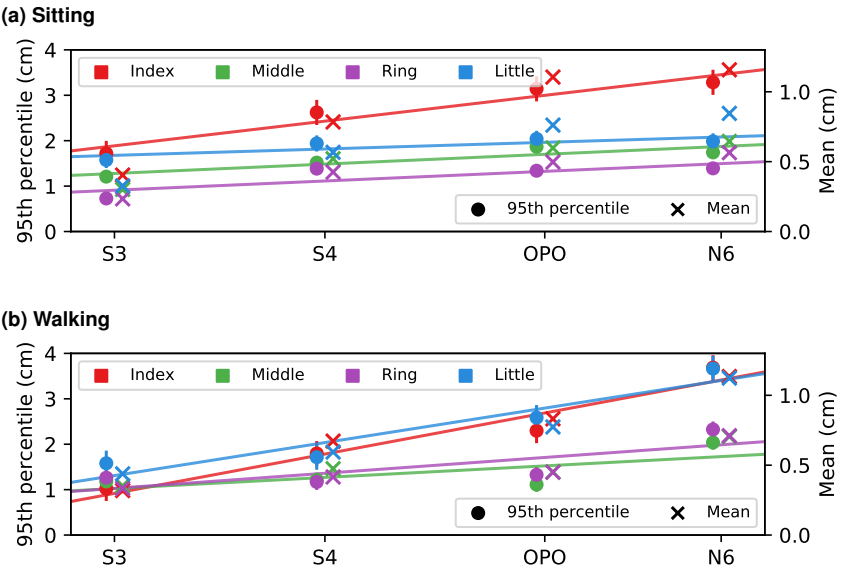


Figure 3.15: Length of finger trajectories in *cm* across the abstract input task. The dots (right axis) represent the 95th percentile and its linear growth (mean R^2 = .75). The crosses (right axis) represent the mean length.

way interaction effect between FINGER × PHONE ($F_{3.33,49.99} = 4.14$, $p < .001$). A Tukey post hoc test did not reveal any significant differences between the phones. For the *walking* SCENARIO, we found significant main effects for FINGER ($F_{1.79,25.90} = 176.35$, $p < .001$), PHONE ($F_{2.13,31.99} = 38.06$, $p < .001$), as well as a two-way interaction effect between FINGER × PHONE ($F_{2.73,41.02} = 17.09$, $p < .001$). A Tukey post hoc test revealed significant differences between S3 and N6, S4 and N6 ($p < .05$, each), and between S3 and OPO and between S3 and S4 ($p < .001$, each).

Length of Finger Trajectories during Grip Shifts

Figure 3.15 depicts the 95th percentile for the length of finger trajectories as dots (left axis) and their means as crosses (right axis). A Pearson's correlation test

revealed significant correlations between the device's diagonal length and the length of finger trajectories in the *sitting* SCENARIO (r = [.957 .988 .969 .974 .997] , p = [.043 .012 .031 .026 .003]) and for walking (r = [.942 .981 .840 .868 .942] , p = [.058 .019 .160 .132 .058]). The correlations can be described as a linear behavior with an average fitness of R^2 = [.92 .98 .94 .95 .99] for *sitting* and R^2 = [.89 .96 .71 .75 .89] for *walking*.

Effect of Phone Size on Perceived Effort

To evaluate the perceived workload and usability of each device averaged over all tasks, we used a raw NASA-TLX questionnaire [84] and five 7-point Likert scale questions as described in Section 4.2.3.

Perceived Workload Figure 3.16a shows the average perceived workload measured with a raw NASA-TLX questionnaire after each condition. A two-way ANOVA revealed significant main effects for PHONE ($F_{3,45} = 12.742$, $p < .001$) on the total workload but neither for SCENARIO ($F_{1,15} = 1.71$, $p = .21$) nor for the two-way interactions between PHONE $\times$ SCENARIO ($F_{3,45} = .429$, $p = .733$). A Tukey post hoc test revealed significant differences between N6 and OPO, between N6 and S3, and between N6 and S4 ($p < .01$, each).

Subjective Perceptions Figure 3.16b and 3.16c show the average perceived ratings when asked for easiness, speed, success, accuracy, and comfort after using a specific PHONE.

We conducted five two-way ANOVAs on the ratings on which we applied the Aligned Rank Transform (ART) procedure using the ARTool [259] to align and rank the data. For all ratings, the two-way ANOVAs revealed significant main effects for PHONE ($p < .05$, each). For the ratings easiness and accuracy, we found significant main effects for SCENARIO ($p < .05$, each). For easiness, we found significant two-way interactions between PHONE and SCENARIO (p = .032). Five corresponding Tukey post hoc tests revealed significant differences between S4 and N6 for all ratings ($p < .05$), between OPO and N6 for easiness, success, and comfort ($p < .05$), and between S3 and N6 for easiness and comfort ($p < .05$).

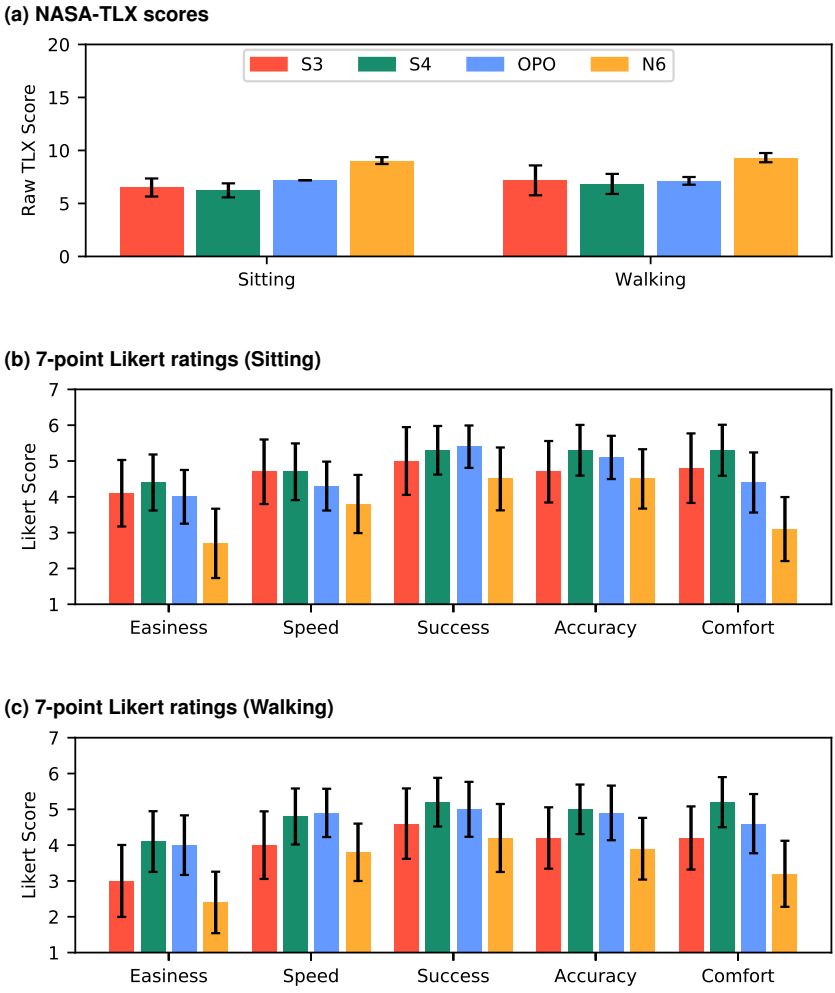


Figure 3.16: Perceived workload (unweighted NASA-TLX) and subjective perceptions of the usability (7-point Likert scale) for each device averaged over all tasks. The colors represent the devices, attached whiskers the standard deviation.

3.3.7 Discussion

Previous work analyzed *comfortable areas* and presented design implications for BoD input controls to consider single-handed reachability. With our analysis, we identified suitable locations for BoD input, ideal device sizes, and further properties which help to minimize unintended inputs while maintaining reachability. We first discuss our results and then present three design implications for BoD input.

Safe Areas: Overlaps of Grip and Comfortable Areas

Safe areas are subsets of the *comfortable areas* in which no *supportive micro-movements* were observed. The *safe areas* cover 46 % (43.5 cm^2) of the *comfortable areas* while sitting and 40 % (38.4 cm^2) while walking on average. The majority of *safe areas* are located in the upper right quarter of the device and thus between the fingertips (when stretched) and the palm. Placing BoD input controls in these areas enable users to easily reach them by subtly flexing their finger. The fingertip of a flexed finger (see Figure 3.17a) provides enough force to activate a physical button (*e.g.*, BoD volume buttons on the LG G-series) and suitable accuracy for touch-based input controls due to a small contact area. While finger parts (*e.g.*, the *intermediate phalanges*) could come in contact with an BoD input control when stretched (see Figure 3.17b), the finger's force towards the device surface is too low (due to the force distribution) to hit the button's activation point. Even if users deliberately apply a force towards the back surface with a stretched finger, the center of pressure is located at the fingertip so that the phalanges cannot apply enough force to unintentionally hit a flat button's activation point.

For touch-based input, fingertips can be differentiated from phalanges by their contact areas. This is feasible with capacitive sensing which previous work used to identify body parts on commodity devices [99]. For fully touch sensitive smartphones, we will present a model to accurately translate contact areas on the device surface to the fingertips' 3D locations in Section 5.2 which also automatically omits touches by the phalanges.

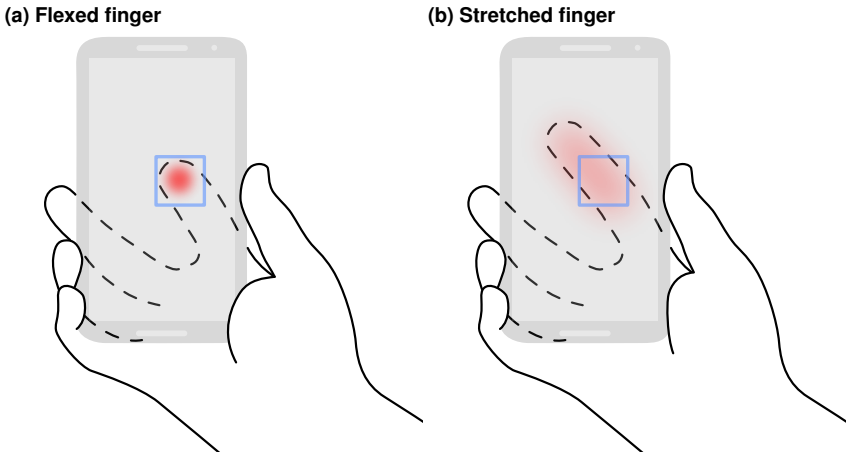


Figure 3.17: Performing input on a flat BoD button (blue square). The finger is flexed in (a) which leads to a small contact area. This bundles the force to hit the button's activation point (red dot). In (b), the finger is stretched, which leads to a larger contact area and thus a more distributed force which is not enough to activate the button (light red area).

Effect of Device Size on Grip Areas and Activities

We found the largest *grip areas* on the N6 in both sitting and walking scenario. For the sitting scenario, we further found a significant correlation between the size of the devices and *grip areas*. In contrast to the other devices, the unusually large size of the N6 requires additional *supportive micro-movements* to maintain a firm grip. The hand spans of our participants were not large enough to apply a firm grip (*i.e.*, power grip [175]) to encompasses the whole device width. More importantly, the $6''$ touchscreen requires an extensive thumb range which can only be achieved with large grip shifts. Analyzing finger trajectories lengths confirms that larger grip shifts were indeed performed on larger devices. This conforms with Eardley *et al.* [54] who found more device movements on larger phones.

Although the S3 entails a smaller *grip area* than the N6 as expected, we observed similar finger activities in a sitting scenario and significantly higher activities

than on any other device in a walking scenario. The high finger activity on the S3 is due to a small touchscreen which requires *supportive micro-movements* for more input precision. As smaller contact areas lead to a more precise input [97, 98], additional *supportive micro-movements* were performed to enable the thumb to touch with a high pitch angle (*i.e.*, nearly perpendicular to the display). Moreover, as the S3 fits well in the hand without a firm grip, users mostly held the device in a loose grip which provides the thumb with more flexibility.

In contrast, we observed the smallest *grip area* on the S4 while walking and on the S3 while sitting. Both the S4 and OPO are between the S3 and N6 in size and entailed less finger activity than the S3 (while walking) and the N6. Moreover, they do neither need additional *supportive micro-movements* for a firm grip nor to enhance the touch precision.

Effect of Walking on Grip Areas and Activities

We observed larger *grip areas* and significantly more BoD finger activity for walking than for sitting. Since walking introduces hand oscillations which previous work showed to affect mobile input [21, 179], additional *supportive micro-movements* are required to compensate the vibrations and avoid dropping the device. The hand oscillations also explain why there is no correlation between *grip area* and device size in a walking scenario as described above. The loose grip on the S3 provides the thumb with more flexibility, but also leads to more device movements in the hand caused by oscillations. Moreover, an analysis of the finger trajectories lengths reveals that the little finger moved significantly more in the walking condition. As the little finger stabilizes the grip from below, its movements indicate re-adjustments of the grips hampered by hand oscillations. These findings suggest that suitable device sizes are required to avoid *supportive micro-movements* for input precision and grip shifts. Thereby, we found that the size of the S4 is favorable for BoD input.

Perceived Usability and Workload

The perceived usability conforms to the observed finger movement activities. Especially for walking, we found a similar behavior to the finger movement

activities in which the S4 and OPO are more favorable than other devices. Both devices received better ratings in easiness, speed, success, accuracy, and comfort which reflects the lower effort for holding and interacting with the devices. For the sitting condition, the N6 received the lowest ratings while the other devices received comparable ratings. This further highlights that the size of an S3 might be suitable for a sitting scenario but not under the influence of hand oscillations while users are walking. Results of the raw NASA-TLX revealed significantly more perceived workload on the N6 than on any other device. Again, we argue that this is due to its unusual size which even surpasses large versions of recent flagship smartphones (*e.g.*, iPhone XS Max, Samsung Galaxy S9 Plus).

3.4 General Discussion

In this chapter, we presented two studies to understand the hand ergonomics for interaction beyond the touchscreen. In particular, the two presented studies focused on the on-device areas which can be used for interaction, and fingers and hand parts which could be used for interaction.

3.4.1 Summary

Addressing RQ1, the results of the first study revealed the comfortable area (*i.e.*, areas on the device which are reachable without stretching fingers or changing the grip) and the maximum range (*i.e.*, reachable without a grip change) of all fingers. Using these measures as a basis, designers can now find suitable on-device locations for BoD input controls to avoid muscle strain and grip changes which could lead to dropping the device. The results further revealed that the index and middle fingers are the most suitable for BoD input due to their flexibility. In addition, we found that the palm covers the bottom right quadrant of the back side and parts of the bottom left quadrant (directly opposite). While this means that placing input controls in these regions should be avoided if possible, the controlled presence of the palm (*e.g.*, stretching the thumb places the palm on the display) could potentially be used as an input modality.

Addressing RQ2, the second study extends the results of the first one. We analyzed the *supportive micro-movements*, which occur during interaction with

smartphones due to maintaining a stable grip, limited independence of the fingers [76], and to increase the range of the thumb on the front side. As these could lead to unintended input on the back side, we introduced the *safe areas* which are a subset of the *comfortable area*. The *safe areas* are reachable without a grip change and finger stretching while they entail the least amount of *supportive micro-movements* to minimize unintended BoD input.

3.4.2 Design Implications

We present seven design implications in the following which helps designers in finding suitable locations for BoD input and to design them in a way in which unintended inputs can be minimized while considering the reachability.

Do not place input controls on the bottom right. The bottom right quadrant on the back of the device is not reachable for any of the four fingers on the back without a grip change. When holding a phone, this area is also covered by the hand's palm. Hence, no input controls should be placed at the bottom right corner of the device to avoid grip changes and unintentional input.

Increase reachability by placing input controls within the comfortable area. Fingers can move freely within the comfortable area without grip changes or losing grip stability. The majority of the comfortable area is located on the upper half of the device's rear and reachable by the index and the middle finger. To avoid dropping the device and muscle strains, input controls should be placed so that interaction takes place within the comfortable area.

Avoid unintended inputs by restricting comfortable area to the safe areas. In addition to the *comfortable areas*, which avoids grip shifts, *safe areas* are a subset of the *comfortable areas* in which no *supportive micro-movements* occurred. These areas are located in the upper right quarter of the device and can be reached by subtly flexing the index finger, which is also the most suitable finger for BoD input [130]. To avoid unintended input by other finger parts (*e.g.*, *intermediate phalanges*), physical flat buttons should be used. Touch-based controls can use the contact surface or a contact translation model [123] to omit touches by other finger parts.

Place input controls higher on larger devices. We found that both the comfortable areas and the average position of the finger's MCP are shifting towards the top edge of the device with increasing devices sizes. This indicates that users are holding the device higher the larger the device is. Thus, we recommend to place input controls higher for larger devices, including buttons on the left and right edges.

Use the index finger for complex and frequent BoD input. The index finger has the largest comfortable area on the back side of all four devices due to its flexibility. Complex and frequent movements such as BoD gestures and location-dependent tapping (*e.g.*, fingerprint scanners) benefit from this flexibility and should be performed with the index finger.

Consider 5″ devices for single-handed BoD inputs. Large devices (*e.g.*, Nexus 6) do not enable firm grips due to their width while small devices (*e.g.*, Samsung Galaxy S3 mini) require a nearly perpendicular thumb and thus additional *supportive micro-movements* for input precision. We found that 5″ devices (*e.g.*, Samsung Galaxy S4) were perceived as the most usable while entailing the lowest amount of *supportive micro-movements* for single-handed use while sitting and walking. Thus, we recommend 5″ devices for BoD inputs.

Expect longer finger trajectories on larger devices. With increasing touchscreen sizes, larger grip shifts are required to provide the thumb with the required reachability. Thus, users perform a longer trajectory of *supportive micro-movements* (*e.g.*, unintended stroke gesture) while shifting the grip. While the input trajectory length is a common feature to filter unintended inputs [153, 211], we recommend considering the device size as a factor when choosing the threshold. Figure 3.15 suggests threshold values in *cm* observed during grip shifts.

4

Hand-and-Finger-Awareness on Mobile Touchscreens

In the previous chapter, we explored the hand ergonomics to understand which parts of the hand can be used for interaction in a single-handed grip (*e.g.*, index and middle finger, and the palm). This corresponds to the first and second step of the UCDDL. In this chapter, we focus on the steps 3, 4, and 5 of the UCDDL by implementing and evaluating novel touch-based interaction techniques based on deep learning.

In particular, we present an approach which applies state-of-the-art deep learning techniques to identify different sources of touch input based on the raw data of mass-market capacitive touchscreens. This demonstrates that the touch input vocabulary on commodity smartphones can already be meaningfully extended with a high accuracy input method even without any additional sensors. The research approach and technical pipeline presented in this chapter serve as a basis for further studies on hand-and-finger-aware touch input on smartphones which we will present in the subsequent chapters.

Parts of this chapter are based on the following publication:

H. V. Le, T. Kosch, P. Bader, S. Mayer, and N. Henze. "PalmTouch: Using the Palm as an Additional Input Modality on Commodity Smartphones." In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI '18. New York, NY, USA: ACM, 2018. DOI: 10.1145/3173574.3173934[a]

H. V. Le, S. Mayer, and N. Henze. "Investigating the Feasibility of Finger Identification on Capacitive Touchscreens using Deep Learning." In: *24th International Conference on Intelligent User Interfaces*. IUI '19. Marina del Ray, CA, USA: ACM, 2019. DOI: 10.1145/3301275.3302295[b]

[a] Video Preview: https://www.youtube.com/watch?v=GFSbboPV7NI
[b] Video Preview: https://www.youtube.com/watch?v=jod_-FprYf4

4.1 Identifying the Source of Touch

Based on the findings of Chapter 3 and related work, we present the concept of two novel and additional input modalities for capacitive touchscreens. This section includes the context of use and requirements as described in the UCDDL.

4.1.1 The Palm as an Additional Input Modality

Previous work presented a number of alternative input modalities to support traditional multi-touch input. This includes using the finger's 3D orientation [156, 198, 202, 265], contact size [24], pressure [91], or the shear force [80]. While these enrich the information of a finger's touch, they also bring restrictions since specific finger postures may now trigger unwanted actions. One solution to lower the likelihood of triggering unwanted actions is to differentiate between fingers or parts of fingers, which prevents interference with the main finger for interaction. Previous work [38, 63, 82] differentiated between different parts of the finger (*e.g.*, knuckle) or fingers themselves to assign unique touch actions.

We presume that touching the display with the palm can be a natural gesture. Since the palm can be used in a single-handed as well as in a two-handed grip, we present an additional input modality that enables people to use the palm to trigger pre-defined functions instead of simply rejecting palm input as recent smartphones

do. We refer to this input modality as *PalmTouch* and show that it is a natural and fast gesture especially when the device is held one-handed. Stretching the thumb towards the top edge to access targets that are out of reach often places the palm on the touchscreen implicitly and subtly as shown in Figure 4.1a. The placement is often caused by unawareness of users which suggests that this gesture can be performed naturally. Figure 4.1 shows three examples of using *PalmTouch* in one-handed and two-handed scenarios to trigger assigned functions.

Regarding the technical approach, previous work presented different features to detect a palm on a touchscreen. These include spatiotemporal touch features [211], and hand model filters [227] to detect the palm in inking scenarios on tablets. Moreover, Matero and Colley [153] presented characteristic patterns of unintentional touches, including touch duration which had the largest influence on rejection performance. However, these approaches require at least two touch points (pen and palm) or introduce latency due to temporal features which makes them not suitable for our proposed palm input modality. Recent smartphones feature a basic palm rejection which omits input in case the contact area is larger than a usual finger. However, they work on a driver level and are not reliable enough to be used for interaction.

In this chapter, we present *PalmTouch*, an additional touch input modality to trigger pre-defined functions by placing the palm on the touchscreen. Accordingly, we present four use cases for *PalmTouch* and evaluate the input modality as a shortcut and to improve reachability during one-handed smartphone interaction. To evaluate *PalmTouch*, we have developed a palm detection model that differentiates between finger touches and palm touches with a high accuracy. In contrast to previous work, we use the raw capacitive image of the touchscreen to classify the low-resolution fingerprint using a convolutional neural network. We show that this runs on off-the-shelf smartphones, also works with single touch points and introduces no latency opposed to previous work.

PalmTouch Concept and Use Cases

PalmTouch is an additional input modality for a wide range of functions. We applied the idea of hand part specific touch interaction presented in previous work (*e.g.*, using different fingers [38, 63] or finger parts [82]) for one-handed as well

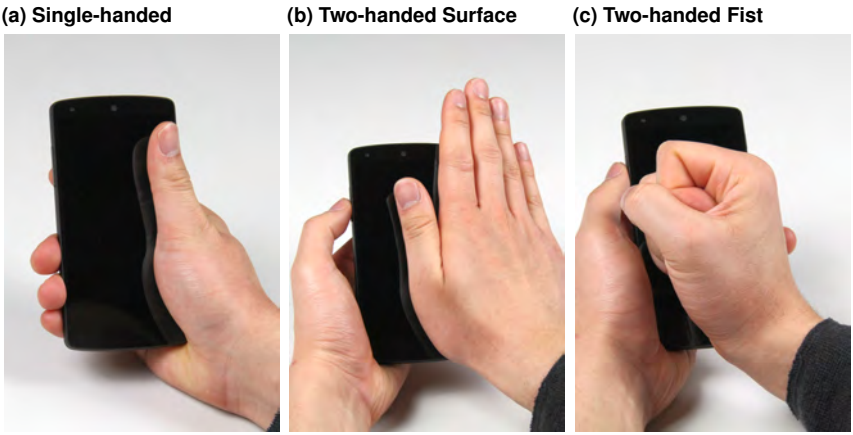


Figure 4.1: Using the palm as an additional input modality on smartphones. Figure (a) shows a palm touch when holding the device one-handed, Figure (b) and (c) show palm touches for two-handed interaction.

as two-handed interaction scenarios. Since using other fingers than the thumb or other parts of the hand (such as a knuckle) can be inconvenient or even infeasible during one-handed interaction, we instead use the palm for interaction.

During one-handed interaction, the palm can be placed subtly on the touchscreen by moving the thumb towards the upper edge of the device while stabilizing the device with fingers on the left edge as shown in Figure 4.1a. Since we use the palm of the same hand that is holding the smartphone, we refer to this movement as a *same-side palm touch*. During two-handed interaction, *PalmTouch* can be used by placing the flat hand (see Figure 4.1b) or by forming a fist on the touchscreen (see Figure 4.1c). Since we use the opposite hand to the one holding the device, we refer to this movement as an *opposite-side palm touch* based on the terminology used by Kerber *et al.* [111]. In the following, we present four use cases and discuss further input dimensions that extend the *PalmTouch* input modality.

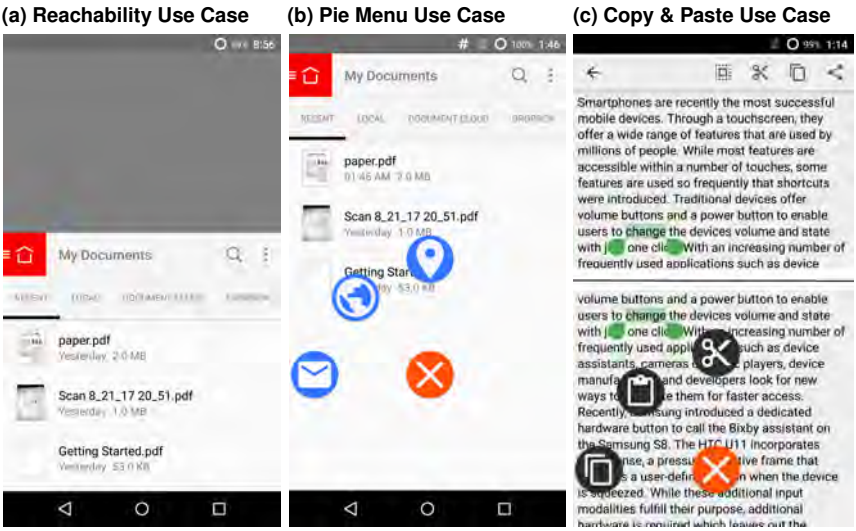


Figure 4.2: Use cases for *PalmTouch*. Figure (a) demonstrates how *PalmTouch* improves reachability by moving down the screen by half its size; Figure (b) shows the pie menu for application launching and Figure (c) shows the pie menu for clipboard management.

Improving Reachability during One-Handed Interaction Large smartphones pose challenges in reachability since they require changing the hand grip when used one-handed. With *PalmTouch*, users can stretch the thumb towards the top as if they would tap the target. This action implicitly places the palm on the touchscreen and can be used by *PalmTouch* to shift down the screen by half its size. A screen shift is exemplarily shown in Figure 4.2a and is similar to the iPhone's *Reachability* feature that can be activated by a double tap on the home button. Similarly, instead of dragging down the notification bar which poses the same reachability challenge on large smartphones, *PalmTouch* can be used to open the notification drawer. Further difficult to reach UI elements include toolbars (*e.g.*, ActionBar[1]), URL bars in most browsers, search bars, menu buttons, and tabs.

[1] `developer.android.com/design/patterns/actionbar.html`

Custom Actions and Applications Smartphone manufacturers recently integrated simple and binary input modalities such as an extra button (Bixby button on the Samsung Galaxy S8) or a squeeze on the device's edge (Edge Sense on the HTC U11) to launch pre-defined applications. While these features require additional hardware, *PalmTouch* can be readily deployed onto recent and older off-the-shelf smartphones, *e.g.*, through software updates. Moreover, with the emergence of edge-to-edge displays on devices such as the iPhone X and Samsung Galaxy S8, the lack of a home button can be compensated with *PalmTouch*.

Instead of launching a single pre-defined action or application, a pie menu as shown in Figure 4.2b can be used to provide multiple options. The arrangement of buttons in a pie menu further benefits one-handed interaction. Previous work [20, 130] showed that the range of the thumb on a touchscreen is parabolic around the *carpometacarpal joint* (CMC) of the thumb. The CMC is located in the lower part of the palm. Since the palm is placed on the touchscreen to launch the pie menu, the thumb is already in a suitable position to tap the menu items. *PalmTouch* can also be used for application-dependent functions. For example, a palm touch could send away a message in a messaging application, while it accepts a call in the phone application or switch layers in Maps or CAD application. Since *PalmTouch* can be used eyes-free similar to a hardware button or squeeze, actions such turning off the screen or accepting a call can be mapped to a palm touch.

Clipboard Management Copying and pasting from the clipboard are common actions in text editing tasks. While computer keyboards provide simple shortcuts, touch-based operating systems such as Android and iOS handle the access through context menus or buttons in the toolbar. A context menu requires a long press that takes between $500\,ms$ and $1000\,ms$ and could further move the caret to another location unintentionally due to the fat-finger problem [16]. Toolbar buttons require additional screen space. Therefore, we propose *PalmTouch* as a shortcut to the clipboard menu which avoids long-pressing and altering the caret position. To paste text, PalmTouch can open a menu which offers the function without a long-press. For text selection and copy/cut, users can perform a palm touch to

start text selection and then use the menu as soon as the selection via finger was done to avoid a long-press. Figure 4.2c shows an example where users can select between copy, paste and cut after placing the palm on the touchscreen.

Unlocking the Device *PalmTouch* can be used to unlock the smartphone by placing the palm on the touchscreen. This action can be done with a same-side palm touch while holding the device, or with one of the opposite-side variants when the device is, *e.g.*, lying on a table. In addition to the palm detection, *PalmTouch* can be extended to use the biometric features presented in *BodyPrint* [99] for authentication based on the capacitive images.

Additional Input Dimensions In addition to a binary action, *PalmTouch* offers further dimensions that can be used for interaction. The contact area's centroid can be used as a proxy for the palm touch location. This enables the implementation of directional gestures, such as swiping up with the opposite hand's palm to exit an app and swiping left or right to switch between apps. The location of the opposite hand's palm can also be used to map functions to different locations of the touchscreen. For example, a palm touching the top half of the display skips to the next music title while a touch on the lower half plays the previous title. The location can also be used for a same-side palm touch (*e.g.*, x-position describes the used hand) to launch different actions depending on the hand that performed the palm touch.

4.1.2 Investigating the Feasiblity of Finger Identification

In addition to *PalmTouch* as an accurate input technique based on deep learning and the raw capacitive data, we investigate the feasibility of finger identification based on the same pipeline. A large body of work has already explored techniques to enable finger-aware touch interaction. By executing similar actions, but with different fingers, users can enter different commands similar to the use of multiple buttons on a computer mouse or modifier keys on keyboards. These techniques enable promising use cases, such as improving text entry on small touch displays [75], providing finger-aware shortcuts on touch keyboards [284], and enhancing multitasking on smartphones [74]. As a result, a number of hardware

prototypes that enable finger-aware interaction were presented. This includes sensors attached to the fingers [74, 75, 152], gloves [149], electromyography [19], and cameras attached to the device (*e.g.*, RGB [245, 284], depth sensor [172, 252]). While these approaches are accurate, they require sensors to be attached to the user or the device which reduces mobility. There is no standalone solution yet that identifies fingers on a commodity smartphone.

One solution to avoid additional sensors for finger identification is to use the contact geometry of touches. Previous research focused predominantly on tabletops that provide high-resolution images of touches [8, 56, 62] to identify fingers based on multi-touch hand models. By modifying the firmware of smartphones, researchers used the raw data of commodity touchscreens (referred to as *capacitive image*) to infer further input dimensions. Capacitive images represent low-resolution fingerprints and can be used to estimate the finger orientation [156, 265], recognize body parts [99], palm touches [136], and hand poses [171]. Gil *et al.* [63] used capacitive images of a smartwatch prototype to differentiate between touches of thumb, index and middle finger. However, they used exaggerated poses on smartwatches so that each finger touched with a distinct angle. Expecting these poses does not only impact the usability but they are also not common and ergonomic for smartphone use (*e.g.*, touching with half the middle finger).

Previous work showed that capacitive images provided by mobile devices do not contain sufficient signal to identify each finger during regular interaction [63]. However, being able to differentiate between the primary input fingers (*e.g.*, right thumb) and others is already a useful addition to the input vocabulary. For example, a second finger could perform shortcuts, secondary actions, and even improve multitasking [74] or text entry [75]. Prior work required wearable sensors [74, 75, 152], sequential data such as gestures [144], pre-defined targets [29], or temporal features [275] to differentiate between a set of fingers (*e.g.*, left/right thumb). In contrast, we use *capacitive images* to identify fingers within single frames independent from context, position, and additional sensors. We collected a data set comprising of capacitive images for each finger and empirically studied finger combinations which can be differentiated with a usable accuracy. While a feature engineering approach with basic machine learning achieved inferior results, we present a user/position-independent deep learning model to differentiate between

left and right thumbs with over 92 % accuracy. We evaluated it with novel use cases that users find intuitive and useful. Moreover, we publicly release our data set and models to enable future work using and improving finger identification on commodity smartphones.

4.2 Input Technique I: Palm as an Additional Input Modality (*PalmTouch*)

This section describes the development and evaluation of the *PalmTouch* interaction technique following the steps 3 to 5 of the UCDDL.

4.2.1 Data Collection Study

To implement the use cases based on a palm recognition, the touchscreen needs to differentiate between finger and palm touches. Previous work used the 2D touch location provided by the touchscreen which is either limited through latency [153, 211] or requires at least two touch points (pen and palm) [211, 227]. In contrast, we use capacitive images provided by the touchscreen which contain low-resolution fingerprints of the touch (*e.g.*, finger or palm). Since we apply machine learning to classify the touch, we conducted a user study to collect labeled touch data of fingers and the palm while participants perform representative touch actions. With this data, we train and evaluate a palm detection model to differentiate between touches from fingers and palms.

Study Design & Tasks

The purpose of this study is to collect a wide variety of finger and palm touch input. We designed six different tasks which instruct each participant to perform a total number of 240 representative touch actions. The first five tasks in Figure 4.3 (*finger tasks*) require participants to use their fingers whereas the rightmost task (*palm task*) instructs participants to place their palm on the screen (see Figure 4.1). The order of the finger tasks was randomized, and a palm task was performed after each *finger task* in an alternating order. Each *finger task* was performed 15 times.

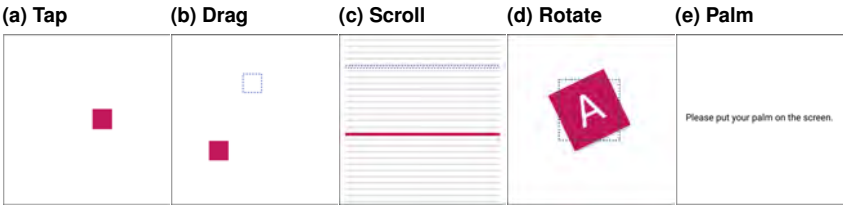


Figure 4.3: All six tasks performed by participants in the data collection study.

Participants performed these tasks in two conditions, ONE-HANDED with the thumb as the main input finger and TWO-HANDED with the index finger as the main input finger. We conducted these tasks to capture different finger and palm touches in our data set. In both conditions, participants had to perform *tapping*, *dragging*, and *scrolling* movements. The TWO-HANDED condition also cover *zooming* and *pinching* movements. After each task, participants placed their palm on the touchscreen until they were told to remove it by the apparatus. We counterbalanced the variant of the opposite-side palm touch between participants (*flat hand* and *forming a fist*). We instructed participants to place their palm as if that would activate an function, such as launching the application drawer.

Participants & Study Procedure

We recruited 22 participants (5 female) between the ages of 21 and 34 (M = 25.1, SD = 3.2). All participants were right-handed. The average hand size was measured from the wrist crease to the middle fingertip, and ranged from 17.0 cm to 21.9 cm (M = 19.2 cm, SD = 1.6 cm). Our collected data comprise samples from the 5th and 95th percentile of the anthropometric data reported in prior work [191]. Thus, the sample can be considered as representative.

After participants signed the consent form, we measured their hand size. We then explained the procedure including the palm input modality as shown in Figure 4.1 and handed them an instruction sheet which explains all tasks of the study. We asked participants to repeatedly try out the movements until they felt

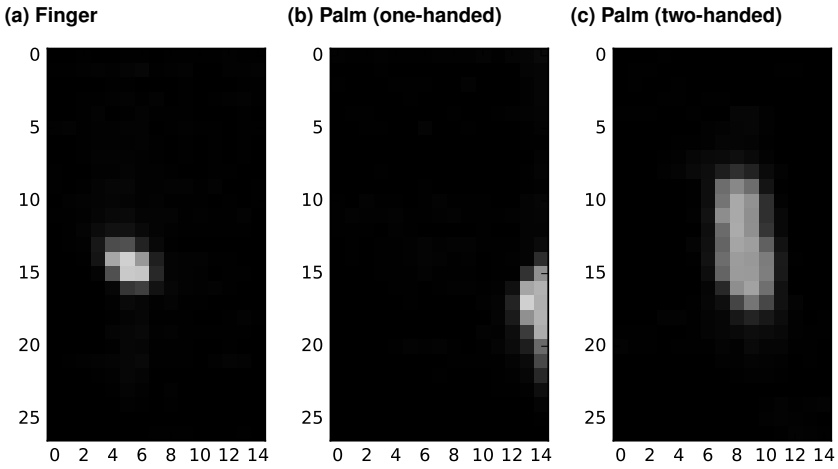


Figure 4.4: Exemplary raw capacitive images from the data collection study. Figure (a) shows the finger of participant 5 during the dragging task; (b) shows the palm of participant 19 in the one-handed condition and (c) shows the palm of participant 9 in the two-handed condition.

comfortable to repeat a palm touch at any given moment. Participants performed all tasks in 20 minutes on average and were rewarded with sweets for their participation.

Apparatus

We used an LG Nexus 5 running Android 5.1.1 with a modified kernel to access the 15×27 raw capacitive image of the Synaptics ClearPad 3350 touch sensor (see Figure 4.4). 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 for the tasks described above which logs a capacitive image every $50\,ms$ ($20\,fps$). Each image is logged with the respective task name so that every touch is automatically labeled.

4.2.2 Model Development

Based on the collected data set, we train a model to classify touches as being made by a finger or a palm. We will first show simple approaches based on feature engineering and established machine learning algorithms known from previous HCI work. Afterwards, we show that representation learning techniques such as neural networks (NNs) and convolutional neural networks (CNNs) outperform the simpler approaches regarding the classification accuracy. Models and test results are summarized in Table 4.1.

Data Set & Preprocessing

After filtering empty (due to not touching) and erroneous images (which do not contain the expected number of touches) to avoid wrong labeling, we have a data set comprising 138,223 capacitive images which represent blobs of valid touches. We extended the data set with flipped versions (vertical, horizontal, and both) of all remaining capacitive images to train the model for different device orientations. To train a position-invariant model and enable classification of multiple blobs within one capacitive image, we performed a blob detection, cropped the results and pasted each blob into an empty 15×27 matrix (referred to as *blob image*). 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. In total, our data set consists of 552,892 blob images. We trained and tested all models with a participant-wise split of 80% to 20% (18:4) to avoid samples of the same participant being in both training and test set.

Feature Exploration and Basic Machine Learning

Due to their prominence in previous HCI work (*e.g.*, [82, 220, 274]), we trained and evaluated palm touch models based on Support Vector Machines (SVMs), k-nearest neighbors (kNNs), Decision Trees (DTs) and Random Forests (RFs). In contrast to representation learning approaches [17], these algorithms require the training data to be processed into features (*i.e.*, feature engineering). Using scikit-learn 0.18.2[1], we trained different models and performed a grid search as

[1] `scikit-learn.org/0.18/documentation.html`

proposed by Hsu *et al.* [101] to determine the most suitable hyperparameters. If we did not report a hyperparameter, we applied the standard value as reported in scikit-learn's documentation. Since the palm's average contact area on the touchscreen ($M = 932.06 mm^2$, $SD = 503.17 mm^2$) is larger than the finger's ($M = 164.86 mm^2$, $SD = 50.77 mm^2$), we first used the blob area as a single feature to classify the touch. We determined the blob area by fitting an ellipse around the blob[1]. With an accuracy of 96.80% (prec = 97.66%; rec = 92.05%), the DT (with $max\_depth = 4$ to avoid overfitting) achieved the highest accuracy of the aforementioned algorithms. After experimenting with a wide range of additional features including the ellipse parameters and the capacitance represented by the blob, we found that a feature set comprising the ellipse (area, width and height) and the capacitance (mean and sum) achieved the highest accuracy of 98.17% (prec = 96.10%, rec = 98.18%) using an RF.

Representation learning algorithms learn features in part with the labeled input data and have been shown to be more successful than manual feature engineering for image data [17]. Thus, we implemented a multilayer feedforward neural network using *TensorFlow*[2] and performed a grid search over different network configurations, including the number of neurons in steps of 50, layers in steps of 1, activation functions, and optimizers provided by *TensorFlow*. Our final network architecture is shown in Table 4.1. Training was done with a batch size of 100 using the Adaptive Gradient Algorithm (AdaGrad) [51] with an adaptive learning rate starting from 0.001. We initialized the network weights using the Xavier initialization scheme [65]. While we experimented with L2 Regularization and batch normalization [103], this did not improve the overall accuracy. We achieved an accuracy of 98.74% (prec = 97.49%, rec = 98.46%) with this network configuration.

PalmTouch using a Convolutional Neural Network

CNNs are the recent state-of-the-art method for image classification [115]. As blobs are represented by low-resolution images, we implemented a CNNs using

[1] 2D least squares estimator for ellipses: `scikit-image.org/docs/dev/api/skimage.measure.html#skimage.measure.EllipseModel`

[2] `www.tensorflow.org/`

features	algorithm	parameters / layers	prec	rec	acc
–	Baseline (ZeroR)	Always predicting `Finger` as this is the majority class.	-	0.0	68.54
finger blob size	kNN	$k\ (neighbors) = 197$	97.90	91.78	96.80
	DT	$max\ depth = 4$	97.66	92.05	96.80
	RF	$estimators = 1$; $max\ depth = 1$	97.64	92.06	96.80
	SVM	$C = .1$; linear kernel	98.99	90.54	96.73
ellipse & capacitance	kNN	$k\ (neighbors) = 9$	98.40	94.34	97.73
	DT	$max\ depth = 6$	97.10	96.70	98.05
	RF	$estimators = 16$; $max\ depth = 10$	96.10	98.18	98.17
	SVM	$C = 10$; linear kernel	99.96	79.06	93.40
raw data (RL)	NN	*Input*: 405 *Hidden Layer 1*: 500 (ReLU) *Hidden Layer 2*: 300 (ReLU) *Softmax (output)*: 2	97.49	98.46	98.74
	CNN	*Input*: $27 \times 15 \times 1$ *Convolution*: $7 \times 7 \times 16$ (ReLU) *Max Pooling*: 2×2 (stride = 2) *Convolution*: $7 \times 7 \times 36$ (ReLU) *Max Pooling*: 2×2 (stride = 2) *FC Layer 1*: 350 (ReLU) *FC Layer 2*: 350 (ReLU) *Softmax (output)*: 2	99.38	99.28	99.58

Table 4.1: Performance of the trained models with the hyperparameters after a grid search for the highest accuracy. Results (in percent) were calculated using the test set described above.

TensorFlow. We performed a grid search over the number of layers, filters and their sizes in steps of 1, the number of neurons in the fully connected layer in steps of 50, as well as activation functions and optimizers provided by *TensorFlow*. Our final network architecture is shown in Table 4.1. We trained the CNN using AdaGrad as the optimizer with a batch size of 100 and used the Xavier initialization scheme to initialize the network weights. We initialized the biases with random values from a normal distribution. An exponential decay (rate = 0.2 in 1000 steps) was used to decrease the initial learning rate of 0.009. We used L2 Regularization to compensate overfitting by adding 0.01 of the weights to the cost function. Moreover, we used an early stopping approach as proposed by Caruana *et al.* [31] to further avoid overfitting. While we experimented with batch normalization [103], this did not improve the overall accuracy. Our CNN achieved an accuracy of 99.58% (prec = 99.38%, rec = 99.28%) which is the highest of all presented approaches.

Mobile Implementation

After freezing the CNN to a protocol buffer file, we used *TensorFlow Mobile*[1] for Android to run the CNN on an LG Nexus 5 that provides the same capacitive images as in the data collection study. Classifying one capacitive image including the blob detection takes 7.5 *ms* on average ($min = 4\,ms$, $max = 11\,ms$, $SD = 1.6\,ms$) over 1000 runs. As this is faster than the sampling rate for the capacitive images, it can be used to classify each sample when running in the background. With processor manufacturers recently starting to optimize their processors for machine learning (*e.g.*, Snapdragon 835), the classification can be sped up significantly[2]. The model can be further optimized for mobile devices with techniques such as quantization [78] and pruning [7] for a small loss of accuracy.

Discussion

We presented an overview of machine learning algorithms which we used to train a palm classifier and showed that a CNN achieved the highest classification accuracy of 99.58%. This improves the baseline accuracy by 31.0%. While our grid search already yields reasonable results for the basic machine learning approaches, further optimizing accuracy and especially precision is necessary as the palm classifier is supposed to run in the background to classify a large number of input frames over time (*i.e.*, 20 frames per second). As fingers are used most of the time to perform input on the touchscreen while a detected palm triggers a defined action, false positives (affecting the precision score) lead to a visible unexpected behavior. In contrast, a lower recall score (and thus a higher false negative rate) could be partly compensated by the UI through, *e.g.*, recovering previous wrongly classified palm touches as soon as the palm is correctly detected. Thus, we prioritized precision over recall in the training process. While the SVM with ellipse and capacitance properties as features achieved the highest precision of all approaches, the trade-off is by far the lowest recall and also accuracy. In

[1] `www.tensorflow.org/mobile/`

[2] `www.qualcomm.com/news/onq/2017/01/09/tensorflow-machine-learning-now-optimized-snapdragon-835-and-hexagon-682-dsp`

total, the CNN achieved the best results while the preparation and classification are feasible to perform on an off-the-shelf mobile device. We will refer to this model as *CNN-PalmTouch*.

4.2.3 Evaluation

We conducted a study to evaluate *PalmTouch* and the model accuracy in realistic scenarios. Specifically, we focus on the following three aspects: 1) classification accuracy of *CNN-PalmTouch* in realistic scenarios, 2) qualitative feedback after using *PalmTouch*, and 3) a quantitative evaluation of the reachability use case. We used a Nexus 5 running the mobile version of *CNN-PalmTouch* described above and custom applications to record and implement the study scenarios.

Study Procedure & Design

We designed four tasks to evaluate the three aspects described above. We measured the participants' hand and finger sizes after we obtained informed consent and then handed them an instruction sheet that explained all parts of the study so that participants could refer to the instructions at any time.

Part 1 (Realistic Scenario for Evaluating False Positives) In a counterbalanced order, we instructed participants to perform tasks one-handed and two-handed which we refer to as realistic scenarios. While participants used the smartphones, we collected the classifier output in the background to test the model on false positives as palms are not expected in this part. We designed the realistic scenarios to cover commonly used touch input gestures including tapping, dragging, scrolling, and additionally pinching and rotating for two-handed use. To keep the scenarios as realistic as possible, participants performed this part on a pure Android system using common applications such as the onboard SMS messenger, Google Chrome, and Maps.

We handed the Nexus 5 in standby mode to the participant who received a (simulated) notification after unlocking the device. Tapping the message in the notification drawer then opens a text message with questions that the participant needs to answer by using Google searches or Maps. We further provided an

instruction sheet that describes each step that the participant is required to do. The gestures described above were used especially using the system (tapping and scrolling), selecting text on a website (long press and dragging), and navigating in Google Maps in the two-handed scenario (pinching and rotating). Each of the two scenarios ended with replying to the initial SMS message with the search results. In total, this part took 10 minutes per participant.

Part 2 (Realistic Scenario for Qualitative Feedback) We introduced and demonstrated *PalmTouch* to the participants and let them practice the same-side and one of the opposite-side palm touches using a demo application. Afterwards, participants performed a modified version of the Part 1 scenarios. Instead of pulling down the notification bar, participants now use the palm to access the notifications. Further, we replaced all application switching actions with the pie menu containing the messaging, browser and maps application. After completion, participants filled out a questionnaire and we interviewed them about their impression of *PalmTouch*. In total, this part took around 15 minutes per participant.

Part 3 (Reachability Task) We evaluated the reachability use case regarding the task completion time (TCT) and qualitative feedback. Specifically, we compared accessing notifications supported by *PalmTouch* with dragging down the notification bar manually. We used a 2×2 within-subjects design with the independent variables being the number of hands (ONE-HANDED and TWO-HANDED) and the access method (PALM and DRAG). Each condition comprised 20 repetitions of opening the notification drawer to click on a notification displayed at a random height. Between these repetitions, participants completed 2 - 5 Fitts' Law tasks as shown in Figure 4.5b to ensure that they returned to their usual hand grip after clicking the notification. We measured the TCT for opening the notification drawer and clicking on the notification. We further collected qualitative feedback about the perceived easiness, speed, success, accuracy, and comfort using a 7-point Likert scale.

The apparatus simulates a notification bar (see Figure 4.5a) for the respective conditions. To simulate the DRAG condition as realistic as possible, the notifica-

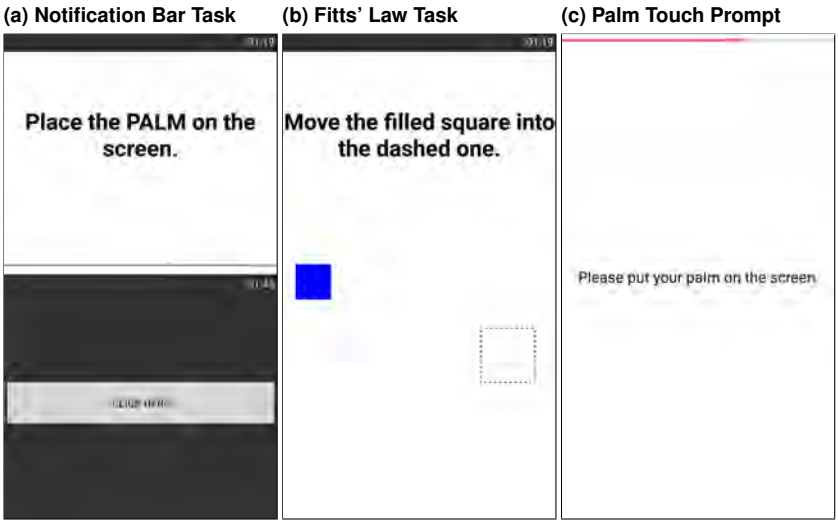


Figure 4.5: Screenshots of (a) the notification drawer in Part 3; (b) Fitts' Law task as a distraction task in Part 3; and (c) a prompt to place the palm on the touchscreen until the progressbar on top is full (Part 4).

tion drawer can also be opened with a fling downwards. By rooting the Nexus 5, we disabled Android's notification bar and navigation bar to avoid any disruptions during this part. This part took 10 minutes.

Part 4 (Palm Touch Input for Evaluating False Negatives) We tested the model on false negatives. Our study application instructed participants to repeatedly place their palm on the screen for one second and remove it afterward (see Figure 4.5c). The duration ensures that participants contribute a similar number of samples and avoids palm touches being done too quickly or slowly. Both same-side and opposite-side palm touches were performed 20 times each in a counterbalanced order. We collected the classifier output during this part to test the model on false negatives as fingers are not expected in this part. We let

participants perform this part at the end of the study since repeatedly placing the palm on the touchscreen and waiting could lead to fatigue and therefore influence the other parts. This part took 5 minutes.

Participants

We recruited 22 participants (6 female) with an average age of 21.9 ($SD = 2.1$) who had not participated in the previous study. All except two participants were right-handed. The average hand size measured from the wrist crease to the middle fingertip ranged from $17.2\,cm$ to $20.5\,cm$ ($M = 18.6\,cm$, $SD = 0.9\,cm$). Three participants preferred to use their smartphone two-handed, while 14 preferred to use it one-handed and five use both variants in everyday life.

Results

We present the results of the evaluation study which covers model accuracy, evaluation of the reachability use case, heatmaps of how *PalmTouch* was used, and qualitative feedback.

Model Accuracy in Realistic Scenarios We obtained labeled capacitive images of touch input in a realistic scenario. With the labels providing us with the number of false positives and true negatives (Task 1), and true positives and false negatives (Task 4), we derived the precision, recall, and accuracy of the classifiers. After calculating all three metrics for each participant, the mean accuracy yielded by *CNN-PalmTouch* is 99.53 % ($SD = 0.71\,\%$). The mean precision is 99.35 % ($SD = 2.53\,\%$) and the mean recall is 97.94 % ($SD = 2.87\,\%$). The false positive rate which describes the likelihood of unintentionally triggering a palm input is 0.09 %.

Reachability Use Case Evaluation For the ONE-HANDED condition, the average time for DRAG to open the notification drawer is $1689.62\,ms$ ($SD = 700.32\,ms$) while the total time including tapping the notification is $2352.74\,ms$ on average ($SD = 817.14\,ms$). In contrast, the average time for PALM to open the notification drawer is $1396.31\,ms$ ($SD = 449.75\,ms$) and $2114.73\,ms$ ($SD = 647.14\,ms$)

including tapping the notification. A two-tailed paired t-Test revealed a significant difference in TCT to open the notification drawer in the PALM and DRAG condition ($t_{329} = 6.71$, $p < .001$) and in TCT including tapping the notification ($t_{329} = 4.40$, $p < .001$). We used a Wilcoxon signed-rank test to analyze the Likert scores as shown in Table 4.2 for the ONE-HANDED condition and found a statistically significant difference between PALM and DRAG in perceived easiness ($Z = -2.201$, $p = .028$), speed ($Z = -1.985$, $p = .047$), success ($Z = -2.069$, $p = .039$) and accuracy ($Z = -2.087$, $p = .037$). No statistically significant difference was found for the perceived comfort ($Z = -.508$, $p = .612$).

For the TWO-HANDED condition, the average time for DRAG to open the notification drawer is 1462.54 *ms* ($SD = 873.14$ *ms*) while the total time including tapping the notification is 2103.94 *ms* on average ($SD = 1049.05$ *ms*). In contrast, the average time for PALM to open the notification drawer is 1394.29 *ms* ($SD = 588.76$ *ms*) and 2110.40 *ms* ($SD = 742.37$ *ms*) including tapping the notification. A two-tailed paired t-Test showed neither a significant difference in TCT between the PALM and DRAG condition to open the notification drawer ($t_{329} = 1.19$, $p = .236$) nor in the TCT including tapping the notification ($t_{329} = -0.09$, $p = .925$). We used a Wilcoxon signed-rank test to analyze the Likert scores for the TWO-HANDED condition and did not find a statistically significant difference between PALM and DRAG neither in perceived easiness ($Z = -.626$, $p = .531$), speed ($Z = -.019$, $p = .985$), success ($Z = -1.562$, $p = .118$), accuracy ($Z = -.894$, $p = .371$), nor comfort ($Z = -1.326$, $p = .185$).

Type and Location of Palm Placement Figure 4.6 shows heatmaps of the locations at which participants performed palm touches in task 4, and indicate the touches' average position. All three images represent the average capacitive images over each participant. We separated the capacitive images into three palm input types that we showed in Figure 4.1: same-side (as shown in Figure 4.1a), opposite-side using the flat hand (Figure 4.1b), and opposite-side by forming a fist (Figure 4.1c). Nine participants decided to use the flat hand for opposite-side palm touch and 13 participants the fist.

Figure 4.6: Average capacitive images representing the location of the palm touches for (a) same-side, (b) opposite-side using the flat hand, and (c) opposite-side by forming a fist. All images describe the average capacitive images over each participant.

Qualitative Feedback in Realistic Scenarios Participants filled out a SUS questionnaire [28] about *PalmTouch* as an additional input modality in the two realistic scenarios. SUS scores range between 0 to 100 whereas any score above 68 is considered to be above average in terms of usability [27]. The SUS score for *PalmTouch* ranged from 52.5 to 95.5 with an average of 80.1 (SD = 10.0). With this, the average score lies between "good" and "excellent" in the adjective rating of Bangor *et al.* [14]. Realistic scenarios without *PalmTouch* yielded an SUS score of M = 74.0 (SD = 16.1).

When asked about the first impression after using *PalmTouch* in realistic scenarios, the majority (18) were positive about using the palm as an additional input modality. Especially when used one-handed, participants found using the palm intuitive and natural (P7, P10, P11, P12, P13, P14, P20), comfortable (P3,

Perception	One-Handed		Two-Handed	
	Palm	Drag	Palm	Drag
Easiness *	5.7 (1.1)	4.8 (1.6)	6.0 (1.2)	5.6 (1.7)
Speed *	5.6 (1.2)	4.6 (1.8)	5.4 (1.6)	5.2 (1.8)
Success *	6.0 (1.4)	5.1 (1.6)	6.2 (1.0)	5.5 (1.6)
Accuracy *	5.7 (0.9)	5.0 (1.4)	5.9 (1.0)	5.4 (1.6)
Comfort	4.5 (1.6)	4.5 (1.9)	6.1 (1.0)	5.4 (1.5)

Table 4.2: Subjective perceptions (7-point Likert scale) to open the notification drawer in both conditions. Values in brackets indicate the SD, an asterisk (*) indicate a statistically significant difference between one-handed PALM and DRAG ($p < .05$).

P6, P9) and convenient (P2, P3, P13). While participants needed a short period of time to get familiar with the input modality, they (P12, P13, P15, P21) appreciate that it helps them to use the system faster than before ("*It felt strange for a short while, but then I became familiar with it really fast. After that, it feels intuitive, and I am faster than without it.* " - P21). Moreover, they were surprised about the stability of the system ("*I was surprised that the system worked really well, especially for app switching.*" - P13; "*It worked well*" - P9). In contrast, four participants reportedly had concerns to perform a palm touch ("*I was afraid to drop the phone*" - P22) or "*had the feeling that [they] touch something on the screen unintentionally*" when used the first time (P17).

When asked about advantages of *PalmTouch*, eight participants reportedly find the provided shortcuts useful. They identified that these shortcuts provide faster access to apps (P9, P11, P12, P17, P18, P19) and improve reachability, especially when using the device one-handed (P7, P10, P20). As an example for the faster access, P16 explained that the most important apps are "*always available when placing the palm onto the touchscreen*". Further, P7 suggested that "*systems could use palm input to [allow users to] access the app drawer from every screen*" to make launching apps faster. Three participants (P7, P10, P20) argued that *PalmTouch* saves grip changes as "*shortcuts help to reach buttons on the top side of the touchscreen*" (P20).

We asked participants about perceived disadvantages of *PalmTouch*. For *PalmTouch* in the one-handed condition, only two participants (P3, P7) reported

that when "*tapping something on the upper left edge of the device, one could accidentally place the palm on the screen*" (P3) which could be solved with the reachability use case. In general, participants see more advantages than disadvantages for *PalmTouch* when they use the device one-handed. In contrast, they reported more disadvantages after the two-handed scenarios. Holding the device with one hand and placing the palm of the second hand on the touchscreen feels unintuitive (P2, P12, P16) and unnatural (P5, P6, P7, P11). As an example, P12 explained that "*switching from index finger to the palm requires either moving the hand or turning it*" which makes it inconvenient. Further, three participants (P3, P20, P22) argued that it is faster to use the index finger of the second hand to reach targets on the top side of the touchscreen instead of the palm. Since two-handed interaction does not pose reachability challenges, participants found that *PalmTouch* was less useful in the two-handed scenarios.

We asked participants for further scenarios in which *PalmTouch* can be useful. They preferred and suggested the possibility to start custom applications and actions (P2, P3, P6, P11, P14, P15, P17, P18), such as the camera (P2, P6), settings within an application (P11, P17) or splitting the screen (P18) which is shipped with Android 7.0. P1 and P22 even suggested mapping more critical functions since they find it unlikely to trigger a function mapped to the palm accidentally. These functions include closing the foreground application (P1), accepting a call (P20), or stopping the music (P22).

Discussion

We implemented *PalmTouch* and deployed it on an off-the-shelf smartphone to enable users to trigger specific functions by placing the palm on the touchscreen. The palm as an additional input modality received a SUS score of 80.1 which is considered above average in terms of usability [27]. The SUS score conforms with subjective feedback of participants who found *PalmTouch* intuitive and natural as a way to improve reachability and as a shortcut. Using the notification bar as an abstract scenario of the reachability problem, we found that participants perceive *PalmTouch* as significantly easier, faster and more accurate than changing the grip which could lead to dropping the device. For the one-handed scenarios, an analysis of the task completion time (TCT) revealed that *PalmTouch* is indeed significantly

faster than a grip change to open the notification drawer manually. This finding can be transferred to other interface elements such as toolbars, the application's menu button, and URL bars in most browsers. Further, with the emergence of edge-to-edge displays on devices such as the iPhone X and Samsung Galaxy S8, the lack of a dedicated home button can be compensated with *PalmTouch*.

Participants gave more positive feedback for *PalmTouch* during the one-handed scenario. The reason is that two-handed interaction does not pose any reachability challenges since the interacting hand can move freely over the whole display. Moreover, placing the other hand's palm on the display feels reportedly less subtle and thus can feel unusual. In both scenarios, all except two participants had no difficulties to place their palms on the touchscreen after a short practice phase. Due to small hand sizes (17 *cm*), two participants lack a stable grip while holding the device one-handed. Moreover, bending the *thenar muscles*[1] to place the palm on the touchscreen causes the hand to bend. Thus, all other fingers move towards the palm which leads to an unstable grip while the device is tilted. In this situation, a controllable input is not possible since the device needs to be balanced. However, this also applies to stretching the thumb or changing the grip. Thus, we recommend *PalmTouch* as an additional input modality while still providing alternative touch input in case the user cannot ensure a stable hand grip.

We implemented *PalmTouch* using capacitive images of the touchscreen and trained a CNN which achieves a high accuracy in detecting the palm. Compared to basic machine learning approaches and neural networks, the CNN achieved the highest accuracy with an applicable classification time when deployed on an off-the-shelf LG Nexus 5. We showed that our model classifies touches reliably during realistic scenarios with an accuracy of 99.53%. With a precision of 99.35%, the likelihood of unintended triggers either through classification errors or unwanted palm input is neglectable. With a recall of 97.94%, our classifier also recognized the palm reliably when users placed them on the screen. False negatives were caused by capacitive images in which the palm was about to be placed on the screen (in Task 4), and can be corrected by recovering previous wrongly classified palm touches as soon as the palm is correctly detected. The accuracy can be further improved by taking the average locations of palm input

[1] Thenar muscles refers to a group of muscles located at the base of the thumb [107].

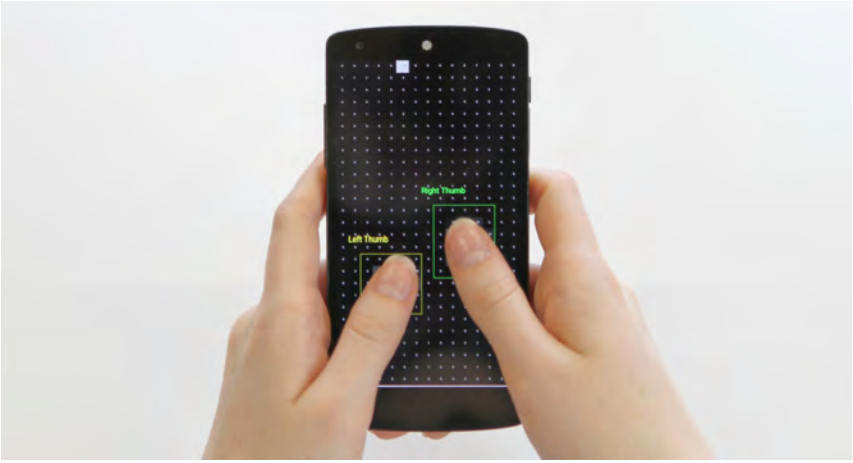


Figure 4.7: Identifying left and right thumbs on a commodity smartphone using the raw capacitive data of touches.

as shown in Figure 4.6 into account. Since accuracies in offline validation and realistic scenarios are similar, this shows that our model is generalizing well and does not overfit. In summary, this shows that using the palm to activate functions is feasible with a high accuracy while perceived as natural and fast by users.

4.3 Input Technique II: Finger Identification

This section describes the development and evaluation of finger identification models following the steps 3 to 5 of the UCDDL.

4.3.1 Data Collection Study

We conducted a user study to collect labeled touch data while participants performed representative touch actions. This data enables us to train and evaluate models based on supervised learning for distinguishing between different fingers. We adopted the study design which we presented in Section 4.2 which also described tasks that cover typical touch gestures such as tapping, scrolling, and dragging to include representative actions.

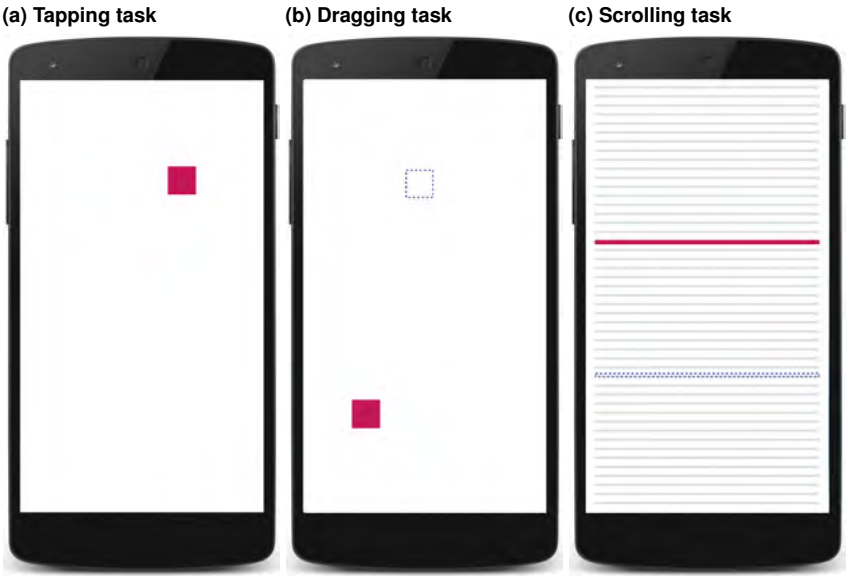


Figure 4.8: Tasks adapted from Section 4.2.1 to cover typical touch inputs.

Study Design & Tasks

To record representative touch input, we instructed participants to perform three different tasks with each of the ten fingers. The tasks are shown in Figure 4.8 and include a *tapping task* in which participants tapped and held the target for 1.5 seconds to generate sufficient data; a *scrolling task* in which a red line needs to match a blue line (horizontal and vertical); and a *dragging task* in which participants dragged a tile into a target. The targets and shapes appeared at randomized positions.

We used a 10×3 within-subjects design with the independent variables being the fingers and the tasks. With each finger, participants performed 30 repetitions of all three tasks resulting in $10 \times 30 \times 3 = 900$ tasks per participant. We further

Figure 4.9: The study apparatus showing a participant solving a scrolling task with the index finger.

divided the 30 repetitions of the scrolling task into 15 vertical and 15 horizontal tasks to cover all scrolling directions. The order of fingers was balanced using a Latin square while the tasks appeared in a shuffled order.

Participants & Study Procedure

We recruited 20 participants (15 male and 5 female) with an average age of 22.4 ($SD = 2.8$). All except two were right-handed. The average hand size measured from the wrist crease to the middle fingertip ranged from 16.3 cm to 20.8 cm ($M = 18.9\,cm$, $SD = 1.3\,cm$). Our data includes samples from the 5th and 95th percentile of the anthropometric data [191]. Thus, the participants can be considered as representative.

After obtaining informed consent, we measured the participants' hand size and handed them an instruction sheet which explained all three tasks. Participants were seated on a chair without armrests and instructed to hold the device one-handed when touching with the thumb, and two-handed for all other fingers. We

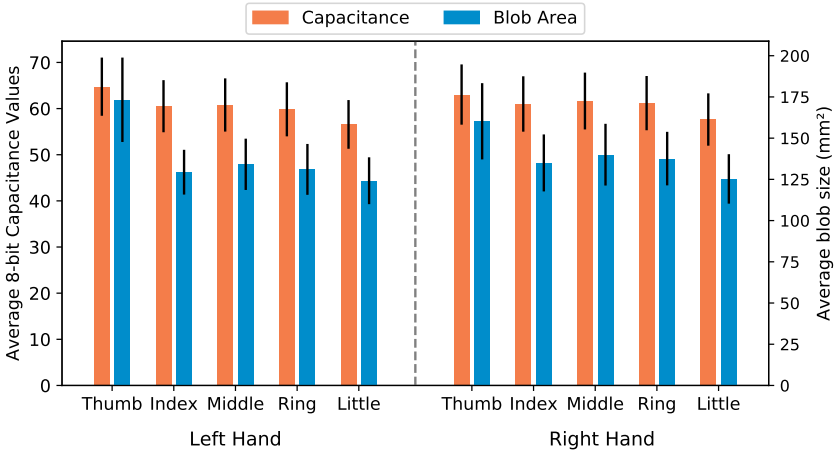


Figure 4.10: The average capacitance and blob size for each finger.

instructed participants to hold the device in the same angle for all fingers (*i.e.* the angle they used first) to avoid the models potentially overfitting to the angle between device and fingers (*e.g.*, participants shifting their grip or changing their body posture after a condition). On average, participants finished all tasks within 45 minutes including optional breaks. We reimbursed participants with 5 EUR for their participation.

Apparatus

We used an LG Nexus 5 with a modified kernel as described in Section 4.2 to access the 15×27 8-bit capacitive image of the Synaptics ClearPad 3350 touch sensor. Exemplary images of the raw capacitive data are shown in Figures 4.4 and 4.7, where 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 for the tasks described above which logs a capacitive

image every $50\,ms$ ($20\,fps$). Images were logged with the respective task name and finger so that each touch was automatically labeled. Figure 4.9 shows the study apparatus.

4.3.2 Model Development

We present the gathered data set and describe three steps towards developing finger identification models: (1) cleaning the data set, (2) exploring the data set to understand distinctive features between touches of individual fingers, and (3) using deep learning to train models for finger identification.

Data Set & Preprocessing

We collected 921,538 capacitive images in the data collection 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. To train a position-invariant model and enable classification of multiple blobs within one capacitive image, we performed a blob detection, cropped the results and pasted each blob into an empty 15×27 matrix (referred to as *blob image*). 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. In total, our data set consists of $455,709$ blob images ($194,571$ while tapping; $111,758$ while dragging; $149,380$ while scrolling).

Feature Exploration

We visually inspected the generated touch blobs of each finger during all tasks to find distinctive features. Figure 4.11 shows average touch blobs for each finger including the blob size and distribution of the measured capacitance. We generated these images by upscaling the capacitive images by a factor of 5 using the Lanczos4 algorithm [233] to increase clarity of the capacitance distribution. We then cropped the blobs and overlayed them for each finger. To describe the blobs, we fitted an ellipse around them using a 2D least squares estimator

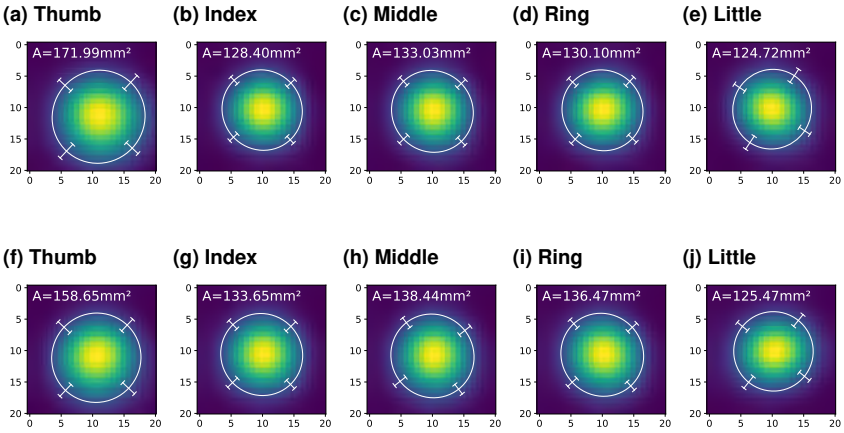


Figure 4.11: Average capacitive image for touches of each finger upscaled by a factor of 5 (for clarity purposes). Fitted ellipses represent the average area of touches in *mm* and the orientation θ thereof in degrees. The bars represent the standard deviation of the minor-axis a and major-axis b.

for ellipses[1]. The resulting ellipse parameters (minor-axis a, major-axis b, and orientation θ) in *mm* are averaged and shown in Table 4.3. We further explored the ellipse areas ($A = \pi * a * b$) and the average measured capacitance of a blob. We determined the average capacitance by averaging all electrode measurements of a blob larger than 0. Figure 4.10 shows the average capacitance (8-bit) and average blob size (in *mm*).

Similar to previous work [63, 265], we used all five features (*i.e.*, mean capacitance, the ellipse area, a, b, and θ) to explore whether basic machine learning models based on feature engineering are sufficient for finger identification. For the sake of clarity, we focused on random forests over which we performed a grid search to find the best hyperparameters for each combination of fingers. Results are reported in Table 4.4 (see RF column) and are inferior to deep learning algorithms.

[1] `scikit-image.org/docs/dev/api/skimage.measure.html#skimage.measure.EllipseModel`

Hand	Finger	Count	a		b		θ	
			M	SD	M	SD	M	SD
Left	Thumb	50,897	7.32	1.27	7.48	1.47	43.05	49.77
	Index	41,379	6.51	0.74	6.28	0.82	46.62	52.72
	Middle	39,079	6.64	0.84	6.38	0.91	46.09	52.03
	Ring	44,718	6.55	0.86	6.32	0.93	43.31	53.03
	Little	45,794	6.21	1.00	6.39	1.24	33.57	53.06
Right	Thumb	44,674	7.07	1.28	7.15	1.27	43.37	52.72
	Index	46,507	6.60	0.91	6.45	1.06	46.04	52.76
	Middle	47,082	6.73	0.95	6.55	1.10	51.86	49.33
	Ring	47,229	6.71	0.88	6.47	0.96	47.55	49.07
	Little	48,350	6.33	1.04	6.31	1.19	38.80	50.02

Table 4.3: Parameters of all fitted ellipses. Parameters a and b represent the length of minor and major semi-axes (in *mm*). θ represents the ellipse rotation in a counter-clockwise orientation in degrees.

Finger Identification using Convolutional Neural Networks

Deep learning algorithms such as CNNs learn features in part with the labeled input data and have been shown to be more successful than manual feature engineering [17]. Thus, we implemented CNNs using *Keras* (based on the *TensorFlow* backend) and performed a grid search as proposed by Hsu *et al.* [101] to determine the model parameters that achieve the highest test accuracy for all models as shown in Table 4.4. If we do not report a hyperparameter in the following, we applied the standard value (*e.g.*, optimizer settings) as reported in Keras' documentation. We started our grid search based on a CNN architecture which previous work found to perform the best on *capacitive images* [123, 136]. We performed our grid search as follows: We experimented with the number of convolution and dense layers in steps of 1. For the convolution part of the CNN, we varied the kernel size in steps of 1×1 and number of filters in steps of 16. For the dense layers, we experimented with the number of neurons in steps of 32. Moreover, we adapted the dropout factor in steps of 0.1. Figure 4.12 shows the final network architecture which achieved the highest test accuracy.

We trained the CNNs using an RMSprop optimizer [229] (similar to AdaGrad [51] but with a less radical approach to decrease the learning rate) with a batch size of 100. Further, we used the Xavier initialization scheme [65] to initialize the network weights. We used L2 regularization with a factor of 0.05, a

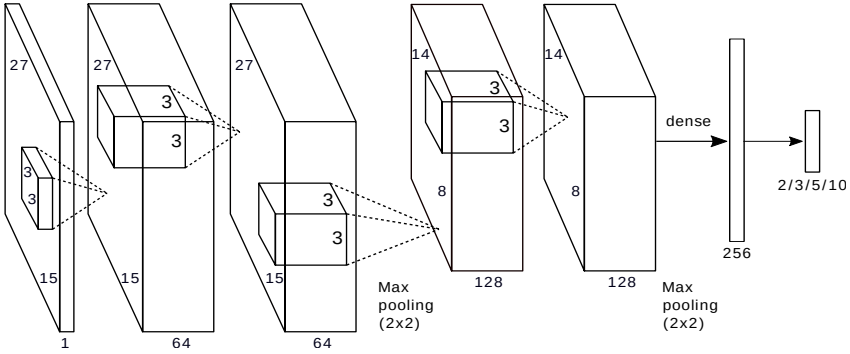


Figure 4.12: General architecture used after performing an initial grid search for all finger combinations listed in Table 4.4.

0.5 dropout after each pooling layer and the dense layer, and Batch Normalization to prevent overfitting during training. Our model expects a 15×27 *blob image* as input and returns the probability of each class (*i.e.* finger) as output.

Model Accuracies

Table 4.4 shows the models that we trained and their accuracies on a test set. We trained and tested all models with a participant-wise split of 80% to 20% (16:4) to avoid samples of the same participant being in both training and test set.

The THUMB L/R and INDEX L/R models differentiate between touches of the respective finger from the left hand and the right hand. While the INDEX L/R model achieved an accuracy of 65.23 %, the THUMB L/R model discriminates left and right thumbs with an accuracy of 90.12 %. Differentiating between thumb and index finger independent from the hand (THUMB/INDEX) is feasible with an accuracy of 84.01 %. Similarly, identifying whether a touch was performed by the thumb or any other finger (THUMB/OTHERS) yields an accuracy of 86.44 %.

Identifying touches from the left or the right hand (HAND L/R) is feasible with an accuracy of 59.23 %. We further explored the differentiation between three fingers (*i.e.* thumb, index, and middle finger) similar to previous work by Gil *et al.* [63]. With our TRITAP model, we improved their accuracy by 2.92 %

	Full	Tap	M_{user}	SD_{user}	Min_{user}	Max_{user}	ZeroR	RF	#
THUMB L/R	90.12	93.14	88.61	7.18	72.17	97.30	52.97	66.20	2
INDEX L/R	65.23	64.31	88.63	7.39	67.37	99.87	51.21	54.34	2
THUMB/INDEX	84.01	81.81	89.11	5.77	74.95	98.04	54.04	73.59	2
THUMB/OTHERS	86.44	88.89	84.52	12.62	48.37	95.55	78.92	79.91	2
HAND L/R	59.27	62.18	63.34	15.99	37.83	89.70	50.90	50.54	2
TRITAP	67.17	70.92	82.12	6.63	68.67	95.44	31.73	56.54	3
5 FINGERS	46.13	47.15	64.35	7.86	48.87	79.07	21.08	32.14	5
10 FINGERS	35.55	37.86	67.95	7.44	58.67	83.91	11.60	17.93	10

Table 4.4: Accuracies for differentiating between finger combinations. The first two columns show the accuracy on the test set based on a participant-wise 80%:20% (16:4) split. The third to sixth columns show user-based accuracies averaged over participants with a 80%:20% split (sorted by timestamp). ZeroR represents the baseline accuracy (using most frequent label) and RF represents the accuracy of random forests and feature engineering. # represents the number of classes for the respective model.

which results in 70.92 %. Increasing the number of fingers to identify decreases the accuracy. A hand-independent finger identification (5 FINGERS) leads to an accuracy of 46.13 % while additionally differentiating between hands (10 FINGERS) yields an accuracy of 35.55 %.

In addition, we trained models using a subset of the data set consisting of touches of the tapping task (*Tap Data*). Similar to Gil *et al.* [63], we achieved improvements in accuracy of up to 3.75 % compared to the full data set. Moreover, we trained models for each participant (*user-based models*) using their full datasets with a 80%:20% split sorted by timestamps. This increased the average accuracy by up to 32.4 % while reaching maximum accuracies of 80 % to 99 % per user. The improvements are substantial for 10 FINGERS, 5 FINGERS, TRITAP and INDEX L/R but not for models such as THUMB L/R with an already high accuracy. Out of all models, the THUMB L/R and THUMB/OTHERS achieved the highest accuracy.

Mobile Implementation

After freezing and exporting the trained model into a protocol buffer file, we used *TensorFlow Mobile* for Android to run the CNN on an LG Nexus 5. A classification including blob detection and cropping takes 19.2 *ms* on average

($min = 12\,ms$, $max = 25\,ms$, $SD = 4.2\,ms$) over 1000 runs. As this is faster than the $20\,fps$ sampling rate for the capacitive images, the inference can be performed on each sample in the background. Since recent processors (*e.g.*, Snapdragon 845) are optimized for machine learning, the classification time can be reduced to a neglectable duration[1]. The model can be further optimized for mobile devices with techniques such as quantization [78] and pruning [7] for a small loss of accuracy.

Discussion

We started the model development by exploring the data set and training random forests based on features derived from the capacitive images. The results did not reveal any distinctive features which basic machine learning algorithms could use for finger identification. Thus, we applied CNNs to develop models to differentiate between combinations of fingers. The achieved accuracies are shown in Table 4.4.

As expected, the model for identifying 10 FINGERS leads to an accuracy of 35.55 %, which is not practical for interaction. Confirming previous work by Gil *et al.* [63], this indicates that the information provided by the low-resolution capacitive images does not reveal enough differences between the fingers. To improve upon this, we then combined the same fingers of both hands into one class (5 FINGERS model) to achieve a higher accuracy (46.13 %). However, when considering the improvement factor over the baseline as suggested by Kostakos and Musolesi [113], we found that this factor decreases when combining fingers of both hands (2.1 for 10 FINGERS, 1.2 for 5 FINGERS). Similarly, combining all fingers of a hand into one class (HAND L/R) leads to an accuracy of 59.27 % but with an even lower improvement factor of 0.2. Moreover, discriminating thumbs from other fingers (THUMB/OTHERS) resulted in an improvement factor of 0.1. This further suggests that combining touches from multiple fingers into one class leads to more overlaps between classes and a decrease of accuracy improvements over the baseline. These results suggest that involving multiple fingers and classes in the classification leads to accuracies that are not sufficient for interaction.

To improve the accuracy, we explored models to differentiate between the two fingers mainly used for input: THUMB L/R, INDEX L/R, and THUMB/INDEX.

[1] `www.qualcomm.com/snapdragon/artificial-intelligence`

While INDEX L/R and THUMB/INDEX achieved accuracies of 65.23 % and 84.01 % respectively, THUMB L/R achieved the highest accuracy of all models (90.12 %). The high accuracy of the THUMB L/R model could be due to different reasons. We observed that the thumb does not touch the display in a nearly perpendicular angle as other fingers do. This results in a larger contact surface which provides more information for classification. Amongst others, this includes the thumb's yaw angle (angle between thumb and vertical axis of the touchscreen) which is different for touches of the left and the right thumb (see yellow hotspots in Figure 4.11). While this works for the CNN, the pure orientation of the blob is not enough to use for basic ML algorithms due to the high standard deviation.

In an interaction scenario, fingers should be identified directly after touching the display. Since the first touch is always a tap, we trained models using only the tap data. We achieved accuracy improvements of up to 3 % (*e.g.*, 93.14 % for THUMB L/R) as moving fingers add additional noise, especially at a lower frame rate. We further explored user-based models as collecting touches for on-device training works similar to the setup of fingerprint scanners. While THUMB L/R did not improve, the 10 FINGERS model improved by over 32 %. The accuracy increase for user-based models could be explained by individual postures (*e.g.* orientation) of each finger which resulted in differentiable touch shapes. Our models can be applied to other devices by retraining or scaling the raw data.

In summary, we found that reducing the number of fingers to identify increases the overall accuracy. While identifying all 10 fingers is not sufficiently accurate on capacitive touchscreens of commodity smartphones, differentiating between the left and right thumb is feasible with an accuracy of over 92 %. This extends the touch input vocabulary as the second thumb can be used for secondary actions, similar to the right mouse button. Moreover, previous work showed that using both thumbs is already a common posture for most users [52, 54, 129, 187]. In addition to an offline validation, we demonstrate the usefulness of our THUMB L/R model, suitable use cases, and the model's accuracy during real use cases on a commodity smartphone in the following.

4.3.3 Evaluation

We conducted a study to validate the model's accuracy and to evaluate our sample applications with users. We focused on the following two aspects: 1) model validation with new participants and thus a dataset that was not involved in training and test, and 2) collecting qualitative feedback on the sample use cases and the concept of *thumb-aware interaction*.

Study Procedure and Design

We designed three tasks to evaluate the two aspects described above. After we obtained informed consent from participants, we measured their hand sizes and collected demographic data. We handed them an instruction sheet that explained all parts of the study so they could refer to the instructions at any time.

Part 1 (Model Validation) In this part, we collect the validation set to evaluate the model performance with data from different participants than the ones used to train and test the model. We used the same tasks as in the data collection study (see Figure 4.8) and instructed participants to perform dragging, tapping, and scrolling tasks in a randomized order. All tasks were performed with the left and the right thumb in a counterbalanced order so that we could collect ground truth labels for the validation of the THUMB L/R model. Additionally, participants filled in a raw NASA-TLX questionnaire [72, 84] to compare the perceived workload with results from part 2.

Part 2 (Abstract Porous Interface) In addition to the first part, we evaluate the effective accuracy which includes the model's classification accuracy and human errors. The human error describes the user's error-proneness to use the correct fingers to solve the tasks. To do so, we used the porous interface application to instruct participants to solve dragging and scrolling tasks with different thumbs. To collect ground truth labels for accuracy evaluation, new targets appear as soon as the previous target was filled (*e.g.*, in Figure 4.13b a new target for dragging appeared after the scrolling target was filled). Thus, the current task (*e.g.*, dragging

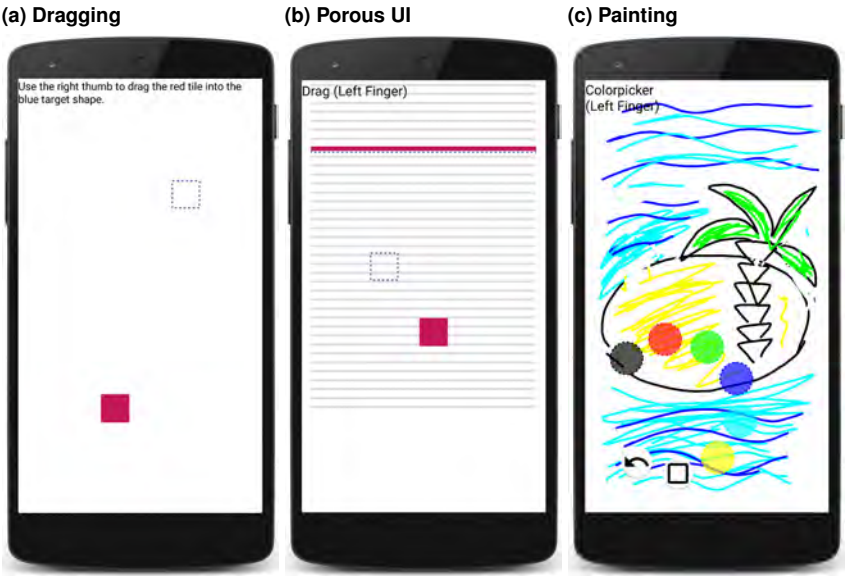


Figure 4.13: Screenshots of (a) a dragging task in part 1; (b) a combined dragging and scrolling task as an abstract porous interface in part 2; (c) the drawing application in part 3 with a pie menu for color selection.

in Figure 4.13b) can be used as the ground truth label. We asked participants to fill in a NASA-TLX questionnaire to assess the perceived workload for using the correct thumb to solve the task.

Part 3 (Painting Application) To evaluate the THUMB L/R model in a concrete scenario, we used the painting application shown in Figure 4.13c in which users can draw using the right thumb and use the left thumb for secondary tools (*e.g.*, erasing or selecting colors using a pie menu). Similar to the previous part, the upper left corner displays which thumb was recognized and thus which action the user is performing. We use this part to collect qualitative feedback from participants on the concept of *thumb-aware interaction* on a commodity smartphone. The qualitative feedback includes a questionnaire for ratings, and an

interview focused on the advantages and disadvantages of the interaction method. Further, we asked for use cases that participants envisioned for *thumb-aware interaction* on smartphones.

Participants

We recruited 10 participants (6 male, 4 females) with an average age of 24.1 ($SD = 3.0$) who had not participated in the previous study. All participants were right-handed. The average hand size measured from the wrist crease to the middle fingertip ranged from 17.3 *cm* to 21.0 *cm* ($M = 18.5\,cm$, $SD = 1.1\,cm$). We reimbursed participants with 5 EUR for their participation.

Results

We present the evaluation results which covers a model validation, the effective (model and human) accuracy in an abstract use case, and qualitative feedback on *thumb-aware interaction*.

Model Validation Based on the collected capacitive images of new participants, the THUMB L/R model (trained with full data) achieved a mean accuracy of 89.78 % ($SD = 3.30\,\%$, $min = 84.90\,\%$, $max = 96.50\,\%$). The mean precision for detecting the left thumb was 88.72 % ($SD = 4.43\,\%$, $min = 82.31\,\%$, $max = 95.68\,\%$) and the recall was 89.85 % ($SD = 3.90\,\%$, $min = 82.12\,\%$, $max = 95.87\,\%$). Pearson's correlation test did not reveal a significant correlation between the hand size and accuracy ($r = -0.03$, $p = 0.94$).

A validation of the THUMB L/R model (trained with tap data) with the tap data subset resulted in a mean accuracy of 91.98 % ($SD = 5.24\,\%$, $min = 81.98\,\%$, $max = 99.23\,\%$). The mean precision for detecting the left thumb was 90.80 % ($SD = 4.40\,\%$, $min = 85.29\,\%$, $max = 98.84\,\%$) and the recall was 91.77 % ($SD = 7.81\,\%$, $min = 77.15\,\%$, $max = 99.48\,\%$). Again, Pearson's correlation test did not reveal a significant correlation between hand size and accuracy ($r = -0.04$, $p = 0.92$).

Effective Accuracy in Porous Interfaces Based on the performed task as ground truth (*i.e.*, scroll or drag), the following results represent the effective accuracy

when considering both model and human errors. Human errors occured when participants mistake, *e.g.*, the left for the right thumb for the scroll task. Therefore, these results describe the accuracy that one can expect when also considering the error-proneness of users to use the correct thumb for the tasks.

When classifying touches using the THUMB L/R model (trained with full data), the effective accuracy was 85.16 % (*SD* = 3.50 %, *min* = 78.16 %, *max* = 91.36 %) with a precision of 86.77 % (*SD* = 3.60 %, *min* = 81.19 %, *max* = 92.34 %) and recall of 84.17 % (*SD* = 4.74 %, *min* = 74.03 %, *max* = 89.96 %) for detecting the left thumb. Pearson's correlation test did not reveal a significant correlation between the participant's hand size and classification accuracy ($r = -0.46$, $p = 0.18$).

Subjective Feedback We present the subjective feedback on the use cases. For the interviews, two researchers employed a simplified version of qualitative coding with affinity diagramming [79] by coding the answers, printing them on paper cards, and finally clustering the answers.

Perceived Workload Ratings: We used a raw NASA-TLX questionnaire [72] to assess participants' perceived workload after using the abstract porous interface. Moreover, we assessed the perceived workload after part 1 as a comparison. Mauchly's Test of Sphericity indicated that the assumption of sphericity had not been violated, $\chi^2(2) = .745$, $p = .689$. A one-way ANOVA with repeated-measures does not reveal any statistically significant differences ($F_{2,18} = 2.711$, $p = .093$) between the perceived cognitive load when using the left hand ($M = 13.3$, $SD = 9.2$), right hand ($M = 7.3$, $SD = 7.3$), or both hands for the abstract porous interface task ($M = 11.2$, $SD = 6.1$).

Interview: When asked about the first impression after using *thumb-aware interaction*, the majority (8) provided positive feedback. In particular, participants found it useful in general ("*very useful*" - P7), for painting applications ("*it is faster, especially since one can switch color with the left hand*" - P1), for multitasking purposes ("*very useful, especially to use two apps simultaneously*" - P5), and to avoid unintended touches ("*one can not activate something unintentionally*" - P4). They commended the

idea ("*cool and innovative idea*" - P10) especially for the abstract porous interface task ("*the first task is easier to solve with two thumbs*" - P5) and the painting task ("*makes painting easier, even quite good when holding the device slightly different*" - P1). Moreover, they (6) found the interaction method intuitive ("*more intuitive [than without]*" - P7) and easy to learn ("*I am already used to using both thumbs. This makes learning this interaction method easier.*" - P6).

Confirming the perceived workload ratings, participants found that they had no difficulties to coordinate the thumbs for the two layers of the porous interface ("*I had no cognitive difficulties*" - P2, "*Needed to get used to in the beginning, but then it became easy*" - P4). Only one participant (P3) mentioned that it might be "*confusing to focus on two things simultaneously*". While two participants were impressed by the finger identification accuracy ("*Recognition was already very good - there were only two cases in which my finger was wrongly identified.*" - P5), other (6) participants clearly noticed it when fingers were wrongly identified ("*little bit frustrating since false recognitions leads to [unintended lines] that needs to be erased*" - P7). However, in the porous interface task, such identification errors could be "*easily fixed by touching the display once again*" (P5). Further, P5 noted that he "*quickly learned how to [place] the thumbs to control [the user interface]*".

When asked about use cases which they envision for *thumb-aware interaction*, all participants were unanimous about multitasking and shortcuts as the main use case. Moreover, they suggested using the interaction method for mobile games and image editing. For example, applications could offer multiple modes that make use of the porous interface concept (P9, P10) to avoid manual switches. Further, *thumb-aware interaction* could be used to interact with 3D objects so that each finger manipulated one dimension (P2, P5, P9). This would also benefit mobile games so that each finger could be assigned to one joystick or button so that fixed positions for control elements would not be required (P1, P4, P6). When asked about which fingers participants would use if all 10 fingers could be recognized with a sufficient accuracy, participants were unanimous that the thumb is the main

finger for interacting with smartphones. Further, 4 participants considered the index finger for interaction while 2 would additionally consider the middle finger. To interact with tablets on a table, all participants would use all fingers while one participant further suggested using knuckles. In general, nine participants would use the concept of thumb-aware interaction on their devices ("*definitely, if apps support it*" - P4) while one would not.

Discussion

We conducted a user study to validate the accuracy of the THUMB L/R model with participants who had not participated in the data collection study. Further, we combined the model validation with an evaluation of two use cases that we implemented using *thumb-aware touch interaction*. This includes an abstract scenario of porous interfaces initially proposed by Gupta *et al.* [74], and a painting application in which the right thumb can draw while the left thumb is responsible for the settings (*e.g.*, color and tool selection).

Model Validation Accuracy and Qualitative Feedback The model validation resulted in accuracies similar to the results achieved in the offline validation with the test set. This suggests that the THUMB L/R model generalizes well across different users and does not overfit. We also considered human errors (*i.e.*, mixing up between fingers) together with the model accuracy which resulted in an effective accuracy of 85.16 %. The 5 % difference in contrast to the model validation could be due to a number of reasons. Human errors are inevitable especially since users are not yet fully familiar with this interaction concept. This conforms with the statements in the interview. Further, there are technical limitations that affect the accuracy of this live scenario. Due to the low retrieval rate of the capacitive images (20*fps*), the classification could have happened on images in which the thumb was still in motion so that it just barely touched the display. While one solution could be using multiple frames and taking the most predicted class, this would have introduced latency.

Despite a decrease of 5 % accuracy in a live scenario, participants were positive about the use cases for *thumb-aware interaction* and argued that wrong classifications could be fixed effortlessly by placing the finger on the display

again. One participant even mentioned that he learned how to place the thumb on the screen to avoid wrong classifications after the first iterations. The qualitative feedback revealed that participants were unanimously positive about the interaction method and that it can be a useful addition to the touch input vocabulary. Moreover, the ratings showed that interacting with porous interfaces using *thumb-aware interaction* does not increase the perceived workload. This suggests that interacting with two applications simultaneously can be intuitive for users and further avoids repeatedly switching between applications or splitting the screen which decreases the interaction space. Shortcuts (*e.g.*, pie menu for color selection) were perceived as intuitive and can save screen space that is used for menu bars otherwise. However, wrong identifications are reportedly more noticeable in this use case.

Improving the Classification Performance While the thumb models (*i.e.*, THUMB L/R, THUMB/INDEX, and THUMB/OTHERS) achieved accuracies well beyond the 80 % that previous work considered sufficient in general [113], sufficiency also depends on the action's consequence (*e.g.*, easily recoverable action vs. permanent action) and how classifications are translated to actions. While the consequence depends on the application/developer, we discuss translation approaches and improvements that can further minimize accidental activations to a neglectable amount in the following.

Instead of translating a single classification result into an action, previous work showed that taking the majority class of a set of results noticeably improves the accuracy (*i.e.*, majority voting [120]). Since multiple results are considered, single incorrect results (*e.g.*, due to outliers) can be compensated. This is especially useful for recoverable actions and scenarios that provide enough time to gather multiple classifications (*e.g.*, finger identification while performing a gesture). Further, a threshold for the confidence score [147] of the most likely class could be used to avoid incorrect translations due to similarities. In case of a low confidence score, a touch could be either omitted with a warning to the user, or a fallback function could be activated that can easily be recovered. Especially with

recoverable functions in case of a wrong identification, the system can collect touch data in the background to continuously improve the finger identification model using on-device learning.

Our approach is solely based on capacitive images to investigate the feasibility of identifying fingers within a single frame and independent from context and position. Finger identification, in general, could be improved with additional context information from the touchscreen or additional sensors. The touch position provides more information about the finger's yaw angle for thumb identification since distant touches (*e.g.*, close to top edge) lead to larger contact surfaces due to a stretched thumb. Similarly, touch offsets on smaller targets (*e.g.*, right thumb tends to hit the right of the target and vice versa for the left thumb) represent an additional feature to predict hand postures [29]. Further, gestures (*e.g.*, unlock trajectories) could be used to detect the handedness of users [144] and combined with the majority voting approach described above. Sequential models (*e.g.*, recurrent neural networks (RNN) and long short-term memory (LSTM)) can be trained with sequences of capacitive images (*i.e.,* trajectories of touches) to consider the information that gestures provide for detecting handedness.

Besides software-based approaches, touchscreens with a higher sensing resolution could be used. The Samsung SUR40 display offers touch images in a higher resolution based on IR sensing which contain more signal to improve the classification accuracy. However, such touchscreens need yet to be produced and incorporated into mass-market mobile devices. Not only are they more complex to manufacture but would also need more resources to be operated. Further improvements includes pre-touch sensing [93] to sense the finger above the display or built-in inertial measurement units [66, 89, 213].

4.4 General Discussion

In this chapter, we presented two novel touch-based interaction techniques for capacitive touchscreens as incorporated in commodity smartphones. We followed the UCDDL as presented in Section 1.2.3 to develop and evaluate the interaction techniques in three steps: (i) a data collection study to gather ground truth data for the model, (ii) the modeling phase in which we explore the data set and features,

as well as train a deep learning model, and (iii) an evaluation in which we validate the model accuracy and evaluate the interaction technique in a real scenario based on our implementation.

4.4.1 Summary

Addressing RQ3, we presented a deep learning based approach and showed that hand parts (*e.g.*, palm) and fingers (*e.g.*, left and right thumb) can be differentiated with a high and usable accuracy. For the data collection study, we designed tasks to gather a representative data set on which we train the recognition model. In contrast to a typical evaluation in the field of deep learning which uses a validation set for the final model evaluation, our approach is more HCI-oriented which consists not only of a validation with a new set of participants but also an evaluation of the model and interaction method in a real use scenario. This also enables us to evaluate the usability (*e.g.*, noise of estimations over time for none to small variations) of the model which is not evaluated by simply determining the accuracy. In summary, we showed that our approach enables to differentiate between the main input finger (*e.g.*, right thumb) and an additional hand part or finger with a high and usable accuracy. Despite this success, the signal of individual touches on a capacitive touchscreen is not sufficient to enable the identification of all fingers.

A solution to the limited signal is to involve more information about the hand itself into the model inference. This would assume that the full hand is touching the device which is the case in a single-handed grip. Since the degrees of freedom of a human hand is limited, a model could use the constraints of the hand movements to infer the position of the other hand parts. However, this would require a smartphone which senses capacitive touch input on the whole device surface and provides the capacitive images for inference.

4.4.2 Lessons Learned

Based on the developed recognition models and results of the evaluation study, we derive the following insights:

Simple touch source identification improves the input vocabulary. An evaluation of our novel interaction techniques using binary classifiers based on deep learning and the raw capacitive data revealed that they extend the touch input vocabulary in a meaningful way. Differentiating between touches of palm and finger enable additional shortcuts to frequently used functions, and improve reachability especially during single-handed input. Moreover, differentiating between left and right thumbs improve multitasking, add an additional input modality (comparable with left and right mouse buttons), and enable further use cases such as 3D navigation, UI components with multiple functions, and handedness-aware UIs. Using our interaction techniques in real use scenarios showed that users perceive them as useful, as well as natural and intuitive to use.

Simple touch source identification is feasible with a high accuracy. Based on the raw capacitive data and deep learning, we showed that differentiating between two touch sources is possible with a high and usable accuracy. Palm and finger touches can be identified with an accuracy of 99.57 % in a real scenario, while left and right thumbs are identified with an accuracy of 93.14 %. Our pipeline and publicly released notebooks enable other researchers to easily re-implement our models or use our pipeline to differentiate between other touch sources.

A single touchscreen is not sufficient to identify all fingers. Despite the high accuracy for differentiating between two touch sources (*e.g.*, palm/fingers, and left/right fingers), the signal contained in capacitive images is not sufficient to identify all five or ten fingers as shown in the previous section and conforming to previous work [63]. We conclude that more information about the hand (*e.g.*, capacitive images of the hand grasping the device including the position of all other fingers) is required to differentiate between touches of all fingers. By sensing input on the rear, this would also enable all other fingers to perform finger-aware input in a single-handed grip.

4.4.3 Data Sets

An outcome of the two studies in this chapter are two data sets which consists of labeled capacitive images representing touches from palms as well as all ten fingers. By publicly releasing our data sets, we enable the community to re-produce our interaction techniques similar to algorithm-based contributions which are described detailed enough for the reader to re-implement it. We are publicly releasing the data sets together with Python 3.6 scripts to preprocess the data as well as train and test the models described in this chapter under the MIT license. We further provide the trained models, the software to run our models, and implementations of the use cases readily deployable on Android. These will enable the community to run our models on their devices. We hope that the provided models in combination with the dataset can serve as a baseline that enables other researchers to further improve the accuracy.

1. PalmTouch: `https://github.com/interactionlab/PalmTouch`
2. Finger Identification:
   `https://github.com/interactionlab/CapFingerId`

5

Hand-and-Finger-Awareness on Full-Touch Mobile Devices

In Chapter 3, we investigated which fingers can be used for single-handed interaction and their comfortably reachable area beyond the touchscreen. In the previous chapter, we presented an approach in which we use the raw capacitive data to differentiate between touches of fingers and palms. In this chapter, we combine these findings and develop a fully touch sensitive smartphone prototype which is capable of identifying fingers with thw raw capacitive measurements on the whole device surface. Based on this prototype, we further collaborated with experienced interaction designers to elicit novel techniques to solve the limitations of current touch interaction.

Parts of this chapter are based on the following publication:

H. V. Le, S. Mayer, and N. Henze. "InfiniTouch: Finger-Aware Interaction on Fully Touch Sensitive Smartphones." In: *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. UIST '18. Berlin, Germany: ACM, 2018. ISBN: 978-1-4503-5948-1. DOI: 10.1145/3242587.3242605[a]

H. V. Le, S. Mayer, P. Bader, F. Bastian, and N. Henze. "Interaction Methods and Use Cases for a Full-Touch Sensing Smartphone." In: *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems.* CHI EA '17. Denver, Colorado, USA: ACM, 2017. ISBN: 978-1-4503-4656-6. DOI: 10.1145/3027063.3053196

[a] Video Figure: https://www.youtube.com/watch?v=0vvZwMJCyVU

5.1 *InfiniTouch*: Finger-Aware Input on Full-Touch Smartphones

To enable hand-and-finger-aware input on the whole device surface, we develop *InfiniTouch*, a system for finger-aware interaction on a smartphone's whole device surface. Our prototype has the form factor of a standard smartphone to avoid influencing the usual hand grip of users [133], and does not require external hardware. We use the contact areas on the whole device surface to train a CNN for finger identification. The model achieved an accuracy of 95.78 % for identifying fingers on the device surface while estimating their 3D position with a mean absolute error (MAE) of 0.74 *cm*. Besides simple grip recognition, our model also enables fingers to perform implicit as well as explicit input. We implemented multiple use cases to showcase the performance of our model and the usefulness of finger-aware interaction on the whole device surface of smartphones during one-handed interaction.

5.1.1 Full-Touch Smartphone Prototype

We developed a full-touch smartphone prototype that provides capacitive images for a finger identification model. We used two LG Nexus 5 as basis which provides capacitive images with a resolution of 27×15 *px* on the front and back side. As using the full hardware of both smartphones (*i.e.*, stacking the devices) lead to a noticeable increase in thickness, we separated the prototype into two modules to gain flexibility in form factor: a *Handheld Device*, and a *Hardware Container*. Each module is comprised of a self-designed printed circuit board (PCB) that act as extension adapters to connect the *Handheld Device* (PCB_{HD}) and *Hardware Container* (PCB_{HC}) with each other via flexible flat cables (FFC).

Figure 5.1: Our full-touch smartphone prototype based on two LG Nexus 5 and a Genuino MKR1000 for touch sensing on the edges.

Instead of manufacturing proprietary sensors, we based our prototype on two commodity smartphones and release the schemes of our self-designed PCB so that the community can re-implement our prototype. This enables further exploration of interaction techniques based on finger-aware interaction on the whole device surface. The PCB schemes, 3D models, component descriptions, and source code to reproduce our prototype are available on our project page[1] under the MIT license.

Handheld Device

The *Handheld Device* (see Figures 5.1 and 5.2a) consists of a 3D printed frame, two Nexus 5 touchscreens, 37 copper plates as capacitive touch sensors on the edges, and a PCB_{HD}. The 3D printed frame holds both touchscreens and encloses PCB_{HD}. The capacitive touch sensors are fixated on the left, right and bottom side of the frame. Each touch sensor is a $6 \times y \times 0.5\,mm$ (with $y = 6\,mm$ for left and right, and $y = 12\,mm$ for the bottom side) copper plate which is glued

[1] `https://github.com/interactionlab/InfiniTouch`

(a) Handheld Device (b) Hardware Container

Figure 5.2: Full-touch smartphone prototype: (a) the *Handheld Device*, and (b) *Hardware Container* containing the processing units. Both components are connected via our self-designed PCB and flexible flat cables.

into engravings of the frame with a gap of $1.0\,mm$ in between and sanded down for a smooth feeling. The copper plates are connected to the PCB_{HD} which in turn comprises three MPR121 capacitive touch controllers operated by a Genuino MKR 1000 microcontroller in the *Hardware Container*. Similarly, both touchscreens are connected via a board-to-board connector on the PCB_{HD} and are operated by the remaining components of the Nexus 5 located in the *Hardware Container*. The two FFCs are routed through the top side of the *Handheld Device* as there is a low likelihood that it disturbs the user when holding the phone in a usual grip [129, 130]. The dimensions of the *Handheld Device* are $137.6 \times 68.7 \times 8.9\,mm$ ($115\,g$). In comparison, the dimensions of an off-the-shelf Nexus 5 are $137.8 \times 69.1 \times 8.6\,mm$ ($130\,g$).

Hardware Container

The *Hardware Container* (see Figure 5.2b) is a 3D printed box that contains two Nexus 5 circuit boards and batteries, a Genuino MKR 1000 microcontroller, three

micro USB breakout boards, and two tactile buttons. The circuit board of the Nexus 5 is connected to a compatible board-to-board connector on PCB_{HC} which in turn is connected to the touchscreens. To access the power buttons and USB ports of the two Nexus 5, we replaced them with tactile buttons and USB micro breakouts integrated in the *Hardware Container*. Moreover, we extended the Genuino's USB port to a USB micro breakout board. The Genuino MKR 1000 is connected to the PCB_{HC} to operate the side sensors connected to the PCB_{HD}, and can be powered by battery via the JST connector or through USB.

(a) Motion Capture Setup **(b) Marker Placement**

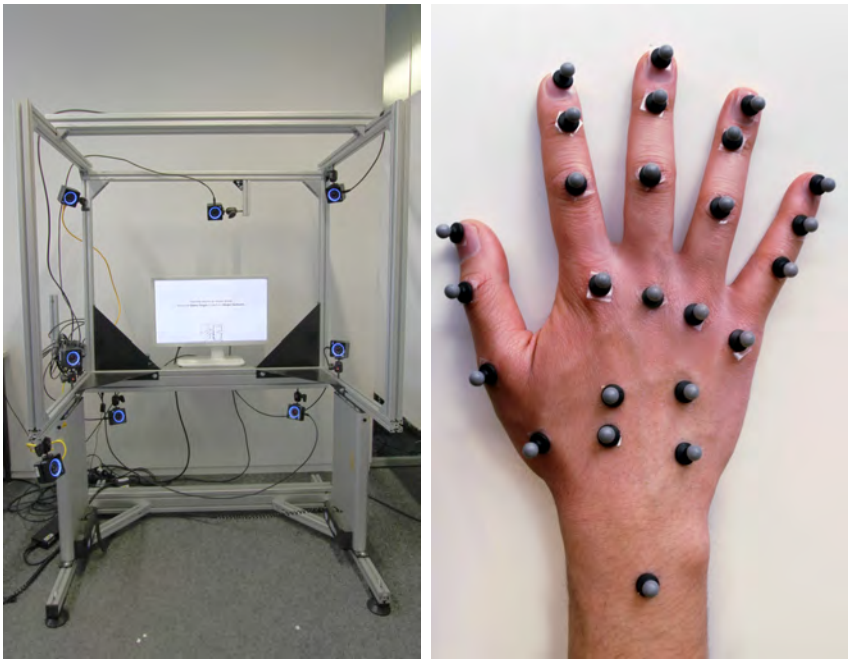


Figure 5.3: Study setup: (a) motion capture system with 8 cameras mounted on a aluminum profiles and (b) reflective markers ($6.4\,mm$ spheres) attached to the hand of a participant.

5.1.2 Ground Truth Data Collection

Using our prototype, we conducted a study to collect a dataset comprising the capacitive images and respective 3D motion data of each joint of the hand. The former will be the input for our model and the fingertips of the latter the output. Participants performed finger movements starting from five hand grips as shown in Figure 5.4 to cover possible finger positions.

Capacitive Images and Interconnection

We accessed the $27 \times 15\,px$ capacitive images of the front and back touchscreen by modifying the Android kernel. Each pixel corresponds to a $4.1 \times 4.1\,mm$ 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 used I2C calls to access the register for test reporting as described in the RMI4 specification (511-000405-01 Rev.D) and the driver's source code[1]. We pulled the capacitive images from the debugging interface with $20\,fps$ and stored them in the proc filesystem (procfs) to make them accessible in the application layer. As the edge sensors are square electrodes, we simply read their values with the MPR121 library to retrieve a capacitive image.

To generate a merged capacitive image of the sensor values on all sides, the Nexus 5 responsible for the front opens a WiFi hotspot to receive the values from the Nexus 5 on the back and the Genuino MKR 1000. The transfer latency measured by an average round trip time over 1000 samples is $7.2\,ms$ ($SD = 2.6\,ms$). As the capacitive images can be pulled from the debugging interface with $20\,fps$ at most, the transfer latency can be neglected. Data from side sensors can be retrieved at $130\,fps$. We developed an Android library that retrieves the capacitive images, establishes a connection between front, back and side, and provides a callback function in which developers can retrieve the merged capacitive images.

[1] `github.com/CyanogenMod/android_kernel_lge_hammerhead/blob/cm-12.1/drivers/input/touchscreen/touch_synaptics_ds5.c`

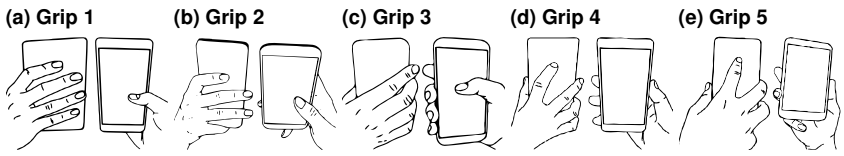


Figure 5.4: Five HAND GRIPS used in the study and adopted from previous work [139].

Apparatus

To record finger motions with sub-millimeter accuracy, we used an *OptiTrack* motion capture system with eight cameras (*OptiTrack* capturing at 240*fps*). The cameras were firmly mounted to an aluminum profile grid as shown in Figure 5.3a. To enable these infrared cameras to record the finger movements, we attached 25 reflective markers (6.4*mm* spherical markers with skin adhesive M3 base) on all joints of the hand similar to previous work [58, 130] and as shown in Figure 5.3b. Additionally, we attached four markers on the top part of the full-touch smartphone which enables us to track the device in six degrees of freedom. We installed a display in front of the participant to show instructions (see Figure 5.3a).

Design

The study has three independent variables, HAND GRIP, FINGER and TASK. For HAND GRIP we used known hand grips that were shown in previous work [139] and in Figure 5.4. For FINGER, we used all five fingers of the right hand. As tasks, we used *free movements*, in which participants freely moved a specified finger; *swipe gestures*, in which participants performed swipe gestures into left, right, bottom and up directions; and *free placements with thumb* in which participants placed the specified finger followed by a thumb movement to simulate using fingers on the rear as modifiers.

The three independent variables result in a $5 \times 5 \times 3$ within-subject design. We counterbalanced the GRIP using a balanced Latin square and used a random order for FINGER and TASK. The duration of each task was 30 seconds which results in a total duration of $30\,sec \times 5 \times 5 \times 3 = 37.5$ minutes. During these tasks,

participants were surrounded by eight motion capture cameras and were seated on a chair without armrests as shown in Figure 5.3a. Including the briefing, optional breaks, and attaching markers, the study took around 60 minutes.

Procedure

After we obtained informed consent, we collected demographic data using a questionnaire and measured the participants' hand size and finger lengths. We then proceeded to attach 25 skin adhesive markers on their right hand to enable motion tracking. Instruction slides were shown on the display which explains the procedure of the study as well as the finger movements and hand grips that participants should perform. We further showed them a demo of the required movements and asked them to perform it on trial to ensure that everything was fully understood.

After handing the full-touch smartphone to the participants, they first imitated a grip shown on the instruction display and were then instructed to perform the shown task. While the displayed finger specifies the main finger to move, we allowed the participants to also move other fingers if this was necessary to move the main finger. This is necessary to record hand grip states that are as realistic as possible (*e.g.*, the ring finger can only be moved individually to a lesser extent [76]). The described process was repeated for all HAND GRIPS, FINGERS, and TASKS. The experimenter monitored the markers throughout the study to ensure that the finger was moved at an adequate speed and that all markers are visible in the motion capturing.

Participants

We recruited 20 participants (7 female) between the ages of 20 and 29 (M = 24.1, SD = 2.5). All participants were right-handed. The average hand size was measured from the wrist crease to the middle fingertip and ranged from 15.6 cm to 25.0 cm (M = 19.3 cm, SD = 2.0 cm). Our collected data comprise samples from the 5th and 95th percentile of the anthropometric data reported in prior work [191]. Thus, the sample can be considered as representative. Participants were reimbursed with 10 EUR for their participation.

5.1.3 Finger Identification Model

We train a model to estimate the fingertip locations using the capacitive images as input. The model output contains an estimated 3D location for each finger which can also be used to identify the source of the contact areas (referred to as *blobs*).

Data Set & Preprocessing

We synchronized the motion data with the capacitive images of the front, back, and edges of the full-touch smartphone. We used the capacitive images as input and the 3D motion data of the fingertips as ground truth for our machine learning model. We performed the following four data preprocessing steps:

1. *Labeling and cleaning motion data*: We labeled all markers in the captured 3D motion data using semi-automatic labeling provided by OptiTrack's *Motive:Body* software. We did not use any reconstruction and smoothing approaches to avoid generating artificial marker positions.
2. *Transforming global to local coordinate system*: We transformed each hand marker from the global coordinate system into the phone's coordinate system and projected them onto the device surfaces. We validated the transformation by sampling five random frames per participant which we manually checked for correctness.
3. *Removing incomplete and erroneous data*: To ensure a complete and valid data set for model training, we keep only frames in which the rigid body and finger tips are fully available. Further, we applied a heuristic to detect erroneous rigid body tracking by assuming that the phone was not held in uncommon poses (*e.g.*, up-side-down, flipped). This heuristic removed 0.21 % of all frames.
4. *Synchronizing motion data and capacitive images*: We merged the capacitive images with the transformed motion data using timestamps as the merge criteria. As touchscreen latency is unavoidable and higher than the latency of a motion capture systems [32], the finger's ground truth position is ahead of the touch, especially during fast movements. To counteract the latency, we used a sliding window of 240 frames for each capacitive

image to find a motion capture frame in which the currently moving finger is within the generated blob (preferably in the center). We validated the merging process by checking whether the motion data corresponds to the blobs in the capacitive images. This was done by determining the blob's contour and checking whether the 2D position of the fingertip lies within the contour.

In total, our dataset consists of 9,435,903 valid samples stored in a 67.3 *GB* HDF5 file[1] to enable training on a large dataset.

Estimating the Fingertip Positions using CNNs

To develop the model, we used a participant-wise split of 70 %:20 %:10 % for the training, test, and validation set. *I.e.*, the model is trained on data from 14 participants, tested on 4 participants, and validated on the remaining 2 participants. We implemented CNNs using *Keras* 2.1.3 based on the *TensorFlow* backend. We performed a grid search as proposed by Hsu *et al.* [101] to determine the most suitable network architecture and 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 5.5. The input consists of capacitive images with 28×32 pixels normalized to a range between 0 and 1. The output consists of 15 values ((x,y,z) for five fingers) that represent the estimated 3D finger positions relative to the upper left corner of the display in *mm*. Thereby, a finger farther away from the device (*e.g.* lifted finger) has a higher distance in the z-axis as captured in the data collection study. We trained the CNN using an RMSprop optimizer [229] (similar to the AdaGrad [51] optimizer but with a less radical learning rate decay) with a batch size of 500. We experimented with different learning rates and found that an initial learning rate of .0001 leads to the best performance. We used batch normalization [103] and a 0.5 dropout after each pooling and dense layer to prevent overfitting. 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 [65].

[1] `support.hdfgroup.org/HDF5/whatishdf5.html`

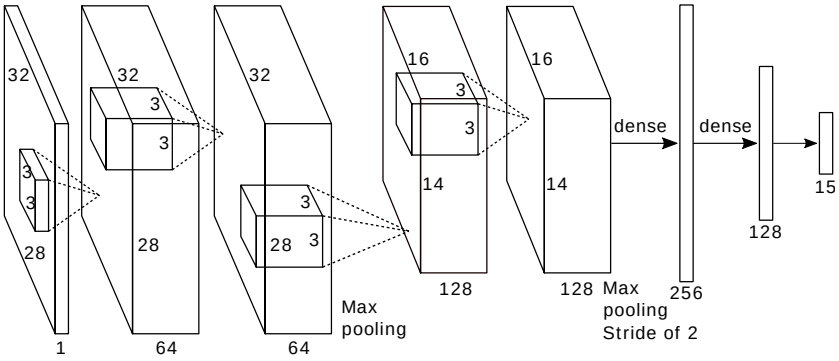


Figure 5.5: An illustration of the architecture of our CNN for finger position estimation. The network input is 896-dimensional, and the number of neurons in the network's remaining layers is given by 57,334–57,334–28,672–28,672–256–128–15.

After experimenting with traditional loss functions for regression such as root mean squared error (RMSE), we developed a custom loss function to train our CNN. As fingers above or below the device cannot be physically tracked by the touchscreen (*e.g.*, thumb resting above the display), errors induced by movements perpendicular to the touchscreen would affect the RMSE loss function as substantial as an error in horizontal (x) or vertical (y) direction. However, when omitting the z axis, the CNN would lose a feature to differentiate whether fingers are touching the device. Since a less accurate estimation of the z-axis can be easily compensated by checking the blob availability at the time using the model, we lowered the influence of the z-axis error by using an RMSE for the x and y axis, and a root mean squared logarithmic error (RMSLE) [104] for the z-axis as follows:

$$loss = \sqrt{\frac{\sum_i^n (p_{xy_i} - \hat{p}_{xy_i})^2}{n}} + \sqrt{\frac{\sum_i^n log_e((p_{z_i} - \hat{p}_{z_i}) + 1)^2}{n}} \tag{5.1}$$

with $n = 5$ representing the five finger tips, p for the ground truth point, and $\hat{p}$ for the estimated point.

Identifying Touches from Individual Fingers

To identify the finger touching the device (*i.e.*, the responsible finger for a specific contact area), we used a nearest neighbor approach to map the estimated positions to the blobs in the capacitive images. This approach has two benefits over using the estimated positions directly from the CNN. Firstly, the jitter caused by noise and the nature of machine learning can be prevented since the contact areas on the capacitive images are more stable. As recent touch controllers have shown, a blob can be converted to a precise touch position without any jitter. Secondly, the processor workload can be reduced since model inference is only necessary when fingers are initially touching (*i.e.*, down event) and releasing (*i.e.*, up event) the device. In other cases (*i.e.*, finger moving), fingers can be tracked using the blob position on the capacitive images.

On a technical basis, we performed a contour detection on a $5\times$ up-scaled capacitive image to determine the blobs. We then used the contour points stored in a k-d tree [60] to find the closest blob for an estimated finger position in $O(\log n)$ time. We used OpenCV for the contour detection and the Lanczos4 algorithm [233] to scale up the capacitive image. We used the k-d tree implementation from scipy for the validation and a reimplementation thereof in our Android demo applications.

5.1.4 Validation

While we used the training and test set to experiment with hyperparameters, we used the validation set to evaluate our best model. Our CNN achieved an MAE of $0.74\,cm$. Table 5.1 shows the errors for each finger and axis. The MAE for the axes are $0.85\,cm$, $0.85\,cm$, and $0.53\,cm$ for the x, y, and z axis respectively while the RMSEs are $1.41\,cm$, $1.39\,cm$, and $0.87\,cm$. The average Euclidean distance for all fingers when considering the error in 2D space (on screen) is $1.33\,cm$ whereas the average error in 3D space is $1.52\,cm$. Since the RMSE involves a larger penalty for larger errors (*e.g.*, outliers), an RMSE $>$ MAE indicates that errors can occur especially for uncommon finger placements. As expected, the z axis has the lowest error since the usable movement range perpendicular to the displays is the smallest of all axes.

These errors can be compensated with the finger identification approach as described above. The accuracy of our model with this approach can be evaluated as a multi-label classification problem; multi-label since multiple fingers could be matched to one blob due to the low resolution of the capacitive image. We used both the ground truth fingertip positions and the estimated fingertip positions and matched them with their closest blobs. Based on the matchings, we used the

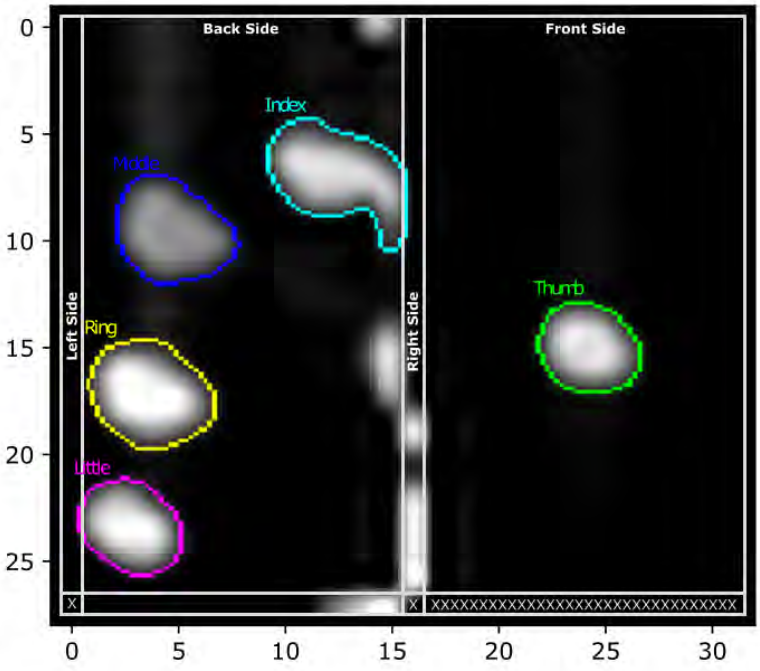


Figure 5.6: This image shows exemplary capacitive data retrieved from our prototype when held with Grip 1 as shown in Figure 5.4a. The colored contours represents the results of our finger identification model after mapping the estimations to the blob. The X's represents placeholder values that are required to build a 32×28 input matrix for the model.

	Thumb	Index	Middle	Ring	Little	Average
MAE (x)	1.04	1.03	0.98	0.63	0.56	0.85
MAE (y)	0.73	0.52	0.96	0.99	1.06	0.85
MAE (z)	0.50	0.28	0.46	0.50	0.89	0.53
RMSE (x)	1.80	1.75	1.63	1.06	0.79	1.41
RMSE (y)	0.98	0.87	1.43	1.74	1.93	1.39
RMSE (z)	0.73	0.86	0.79	0.72	1.26	0.87
Eucl. dist. (x, y)	1.40	1.25	1.45	1.25	1.30	1.33
Eucl. dist. (x, y, z)	1.55	1.32	1.57	1.42	1.76	1.52

Table 5.1: The mean absolute error (MAE), root mean squared error (RMSE) and Euclidean distances for each axis in *cm*.

Hamming score [232] which describes the accuracy of a multi-label classification on a scale between 0 (worst) to 1 (best). Our model achieved an average Hamming score of 0.9578.

5.1.5 Mobile Implementation and Sample Applications

We combine the full-touch smartphone, CNN, and nearest neighbor approach to implement *InfiniTouch*. We present our implementation and a set of sample applications.

Mobile Implementation

We used *TensorFlow Mobile*[1] for Android on the processing unit responsible for the front display to run the CNN that estimates the fingertip positions. Capacitive images from the back side and the edges are sent to the front device that merges the data into an input matrix. The input consists of a 32×28 8-bit image representing the front, back, and edges as shown in Figure 5.6. A model inference for one capacitive image takes $24.7\,ms$ on average ($min = 17\,ms$, $max = 29\,ms$, $SD = 2.8\,ms$) over 1000 runs on our prototype. As this is faster than the sampling rate for the touchscreens' capacitive images, the inference can be performed on each sample in the background. With processor manufacturers recently optimizing their processors for machine learning (*e.g.*, Snapdragon 845), model inference

[1] `www.tensorflow.org/mobile/`

can be sped up significantly.[1] The model can be further optimized for mobile devices with techniques such as quantization [78] and pruning [7] for a small loss of accuracy.

For the finger identification, the contour detection, including a scale up, takes $M = 2.85\,ms$ ($SD = 0.77\,ms$, $min = 1\,ms$, $max = 4\,ms$) while finding the closest blob takes $M = 0.48\,ms$ ($SD = 0.12\,ms$, $min = 0.19\,ms$, $max = 0.96\,ms$) over 1000 runs on our prototype. Tracking the blobs take $M = 0.08\,ms$ ($SD = 0.04\,ms$, $min = 0.001\,ms$, $max = 0.82\,ms$). During these benchmarks, the device was held one-handedly with all five fingers touching the device (*c.f.* Figure 5.4a).

Using Finger Identification in the Application Layer

We extended our Android library described above to provide the finger position estimations from the model and the respective position of the blob (*i.e.*, position of the upper-left contour point, and size) for each finger in a callback function. This enables developers to access the finger positions similar to Android's `OnTouchListener` interface. Besides the position (in an on-device coordinate system with the upper left corner of the front display being $(0,0,0)$), we also provide the event (*i.e.*, down, up, and move). With this, the blob's position and estimation can directly be fed into Android's `MotionEvent` which enables to use Android's own `GestureDetector`, or third-party gesture recognizers such as $P [236], $1 [258], $N [6], and the gesture recognition toolkit [64].

Sample Use Cases

Based on the mobile implementation of our model, we implemented two use cases for finger-aware interaction on the full-touch smartphone. We describe our implementation in the following and showcase them in the accompanying video.

Finger-Specific Touch Gestures Implementations of BoD interaction in previous work [133, 197, 215] treated inputs of all fingers equally. Thus, performing a gesture with the index finger would result in the same function as a gesture performed with the middle finger. With *InfiniTouch*, the same gesture can activate

[1] `www.qualcomm.com/snapdragon/artificial-intelligence`

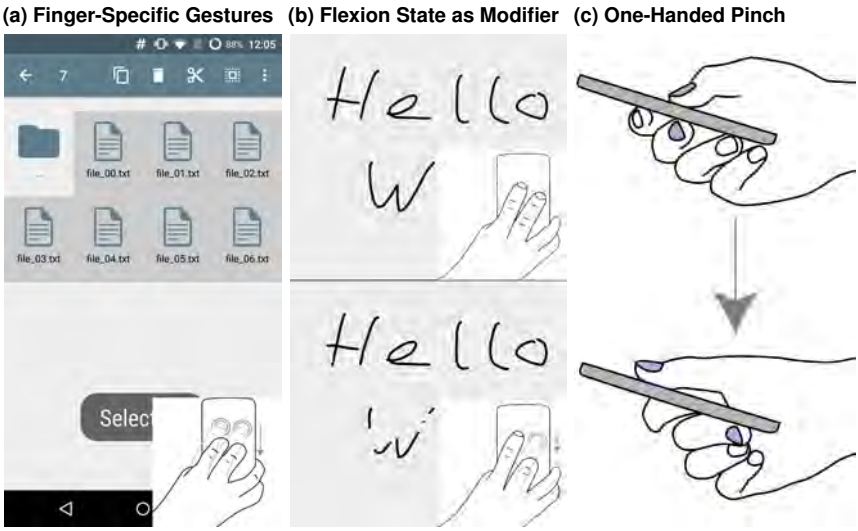


Figure 5.7: Screenshots of our sample applications implemented on the *InfiniTouch*. Figure (a) showcases how a down-swipe with both index and middle finger selects all files in a file manager, Figure (b) demonstrates how the position of the middle finger can be used to switch between a pen and an eraser, and Figure (c) demonstrates an exemplary one-handed pinch gesture.

different functions depending on which finger performed the input. This extends the input space similar to a computer mouse where the index finger is used for main actions, while the middle finger is used for the right mouse button to activate secondary actions.

In our sample use case, we mapped a swipe down performed by the index finger to copying selected items into the clipboard (inspired by the *come to me* gesture) while a swipe down by the middle finger pastes from the clipboard. A swipe down performed by both index and middle finger simultaneously selects all items as shown in Figure 5.7a. While we demonstrated this concept within a file manager, it can also be used in text editing applications, galleries, and further applications that support the clipboard.

BoD Finger Flexion State as Action Modifier While hardware keyboards provide modifier keys to modify the action of another key, touchscreens implement this concept only via dwell times or applied force which requires additional execution time. We propose to use the position of the fingertips (*i.e.*, their flexion state) on the back to modify the actions performed on the front screen. For example, bending a specific finger can be done comfortably [130] and could be used similarly to a pressed `Ctrl` key on a hardware keyboard.

We implemented a simple paint application that maps drawing and erasing to the flexion state of the middle finger. When the middle finger is flexed, the pen is activated which enables the user to draw. When bending the middle finger (*c.f.* Figure 5.7b), the eraser will be activated to remove parts of the drawing. While we demonstrated this concept within a paint application, it can be applied to a wide range of applications that benefit from action modifiers and with all four fingers. Amongst others, this includes opening context menus similar to the right mouse button, text selection and highlighting, mode switching (*e.g.*, slower and faster scrolling), 3D navigation, and providing shortcuts.

Further Use Cases

We present further use cases for *InfiniTouch*.

One-Handed Pinch and Rotation Gestures Users need to hold smartphones two-handed or place it on a surface to perform a pinch or a rotation gesture. We propose to use a pre-defined finger on the back of the device as the second finger to perform a pinch/rotation gesture with the thumb on the front screen. This enables users to zoom or rotate objects in a one-handed grip as shown in Figure 5.7c.

Enabling Transient Actions Avery *et al.* [9] proposed transient gestures to enable users to temporarily change the view of an application which can be rapidly undone. As a zoom in always requires a zoom out to return to the initial state, they used an additional finger on a tablet to save the initial state. When this additional

finger is released, it restores the initial state so that users can alter the view in between. Using our concept of finger positions as a modifier, we could replace the additional finger with a finger on the rear that is able to bend and flex.

Improving Reachability Bergstrom-Lehtovirta and Oulasvirta [20] showed that the thumb's range could be modeled with the position of the index finger's tip as input. With *InfiniTouch*, we can determine the position of the index finger and can thus adapt the user interface to optimize reachability during one-handed interaction. Moreover, we can assign the functionality to move the screen content to a specific finger. This enables the finger to move the screen content to a more reachable position to improve one-handed interaction as proposed in previous work [133].

5.1.6 Discussion and Limitations

We developed *InfiniTouch*, a system that enables finger-aware interaction on full-touch smartphones. We developed a full-touch smartphone prototype and trained a CNN to identify fingers touching the device surface. We implemented and showcased a number of applications.

Model Accuracy

We trained a CNN that estimates the fingertip positions with an MAE of $0.74\,cm$ over all three axes. As a comparison, the average diameter of a human index finger is $1.6\,cm$ - $2.0\,cm$ [44] while Holz *et al.* [98] found that traditional touch interaction has a systematic offset of $0.4\,cm$. Even without using the positions of the blobs, this already enables users to perform precise interactions, such as gestures or finger placements as modifiers. Moreover, using the estimated positions enables differentiation between fingers even if their contact areas are united due to a low-resolution image. A limitation of our model is that estimations of more distant fingers (*e.g.*, a finger moving without touching the device) become less accurate since they cannot be sensed physically.

Based on the estimations, we used a nearest neighbor approach to identify the responsible finger for each blob with an accuracy of $95.78\,\%$. As we perform

this process only when the number of blobs in the capacitive image changes, we reduce the processor workload and potential jitter due to noise and the nature of machine learning. Moreover, since we track the blobs while keeping their label (if the number of blobs did not change), labeled blobs stay correctly labeled even if the model yields an inaccurate estimation in a rare hand posture. This means that we only identify fingers when they initially touch or release the device (*e.g.*, new hand grip) with an accuracy of 95.78 % while classification errors cannot occur afterwards. We provide both blobs as well as estimated positions in our Android library, and successfully implemented our sample use cases with both approaches. Further, the estimated location could be used as a fallback in case the blob detection is not capable of telling two blobs apart due to the low resolution.

Improving Accuracy and its Sufficiency for Use Cases

Our model estimates the 3D finger positions with an MAE of 0.74 *cm* and classifies blobs with an accuracy of 95.78 %. While this is sufficient for a reliable recognition of gestures and the use of absolute positions, future work could further improve the accuracy as follows. Although our 32×28 capacitive images already comprise over 14 times the amount of sensors of previous approaches based on flexible PCBs (*e.g.*, 64 [33] or 24 [168, 169] measurements), further increasing the resolution could help to improve classification accuracy. High-resolution capacitive images certainly benefit the blob matching due to clearer contact area boundaries and also benefit the MAE since more features of the finger become detectable. Possible technologies include FTIR that enables high-resolution multi-touch sensing [77] and infrared sensors integrated into the LCD layer similar to the SUR40 interactive display by Samsung[1]. While these technologies are yet to be mass-produced for mobile devices, our prototype is based on hardware that is already publicly available which enables the community to reproduce *InfiniTouch*.

The accuracy of our model is well beyond the 80 % that previous work considered sufficient [113]. However, sufficiency also depends on the action's consequence (easily recoverable action vs. permanent action) and how inferences are translated to actions. For *InfiniTouch*, the consequence depends on the action that future developers implement while a wide range of translation approaches can

[1] `www.samsung.com/ae/business/smart-signage/interactive-display-sur40/`

further minimize accidental activations. For example, accidental activations of BoD gestures can be minimized using the confidence score of gesture recognizers, using thresholds for a minimum gesture length, or using heuristics to differentiate gestures from grip changes (*e.g.*, only one finger can move at a time). While our implementation works reliably for the implemented use cases, we also suggest that future BoD gestures should be designed with false positives and negatives in mind. Moreover, the flexion state example could also involve users in avoiding unintended actions by using visual elements to indicate the recognized flexion state (*i.e.* action).

Practicality of Use Cases

We designed the sample use cases solely to demonstrate the possibilities offered by *InfiniTouch*. Thus, we chose fingers and movements that are easy to explain and understand, but we also designed them to be ergonomically viable based on findings described in Chapter 3. Designing our explicit BoD gestures, we considered these findings that showed that index and middle fingers can move comfortably within a large area (around top to center for similar devices) without grip changes and independent from the grip. This indicates that our presented BoD gestures can be performed comfortably without a grip change. Moreover, subtly bending the middle finger for the second use case also takes place within the comfortable area. As we focus on the technical contribution, future work could investigate the comfort of such BoD gestures and how to communicate them to end users [159].

Reproducibility with Publicly Available Hardware

We presented an approach to prototype a full-touch smartphone with publicly available hardware. While we used an LG Nexus 5 as the basis, our approach can be applied with any smartphone. This enables the community to reproduce our prototype and use our model to explore finger-aware interaction with *InfiniTouch*. As a tradeoff, data from both touchscreens and the edge sensors need to be synchronized over network which adds a latency of $7.2\,ms$ while an additional hardware container is required. Despite an additional container, our

prototype can still be used in mobile situations (*e.g.*, in walking studies) since the hardware container is designed to be fixated on the forearm. Moreover, smartphone manufacturers could produce proprietary components for future commodity smartphones so that a hardware container is not needed in a mass-market version. These components could comprise flexible PCBs with a sufficient amount of sensors. These are already used in consumer products (*e.g.*, the Microsoft Touch Mouse) and provide capacitive images of touches. Using such components would also avoid the synchronization of data over the network while manufacturers can directly use our model for finger-aware touch interaction on the whole device surface.

Specialization on Common One-Handed Grips

The model presented in this chapter focuses on one-handed grips. Previous work has shown that fingers can comfortably reach around 70 % of the back (for similar device sizes [129, 130]) during one-handed smartphone interaction without grip changes. This enables the fingers on the back to be used for a wide range of explicit (*e.g.* BoD/side gestures) and implicit interactions (*e.g.* flexion/grip sensing) to increase the expressiveness of one-handed touch input. Since our comprehensive dataset covers a wide range of typical one-handed grips as performed in the study, our model also works when some fingers of the holding hand are not touching and stays robust even when other hand parts (*e.g.* palm) are touching or releasing. Two-handed grips and further touches beyond usual one-handed grips (*e.g.*, using other body parts) are currently not expected by our model and would lead to unexpected estimations. However, with minor adaptations to the implementation, our model can even be used to identify finger positions of the holding hand while the other hand performs input on the front. While we focused on right-handed grips to show the feasibility of our approach, our procedure and publicly available source code enables researchers to easily extend our work to other devices and use cases (*e.g.*, left-handed or bimanual grips for tablets).

5.2 Exploring Interaction Methods and Use Cases

After we showed the technical feasibility of fully touch sensitive smartphones, we collaborated with experienced interaction designers to elicit novel ways to solve the limitations of current touch input. To avoid the influence of the technical details of our prototype on the participants, we used a mockup of our prototype to demonstrate *InfiniTouch*.

5.2.1 Interviews

We conducted semi-structured interviews to explore interaction methods and use cases for a *full-touch smartphone*. Particularly, we focus on following questions:

1. How can we use a *full-touch smartphone* to address common touch input limitations such as the fat-finger problem or reachability issues?
2. What are novel use cases for a *full-touch smartphone*?

Participants & Prototype

Since we are interested in answers based on interaction design experiences, we recruited 8 participants who have worked with smartphones from an interaction design perspective. Participants were between 23 and 50 years old (M = 31.6, SD = 9.2) with two of them being female. The participants comprised two professors for mobile communication and mobile application development from a local university, one project lead for strategy and interaction design at a design company, and graduate and PhD students in the field of interaction or communication design.

To give participants a better vision of a *full-touch smartphone*, we presented a video of the concept and handed them a mockup during the interview (see Figure 5.8a). The mockup consists of the same 3d-printed frame and two $5''$ touchscreens as used for the *InfiniTouch* prototype presented in Section 5.1. We removed all cables, copper plates, as well as the hardware container to avoid influencing the participants with technical details of our prototype. Participants used the mockup to demonstrate actions of which we took photos.

(a) Mock Up Device **(b) Coding and Clustering Example**

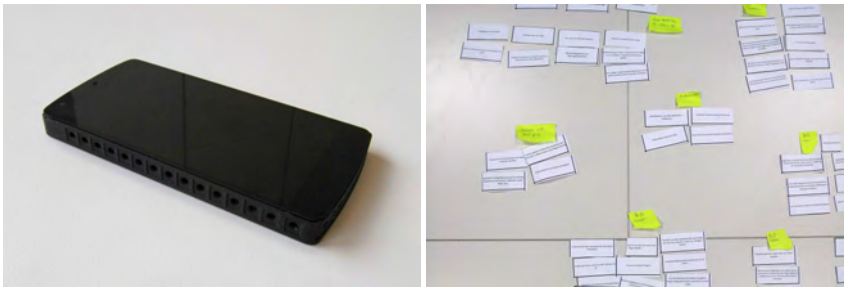


Figure 5.8: Figure (a) a mockup of our full-touch smartphone prototype to avoid affecting the participant with unusual sensors. Figure (b) shows how we coded and clustered the participants' answers into respective clusters.

Procedure

The semi-structured interviews took place in a quiet room within the company or institution of the participant and were audio-recorded. Interviews lasted about 40 minutes and comprised four parts: Firstly, we asked ice-breaker questions about participants' smartphones and situations in which they are using it. Secondly, we asked participants about limitations and difficulties that they encounter while dealing with touch input on usual smartphones. Prior to the third part, we introduced the prototype as described above. We then explored interaction techniques on a *full-touch smartphone* which addresses the limitations mentioned by the participant in the previous part. We ensured that all participants proposed solutions for at least the fat-finger and occlusion problem, as well as the reachability issue. In the last part, we explored novel use cases for a full-touch smartphone. Participants were asked to suggest scenarios in which the additional surfaces of such a smartphone can be used to implement functionality which is not feasible on recent smartphones.

5.2.2 Results

All audio recordings were transcribed. Based on the transcript, we extracted all comments and printed them on paper. Two researchers then employed a simplified version of qualitative coding with affinity diagramming [79] to analyze the results (see Figure 5.8b for an example).

Limitations of Smartphone Input

When asked about limitations and difficulties in interacting with recent smartphones, the majority of participants were unanimous about the fat-finger problem [16]. They described this through "*too big fingers*" (P1, P3, P6) and "*undersized user interface elements*" (P1, P3, P5, P6, P8). Consequence of this are occlusion issues ("*When drawing, I cannot see the result.*" - P6) which also leads to users "*[not knowing] what a touch is triggering*" (P8). The latter phenomenon is caused by a misconception of the registered touch point between user and touchscreen [98] and the lack of haptical feedback which renders blind input nearly impossible (P3, P6). Thus, participants argue that users are required to frequently look at the touchscreen to adjust their input which leads to a high cognitive demand when doing something else simultaneously ("*[..] is difficult as I need to see where I am walking and where I want to touch simultaneously.*" - P3). This becomes even more detrimental when external disturbances, such as jerks while running (P2) or bumps in public transport (P3), affects the user.

Despite software-based solutions like iPhone's *Reachability*[1] or Samsung's one-handed mode[2], participants (P2, P5, P7) still regard the limited thumb range during one-handed use as a input limitation (see Figure 5.9a). As these methods require a manual activation, participants "*do not see any additional value compared to just [adapting] the hand grip.*" (P2). However, adapting the hand grip and therefore tilting the device while stretching the thumb leads to unintentional input ("*[..] when trying to reach targets on the upper left corner, my palm uninten-*

[1] "How to use Reachability on your iPhone". 2016. URL: `https://www.cnet.com/how-to/how-to-use-reachability-on-iphone-6-6-plus/`

[2] "How to use the Samsung Galaxy Note 4 with one hand". 2016. URL: `https://www.cnet.com/how-to/how-to-use-the-samsung-galaxy-note-4-with-one-hand/`

(a) Reachability Problem Demonstration (b) Changing Camera Settings

Figure 5.9: Figure (a) depicts a participant demonstrating reachability issues on smartphones. Figure (b) depicts a participant showing how to change camera settings on the edges.

tionally touches the touchscreen which is not filtered out by the operating system." - P7). Especially when holding objects in the other hand (i.e. being encumbered [177]), this can become a critical issue for the users according to P1, P3 and P5.

Improving Smartphone Interaction

With experienced interaction designers, we explored different interaction methods to overcome the described limitations of touch input on smartphones. We describe the interaction methods clustered into categories and explain how they help to overcome the limitations.

Back-of-Device Input and Feedback. As occlusion issues and lack of feedback on the registered touch position can be detrimental, participants suggested two methods based on BoD input to tackle these limitations. P2-P8 envisioned to use the back side to control a cursor on the front screen to avoid occlusion through the finger. As the lower area of the back side is already covered by the hand holding the device, P2 suggested to only use the upper half either by mapping the front screen to this area, or to control the cursor in a relative manner similar to laptop's touch pads. Moreover, participants all agreed that a confirmation is required

(a) Scrolling on the Right Edge (b) Metaphorical Grip Pattern

Figure 5.10: Figure (a) depicts a participant demonstrating scrolling on the device's right edge by swiping down the thumb. Figure (b) depicts a participant demonstrating a metaphorical grip pattern.

to avoid unintentional input, *e.g.* by squeezing the device or applying pressure onto the desired touch position (P2). Similar to prior work [16, 250], P2 and P3 envisioned a pseudo-transparent touchscreen by showing the registered touch point and finger shape of the back side as an overlay on the front screen. Thus, users would receive feedback on their finger and touch position while occlusion can be avoided.

Gestures & UI on Adjacent Sides. Participants (P1, P3-P5, P8) argued that not only fingers do occlude desired content but also input controls such as buttons, menus or sliders. This is especially the case for games (P1, P3, P8), camera applications (P4, P8), image editors (P1) or maps (P8) as their main focus lies on the graphical content. Thus, participants suggested to move input controls to the device's edge (P1, P3-P7) or back (P2, P3, P6).

When asked for examples, P5 and P8 envisioned a camera application with input controls on the edges (see Figure 5.9b). Similar to usual cameras, adjustments (*e.g.* focus, brightness, etc.) can be made on the device edges without occluding the front screen. Other examples include movements such as pinching or dragging a slider: P8 suggested to use the back side to perform scrolling or zooming operations while P3 envisioned the edges for scrolling or for manipulation of values

similar to sliders (see Figure 5.10a). Interestingly, when demonstrating the slider on the edge, participants reportedly stated that "*it feels more comfortable and natural than on the front screen, especially when using the device one-handed*" (P1, P3, P8). Similarly, games also profit from a move of input controls to the edge or back of the device (P1, P3, P8).

As touch buttons and sliders do not provide any haptical feedback which makes it difficult to locate them, participants suggested to visualize buttons and sliders with ambient lights on the edges while augmenting them with vibration feedback similar to the home button of an iPhone 7 (P5).

Simultaneous Use of Multiple Sides. Conforming to prior work [276], participants (P1, P3, P6, P7) suggested to use the edge and back side as a proxy space for areas that are not reachable by the thumb due to its limited length. For example, input controls on the top half can be accessed by the index finger from the back side while input controls on the lower half can be accessed by the thumb on the front. Moreover, due to thumb and index finger moving independently, three participants envisioned simple gestures on the back side to *e.g.* trigger system actions (*e.g.* "*switching or closing apps*" - P6) or to move the screen content to a more reachable position (P2, P5) (cf. [133]).

Similarly, P7 suggested a function to zoom into parts of the screen depending on the position chosen on the device's edges. P1 suggested double-sided buttons that trigger different actions depending on the touching side. For example, "*clicking the button from the front side opens a window to write a message while clicking from the back side opens a window to write a direct message to a pre-defined contact*" (P1).

Squeeze Interaction. Participants envisioned actions to be triggered when the phone is squeezed. This includes accepting calls or hanging up (P5), taking photos (P1), zooming in and out (P5), or spawning a quick-launch bar (P1). This is beneficial as prior work found that squeeze interaction is not affected by encumbrance or walking jerks [59].

Hand Grip Pattern Recognition. Participants envisioned to train specific hand grips to accept or decline calls (P2), change the song or volume (P4) or to launch applications (P2). Metaphorical grip patterns (*e.g.* a finger pinching the corner) could be interpreted as modifiers by *e.g.* keeping the screen as it is when rotating the device (P7, see Figure 5.10b).

Moreover, users' natural hand grip can be recognized to adapt the user interface. For example, the user interface adapts to the user's handedness (P3, P6), or arrange controls based on the finger's position (P3, P4). Grip patterns can also be used to suggest subsequent actions, or facilitate actions by *e.g.* enlarging the keyboard when needed (P2).

Use Cases and Opinions

With more information available about the hand grip and finger placement, participants envisioned the system to use this information to recognize different features, such as handedness, grip stability, range of the thumb for a dynamic placement of buttons, or the users frustration (P6). Moreover, patterns can be used to authenticate the user similar to what *Bodyprint* [99] does for the front screen (P1, P2, P7). In general, these ideas require research to be done which is why P3 also envisioned a *full-touch smartphone* as a research tool. We imagine to use such a device to seek understanding on how the hand interacts with the device without the need of cameras or motion trackers. This enables studies also to be conducted in mobile situations.

In general, participants liked the idea of a *full-touch smartphone* (*e.g.* "*super exciting*" - P2; "*attracts attention*" - P5; "*exciting possibilities*" - P8) and thus came up with 17.8 (SD = 3.0) ideas on average per participant. Despite the excitement, some participants were concerned about unintentional input (P1, P4, P5, P7), lack of compatibility with recent user interfaces (P3, P8), and increased battery consumption (P6).

5.2.3 Discussion

In the context of semi-structured interviews, eight participants suggested different interaction methods for a *full-touch smartphone*. Based on their experiences in

interaction design, participants argue that these interaction methods are potential solutions to common touch input limitations. Suggestions to deal with the *fat-finger* and *occlusion problem* include performing input on the back of the device augmented with positional feedback on the front side, and outsourcing UI components to the edge of the device. As solutions to the *reachability issue*, participants suggested to use adjacent sides as a proxy space to perform input or scroll operations since these are easier to reach. They further suggested interaction by squeezing the device, or to map certain hand grip patterns to functionality. As both interaction methods can be blindly performed, they are suitable for interaction when less focus is available, *e.g.*, while being encumbered or while walking [21, 177].

Some suggestions, such as performing BoD input [16] or arranging the UI according to the grip location [35], were already researched in prior work in HCI and shown to be effective. Besides this, participants also explored novel ideas. Amongst others, these include outsourcing the UI and occluding input (*e.g.* scrolling gestures) to the edge of the device, or the use of multiple sides simultaneously (i.e. proxy space) to increase reachability. Evaluating these ideas requires a *full-touch smartphone* with dimensions and haptics similar to a mass-market smartphone to avoid influencing the usual hand grip and behavior of the user.

5.3 General Discussion

In this chapter, we presented a fully touch sensitive smartphone prototype which uses deep learning to identify input from all fingers in a single-handed grip. For this device, we interviewed experienced interaction designers to elicit novel methods to solve the challenges of recent touch input.

5.3.1 Summary

Addressing RQ4, we first developed a fully touch sensitive smartphone prototype which provides capacitive images representing touches on the whole device surface. We designed the smartphone prototype to achieve the form factor of a standard smartphone to avoid affecting how users usually hold a mobile device.

Based on the capacitive images, we then applied deep learning to train a model which estimates the 3D position of the finger tips holding the device. The estimation achieved an MAE of 0.74 *cm* and can be used to identify the input of individual fingers with a nearest neighbor mapping to the blobs on the capacitive image. Previous work [63] and our investigation in Section 4.3 suggested that capacitive images from the front touchscreen do not contain sufficient signal to identify each finger during regular interaction. Thus, previous work proposed using additional sensors (*e.g.*, cameras [38, 284]) and wearable sensors [74, 75, 152] to enable finger identification. While these approaches are inconvenient and immobile, we showed that all fingers of the holding hand can be accurately identified with our approach on a fully touch sensitive smartphone. Since our prototype provides the capacitive images of the whole hand (instead of just a single finger tip as on a recent smartphone with a single touchscreen), the model has enough information to reconstruct the 3D positions of the holding hand. Since our model is solely based on capacitive sensing, finger identification can be integrated into future commodity smartphones (with touch sensing on the whole device surface) without any external or wearable sensors.

Addressing RQ5, we interviewed experienced interaction designers from industry and academia to elicit use cases for fully touch sensitive smartphones as well as solutions to the limitations of recent touch input. Amongst others, the experts perceived the fat-finger problem, reachability issues, as well as a lack of mechanism to access frequently used functions (*e.g.*, camera settings) as notable limitations of recent touch input. Accordingly, our participants proposed a wide range of solutions to these challenges. In the next chapter, we will present the whole process to design, implement, and solve the challenge of the lack of shortcuts.

5.3.2 Lessons Learned

Based on our developed prototype and the expert interviews, we derive the following insights:

Identifying fingers is feasible on a fully touch sensitive smartphone. In contrast to recent commodity smartphones, our fully touch sensitive smartphone provides capacitive images representing touches on the whole device

surface. Information about the grip enables our deep learning model to accurately estimate the 3D position of the finger tip and, with a nearest neighbor algorithm, to also accurately identify the input from different fingers. Since our approach is based on capacitive sensing only, no further sensors are required which reduces mobility, convenience, or would notably increase the size of the device.

Keeping standard device form factor by outsourcing processing units. In contrast to stacking or attaching an external touch pad to the back of a device, we presented a novel approach to prototype BoD and edge interaction without notably altering the device size. This avoids affecting the usual hand grip of users.

Full-touch smartphones can solve common touch input limitations. The expert interviews revealed that a fully touch sensitive smartphone with hand-and-finger-awareness provides a wide range of opportunities to solve typical touch input limitations such as the fat-finger problem, reachability issues, and the lack of shortcuts.

6

Improving Shortcuts for Text Editing

In the previous chapters, we investigated the hand ergonomics and the technical feasibility of using multiple fingers and hand parts for interaction with mobile devices. With this, we understood and specified the context of use and laid the groundwork for developing novel interaction techniques which can solve the limitations of touch input.

In this chapter, we focus on addressing the limitations of mobile text editing as a specific challenge. We present four studies which cover all steps of the UCDDL as presented in Section 1.2.3 to solve the limited shortcut capability in order to increase the usability of mobile text editing.

This chapter is planned to be published as follows:

H. V. Le, S. Mayer, J. Vogelsang, H. Weingärtner, M. Weiß, and N. Henze. “On-Device Touch Gestures for Text Editing on Mobile Devices.” In: *Under Review at the ACM Transactions on Computer-Human Interaction*. TOCHI.

6.1 Text Editing on Mobile Devices

Due to their mobility, smartphones are utilized for tasks that were previously exclusive to desktop computers. Mobile word processors [166], spreadsheets [164], and even presentation programs [165] are installed over 100 million times and became viable alternatives to their desktop counterparts. Moreover, recent advances in the field of mobile text entry enabled users to type with a speed that is almost comparable to hardware keyboards. This indicates that users strive to use smartphones as a mobile alternative to desktop computers for tasks such as text editing. Indeed, tasks such as writing emails or browsing the internet are already common tasks for most users on their smartphones [219]. However, the trade-off for the mobility is a small display size. This poses a number of challenges especially for text editing on smartphones, which could be one reason why it is not widely adopted yet.

The fat-finger problem [16, 217] is a well-known challenge and makes precise caret placement and text selection frustrating. Further, a large number of shortcuts known from hardware keyboards (*e.g.*, `Ctrl+C`) are not available or hard to access. Instead, inferior concepts such as long-presses are used that require a dwell time. The lack of shortcuts contradicts Shneiderman's golden rules for interface design [210] and slows down text-heavy activities especially for experienced users. Previous work [61] and commercial products used on-screen gestures as shortcuts to frequently used functions. However, a comprehensive set of gestures would interfere with the UI (*e.g.*, conflicts with gesture keyboards) while the gesture recognition accuracy would decrease due to ambiguity errors.

Additional on-screen buttons for frequently used functions could be a solution. However, a large number of buttons would clutter the interface. Thus, more and more smartphones incorporate input controls beyond the touchscreen. This includes BoD touch panels (*e.g.*, Meizu Pro 7), pressure sensitive frames (*e.g.*, Google Pixel 2), and physical buttons such as the Bixby button on Samsung devices. In addition, our *full-touch smartphone* prototype enables users to perform input on the edge and rear which increases the input capabilities. This enables

recently unused fingers on the back to activate functions which support text editing activities on the front side, and avoids interfering with the UI in contrast to gestures performed on the front screen [61].

6.1.1 Study Overview

Before using the whole device surface to provide text editing shortcuts, it is vital to understand which shortcuts are frequently needed and how they could be accessed on a full-touch smartphone. Thus, this chapter presents the results of four studies to bring frequently used shortcuts for text-heavy activities from hardware keyboards to fully touch sensitive smartphones. This chapter follows the UCDDL process as presented in Section 1.2.3.

Stationary computers with hardware keyboard and mouse are recently the primary device for text editing and thus suitable to study the used shortcuts. Thus, Study I analyses shortcuts performed by 15 expert users over five workdays in an in-the-wild study. Subsequent interviews revealed that major challenges for text-heavy activities on smartphones are limited input precision (*e.g.*, placing the caret) and the lack of shortcuts. Despite these challenges, users still find text editing on smartphones essential. As these limitations can be solved by offering more shortcuts (*e.g.*, for navigating the caret), we conducted Study II to elicit gestures that provide shortcuts to frequently used functions. We followed the formal methods proposed by Wobbrock *et al.* [255] and focus on *full-touch smartphones* to avoid conflicts with gesture keyboards. To evaluate the elicited gesture set and its usability in realistic text editing scenarios, we conducted two further studies to implement and evaluate the prototype. Thus, in Study III, we collect a ground truth data set of participants performing the gesture on our fully touch sensitive smartphone to develop a deep learning model. In Study IV, we then evaluate the gesture set with realistic text editing scenarios to gather qualitative feedback on the gestures and their usability.

6.2 Study I: Shortcuts on Hardware Keyboards

We conducted an in-the-wild study to analyze shortcuts performed on hardware keyboards by expert users. Specifically, we investigate single and combination of keys that activate functions, and that are not available on recent smartphone keyboards, *e.g.*, the standard keyboard on Android and iOS.

6.2.1 Apparatus

We used a low-level system-wide hook for keyboards to log all performed shortcuts. Based on the *Java System Hook library*[1], we developed an application that runs in the background to log shortcuts into a text file. Shortcuts include all keys and key combinations that do not generate a new visible character on the screen, amongst others key combination that involves the modifier keys (`Ctrl`, `Win`, `Alt` as well as `Shift` for non-alphanumeric and non-punctuation keys), cursor keys, command keys (*e.g.*, `Del`, and `Insert`) and function keys (*i.e.*, `F1-F12` which are commonly unavailable on mobile on-screen keyboards). For each shortcut, we logged the foreground application's file name (*e.g.*, *winword.exe*) and the timestamp. Our application automatically starts when the system boots. All participants were running our application on their main work computer that runs Microsoft Windows.

6.2.2 Procedure and Participants

After obtaining the participants' informed consent, we set up the application on their main work computer. We briefed them on the collected data and their right to delete lines out of the log before the end of the study. Our application logged keyboard shortcuts over five full work days while weekend days and the starting day were excluded. After five work days, we invited the participants back to our lab for a copy of their logs and an interview. Specifically, we asked them about their experiences in text-heavy activities on hardware and touchscreen keyboards, and perceived advantages and disadvantages of both input modalities. In semi-structured interviews, participants provided comments orally and summarized them in written form afterward. This took around 20 minutes per participant.

[1] Java (low-level) System Hook library: `http://github.com/kristian/system-hook/`

We recruited 15 participants (7 female) between the ages of 20 and 34 (M = 25.5, SD = 4.2). Seven participants were research associates while eight participants were computer science students at a technical university located in central Europe. All participants were reportedly using their computer actively during the study. All participants stated that they write or edit text multiple times per day on hardware keyboards for tasks such as taking notes, writing reports and papers, and programming. Moreover, 12 participants stated that they write and edit text multiple times per day on smartphones to communicate with friends and family while two participants performed these tasks at least once per day.

6.2.3 Log Analysis: Shortcuts on Hardware Keyboards

We filtered all repetitions of the same shortcut within 1000ms to avoid counting consecutively performed shortcuts multiple times (*e.g.*, pressing `Alt+Tab` multiple times to select the desired window). Moreover, we removed all shortcut keys that were used only by a single participant to avoid user-specific shortcuts. After filtering, the data set consisted of 67,943 shortcuts with 96 unique ones. Each participant performed a total of 787.5 (SD = 900.9) shortcuts per day on average of which 28.1 (SD = 13.4) were unique. During the five work days, participants performed shortcuts within 8.8 hours (SD = 3.2) per day on average. In addition, two researchers independently classified all logged applications into categories and discussed them afterward. Based on the discussion, we analyzed shortcuts within the following three categories: *word processing* which includes all applications focusing on editing and formatting text; *programming* which focus on coding tasks; and *system-wide* which includes all logged applications (*e.g.*, system applications, browsers, media player). Figure 6.1 visualizes the average use of shortcuts over a working day.

When categorizing performed shortcuts by the number of keys, 43.0% were single-key shortcuts, 52.2% were double-key shortcuts and 4.1% were triple-key shortcuts. Table 6.1 shows all shortcuts that were performed at least 1% in total averaged over all participants. To retrieve these results, we first calculated the percentage for each shortcut per participant and then averaged them over all participants. Overall, the arrow keys were the most used amongst all shortcut lengths and application categories followed by clipboard management shortcuts

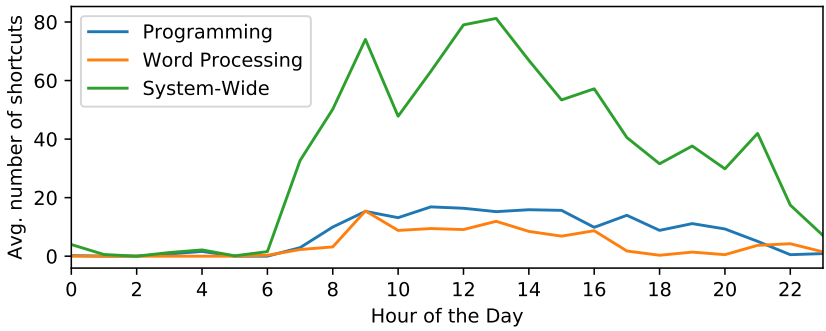


Figure 6.1: Average number of shortcuts performed over the days per participant.

(*e.g.*, `Ctrl+V`) and the window switching shortcut (`Alt+TAB`). Shortcuts for caret movement and text selection were more frequently used in word processing applications than in programming environments and system-wide. In contrast, the `Left` and `Right` keys, as well as shortcuts for undo (`Ctrl+Z`), pasting (`Ctrl+C`) and saving (`Ctrl+S`) were performed more often in programming environments than system-wide on average. In general, 19 of the 24 most frequently used shortcuts benefit text-heavy activities (*e.g.*, navigation, selecting, and clipboard). Three are widely available in text editors (undo, save, and search) while two are system-wide shortcuts (switching windows and open file manager).

6.2.4 Interviews: Hardware and Touchscreen Keyboards

We extracted arguments from the participants' answers and printed them on paper cards. Two researchers then coded and clustered the comments. We focused on WORD PROCESSING and PROGRAMMING as two text-heavy activities and identified two main clusters which address the INPUT and the OUTPUT aspect. For each cluster, we found advantages and disadvantages for touchscreen and hardware keyboards. For the WORD PROCESSING task, participants made 46 comments addressing INPUT and 2 about OUTPUT challenges. Similarly, participants made 77 comments about INPUT and 12 about OUTPUT challenges for the PROGRAMMING task.

Word Processing Two participants mentioned that the display size is one major concern that affects the OUTPUT. Specifically, "*the display is too small*" (P7) and thus "*lacks display size to fix spelling mistakes while writing messages*" (P9). Regarding the INPUT, participants commented on the lack of haptics (25 comments) and shortcuts (21 comments). The lack of haptic feedback on touchscreen keyboards caused participants to be reportedly less accurate and thus slower ("*the large keys [on hardware keyboards] allow me to type faster*" – P3; "*[hardware keyboards] provide haptic feedback so that I do not have to look at the keyboard while typing*" – P7). Specifically, the lack of haptic feedback makes it difficult to perform precise operations, such as placing the caret within a text (*e.g.* "*It is difficult to put the cursor to the desired text position.*" – P6; "*Choosing the position is inaccurate.*" – P4).

The latter challenge can be solved with shortcuts (*e.g.*, `Home`, `End`, and arrow keys) to counteract the lack of precision. This conforms with statements from P2 who found that "*hardware keyboards offer many more options to change the caret positions*" and P12 who found that "*hardware keyboards are more precise since it is easy to switch between words and select whole lines*". Moreover, "*[hardware keyboards] offer a lot of functions like select all, copy, paste, etc*" (P3). This makes text-heavy activities on hardware keyboards more convenient and faster ("*knowing the shortcuts makes [text editing] very fast*" – P4) than on recent mobile devices ("*replacing existing text with copied text in Android at the right position is not really doable*" – P2). Shortcuts would circumvent the fat-finger problem to avoid imprecise selection, and would save time since a long-press to access the clipboard can be replaced.

The statements indicate that hardware keyboards are superior due to higher precision and more shortcuts. However, participants also stated that text-heavy activities on mobile devices are essential. Smartphones are mobile, which enables users to do a wide range of tasks while on the move. P13 stated that she "*[is] often doing things on the go and [does] not always have access to [her] laptop*" while P11 sees a clear advantage when "*[having] to send an email outside of the office, or to edit a document urgently over Dropbox or Google*". Additionally, touchscreen keyboards comprise useful features that are not available on their physical counterpart. This includes auto correction ("*with auto correct, I do*

Shortcut	Function	All	Word P.	Progr.
	Single-Key Shortcuts			
DOWN	Move cursor down.	10.00	7.47	8.41
RIGHT	Move cursor right	8.79	10.20	12.09
LEFT	Move cursor left.	7.80	11.45	9.73
UP	Move cursor up.	6.56	5.79	5.55
DELETE	Delete character ahead.	2.60	4.29	2.44
END	Move cursor to end of line.	2.08	3.95	2.24
HOME	Move cursor to start of line.	1.17	1.17	1.07
	Double-Key Shortcuts			
Ctrl+V	Paste from clipboard.	9.18	9.90	11.20
Alt+TAB	Switch windows.	7.84	4.84	9.63
Ctrl+C	Copy to clipboard.	7.64	5.31	6.21
Ctrl+S	Save document.	4.82	9.29	13.48
Ctrl+A	Select all.	1.68	0.41	0.76
Win+E	Launch file manager.	1.61	1.37	0.19
Ctrl+X	Cut to clipboard.	1.61	1.66	1.78
Ctrl+F	Open Search.	1.05	0.98	0.55
Ctrl+T	New Tab.	1.02	0.09	0.03
Ctrl+Z	Undo last action.	0.87	0.86	1.53
Shift+HOME	Select until start of line.	0.85	0.70	1.85
Shift+RIGHT	Select character ahead.	0.72	1.09	0.70
Shift+LEFT	Select previous character.	0.63	1.47	0.89
Ctrl+LEFT	Move cursor to prev. word.	0.41	1.31	0.21
Ctrl+RIGHT	Move cursor to next word.	0.33	1.27	0.11
	Triple-Key Shortcuts			
Ctrl+Shift+LEFT	Select last word.	0.49	1.27	0.86
Ctrl+Shift+RIGHT	Select word ahead.	0.41	1.11	0.36

Table 6.1: This table shows the percentage of occurrence of all shortcuts that represent at least 1% in the logs, averaged over all participants for system-wide, word processing and programming applications.

not always need to write the word completely" - P4) and one-handed interaction which can be useful during secondary tasks ("*I find on-screen keyboards easier than a normal keyboard on a computer because I can type with just one finger on my phone*" - P8).

Programming For PROGRAMMING tasks, we found 12 comments that address the OUTPUT. Compared to WORD PROCESSING, a "*large screen is required to analyze code in a comfortable way*" (P10) since "*code is often hundreds of lines long and can be barely interpreted, even on a normal-sized laptop screen. Viewing a large amount of text on a small screen would be annoying*" (P13). For challenges regarding INPUT, we found 77 comments that we clustered into three categories: typing speed, special characters and shortcuts.

Conforming with comments for WORD PROCESSING, the typing speed on touchscreen keyboards is affected by the small display size (*e.g.*, "*I cannot type as efficiently on an on-screen keyboard as on a hardware keyboard because I usually type with my two index fingers/thumbs instead of all ten fingers*" - P13). Moreover, special characters can only be accessed through switching layers or long-presses on most touchscreen keyboards due to the lack of modifier keys. This makes programming on touchscreens inconvenient since special characters are often required ("*I often need some specific characters which are only available by long-press*" - P7).

Participants predominantly mentioned the lack of shortcuts that make programming on hardware keyboards easier and faster ("*The use of shortcuts when using a hardware keyboard make coding tasks much easier and faster*" - P3). Specifically, shortcuts enable to navigate faster through structured code ("*functions to switch between methods (e.g., F3 in Eclipse)*" - P2) and text ("*Navigating text, which is often required when programming, is really hard on mobile devices.*" - P7). Moreover, programming environments enable users to create own shortcuts (*e.g.*, for formatting and renaming) that make programming faster in general ("*You can use a lot of shortcuts and also create your own ones, what results in faster performance.*" - P10). Thus, the majority (10 participants) reportedly look up shortcuts on the internet (*e.g.* "*Looking up through search engines.*" - P13) while

three participants reportedly look for shortcuts in menus. Despite the hardware keyboard's superiority, participants still find programming tasks essential in some situations (*e.g.* "*for emergency bug fixes*" - P1).

6.2.5 Discussion

We conducted an in-the-wild study with 15 expert users in which we analyzed the usage of shortcuts on hardware keyboards. We further interviewed them afterward about their experiences in text-heavy activities on touchscreen and hardware keyboards. We found that the participants performed around 800 shortcuts on average per day and identified 24 unique gestures that they frequently used. Of these, 22 benefit text-heavy applications and enable one to select text, place the caret, access the clipboard, and activate helper functions (*e.g.*, save and search). The majority of shortcuts are double-key shortcuts which are also available in the application menu. The frequent usage of shortcuts indicates that users prefer them over buttons and menus in the user interface. This is further supported by participants who voluntarily look up shortcuts in the internet or try them out based on the menu description.

Interviews revealed that touchscreen keyboards are inferior to hardware keyboards due to the lack of precision (*e.g.*, caret placement), shortcuts, and special characters for programming. Despite the disadvantages, participants highlighted that text-heavy activities are still essential on smartphones. Smartphones enable users to perform tasks in mobile situations (*e.g.*, writing emails or doing emergency bug fixes while out of office). This conforms with the prominence of mobile alternatives of established computer programs (*e.g.*, Word[1] and Excel[2]). Major challenges of touchscreen keyboards can be addressed by providing shortcuts to support mobile text editing. While the precision of caret placement can be increased with shortcuts known from hardware keyboards, shortcuts are necessary to provide a faster access to functions and special characters. Despite the necessity, recent touchscreen operating systems and keyboards do not provide shortcuts

[1] Microsoft Word: `https://play.google.com/store/apps/details?id=com.microsoft.office.word&hl=en`

[2] Microsoft Excel: `https://play.google.com/store/apps/details?id=com.microsoft.office.excel&hl=en`

that enable a quick access to these functions. Instead, direct touch makes precise caret placement and text selection difficult while long presses for accessing the clipboard slows down the usage.

Solutions based on on-screen gestures are not applicable for a larger number of shortcuts due to interference with gesture keyboards and ambiguity errors in recognition. Instead, we propose to extend the gesture input space to the whole device surface on *full-touch smartphones*. This extends the gesture input space which avoids interference and enables different gesture types (*e.g.* based on hand grip, pressure, side-dependent gestures) that can be used similar to modifier keys on hardware keyboards. Since user-defined gestures are easier to perform and more appropriate than designer-defined gestures [170], we conducted a gesture elicitation study to derive a gesture set for improving text-heavy activities.

6.3 Study II: Gesture Elicitation

We conducted a study to elicit on-device gestures to provide frequently used shortcuts on *full-touch smartphones*. We followed the method for gesture elicitation studies introduced by Wobbrock *et al.* [255] and used the AGreement Analysis Toolkit (AGATe) [237, 238] to analyze the collected gestures. The method by Wobbrock *et al.* [255] uses a within-subjects design to ask participants for gesture proposals in a randomized order.

6.3.1 Referents

We derived 22 distinct referents as shown in Table 6.3 based on the frequently used shortcuts of the previous study. We considered all shortcuts, except application or system specific shortcuts that are not necessarily required on mobile devices. This includes the shortcut for application switching (`Alt+TAB`) for which mobile platforms already provide their own methods, and the save shortcut (`Ctrl+S`) for saving a document which a number of text editors already do automatically. The referents shown in Table 6.1 represent basic actions that are either related to caret placement, text selection, clipboard management, or document switching.

Figure 6.2: Setup for the elicitation study. We used the tablet for camera preview and the laptop to control the referents on the smartphone.

6.3.2 Apparatus and Procedure

To avoid distracting participants with an unfamiliar prototype or recognizer in the gesture elicitation process, we followed the concept of a *magic brick* [204] capable of detecting any gesture that was performed. Participants tested and demonstrated their proposed gestures on an off-the-shelf LG Nexus 5. Further, we used a custom Android application to show the referents using screenshots of the state before and after the action was performed. Displayed screenshots were controlled by an application on the experimenter's computer while input on the smartphone was disabled to avoid any reactions of the UI. All proposed gestures were recorded with a GoPro Hero 3 (audio and video) as shown in Figure 6.2.

After obtaining informed consent, we briefed participants on the procedure and collected demographic data including their experiences in using mobile devices for text editing. We instructed participants to think-aloud and explain the thought process as well as the proposed gesture. We briefed participants that

Dimension	Category	Description
Nature	physical	Gesture acts physically on the object.
	symbolic	Gesture visually depicts a symbol.
	metaphorical	Gesture indicates a metaphor.
	abstract	Gesture-referent mapping is arbitrary.
Flow	discrete	Response occurs after the user acts.
	continuous	Response occurs while the user acts.
Complexity	simple	Gesture is atomic.
	compound	Gesture consists of atomic gestures.
Binding	object-centric	Location relative to object features.
	caret-centric	Location relative to caret features.
	mixed dep.	Any combination of above bindings.
	independent	Independent to any features.
Spatial	one-sided	Gesture performed on a single side.
	multi-sided	Gesture performed on multiple sides.

Table 6.2: Taxonomy of gestures for text editing shortcuts on a full-touch smartphone based on over 400 elicited gestures.

gestures could be performed on the whole device surface with one or two hands. The order of the referents was randomized. In total, the study took around 45 minutes.

6.3.3 Participants

We recruited 18 participants (5 female) between the ages of 20 and 34 (M = 23.9, SD = 3.5) who were staff or students at a technical university located in central Europe. None of the participants had participated in the previous study. All participants were right-handed with an average hand size of 19.1 cm (SD = 1.2 cm) and used their smartphones multiple times per day. We reimbursed them with 5 EUR.

6.3.4 Results

Three researchers transcribed each proposed gesture, grouped them in case they were identical, and assigned them to the taxonomy shown in Table 6.2. Moreover,

Category	Referent	$\mathcal{AR}$	$\mathcal{MAR}$	$\mathcal{CR}$
Caret Positioning	Move Up	.523	.418	.183
	Move Down	.431		
	Move Left	.359		
	Move Right	.359		
	Move to previous word	.183	.245	.111
	Move to next word	.307		
	Move to start of line	.294	.285	.098
	Move to end of line	.275		
Text Selection	Select Up	.242	.275	.098
	Select Down	.235		
	Select Left	.379		
	Select Right	.242		
	Select left word	.235	.268	.137
	Select right word	.301		
	Select to start of line	.196	.219	.065
	Select to end of line	.242		
	Select All	.137	-	
Clipboard management	Copy	.059	.076	.000
	Cut	.072		
	Paste	.098		
Navigation	Switch to next document	.098	.098	.098
	Switch to previous document	.098		

Table 6.3: Overview of referents. $\mathcal{AR}$ represent the agreement score of each gesture while $\mathcal{MAR}$ represents the average agreement score for the respective group. $\mathcal{CR}$ represents the coagreement score.

we transcribed the think-aloud protocols to analyze the thought process behind the proposed gestures. In total, participants performed 414 gestures from which we identified 138 unique gestures.

Taxonomy We classified the transcribed gestures along the dimensions shown in Table 6.2. We adapted the taxonomy by Wobbrock *et al.* [255] originally proposed for touch surfaces and added the dimensions *complexity* and *spatial* while changing the *binding* dimension to match on-device gestures. This resulted in five dimensions: *nature*, *temporal*, *complexity*, *binding* and *spatial*.

Nature describes the meaning of the gesture. *Physical* gestures directly manipulate objects (*e.g.*, dragging the caret) while *symbolic* gestures depict an object (*e.g.*, drawing scissors for cut). *Metaphorical* gestures manipulate imaginary objects (*e.g.*, tracing a finger in a circle to simulate a scroll ring). We categorized gestures as *abstract* if the meaning was arbitrary (*e.g.*, tapping three times to delete a selected word). *Flow* describes the visibility of a response; *discrete* if response is visible after a gesture was performed and *continuous* when response is shown while performing the gesture (*e.g.*, scrolling on the right edge). The *Complexity* describes the composition of a gesture; *compound* if it comprises of atomic gestures and *atomic* if not. *Binding* describes the relation to the referent. An *object-centric* gesture refers to an object (*e.g.*, a selected word) and *caret-centric* gestures refer to objects relative to the caret (*e.g.*, delete previous word). *Independent* gestures do not refer to any object. *Spatiality* describes whether gestures are performed on a single side or on multiple sides simultaneously or in succession.

Categorization of Gestures Figure 6.3 shows the taxonometric breakdown of the proposed gestures. Nearly half of the total gestures are physical gestures (44.9%). A large portion of the gestures are discrete (88.3%) and compound gestures (70.0%), whereas the majority refer to the caret's position (cursor-centric). Proposed gestures were performed on one side of the device (one-sided with 57.9%) and on multiple sides (41.7%).

Agreement Analysis We used AGATe by Vatavu and Wobbrock [237, 238] to calculate the agreement and coagreement rates. The agreement rate $\mathcal{AR}$ describes the participants' consensus on the proposed gestures while the coagreement rate $\mathcal{CR}$ represents the agreement shared between two referents r_1 and r_2. The scores range between 0 and 1, whereas 1 represents the highest agreement. The overall agreement rate for the elicited gesture set is $\mathcal{AR}$ = .236. We report the $\mathcal{AR}$ and $\mathcal{CR}$ for each referent in Table 6.3. We found a significant effect of referent type on agreement rate ($V_{rd(22,N=414)} = 314.798$, $p < .001$). Agreement rates for *caret positioning* referents are higher than the average with an $\mathcal{AR}$ of .359. We found a significant effect in this category ($V_{rd(7,N=144)} = 63.794$, $p < .001$). The agreement of *text selection* referents is $\mathcal{AR} = .214$ on average with a significant effect within the category ($V_{rd(8,N=162)} = 33.122$, $p < .001$). In contrast, the clipboard management category yielded a lower $\mathcal{AR}$ of .076 on average, while we found no significant agreement for all referents in this category ($V_{rd(2,N=54)} = 2.074$, $p = 1.000$). The group for moving the cursor yields the highest coagreement score with $\mathcal{CR} = .183$.

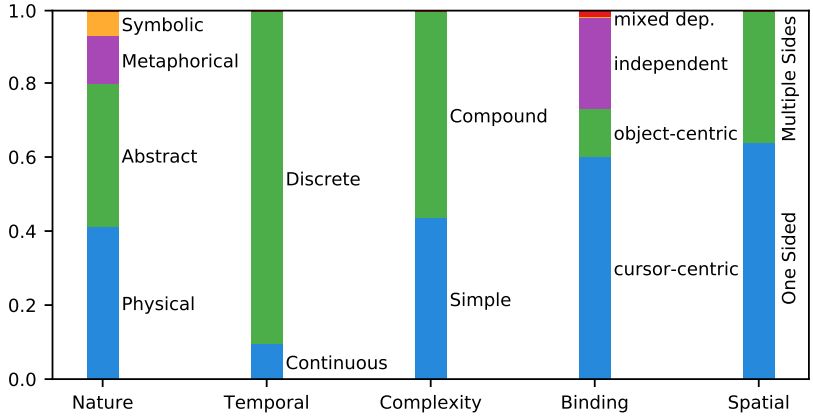


Figure 6.3: Distribution of gestures in taxonomy categories as shown in Table 6.2.

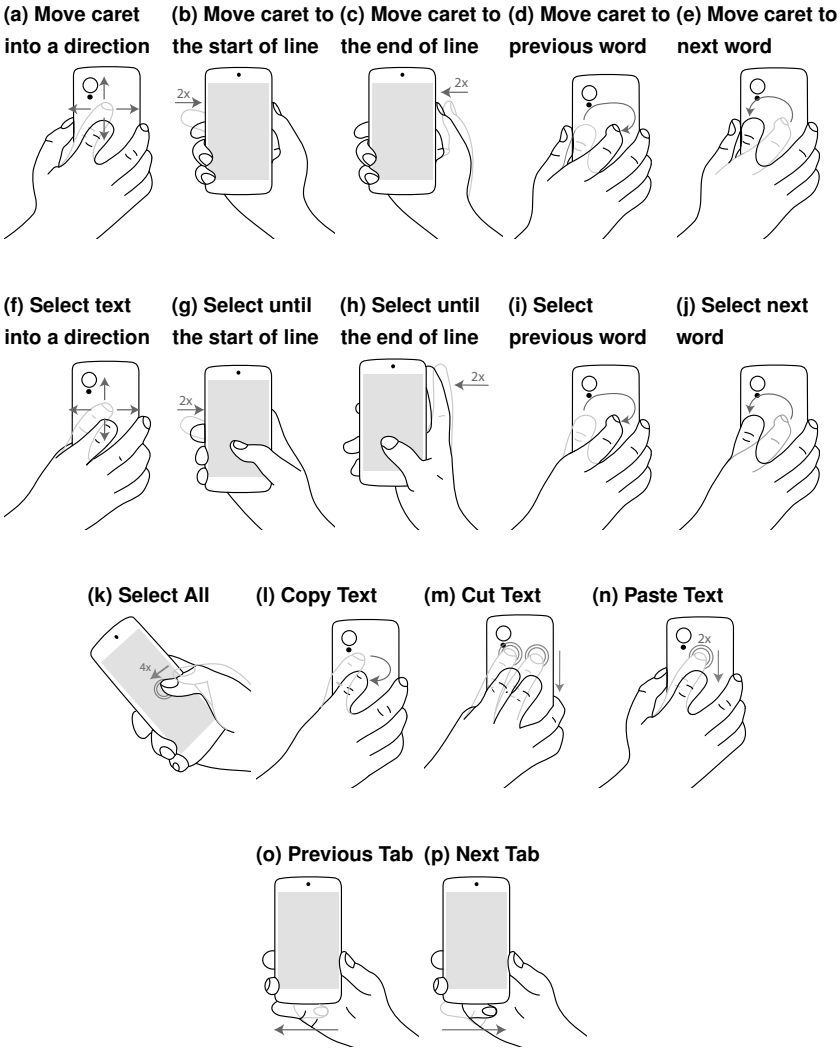


Figure 6.4: The gesture set to provide shortcuts for word processing and programming activities. Shortcuts are performed on a smartphone that is capable of sensing touch input on the whole device surface. Gestures for moving and selecting the caret are summarized into one Figure for all directions.

Mental Model Observations All participants aimed to propose gestures that were consistent with each other. Consistency as an explanation was given for 12.6% of the proposed gestures whereas each participant mentioned it 5.1 times on average. We observed that participants tried to recall which gesture they had proposed for a similar referent and even asked which gesture they had used before. This was especially the case for gestures in the *caret positioning* and *text selection* category. Six participants (P2, P3, P6, P9, P12, P14) tried to propose gestures that could be easily performed, especially when holding the device one-handed ("*[..] easy to do because index finger is already there*" - P2). Further, participants also explained that gestures were proposed to avoid unintended activations (*e.g.* "*more complex than just swiping so it doesn't happen accidentally*" - P1).

While the concept of on-device gestures was new to our participants, they (P1–P3, P6, P7, P10, P14, P17) proposed gestures based on their previous technical experiences. For example, holding different positions on the rear was compared with modifier keys from hardware keyboards ("*Pinkie and Ring, are like Ctrl and shift*" - P7, "*[..] holding like Ctrl button*" - P17). Further, we observed that participants changed the gesture they had in mind if they could not perform it with a sufficiently stable grip.

6.3.5 Gesture Set for Shortcuts in Text-Heavy Activities

The gesture set shown in Figure 6.4 consists of gestures with the highest agreement rate for each referent. Moving the caret can be done with the index finger on the back to avoid occlusion issues (see Figure 6.4a). Participants envisioned the caret to move relative to the finger and suggested a movement threshold to avoid unintended movements during grip changes. Inspired by hardware keyboards, text selection can be done analogously to caret movement using the thumb as a modifier (*c.f.* `Ctrl` and arrow keys, see Figures 6.4f to 6.4j). Moving the caret word-wise can be done by swiping the form of an arc to the respective direction (see Figures 6.4d and 6.4e). Participants explained this as a metaphorical leap over the previous/next word. Placing the caret at the start/end of the current line can be done with a double tap on the left/right side (see Figures 6.4g and 6.4h).

Selecting the whole text can be done with four taps on the front, while selected text can be copied by drawing the letter `C` on the back of the device

(see Figure 6.4l). This refers to the shortcut on hardware keyboards (`Ctrl+C`). Participants cut text into the clipboard by swiping down two fingers on the rear (see Figure 6.4m) symbolizing a scissor, and pasting text using a double tap followed by a swipe down on the rear (see Figure 6.4n). To switch between tabs, participants proposed swiping left and right on the bottom edge which can be done with the little finger when used one-handed, and the thumb when used two-handed. Similar to the caret placement, the tab should not be switched before the swipe is performed for a minimum distance to avoid unintentional switches.

6.3.6 Discussion

We conducted two studies to derive a gesture set that brings frequently used shortcuts from hardware keyboards to *full-touch smartphones*. Figure 6.4 shows the gesture set that we discuss in the following. The gesture set achieves an overall agreement rate of .236 which is in line with gesture sets elicited in other domains [13, 234]. The agreement scores for *caret positioning* and *text selection* are higher than the average which could be due to the simplicity of the action that can be projected to a physical gesture (*e.g.*, dragging the caret). In contrast, *clipboard management*, and *tab navigation* have lower agreement scores. This could be due to the abstractness of referents such as copying and cutting text that have no physical relations. Mobile operating systems often provide these functions through abstract buttons or pop-ups. Therefore, participants expressed these functions through symbolic gestures representing hardware keyboard shortcuts (*e.g.*, `Ctrl+C`), or real-world objects such as scissors to cut, and convenient combinations of abstract taps and swipes.

While participants assumed that the device is capable of detecting any performed gesture [204], recognizing the gesture set on a full-touch smartphone would comprise unintended activations, *e.g.*, through grip changes. Participants considered this challenge and proposed compound gestures (*e.g.*, tap then swipe) to counteract unintended activations. Moreover, the majority of the gestures are discrete so that the effect of activation is only shown after the gesture is fully performed. Thus, a series of distinct movements is required which makes unintended activations less likely. For continuous gestures (*e.g.*, moving the caret), participants proposed to use movement thresholds to discriminate unintended

movements (*e.g.*, grip change) from intended movements. For example, the index finger on the back needs to move a minimum distance before the caret moves. Moreover, unintended activations could be further decreased by only accepting gestures when the expected number of fingers is moved (*e.g.*, only the index finger is moving to position the caret).

6.4 Study III: Implementing the Gesture Set on a Full-Touch Smartphone

We implemented the elicited gesture set on our full-touch smartphone prototype which we presented in Section 5.1. We follow a similar approach based on deep learning as already shown in Chapter 4. First, we invited a set of new participants which we instructed to perform our gesture set on our full-touch smartphone. Second, we use the collected data set to train a Long short-term memory (LSTM) model which recognizes the gestures. In the following, we describe the study, model training and validation, as well as our mobile implementation.

We conducted a user study to collect a series of capacitive images for each gesture shown in Figure 6.4.

6.4.1 Apparatus

Based on the fully touch sensitive smartphone (FTSP) presented in Section 5.1, we collected capacitive images at $20\,fps$ with a resolution of $28 \times 32\,px$. Participants were seated on a chair without armrests in front of a display. The display shows instructions to the participant including the gesture to perform. Moreover, participants rated the gestures using a mouse after performing them on the full-touch smartphone.

6.4.2 Participants

We recruited 28 participants (21 male, 7 female) between the ages of 20 and 29 ($M = 23.7$, $SD = 3.4$) which did not participate in the previous studies. Two participants were left-handed, 26 right-handed. The average hand size was measured from the wrist crease to the middle fingertip and ranged from $16.0\,cm$ to $25.0\,cm$

($M = 19.3\,cm$, $SD = 1.6\,cm$). With this, our set of participants include samples of the 5th and 95th percentile of the anthropometric data reported in previous work [191]. Thus, the sample can be considered as representative. Participants were reimbursed with 10 EUR for their participation.

6.4.3 Procedure and Study Design

After obtaining informed consent, we asked participants to fill out a demographics questionnaire and measured their hand size. We then briefed them about the smartphone prototype and the purpose of the study. The study consists of six phases in which each of the gesture has to be performed by the participant. Using a custom application, the experimenter started and stopped the recording while instructing participants when to perform the gesture.

In the first phase, the experimenter instructed participants to perform the gesture as shown on the display in front of them. In case it was unclear, the experimenter further demonstrated the gesture on another smartphone. Before starting the recording of the respective gesture, participants were asked to perform the gesture on trial to ensure that everything was understood. In the second phase, we explained participants the function of each gesture before they were recorded. The display in front of them shows the function name while the smartphone shows exemplary before and after states of a text editor. Moreover, the experimenter explained each gesture orally and provided examples in case participants did not fully understand the function. After everything was understood, participants were asked to perform the gesture as if they are currently doing text editing tasks.

In the remaining four phases, we instructed participants to perform the gesture as shown on the display while the smartphone showed exemplary before and after states. These phases are used for collecting more data from the participants. In total, participants performed each gesture 6 times which results in $28 \times 6 = 168$ samples per gesture per participant.

6.4.4 Modeling

We describe the preprocessing and training steps which we performed to train a model for gesture recognition.

Data Set & Pre-Processing

We collected 477,605 capacitive images in total throughout the study. Our preprocessing steps includes synchronizing the capacitive images from all sides, cleaning the data set, and preparing the samples for an LSTM model. We performed the following four data preprocessing steps:

1. *Matching capacitive images from all sides:* Based on the timestamps of the capacitive images, we merged the capacitive images of all sides into a combined one with a size of $32 \times 28\,px$. To reduce the processor workload of a mobile deployment (*i.e.* the average number of capacitive images within one gesture), we merged on the front image and omitted all changes of the side sensors which happened between two capacitive images.

2. *Data Set Cleaning:* For each recorded gesture, we removed all frames in which no movements were happening (*i.e.* frames in which the participant only held the device to wait for the next task). This was done by determining blobs in the capacitive image and an element-wise equal operation[1] within a tolerance of $4.1\,mm$ (equals to the size of a pixel in the capacitive image). We further removed all gestures from our data set which did not include any movements to avoid errors during training.

3. *Summarizing Gestures*: Since the input on the front touchscreen is limited to either a thumb placement as a modifier (i.e. switching from moving cursor gestures to selecting gestures) or the 4x tap gesture for the select all action, we omitted the front data and summarized the gestures accordingly based on their back and side data. In particular, we summarized gestures (a-e) with (f-j) into the five respective classes and differentiate between selection and movement during run time based on the touch API from Android (i.e. whether a touch is registered on the front or not). Moreover, we removed the gestures b, c, g, and h since a simple double tap on the sides can be easily recognized manually. We ended up with 12 classes with the resulting capacitive images being $17 \times 28\,px$.

[1] We used numpy's allclose operation: `abs(a-b) <= (atol + rtol * abs(b))`

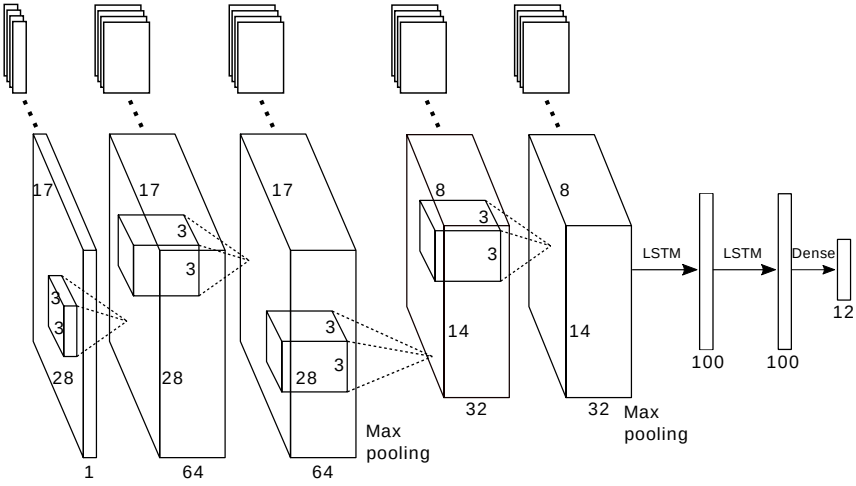


Figure 6.5: An illustration of the architecture of our CNN-LSTM model which we used to classify gestures on the full-touch smartphone. The first four layers are convolution layers wrapped in a `TimeDistributed` layer whose output is then fed into two subsequent LSTM layers with 100 units each. The output are 12 values representing the probabilities for each gesture class.

4. *Preparing samples for LSTM training*: Since LSTM models require a fixed input size (*i.e.* equal number of timesteps per gesture), we padded and trimmed gestures respectively to a sample size of 20 based on the average length of a gesture in our data set ($M = 16.04$, $SD = 5.7$).

In total, our data set consists of 114,720 capacitive images.

Model Architecture & Training

To train the model, we used a participant-wise split of 80%:20% for training and testing, *i.e.*, we trained the model on data from 22 participants and tested the model on the remaining 6 participants. We did not use a validation set as we evaluate the validation accuracy in the next study described in Section 6.5.

We implemented a CNN-LSTM [49, 57] using *Keras* 2.1.3 based on the *TensorFlow* backend. While we experimented with convolutional LSTMs [267, 281]

(a fully connected LSTM with convolutional input and recurrent transformations) in Keras[1] and stateful LSTMs with continuous capacitive image input, we found that a CNN-LSTM leads to the highest classification accuracy. We performed a grid search as proposed by Hsu *et al.* [101] to determine the most suitable network architecture and 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.

The architecture of our final CNN-LSTM is shown in Figure 6.5. We adapted the convolution layers of our *InfiniTouch* model (see Section 5.1) and used a `TimeDistributed`[2] layer to attach it to the LSTM part; a grid search focusing on the number of filters and kernel size in the convolution layer does not show any improvements in accuracy. After a further grid search on the LSTM part, we found that 100 LSTM units and a dropout factor of 0.7 achieved the highest accuracy. With this model, we further experimented with different number of timesteps (i.e. capacitive images) per gesture by increasing/decreasing them in steps of 5. We found that a window size of 20 (four more than the average) yielded the highest performance.

Similar to Section 5.1, we trained the CNN-LSTM with an RMSprop optimizer [229] but with a batch size of 16. Further, we determined .001 to be the most suitable initial learning rate after testing in steps of negative powers of 10. We experimented with batch normalization [103] and L2 Regularization, but did not find any improvements in the overall performance in our experiments.

Model Accuracy

Based on the test set, our model identifies the 12 gesture classes with an accuracy of 80.92 %, precision of 79.32 %, and recall of 78.28 %. Figure 6.8 shows the confusion matrix.

[1] Convolutional LSTMs are provided as ConvLSTM2D in Keras:
`https://keras.io/layers/recurrent/#ConvLSTM2D`

[2] `TimeDistributed` layer in *Keras*:
`https://keras.io/layers/wrappers/#TimeDistributed`

Figure 6.6: Confusion matrix of the CNN-LSTM for identifying 12 gesture classes with an accuracy of $80.92\,\%$. The values shown in the figure represent the accuracy in percent (%). The x-axis represents the predicted class while the y-axis represents the actual class.

6.4.5 Mobile Implementation

We used *TensorFlow Mobile*[1] for Android on the processing unit responsible for the front display to run the CNN that estimates the fingertip positions. Capacitive images from the back side and the edges are sent to the front device that merges the data into an input matrix. The input consists of a 17×28 8-bit image representing the front, back, and edges as shown in Section 5.1. A model inference for one capacitive image takes $163.7\,ms$ on average ($SD = 34.4\,ms$, $min = 101\,ms$, $max = 252\,ms$) over 1000 runs on our prototype. While we exported the model without any modifications, the inference time can be reduced significantly with optimization techniques such as quantization [78] and pruning [7] for a small loss of accuracy, or using recent processors which are optimized for neural networks[2] (*e.g.*, Snapdragon 845).

[1] TensorFlow Mobile website: `www.tensorflow.org/mobile/`

[2] `www.qualcomm.com/snapdragon/artificial-intelligence`

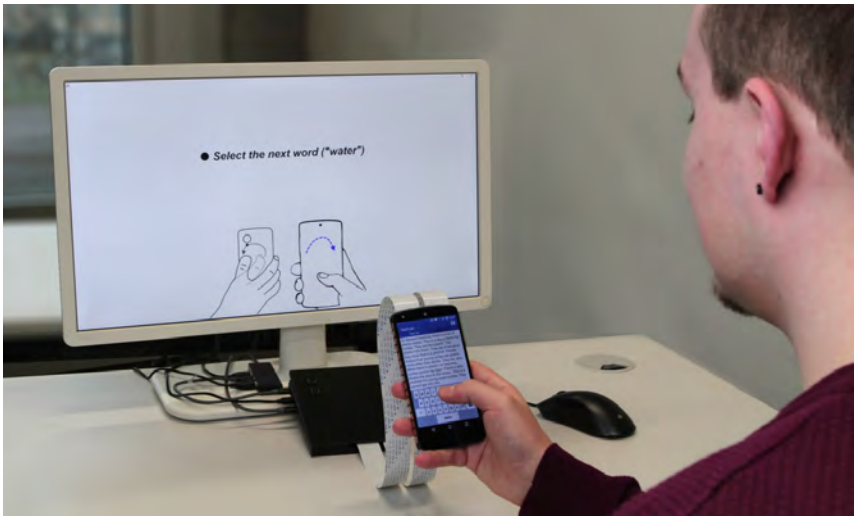


Figure 6.7: This figure shows the apparatus of the study. A participant holds our smartphone prototype to edit a text based on instructions shown on the screen in front of the participant.

6.5 Study IV: Evaluation of Shortcut Gestures

We conducted a user study in which we (i) validate the model accuracy with a new set of participants which were not involved in the model development process, (ii) collect qualitative feedback about our gesture set in a Wizard-of-Oz setting which excludes the effect of potential recognition errors, and (iii) collect qualitative feedback about the perceived usability of our prototype including the gesture recognizer, full-touch smartphone, and the text editing functions.

6.5.1 Study Procedure and Design

We designed three tasks to evaluate the three aspects described above. Prior to the study, we obtained informed consent, measured the participants' hand sizes, and handed them an instruction sheet which explains all parts of the study. Participants could refer to the instruction sheet at any time during the study.

Part 1: Accuracy Validation and Introduction of Gestures

This part is similar to the data collection study as described in Section 6.4. The focus is on obtaining a validation set with new participants while introducing and explaining the shortcuts for the next parts of the study. The experimenter explained each gesture to the participant using an image of the gesture displayed on the screen in front of the participant (see Figure 6.7), and by demonstrating it on another smartphone. Moreover, the experimenter explains the function of the gesture which is supported by an exemplary before and after state image shown on the full-touch smartphone.

Part 2: Wizard-of-Oz Evaluation of the Gesture Set

Based on a Wizard-of-Oz implementation of our gesture recognizer, we instructed participants to perform a set of pre-defined text editing tasks in our application. This part uses a Wizard-of-Oz implementation to avoid influencing participants with potentially wrong recognitions of our model. Using a 2×3 within-subjects design, we compare accessing functions for caret movement, text selection, and clipboard management of Android's recent methods (STANDARD) with our gesture set (GESTURE). As different font sizes affect the input precision, we used three font sizes from previous work by Fuccella *et al.* [61] to compare the two input methods: $1.75\,mm$, $3.25\,mm$, $4.75\,mm$. The conditions were counterbalanced with a Latin square and used six different texts to avoid learning effects.

Participants performed a set of text editing operations which were pre-ordered and described on the screen in front of the participant. Due to the fixed order, a trained experimenter could carefully observe the performed input of the participant to trigger the respective action using a Wizard-of-Oz controller on another smartphone. With the STANDARD method, the cursor can be moved by touching at the desired location. Text selection works similar with a long-press prior to the selection which also displays the clipboard functions in a menu on the top menu in the taskbar (*i.e.* ActionBar).

After each condition, we collected qualitative feedback in the form of a SUS, and the 7-point Likert scale questions as used in Section 3.3.6 and Section 4.2.3. At the end of this part, we further conducted a semi-structured interview in which we focused on comparing the conditions.

Part 3: Gesture Set Evaluation based on the Model

This part focuses on collecting qualitative feedback on the perceived usability of our gesture classifier for a new text with a font size of $3.25\,mm$. Similar to the previous step, participants were given a set of text editing operations which they performed based on our gesture set. After finishing the task, we conducted a semi-structured interview in which we focus on the perceived usability and usefulness of our gesture set for text editing tasks.

6.5.2 Apparatus

We developed a custom text editor application which provides the required functions for caret movement, selection, and clipboard management. For the use case evaluations, we included seven simple texts (six for the second part; one for the third part) from an English learning website[1] which are editable with an on-screen keyboard, as well as shortcut gestures or Android's standard text editing methods. The shortcuts can be activated either with a remote Android application on another smartphone (Wizard-of-Oz controller for Part 2), or by performing the shortcut gestures on the FTSP (Part 3). The Android application deployed on the back unit performs the gesture classification based on its own and the capacitive images from the sides and sends the classification result to the text editor application on the front unit.

Since gestures which require a thumb placement on the front touchscreen would recently interfere with the long-press mechanism of most Android keyboards (*e.g.*, punctuation marks on the built-in keyboard or GBoard), we developed our custom keyboard which detects long-presses and uses them to differentiate be-

[1] `https://www.newsinlevels.com/`

tween caret movement gestures (no thumb) and selection gestures (thumb on the front) instead of for inserting punctuation marks. The keyboard further changes its background color to indicate the selection mode.

6.5.3 Participants

We recruited 12 participants (6 female, 6 male) between the ages of 17 and 25 ($M = 21.1$, $SD = 2.7$). These participants did neither participate in Study II nor in Study III. All participants were right-handed (one was ambidextrous) and reportedly use mobile touch-based devices multiple times per day. Participants were reimbursed with a credit point for their lecture.

6.5.4 Results

We present the results of each part of the user study. The validation accuracy, as well as the ratings on easiness and suitability, are a result of the first part. Subjective ratings and the feedback gathered in the semi-structured interview are collected after the second part. The implementation which we evaluated in the third part was evaluated with semi-structured interviews and a questionnaire focusing on whether the participants would prefer our implementation or the recent text editing features integrated into Android. In total, the results describe the usefulness and usability of the gesture set, its use cases, and our prototypical implementation.

Validation Accuracy

Based on the capacitive images of gestures performed by the new set of participants, our model achieved a mean accuracy of 79.19 % ($SD = 8.21$ %, $min = 63.64$ %, $max = 89.66$ %). The mean precision is 80.88 % ($SD = 8.27$ %, $min = 62.22$ %, $max = 95.0$ %) while the recall is 77.67 % ($SD = 7.05$ %, $min = 62.5$ %, $max = 88.89$ %). The confusion matrix for this validation is shown in Figure 6.8.

Easiness and suitability of each gesture

Adapting the approach from previous work by Wobbrock *et al.* [255], we asked participants to rate the ease (*i.e.*, the gesture is easy to perform) and goodness

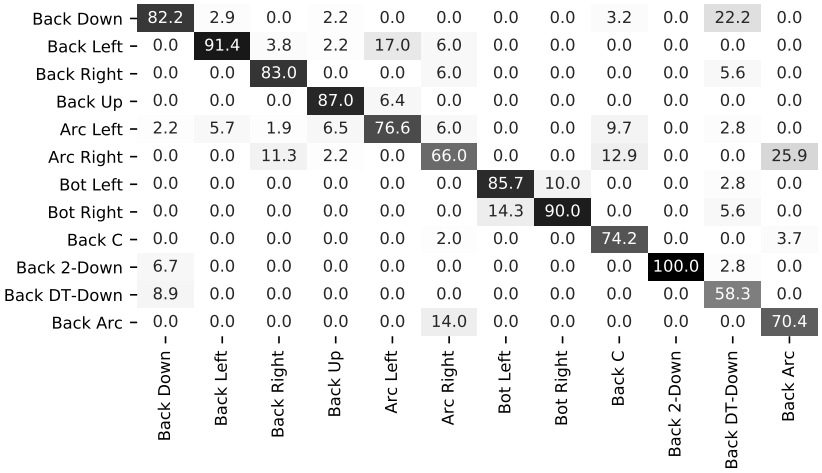


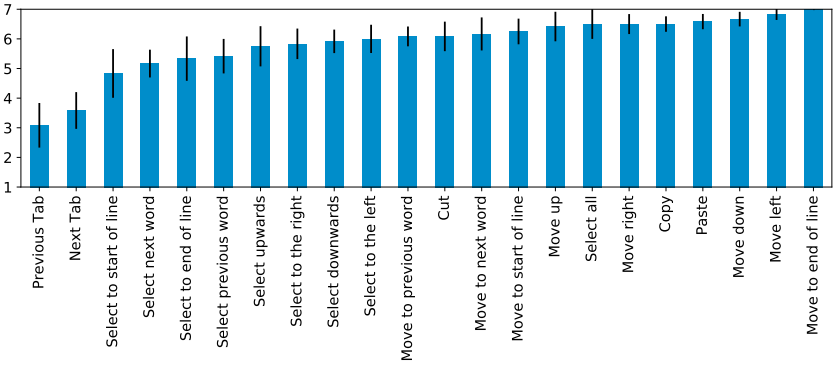
Figure 6.8: Confusion matrix describing the validation accuracy of the CNN-LSTM presented in Section 6.4.4. The values represent the accuracy in percent (%). The x-axis represents the predicted class while the y-axis represents the actual class.

(*i.e.*, the gesture is a good match for its intended purpose) of each gesture and its assigned function on a 7-point Likert scale. The results are shown in Figure 6.9. The average rating for ease is 5.84 (SD = .91, min = 3.31, max = 7.0), while the average goodness was rated at 6.06 (SD = .57, min = 4.77, max = 6.85).

Subjective Ratings

For each of the four gesture categories (*i.e.* placing, selecting, clipboard access, and tab switching), we collected subjective ratings in the form of 7-point Likert scale questions and semi-structured interviews. These are used to compare text editing using our gesture set with text editing operations as implemented in recent Android systems. Figure 6.10 shows the results. For each gesture category, we conducted two-way ANOVAs on the five ratings on which we applied the Aligned Rank Transform (ART) procedure using the ARTool [259] to align and rank the data.

(a) Ease rating ("The gesture is easy to perform").



(b) Goodness rating ("The gesture is a good match for its intended purpose").

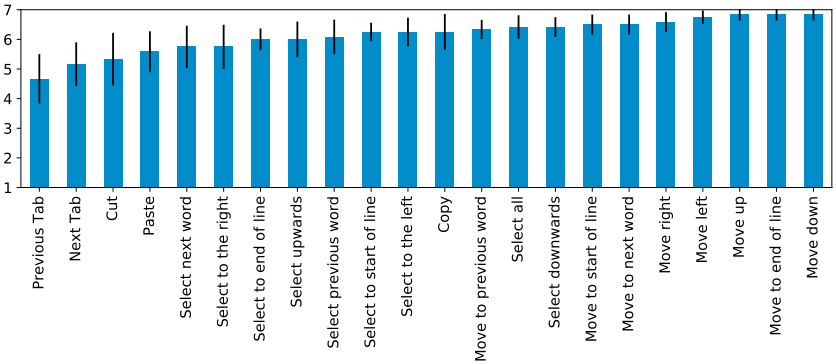


Figure 6.9: Participants' ratings on a 7-point Likert scale for the (a) easiness and (b) goodness of each gesture.

Variable	Easiness		Speed		Comfort		Suitability		Effort	
	F	p	F	p	F	p	F	p	F	p
SIZE	13.6	< .001	7.0	.002	6.6	.003	2.0	.147	5.6	.006
METHOD	62.9	< .001	3.4	< .001	32.4	< .001	1.4	.248	33.0	< .001
S×M	13.2	< .001	2.0	.1517	2.3	.138	.7	.522	3.8	.027
SIZE	4.9	.01	1.3	.28	2.2	.127	.9	.40	1.4	.246
METHOD	18.0	< .001	9.0	.004	4.7	.035	8.9	.004	14.5	< .001
S×M	6.5	< .001	1.6	.214	3.8	.027	1.2	.322	2.8	.07
SIZE	.3	.757	.1	.939	.2	.785	.1	.938	.1	.937
METHOD	14.2	< .001	2.1	.155	2.7	.105	4.6	.035	3.0	.087
S×M	1.1	.331	.7	.506	.2	.813	1.2	.318	.3	.764
SIZE	1.1	.327	.9	.413	1.1	.333	.4	.655	.4	.656
METHOD	189.9	< .001	167.2	< .001	189.2	< .001	111.1	< .001	174.0	< .001
S×M	1.8	.186	.3	.711	.8	.472	.1	.883	1.6	.217

Table 6.4: Results of the two-way ANOVAs for each gesture class and property. The row blocks refer to the gesture classes, from top to bottom: caret placement, text selection, clipboard management, tab switching. S × M refer to the two-way interaction effect between SIZE × METHOD.

For the *caret placing* gestures, we found significant main effects for SIZE and METHOD for the properties *easiness*, *speed*, *comfort*, and *effort* ($p < .01$) while we found a significant two-way interaction effect between SIZE × METHOD for the properties *easiness* and *effort* ($p < .05$). For the *text selecting* gestures, we found significant main effects for SIZE for the *easiness* property ($p = .01$), for METHOD for all properties ($p < .01$), and a significant two-way interaction effect between SIZE × METHOD for the properties *easiness* and *comfort* ($p < .05$). For the *clipboard access* gestures, we found significant main effects for METHOD for the properties *easiness* and *suitability* ($p < .05$). For the *tab switching* gestures, we found significant main effects for METHOD for all properties ($p < .001$). The tests did not reveal any other significant effects besides the ones described above. Table 6.4 provides an overview of all F and p values of the two-way ANOVA tests while Figure 6.10 shows the rating means and standard deviations.

Semi-Structured Interviews

We interviewed the participants after they performed the text editing tasks with our gesture set and Android's standard text editing mechanisms. Two researchers em-

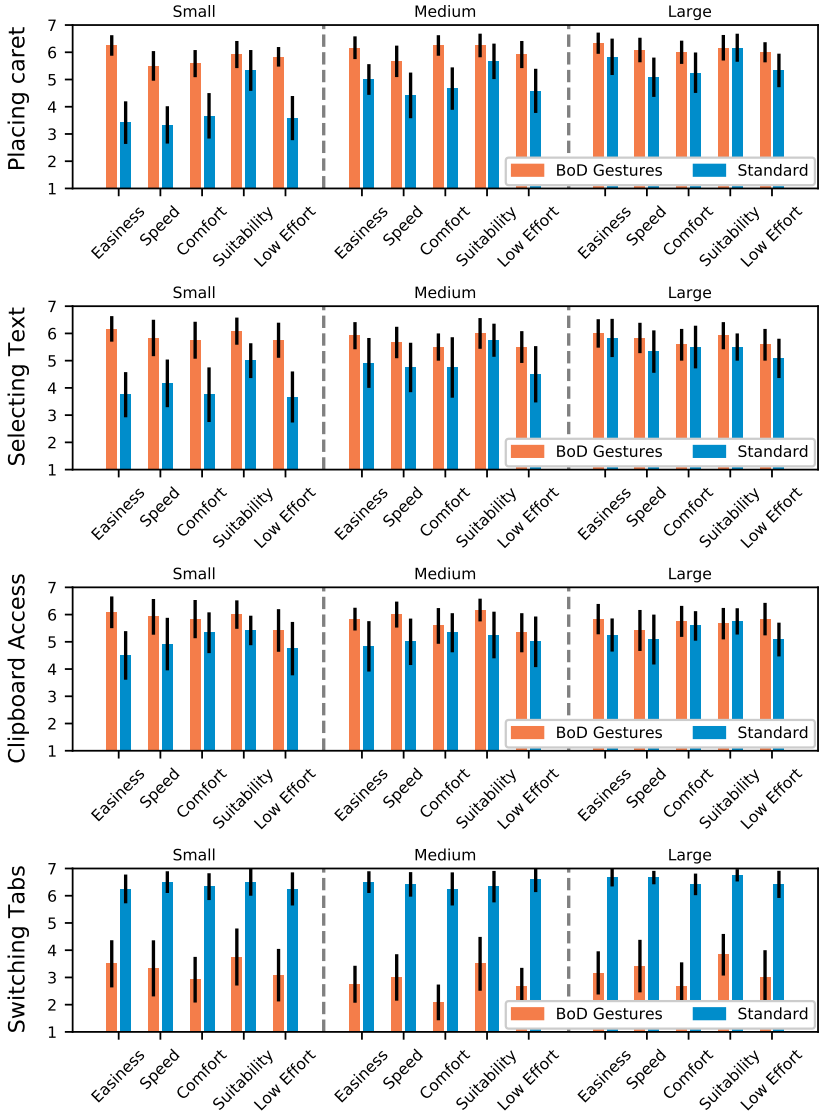


Figure 6.10: Subjective ratings after editing a text with standard touch input and our BoD gestures.

ployed a simplified version of qualitative coding with affinity diagramming [79] to analyze the results. This includes transcribing the interviews, extracting the arguments from the participants' answers, printing them on paper cards, and finally clustering the answers. In the following, we first present general impressions of text editing with gestures (including advantages, disadvantages, and neutral comments) and then present the feedback for each gesture group separately.

General Impressions When asked about the first impression of our text editing gestures, participants made 34 positive, 15 neutral, and 7 negative comments. All participants mentioned at least one positive aspect for text editing with our gestures. In particular, participants preferred our gestures over Android's standard text editing mechanisms due to the following reasons. Firstly, participants preferred the gestures to improve the usability and input pace with fingers on the back which were previously unused ("*I really like the idea, since we have many unused fingers on the back*" - P13; "*Some gestures improve the speed of use, especially if the user gets used to it*" - P11). Further, participants recognized that BoD gestures improve the input precision. Amongst others, P8 found that "*[BoD gestures] makes placing the cursor much easier*" and "*more precise*" (P2) which "*is much better especially for people with fat fingers*" (P12). While participants mentioned that the usability of Android's standard text editing mechanisms highly depends on the text size ("*the smaller the text is, the more you had to fiddle around*" - P12; "*if the text is very large, the normal version worked fine*" - P5), the usability of the our gesture set is independent from sizes of text, fingers, and hands ("*For me, the gestures were better for all text sizes*" - P4, "*an advantage is that, independent from the size of the hand and fingers, you can work with it relatively well*" - P10).

Participants also made comments which are positive but includes minor concerns. In particular, participants "*found some of the gestures difficult [to perform]*" (P3), however, "*normal touch is still [perceived as] worse*" (P3). Since some gestures were perceived as difficult (*e.g.*, "*I found the gesture for tab switching intuitive, but it was difficult to perform*" - P9), participants envisioned that they "*would use a mixture of both methods for text editing*" (P5, P8, P12). Regarding the

FTSP prototype, participants also found that "*the glass [of the back touchscreen] induces friction*" (P7) which makes performing gestures uncomfortable. However, this can be readily improved in a market-ready version.

Negative comments focus on the extended input space and indicate two challenges which concerned the participants. Firstly, they assumed that devices could become more expensive and fragile due to the additional touch sensors ("*the device could become much more complex with more sensors which could break*" - P13). Secondly, regarding the ergonomics, participants assumed that disadvantages include that more fingers need to be moved ("*[the user] needs all five fingers*" - P1; "*I need to move my fingers a lot*" - P6) and grip stability challenges for users which already have trouble holding a device in a single-handed grip ("*with small hands, some gestures are difficult to enter while holding the device*" - P10).

Impressions on Cursor Placing Gestures When asked about impressions on the cursor placing gestures, we received 10 positive, one neutral and one negative comment. All except P7 and P11 provided positive comments which describe the placing gestures as "*intuitive*" (P9), "*(very) helpful*" (P5, P8), and well working (P1, P2, P3, P4, P6, P9, P10, P12). P12 explained that "*the gestures on the back are much better, because [she] often moves the cursor beyond the target with normal touch because of [her] fat finger]*". In contrast to placing the cursor with direct touch (which is affected by the fat-finger problem), P5 and P9 found that "*[the gestures] work independently from the text size*" due to the indirect placement.

Participants further discussed the perceived effect of text size and explained that the standard text editing mechanisms are only usable for large font sizes while gestures are usable for all sizes ("*I found the standard method indeed not that bad when the text was large. However, for the smaller font, it became obvious that the gesture method is by far the more comfortable and faster method because placing the cursor with the finger [on the front display] is grisly.*" - P1). P7 confirms this and envisioned that "*[he] would sometimes combine it with the standard methods*" - P7). Only one participant "*[..] did not think that [he] is much better with the gestures than with the normal methods*" (P11).

Impressions on Text Selection Gestures Ten participants were positive towards the text selection gestures while two provided negative feedback. The majority of the participants found the text selection gestures "*intuitive*" (P12), "*helpful*" (P5, P8), "*very good*" (P6) and "*quick*" (P1, P2, P11). In general, participants "*found it great that a whole word can be selected with a quick gesture*" (P11). Two participants (P3, P13) found the text selection gestures difficult to perform as holding with the thumb while performing rear input is uncommon for them.

Impression on the Clipboard Gestures Eight participants provided positive, two neutral, and two negative comments on the clipboard gestures. In general, participants found the clipboard gesture "very good" (P6), "easy" (P10), "optimal" (P4), and "really fast compared to the normal touch version" (P1, P4, P12). P5 found the gesture fun to perform, "*especially since the cutting gesture feels like a claw*". While P2 and P3 were neutral towards the clipboard gestures, P2 felt that "*it was not very different to normal touch*" and P3 felt that "*[she] does not need the cut feature that often*". In terms of negative feedback, P11 found that "*it is sometimes difficult to move two fingers on the back*" while P7 "*finds buttons a little bit easier since [he] can just tap them instead of drawing a gesture*".

Impression on the Tab Switching Gestures Two participants were positive towards the tab switching gestures while ten provided negative comments. While P5 and P2 found tab switching "*intuitive*" and "*fine*", other participants reportedly commented that "*they would rather switch tabs with the [standard method]*" (P3, P4, P8, P5, P7, P12) especially since "*the gestures were difficult to perform*" (P4, P9). P4 and P5 thus suggested to place the tab switching gestures on the "*side close to the volume buttons*" (P4) and "*the top side*" (P5).

Feedback on the Implementation

We asked participants about their impressions after using our implementation of the text editing gestures. Four participants commended the implementation since they did not encounter any unexpected results which affected their perceived usability (*e.g.*, "*the recognition rate was fine!*" - P3; "*I completed all tasks and it worked fine*" - P4). The eight remaining participants also completed their tasks

successfully, but some gestures did not work well for them. For example, P7 found that "*placing the cursor and other [operation such as text selection and clipboard] worked fine, but gestures on the bottom side [for tab switching] did not work well for me*". This conforms with comments by P1, P2, P3, P5, and P12. P10 further mentioned a situation in which the cursor unexpectedly moved to the beginning of the line ("*most gestures worked quite well, but I unintentionally jumped to the beginning of the sentence once because I moved my hand*").

6.5.5 Discussion

We conducted an evaluation study in which we focused on three aspects: (i) validating the model accuracy with a new set of participants which were not involved in the data collection, (ii) collecting feedback on the gesture set, including ease and goodness as proposed by Wobbrock *et al.* [255] as well as qualitative feedback based on an evaluation within a realistic scenario, and (iii) collecting qualitative feedback on the usability of a prototype of the implemented gestures.

Model Validation

The model validation resulted in an accuracy which is close to the test accuracy in the development phase (79.19 % validation vs. 80.92 % test for 12 classes). Thus, we can assume that our model did not overfit to the participants of the data collection study. In general, these results indicate that BoD gestures on fully touch sensitive smartphones are feasible with a usable accuracy. While our prototype provided capacitive images with a resolution of $4.1\,mm \times 4.1\,mm$ per pixel which is common for mutual capacitive touchscreens, we expect the accuracy to further improve with a higher resolution based on sensing techniques such as FTIR [77] and IR sensing in the LCD layer (*e.g.* Samsung SUR40).

Feedback on the Gesture Set

The rated goodness and ease of all gestures (except the tab switching gestures) are in the positive range and in line with previous work [13, 234, 255]. This indicates that participants found the gestures easy to perform on our smartphone prototype and that they match their intended purpose.

We further evaluated the gesture set in realistic text editing scenarios and compared it with Android's standard text editing mechanisms. Participants rated the perceived easiness, speed, comfort, suitability, and effort after each condition. We adapted these properties from studies presented in Section 3.3 and Section 4.2. The results revealed that the caret placing, text selection, and clipboard access gestures are generally rated higher than their standard text editing counterparts. This conforms with the results from the interviews in which the majority of the participants preferred the gestures. Participants further mentioned in the interviews that the usability of the standard text editing mechanisms decreases with smaller font sizes while the gestures' usability stays consistent for all font sizes. This is also visible in the ratings (Figure 6.10). The tab switching gestures were generally rated as worse than simply touching a tab to switch. In summary, these results show that on-device gestures to activate text editing operations is feasible with a usable accuracy while perceived as more usable than the standard text editing operations implemented in recent touch-based operating systems.

Feedback on the Prototype Implementation

We asked participants for feedback on our prototypical implementation of the gesture set after they used them in another text editing scenario. All participants successfully completed the text editing task using our implementation. While four participants did not activate any function unintentionally, the remaining eight participants encountered unexpected cursor movements due to recognition errors. The interviews and observations of the experimenter revealed that recognition errors are a result of hand grip changes, performing a gesture in a way which does not conform to the data collection study (*e.g.*, different types of executions for the tab switching gestures due to ergonomic constraints), and the similarity of a number of gestures (*e.g.*, swiping down vs. double-tap and then swiping down) due to the low frame rate of the touch sensor. In summary, the feedback showed that our prototype already enables the use of on-device gestures as shortcuts to text editing operations with a usable accuracy and noticeable improvements over the standard text editing mechanisms.

6.6 General Discussion

We presented four studies focusing on bringing shortcuts for text editing from hardware keyboards to mobile devices. Each study represents a step in the UCDDL process presented in Section 1.2.3.

6.6.1 Summary

This chapter addresses RQ6 which focuses on the design and development of text editing shortcuts for fully touch sensitive smartphones. We first conducted an in-the-wild study to collect data about shortcuts which expert users perform during their work days. Based on the results, we derived a set of frequently used shortcuts. In the second study, we followed the approach by Wobbrock *et al.* [237, 238, 255] to elicit on-device gestures to provide text editing shortcuts for fully touch sensitive smartphone. The results are thus 23 user-defined gestures.

The remaining two studies focused on evaluating the gesture set based on metrics presented in previous work and on realistic text editing scenarios. We evaluate the gestures using a Wizard-of-Oz implementation as well as a prototype implementation based on the fully touch sensitive smartphone as presented in Section 5.1. Since the prototype is based on a functional gesture recognizer, we first collect a ground truth data set in the third study. After training a gesture recognizer, we then conduct the fourth study which focuses on gathering qualitative feedback on the gestures, their usability in realistic text editing scenarios, as well as the gesture recognizer performance to assess the feasibility of such gestures.

In summary, this set of studies revealed that on-device gestures for text editing were perceived as useful with a higher usability than state-of-the-art text editing mechanisms on recent operating systems. This chapter contributes with a set of frequently used shortcuts for hardware keyboards, the respective user-defined gesture set for fully touch sensitive smartphones, as well as a working and evaluated gesture recognizer. While this is a first milestone towards providing shortcuts for mobile text editing, future work could investigate whether mobile text editing requires further or other types of frequently used functions compared to hardware keyboards. This helps to refine our gesture set especially with a focus on the attributes of mobile devices, such as a rather small display.

6.6.2 Lessons Learned

Based on the results of our shortcut analysis, gesture elicitation, as well as the prototype development and evaluation, we derived the following insights:

Expert users leverage shortcuts to be more efficient in text editing. While text editing shortcuts are not widely adopted yet on mobile devices, expert users leverage shortcuts on hardware keyboards to speed up operations such as cursor navigation, text selection, managing the clipboard, or navigating through documents by switching tabs. Our results showed that experts perform around 800 shortcuts on keyboards of which 24 are unique. Of these, 22 benefit text-heavy applications.

Despite the limitations of mobile text editing, it is still needed. Our participants preferred text editing on computers based on a hardware keyboard over touch-based text editing on mobile devices. While this is due to the limitations of text editing functionalities and shortcuts, participants still emphasized that text editing on mobile devices is still needed and useful. Example use cases include writing emails, editing text documents, or doing emergency bug fixes while on the move.

Gestures should be simple but consider unintended inputs. We elicited on-device gestures to provide text editing shortcuts on fully touch sensitive smartphones. Analyzing the mental model of the participants, we found that they suggested gestures which are simple enough but still consider unintended activations by using compound gestures for functions which are more difficult to recover (*e.g.*, double tap and swipe down for paste).

On-device gestures are feasible with a usable accuracy. With our prototype and the gathered qualitative feedback, we showed that gestures performed on the device surface is feasible with a usable accuracy of around 80 %. This is made possible with a data-driven approach in which we first collect a ground truth data set of performed gestures, and then train a deep learning model to recognize these.

7

Conclusion and Future Work

Touchscreens are the main interface of mobile devices. While they enable intuitive interaction on handheld devices, their limited input capabilities slow down the interaction and pose a number of challenges which affect the usability.

In this thesis, we explored the concept of *hand-and-finger-awareness* for mobile touch interaction. This concept describes the use of multiple fingers for touch interaction, and the identification of these fingers to extend the input vocabulary. The thesis started with the description of the adapted version of the user-centered design (UCD) for deep leaning as the main methodology of this work, a presentation of the technical background, and a review of related work on extending touch interaction. In the first two studies presented in Chapter 3, we focused on understanding the hand ergonomics and behavior during mobile touch interaction. This enabled us to derive implications for the design of novel input methods using multiple fingers and on the whole device surface. With four studies presented in Chapter 4, we developed and evaluated two interaction methods for commodity smartphones which identify the source of touches and consider this information for extending the input vocabulary. Going one step further, we presented our fully touch sensitive smartphone prototype in Chapter 5 which enables touch input on the whole device surface and identifies the fingers

touching the device using deep learning. In addition, we interviewed experienced interaction designers to elicit solutions for addressing the challenges of mobile touch input using fully touch sensitive smartphones. In Chapter 6, we focus on mobile text editing as one specific use case and present a series of studies following the UCDDL to elicit, implement, and evaluate shortcuts for frequently used text editing operations on fully touch sensitive smartphones.

In the following, we summarize the research contribution, answer the research questions, and conclude with an outlook on future research directions.

7.1 Summary of Research Contributions

We contributed to three areas: (i) ergonomics and behavior of fingers for mobile interaction, (ii) methods for identifying the source of touch using deep learning, and (iii) solutions for addressing limitations of mobile touch input. We followed the UCDDL for the development and evaluation of the interaction methods.

Chapter 3 focuses on understanding the ergonomics and behavior of all fingers for mobile interaction. In Section 3.2, we studied the ergonomics of all fingers while holding mobile devices in a single-handed grip. This contributes to the understanding of the comfortable area and the maximum range of all fingers which are an important basis for the design of input controls especially on fully touch sensitive smartphones. With the study presented in Section 3.3, we investigated supportive micro-movements of fingers which users implicitly perform to maintain a stable grip, increase reachability of the thumb, and due to the limited independence of the fingers. Understanding supportive micro-movements of all fingers helps to minimize unintended inputs on fully touch sensitive smartphones.

Chapter 4 focuses on identifying the source of touch on commodity capacitive touchscreens using deep learning. We contribute with two novel input techniques for touchscreens including the development and validation of the models, as well as an evaluation of the input techniques in realistic scenarios. We showed that identifying and using the palm for activating functions is feasible with a high accuracy while perceived as natural and fast by users. Moreover, we investigated the feasibility of differentiating between different fingers and found that left and right thumbs can be identified with a usable accuracy.

Chapter 5 contains a technical contribution in the form of a reproducible fully touch sensitive smartphone prototype which identifies touches of all fingers holding the device. Further, we present a set of use cases made possible with our prototype, and further interviewed experienced interaction designers to contribute solutions which address common touch input limitations with our prototype.

Finally, Chapter 6 focuses on mobile text editing as a specific use case and contributes a gesture set which was inspired by shortcuts on hardware keyboards, elicited using a user-centered approach, and evaluated on our fully touch sensitive smartphone. In the following, we revisit the research questions addressed in this thesis:

RQ1: *How can we design Back-of-Device input controls to consider the reachability of fingers in a single-handed grip?* Our results presented in Section 3.2 revealed that the index and middle finger are the most suited for BoD input. We further reveal the areas in which the fingers can move without changing the grip or stretching the thumb in an uncomfortable way. Placing input controls within the comfortable area minimizes finger movements which lead to muscle strain or to losing the stable grip which could result in dropping the device.

RQ2: *How can we design Back-of-Device input controls to minimize unintended inputs, e.g. due to supportive finger movements caused by maintaining the grip?* The work presented in Section 3.3 analyzed and describes supportive finger movements (*e.g.*, to maintain a stable grip, increase the thumb's reachability and due to the limited independence of the fingers [76]) which could lead to unintended inputs on the back. We derived the *safe areas* which are comfortably reachable (supporting RQ1) but also entails the least amount of unintended inputs (addressing RQ2). We further found that the least unintended inputs occurred on a $5''$ devices due to a suitable compromise of grip stability, reachability, and size of on-screen targets.

RQ3: *How can we differentiate between fingers and hand parts on a capacitive touchscreen?* Based on the approach presented in Section 4.2, we showed that the raw data of capacitive touch sensors can be used to reliably identify touches of fingers and hand parts. Evaluating *PalmTouch*, we showed that

the palm can be identified with an accuracy of 99.53 % while perceived as an intuitive and natural way to perform touch input. Moreover, we showed that the left and right thumb can be identified with a usable accuracy of over 92 %. In general, we found that the raw data of capacitive touchscreens in combination with deep learning can be used to develop novel input techniques which were not feasible before without wearable sensors or additional sensors attached to the device.

RQ4: *How can we estimate the position of individual fingers and identify them on a fully touch sensitive smartphone?* The raw data of capacitive touchscreens enable to identify palm touches and differentiate between left and right thumbs with accuracies usable for representative tasks. With our fully touch sensitive smartphone prototype, we showed that all fingers of the holding hand can be identified with a similar accuracy. Our prototype enables all fingers on the back (which were previously unused) to perform individual input. Again, we used a deep learning based approach and the raw capacitive data of our prototype to estimate the position of individual fingers on the device. Combined with a simple mapping approach, individual fingers can be identified with an accuracy of 95.78 %.

RQ5: *Which typical touch input limitations could be solved with a fully touch sensitive smartphone?* With the support of experienced interaction designers, we derived a number of solutions and interaction techniques for fully touch sensitive smartphones which address touch input challenges such as the reachability issues, limited input capabilities, and the fat-finger problem. Among others, reachability issues can be addressed by combining the range of the thumb and the fingers on the back side. Further, a wide range of shortcuts can be provided using finger-specific gestures on the back or using the rear fingers' placement as action modifiers while the occlusion issues caused by the fat-finger problem could be addressed with moving input controls outside of the front display (*e.g.*, edge or back).

RQ6: *How can we design and use shortcuts on a fully touch sensitive smartphone to improve text editing?* With four user studies presented in Chapter 6, we focused on improving text editing on mobile devices with shortcuts as a

specific use case. We identified text editing operations which are frequently performed on computers via shortcuts, and conducted a gesture elicitation study to understand how users envision to perform these shortcuts on fully touch sensitive smartphones. With the two subsequent studies, we developed a gesture recognizer and evaluated it with realistic text editing scenarios on a fully touch sensitive smartphone. In total, we showed that users prefer editing text with our prototype over text editing operations recently implemented on mobile operating systems, and that shortcuts significantly improve the usability of mobile text editing tasks.

7.2 Future Work

While this thesis focused on hand-and-finger-aware interaction on mobile touch-based devices, in the course of this work, we discovered new challenges beyond the scope of this thesis. In the following, we point directions for future research.

Hand-and-Finger-Aware Interaction on Tangible Objects. While we enabled all fingers and the palm to perform individual input on fully touch sensitive smartphones, this concept could be transferred to tangibles with touch sensing capability on the whole object surface. With flexible capacitive touch sensors becoming inexpensive and readily available (*e.g.*, [180]), tangibles could recognize the hand grip for context-awareness, enable different fingers to perform finger-specific gestures, or use the finger placement as a modifier. This concept enriches the input vocabulary of tangibles without changing its form factor or adding additional input controls.

Machine Learning for Interaction with Humans in the Loop. With users getting used to specific interaction methods, they also adapt the way they perform input over the time to achieve the best result with the machine learning model. However, this type of input was not considered and captured during the data collection study for training, so that models can become unstable over time against the user's expectation. With an online learning approach, user interfaces could collect data and context of an interaction to continuously improve over time and adapt to the way users perform input.

Generalizability beyond Specific Devices. We showed the feasibility of our interaction techniques using commodity sensors and optimized our models accordingly. Future work could train device and sensor independent models which work with capacitive images of every type of device (*e.g.*, different sizes, resolutions, and sensitivities). This could be done by training a general model (*e.g.*, training based on normalized, fixed-sized capacitive images) based on data from a set of devices and evaluating it with another set of devices.

Modeling Hand Ergonomics and Behavior for Computational Interaction. In this thesis, we empirically studied the ergonomics and behavior of fingers in order to derive design implications for novel interaction methods. While we developed and evaluated novel interaction techniques based on the design implications and further user feedback, future work could derive mathematical models of ergonomics and behavior of fingers in order to use computational interaction [186] for model-driven UI optimization and derivation of novel interaction techniques.

Discoverability of Novel Interaction Techniques. While this thesis focused on the usability and technical feasibility of *hand-and-finger-awareness* for mobile touch interaction, we suggest future work to investigate methods to communicate these interaction techniques to new users. This is an important topic especially for bringing novel interaction methods on a new type of mobile device (*i.e.* a fully touch sensitive smartphone) to the mass-market for a wide range of users. As the affordance of touch-based input techniques is limited, novel input techniques needs to be communicated to improve the discoverability as emphasized by Shneiderman *et al.* [216] and Norman [181].

Bibliography

[1] C. Ackad, A. Clayphan, R. M. Maldonado, J. Kay. “Seamless and Continuous User Identification for Interactive Tabletops Using Personal Device Handshaking and Body Tracking.” In: *CHI ’12 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’12. Austin, Texas, USA: ACM, 2012, pp. 1775–1780. ISBN: 978-1-4503-1016-1. DOI: 10.1145/2212776.2223708 (cit. on p. 47).

[2] A. Agarwal, S. Izadi, M. Chandraker, A. Blake. “High Precision Multi-touch Sensing on Surfaces using Overhead Cameras.” In: *Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP’07)*. 2007, pp. 197–200. DOI: 10.1109/TABLETOP.2007.29 (cit. on p. 33).

[3] M. Annett, T. Grossman, D. Wigdor, G. Fitzmaurice. “Medusa: A Proximity-aware Multi-touch Tabletop.” In: *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. UIST ’11. Santa Barbara, California, USA: ACM, 2011, pp. 337–346. ISBN: 978-1-4503-0716-1. DOI: 10.1145/2047196.2047240 (cit. on p. 45).

[4] L. Anthony, R.-D. Vatavu, J. O. Wobbrock. “Understanding the Consistency of Users’ Pen and Finger Stroke Gesture Articulation.” In: *Proceedings of Graphics Interface 2013*. GI ’13. Regina, Sascatchewan, Canada: Canadian Information Processing Society, 2013, pp. 87–94. ISBN: 978-1-4822-1680-6 (cit. on p. 40).

[5] L. Anthony, J. O. Wobbrock. “A Lightweight Multistroke Recognizer for User Interface Prototypes.” In: *Proceedings of Graphics Interface 2010*. GI ’10. Ottawa, Ontario, Canada: Canadian Information Processing Society, 2010, pp. 245–252. ISBN: 978-1-56881-712-5 (cit. on p. 40).

[6] L. Anthony, J. O. Wobbrock. “$N-protractor: A Fast and Accurate Multistroke Recognizer.” In: *Proceedings of Graphics Interface 2012*. GI ’12. Toronto, Ontario, Canada: Canadian Information Processing Society, 2012, pp. 117–120. ISBN: 978-1-4503-1420-6 (cit. on p. 153).

[7] S. Anwar, K. Hwang, W. Sung. “Structured Pruning of Deep Convolutional Neural Networks.” In: *J. Emerg. Technol. Comput. Syst.* 13.3 (Feb. 2017), 32:1–32:18. ISSN: 1550-4832. DOI: 10.1145/3005348 (cit. on pp. 107, 126, 153, 195).

[8] O. K.-C. Au, C.-L. Tai. “Multitouch Finger Registration and Its Applications.” In: *Proceedings of the 22Nd Conference of the Computer-Human Interaction Special Interest Group of Australia on Computer-Human Interaction*. OZCHI ’10. Brisbane, Australia: ACM, 2010, pp. 41–48. ISBN: 978-1-4503-0502-0. DOI: 10.1145/1952222.1952233 (cit. on pp. 46, 100).

[9] J. Avery, S. Malacria, M. Nancel, G. Casiez, E. Lank. “Introducing Transient Gestures to Improve Pan and Zoom on Touch Surfaces.” In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI ’18. New York, NY, USA: ACM, 2018. DOI: 10.1145/3173574.3173599 (cit. on p. 155).

[10] P. Bader, V. Schwind, N. Henze, S. Schneegass, N. Broy, A. Schmidt. “Design and Evaluation of a Layered Handheld 3D Display with Touch-sensitive Front and Back.” In: *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*. NordiCHI ’14. Helsinki, Finland: ACM, 2014, pp. 315–318. ISBN: 978-1-4503-2542-4. DOI: 10.1145/2639189.2639257 (cit. on pp. 17, 48, 49, 53).

[11] P. Bader, H. V. Le, J. Strotzer, N. Henze. “Exploring Interactions with Smart Windows for Sunlight Control.” In: *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’17. Denver, Colorado, USA: ACM, 2017, pp. 2373–2380. ISBN: 978-1-4503-4656-6. DOI: 10.1145/3027063.3053242 (cit. on p. 26).

[12] P. Bader, A. Voit, H. V. Le, P. W. Woźniak, N. Henze, A. Schmidt. “WindowWall: Towards Adaptive Buildings with Interactive Windows As Ubiquitous Displays.” In: *ACM Trans. Comput.-Hum. Interact.* TOCHI 26.2 (Mar. 2019), 11:1–11:42. ISSN: 1073-0516. DOI: 10.1145/3310275 (cit. on p. 26).

[13] G. Bailly, T. Pietrzak, J. Deber, D. J. Wigdor. “MéTamorphe: Augmenting Hotkey Usage with Actuated Keys.” In: *Proceedings of the SIGCHI Conference on Human*

Factors in Computing Systems. CHI '13. Paris, France: ACM, 2013, pp. 563–572. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2470734 (cit. on pp. 189, 207).

[14] A. Bangor, P. Kortum, J. Miller. "Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale." In: *J. Usability Studies* 4.3 (May 2009), pp. 114–123. ISSN: 1931-3357 (cit. on p. 113).

[15] G. Barrett, R. Omote. "Projected-capacitive touch technology." In: *Information Display* 26.3 (2010), pp. 16–21 (cit. on p. 33).

[16] P. Baudisch, G. Chu. "Back-of-device Interaction Allows Creating Very Small Touch Devices." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '09. Boston, MA, USA: ACM, 2009, pp. 1923–1932. ISBN: 978-1-60558-246-7. DOI: 10.1145/1518701.1518995 (cit. on pp. 17, 47, 48, 53, 55, 98, 162, 164, 167, 172).

[17] Y. Bengio, A. Courville, P. Vincent. "Representation Learning: A Review and New Perspectives." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8 (2013), pp. 1798–1828. DOI: 10.1109/TPAMI.2013.50 (cit. on pp. 104, 105, 123).

[18] H. Benko, A. D. Wilson, P. Baudisch. "Precise Selection Techniques for Multi-touch Screens." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '06. Montreal, Quebec, Canada: ACM, 2006, pp. 1263–1272. ISBN: 1-59593-372-7. DOI: 10.1145/1124772.1124963 (cit. on p. 46).

[19] H. Benko, T. S. Saponas, D. Morris, D. Tan. "Enhancing Input on and Above the Interactive Surface with Muscle Sensing." In: *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ITS '09. Banff, Alberta, Canada: ACM, 2009, pp. 93–100. ISBN: 978-1-60558-733-2. DOI: 10.1145/1731903.1731924 (cit. on pp. 45, 100).

[20] J. Bergstrom-Lehtovirta, A. Oulasvirta. "Modeling the Functional Area of the Thumb on Mobile Touchscreen Surfaces." In: *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*. CHI '14. Toronto, Ontario, Canada: ACM, 2014, pp. 1991–2000. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557354 (cit. on pp. 17, 36, 55, 63, 65, 70, 98, 156).

[21] J. Bergstrom-Lehtovirta, A. Oulasvirta, S. Brewster. “The Effects of Walking Speed on Target Acquisition on a Touchscreen Interface.” In: *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*. MobileHCI ’11. Stockholm, Sweden: ACM, 2011, pp. 143–146. ISBN: 978-1-4503-0541-9. DOI: 10.1145/2037373.2037396 (cit. on pp. 72, 74, 89, 167).

[22] B. Blažica, D. Vladušič, D. Mladenić. “MTi: A method for user identification for multitouch displays.” In: *International Journal of Human-Computer Studies* 71.6 (2013), pp. 691 –702. ISSN: 1071-5819. DOI: https://doi.org/10.1016/j.ijhcs.2013.03.002 (cit. on p. 47).

[23] D. Bonnet, C. Appert, M. Beaudouin-Lafon. “Extending the Vocabulary of Touch Events with ThumbRock.” In: *Proceedings of Graphics Interface 2013*. GI ’13. Regina, Sascatchewan, Canada: Canadian Information Processing Society, 2013, pp. 221–228. ISBN: 978-1-4822-1680-6 (cit. on pp. 20, 40, 41).

[24] S. Boring, D. Ledo, X. A. Chen, N. Marquardt, A. Tang, S. Greenberg. “The Fat Thumb: Using the Thumb’s Contact Size for Single-handed Mobile Interaction.” In: *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services*. MobileHCI ’12. San Francisco, California, USA: ACM, 2012, pp. 39–48. ISBN: 978-1-4503-1105-2. DOI: 10.1145/2371574.2371582 (cit. on pp. 16, 20, 41, 94).

[25] S. Boring, D. Baur, A. Butz, S. Gustafson, P. Baudisch. “Touch Projector: Mobile Interaction Through Video.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’10. Atlanta, Georgia, USA: ACM, 2010, pp. 2287–2296. ISBN: 978-1-60558-929-9. DOI: 10.1145/1753326.1753671 (cit. on p. 43).

[26] N Brook, J Mizrahi, M Shoham, J Dayan. “A biomechanical model of index finger dynamics.” In: *Medical engineering & physics* 17.1 (1995), pp. 54–63 (cit. on p. 37).

[27] J. Brooke. “SUS: A Retrospective.” In: *J. Usability Studies* 8.2 (Feb. 2013), pp. 29–40. ISSN: 1931-3357 (cit. on pp. 113, 115).

[28] J. Brooke. “SUS-A quick and dirty usability scale.” In: *Usability evaluation in industry* 189.194 (1996), pp. 4–7 (cit. on p. 113).

[29] D. Buschek, F. Alt. “TouchML: A Machine Learning Toolkit for Modelling Spatial Touch Targeting Behaviour.” In: *Proceedings of the 20th International Conference on Intelligent User Interfaces*. IUI ’15. Atlanta, Georgia, USA: ACM, 2015, pp. 110–114. ISBN: 978-1-4503-3306-1. DOI: 10.1145/2678025.2701381 (cit. on pp. 100, 135).

[30] X. Cao, A. D. Wilson, R. Balakrishnan, K. Hinckley, S. E. Hudson. “ShapeTouch: Leveraging contact shape on interactive surfaces.” In: *2008 3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems*. 2008, pp. 129–136. DOI: 10.1109/TABLETOP.2008.4660195 (cit. on p. 46).

[31] R. Caruana, S. Lawrence, C. L. Giles. “Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping.” In: *Advances in neural information processing systems*. 2001, pp. 402–408 (cit. on p. 106).

[32] E. Cattan, A. Rochet-Capellan, P. Perrier, F. Bérard. “Reducing Latency with a Continuous Prediction: Effects on Users’ Performance in Direct-Touch Target Acquisitions.” In: *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*. ITS ’15. Madeira, Portugal: ACM, 2015, pp. 205–214. ISBN: 978-1-4503-3899-8. DOI: 10.1145/2817721.2817736 (cit. on p. 147).

[33] W. Chang, K. E. Kim, H. Lee, J. K. Cho, B. S. Soh, J. H. Shim, G. Yang, S.-J. Cho, J. Park. “Recognition of grip-patterns by using capacitive touch sensors.” In: *Industrial Electronics, 2006 IEEE International Symposium on*. Vol. 4. IEEE. 2006, pp. 2936–2941 (cit. on pp. 17, 49, 157).

[34] Y. Chang, S. L’Yi, K. Koh, J. Seo. “Understanding Users’ Touch Behavior on Large Mobile Touch-Screens and Assisted Targeting by Tilting Gesture.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI ’15. Seoul, Republic of Korea: ACM, 2015, pp. 1499–1508. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702425 (cit. on pp. 38, 50, 55, 56).

[35] L.-P. Cheng, H.-S. Liang, C.-Y. Wu, M. Y. Chen. “iGrasp: Grasp-based Adaptive Keyboard for Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’13. Paris, France: ACM, 2013, pp. 3037–3046. DOI: 10.1145/2470654.2481422 (cit. on pp. 17, 167).

[36] L.-P. Cheng, F.-I. Hsiao, Y.-T. Liu, M. Y. Chen. “iRotate Grasp: Automatic Screen Rotation Based on Grasp of Mobile Devices.” In: *Adjunct Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST Adjunct Proceedings ’12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 15–16. ISBN: 978-1-4503-1582-1. DOI: 10.1145/2380296.2380305 (cit. on p. 17).

[37] V. Cheung, J. Heydekorn, S. Scott, R. Dachselt. “Revisiting Hovering: Interaction Guides for Interactive Surfaces.” In: *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 355–358. ISBN: 978-1-4503-1209-7. DOI: 10.1145/2396636.2396699 (cit. on p. 44).

[38] A. Colley, J. Häkkilä. “Exploring Finger Specific Touch Screen Interaction for Mobile Phone User Interfaces.” In: *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: The Future of Design*. OzCHI ’14. Sydney, New South Wales, Australia: ACM, 2014, pp. 539–548. ISBN: 978-1-4503-0653-9. DOI: 10.1145/2686612.2686699 (cit. on pp. 16, 46, 51, 94, 95, 168).

[39] C. Corsten, B. Daehlmann, S. Voelker, J. Borchers. “BackXPress: Using Back-of-Device Finger Pressure to Augment Touchscreen Input on Smartphones.” In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI ’17. Denver, Colorado, USA: ACM, 2017, pp. 4654–4666. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025565 (cit. on pp. 17, 20, 48, 49, 53).

[40] C. Corsten, C. Cherek, T. Karrer, J. Borchers. “HaptiCase: Back-of-Device Tactile Landmarks for Eyes-Free Absolute Indirect Touch.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI ’15. Seoul, Republic of Korea: ACM, 2015, pp. 2171–2180. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702277 (cit. on p. 37).

[41] C. Corsten, A. Link, T. Karrer, J. Borchers. “Understanding Back-to-front Pinching for Eyes-free Mobile Touch Input.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI ’16. Florence, Italy: ACM, 2016, pp. 185–189. ISBN: 978-1-4503-4408-1. DOI: 10.1145/2935334.2935371 (cit. on p. 37).

[42] P. S. Crowther, R. J. Cox. “A method for optimal division of data sets for use in neural networks.” In: *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*. Springer. 2005, pp. 1–7 (cit. on p. 21).

[43] K. Curtis. “Inductive touch sensor design.” In: *Microchip Technology Inc., Brochure AN1239* 14 (2008) (cit. on p. 33).

[44] K. Dandekar, B. I. Raju, M. A. Srinivasan. “3-D finite-element models of human and monkey fingertips to investigate the mechanics of tactile sense.” In: *Journal of biomechanical engineering* 125.5 (2003), pp. 682–691 (cit. on p. 156).

[45] C. T. Dang, M. Straub, E. André. “Hand Distinction for Multi-touch Tabletop Interaction.” In: *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’09. Banff, Alberta, Canada: ACM, 2009, pp. 101–108. ISBN: 978-1-60558-733-2. DOI: 10.1145/1731903.1731925 (cit. on p. 47).

[46] A. De Luca, E. von Zezschwitz, N. D. H. Nguyen, M.-E. Maurer, E. Rubegni, M. P. Scipioni, M. Langheinrich. “Back-of-device Authentication on Smartphones.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’13. Paris, France: ACM, 2013, pp. 2389–2398. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2481330 (cit. on pp. 17, 47–49, 53).

[47] P. Dietz, D. Leigh. “DiamondTouch: A Multi-user Touch Technology.” In: *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*. UIST ’01. Orlando, Florida: ACM, 2001, pp. 219–226. ISBN: 1-58113-438-X. DOI: 10.1145/502348.502389 (cit. on pp. 45, 47).

[48] T. Dingler, P. El Agroudy, H. V. Le, A. Schmidt, E. Niforatos, A. Bexheti, M. Langheinrich. “Multimedia Memory Cues for Augmenting Human Memory.” In: *IEEE MultiMedia* 23.2 (2016), pp. 4–11. ISSN: 1070-986X. DOI: 10.1109/MMUL.2016.31 (cit. on pp. 26, 27).

[49] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, T. Darrell. “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description.” In: *IEEE Trans. Pattern Anal. Mach. Intell.* 39.4 (Apr. 2017), pp. 677–691. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2016.2599174 (cit. on p. 193).

[50] L. Du. “An Overview of Mobile Capacitive Touch Technologies Trends.” In: *arXiv preprint arXiv:1612.08227* (2016) (cit. on p. 33).

[51] J. Duchi, E. Hazan, Y. Singer. “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization.” In: *Journal of Machine Learning Research* 12 (July 2011), pp. 2121–2159. ISSN: 1532-4435 (cit. on pp. 105, 123, 148).

[52] R. Eardley, A. Roudaut, S. Gill, S. J. Thompson. “Investigating How Smartphone Movement is Affected by Body Posture.” In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI ’18. Montreal QC, Canada: ACM, 2018, 202:1–202:8. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3173776 (cit. on pp. 37, 38, 50, 56, 72, 127).

[53] R. Eardley, S. Gill, A. Roudaut, S. Thompson, J. Hare. “Investigating How the Hand Interacts with Different Mobile Phones.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. MobileHCI ’16. Florence, Italy: ACM, 2016, pp. 698–705. ISBN: 978-1-4503-4413-5. DOI: 10.1145/2957265.2961840 (cit. on pp. 37, 38, 56).

[54] R. Eardley, A. Roudaut, S. Gill, S. J. Thompson. “Understanding Grip Shifts: How Form Factors Impact Hand Movements on Mobile Phones.” In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI ’17. Denver, Colorado, USA: ACM, 2017, pp. 4680–4691. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025835 (cit. on pp. 35, 37, 38, 50, 54–56, 88, 127).

[55] F Ebeling, R Johnson, R Goldhor. *Infrared light beam xy position encoder for display devices*. US Patent 3,775,560. 1973 (cit. on p. 31).

[56] P. Ewerling, A. Kulik, B. Froehlich. “Finger and Hand Detection for Multi-touch Interfaces Based on Maximally Stable Extremal Regions.” In: *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 173–182. ISBN: 978-1-4503-1209-7. DOI: 10.1145/2396636.2396663 (cit. on pp. 46, 100).

[57] Y. Fan, X. Lu, D. Li, Y. Liu. “Video-based Emotion Recognition Using CNN-RNN and C3D Hybrid Networks.” In: *Proceedings of the 18th ACM International Conference on Multimodal Interaction*. ICMI 2016. Tokyo, Japan: ACM, 2016, pp. 445–450. ISBN: 978-1-4503-4556-9. DOI: 10.1145/2993148.2997632 (cit. on p. 193).

[58] A. M. Feit, D. Weir, A. Oulasvirta. “How We Type: Movement Strategies and Performance in Everyday Typing.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. San Jose, California, USA: ACM, 2016, pp. 4262–4273. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858233 (cit. on pp. 59, 145).

[59] S. Feng, G. Wilson, A. Ng, S. Brewster. “Investigating Pressure-based Interactions with Mobile Phones While Walking and Encumbered.” In: *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. MobileHCI ’15. Copenhagen, Denmark: ACM, 2015, pp. 854–861. ISBN: 978-1-4503-3653-6. DOI: 10.1145/2786567.2793711 (cit. on pp. 47, 48, 165).

[60] J. H. Friedman, J. L. Bentley, R. A. Finkel. “An Algorithm for Finding Best Matches in Logarithmic Expected Time.” In: *ACM Trans. Math. Softw.* 3.3 (Sept. 1977), pp. 209–226. ISSN: 0098-3500. DOI: 10.1145/355744.355745 (cit. on p. 150).

[61] V. Fuccella, P. Isokoski, B. Martin. “Gestures and Widgets: Performance in Text Editing on Multi-touch Capable Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’13. Paris, France: ACM, 2013, pp. 2785–2794. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2481385 (cit. on pp. 172, 173, 197).

[62] E. Ghomi, S. Huot, O. Bau, M. Beaudouin-Lafon, W. E. Mackay. “ArpèGe: Learning Multitouch Chord Gestures Vocabularies.” In: *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’13. St. Andrews, Scotland, United Kingdom: ACM, 2013, pp. 209–218. ISBN: 978-1-4503-2271-3. DOI: 10.1145/2512349.2512795 (cit. on pp. 46, 100).

[63] H. Gil, D. Lee, S. Im, I. Oakley. “TriTap: Identifying Finger Touches on Smartwatches.” In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI ’17. Denver, Colorado, USA: ACM, 2017, pp. 3879–3890. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025561 (cit. on pp. 16, 41, 51, 94, 95, 100, 122, 124–126, 137, 168).

[64] N. Gillian, J. A. Paradiso. “The Gesture Recognition Toolkit.” In: *J. Mach. Learn. Res.* 15.1 (Jan. 2014), pp. 3483–3487. ISSN: 1532-4435 (cit. on p. 153).

[65] X. Glorot, Y. Bengio. “Understanding the difficulty of training deep feedforward neural networks.” In: *In Proceedings of the International Conference on Artificial Intelligence and Statistics.* Vol. 9. AISTATS’10. Chia Laguna Resort, Sardinia, Italy: JMLR.org, 2010, pp. 249–256 (cit. on pp. 105, 123, 148).

[66] M. Goel, J. Wobbrock, S. Patel. “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. Cambridge, Massachusetts, USA: ACM, 2012, pp. 545–554. ISBN: 978-1-4503-1580-7. DOI: 10.1145/2380116.2380184 (cit. on pp. 44, 135).

[67] N.-W. Gong, J. Steimle, S. Olberding, S. Hodges, N. E. Gillian, Y. Kawahara, J. A. Paradiso. “PrintSense: A Versatile Sensing Technique to Support Multimodal Flexible Surface Interaction.” In: *Proceedings of the SIGCHI Conference*

on Human Factors in Computing Systems. CHI '14. Toronto, Ontario, Canada: ACM, 2014, pp. 1407–1410. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557173 (cit. on p. 49).

[68] N. Goyal. "COMET: Collaboration in Applications for Mobile Environments by Twisting." In: (2009) (cit. on p. 42).

[69] N. Goyal. "COMET: Collaboration in Mobile Environments by Twisting." In: *Supplementary Proceedings of the 11th European Conference on Computer Supported Cooperative Work*. 2009, p. 29 (cit. on p. 42).

[70] E. Granell, L. A. Leiva. "Less Is More: Efficient Back-of-Device Tap Input Detection Using Built-in Smartphone Sensors." In: *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces*. ISS '16. Niagara Falls, Ontario, Canada: ACM, 2016, pp. 5–11. ISBN: 978-1-4503-4248-3. DOI: 10.1145/2992154.2992166 (cit. on p. 49).

[71] T. Grossman, D. Wigdor, R. Balakrishnan. "Multi-finger Gestural Interaction with 3D Volumetric Displays." In: *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. UIST '04. Santa Fe, NM, USA: ACM, 2004, pp. 61–70. ISBN: 1-58113-957-8. DOI: 10.1145/1029632.1029644 (cit. on p. 44).

[72] H. P. R. Group et al. "NASA Task Load Index (TLX) v1. 0." In: *Paper and Pencil Package. NASA Ames Research Center, Moffett Field CA* (1988) (cit. on pp. 128, 131).

[73] A. Guo, R. Xiao, C. Harrison. "CapAuth: Identifying and Differentiating User Handprints on Commodity Capacitive Touchscreens." In: *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*. ITS '15. Madeira, Portugal: ACM, 2015, pp. 59–62. ISBN: 978-1-4503-3899-8. DOI: 10.1145/2817721.2817722 (cit. on pp. 33, 41).

[74] A. Gupta, M. Anwar, R. Balakrishnan. "Porous Interfaces for Small Screen Multitasking Using Finger Identification." In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. UIST '16. Tokyo, Japan: ACM, 2016, pp. 145–156. ISBN: 978-1-4503-4189-9. DOI: 10.1145/2984511.2984557 (cit. on pp. 16, 20, 45, 99, 100, 133, 168).

[75] A. Gupta, R. Balakrishnan. “DualKey: Miniature Screen Text Entry via Finger Identification.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. San Jose, California, USA: ACM, 2016, pp. 59–70. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858052 (cit. on pp. 16, 45, 99, 100, 168).

[76] C. Häger-Ross, M. H. Schieber. “Quantifying the independence of human finger movements: comparisons of digits, hands, and movement frequencies.” In: *The Journal of neuroscience* 20.22 (2000), pp. 8542–8550 (cit. on pp. 38, 50, 54, 56, 70, 91, 146, 213).

[77] J. Y. Han. “Low-cost Multi-touch Sensing Through Frustrated Total Internal Reflection.” In: *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*. UIST ’05. Seattle, WA, USA: ACM, 2005, pp. 115–118. ISBN: 1-59593-271-2. DOI: 10.1145/1095034.1095054 (cit. on pp. 33, 46, 157, 207).

[78] S. Han, H. Mao, W. J. Dally. “Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding.” In: *CoRR* abs/1510.00149 (2015) (cit. on pp. 107, 126, 153, 195).

[79] G. Harboe, E. M. Huang. “Real-World Affinity Diagramming Practices: Bridging the Paper-Digital Gap.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI ’15. Seoul, Republic of Korea: ACM, 2015, pp. 95–104. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702561 (cit. on pp. 131, 162, 204).

[80] C. Harrison, S. Hudson. “Using Shear As a Supplemental Two-dimensional Input Channel for Rich Touchscreen Interaction.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’12. Austin, Texas, USA: ACM, 2012, pp. 3149–3152. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208730 (cit. on pp. 42, 94).

[81] C. Harrison, M. Sato, I. Poupyrev. “Capacitive Fingerprinting: Exploring User Differentiation by Sensing Electrical Properties of the Human Body.” In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST ’12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 537–544. ISBN: 978-1-4503-1580-7. DOI: 10.1145/2380116.2380183 (cit. on p. 45).

[82] C. Harrison, J. Schwarz, S. E. Hudson. “TapSense: Enhancing Finger Interaction on Touch Surfaces.” In: *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*. UIST ’11. Santa Barbara, California, USA: ACM, 2011, pp. 627–636. ISBN: 978-1-4503-0716-1. DOI: 10.1145/2047196.2047279 (cit. on pp. 16, 42, 51, 94, 95, 104).

[83] C. Harrison, R. Xiao, J. Schwarz, S. E. Hudson. “TouchTools: Leveraging Familiarity and Skill with Physical Tools to Augment Touch Interaction.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’14. Toronto, Ontario, Canada: ACM, 2014, pp. 2913–2916. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557012 (cit. on p. 46).

[84] S. G. Hart. “NASA-task load index (NASA-TLX); 20 years later.” In: *Proceedings of the human factors and ergonomics society annual meeting*. Vol. 50. 9. Sage Publications Sage CA: Los Angeles, CA. 2006, pp. 904–908 (cit. on pp. 71, 74, 85, 128).

[85] B. Hartmann, M. R. Morris, H. Benko, A. D. Wilson. “Augmenting Interactive Tables with Mice & Keyboards.” In: *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology*. UIST ’09. Victoria, BC, Canada: ACM, 2009, pp. 149–152. ISBN: 978-1-60558-745-5. DOI: 10.1145/1622176.1622204 (cit. on p. 45).

[86] N. Hassan, M. M. Rahman, P. Irani, P. Graham. “Chucking: A one-handed document sharing technique.” In: *IFIP Conference on Human-Computer Interaction*. Springer. 2009, pp. 264–278 (cit. on p. 43).

[87] K. He, X. Zhang, S. Ren, J. Sun. “Deep residual learning for image recognition.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778 (cit. on p. 20).

[88] K. He, X. Zhang, S. Ren, J. Sun. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.” In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034 (cit. on p. 20).

[89] N. Henze, S. Mayer, H. V. Le, V. Schwind. “Improving Software-reduced Touchscreen Latency.” In: *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI ’17. Vienna, Austria: ACM, 2017, pp. 1071–1078. ISBN: 978-1-4503-5075-4. DOI: 10.1145/3098279.3122150 (cit. on pp. 26, 135).

[90] S. Heo, G. Lee. “Force Gestures: Augmented Touch Screen Gestures Using Normal and Tangential Force.” In: *CHI ’11 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’11. Vancouver, BC, Canada: ACM, 2011, pp. 1909–1914. ISBN: 978-1-4503-0268-5. DOI: 10.1145/1979742.1979895 (cit. on p. 42).

[91] S. Heo, G. Lee. “Forcetap: Extending the Input Vocabulary of Mobile Touch Screens by Adding Tap Gestures.” In: *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*. MobileHCI ’11. Stockholm, Sweden: ACM, 2011, pp. 113–122. ISBN: 978-1-4503-0541-9. DOI: 10.1145/2037373.2037393 (cit. on pp. 42, 94).

[92] K. Hinckley, H. Song. “Sensor Synaesthesia: Touch in Motion, and Motion in Touch.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: ACM, 2011, pp. 801–810. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979059 (cit. on p. 43).

[93] K. Hinckley, S. Heo, M. Pahud, C. Holz, H. Benko, A. Sellen, R. Banks, K. O’Hara, G. Smyth, W. Buxton. “Pre-Touch Sensing for Mobile Interaction.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. New York, NY, USA: ACM, 2016, pp. 2869–2881. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858095 (cit. on pp. 20, 44, 135).

[94] S. Hiraoka, I. Miyamoto, K. Tomimatsu. “Behind Touch, a Text Input Method for Mobile Phones by The Back and Tactile Sense Interface.” In: *Information Processing Society of Japan, Interaction 2003*. IPSJ ’03 (2003), pp. 131–138 (cit. on p. 53).

[95] D. Holman, A. Hollatz, A. Banerjee, R. Vertegaal. “Unifone: Designing for Auxiliary Finger Input in One-handed Mobile Interactions.” In: *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*. TEI ’13. Barcelona, Spain: ACM, 2013, pp. 177–184. ISBN: 978-1-4503-1898-3. DOI: 10.1145/2460625.2460653 (cit. on p. 48).

[96] C. Holz, P. Baudisch. “Fiberio: A Touchscreen That Senses Fingerprints.” In: *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*. UIST ’13. St. Andrews, Scotland, United Kingdom: ACM, 2013, pp. 41–50. ISBN: 978-1-4503-2268-3. DOI: 10.1145/2501988.2502021 (cit. on pp. 20, 46).

[97] C. Holz, P. Baudisch. “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. Atlanta, Georgia, USA: ACM, 2010, pp. 581–590. ISBN: 978-1-60558-929-9. DOI: 10.1145/1753326.1753413 (cit. on p. 89).

[98] C. Holz, P. Baudisch. “Understanding Touch.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: ACM, 2011, pp. 2501–2510. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979308 (cit. on pp. 40, 89, 156, 162).

[99] C. Holz, S. Buthpitiya, M. Knaust. “Bodyprint: Biometric User Identification on Mobile Devices Using the Capacitive Touchscreen to Scan Body Parts.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI ’15. Seoul, Republic of Korea: ACM, 2015, pp. 3011–3014. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702518 (cit. on pp. 33, 41, 87, 99, 100, 166).

[100] C. Holz, M. Knaust. “Biometric Touch Sensing: Seamlessly Augmenting Each Touch with Continuous Authentication.” In: *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. UIST ’15. Charlotte, NC, USA: ACM, 2015, pp. 303–312. ISBN: 978-1-4503-3779-3. DOI: 10.1145/2807442.2807458 (cit. on p. 45).

[101] C.-W. Hsu, C.-C. Chang, C.-J. Lin, et al. “A practical guide to support vector classification.” In: (2003) (cit. on pp. 21, 105, 123, 148, 194).

[102] ISO-9241-210. “ISO 9241-210:2010 - Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems.” In: *International Standardization Organization (ISO). Switzerland* (2010) (cit. on pp. 19, 22).

[103] S. Ioffe, C. Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.” In: *CoRR* abs/1502.03167 (2015) (cit. on pp. 105, 106, 148, 194).

[104] S. Jachner, G Van den Boogaart, T. Petzoldt, et al. “Statistical methods for the qualitative assessment of dynamic models with time delay (R Package qualV).” In: *Journal of Statistical Software* 22.8 (2007), pp. 1–30 (cit. on p. 149).

[105] E. A. Johnson. “Touch display – a novel input/output device for computers.” In: *Electronics Letters* 1.8 (1965), pp. 219–220. ISSN: 0013-5194. DOI: 10.1049/el:19650200 (cit. on pp. 30, 53).

[106] E. A. Johnson. “Touch displays: a programmed man-machine interface.” In: *Ergonomics* 10.2 (1967), pp. 271–277 (cit. on p. 30).

[107] L. A. Jones, S. J. Lederman. *Human hand function*. Oxford University Press, 2006 (cit. on p. 116).

[108] M. Kaltenbrunner, R. Bencina. “reacTIVision: A Computer-vision Framework for Table-based Tangible Interaction.” In: *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*. TEI ’07. Baton Rouge, Louisiana: ACM, 2007, pp. 69–74. ISBN: 978-1-59593-619-6. DOI: 10.1145/1226969.1226983 (cit. on p. 45).

[109] A. K. Karlson, B. B. Bederson. “Studies in One-Handed Mobile Design: Habit, Desire and Agility.” In: *Proceedings of the 4th ERCIM Workshop on User Interfaces for All*. UI4ALL. 2006 (cit. on pp. 16, 35, 54, 55).

[110] A. K. Karlson, B. B. Bederson, J. L. Contreras-Vidal. “Understanding one-handed use of mobile devices.” In: *Handbook of research on user interface design and evaluation for mobile technology*. IGI Global, 2008, pp. 86–101 (cit. on pp. 16, 35, 54, 55).

[111] F. Kerber, P. Lessel, A. Krüger. “Same-side Hand Interactions with Arm-placed Devices Using EMG.” In: *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’15. Seoul, Republic of Korea: ACM, 2015, pp. 1367–1372. ISBN: 978-1-4503-3146-3. DOI: 10.1145/2702613.2732895 (cit. on p. 96).

[112] S. Kim, J. Yu, G. Lee. “Interaction Techniques for Unreachable Objects on the Touchscreen.” In: *Proceedings of the 24th Australian Computer-Human Interaction Conference*. OzCHI ’12. Melbourne, Australia: ACM, 2012, pp. 295–298. ISBN: 978-1-4503-1438-1. DOI: 10.1145/2414536.2414585 (cit. on pp. 40, 55).

[113] V. Kostakos, M. Musolesi. “Avoiding Pitfalls when Using Machine Learning in HCI Studies.” In: *interactions* 24.4 (June 2017), pp. 34–37. ISSN: 1072-5520. DOI: 10.1145/3085556 (cit. on pp. 21, 126, 134, 157).

[114] S. Koura, S. Suo, A. Kimura, F. Shibata, H. Tamura. “Amazing Forearm As an Innovative Interaction Device and Data Storage on Tabletop Display.” In: *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 383–386. ISBN: 978-1-4503-1209-7. DOI: 10.1145/2396636.2396706 (cit. on p. 46).

[115] A. Krizhevsky, I. Sutskever, G. E. Hinton. “Imagenet classification with deep convolutional neural networks.” In: *Advances in neural information processing systems*. 2012, pp. 1097–1105 (cit. on p. 105).

[116] M. W. Krueger, T. Gionfriddo, K. Hinrichsen. “VIDEOPLACE – an Artificial Reality.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’85. San Francisco, California, USA: ACM, 1985, pp. 35–40. ISBN: 0-89791-149-0. DOI: 10.1145/317456.317463 (cit. on p. 31).

[117] A. Kumar, A. Radjesh, S. Mayer, H. V. Le. “Improving the Input Accuracy of Touchscreens using Deep Learning.” In: *Proceedings of the 2019 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’19. Glasgow, Scotland, UK: ACM, 2019. ISBN: 978-1-4503-5971-9. DOI: 10.1145/3290607.3312928 (cit. on p. 26).

[118] L.-C. Kuo, H.-Y. Chiu, C.-W. Chang, H.-Y. Hsu, Y.-N. Sun. “Functional workspace for precision manipulation between thumb and fingers in normal hands.” In: *Journal of Electromyography and Kinesiology* 19.5 (2009), pp. 829 –839. DOI: http://dx.doi.org/10.1016/j.jelekin.2008.07.008 (cit. on pp. 36, 37).

[119] B. Lahey, A. Girouard, W. Burleson, R. Vertegaal. “PaperPhone: Understanding the Use of Bend Gestures in Mobile Devices with Flexible Electronic Paper Displays.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: ACM, 2011, pp. 1303–1312. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979136 (cit. on p. 42).

[120] L. Lam, S. Y. Suen. “Application of majority voting to pattern recognition: an analysis of its behavior and performance.” In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 27.5 (1997), pp. 553–568. ISSN: 1083-4427. DOI: 10.1109/3468.618255 (cit. on p. 134).

[121] H. V. Le. “Fully Hand-and-Finger-Aware Smartphone Interaction.” In: *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI EA ’18. Montreal QC, Canada: ACM, 2018, DC13:1–DC13:4. ISBN: 978-1-4503-5621-3. DOI: 10.1145/3170427.3173023 (cit. on p. 26).

[122] H. V. Le. “Modeling Human Behavior During Touchscreen Interaction in Mobile Situations.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. MobileHCI ’16. Florence, Italy: ACM, 2016, pp. 901–902. ISBN: 978-1-4503-4413-5. DOI: 10.1145/2957265.2963113 (cit. on p. 26).

[123] H. V. Le, S. Mayer, N. Henze. "InfiniTouch: Finger-Aware Interaction on Fully Touch Sensitive Smartphones." In: *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. UIST '18. Berlin, Germany: ACM, 2018. ISBN: 978-1-4503-5948-1. DOI: 10.1145/3242587.3242605 (cit. on pp. 26, 91, 123, 139).

[124] H. V. Le, S. Mayer, N. Henze. "Investigating Unintended Inputs for One-Handed Touch Interaction Beyond the Touchscreen." In: *Currently under review.* (Cit. on pp. 26, 54).

[125] H. V. Le, S. Mayer, N. Henze. "Investigating the Feasibility of Finger Identification on Capacitive Touchscreens using Deep Learning." In: *24th International Conference on Intelligent User Interfaces*. IUI '19. Marina del Ray, CA, USA: ACM, 2019. DOI: 10.1145/3301275.3302295 (cit. on pp. 26, 94).

[126] H. V. Le, S. Mayer, N. Henze. "Machine Learning with Tensorflow for Mobile and Ubiquitous Interaction." In: *Proceedings of the 16th International Conference on Mobile and Ubiquitous Multimedia*. MUM '17. Stuttgart, Germany: ACM, 2017, pp. 567–572. ISBN: 978-1-4503-5378-6. DOI: 10.1145/3152832.3156559 (cit. on p. 26).

[127] H. V. Le, S. Mayer, P. Bader, N. Henze. "A Smartphone Prototype for Touch Interaction on the Whole Device Surface." In: *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI EA '17. Vienna, Austria: ACM, 2017, pp. 1001–1008. ISBN: 978-1-4503-5075-4. DOI: 10.1145/3098279.3122143 (cit. on p. 48).

[128] H. V. Le, S. Mayer, T. Kosch, N. Henze. "Demonstrating PalmTouch: The Palm as An Additional Input Modality on Commodity Smartphones." In: *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. MobileHCI '18. Barcelona, Spain: ACM, 2018. ISBN: 978-1-4503-5941-2. DOI: 10.1145/3236112.3236163 (cit. on p. 26).

[129] H. V. Le, S. Mayer, K. Wolf, N. Henze. "Finger Placement and Hand Grasp During Smartphone Interaction." In: *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA '16. Santa Clara, California, USA: ACM, 2016, pp. 2576–2584. ISBN: 978-1-4503-4082-3. DOI: 10.1145/2851581.2892462 (cit. on pp. 27, 127, 142, 159).

[130] H. V. Le, S. Mayer, P. Bader, N. Henze. “Fingers’ Range and Comfortable Area for One-Handed Smartphone Interaction Beyond the Touchscreen.” In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI’18. New York, NY, USA: ACM, 2018. DOI: 10.1145/3173574.3173605 (cit. on pp. 26, 54, 91, 98, 142, 145, 155, 159).

[131] H. V. Le, S. Clinch, C. Sas, T. Dingler, N. Henze, N. Davies. “Impact of Video Summary Viewing on Episodic Memory Recall: Design Guidelines for Video Summarizations.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. San Jose, California, USA: ACM, 2016, pp. 4793–4805. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858413 (cit. on pp. 26, 27).

[132] H. V. Le, S. Mayer, P. Bader, F. Bastian, N. Henze. “Interaction Methods and Use Cases for a Full-Touch Sensing Smartphone.” In: *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’17. Denver, Colorado, USA: ACM, 2017. ISBN: 978-1-4503-4656-6. DOI: 10.1145/3027063.3053196 (cit. on pp. 26, 48, 140).

[133] H. V. Le, P. Bader, T. Kosch, N. Henze. “Investigating Screen Shifting Techniques to Improve One-Handed Smartphone Usage.” In: *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*. NordiCHI ’16. Gothenburg, Sweden: ACM, 2016, 27:1–27:10. ISBN: 978-1-4503-4763-1. DOI: 10.1145/2971485.2971562 (cit. on pp. 17, 26, 48, 49, 53, 55, 69, 73, 140, 153, 156, 165).

[134] H. V. Le, S. Mayer, A. E. Ali, N. Henze. “Machine Learning For Intelligent Mobile User Interfaces using Keras.” In: *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. MobileHCI’18. New York, NY, USA: ACM, Sept. 3, 2018. Forthcoming (cit. on p. 26).

[135] H. V. Le, S. Mayer, J. Vogelsang, H. Weingärtner, M. Weiß, N. Henze. “On-Device Touch Gestures for Text Editing on Mobile Devices.” In: *Under Review at the ACM Transactions on Computer-Human Interaction*. TOCHI. (Cit. on p. 171).

[136] H. V. Le, T. Kosch, P. Bader, S. Mayer, N. Henze. “PalmTouch: Using the Palm as an Additional Input Modality on Commodity Smartphones.” In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI ’18. New York, NY, USA: ACM, 2018. DOI: 10.1145/3173574.3173934 (cit. on pp. 26, 33, 41, 94, 100, 123).

[137] H. V. Le, V. Schwind, P. Göttlich, N. Henze. “PredicTouch: A System to Reduce Touchscreen Latency using Neural Networks and Inertial Measurement Units.” In: *Proceedings of the 2017 International Conference on Interactive Surfaces and Spaces*. Vol. 17. ISS’17. New York, NY, USA: ACM, Oct. 17, 2017. DOI: https://doi.org/10.1145/3132272.3134138 (cit. on p. 26).

[138] S. Lee, W. Buxton, K. C. Smith. “A Multi-touch Three Dimensional Touch-sensitive Tablet.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’85. San Francisco, California, USA: ACM, 1985, pp. 21–25. ISBN: 0-89791-149-0. DOI: 10.1145/317456.317461 (cit. on p. 31).

[139] S. Lee, G. Kyung, J. Lee, S. K. Moon, K. J. Park. “Grasp and index finger reach zone during one-handed smartphone rear interaction: effects of task type, phone width and hand length.” In: *Ergonomics* 59.11 (2016), pp. 1462–1472. DOI: 10.1080/00140139.2016.1146346 (cit. on pp. 36, 145).

[140] L. A. Leiva, A. Català. “BoD Taps: An Improved Back-of-device Authentication Technique on Smartphones.” In: *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*. MobileHCI ’14. Toronto, ON, Canada: ACM, 2014, pp. 63–66. ISBN: 978-1-4503-3004-6. DOI: 10.1145/2628363.2628372 (cit. on p. 48).

[141] Y. Li. “Gesture Search: A Tool for Fast Mobile Data Access.” In: *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*. UIST ’10. New York, New York, USA: ACM, 2010, pp. 87–96. ISBN: 978-1-4503-0271-5. DOI: 10.1145/1866029.1866044 (cit. on p. 40).

[142] G. Lindner. “Sensors and actuators based on surface acoustic waves propagating along solid–liquid interfaces.” In: *Journal of Physics D: Applied Physics* 41.12 (2008), p. 123002 (cit. on p. 33).

[143] M. Löchtefeld, C. Hirtz, S. Gehring. “Evaluation of Hybrid Front- and Back-of-device Interaction on Mobile Devices.” In: *Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia*. MUM ’13. Luleå, Sweden: ACM, 2013, 17:1–17:4. ISBN: 978-1-4503-2648-3. DOI: 10.1145/2541831.2541865 (cit. on p. 37).

[144] M. Löchtefeld, P. Schardt, A. Krüger, S. Boring. “Detecting Users Handedness for Ergonomic Adaptation of Mobile User Interfaces.” In: *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia*. MUM ’15. Linz, Austria: ACM, 2015, pp. 245–249. ISBN: 978-1-4503-3605-5. DOI: 10.1145/2836041.2836066 (cit. on pp. 100, 135).

[145] P. Lopes, R. Jota, J. A. Jorge. “Augmenting Touch Interaction Through Acoustic Sensing.” In: *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’11. Kobe, Japan: ACM, 2011, pp. 53–56. ISBN: 978-1-4503-0871-7. DOI: 10.1145/2076354.2076364 (cit. on p. 43).

[146] I. S. MacKenzie, R. W. Soukoreff. “Phrase Sets for Evaluating Text Entry Techniques.” In: *CHI ’03 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’03. Ft. Lauderdale, Florida, USA: ACM, 2003, pp. 754–755. ISBN: 1-58113-637-4. DOI: 10.1145/765891.765971 (cit. on p. 74).

[147] A. Mandelbaum, D. Weinshall. “Distance-based Confidence Score for Neural Network Classifiers.” In: *arXiv preprint arXiv:1709.09844* (2017) (cit. on p. 134).

[148] N. Marquardt, J. Kiemer, S. Greenberg. “What Caused That Touch?: Expressive Interaction with a Surface Through Fiduciary-tagged Gloves.” In: *ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’10. Saarbrücken, Germany: ACM, 2010, pp. 139–142. ISBN: 978-1-4503-0399-6. DOI: 10.1145/1936652.1936680 (cit. on p. 47).

[149] N. Marquardt, J. Kiemer, D. Ledo, S. Boring, S. Greenberg. “Designing User-, Hand-, and Handpart-aware Tabletop Interactions with the TouchID Toolkit.” In: *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’11. Kobe, Japan: ACM, 2011, pp. 21–30. ISBN: 978-1-4503-0871-7. DOI: 10.1145/2076354.2076358 (cit. on pp. 45, 100).

[150] N. Marquardt, R. Jota, S. Greenberg, J. A. Jorge. “The continuous interaction space: interaction techniques unifying touch and gesture on and above a digital surface.” In: *IFIP Conference on Human-Computer Interaction*. Springer. 2011, pp. 461–476 (cit. on p. 44).

[151] A. Martonik. *How to use the Heart Rate Monitor on the Galaxy S5 (Androidcentral.com)*. Last access: 2017-09-09. 2015. URL: http://www.androidcentral.com/how-use-heart-rate-monitor-galaxy-s5 (cit. on p. 47).

[152] D. Masson, A. Goguey, S. Malacria, G. Casiez. “WhichFingers: Identifying Fingers on Touch Surfaces and Keyboards Using Vibration Sensors.” In: *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. UIST ’17. Québec City, QC, Canada: ACM, 2017, pp. 41–48. ISBN: 978-1-4503-4981-9. DOI: 10.1145/3126594.3126619 (cit. on pp. 16, 20, 45, 51, 100, 168).

[153] J. Matero, A. Colley. “Identifying Unintentional Touches on Handheld Touch Screen Devices.” In: *Proceedings of the Designing Interactive Systems Conference*. DIS ’12. Newcastle Upon Tyne, United Kingdom: ACM, 2012, pp. 506–509. ISBN: 978-1-4503-1210-3. DOI: 10.1145/2317956.2318031 (cit. on pp. 92, 95, 101).

[154] F. Matulic, D. Vogel, R. Dachselt. “Hand Contact Shape Recognition for Posture-Based Tabletop Widgets and Interaction.” In: *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces*. ISS ’17. Brighton, United Kingdom: ACM, 2017, pp. 3–11. ISBN: 978-1-4503-4691-7. DOI: 10.1145/3132272.3134126 (cit. on p. 46).

[155] S. Mayer, H. V. Le, N. Henze. “Designing Finger Orientation Input for Mobile Touchscreens.” In: *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI’18. New York, NY, USA: ACM, 2018. DOI: 10.1145/3229434.3229444 (cit. on p. 26).

[156] S. Mayer, H. V. Le, N. Henze. “Estimating the Finger Orientation on Capacitive Touchscreens Using Convolutional Neural Networks.” In: *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces*. ISS ’17. Brighton, United Kingdom: ACM, 2017, pp. 220–229. ISBN: 978-1-4503-4691-7. DOI: 10.1145/3132272.3134130 (cit. on pp. 16, 26, 33, 41, 94, 100).

[157] S. Mayer, H. V. Le, N. Henze. “Machine Learning for Intelligent Mobile User Interfaces Using TensorFlow.” In: *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI ’17. Vienna, Austria: ACM, 2017, pp. 621–625. ISBN: 978-1-4503-5075-4. DOI: 10.1145/3098279.3119915 (cit. on p. 26).

[158] S. Mayer, V. Schwind, H. V. Le, D. Weber, J. Vogelsang, J. Wolf, N. Henze. “Effect of Orientation on Unistroke Touch Gestures.” In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI’19. New York, NY, USA: ACM, 2019. DOI: 10.1145/3290605.3300928 (cit. on p. 26).

[159] S. Mayer, L. Lischke, A. Lanksweirt, H. V. Le, N. Henze. “How to Communicate New Input Techniques.” In: *Proceedings of the 10th Nordic Conference on Human-Computer Interaction*. NordiCHI ’18. New York, NY, USA: ACM, 2018, p. 13. DOI: 10.1145/3240167.3240176 (cit. on pp. 26, 158).

[160] S. Mayer, H. V. Le, A. Nesti, N. Henze, H. H. Bülthoff, L. L. Chuang. “The Effect of Road Bumps on Touch Interaction in Cars.” In: *10th International Conference on Automotive User Interfaces and Interactive Vehicular Applications Proceedings*. AutomotiveUI ’18. 2018. DOI: 10.1145/3239060.3239071 (cit. on pp. 26, 27).

[161] W. McGrath, Y. Li. “Detecting Tapping Motion on the Side of Mobile Devices by Probabilistically Combining Hand Postures.” In: *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*. UIST ’14. Honolulu, Hawaii, USA: ACM, 2014, pp. 215–219. ISBN: 978-1-4503-3069-5. DOI: 10.1145/2642918.2647363 (cit. on pp. 47, 49).

[162] N. Mehta. “A flexible machine interface.” In: *MA Sc. Thesis, Department of Electrical Engineering, University of Toronto supervised by Professor KC Smith* (1982) (cit. on p. 31).

[163] T. Meyer, D. Schmidt. “IdWristbands: IR-based User Identification on Multi-touch Surfaces.” In: *ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’10. Saarbrücken, Germany: ACM, 2010, pp. 277–278. ISBN: 978-1-4503-0399-6. DOI: 10.1145/1936652.1936714 (cit. on p. 47).

[164] Microsoft Corporation. *Google Play: Microsoft Excel*. https://play.google.com/store/apps/details?id=com.microsoft.office.excel&hl=en. 2018 (cit. on p. 172).

[165] Microsoft Corporation. *Google Play: Microsoft PowerPoint*. https://play.google.com/store/apps/details?id=com.microsoft.office.powerpoint&hl=en. 2018 (cit. on p. 172).

[166] Microsoft Corporation. *Google Play: Microsoft Word*. https://play.google.com/store/apps/details?id=com.microsoft.office.word&hl=en. 2018 (cit. on p. 172).

[167] T. Miyaki, J. Rekimoto. “GraspZoom: Zooming and Scrolling Control Model for Single-handed Mobile Interaction.” In: *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI ’09. Bonn, Germany: ACM, 2009, 11:1–11:4. ISBN: 978-1-60558-281-8. DOI: 10.1145/1613858.1613872 (cit. on p. 42).

[168] M. F. Mohd Noor, S. Rogers, J. Williamson. “Detecting Swipe Errors on Touchscreens Using Grip Modulation.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. Santa Clara, California, USA: ACM, 2016, pp. 1909–1920. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858474 (cit. on pp. 48, 49, 53, 157).

[169] M. F. Mohd Noor, A. Ramsay, S. Hughes, S. Rogers, J. Williamson, R. Murray-Smith. "28 Frames Later: Predicting Screen Touches from Back-of-device Grip Changes." In: *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems*. CHI '14. Toronto, Ontario, Canada: ACM, 2014, pp. 2005–2008. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557148 (cit. on pp. 49, 53, 157).

[170] M. R. Morris, J. O. Wobbrock, A. D. Wilson. "Understanding Users' Preferences for Surface Gestures." In: *Proceedings of Graphics Interface 2010*. GI '10. Ottawa, Ontario, Canada: Canadian Information Processing Society, 2010, pp. 261–268. ISBN: 978-1-56881-712-5 (cit. on p. 181).

[171] R. Murray-Smith. "Stratified, computational interaction via machine learning." In: *Eighteenth Yale Workshop on Adaptive and Learning Systems (New Haven, CT, USA*. 2017, pp. 95–101 (cit. on p. 100).

[172] S. Murugappan, Vinayak, N. Elmqvist, K. Ramani. "Extended Multitouch: Recovering Touch Posture and Differentiating Users Using a Depth Camera." In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST '12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 487–496. ISBN: 978-1-4503-1580-7. DOI: 10.1145/2380116.2380177 (cit. on pp. 46, 100).

[173] M. A. Nacenta, Y. Kamber, Y. Qiang, P. O. Kristensson. "Memorability of Pre-designed and User-defined Gesture Sets." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 1099–1108. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2466142 (cit. on p. 40).

[174] R. Nandakumar, V. Iyer, D. Tan, S. Gollakota. "FingerIO: Using Active Sonar for Fine-Grained Finger Tracking." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. Santa Clara, California, USA: ACM, 2016, pp. 1515–1525. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858580 (cit. on p. 49).

[175] J. R. Napier. "The prehensile movements of the human hand." In: *Bone & Joint Journal* 38.4 (1956), pp. 902–913 (cit. on pp. 37, 39, 88).

[176] K. Nelavelli, T. Ploetz. "Adaptive App Design by Detecting Handedness." In: *arXiv preprint arXiv:1805.08367* (2018) (cit. on pp. 16, 35, 54, 55).

[177] A. Ng, S. A. Brewster, J. H. Williamson. “Investigating the Effects of Encumbrance on One- and Two- Handed Interactions with Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’14. Toronto, Ontario, Canada: ACM, 2014, pp. 1981–1990. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557312 (cit. on pp. 163, 167).

[178] A. Ng, J. Williamson, S. Brewster. “The Effects of Encumbrance and Mobility on Touch-Based Gesture Interactions for Mobile Phones.” In: *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI ’15. Copenhagen, Denmark: ACM, 2015, pp. 536–546. ISBN: 978-1-4503-3652-9. DOI: 10.1145/2785830.2785853 (cit. on pp. 16, 72).

[179] H. Nicolau, J. Jorge. “Touch Typing Using Thumbs: Understanding the Effect of Mobility and Hand Posture.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’12. Austin, Texas, USA: ACM, 2012, pp. 2683–2686. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208661 (cit. on p. 89).

[180] A. S. Nittala, A. Withana, N. Pourjafarian, J. Steimle. “Multi-Touch Skin: A Thin and Flexible Multi-Touch Sensor for On-Skin Input.” In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI ’18. Montreal QC, Canada: ACM, 2018, 33:1–33:12. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3173607 (cit. on p. 215).

[181] D. Norman. *The design of everyday things: Revised and expanded edition*. Constellation, 2013 (cit. on p. 216).

[182] I. Oakley, C. Lindahl, K. Le, D. Lee, M. R. Islam. “The Flat Finger: Exploring Area Touches on Smartwatches.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. Santa Clara, California, USA: ACM, 2016, pp. 4238–4249. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858179 (cit. on pp. 16, 41).

[183] A. Olwal, A. D. Wilson. “SurfaceFusion: Unobtrusive Tracking of Everyday Objects in Tangible User Interfaces.” In: *Proceedings of Graphics Interface 2008*. GI ’08. Windsor, Ontario, Canada: Canadian Information Processing Society, 2008, pp. 235–242. ISBN: 978-1-56881-423-0 (cit. on p. 45).

[184] N. Orr, V. Hopkin. *The role of the touch display in air traffic control*. Tech. rep. RAF INST OF AVIATION MEDICINE FARNBOROUGH (ENGLAND), 1968 (cit. on p. 30).

[185] A. Ostberg, M. Sheik-Nainar, N. Matic. “Using a Mobile Device Fingerprint Sensor As a Gestural Input Device.” In: *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’16. Santa Clara, California, USA: ACM, 2016, pp. 2625–2631. ISBN: 978-1-4503-4082-3. DOI: 10.1145/2851581.2892419 (cit. on pp. 48, 55).

[186] A. Oulasvirta, P. O. Kristensson, X. Bi, A. Howes. *Computational Interaction*. Oxford University Press, 2018. ISBN: 978-0-1987-9961-0 (cit. on p. 216).

[187] A. Oulasvirta, A. Reichel, W. Li, Y. Zhang, M. Bachynskyi, K. Vertanen, P. O. Kristensson. “Improving Two-thumb Text Entry on Touchscreen Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’13. New York, NY, USA: ACM, 2013, pp. 2765–2774. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2481383 (cit. on p. 127).

[188] J. A. Paradiso, C. K. Leo, N. Checka, K. Hsiao. “Passive Acoustic Knock Tracking for Interactive Windows.” In: *CHI ’02 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’02. Minneapolis, Minnesota, USA: ACM, 2002, pp. 732–733. ISBN: 1-58113-454-1. DOI: 10.1145/506443.506570 (cit. on p. 43).

[189] Y. S. Park, S. H. Han. “One-handed thumb interaction of mobile devices from the input accuracy perspective.” In: *International Journal of Industrial Ergonomics* 40.6 (2010), pp. 746 –756. ISSN: 0169-8141. DOI: http://dx.doi.org/10.1016/j.ergon.2010.08.001 (cit. on p. 36).

[190] B. Poppinga, A. Sahami Shirazi, N. Henze, W. Heuten, S. Boll. “Understanding Shortcut Gestures on Mobile Touch Devices.” In: *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*. MobileHCI ’14. Toronto, ON, Canada: ACM, 2014, pp. 173–182. ISBN: 978-1-4503-3004-6. DOI: 10.1145/2628363.2628378 (cit. on p. 40).

[191] A Poston. “Human engineering design data digest.” In: *Washington, DC: Department of Defense Human Factors Engineering Technical Advisory Group* (2000) (cit. on pp. 61, 75, 102, 119, 146, 191).

[192] E Quinn, I. Nation, S. Millett. “Asian and Pacific speed readings for ESL learners.” In: *ELI Occasional Publication* 24 (2007) (cit. on p. 74).

[193] R. Ramakers, D. Vanacken, K. Luyten, K. Coninx, J. Schöning. “Carpus: A Non-intrusive User Identification Technique for Interactive Surfaces.” In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST ’12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 35–44. ISBN: 978-1-4503-1580-7. DOI: 10.1145/2380116.2380123 (cit. on p. 47).

[194] J. Rekimoto. “SmartSkin: An Infrastructure for Freehand Manipulation on Interactive Surfaces.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’02. Minneapolis, Minnesota, USA: ACM, 2002, pp. 113–120. ISBN: 1-58113-453-3. DOI: 10.1145/503376.503397 (cit. on pp. 32, 44, 45).

[195] J. Rekimoto, M. Saitoh. “Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’99. Pittsburgh, Pennsylvania, USA: ACM, 1999, pp. 378–385. ISBN: 0-201-48559-1. DOI: 10.1145/302979.303113 (cit. on p. 45).

[196] S. Richter, C. Holz, P. Baudisch. “Bootstrapper: Recognizing Tabletop Users by Their Shoes.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’12. Austin, Texas, USA: ACM, 2012, pp. 1249–1252. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208577 (cit. on p. 47).

[197] S. Robinson, N. Rajput, M. Jones, A. Jain, S. Sahay, A. Nanavati. “TapBack: Towards Richer Mobile Interfaces in Impoverished Contexts.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: ACM, 2011, pp. 2733–2736. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979345 (cit. on pp. 17, 47, 48, 53, 55, 153).

[198] S. Rogers, J. Williamson, C. Stewart, R. Murray-Smith. “AnglePose: Robust, Precise Capacitive Touch Tracking via 3D Orientation Estimation.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: ACM, 2011, pp. 2575–2584. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979318 (cit. on pp. 16, 44, 94).

[199] V. Roth, P. Schmidt, B. Güldenring. “The IR Ring: Authenticating Users’ Touches on a Multi-touch Display.” In: *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*. UIST ’10. New York, New York, USA: ACM, 2010, pp. 259–262. ISBN: 978-1-4503-0271-5. DOI: 10.1145/1866029.1866071 (cit. on p. 47).

[200] V. Roth, T. Turner. “Bezel Swipe: Conflict-free Scrolling and Multiple Selection on Mobile Touch Screen Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’09. Boston, MA, USA: ACM, 2009, pp. 1523–1526. ISBN: 978-1-60558-246-7. DOI: 10.1145/1518701.1518933 (cit. on p. 40).

[201] A. Roudaut, M. Baglioni, E. Lecolinet. “TimeTilt: using sensor-based gestures to travel through multiple applications on a mobile device.” In: *IFIP Conference on Human-Computer Interaction*. Springer. 2009, pp. 830–834 (cit. on p. 49).

[202] A. Roudaut, E. Lecolinet, Y. Guiard. “MicroRolls: Expanding Touch-screen Input Vocabulary by Distinguishing Rolls vs. Slides of the Thumb.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’09. Boston, MA, USA: ACM, 2009, pp. 927–936. ISBN: 978-1-60558-246-7. DOI: 10.1145/1518701.1518843 (cit. on pp. 20, 40, 94).

[203] W. Ruan, Q. Z. Sheng, L. Yang, T. Gu, P. Xu, L. Shangguan. “AudioGest: Enabling Fine-grained Hand Gesture Detection by Decoding Echo Signal.” In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. UbiComp ’16. Heidelberg, Germany: ACM, 2016, pp. 474–485. ISBN: 978-1-4503-4461-6. DOI: 10.1145/2971648.2971736 (cit. on p. 49).

[204] J. Ruiz, Y. Li, E. Lank. “User-defined Motion Gestures for Mobile Interaction.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: ACM, 2011, pp. 197–206. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1978971 (cit. on pp. 182, 189).

[205] J. L. Sancho-Bru, A Perez-Gonzalez, M Vergara, D. Giurintano. “A 3D biomechanical model of the hand for power grip.” In: *Journal of biomechanical engineering* 125.1 (2003), pp. 78–83 (cit. on pp. 37, 39).

[206] M. Sato, I. Poupyrev, C. Harrison. “Touché: Enhancing Touch Interaction on Humans, Screens, Liquids, and Everyday Objects.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’12. Austin, Texas, USA: ACM, 2012, pp. 483–492. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2207743 (cit. on p. 45).

[207] M. H. Schieber, M. Chua, J. Petit, C. C. Hunt. “Tension distribution of single motor units in multitendoned muscles: comparison of a homologous digit muscle in cats and monkeys.” In: *Journal of Neuroscience* 17.5 (1997), pp. 1734–1747 (cit. on p. 38).

[208] D. Schmidt, M. K. Chong, H. Gellersen. “HandsDown: Hand-contour-based User Identification for Interactive Surfaces.” In: *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*. NordiCHI ’10. Reykjavik, Iceland: ACM, 2010, pp. 432–441. ISBN: 978-1-60558-934-3. DOI: 10.1145/1868914.1868964 (cit. on p. 47).

[209] B. Schneiderman. “Eight golden rules of interface design.” In: *Disponible en* (1986) (cit. on p. 16).

[210] B. Schneiderman. “Eight golden rules of interface design.” In: *Disponible en* (1986) (cit. on p. 172).

[211] J. Schwarz, R. Xiao, J. Mankoff, S. E. Hudson, C. Harrison. “Probabilistic palm rejection using spatiotemporal touch features and iterative classification.” In: *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*. CHI ’14. Toronto, Ontario, Canada: ACM, 2014, pp. 2009–2012. ISBN: 978-1-4503-2473-1. DOI: 10.1145/2556288.2557056 (cit. on pp. 92, 95, 101).

[212] K. Seipp, K. Devlin. “BackPat: One-handed Off-screen Patting Gestures.” In: *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*. MobileHCI ’14. Toronto, ON, Canada: ACM, 2014, pp. 77–80. ISBN: 978-1-4503-3004-6. DOI: 10.1145/2628363.2628396 (cit. on p. 55).

[213] K. Seipp, K. Devlin. “One-Touch Pose Detection on Touchscreen Smartphones.” In: *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*. ITS ’15. Madeira, Portugal: ACM, 2015, pp. 51–54. ISBN: 978-1-4503-3899-8. DOI: 10.1145/2817721.2817739 (cit. on pp. 44, 135).

[214] E.-l. E. Shen, S.-s. D. Tsai, H.-h. Chu, Y.-j. J. Hsu, C.-w. E. Chen. “Double-side Multi-touch Input for Mobile Devices.” In: *CHI ’09 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’09. Boston, MA, USA: ACM, 2009, pp. 4339–4344. ISBN: 978-1-60558-247-4. DOI: 10.1145/1520340.1520663 (cit. on pp. 48, 49).

[215] S. S. A. Shimon, S. Morrison-Smith, N. John, G. Fahimi, J. Ruiz. “Exploring User-Defined Back-Of-Device Gestures for Mobile Devices.” In: *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI ’15. Copenhagen, Denmark: ACM, 2015, pp. 227–232. ISBN: 978-1-4503-3652-9. DOI: 10.1145/2785830.2785890 (cit. on pp. 48, 55, 69, 153).

[216] B. Shneiderman. *Designing the user interface: strategies for effective human-computer interaction*. Pearson Education India, 2010 (cit. on p. 216).

[217] K. A. Siek, Y. Rogers, K. H. Connelly. "Fat Finger Worries: How Older and Younger Users Physically Interact with PDAs." In: *Proceedings of the 2005 IFIP TC13 International Conference on Human-Computer Interaction*. INTERACT '05. Rome, Italy: Springer Berlin Heidelberg, 2005, pp. 267–280. ISBN: 978-3-540-31722-7. DOI: 10.1007/11555261_24 (cit. on pp. 17, 51, 172).

[218] K. Simonyan, A. Zisserman. "Very deep convolutional networks for large-scale image recognition." In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on p. 20).

[219] Smith Aaron. *A "Week in the Life" Analysis of Smartphone Users*. http://www.pewinternet.org/2015/04/01/chapter-three-a-week-in-the-life-analysis\-of-smartphone-users/. 2015 (cit. on p. 172).

[220] J. Song, G. Sörös, F. Pece, S. R. Fanello, S. Izadi, C. Keskin, O. Hilliges. "In-air Gestures Around Unmodified Mobile Devices." In: *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*. UIST '14. Honolulu, Hawaii, USA: ACM, 2014, pp. 319–329. ISBN: 978-1-4503-3069-5. DOI: 10.1145/2642918.2647373 (cit. on p. 104).

[221] D. Spelmezan, C. Appert, O. Chapuis, E. Pietriga. "Side Pressure for Bidirectional Navigation on Small Devices." In: *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services*. MobileHCI '13. Munich, Germany: ACM, 2013, pp. 11–20. ISBN: 978-1-4503-2273-7. DOI: 10.1145/2493190.2493199 (cit. on pp. 47, 48).

[222] M. Spindler, M. Martsch, R. Dachselt. "Going Beyond the Surface: Studying Multi-layer Interaction Above the Tabletop." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '12. Austin, Texas, USA: ACM, 2012, pp. 1277–1286. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208583 (cit. on p. 44).

[223] C. Stewart, M. Rohs, S. Kratz, G. Essl. "Characteristics of Pressure-based Input for Mobile Devices." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '10. Atlanta, Georgia, USA: ACM, 2010, pp. 801–810. ISBN: 978-1-60558-929-9. DOI: 10.1145/1753326.1753444 (cit. on p. 42).

[224] M. Sugimoto, K. Hiroki. “HybridTouch: An Intuitive Manipulation Technique for PDAs Using Their Front and Rear Surfaces.” In: *Proceedings of the 8th Conference on Human-computer Interaction with Mobile Devices and Services*. MobileHCI ’06. Helsinki, Finland: ACM, 2006, pp. 137–140. ISBN: 1-59593-390-5. DOI: 10.1145/1152215.1152243 (cit. on p. 53).

[225] N. Sugita, D. Iwai, K. Sato. “Touch sensing by image analysis of fingernail.” In: *2008 SICE Annual Conference*. 2008, pp. 1520–1525. DOI: 10.1109/SICE.2008.4654901 (cit. on p. 33).

[226] K. Sung, J. Cho, A. Freivalds. “Effects of grip span in one-handed thumb interaction with a smartphone.” In: vol. 60. HFES ’16 1. 2016, pp. 1048–1052. DOI: 10.1177/1541931213601243 (cit. on pp. 49, 73).

[227] R. P. Tanyag, R. O. Atienza. “Implicit Palm Rejection Using Real-Time Hand Model Filters on Tablet Devices.” In: *9th International Conference on Next Generation Mobile Applications, Services and Technologies*. NGMAST’15. IEEE. 2015, pp. 347–352. DOI: 10.1109/NGMAST.2015.45 (cit. on pp. 95, 101).

[228] B. T. Taylor, V. M. Bove. “The Bar of Soap: A Grasp Recognition System Implemented in a Multi-functional Handheld Device.” In: *CHI ’08 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’08. Florence, Italy: ACM, 2008, pp. 3459–3464. ISBN: 978-1-60558-012-8. DOI: 10.1145/1358628.1358874 (cit. on p. 49).

[229] T. Tieleman, G. Hinton. “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude.” In: *COURSERA: Neural networks for machine learning* 4.2 (2012), pp. 26–31 (cit. on pp. 123, 148, 194).

[230] M. B. Trudeau, J. G. Young, D. L. Jindrich, J. T. Dennerlein. “Thumb motor performance is greater for two-handed grip compared to single-handed grip on a mobile phone.” In: *Proceedings of the human factors and ergonomics society annual meeting*. Vol. 56. 1. SAGE Publications Sage CA: Los Angeles, CA. 2012, pp. 1887–1891 (cit. on p. 38).

[231] M. B. Trudeau, J. G. Young, D. L. Jindrich, J. T. Dennerlein. “Thumb motor performance varies with thumb and wrist posture during single-handed mobile phone use.” In: *Journal of Biomechanics* 45.14 (2012), pp. 2349 –2354. ISSN: 0021-9290. DOI: http://dx.doi.org/10.1016/j.jbiomech.2012.07.012 (cit. on pp. 36, 38).

[232] G. Tsoumakas, I. Katakis. “Multi-label classification: An overview.” In: *International Journal of Data Warehousing and Mining* 3.3 (2006) (cit. on p. 152).

[233] K. Turkowski. “Filters for common resampling tasks.” In: *Graphics gems*. Academic Press Professional, Inc. 1990, pp. 147–165 (cit. on pp. 121, 150).

[234] R.-D. Vatavu. “User-defined Gestures for Free-hand TV Control.” In: *Proceedings of the 10th European Conference on Interactive TV and Video*. EuroITV ’12. Berlin, Germany: ACM, 2012, pp. 45–48. ISBN: 978-1-4503-1107-6. DOI: 10.1145/2325616.2325626 (cit. on pp. 189, 207).

[235] R.-D. Vatavu, L. Anthony, J. O. Wobbrock. “Gestures As Point Clouds: A $P Recognizer for User Interface Prototypes.” In: *Proceedings of the 14th ACM International Conference on Multimodal Interaction*. ICMI ’12. Santa Monica, California, USA: ACM, 2012, pp. 273–280. ISBN: 978-1-4503-1467-1. DOI: 10.1145/2388676.2388732 (cit. on p. 40).

[236] R.-D. Vatavu, L. Anthony, J. O. Wobbrock. “Gestures As Point Clouds: A $P Recognizer for User Interface Prototypes.” In: *Proceedings of the 14th ACM International Conference on Multimodal Interaction*. ICMI ’12. Santa Monica, California, USA: ACM, 2012, pp. 273–280. ISBN: 978-1-4503-1467-1. DOI: 10.1145/2388676.2388732 (cit. on p. 153).

[237] R.-D. Vatavu, J. O. Wobbrock. “Between-Subjects Elicitation Studies: Formalization and Tool Support.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. San Jose, California, USA: ACM, 2016, pp. 3390–3402. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858228 (cit. on pp. 40, 181, 186, 209).

[238] R.-D. Vatavu, J. O. Wobbrock. “Formalizing Agreement Analysis for Elicitation Studies: New Measures, Significance Test, and Toolkit.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI ’15. Seoul, Republic of Korea: ACM, 2015, pp. 1325–1334. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702223 (cit. on pp. 40, 181, 186, 209).

[239] K. Vega, H. Fuks. “Beauty Tech Nails: Interactive Technology at Your Fingertips.” In: *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction*. TEI ’14. Munich, Germany: ACM, 2013, pp. 61–64. ISBN: 978-1-4503-2635-3. DOI: 10.1145/2540930.2540961 (cit. on p. 46).

[240] N. Villar, D. Cletheroe, G. Saul, C. Holz, T. Regan, O. Salandin, M. Sra, H.-S. Yeo, W. Field, H. Zhang. "Project Zanzibar: A Portable and Flexible Tangible Interaction Platform." In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI '18. Montreal QC, Canada: ACM, 2018, 515:1–515:13. ISBN: 978-1-4503-5620-6. DOI: 10.1145/3173574.3174089 (cit. on p. 45).

[241] N. Villar, D. Cletheroe, G. Saul, C. Holz, T. Regan, O. Salandin, M. Sra, H.-S. Yeo, W. Field, H. Zhang. "Project Zanzibar Demonstration: A Portable and Flexible Tangible Interaction Platform." In: *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI EA '18. Montreal QC, Canada: ACM, 2018, D324:1–D324:4. ISBN: 978-1-4503-5621-3. DOI: 10.1145/3170427.3186484 (cit. on p. 45).

[242] H. P. von Schroeder, M. J. Botte. "The functional significance of the long extensors and juncturae tendinum in finger extension." In: *Journal of Hand Surgery* 18.4 (1993), pp. 641–647 (cit. on p. 38).

[243] E. J. Wang, J. Garrison, E. Whitmire, M. Goel, S. Patel. "Carpacio: Repurposing Capacitive Sensors to Distinguish Driver and Passenger Touches on In-Vehicle Screens." In: *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. UIST '17. Québec City, QC, Canada: ACM, 2017, pp. 49–55. ISBN: 978-1-4503-4981-9. DOI: 10.1145/3126594.3126623 (cit. on pp. 45, 47).

[244] F. Wang, X. Cao, X. Ren, P. Irani. "Detecting and Leveraging Finger Orientation for Interaction with Direct-touch Surfaces." In: *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology*. UIST '09. Victoria, BC, Canada: ACM, 2009, pp. 23–32. ISBN: 978-1-60558-745-5. DOI: 10.1145/1622176.1622182 (cit. on p. 41).

[245] J. Wang, J. Canny. "FingerSense: Augmenting Expressiveness to Physical Pushing Button by Fingertip Identification." In: *CHI '04 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '04. Vienna, Austria: ACM, 2004, pp. 1267–1270. ISBN: 1-58113-703-6. DOI: 10.1145/985921.986040 (cit. on pp. 46, 100).

[246] W. Wang, A. X. Liu, K. Sun. "Device-free Gesture Tracking Using Acoustic Signals." In: *Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking*. MobiCom '16. New York City, New York: ACM, 2016, pp. 82–94. ISBN: 978-1-4503-4226-1. DOI: 10.1145/2973750.2973764 (cit. on p. 49).

[247] D. Weber, A. Voit, H. V. Le, N. Henze. “Notification Dashboard: Enabling Reflection on Mobile Notifications.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. MobileHCI ’16. Florence, Italy: ACM, 2016, pp. 936–941. ISBN: 978-1-4503-4413-5. DOI: 10.1145/2957265.2962660 (cit. on p. 26).

[248] F. Weichert, D. Bachmann, B. Rudak, D. Fisseler. “Analysis of the accuracy and robustness of the leap motion controller.” In: *Sensors* 13.5 (2013), pp. 6380–6393 (cit. on p. 46).

[249] D. Wigdor, R. Balakrishnan. “TiltText: Using Tilt for Text Input to Mobile Phones.” In: *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*. UIST ’03. Vancouver, Canada: ACM, 2003, pp. 81–90. ISBN: 1-58113-636-6. DOI: 10.1145/964696.964705 (cit. on p. 43).

[250] D. Wigdor, C. Forlines, P. Baudisch, J. Barnwell, C. Shen. “Lucid Touch: A See-through Mobile Device.” In: *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*. UIST ’07. Newport, Rhode Island, USA: ACM, 2007, pp. 269–278. ISBN: 978-1-59593-679-0. DOI: 10.1145/1294211.1294259 (cit. on pp. 17, 53, 55, 164).

[251] G. Wilkinson, A. Kharrufa, J. Hook, B. Pursglove, G. Wood, H. Haeuser, N. Y. Hammerla, S. Hodges, P. Olivier. “Expressy: Using a Wrist-worn Inertial Measurement Unit to Add Expressiveness to Touch-based Interactions.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. Santa Clara, California, USA: ACM, 2016, pp. 2832–2844. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858223 (cit. on p. 44).

[252] A. D. Wilson. “Using a Depth Camera As a Touch Sensor.” In: *ACM International Conference on Interactive Tabletops and Surfaces*. ITS ’10. Saarbrücken, Germany: ACM, 2010, pp. 69–72. ISBN: 978-1-4503-0399-6. DOI: 10.1145/1936652.1936665 (cit. on pp. 33, 46, 100).

[253] G. Wilson, S. Brewster, M. Halvey. “Towards Utilising One-handed Multi-digit Pressure Input.” In: *CHI ’13 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’13. Paris, France: ACM, 2013, pp. 1317–1322. ISBN: 978-1-4503-1952-2. DOI: 10.1145/2468356.2468591 (cit. on p. 48).

[254] R. Wimmer. “Grasp Sensing for Human-computer Interaction.” In: *Proceedings of the Fifth International Conference on Tangible, Embedded, and Embodied Interaction*. TEI ’11. Funchal, Portugal: ACM, 2011, pp. 221–228. ISBN: 978-1-4503-0478-8. DOI: 10.1145/1935701.1935745 (cit. on p. 37).

[255] J. O. Wobbrock, M. R. Morris, A. D. Wilson. “User-defined Gestures for Surface Computing.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’09. Boston, MA, USA: ACM, 2009, pp. 1083–1092. ISBN: 978-1-60558-246-7. DOI: 10.1145/1518701.1518866 (cit. on pp. 40, 173, 181, 185, 199, 207, 209).

[256] J. O. Wobbrock, B. A. Myers, H. H. Aung. “The performance of hand postures in front- and back-of-device interaction for mobile computing.” In: *International Journal of Human-Computer Studies* 66.12 (2008), pp. 857 –875. ISSN: 1071-5819. DOI: http://dx.doi.org/10.1016/j.ijhcs.2008.03.004 (cit. on pp. 37, 40, 69).

[257] J. O. Wobbrock, A. D. Wilson, Y. Li. “Gestures Without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes.” In: *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*. UIST ’07. Newport, Rhode Island, USA: ACM, 2007, pp. 159–168. ISBN: 978-1-59593-679-0. DOI: 10.1145/1294211.1294238 (cit. on p. 40).

[258] J. O. Wobbrock, A. D. Wilson, Y. Li. “Gestures Without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes.” In: *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*. UIST ’07. Newport, Rhode Island, USA: ACM, 2007, pp. 159–168. ISBN: 978-1-59593-679-0. DOI: 10.1145/1294211.1294238 (cit. on p. 153).

[259] J. O. Wobbrock, L. Findlater, D. Gergle, J. J. Higgins. “The Aligned Rank Transform for Nonparametric Factorial Analyses Using Only Anova Procedures.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: ACM, 2011, pp. 143–146. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1978963 (cit. on pp. 85, 200).

[260] K. Wolf, R. Schleicher, M. Rohs. “Ergonomic Characteristics of Gestures for Front- and Back-of-tablets Interaction with Grasping Hands.” In: *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*. MobileHCI ’14. Toronto, ON, Canada: ACM, 2014, pp. 453–458. ISBN: 978-1-4503-3004-6. DOI: 10.1145/2628363.2634214 (cit. on p. 37).

[261] K. Wolf, C. Müller-Tomfelde, K. Cheng, I. Wechsung. “Does Proprioception Guide Back-of-device Pointing As Well As Vision?” In: *CHI ’12 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’12. Austin, Texas, USA: ACM, 2012, pp. 1739–1744. ISBN: 978-1-4503-1016-1. DOI: 10.1145/2212776.2223702 (cit. on p. 49).

[262] K. Wolf, C. Müller-Tomfelde, K. Cheng, I. Wechsung. “PinchPad: Performance of Touch-based Gestures While Grasping Devices.” In: *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*. TEI ’12. Kingston, Ontario, Canada: ACM, 2012, pp. 103–110. ISBN: 978-1-4503-1174-8. DOI: 10.1145/2148131.2148155 (cit. on p. 49).

[263] P. C. Wong, H. Fu, K. Zhu. “Back-Mirror: Back-of-device One-handed Interaction on Smartphones.” In: *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications*. SA ’16. Macau: ACM, 2016, 10:1–10:5. ISBN: 978-1-4503-4551-4. DOI: 10.1145/2999508.2999522 (cit. on p. 49).

[264] H. Xia, R. Jota, B. McCanny, Z. Yu, C. Forlines, K. Singh, D. Wigdor. “Zero-latency Tapping: Using Hover Information to Predict Touch Locations and Eliminate Touchdown Latency.” In: *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*. UIST ’14. Honolulu, Hawaii, USA: ACM, 2014, pp. 205–214. ISBN: 978-1-4503-3069-5. DOI: 10.1145/2642918.2647348 (cit. on p. 44).

[265] R. Xiao, J. Schwarz, C. Harrison. “Estimating 3D Finger Angle on Commodity Touchscreens.” In: *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*. ITS ’15. Madeira, Portugal: ACM, 2015, pp. 47–50. ISBN: 978-1-4503-3899-8. DOI: 10.1145/2817721.2817737 (cit. on pp. 16, 41, 94, 100, 122).

[266] X. Xiao, T. Han, J. Wang. “LensGesture: Augmenting Mobile Interactions with Back-of-device Finger Gestures.” In: *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*. ICMI ’13. Sydney, Australia: ACM, 2013, pp. 287–294. ISBN: 978-1-4503-2129-7. DOI: 10.1145/2522848.2522850 (cit. on pp. 47–49).

[267] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, W.-c. Woo. “Convolutional LSTM network: A machine learning approach for precipitation nowcasting.” In: *Advances in neural information processing systems*. 2015, pp. 802–810 (cit. on p. 193).

[268] J. Xiong, S. Muraki. “An ergonomics study of thumb movements on smartphone touch screen.” In: *Ergonomics* 57.6 (2014), pp. 943–955. DOI: 10.1080/00140139.2014.904007 (cit. on p. 36).

[269] X.-D. Yang, P. Irani, P. Boulanger, W. Bischof. “One-handed Behind-the-display Cursor Input on Mobile Devices.” In: *CHI ’09 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’09. Boston, MA, USA: ACM, 2009, pp. 4501–4506. ISBN: 978-1-60558-247-4. DOI: 10.1145/1520340.1520690 (cit. on pp. 17, 53).

[270] X.-D. Yang, T. Grossman, P. Irani, G. Fitzmaurice. “TouchCuts and TouchZoom: Enhanced Target Selection for Touch Displays Using Finger Proximity Sensing.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’11. Vancouver, BC, Canada: ACM, 2011, pp. 2585–2594. ISBN: 978-1-4503-0228-9. DOI: 10.1145/1978942.1979319 (cit. on p. 44).

[271] K. Yatani, K. Partridge, M. Bern, M. W. Newman. “Escape: A Target Selection Technique Using Visually-cued Gestures.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’08. Florence, Italy: ACM, 2008, pp. 285–294. ISBN: 978-1-60558-011-1. DOI: 10.1145/1357054.1357104 (cit. on p. 55).

[272] S. Ye. *The science behind Force Touch and the Taptic Engine (iMore.com)*. Last access: 2017-09-09. 2015. URL: http://www.imore.com/science-behind-taptics-and-force-touch (visited on 09/01/2010) (cit. on p. 47).

[273] H.-S. Yeo, X.-S. Phang, S. J. Castellucci, P. O. Kristensson, A. Quigley. “Investigating Tilt-based Gesture Keyboard Entry for Single-Handed Text Entry on Large Devices.” In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI ’17. Denver, Colorado, USA: ACM, 2017, pp. 4194–4202. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025520 (cit. on p. 43).

[274] H.-S. Yeo, G. Flamich, P. Schrempf, D. Harris-Birtill, A. Quigley. “RadarCat: Radar Categorization for Input & Interaction.” In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. UIST ’16. Tokyo, Japan: ACM, 2016, pp. 833–841. ISBN: 978-1-4503-4189-9. DOI: 10.1145/2984511.2984515 (cit. on p. 104).

[275] Y. Yin, T. Y. Ouyang, K. Partridge, S. Zhai. “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. Paris, France: ACM, 2013, pp. 2775–2784. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2481384 (cit. on p. 100).

[276] H. Yoo, J. Yoon, H. Ji. “Index Finger Zone: Study on Touchable Area Expandability Using Thumb and Index Finger.” In: *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. MobileHCI ’15. Copenhagen, Denmark: ACM, 2015, pp. 803–810. ISBN: 978-1-4503-3653-6. DOI: 10.1145/2786567.2793704 (cit. on pp. 36, 55, 165).

[277] S. Zhai, P.-O. Kristensson. “Shorthand Writing on Stylus Keyboard.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’03. Ft. Lauderdale, Florida, USA: ACM, 2003, pp. 97–104. ISBN: 1-58113-630-7. DOI: 10.1145/642611.642630 (cit. on p. 40).

[278] S. Zhai, P. O. Kristensson, C. Appert, T. H. Anderson, X. Cao, et al. “Foundational issues in touch-surface stroke gesture design—an integrative review.” In: *Foundations and Trends in Human–Computer Interaction* 5.2 (2012) (cit. on p. 40).

[279] C. Zhang, A. Guo, D. Zhang, C. Southern, R. Arriaga, G. Abowd. “BeyondTouch: Extending the Input Language with Built-in Sensors on Commodity Smartphones.” In: *Proceedings of the 20th International Conference on Intelligent User Interfaces*. IUI ’15. Atlanta, Georgia, USA: ACM, 2015, pp. 67–77. ISBN: 978-1-4503-3306-1. DOI: 10.1145/2678025.2701374 (cit. on p. 49).

[280] H. Zhang, X.-D. Yang, B. Ens, H.-N. Liang, P. Boulanger, P. Irani. “See Me, See You: A Lightweight Method for Discriminating User Touches on Tabletop Displays.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’12. Austin, Texas, USA: ACM, 2012, pp. 2327–2336. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208392 (cit. on p. 47).

[281] Y. Zhang, W. Chan, N. Jaitly. “Very deep convolutional networks for end-to-end speech recognition.” In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017, pp. 4845–4849. DOI: 10.1109/ICASSP.2017.7953077 (cit. on p. 193).

[282] Y. Zhang, G. Laput, C. Harrison. “Electrick: Low-Cost Touch Sensing Using Electric Field Tomography.” In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI ’17. Denver, Colorado, USA: ACM, 2017, pp. 1–14. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025842 (cit. on p. 49).

[283] Y. Zhang, J. Zhou, G. Laput, C. Harrison. “SkinTrack: Using the Body As an Electrical Waveguide for Continuous Finger Tracking on the Skin.” In: *Proceedings*

of the 2016 CHI Conference on Human Factors in Computing Systems. CHI '16. Santa Clara, California, USA: ACM, 2016, pp. 1491–1503. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858082 (cit. on p. 49).

[284] J. Zheng, D. Vogel. "Finger-Aware Shortcuts." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. San Jose, California, USA: ACM, 2016, pp. 4274–4285. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858355 (cit. on pp. 46, 51, 99, 100, 168).

[285] statista. *Number of smartphone users worldwide from 2014 to 2020 (in billions)*. https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/. 2019 (cit. on pp. 15, 29).

All URLs cited were checked in January 2019.

List of Acronyms

AdaGrad Adaptive Gradient Algorithm

AGATe AGreement Analysis Toolkit

BoD Back-of-Device

CNN convolutional neural network

DoF degrees of freedom

DT Decision Tree

FTIR frustrated total internal reflection

FTSP fully touch sensitive smartphone

HCI human-computer interaction

IMU inertial measurement unit

IR infrared

ITO indium tin oxide

***k*NN** *k*-nearest neighbor

LCD liquid crystal display

LSTM Long short-term memory

MAE mean absolute error

MCP metacarpophalangeal joint

NN neural network

PDA personal digital assistant

RF Random Forest

RMSE root mean squared error

RMSLE root mean squared logarithmic error

SFCS Swept Frequency Capacitive Sensing

SimTech Cluster of Excellence in Simulation Technology

SVM Support Vector Machine

TCT task completion time

UCD user-centered design

UCDDL user-centered design process for deep learning

UI user interface

Huy Viet Le

Hand-and-Finger-Awareness for Mobile Touch Interaction using Deep Learning

Mobile devices such as smartphones and tablets have replaced desktop computers for a wide range of everyday tasks. Virtually every smartphone incorporates a touchscreen which enables an intuitive interaction through a combination of input and output in a single interface. Despite the success of touchscreens, traditional input devices such as keyboard and mouse are still superior due to their rich input capabilities. Touch input is limited to the two-dimensional location of touches which slow down the interaction and pose a number of challenges which affect the usability. Novel touch-based interaction techniques are needed to extend the touch input capabilities and enable multiple fingers and parts of the hand to perform input similar to traditional input devices.

This dissertation presents the results of twelve studies examining how individual fingers as well as other parts of the hand can be recognized and used for touch input. We refer to this concept as hand-and-finger-awareness for mobile touch interaction. By identifying the source of input, different functions and action modifiers can be assigned to individual fingers and parts of the hand. We show that this concept increases the touch input capabilities and solves a number of touch input challenges. The contribution of this dissertation ranges from insights on the use of different fingers and parts of the hand for interaction, through technical contributions for the identification of the touch source using deep learning, to solutions for addressing limitations of mobile touch input.