Interacting with Cyber-Physical Systems
Advancements in Gesture Control and Eye-based Human-Computer Interaction

A thesis submitted to attain the degree of
DOCTOR OF SCIENCES of ETH ZURICH
(Dr. Sc. ETH Zurich)

presented by
MIHAI BĂCE
M.Sc. in Computer Science, École Polytechnique Fédérale de Lausanne
born on 14 June 1989, citizen of Romania

accepted on the recommendation of
Prof. Dr. Friedemann Mattern, examiner
Prof. Dr. Andreas Bulling, co-examiner
Prof. Dr. Otmar Hilliges, co-examiner

2020
Abstract

With an ever-increasing number of smart devices in the users’ surrounding that create ubiquitous opportunities for interaction, enabling natural and intuitive interaction with digital systems is a key challenge in Human-Computer Interaction (HCI). Computing technology is steadily getting smaller, cheaper, and more interconnected, including sensing and communication technology, currently witnessed by mobile and wearable devices of various forms. Smartwatches and fitness trackers are already ubiquitous, while smart glasses or augmented reality headsets such as the Microsoft HoloLens are closer to real products than to lab prototypes. Given such a changing landscape of devices in our environment, current input modalities may limit the effective utilisation of the available information flow.

When people interact with each other, they use multiple modalities including voice, body posture, hand gestures, facial expressions, or eye gaze. Touch interactions coupled with apps on our smartphones are today’s universal interaction devices and have recently been extended with voice control. However, such input modalities are no longer sufficient for complex applications like augmented or virtual reality. In this dissertation, we broaden and facilitate the interface between technology and people and contribute to the following research areas:

i) **Collocated Multi-user Gestural Interactions.** Natural gestures are a promising input modality in HCI because they enrich the way we interact with complex systems. However, few works have explored this input technique when multiple users are physically collocated. We explore multi-user interaction through gestures between people who are physically close to one another. The proximity of users is detected through inaudible acoustic ranging while in-air hand gestures are recognised by leveraging inertial sensors. To ensure scalability, the underlying communication protocol between users and devices is handled over Bluetooth. Through extensive evaluations, we demonstrate not only the robustness of our approach but also the feasibility of using off-the-shelf mobile and wearable devices. We first demonstrate our concept with *HandshakAR*, a wearable system...
that facilitates the exchange of digital information between two users. We then extend this concept to multiple users and show three different practical application scenarios.

ii) **Eye-based Human-Computer Interaction.** We investigate eye gaze as a hands-free high-bandwidth input modality and explore how users can augment their interaction capabilities with the gaze direction. We propose *ubiGaze*, a novel wearable system that enables attaching virtual content to any real-world object through gaze gestures, which are detected using a wearable eye tracker. While gaze gestures are less sensitive to accuracy problems or calibration shifts, many practical applications require calibrated eye trackers. However, existing calibration techniques are tedious and rely on special markers. We propose finger calibration, a novel method for head-mounted eye trackers in which users only have to point with their fingers at locations in the scene. This eliminates the need for additional assistance or specialised markers. Lastly, while eye trackers have become more accessible, the need for special-purpose equipment hinders many large-scale deployments. Therefore, we explore gaze-based interaction using a single off-the-shelf camera. We propose a method to detect pursuit eye movements, which have become widely popular because they enable spontaneous interaction. Our method combines appearance-based gaze estimation with optical flow in the eye region to jointly analyse eye movement dynamics in a single pipeline. Our results not only show the feasibility of our approach but point the way towards new methods that only require standard cameras, which are readily available in an ever increasing number of devices.

iii) **Quantification of Visual Attention in Mobile HCI.** Eye contact is a key measure of overt visual attention in mobile HCI as it enables understanding when, how often, or for how long users look at their devices. However, robustly detecting shifts of attention during everyday mobile interactions is challenging. Encouraged by recent advances in automatic eye contact detection, we provide a fundamental investigation into the feasibility of quantifying mobile visual attention. We identify core challenges and sources of errors associated with sensing visual attention in the wild, including the impact of face and eye visibility, the importance of robust head pose estimation, and the need for accurate gaze estimation. Guided by our analysis, we present a method to accurately and robustly detect eye contact in images captured with the front-facing camera of common mobile devices. Based on our evaluations, we show how eye contact is the fundamental building block for calculating higher-level attention metrics and, as such, enables studying visual attention in the wild. Finally, we present the *Everyday Mobile Visual Attention (EMVA) dataset* and quantitative evaluations of visual attention of mobile device users in-situ, i.e. while they use their devices during everyday routine. Using our proposed method for eye contact detection, we quantify the highly dynamic nature of everyday visual attention allocation across users, mobile applications, and usage contexts. We then discuss our findings which highlight the potential and inform the design of future mobile attentive user interfaces.
Along with this dissertation, we deliver open-source implementations of some of our contributions. The fingertip calibration method is available as a plugin for the open-source Pupil platform. Additionally, the EMVA dataset, which contains around 472 hours of video snippets from 32 participants recorded over more than two weeks in real life using the front-facing camera as well as associated usage logs, interaction events, and sensor data, is publicly available to support future work in this area of research.
Zusammenfassung


Für die Interaktion untereinander verwenden Menschen vielfältige Kanäle, u.a. Sprache, Körperhaltung, Handzeichen, Gesichtsausdrücke und aussagekräftige Blicke. Im Unterschied dazu interagieren wir mit neueren elektronischen Geräten im Wesentlichen über berührungsempfindliche Schnittstellen und Displays (die ihrerseits Apps ansteuern); in jüngster Zeit auch über Sprachsteuerung. Diese Interaktionsparadigmen erscheinen jedoch immer weniger ausreichend für komplexer werdende Anwendungen in Bereichen wie erweiterte oder virtuelle Realität. Die vorliegende Dissertation setzt sich daher das Ziel, die Schnittstelle zwischen Mensch und Maschine zu erweitern und gleichzeitig in der Nutzung vereinfachen. Im Einzelnen tragen wir zu folgenden Forschungsgebieten bei:

i) Gestenbasierte Interaktion bei mehreren Personen. Natürliche Gesten sind eine vielversprechende Eingabemöglichkeit für HCI; sie bereichern die Art und Weise, wie wir mit komplexen Systemen interagieren können. Erst wenige Arbeiten haben die Gesteerkennung für Situationen untersucht, wo mehrere Nutzer nah beieinander sind. Wir analy-
Zusammenfassung


Acknowledgements

One of the key elements of a successful PhD is, in my view, having the right people around you. I was lucky enough to have the full support not only of those that I have worked with but also my family and friends.

First and foremost, I would like to thank my advisor, Prof. Friedemann Mattern, who gave me the chance to pursue my PhD at ETH Zurich. He fully supported me, provided guidance, and allowed me to pursue my own research interests. I would also like to thank my co-examiners, Prof. Andreas Bulling and Prof. Otmar Hilliges. In particular, thanks to Andreas Bulling for his mentorship and for contributing to shaping and developing my research direction.

Over the years, in the Distributed Systems Group, I’ve had many colleagues that made everyday life fun and exciting. I would first like to mention my office mates Vincent Becker and Leyna Sadamori for always contributing to a friendly and pleasant office atmosphere. Also, thanks to the current members of the group Liliana Barrios, Lukas Burkhalter, Alexander Viand, and Jing Yang. Thanks to Vlad Coroamă for being there not only as a colleague but also as a friend. From the former group members, I would like to mention Robert Adelmann, Subho Basu, Christian Beckel, Hông-Ân Cao, Christian Floerkemeier, Marian George, Anwar Hithnawi, Wilhelm Kleiminger, Matthias Kovatsch, Simon Mayer, Hossein Shafagh, Aurelia Tamò. Special thanks go to Gábor Sörös and his family for all the great times we’ve had together in Zurich. This list would not be complete without mentioning Barbara von Allmen Wilson, who was always there for us.

I would also like to thank Yvonne-Anne Pignolet for inspiring me to pursue a research career and for putting me in contact with Robert Adelmann, who was my first contact with Prof. Mattern’s group.

The work that I have done during my PhD would not have been possible without my co-authors and collaborators. Moreover, I have been fortunate to work with many excellent students: Erfan Abdi, Amirreza Bahreini, Carlo Beltrame, Michael Franz, Patrick Huwiler,

My family and, in particular, my parents deserve special recognition! They were the ones that supported me for all those years and always encouraged me to aim high and strive to achieve my dreams and goals.

Last but not least, I would like to thank my wife, Ruxandra. She has been a pillar that provided continuous support through the many deadlines, long hours, for good and for worse. She was understanding and supportive of my ambitions even though it has not always been easy. I cannot thank you enough!
Contents

1 Introduction 1
  1.1 Collocated Multi-user Gestural Interactions .................... 5
  1.2 Eye-based Human-Computer Interaction .......................... 8
  1.3 Quantification of Visual Attention in Mobile HCI ............... 13
  1.4 Dissertation Roadmap ............................................. 16
  1.5 Statement of Contributions ........................................ 17
  1.6 Accompanying Publications ....................................... 18

2 Collocated Multi-user Gestural Interactions 21
  2.1 Introduction ....................................................... 22
  2.2 Related Work ...................................................... 23
    2.2.1 Multi-user and Multi-device Interactions .................... 24
    2.2.2 Collocated Interactions ...................................... 24
    2.2.3 Gestural Interaction with Wearables ........................ 26
  2.3 Collocated Multi-User Gestural Interactions ..................... 27
    2.3.1 Overview ..................................................... 27
    2.3.2 Communication via Bluetooth ................................ 27
    2.3.3 Proximity Detection via Acoustic Ranging ................... 28
    2.3.4 Gesture Recognition via Motion Sensors ..................... 31
  2.4 Evaluation ......................................................... 32
    2.4.1 Gesture Recognition .......................................... 32
    2.4.2 Acoustic Ranging ............................................. 36
## Contents

2.4.3 Combined Evaluation: Acoustic Ranging While Performing Hand Gestures ........................................ 43

2.5 Applications .............................................................................................................................. 47
  2.5.1 HandshakAR ........................................................................................................ 47
  2.5.2 Treasure Chest ....................................................................................................... 48
  2.5.3 Collaborative Fitness ............................................................................................. 48
  2.5.4 Collaborative Music Band ....................................................................................... 49

2.6 Discussion .............................................................................................................................. 50

2.7 Conclusion ............................................................................................................................. 51

3 Eye-based Human-Computer Interaction .................................................................................. 53
  3.1 Introduction ......................................................................................................................... 54
  3.2 ubiGaze: Ubiquitous Augmented Reality Messaging Using Gaze Gestures ................. 55
    3.2.1 Related Work ................................................................................................................. 56
    3.2.2 Gaze Gestures .............................................................................................................. 57
    3.2.3 Overview ..................................................................................................................... 58
    3.2.4 Implementation ............................................................................................................ 60
    3.2.5 Challenges and Future Research Directions ............................................................... 61
    3.2.6 Conclusion .................................................................................................................. 62
  3.3 Wearable Eye Tracker Calibration at Your Fingertips ......................................................... 63
    3.3.1 Related Work ................................................................................................................. 64
    3.3.2 Fingertip Calibration ................................................................................................. 65
    3.3.3 Evaluation .................................................................................................................... 67
    3.3.4 Discussion ................................................................................................................... 71
    3.3.5 Conclusion .................................................................................................................. 72
  3.4 Combining Gaze Estimation and Optical Flow for Pursuits Interaction ....................... 73
    3.4.1 Related Work ................................................................................................................. 74
    3.4.2 Method .......................................................................................................................... 77
    3.4.3 Evaluation .................................................................................................................... 80
    3.4.4 Discussion ................................................................................................................... 90
    3.4.5 Conclusion .................................................................................................................. 93
4 Quantification of Users' Visual Attention in Mobile HCI 95

4.1 Introduction .......................... 96

4.2 Related Work .......................... 99
  4.2.1 Gaze Estimation ...................... 100
  4.2.2 Eye Contact Detection ............. 100
  4.2.3 Modelling User Behaviour on Mobile Devices 101
  4.2.4 Visual Attention Sensing ............ 102

4.3 Key Challenges in Quantifying Mobile Visual Attention .............. 103
  4.3.1 Evaluation Datasets ............... 104
  4.3.2 Overall Performance ............... 105
  4.3.3 Challenge 1: Face and Eye (In)visibility ........... 106
  4.3.4 Challenge 2: Robust Head Pose Estimation ............ 108
  4.3.5 Challenge 3: Accurate Gaze Estimation ............. 110
  4.3.6 Discussion ......................... 111

4.4 Accurate and Robust Detection of Eye Contact with Mobile Devices ... 112
  4.4.1 Method Overview ................... 112
  4.4.2 Evaluation ......................... 115
  4.4.3 Discussion ......................... 128

4.5 The Everyday Mobile Visual Attention Dataset ....................... 131
  4.5.1 Data Logging Application .......... 131
  4.5.2 Video Review Component .......... 134
  4.5.3 Annotation Game ................. 135
  4.5.4 Data Collection ................. 136
  4.5.5 Dataset Characteristics .......... 138

4.6 Quantification of Mobile Visual Attention ......................... 140
  4.6.1 Visual Attention Across Participants .......... 142
  4.6.2 Visual Attention Across Applications ............ 145
  4.6.3 Visual Attention Across Usage Contexts .......... 145

4.7 Discussion .................................. 147
  4.7.1 Limitations and Future Work .......... 149
## Contents

4.8 Conclusion .................................................. 150

5 Conclusions and Outlook ...................................... 151

  5.1 Summary of Contributions .................................. 151
  5.2 Future Work ................................................ 154
  5.3 Closing Remarks ............................................ 155

Bibliography ..................................................... 157

Web References and Image Sources ............................ 189
Introduction

In 1991, Mark Weiser envisioned a world in which computers would “weave themselves into the fabric of everyday life until they are indistinguishable from it” [228]. This is not a fundamental consequence of newer fabrication methods or technology miniaturisation but rather human psychology: As things become ever-more present in everyday life, they become less noticeable over time. Nowadays, this is the case for computing technology that has gotten smaller, cheaper, and more interconnected. Technology comes ever-closer to humans as witnessed by a multitude of not only personal computers but also mobile and wearable devices. For example, smartwatches and fitness trackers are already ubiquitous, while smartglasses like Google Glass, Microsoft HoloLens, or the Daqri smart helmet are closer to real, consumer products than to lab prototypes. Such complex systems that have deeply interconnected hardware and software components are called cyber-physical systems (CPS) [232]. CPSs have recently seen an exponential increase in the number of digital services that they offer, which create constant interaction demands to their users. Therefore, the way users interact with digital interfaces has become a fundamental research problem in Human-Computer Interaction (HCI).

Traditionally, mechanical keyboards and mice have been and still are the default modality to provide input and control personal computers. While these techniques are familiar and widely accepted, they are not suitable for the novel generation of computing devices such as smartphones, smart glasses, or so-called smart devices, which are everyday objects that are embedded with computational capabilities [145]. Moreover, users no longer own a single personal computer, but many different devices with a wide range of different functionalities. In this context, keyboards and mice reduce the speed and naturalness with which we can interact with devices leading to an HCI bottleneck, which may limit the effective utilisation of the available information flow [169]. To overcome this major
limitation, there is a shift to replace or to extend these classical input modalities with touch-based interfaces. Touch interactions and “apps” on our smartphones or tablets are today’s universal interaction devices [147]. Controlling a smart appliance, configuring an internet router, or adjusting the heating and ventilation system of a house are all an app-installation away. Besides advancements in touch-sensitive surfaces and displays, this revolution has been, in part, possible because touching an object of interest is natural and intuitive human behaviour that does not require special training or understanding. However, while touch interactions have made significant strides in facilitating the user experience when interacting with complex systems, they have several limitations. They are less precise than a mouse and the hand may occlude parts of the screen and hence limit the available screen real estate. For example, touch interactions on a smartwatch may occlude large parts of the display [8] (also known as the fat finger problem). To address some of these limitations, more recently, touch interactions have been complemented by voice control. Advancements in speech recognition and natural language understanding have made it possible to “talk” to smart devices, control them, or even interact with intelligent assistants such as Apple’s Siri or Amazon’s Alexa. However, just as touch interactions have certain limitations, voice control is mainly bound to spoken commands, it typically requires a handheld device, the physical context is rarely considered and it is not suitable in environments with large amounts of noise. In addition, with new technologies and interactive experiences such as augmented or virtual reality (AR or VR), sensing what the users touch or what they speak is no longer sufficient. In order to avoid a new HCI bottleneck, we argue that advanced interactive technologies and the next generation of user interfaces also have to understand additional contextual cues.

Context can be defined as “any information that can be used to characterise the situation of an entity” [44]. However, it is important that this information is relevant to the interaction between the user and the application. For example, in an indoor interactive scenario, the weather may not provide appropriate context, yet the presence of other people nearby could [44]. When people interact with one another they use different modalities to communicate and exchange information both verbal, i.e. through speech, and non-verbal communication such as body language, interpersonal distance, (hand) gestures, or the eye gaze direction. However, current computing devices can sense only a subset of these communication or contextual cues, as mentioned previously, mostly touch interactions and spoken commands. With the new generation of computing devices that have many features, what makes them particularly interesting for research is that, by coming ever-closer to humans and even on the users’ body, they provide an egocentric perspective, i.e. perceive how users move, where they are, or where they look. For example, smart glasses can see what we see [147], smart watches can sense how we move our hands, and mobile devices can capture insights about human behaviour that were unthinkable before. These devices are almost always on, offer continuous sensing, continuous access
to information, and continuous interaction with their users. This first-person perspective can capture the world from a more detailed and relevant viewpoint, which is not possible with infrastructure sensors or cameras. Furthermore, wearable displays (visual, tactile, audio, etc.) are powerful output channels that can enhance our senses. Life-logging, memory enhancement, and AR are only a few of the applications enabled by this new perspective, but how can we interact with increasingly complex systems? More formally, the general research problem addressed in this dissertation could be described as follows: complex cyber-physical systems in our direct environment, which have access to powerful knowledge bases and analysis tools for sensory data residing in the Internet, could provide users with many helpful services - but how can we enable non-technical users to interact with such systems in a simple, effortless, and natural way?

Main contributions. In this dissertation, we leverage mobile and wearable computing together with advancements in machine learning, in particular deep learning, image understanding, and sensor data analytics to study two contextual cues that advance the interaction between humans and computers: Gestural interactions while taking physical proximity into account and eye-based interaction. We summarise the main contributions as follows:

• Collocated Multi-user Gestural Interactions. We explore multi-user gestural interactions between people who are physically close to one another. To demonstrate the robustness and feasibility of our approach, we thoroughly evaluate our systems with off-the-shelf mobile and wearable devices. Our first prototype implementation is HandshakAR, a novel wearable AR system that enables effortless sharing of digital information between two people [15]. We then extend this concept to multiple users and demonstrate three different practical application scenarios such as collaborative music playing [20]. Our efforts are significant in that they, for the first time, highlight the potential for implicit information exchange between a group of people through off-the-shelf wearable technology and natural gestural interactions.

• Eye-based Human-Computer Interaction. Eye gaze is a fundamental component of human behaviour that can be used to augment human interaction capabilities with a hands-free high-bandwidth input modality. Towards this goal, we first propose ubiGaze, a novel wearable systems that enables tagging and retrieving information from real-objects through gaze gestures, which are detected by a wearable eye tracker. While gaze gestures are less sensitive to calibration issues or inaccuracies from the tracking algorithm, many real-world applications that rely on eye tracking have to go through a tedious and time consuming calibration approach. This problem is even more relevant for wearable eye trackers that are used on the go and, potentially, without access to the classical calibration routines, such as N-point calibration patterns [104]. To address this limitation, we propose fingertip
calibration for wearable head-mounted eye trackers: Users only have to point at locations in the scene using their finger and our algorithm will automatically collect calibration samples [19]. Our implementation of fingertip calibration is openly available for the open-source Pupil platform. While wearable eye trackers bring us closer to the vision of pervasive eye tracking in the wild [31], the need for expensive and dedicated eye tracking equipment significantly limits the number of users to benefit from gaze-based interaction. One such interaction technique is Pursuits, which leverages the smooth pursuit eye movements to enable spontaneous and calibration-free interaction. To overcome the need for specialised equipment, we explore pursuit interaction using a single off-the-shelf RGB camera [11]. Our method is the first to enable the detection of smooth pursuit eye movements in remote settings by combining gaze estimation and optical flow in the eye region to jointly analyse the eye movement dynamics.

- **Quantification of Visual Attention in Mobile HCI.** With an ever-increasing number of digital interfaces competing for the users’ limited attentional resources, studying and quantifying user attention has become a fundamental research problem in HCI. In this work, we investigate a subset of this task, yet a problem that is relevant to millions of mobile device users. While the users’ full attention is, in general, difficult to measure and requires special stimuli [69], what is practically relevant in HCI is visual attention, also known as overt attention, because it enables quantifying when, how often, or for how long users look at their mobile devices. Towards this goal, we first identify the key challenges and sources of error associated with sensing visual attention in the wild [17]. Then, we propose a novel method for eye contact detection using the front-facing camera of mobile devices as a measure of overt attention [16]. Using this method for eye contact detection, we are first to quantify the highly dynamic nature of visual attention during everyday mobile device interactions [18]. Our results and evaluations are based on the Everyday Mobile Visual Attention Dataset (EMVA), which is the first dataset to record high-resolution and high-granularity multi-modal data of mobile device users. To support further research, the EMVA dataset has been made publicly available.

In the sections that follow, we present a general overview of the different areas covered by this dissertation and highlight our contributions in detail, starting with collocated multi-user gestural interactions (Section 1.1), continuing with eye-based interaction (Section 1.2), and finally discussing (visual) attention (Section 1.3).
1.1 Collocated Multi-user Gestural Interactions

Gestures are a form of non-verbal communication that involve the physical movement of different body parts, usually the hands, head, or face. Because gestures are essential in human-to-human communication, this mode of interaction has attracted significant research interest already in the 1990s in an attempt to control personal computers or provide input using gestures [40, 191, 192]. Since then, there have been many different efforts to design and develop gesture recognition methods that leverage a wide range of different sensors.

*Gestural Interactions.* One early approach to the gesture recognition problem was to use a data glove that provided angle measurements of the finger joints [61, 229], which was cumbersome due to the special-purpose glove. Video-based approaches [191, 192] (Figure 1.1), use a camera to monitor the users’ hand and apply computer vision techniques to detect either static gestures (e.g. a specific hand pose) or dynamic gestures, which are a sequence of static gestures. Such methods that rely on cameras have progressed significantly in recent years and have been deployed even on smart watches [202, 203]. WatchMe [215] is also a camera-based system that can track a pen or a laser pointer on a drawing canvas and apply it as an input modality. Other approaches used infrared proximity sensors to recognise in-air hand gestures performed over mobile or wearable devices (GestureWatch [115] and HoverFlow [119]). Ultrasound signals can identify

![Figure 1.1](image.png)

Figure 1.1. An example of an early video-based gesture recognition system. The method uses histograms of local orientation as feature vectors for gesture classification. Figure source: Freeman and Roth [63]. Copyright © Mitsubishi Electric Research Laboratories, Inc., 1994. Note: the original image has been touched for improved readability.
nearby movements [65] and radio waves can be utilised to sense human activities and vital signs [57]. Electromyography (EMG) with force-sensitive sensors is another alternative for gesture recognition that measures the electric potential generated by the muscles (e.g. EMPress [149]). Among wearable devices, the most popular approach is to record data from the motion sensors and use different classification methods such as Hidden Markov Models [180], Support Vector Machines [234], Neural Networks [127], or Dynamic Time Warping (DTW) [131]. Several prior works exist that provide a comprehensive overview of the different methods to recognise gestures [124, 154, 185, 236].

Being able to robustly detect gestures enables several practical applications. ArmTrack [198] is a system that used the motion and magnetic sensors from a smartwatch to track the users’ entire arm in 3D. MoLe [223] proposed a method to detect information leaks, i.e. what the user is typing on a keyboard using off-the-shelf smartwatches. Fine finger gestures like pinching, tapping, or rubbing can also be recognised from the built-in motion sensors, as shown by [230]. Zhang et al. further extend the gesture interaction space around commodity smartwatches by enabling tap and swipe gestures around the bezel or the band of a watch [246]. B2B-Swipe is a system that enables single-finger swipe gestures using the bezel of a smartwatch [121]. Most existing works that recognise gestures are centred around single devices, however, they lay the foundation for our work in which we use multiple collocated cooperating mobile and wearable devices.

Multi-user and multi-device interactions. Social interactions make up an important part of our lives and through technology we can interact with people from all over the world. However, most of our current interactions do not change when people are close by or far away from each other (e.g. when sharing a photo between friends). Because of this, several works have tried to redefine interactions when multiple users or devices are collocated. Fotoswipe [260] is one system which tries to reinvent this information exchange. Users can exchange photos between two devices next to each other through a finger swipe. Alternatively, short text messages can be exchanged by doing a fist bump or handshake (e.g. Hallmark TextBands), or secret messages can be exchanged by drawing shapes on a smartphone screen [270]. PathSync [33] aims to enable multi-user gestural interactions. Users must replicate a screen-presented pattern (e.g. a moving target following the perimeter of a rectangle) with their hand. One important drawback of this approach is having to augment existing objects with such moving targets, which is not always feasible. Hinckley [75] proposes “bumping” as a synchronous coupling gesture, but this is limited to only two devices. Smartphones or embedded systems can be associated through shaking patterns [78]. SurfaceLink [67] is a systems that pairs devices given a specific gesture, based on audio signals, however, devices must be placed on the same shared surface. All of the above examples implicitly assume that the devices or users are physically collocated. For implicit natural interactions, systems have to be able to
detect user and device proximity. But why does proximity play an important role? And how can we detect proximity on mobile and wearable devices?

**The Theory of Proxemics.** Edward Hall was among the first to establish the theory of proxemics and has shown that people interact differently when interpersonal distance varies [70, 142] (Figure 1.2). Moreover, Greenberg et al. demonstrated the importance of proximity as an additional contextual cue in a world of natural user interfaces [68]. The Proximity Toolkit [141] supplies fine-grained proxemic information between people and digital devices. Micro-mobility and F-formations are two constructs that help explore cross-device interactions [143]. Such constructs enable users to discuss digital artefacts by abstracting away the underlying technology, but it requires some instrumentation of the environment such as motion sensors or depth cameras. Gradual engagement, proposed by Marquardt et al. [140], defines connectivity and information exchange capabilities as a function of inter-device proximity. A fundamental requirement for all these systems is to be able to understand the location of the devices or its users.

![Figure 1.2. Edward Hall’s theory on proxemics identified four different zones in which people interact differently based on interpersonal distance. Figure source: Marquardt et al. [142] © 2012 IEEE.](image)

Indoor localisation is a research area on its own and, similar to gesture recognition, a wide range of technologies and methods exist. When using mobile and wearable devices, one of the most common approach is to estimate the distance or distances from known landmarks with either radio or audio waves [243]. A number of prior works have relied on both Wi-Fi and Bluetooth fingerprinting for indoor localisation [23, 116, 135, 226, 238, 256], however, measuring short distances with the radio received signal strength indicator (RSSI) provides only meter-level accuracy [13]. Dead reckoning [95, 99, 207] is another option that only uses inertial sensors to infer the current position from previously known locations, however, such methods tend to drift over time. Ranging methods based on
acoustic signals are also an alternative which offer low cost and high precision at short distances. For example, one of the first systems to implement two-way acoustic ranging was BeepBeep [171]. While the problem of locating a single node in a sensor network has been tackled before, tracking complex (on-body) systems has become relevant only with the proliferation of mobile and especially wearable devices.

**Contribution to Gestural Interactions.** In this dissertation, we explore multi-user gestural interactions among people who are physically collocated. We broaden and facilitate the interface between cyber technology and people by leveraging the cooperation of wearable devices, exploring the use of advanced natural gestures, and substituting today’s use of smartphones as universal interaction devices by less explicit and less obtrusive wearable technology. Concretely, we present the following research contributions.

**Collocated Multi-user Gestural Interactions with Unmodified Wearable Devices** [20]. Many real-life scenarios can benefit from both physical proximity and natural gesture interaction. In this work, we introduce an interaction technique that enables a group of people who are physically close by to effortlessly interact using gestures with complex systems. The proximity of users and devices is detected using two-way acoustic ranging with inaudible signals, while in-air hand gestures are recognised from motion sensors. The underlying wireless communication between devices is handled over Bluetooth for scalability and extensibility. We present an overview of the interaction technique and extensive evaluations using unmodified, off-the-shelf, mobile, and wearable devices, which show the feasibility of our approach. We further demonstrate four practical application scenarios. First, **HandshakAR** [15] is a wearable AR system for effortless information sharing. If two people share the same greeting gesture and are in physical proximity, their contact information is automatically exchanged and displayed on the user’s smart glasses. We then extend this concept to multiple users and present three additional application scenarios: An interactive game for children, collaborative fitness, and a collaborative music band.

**1.2 Eye-based Human-Computer Interaction**

The process that aims to identify the point of gaze and the gaze direction is called eye tracking. Because our eyes are essential to how we perceive the world around us and eye movements lay the foundation for the visual perception process [240], eye tracking has been used broadly in research as a tool for understanding human behaviour [186, 187]. One of the first eye trackers was the apparatus developed by Alfred L. Yarbus in the 1950s (Figure 1.3). Since then, eye trackers have evolved significantly and they offer high-resolution and high-frequency tracking of the eye. For example, the Pupil eye tracker
1.2 Eye-based Human-Computer Interaction

(Figure 1.4a) is a wearable head-mounted device that has multiple cameras: A monocular or binocular setup to track the users’ eyes and cameras that record the scene from an egocentric perspective. Besides wearable eye trackers, other types of eye trackers can be attached to desktop computers and record the users’ on-screen point of regard. One such example is the Tobii 4C eye tracker (Figure 1.4b) that, nowadays, is one of the most affordable eye trackers (costs around $100) and can be readily used for gaze-based interaction. These technological advancements have made eye trackers more affordable and easily available, which has created a huge potential for attentive user interfaces, i.e. interfaces that adapt and react to human attention [218].

In general, eye tracking devices can be categorised into two [136]: Videooculography and electrooculography eye trackers. Electrooculography eye trackers follow the principle of an electrooculogram (EOG) that measures the electric signal around the eye by placing two electrodes on the skin. While early works to leverage EOG eye movements [32] were fairly intrusive, newer EOG eye trackers have been integrated into regular glasses [88]. Videooculography, on the other hand, relies on cameras to observe the eye(s). In this dissertation, we focus on video-based eye trackers both in remote and head-mounted
settings. For comprehensive reviews on different eye tracking approaches, please refer to the following works [53, 54, 79].

In this section, we turn our attention to eye tracking for interaction which, in general, can be used for many different applications. MAGIC [245] was one of the first techniques developed to eliminate mouse dragging and attach the mouse cursor to the movement of the eye. A well-known problem of gaze-based interaction is the Midas touch, which is due to the fact that our eyes are never “off”. To distinguish between intentional or accidental interactions (e.g. simply looking at an interface), a clutch mechanism is necessary. This issue has been addressed either through an additional input modality or through eye gaze gestures. One attempt to a clutch-like mechanism was to use gaze for selection and a key or mouse click to confirm the selection. More recently, Stellmach et al. [208] proposed “look & touch” which relies on commodity smartphones for touch input. Users select a target on a large display by looking at it and confirm the selection with a touch command on their smartphone. Pfeuffer and Gellersen [173] have recently showed how gaze can complement and enhance touch interactions with tablet devices. They proposed an approach where the touch input redirects to the gaze target. The eye gaze direction was captured with a Tobii EyeX eye tracker. Another alternative to clutch mechanisms are gaze gestures. Drewes and Schmidt defined a gaze gesture as consisting “of a sequence of elements, typically strokes, which are performed in a sequential time order” [52]. Such gestures have reduced cognitive load, they are easy to remember, and can be integrated with dwell time to create gaze-controlled interfaces. Gaze gestures have been applied and evaluated in different application scenarios. EyeWrite is a system that uses alphabet-like gestures for text entry [231]. Games can also benefit from gaze and gaze gestures as an additional input modality [89, 219]. Affordable and reliable eye tracking technology together with exciting novel applications has the potential to “deliver the promises” in HCI and usability research [92]. However, the need for special-purpose eye tracking equipment, even though at fairly low costs, severely hinders the wide-spread adoption of gaze-based interaction.

With recent advances in machine learning, more specifically deep learning, it has become possible to estimate the gaze direction from natural images taken, for example, by a webcam. These so-called “software” methods do not require any custom hardware and leverage the increasing computational capabilities in current computing devices together with cameras that are readily built-in in an ever-increasing number of devices. For examples, most laptops, smartphone, and tablet devices have high-resolution user-facing (some even depth) cameras that can capture the users’ appearance with high fidelity. Consequently, appearance-based gaze estimation has emerged as an area of research that aims to estimate the users’ 3D gaze direction or 2D point of regard directly from an image. In contrast to more traditional approaches that use either pupil detection and additional (infrared) illumination [130, 159], appearance-based gaze estimation methods are learning-
1.2 Eye-based Human-Computer Interaction

Figure 1.4. Two examples of current eye trackers. (a) The Pupil is a wearable-head worn eye tracker with cameras that track both the users’ eyes and the scene. (b) The Tobii 4C is one of the most affordable consumer eye trackers that can be attached to any desktop or laptop computer and readily enables gaze-based interaction.
based and leverage large amounts of data that lead to state-of-the-art performance and increased robustness to changes in illumination and appearance [118, 252, 254]. While these novel computational methods require thousands of image examples as training data and are less accurate than special-purpose eye trackers, they highlight the potential of “eye tracking for everyone” [118].

Contributions to Eye-based Human-Computer Interaction. In this dissertation we explore eye gaze as a hands-free high-bandwidth input modality and use it to augment our interaction capabilities. More specifically, we propose the following three contributions:

ubiGaze: Ubiquitous Augmented Reality Messaging Using Gaze Gestures [12]. ubiGaze is a novel AR system that enables augmenting any real-world object with invisible messages through gaze gestures. This interaction technique enables users to discreetly and effortlessly interact with everyday objects and is context and location dependent. Our prototype uses two cooperating wearable devices: a Pupil wearable eye tracker and a smartwatch. The eye tracker follows the users’ gaze, the scene camera captures distinct features from the selected object, and the smartwatch provides both input and output modalities for selecting and displaying messages. Our contribution is a detailed description of the design, the implementation of the system, and a discussion of the implications for further research.

Wearable Eye Tracker Calibration at Your Fingertips [19]. One of the main challenges that prevents wearable eye trackers from being used pervasively is the need for a tedious and time-consuming calibration routine, in which users have to collect calibration samples using specialised makers and/or the assistance of a second person. In this work, we propose fingertip calibration as a novel approach to calibrate wearable head-mounted eye trackers. Users only have to point with their finger at locations in the scene and our method automatically collects the calibration samples. We propose a computer vision pipeline to detect the user’s hand and fingertip, which indicates the user’s interest. We evaluate and compare our method to a marker-based approach and show that not only is the calibration accuracy similar, i.e. no loss in accuracy, but was also the preferred method by our study users. Moreover, our implementation of the fingertip calibration method is available as a plugin for the open-source Pupil platform.

Combining Gaze Estimation and Optical Flow for Pursuits Interaction [11]. Human gaze has a huge potential for interaction as it signals the users’ point of interest [91]. Different types of eye movements such as fixations or saccades can be used for interaction [136], however, pursuits have attracted increased research interest because it enables spontaneous calibration-free interaction. The major limitation of current pursuit-based interactive systems is the need for dedicated eye tracking equipment. In this work, we propose the first methods to detect pursuits using a single off-the-shelf RGB camera in
unconstrained remote settings. The key novelty of our method is to combine appearance-based gaze estimation and optical flow in the eye region to estimate the eye movement dynamics. Results from a 13-participant user study highlight that our method not only significantly outperforms the current state of the art, but also achieves comparable performance to a commercial eye tracker. As such, our results are significant in that they enable pursuit interaction to an ever-increasing number of devices that are readily equipped with cameras.

1.3 Quantification of Visual Attention in Mobile HCI

The study of attention has many applications in mobile HCI, from modelling user behaviour such as user engagement [144, 148], attentiveness [177], alertness [1], or boredom [178], to being fundamental in multiple tasks such as assessing user interruptibility, which is the task of finding the opportune moments to deliver messages or notifications. Moreover, models of human attention may also help reason about user intention such as forecasting where the users’ attention will be next [206]. These examples highlight the importance of understanding, studying, and quantifying user attention, which is a pillar for the next generation of attentive user interfaces [218].

While attention is highly relevant to the above tasks, most of the prior works have focused on alleviating the negative effects of fragmented attention, for example by predicting the distractiveness of mobile notifications. One reason for a lack of research to directly study user attention is that there is no common definition or understanding of attention among multiple disciplines [5]. Studying human attention and its allocation has roots in cognitive psychology, neuroscience, and computer science. But what is attention? It defines how we process information in our environment. While different definitions for attention exist, it is widely accepted that attention can be categorised as overt or covert. Covert attention is concerned with one’s mental focus irrespective of eye movements and is difficult to measure without stimuli or carefully instrumented lab studies. In contrast, overt attention, also known as visual attention, can directly be measured in terms of eye movements. Because of this, in HCI, overt attention also know as visual attention is the most practically useful measure of user attention.

In this dissertation, we investigate visual attention on mobile devices. Mobile devices are used by millions of people from all over the world and, over the years, they have become feature-rich miniaturised computers with extremely powerful computational capabilities (Figure 1.5). Because of this and because of the services or wide range of applications that these devices offer, our usage patterns have changed significantly. Already fifteen years ago, in a time when mobile devices had fewer capabilities and were used less than
today, the nature of attentional resources has become highly fragmented lasting as little as 4 seconds [166]. Others had similar conclusions by investigating the effects of disruptions on users’ tasks in mobile settings, which can severely hinder productivity [102]. Moreover, it is well known that smartphone overuse can have negative consequences to our health, leading to attention deficits, sleep deprivation, or even smartphone addiction [126]. Given the ever-increasing number of mobile devices that are being used in everyday situations and the significant changes in the way we use them, it has become imperative to study and model (visual) attention.

While there is little to no prior work to directly focus on attention [221], modelling different aspects of mobile phone usage has attracted significant research interest. One of the most prominent research areas is concerned with user interruptibility, which is the task of identifying the best moment in time to interrupt the user and deliver messages or notifications [150]. Others have looked at the consequences of interrupting users on task performance, emotional state, and social attribution [2]. Most of these approaches only consider the immediate past before an interruption or an event. Choy et al. proposed a method that looked back at the entire day [36]. Approaches that aim to predict user interruptibility on mobile devices have mostly relied on the built-in sensors and events generated by these devices. Hudson et al. tried to identify which sensors are most useful
for this task by conducting a wizard of Oz study [85]. The users’ surrounding environment, in a particular location or social context [49, 56], can also influence how users interact with their devices. A common aspect among all the above works is that they only implicitly model user attention, i.e. these methods are only proxies that relied on cumbersome and time-consuming manual annotation [166], analysis of application usage logs [98], or self-reported questionnaires collected through methods like experience sampling [214]. The main reason for this is the lack of accurate and robust methods to study attentive behaviour during everyday mobile interactions without special-purpose and obtrusive eye tracking equipment [206].

Modelling user attention and its allocation has applications beyond research lab studies. Recently, large companies have made significant efforts to provide users an in-depth view of their mobile phone “consumption”, as part of the quantified self movement [269]. For example, Apple, starting with iOS version 12, has released an application called Screen Time. This application shows users detailed analytics about their smartphone usage. Google has introduced a similar application called Digital Wellbeing. In both of these cases, in practice, the applications measure screen time (i.e. how long the screen is on) and offer a detailed breakdown of different time periods, applications, or usage contexts. However, we argue that screen time is a naive proxy for user attention and does neither fully nor accurately capture the highly dynamic nature of visual attention.

Contributions to Quantifying Mobile Visual Attention. In this dissertation, we analysed and proposed methods to understand and quantify users’ overt attentive behaviour during everyday mobile device interactions. Concretely, we present the following three contributions:

How far are we from quantifying visual attention in mobile HCI? [17]. With an ever-increasing number of mobile devices that compete for the users’ limited and as such fragmented attention, quantifying when, how often or for how long users visually attend to their mobile devices has emerged as a fundamental research problem. We leverage recent advancements in automatic eye contact detection as a means to sense users’ visual attention during everyday mobile device interactions. Furthermore, we identify key challenges associated with sensing visual attention including the impact of face and eye visibility as captured by the front-facing camera, the importance of robust head pose estimation, and last but not least the need for accurate gaze estimation. Based on our analysis, we propose concrete future research directions that lay the foundation for the next generation of attentive user interfaces.

Accurate and Robust Eye Contact Detection During Everyday Mobile Device Interactions [16]. We present a novel method to detect users’ eye contact with their mobile devices in images recorded using the device-integrated front-facing camera, for example,
in the context of digital wellbeing or quantified self applications. Eye contact is a key measure of overt visual attention and an important source of information in mobile HCI as it enables quantifying when, how often, or for how long users attend to their devices. Based on key insights from our prior work [17], our method addresses fundamental challenges of automatic eye contact detection that are characteristic to natural mobile device use: Face and facial landmark detection including partial occlusions, extreme head poses, and variability across users, devices, and illumination conditions. In-depth evaluations on two real-world, mobile interaction datasets show significant performance improvements and increased robustness over the state of the art. Using a proof-of-concept implementation, we further show how our method can directly be used as a tool to extract and calculate higher-level attention metrics and, as such, represents a significant step towards continuous monitoring of visual attention in mobile HCI.

Quantification of Users’ Visual Attention During Everyday Mobile Device Interactions [18]. In this work, we present the first real-world dataset and quantitative evaluations of visual attention of mobile device users in-situ, i.e. while using their devices during everyday routine. We present the Everyday Mobile Visual Attention (EMVA) dataset - a new 32-participant dataset that contains over 472 hours of video recordings as well as associated metadata, sensor data, device usage logs, device events, and location data. Using our prior work on eye contact detection in mobile settings [16], we are first to quantify the highly dynamic nature of everyday visual attention across users, mobile applications, and usage contexts. Furthermore, we disseminate the key findings from our analyses that not only highlight the potential but also inform the design of the future mobile attentive user interfaces. To support further research in this area, the EMVA is publicly available.

1.4 Dissertation Roadmap

The chapters of this dissertation are organised around the three main contribution areas.

In Chapter 2, we discuss collocated multi-user gestural interactions. We first present the architecture and design of the interaction technique and describe each of the three enabling components in detail: Communication over Bluetooth, inaudible acoustic ranging, and gesture recognition via motion sensors. We then present several evaluations that highlight the feasibility of our approach. We also present four real-world application scenarios and conclude with a discussion on our results.

Chapter 3 presents three different contributions in the area of eye-based HCI. First, in Section 3.2, we present ubiGaze, a novel wearable AR system that enables augmenting any
real-world object with invisible messages through gaze gestures. We describe the concept, an implementation, and conclude with a discussion on implications for research. Second, in Section 3.3, we present the fingertip calibration method for regression-based wearable eye trackers. We describe the method in detail and then present several evaluations that address the performance and usability of our approach. Third, in Section 3.4, we present a novel method to detect smooth pursuit eye movements using a single RGB camera. We first present an overview of our approach. We then evaluate our method on a 13-participant dataset and report the results from our experiments. Lastly, we discuss our findings and present opportunities for future work.

Chapter 4 presents our work on quantifying mobile visual attention. We first identify key challenges for sensing visual attention using the front-facing camera of mobile devices (Section 4.3). We then propose a novel method to accurately and robustly detect eye contact with mobile devices, i.e., when users look at their devices (Section 4.4). In Section 4.5, we present the Everyday Mobile Visual Attention dataset, how it was collected and its most important characteristics. Lastly, in Section 4.6, we analyse visual attention on the EMVA dataset and report our findings. We conclude this chapter with a discussion on our results.

We conclude this dissertation in Chapter 5. We discuss the significance of our findings and provide a basis for future research directions.

1.5 Statement of Contributions

The contributions presented in this thesis were, in part, done by other people. Unless otherwise mentioned, the work presented in this thesis is the original work of the author. We present a statement on the contributions below:

Chapter 2. The methods and the concepts that enable multi-user gestural interaction are original work of the author, who is also the main contributor to this project. Sander Staal helped with the implementation and evaluations of our approach as part of his Bachelor’s thesis [204]. He also implemented three prototypes that leverage this novel interaction technique. The implementation of HandshakAR is the contribution of the author of this dissertation. Sander Staal has received co-authorship in our papers for his contribution [15, 20].

Chapter 3, Section 3.2. The concept and prototype implementation of ubiGaze is joint work with Teemu Leppänen, Argenis Ramirez Gomez, and David Gil de Gomez. Parts of this work originated at the Eyework workshop during the UBI Summer School in 2016, Oulu, Finland. Hans Gellersen, Eduardo Velloso, and Gabor Sörös provided feedback on
this work. The author of this dissertation was the main contributor for this publication. Teemu Leppänen, Argenis Ramirez Gomez, and David Gil de Gomez have received co-authorship for their contribution [12].

Chapter 3, Section 3.3. The fingertip calibration method is original work of the author. Sander Staal helped with the implementation and with running the experiments for evaluation as part of his Distributed Systems Lab project. He received paper co-authorship for his contribution [19].

Chapter 3, Section 3.4. This project is joint work with Vincent Becker and Chenyang Wang. The idea to explore pursuit interactions by jointly analysing gaze estimates and optical flow is the contribution of the author of this dissertation. All three authors contributed to the implementation and the design of the published method. The dataset used for evaluation was collected by Alexander Kayed, as part of his Master’s thesis [106]. Vincent Becker and Chenyang Wang share first co-authorship with the author of this dissertation for their contributions [11].

Chapter 4. The idea to study mobile visual attention originated from a discussion with Andreas Bulling. The author of this dissertation is the main contributor in this project. Sander Staal helped with the implementation and evaluations as part of his Master’s thesis. For his contributions, Sander Staal has received paper co-authorship [16, 17, 18].

In all of the contributions presented in this thesis, all the co-authors of the published papers gave feedback and contributed to the concepts and evaluations presented in the papers.

1.6 Accompanying Publications

The results presented in this work have been published in the following publications that form the basis of this dissertation. The list of papers is presented in reverse chronological order.

* Denotes equal contribution.

1.6 Accompanying Publications


Other publications by the author that are not part of this thesis:


This chapter is based on the following publications:


2.1 Introduction

Mobile and wearable devices have become wide-spread personal companions for many different everyday activities, from tracking physical exercise, enabling communication, gaming, storytelling, to controlling appliances, and many more. At the same time, social interactions and group activities are an important part of our lives and technology allows us to interact with people from all over the world. While many of the activities enabled by these novel computing platforms have a strong social component, the social interaction experience is mostly virtual. A virtual interaction happens through the screen and actuators of our devices, with smartphones being used as universal interaction devices [147]. Moreover, physical proximity between the users is rarely taken into account. For example, the experience of sharing a photo does not change when two users are nearby or in different corners of the world.

Figure 2.1. Interaction paradigm: Three children can interact and open a smart treasure chest if they simultaneously perform the same hand gesture and are close enough to it. This application is a combination of collective gesture recognition, proximity detection, and data exchange among the different devices involved. The treasure chest is a metaphor for any smart object or device.

In this chapter, we explore multi-user interactions between a group of people who are physically close to one another. Our work is inspired by the theory of proxemics established by Edward Hall, which studies the way people mediate their interactions with other people around them [70]. While this theory covers different dimensions, one that
certainly influences and leads to higher interaction engagement is interpersonal distance. Consider the following scenario: Three children are in a theme park and they want to interact with a treasure chest as illustrated in Figure 2.1. For the chest to open, the three children have to be in front of the chest and all of them have to perform the same interaction (e.g. a hand gesture), roughly at the same time. The requirements for this scenario are three-fold: Physical proximity, collaboration between users and devices, and interaction through hand gestures. The treasure chest in our example is just an abstraction for any smart object or device [145] and could be extended to toys, public displays, or other applications.

Our proposed interaction technique builds on existing methods for gesture recognition and proximity detection to enable collocated multi-user gestural interactions. We leverage only standard features of off-the-shelf mobile and wearable devices to recognise gestures and to detect physical proximity: Built-in motion sensors and inaudible acoustic signals are used for an unobtrusive and seamless interaction. The underlying communication is handled over Bluetooth for scalability and extensibility.

To our knowledge, we present the first collocated multi-user gestural interaction technique that uses acoustic ranging and runs directly on mobile devices. We assess the building blocks of this technique in different environments on unmodified, off-the-shelf hardware and give practical recommendations that lead to robust real-world performance. To explore the design space of this interaction technique, we first present HandshakAR, a novel AR application that enables effortless and natural information sharing through hand gestures between two users. We then extend this concept and further demonstrate three additional application scenarios with multiple users: Collaborative fitness, a collaborative music band, and an interactive game for children.

2.2 Related Work

In this section, we discuss relevant prior work that concerns the different dimensions of the proposed multi-user gestural interaction technique. First, we review systems that enable and support multi-user interactions and collaborations. We then focus on proxemics, i.e. systems that use physical proximity as a component in ubicomp applications and methods to detect proximity. Lastly, we review approaches for recognising gestures.
Chapter 2 Collocated Multi-user Gestural Interactions

2.2.1 Multi-user and Multi-device Interactions

The proliferation of affordable smart devices resulted in consumers owning and carrying multiple connected devices with them. Multi-user collaboration scenarios require devices to communicate with each other. Embedded systems or smartphones can be paired through simultaneous shaking patterns [78, 120], simultaneous pressing of a button [189], bumping gestures [75], touch gestures spanning multiple displays [76], or pinching gestures [134]. SurfaceLink [67] is a system that associates devices given a specific gesture. This is an audio-based grouping, but it is limited to devices that must share the same surface. An overview of different ways to connect devices is given by Jokela et al. [97]. Most of these systems focus on the initial pairing problem, i.e. how to connect devices for the first time. In contrast, in our work, we support and enable continuous interaction between devices and, hence, its users.

The problem of designing and developing such cross-device systems is still subject to research [35, 80]. HuddleLamp [183] is a desk lamp with an integrated depth camera that enables multi-user and multi-device interaction around a tabletop. Pass-them-around [133] is a collaborative photo sharing application. A system which aims to enable multi-user gestural interaction is PathSync [33]. To interact with digital objects, users must replicate a screen-presented pattern (e.g. a moving target around the edges of a rectangle) with their hand. This approach is similar to pursuit-based interaction that leverages smooth pursuit eye movements [55]. However, it suffers from the same limitation that real objects have to be augmented with moving patterns which is not always feasible.

A system that is most similar to ours is Tracko [96]. Users’ devices are located in 3D space using audio signals and cross-device interactions are supported with touch gestures on the devices’ display. The fundamental difference between Tracko and our work is that our interaction technique is not limited to the device’s display and it supports in-air hand gestures.

In this work, our initial efforts are to propose collocated gestural interactions between two users. With HandshakAR [15], two users can effortlessly exchange information when they perform the same greeting gesture and are close to each other. We then extend this proposal to multiple users and explore the design space with three application scenarios.

2.2.2 Collocated Interactions

Or why is physical proximity relevant? Proxemic interactions have great potential to enable natural user interfaces [68]. Systems can learn to take advantage of people and devices as they move towards one another. Marquardt et al. introduced a system that explored cross-
2.2 Related Work

device interaction using two different constructs: Micro-mobility and F-formations [143]. F-formations describe the relationships between people, while micro-mobility describes devices in terms of position and orientation. While these methods require extensive hardware instrumentation of the environment, which is a combination of motion sensors, radio modules and over-head depth cameras, the system enabled users to collaborate and seamlessly discuss digital artefacts without focusing on the underlying technology. Gradual engagement is the concept that connectivity and information exchange capabilities are exposed as a function of inter-device proximity [140]. This also requires a hardware infrastructure in the environment, proximity is detected using a motion capture system that uses infrared sensors. The Proximity Toolkit [141] supplies fine-grained proxemic information between people and digital devices. The toolkit gathers data from various hardware sensors and transforms them into rich high-level proxemic information, however, this information cannot be used to infer the proximity of off-the-shelf mobile and wearable devices. Therefore, in the following, we briefly discuss methods to spatially locate such devices.

**Proximity detection.** There are different methods for device localisation both indoors and outdoors. In outdoor spaces, a common approach is to use the Global Positioning System (GPS), however, while good for navigation, the error is generally within a few meters [167], which is not enough to distinguish between users who are close by and want to interact or bystanders who may be nearby. In indoor spaces, most methods estimate the distance or distances from known landmark(s) with the help of audio or radio waves [243]. A number of prior works have relied on both Wi-Fi and Bluetooth fingerprinting for indoor localisation [23, 116, 135, 226, 238, 256], however, measuring short distances with the radio received signal strength indicator (RSSI) provides only meter-level accuracy [13]. Dead reckoning [95, 99, 207], on the other hand, only uses inertial or motion sensors to infer the current position from previously known locations, however, such methods tend to drift over time. Given our requirements for wide-scale availability, low cost, and high precision at short distances, we focus on ranging methods that use acoustic signals.

Tracko [96] is an acoustic tracking system that leverages both Bluetooth Low Energy (BLE) and acoustic signals to locate devices in 3D. The system is accurate for distances up to 1.5 m, but it requires devices to support BLE peripheral mode. This feature is not yet widely implemented on current smartphones or smartwatches and is manufacturer dependent. Another approach based on acoustic signals is two-way ranging, which enables devices to estimate the distance relative to one another. For example, Microsoft’s BeepBeep systems [170, 171] and several follow-ups that improved the original method [38, 157, 181]. In our work, two-way acoustic ranging also lays the basis for our interaction technique.
2.2.3 Gestural Interaction with Wearables

Gesture recognition from motion sensors has been comprehensively investigated in the past. The recognition methods usually rely on data from inertial measurement units (IMUs) because such sensors are ubiquitous and available on many if not all devices. Different classification approaches like Hidden Markov Models [180], Artificial Neural Networks [127], Support Vector Machines [234], or Dynamic Time Warping (DTW) [131] can be used to discriminate between different gestures.

Our interaction technique leverages wearable devices that can capture gestural interactions. ArmTrack [198] is a system that used the motion and magnetic sensors from a smartwatch to track the users’ entire arm in 3D. Motion Leaks (in short, MoLe) [223] proposed a method to detect information leaks, i.e. what the user is typing on a regular keyboard using off-the-shelf smartwatches. Arduser et al. [7] explored a similar idea to recognise what users are writing with a smartwatch. However, their approach was aimed at text written on a whiteboard. Subtle finger gestures like pinching, tapping, or rubbing can also be recognised from the built-in motion sensors, as shown by Wen et al. [230]. Zhang et al. further extended the gesture interaction space around commodity smartwatches by enabling tap and swipe gestures around the bezel or the band of a watch [246].

Recognising gestures is not tied to motion sensors only. Fu et al. [65] have proposed a system where nearby movements can be recognised from sound. BodyScan [57] is a method that provides continues sensing of the users’ entire body, including vital signs from radio signals. GestureWatch [115] and HoverFlow [119] are both systems that recognise in-air hand gestures performed over mobile or wearable devices relying on infrared proximity sensors. Hand gestures can also be recognised from a single RGB camera found on most mobile devices [202, 203]. WatchMe [215] is a camera-based system that can track a pen or a laser pointer on a drawing canvas and use this as an input modality. Electromyography (EMG) with force-sensitive sensors is an alternative to detect hand gestures. EMPress [149] shows how the best arm position of EMG-based systems is the forearm, meaning that such system cannot be efficiently used with off-the-shelf smartwatches.

The above list is a selection of existing systems that hint at how non-touchscreen gestures, in general [8], can extend the input space of wearables. In our work, we are first to propose a gestural interaction technique among multiple users who are physically close to one another while detecting proximity using inaudible acoustic signals. In the following section, we describe the building blocks of our approach.
2.3 Collocated Multi-User Gestural Interactions

2.3.1 Overview

Figure 2.2 shows an overview of the different layers required to enable multi-user gestural interactions. The generic *Communication* layer interconnects and enables interoperability between many different devices using Bluetooth as the communication protocol. The *Proximity Detection* layer estimates the pairwise distances between people and implicitly devices via non-intrusive inaudible acoustic ranging. The *Gesture Recognition* layer detects hand movements and identifies gestures using the motion sensors of smartphones or wearables. The *Application* layer can then use all three fundamental layers for different application or interaction scenarios.

Figure 2.2. Overview of the components that enable collocated multi-user gestural interactions. The *Communication* layer enables devices to exchange data via Bluetooth. The *Proximity Detection* layer estimates distances between devices using inaudible acoustic signals. The *Gesture Recognition* layer uses the motion sensors to identify gestures, i.e. hand or body movements in our case. Finally, the *Application* layer takes advantage of all the above components to enable the proposed interaction technique.

2.3.2 Communication via Bluetooth

Enabling multi-user interaction through wearables requires data exchange between devices. Most modern wearables, whether smartwatches or smart glasses, rely on a companion
smartphone and are used mainly to display calls, show calendar entries, and other types of notifications. The current trend of manufacturers, however, is to switch towards a watch-centric approach in which smartwatches become fully independent computing platforms. Current devices offer a wide range of communication capabilities and methods such as Bluetooth, Wi-Fi, Wi-Fi direct, or near-field communication (NFC).

Taking power consumption into account, the targeted distance ranges, and the support for interoperability between a wide variety of different devices, Bluetooth was the best fit for our approach. Bluetooth networks are, in general, classical client-server architectures with one device or server in the middle. In our prototypes, one of the devices also acted as the server for the Bluetooth network. This creates a vulnerability, in that, if the single server crashes, the communication between the other devices is also lost. In the future, multiple Bluetooth piconets could be connected together, forming mesh networks and thus addressing this limitation.

### 2.3.3 Proximity Detection via Acoustic Ranging

The proximity layer is based on two-way acoustic ranging, which is a cooperative method to estimate the distance between two smart devices. The main advantage of two-way ranging is that the devices do not need to be synchronised [171], which greatly simplifies the protocols. With two-way ranging, both devices have to emit an acoustic signals and both have to, at the same time, record not only their own signal but also the signal produced by the other device. The distance estimate is based on measuring the time elapsed between the two received audio signals.

#### Distance Estimation

Figure 2.3 illustrates the ranging procedure between two devices:

1. Device A emits a signal at time \( t_A \). A records its own signal at time \( t_{A,A} \).
2. Device B records A’s signal at time \( t_{A,B} \).
3. Device B emits a signal at time \( t_B \). B records its own signal at time \( t_{B,B} \).
4. Device A records B’s signal at time \( t_{B,A} \).

To calculate the distance \( D \) (Equation 2.1) between devices A and B, several distance measures are necessary. In general, any distance \( d \) can be calculated as \( d = \text{speed} \times \text{time} \). We denote the distance \( d_{X,Y} \) as the distance between X’s speaker and Y’s microphone. For
Example, $d_{X,X}$, which represents the distance between X’s speaker and its own microphone, can be calculated as $d_{X,X} = c \cdot (t_{X,X} - t_{X})$, i.e. the speed of sound multiplied by the difference between the time when the signal was recorded by X’s microphone and the time when the signal was first played by X’s speaker.

The distance $D$ between A and B is given by the round trip time:

$$D = \frac{1}{2} (d_{A,B} + d_{B,A}) \quad (2.1)$$

Using the above formula for distances, $d_{A,B} = c \cdot (t_{A,B} - t_{A})$ and $d_{B,A} = c \cdot (t_{B,A} - t_{A})$. Similarly, $d_{A,A} = c \cdot (t_{A,A} - t_{A})$ and $d_{B,B} = c \cdot (t_{B,B} - t_{B})$. In practice, distances such as $d_{A,A}$ and $d_{B,B}$ are device-dependent constants, which can either be obtained by measuring the distance between the speaker and microphone of a given device or, if available, from the manufacturer. By expanding Equation 2.1 using the above calculations, we obtain Equation 2.2 that only requires $t_{A,B}, t_{B,A}, t_{A,A}, t_{B,B},$ and the two device-dependent constants $d_{A,A}$ and $d_{B,B}$. For more details on these calculations, please consult the original paper [171].

$$D = \frac{c}{2} \cdot ((t_{B,A} - t_{A,A}) - (t_{B,B} - t_{A,B})) + \frac{1}{2} \cdot (d_{A,A} + d_{B,B}) \quad (2.2)$$

![Figure 2.3. Two-way acoustic ranging between devices A and B: First A emits an audio signal, then B. Devices have to record throughout the entire ranging procedure to ensure that both devices can hear each other’s audio signal. The distance between them is estimated using the Time Difference of Arrival (TDOA). Figure source: [20, 204]](image)
This method requires the devices to have a microphone, a speaker, and an underlying communication framework for exchanging messages and coordinating the transmission of the audio signal, in our case using Bluetooth. The microphones must record sounds through the entire process and speakers have to emit signals sequentially: First device A, then device B.

**Reference Audio Signal**

The reference audio signal must have a strong autocorrelation property that ensures robustness against ambient noise. For example, both Peng et al. [171] and Curtis et al. [38] used linear chirps. In Tracko [38], the authors propose bi-orthogonal chirps that additionally encoded both the device and speaker ID.

In our work, similarly to Curtis et al. [38] who also experimented on Android devices, we opted to use a linear chirp signal with a Gaussian envelope. The maximum sampling rate in Android [96] and in our implementation was 48 kHz for both the microphone and the speaker. With such a sampling rate, we can reconstruct signals that contain frequencies of up to 24 kHz (Nyquist frequency). The aliasing of higher frequencies can be avoided with a low-pass filter.

Two important aspects that affect the method’s robustness are the signal length and the frequency bandwidth of the chirp. The ranging method described in this work works with both audible and inaudible signals. Nevertheless, choosing a signal that is optimal for this interaction technique is not trivial. Moreover, Peng et al. [171] also mentioned that “optimal design of the ranging signal” was left for future work. While audible signals work better in practice, they are disturbing to people. Alternatively, we fill this gap and focus on frequencies between 20 kHz and 24 kHz, since this is above the acceptable range for human hearing, which is between 20 Hz and 20 kHz [190]. In practice, we selected a signal length of 75 ms.

**Signal Detection**

The reference acoustic signal played and recorded by the devices can be detected using cross-correlation. The computational complexity of calculating the correlation can be reduced from $O(n^2)$ to $O(n \times \log(n))$ using the Fast Fourier Transform (FFT). In our prototype implementations, we calculated the FFT and the inverse FFT using the JTransforms Java library [264]. The distance estimation method requires peak detection in the cross-correlation function with sub-sample accuracy. Missing the correct sample number even by one can already negatively influence the result by 0.7 cm in the final distance.
2.3 Collocated Multi-User Gestural Interactions

estimate. This error considers the 48 kHz sampling rate and the speed of sound. To find the correlation peaks, we use the method proposed by Peng et al. [171].

2.3.4 Gesture Recognition via Motion Sensors

There are many ways to recognise hand gestures using unmodified mobile and wearable devices (as seen in Subsection 2.2.3). We use a method based on Dynamic Time Warping (DTW) on measurements collected from a single three-axis accelerometer. We follow a two-step approach when recognising gestures. We first quantised the acceleration measurements and then we match new samples to existing ones using DTW.

Figure 2.4. (1) The accelerometer records a new sample $S$. (2) The sample gets quantised into $S'$. (3) $S'$ is matched against every template $T$ from the template library. The matching cost is given by the similarity between $S'$ and template $T_k$, which is calculated using DTW. (4) A gesture is recognised by selecting the pair with the lowest cost. Figure source: [20, 204]

Mobile and wearable devices have limited computational capabilities. Consequently, reducing the amount of data to be processed is beneficial as it can improve not only the runtime but also the performance of the method. In our implementation, quantisation is achieved through a moving average with a windows size $w$ and step size $v$. Choosing the right parameters for quantisation may improve the performance of the method by eliminating accelerometer noise [131], however, it could also harm the performance of the method by removing essential features. In the prototype applications, based on evaluations presented later in this chapter (Table 2.1), the parameters were set to $w = 250$ ms and $v = 200$ ms.
Gesture recognition using DTW works by building a predefined template dictionary, where at least one template per gesture has to be stored. Gestures are recognised by comparing a newly collected time series with all the samples from the existing template dictionary and then selecting the pair with the lowest matching cost (Figure 2.4). DTW returns a score for each pair, therefore it is more than just a binary classification. Candidates are ranked based on the matching cost.

2.4 Evaluation

In this section, we present a thorough evaluation of the individual components that enable the proposed interaction technique. Our goal is to enable multi-user collocated interactions using unmodified mobile and wearable devices. In our experiments, we use two smartphones (LG G3 and LG Nexus 5X) and two smartwatches (Sony Smartwatch 3 and Motorola 360 Sport 2nd generation). Our evaluation discards the communication layer because Bluetooth is an already established wireless standard and focuses on the remaining two components: Gesture recognition and acoustic ranging.

2.4.1 Gesture Recognition

Figure 2.5. Gesture set 1: Eight geometric gestures. Initially proposed by Kela et al. [107] and reduced to eight gestures in uWave [131], which are the ones we used in one of our evaluations. The dot indicates the start of a hand gesture, while the arrow its end. Figure source: [131]
The gesture recognition component uses template matching based on DTW. To evaluate the recognition accuracy, we tested this method on two different gesture sets: One based on geometric shapes (Figure 2.5, initially introduced by Kela et al. [107], then reused by Liu et al. [131]), while the other represents a set of human-to-human gestures (Figure 2.6). The latter is a small, well-known collection of greeting gestures from different cultures, but it is challenging enough to make simple heuristic approaches fail. To the best of our knowledge, this particular collection of greeting gestures is an original proposal.

Since the geometric gestures have already been evaluated [131], we focus on the newly introduced greeting gestures. For every gesture, we collected 20 samples from 5 participants using a Sony Smartwatch 3, which makes a total of 800 samples. We tested our dataset under two different conditions: user-dependent and user-independent. In the user-dependent case, a test sample is matched against all the other samples, including those belonging to the same user and the same gesture class. In the user-independent case, a test sample cannot be matched to another sample that belongs to the same user and same gesture class.

The best recognition accuracy is achieved with a multi-dimensional DTW algorithm. The distance measure represents the cumulative distances of the three dimensions (x, y, and z) measured independently using DTW. Figure 2.7 shows the confusion matrix for the user-dependent case. The recognition accuracy was around 97%, with Precision = 0.9739, Recall = 0.9736, and $F_1 = 0.9736$. For the user-independent case (Figure 2.8), the
Figure 2.7. Confusion matrix for **user-dependent** gesture recognition evaluation. Average accuracy 97%.

recognition accuracy drops significantly. The accuracy was around 53%, with $\text{Precision} = 0.5372$, $\text{Recall} = 0.5389$, and $F_1 = 0.5354$. These results, while worse than in the user-dependent case, are not surprising since a sample belonging to one user and gesture class can only be matched to samples belonging to different users. Moreover, our study participants were not instructed to perform those eight greeting gestures in a particular way. More freedom to perform those gestures led to more variance in the data.

The above experiments highlight that DTW works well for user-dependent gesture recognition. If we further limit the scope of DTW to only the samples belonging to the same user and average the per-user results, the recognition accuracy is close to 99%.

Data quantisation and compression are essential in gesture recognition methods that use DTW. In this experiment, we identified the best combination of window size and step size and their influence on the method’s performance and runtime. For every window size, we tested multiple step sizes. However, we only present the optimal one, i.e. the one that leads to the highest accuracy. Results show that the accuracy of the method decreases both when the window size is too large or too small. For example, a window size of 2 s, independent of the step size, will lead to an accuracy of at most 68%. This is also a consequence of the gesture set we used, which are fairly short and quick to perform.
Figure 2.8. Confusion matrix for **user-independent** gesture recognition evaluation. Average accuracy 53%. The drop in performance can be explained, in part, by the differences between users and how they performed the greeting gestures. During data collection, the study participants were not instructed to perform the gestures in a particular way.

Table 2.1 summarises the runtime of the optimal parameters, which is a combination of window size and step size. We measured the time needed to recognise one gesture, i.e. to find the closest match of a new sample to one from the template dictionary. The runtime only shows the time for the DTW algorithm, excluding the quantisation of the data. This evaluation was performed on a PC with a Core 2 Quad CPU@2.4 GHz. Results show that the optimal window size, which is a trade-off between accuracy and runtime, is 250 ms, whereas the optimal step size is 200 ms. The optimal runtime is about 7 ms. Performing the same experiment on the smartwatch resulted in a runtime of about 700 ms. While the difference is significant, wearables are becoming ever-more powerful, and this gap between the current generation of wearables and personal computers will become narrower as computing technology constantly progresses.
Chapter 2 Collocated Multi-user Gestural Interactions

<table>
<thead>
<tr>
<th>Window size (ms)</th>
<th>Optimal step size (ms)</th>
<th>Accuracy (%)</th>
<th>Runtime (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>100</td>
<td>92.5</td>
<td>20.8</td>
</tr>
<tr>
<td><strong>250</strong></td>
<td><strong>200</strong></td>
<td><strong>92.5</strong></td>
<td><strong>7.74</strong></td>
</tr>
<tr>
<td>80</td>
<td>50</td>
<td>92.5</td>
<td>96.71</td>
</tr>
<tr>
<td>100</td>
<td>30</td>
<td>91.25</td>
<td>248.41</td>
</tr>
<tr>
<td>50</td>
<td>30</td>
<td>91.25</td>
<td>257.88</td>
</tr>
<tr>
<td>25</td>
<td>20</td>
<td>90</td>
<td>514.32</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>90</td>
<td>1500.88</td>
</tr>
<tr>
<td>1000</td>
<td>200</td>
<td>88.75</td>
<td>4.94</td>
</tr>
<tr>
<td>2000</td>
<td>150</td>
<td>67.5</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 2.1. An overview of the gesture recognition performance for different window sizes and step sizes. Only the optimal step size for each window size is presented in this table. The results are computed on the first gesture set (Figure 2.5). A good trade-off between performance and runtime is given by a window of 250 ms and a step size of 200 ms. Results are sorted in decreasing order of accuracy.

### 2.4.2 Acoustic Ranging

In our work, we use inaudible acoustic chirp signals since they are non intrusive for the human ear. Given such signals, the maximum distance that we were able to measure using acoustic ranging was around 23 m. However, since our focus is on close proximity multi-user scenarios, we evaluated the ranging component at distances of up to 5 m.

#### Hardware Sensitivity Analysis

Speakers and microphones found on consumer mobile devices are designed to reproduce and record human voice mainly; thus, when using inaudible signals, i.e. signals in the ultrasound spectrum, they can show different sensitivity. Figure 2.9 shows four frequency response plots from two smartphones and two smartwatches while recording a linear chirp, starting at 200 Hz and going up to 24 kHz. All four devices were placed at the same position and location relative to the loudspeaker. For each experiment, we used the same loudspeaker. Figure 2.9 shows that the two smartphones, the LG G3 and the LG Nexus 5X, are better at capturing the audio chirp until around 10 kHz when their sensitivity decreases significantly. The smartwatches, on the other hand, are worse, in particular the Motorola 360 smartwatch whose sensitivity decreases significantly after around 8 kHz.

Two-way ranging requires both a microphone and a loudspeaker. Similarly, if we want to capture sounds in the inaudible spectrum, we need speakers that can reliably
2.4 Evaluation

Figure 2.9. *Microphone sensitivity - frequency response.* We used a loudspeaker to play a linear chirp, starting at 200 Hz and going up to 24 kHz, which was recorded by the microphones of two different device classes: two smartwatches at the top and two smartphones at the bottom. These measurements show that the sensitivity of the microphones decreases in the ultrasound spectrum since most of these devices were designed to handle human voice. Figure source: [20, 204]

play such audio signals. In this experiment, we, therefore, tested the sensitivity of the loudspeakers. The results only illustrate the two smartphones because the smartwatches we had in our laboratory did not have speakers. We played the same chirp as before, starting at 200 Hz and going up to 24 kHz, using each of the two smartphones. We then recorded the audio signal using a high-quality Neumann TLM 102 microphone. In contrast to the microphones, the speakers did not vary much among the two smartphones (Figure 2.10). However, we can also observe a significant decrease in sensitivity in the ultrasound spectrum.
Two-way Acoustic Ranging Performance

Most smartwatches nowadays are only equipped with a microphone for voice commands, which can reliably capture audio signals up to 4 kHz. Due to the lack of loudspeakers on our smartwatch models, we were unable to test how well acoustic ranging works on these wearable devices. Instead, in this evaluation, we used the two smartphone models available in our laboratory: the LG G3 and the LG Nexus 5X.

We evaluated the performance of the acoustic ranging method at six distances between the two phones: 0.1, 1, 2, 3, 4, and 5 m. The two factors that we investigated were the distance measurement error and the number of successful and unsuccessful measurements, i.e. the number of cases when the distance could not be measured due to the reference audio chirp not being detected. We also tested multiple longer and shorter frequency bandwidths in both the audible (500–2000 Hz, 2000–6000 Hz) but mostly inaudible spectrum (20’000–22’000 Hz, 21’000–22’000 Hz, 21’000–21’700 Hz, 21’000–21’500 Hz, 21’000–22’250 Hz). For each configuration of parameters (distance and frequency bandwidth), we collected data for five minutes, which is, on average, 104.5 measurements (SD=20.6). The average number of successful measurements varied with distance: 112.7 (SD=3.5) at 0.1 m, 111.9 (SD=4.1) at 1 m, 107 (SD=5.8) at 2 m, 108 (SD=9.2) at 3 m, 101.4 (SD=21.5) at 4 m, and 85.9 (SD=37.9) at 5 m. Figure 2.11 shows the outcome of this experiment. Shorter frequency bandwidths (e.g. 250 Hz or 500 Hz) in the inaudible range were less robust and much fewer measurements could be collected when the phones...
were further apart from each other. When the distance could be measured, i.e. looking only at the number of successful measurements, the measurement errors were quite stable. Nevertheless, for some frequency bandwidths, the mean error and standard deviation was only a few centimetres (e.g. 20 kHz - 22 kHz). All these results were obtained in an indoor environment where there was little to no noise. Moreover, the devices were stationary, i.e. they were not moved while collecting measurements. As such, this experiment serves as the upper bound of the performance of the method in almost ideal conditions.

Another factor that influences the performance of the two-way acoustic ranging method is the recording time frame. The time frame represents the duration of a single recording. Figure 2.12 shows the standard deviation of the ranging error for three time frames (350 ms, 550 ms, and 750 ms) at the distances mentioned above. Longer duration for the time frame means that there is enough time to capture both signals emitted by the devices, yet calculating the cross-correlation takes longer. If the recording time frame is too short, there is a risk that the device cannot detect both signals and, consequently, cannot calculate a distance measurement. Picking the right duration for the time frame directly influences the number of distance measurements per second for the method. Figure 2.12 shows that the 350 ms time frame is not suitable due to the large standard deviation from the mean error. The other two recording time frames are more robust and even at 5 m away, the standard deviation was below 1 m. When smartphones are closer to one another, they perform similarly. Hence, we opted to use 550 ms as the recording time frame since this enabled us to collect multiple measurements. For example, 4 measurements in about 2 s in comparison to 3 measurements for the 750 ms time frame.

Two-way Acoustic Ranging Performance in Noisy Environments

In the previous experiment, we evaluated the performance of the two-way acoustic ranging method in a quiet, indoor environment. Multi-user and multi-device interactions may happen in a variety of settings, both indoors, outdoors, or in places where many people might be around. Because of this, in this experiment, we investigated the performance of our method in noisy environments. Our experiment used two LG Nexus 5X smartphones. For the ranging method, we used the set of parameters that we have shown to be the most robust and produce the best results. We set the recording time frame to 550 ms (Figure 2.12) and the frequency bandwidth was 750 Hz (Figure 2.11). The linear audio chirp started from 21 kHz to 21'750 Hz. Our experiment was conducted both indoors and outdoors. We placed the two smartphones on the sides of a busy path-way (Figure 2.13) and used the audio ranging component to collect 32 successful distance measurements. We repeated this procedure three times.
Figure 2.11. The performance of the ranging method for different distances in both the audible and inaudible spectrum, using multiple frequency bandwidths. The top row shows the number of successful and unsuccessful measurements from the total, while the bottom row shows the method’s performance in terms of error relative to the ground truth.
2.4 Evaluation

Figure 2.12. Standard deviation of the acoustic ranging error for different distances. The longer the time frame, i.e. the recording interval, the smaller the deviation becomes, which makes the method more reliable.

In the indoors setting, we placed the two smartphones on a hallway leading to the entrance of a cafeteria. We collected data around lunchtime when many people were passing by. The ground truth distance between the two devices was 3.5 m (Figure 2.13a). The average measured distance using two-way acoustic ranging was 350.86 cm (SD=49.79 cm). Besides the error, we also counted the number of unsuccessful measurements as a proportion of the total number of measurement attempts until collecting the 32 samples. In contrast to the previous quiet evaluation, the number of unsuccessful measurements increased to 19.9%. Outdoors (Figure 2.13b), the experiment setting was similar to the one indoors, but the ground truth distance was slightly larger, 3.8 m. The average measured distance was 392.9 cm (SD=34.65 cm), with an average error of 12.9 cm. The percentage of unsuccessful measurements was around 15%. While the performance of the method decreased in noisy environments, either indoors or outdoors, results show that the two-way ranging method is robust enough to enable collocated interactions and identify when users are close to one another.

Two-way Acoustic Ranging with a Moving Device

In all the previously presented experiments, the devices were stationary and did not move while collecting distance measurements. Interactive systems are dynamic, where users might move while interacting with other users. Consequently, in this evaluation, we
tested the performance of the two-way acoustic ranging method when one of the devices was mobile. We used the same parameters for the method as described in the previous experiment. The two devices used in this evaluation were the LG Nexus 5X and the LG G3.

Figure 2.14 shows the experiment set-up. One device was placed and fixed on a table while the other device was 5 m away from the stationary one. An experimental assistant started the acoustic ranging method and then proceeded to walk on a straight line towards the fixed device and then back to the initial position. This experiment was repeated multiple times at three different walking speeds: Slow, normal, and fast. Walking slowly, it took around 35 s (around 0.3 m/s) to go towards the fixed device and back. Normal walking took around 20 s (around 0.5 m/s) and fast took around 12 s (around 1 m/s). Figure 2.15 shows the observed paths for the three different walking speeds. Results show that the collected measurements captured the decreasing distance first from around 5 m to 0 and
2.4 Evaluation

Figure 2.14. Experiment setting. One stationary device was placed and fixed on a table. A second mobile device was held by an experimental assistant, initially 5 m away. Once the experiment started, the experimental assistant moved towards the stationary device and then back to the initial position. The experiment was repeated at three different walking speeds: Slow, normal, and fast. Then back again to 5 m. As such, leveraging the interpersonal distance (and implicitly inter-device distance) could be used as a predictor of the moment of interaction between users.

2.4.3 Combined Evaluation: Acoustic Ranging While Performing Hand Gestures

In the previous sections, we evaluated the two main components independently. Since the two components use a completely different set of sensors, there is no apparent dependency between the two. However, when performing in-air hand gestures, the mobile device is in motion. This motion of the device can influence the reliability of the acoustic ranging component because it changes the way sound propagates through the medium. This section investigates this influence.

We conducted an experiment similar to the one where we evaluated acoustic ranging with a moving device (Figure 2.15). Additionally, the person carrying the remote device continuously performed in-air hand gestures. Geometric gestures (Figure 2.5) were more suitable for this evaluation. They needed, however, to be adapted to a repetitive movement,
Figure 2.15. Two-way acoustic ranging with a moving device. The measurements captured the decreasing distance first from around 5 m to 0 and then back again to 5 m.

Figure 2.16. Gesture set 3: Geometric gestures used for the combined evaluation of the main components, acoustic ranging while continuously performing hand gestures.
which could be performed continuously while one device moved towards the other and then back again. Due to this constraint, we introduced a subset of five gestures (Figure 2.16). Both devices used in this experiment were Nexus 5X smartphones. The experiment was performed 30 times: five different gestures, two different speeds for hand movement (slow and fast), each three times. The walking speed was kept constant, with each trial lasting between 25 and 30 s. A slow hand movement translated into approximately one gesture per second, while a fast hand movement generated two to three gestures per second. Figure 2.17 shows the average number of distance measurement points when performing each of the five gestures. For each setting, we counted the points when the distance could be measured, the number of points when the distance cannot be measured (due to one device not hearing the signal played by the other device), and the number of outliers (distance measurements larger than 15 m).

A more detailed analysis of the observed paths can be seen in Figure 2.18. The two

![Acoustic ranging while performing hand gestures](image)

Figure 2.17. An overall comparison of the number of distance measurements when performing different gestures with variable hand movement speed (left is slow, right is fast). The faster the movement, the fewer the number of times when the distance can be measured.
figures on the top row show two of the gestures from the third gesture set (Figure 2.16), performed slowly while the distance between the devices decreased from 5 meters to 0 and increased back to 5 meters. The two figures on the bottom row show two such gestures executed at higher speed. When the hand movement speed increased, there were more situations when the distance could not be measured, as well as an increase in the number of outliers.

The combined evaluation shows that the acoustic ranging component is influenced by the movement of the mobile device and depends on how fast the gesture is performed. Nevertheless, enough distance measurement points can be collected reliably, which can be used in the proposed interaction technique.

Figure 2.18. A selection of four out of the total 30 observed paths which highlight different reliability cases. E.g. the upper left figure shows no points where the distance cannot be measured, while in the lower right figure there are many such points due to gestures being performed at high speed while the user was moving.
2.5 Applications

To demonstrate the feasibility and flexibility of our interaction technique, we implemented several application scenarios. Our first prototype was HandshakAR [15], an application where users can effortlessly share contact information when they perform the same greeting gesture and are close to each other. However, this application is limited to two participants. We demonstrate three additional applications which are suitable for a small group of people. These examples also demonstrate the potential design space for future collocated multi-user gestural interactions.

2.5.1 HandshakAR

This scenario showcases a common problem at cocktail parties. People get introduced to many other people, but keeping track of all these new connections can be difficult. Currently, when two people meet, they have to either exchange business cards or manually exchange contact information through their smartphones. With our method, we enable a seamless exchange of contact data. When two people shake hands (Figure 2.19) and are close to each other, a friend request (a Facebook request in our implementation) is automatically sent. By using proximity, we make sure that the contact information gets exchanged between the right people, because several people could be shaking hands at the same time.

Figure 2.19. HandshakAR prototype. (a) The system is inactive. (b) The devices are in close proximity to each other and, at the same time, a handshake greeting gesture was recognized. The devices exchange contact information, in this case a Facebook friend request. (c) The contact information appears on the user’s HMD (the Google Glass in our prototype).
Figure 2.20. Interacting with a magical treasure chest. Children can open the treasure chest when they are physically close to one another and perform the same hand gesture (spiral like motion with their device). If only two children are close to the treasure chest and do not perform the correct gesture, the chest stays locked (a). When at least three children perform the same gesture and are within 1 m from the treasure chest, the chest opens (b). The treasure chest is a metaphor for any smart or digital object.

2.5.2 Treasure Chest

This scenario shows a treasure chest that is part of a treasure hunt. The chest can only be opened if the whole team, a group of three or more children in this case, are physically close to it (within 1 m) and perform the same hand gesture (Figure 2.20b). If one of the children is further away or if they do not perform the hand gesture together with their teammates (Figure 2.20a), the chest stays locked.

2.5.3 Collaborative Fitness

Fitness activities like running have a strong social component. In this application scenario, teams of friends can compete against one another and compare different statistics like running distance, number of steps, etc. The application tracks the statistics of each individual member. Additionally, if the team members run together (are in proximity) their counts will be accumulated towards a group total (Figure 2.21). If one of the members is slower or much faster and separates from the group, the points will not be counted towards the group score. This creates an incentive for the members to work as a team. This concept can also be extended to medical recovery applications. Patients have to work together with a specialised trainer. Public displays could also benefit from such a framework. Combining proximity with gestures, participants who are actively interacting with such an installation can be separated from those who are simply passing by or just looking at the display.
2.5 Applications

Figure 2.21. Friends running together. Each device counts the individual number of steps and running distance. When teammates are in proximity to one another, their individual counts are accumulated towards the total group count.

2.5.4 Collaborative Music Band

The third application scenario is built around music. When friends get together, they can create an ad-hoc band and simulate playing an instrument (e.g. air guitar) with only their mobile or wearable device. Each participant can play one instrument at a time. The instruments do not have to be allocated in advance. The first user who performs a certain gesture (e.g. play the drums) will control that specific instrument (Figure 2.22). Our

Figure 2.22. An ad-hoc music band. A group of friends can create a band and play different instruments using their own devices. To play a specific instrument, a user has to simulate the motion of that instrument with their hands.
Chapter 2 Collocated Multi-user Gestural Interactions

The prototype supports two different songs and three different instruments: Drums, piano, or guitar.

2.6 Discussion

In this section, we discuss the advantages and the limitations of the building blocks that support collocated multi-user gestural interactions.

**Gesture Recognition via Motion Sensors.** We presented a method based on DTW to recognise hand gestures as user input. This approach does not involve any learning, but it has been shown to provide good results in the user-dependent setting, i.e. when a new sample is matched against templates belonging to the same user. This entails that each user has to record a set of template gestures before using the system. While more tedious at first, the method achieved a recognition rate close to 99%. With the proposed applications, we have shown the flexibility of the approach to reliably recognise different gestures using an accelerometer which is readily available in all current mobile and wearable devices.

**Proximity Detection via Acoustic Ranging.** Our proximity detection component is based on two-way ranging using inaudible signals. This method can estimate the distance between two devices at once. The maximum distance we could measure was 23 m. In a quiet indoor environment, the method achieves an error below 10 cm for distances of up to 5 m, measured with two smartphones. The optimal recording duration (time frame) is around 550 ms and the frequency band which provides the least amount of unsuccessful measurements is in the 21’000 Hz to 21’750 Hz range. For most group applications, such “guarantees” are sufficient. In particular, this method estimates the distance better than methods based on RSSI values using only off-the-shelf devices. One limitation of this method is that only two devices can do the two-way acoustic ranging procedure once. The method fails if multiple devices emit the same audio signal simultaneously. In our prototypes, we avoided this limitation by ranging the different devices in a round-robin fashion. To support multiple users, ranging is done in a round-robin fashion. Jin et al. propose a method to address this issue, but it only works for distances up to 1 m [96].

**Further Aspects and Limitations.** One important requirement of our approach is the need for a microphone and a speaker that work in the inaudible range. This also influences the hardware and, as we saw in the evaluation section, devices like the Motorola 360 smartwatch are not sensitive enough to capture such signals. Moreover, both smartwatches do not have any loudspeakers, which is why our prototypes have been implemented using smartphones. However, future devices are expected to have additional and better sensors. One limitation of Bluetooth is the power consumption which is addressed by the newer
version of the standard, Bluetooth Low Energy. Extended use of the microphone, speaker and motion sensors can have a significant impact on the battery. In the future, we plan to investigate the impact of our methods on mobile and wearable devices in terms of power consumption.

2.7 Conclusion

We presented a collocated multi-user gestural interaction technique with unmodified mobile and wearable devices. We support the development of new interaction possibilities that bring people physically close to one another. Proximity is detected with inaudible signals, hand gestures are recognised from motion sensors, and communication between the devices is handled over Bluetooth. All components are unobtrusive and do not break the interaction experience.

Our in-depth evaluation of the underlying components shows that the proposed interaction technique is feasible on unmodified devices. There are certain hardware limitations, namely the lack of loudspeakers or low-pass filters on microphones on some wearables, which hindered the end-to-end evaluation on smartwatches. However, new wearables might overcome these limitations and our work can be transferred to fully support these devices. Finally, we showcased the practical applicability of having collocated multi-user gestural interactions with four real-world applications.
Eye-based Human-Computer Interaction

This chapter is based on the following publications:


* These authors contributed equally to this work
3.1 Introduction

In this chapter, we present several contributions to the broader research area of eye-based human-computer interaction. We first present *ubiGaze* (Section 3.2), a novel AR system that uses gaze gestures to augment any real-world object with digital content. Eye movements and gaze gestures are detected using a wearable head-mounted eye tracker. Through our work on *ubiGaze*, we identify two fundamental issues that hinder the large-scale adoption of gaze-based interaction. First, simple regression-based wearable eye trackers have to go through a tedious and time-consuming calibration phase that may require specialised markers or additional assistance. In Section 3.3, we alleviate this issue by proposing fingertip calibration, which is a novel calibration approach. While more straightforward calibration methods improve the user experience when utilising eye trackers, the need for special-purpose eye tracking equipment is still a barrier that prevents most users from gaze-based interaction. Therefore, in Section 3.4, we explore an interaction method based on the smooth pursuit eye movement using a single RGB camera.
3.2 ubiGaze: Ubiquitous Augmented Reality Messaging Using Gaze Gestures

Figure 3.1. ubiGaze application scenario: (a) Users select a message from their smartwatch. (b) They then look at, i.e. fixate on, the object of interest to select the target for their message. (c) Users perform a gaze gesture to attach the message to the object of interest, a coffee machine in this case. (d) To retrieve a message, another user also has to first fixate the target of interest. (e) By performing the same gaze gesture, the message is unlocked and can be retrieved. (f) This message will now be visible on the user’s smartwatch.

Augmented Reality (AR) enables the direct or indirect view of a physical, real-world environment whose elements are augmented by a computer. This is an essential paradigm as it allows us to enrich our physical world with digital content without having to alter it. In this section, we present ubiGaze, a wearable AR system that enables users to augment any real-world object with digital information (Figure 3.1). Users can attach context-dependent invisible messages to real objects by first fixating at an object of interest and then, through gaze gestures, they are able to attach a message (or other types of digital information) to that object. We opted to use gaze gestures since they are deliberate and unnatural movements of the eye that follow a specific pattern and can be used for interaction [50]. To retrieve the information attached to a specific object, users have to look at the same object of interest and perform the exact same gaze gesture. This provides a discrete and effortless novel interaction technique that extends the concept of annotations or tags in AR [162] to a messaging service. The key novelty of this work is the coupling of three elements: any real-world object, any gaze gesture, and a message. We demonstrate our
system in a real-world scenario by building a prototype with two wearable devices: A head-mounted eye tracker and a smartwatch.

### 3.2.1 Related Work

In general, many believe that AR is a paradigm that requires head-mounted displays (HMD). However, Ronald Azuma argued that AR should not be limited to certain technologies [9] and further defines three fundamental characteristics: AR combines the real and the virtual, it is interactive in real time, and it is registered in three dimensions. The main technological benefit from AR is to enhance the user’s perception and interaction with the world. Furthermore, the real world can be augmented with information that users cannot see or access with their own senses. As such, Ronald Azuma defined six classes of potential AR applications that have been explored [9]: medical visualisation, maintenance and repair, annotation, robot path planning, entertainment, and military applications. Almost 20 years later, a new study [27] presented another overview of the developments in the field. While the taxonomy for applications has changed, with the authors focusing on marketing, education, entertainment, and architecture scenarios, we observe an ever-increasing interest in enhancing normal, everyday objects with additional information. One component of this vision is the annotation problem or how to attach digital content such as tags to objects in the physical world.

The annotation problem for AR has been studied before. WUW - Wear Ur World [153] is a wearable gestural interface that allowed projecting information out into the real world. It is a wearable system composed of a projector and a tiny camera, which allows the system to see what the user sees and through projection, information can be displayed on any surface, walls, or physical objects around us. SixthSense [152] further extended the previous concept and enabled users to interact and manipulate augmented information through hand gestures. Ambient aNnotation [182] is another AR system that aimed at assisting persons suffering from Alzheimer’s disease. Caregivers could create ambient tags, while patients could use a mobile device to recognise those tags in their own environment and retrieve relevant information. SkiAR [60] is a sports oriented AR system with a focus on skiing and snowboarding. Their system allowed users to share rich content in-situ, i.e. while on the slope, on a printed resort map. SkiAR had two components: a smartwatch for input and a mobile phone used as a display. Nuernberger et al. developed a method to display 2D gesture annotations in 3D augmented reality [164]. The authors did not focus on recognising objects and embedding virtual tags, but rather allowing users to draw annotations in a 2D environment and overlay them in 3D.

An approach similar to ours is Tag It! [162]. It is a wearable system that allowed users
3.2 ubiGaze: Ubiquitous Augmented Reality Messaging Using Gaze Gestures

to place 3D virtual tags and interact with them. While the general goal of annotating real objects with virtual messages is the same, there are two important differences that set our works apart. First, there is a difference in the underlying method. Tag It! [162] proposed a method based on 3D indoor tracking with a chest-worn depth sensor and a head-mounted display, while in our approach we developed a method that detects real objects using the scene camera of a head-mounted eye tracker and a smartwatch for input. Our approach does not require any tracking. Second, we propose an extension to annotation (or tagging) systems by enabling a context and location dependent messaging service. Messages can be attached and retrieved by users if they perform the same gaze gesture on a target object.

3.2.2 Gaze Gestures

The human eye is a promising high-bandwidth input modality for human-computer interaction. Early works to leverage eye movements for gaze-based interaction relied mostly on fixations and dwell time [72]. More recently, gaze gestures have attracted interest in the HCI community because, in comparison to other gaze interaction methods, they do not require any calibration. Only the relative motion of the eye is tracked and gestures are less sensitive to accuracy problems caused by the underlying eye tracking technology. Moreover, gaze gestures also alleviate the well-known Midas touch problem, which occurs because our eyes are never “off”. Consequently, when designing gaze-based interaction, often it is necessary to include a clutch mechanism to differentiate between intentional and accidental interactions [172, 173]. When using gaze gestures, accidental movements of the eyes in the exact same pattern are unlikely.

Drewes and Schmidt defined a gaze gesture as consisting “of a sequence of elements, typically strokes, which are performed in a sequential time order” [52]. A stroke is defined as a movement across one of eight possible directions in 2D and a gesture is composed of a sequence of such stokes. The time required to perform a gesture was around 1.9 s, however, this depends on the number of strokes that defined a specific pattern. There are also gestures with only a single stroke [155]. Such gestures have reduced cognitive load, they are easy to remember, and can be integrated with dwell time to create gaze-controlled interfaces. One of the most common tasks when designing new input techniques is object selection. With gaze gestures, Mollenbach et al. [156] showed that the selection time using single stroke gestures was influenced by multiple factors such as the tracking system or whether the strokes were horizontal or vertical.

Several applications of gaze gestures exist. EyeWrite is a system that uses alphabet-like gestures for text entry [231]. Drewes and Schmidt demonstrated the use of gaze gestures for interaction with a mobile phone [50]. Their findings from a user study highlighted
Chapter 3 Eye-based Human-Computer Interaction

that participants found interactions using dwell time more intuitive, but gaze gestures were more robust since it was unlikely to perform them unwillingly. Gaze gestures were also introduced as a way to reduce the risk of shoulder surfing for the PIN entry task [41]. Istance et al. investigated gaze gestures as a means of enabling people with motor impairments to play online games [89]. In a user study with 24 participants, they showed that gestures are particularly suited for issuing specific commands, rather than for continuous movement where more traditional input modalities (e.g. mouse) work better. Hyrskykari et al. demonstrated how gaze gestures can reduce the number of selection errors in the context of gaming in comparison to dwell-based interaction [86].

Another relevant aspect of gaze-based interaction is user feedback. Kangas et al. [100] evaluated gaze gestures in combination with vibrotactile feedback. Their findings showed that participants made use of haptic feedback to reduce the number of errors when performing gaze gestures. In our application, we also use vibrations to inform users when a message has been received by the smartwatch.

3.2.3 Overview

The main application scenario of ubiGaze is to allow users to attach and retrieve invisible messages from any real-world object. It is a location and context-dependent AR application that uses an authentication mechanism based on gaze gestures. The system is enabled by two wearable computers: A wearable head-mounted eye tracker and a smartwatch. The head-worn eye tracker estimates the gaze direction of the user which, together with the scene camera, indicates where the user is looking in the surrounding physical environment. The smartwatch is used both for input and output. Users can select the message that they want to embed by using the touch interface of this wearable. The smartwatch is also used to display the otherwise invisible messages that users retrieve from objects that were previously augmented.

Attaching and Retrieving Messages

Figure 3.2 illustrates the two different use cases on a practical example: Attaching and retrieving messages from a real-world coffee machine. To augment an object with a message, users have to first find an object of interest. The system relies on the eye tracker and the scene camera, which offers a first-person-view of the world, so the system sees what the user sees. When users fixate on a specific object, the system sets a lock on it. User fixations are based on dwell time, which represents the amount of time users look at a specific target. Using computer vision, we extract relevant features from the
3.2 ubiGaze: Ubiquitous Augmented Reality Messaging Using Gaze Gestures

Figure 3.2. Overview of the two use cases enabled by ubiGaze. (a) Attaching a message to any real-world object (in this example a coffee machine). (b) Retrieving the message from an object that was previously augmented. Note: This figure uses an image of the Pupil eye tracker [105].

region of interest, which is given by the point of regard in the image captured by the scene camera. Next, users select a (predefined) message from their smartwatch and perform a gaze gesture to attach the message to the object (Figure 3.3).

Figure 3.3. First person view of the interaction flow: Users select the object of interest through fixations. Distinctive features are extracted from the selected object. Users perform a gaze gesture and the message is attached to or retrieved from the desired object.

Users can retrieve messages from objects that were previously augmented in a similar way. There are several possibilities. First, the simplest case is when the user knows in
advance the object that contains the message. Second, a visual marker or some form of audio or haptic feedback could inform users when the object they are looking at contains any messages. In our application scenario, users have to know in advance which objects have been augmented with messages. Other scenarios could be explored as well, for instance having a virtual mailbox that is the equivalent in AR of a physical mailbox: Anyone can see it, but only users with the right key, i.e. gesture, may unlock it. In any scenario, users need to know what gesture unlocks the message. When users are in the proximity of the augmented object, they must first fixate, i.e look at the desired object. This is viewed as a selection mechanism.

To detect the object of interest, our system can extract meaningful features from the scene image. Once detected, users have to perform the same eye gaze gesture (e.g. a circle or a triangle) and then the message is unlocked. Users will receive a notification with the content of the message on their smartwatch.

### 3.2.4 Implementation

Our prototype had the following two hardware components: A head-mounted eye tracker and a Sony Smartwatch 3. The Pupil eye tracker is a wearable head-worn device developed by Pupil Labs [268]. The high-resolution scene camera can capture images with a resolution of 1910x1080@30fps. The camera is also equipped with a microphone for audio recordings, but we did not use this feature in our prototype. The eye camera operates at 120 Hz and it is an Infrared (IR) camera with IR illumination for dark pupil tracking. The two devices are interconnected through a computer that processed the eye tracking data (implementation with Python and Processing [267]) and acted as an Internet gateway to the messaging server. The server was implemented with Node.js [265], exposed a RESTful interface, and stored data tuples that contained the gesture, the representation of the object, and the message.

In our implementation, we simplified the object detection component. We used colour markers, i.e clearly visible patches of a single colour, as the distinct feature of the object to be detected. Our algorithm retrieved a region of interest around the point of regard in the scene image. If the average RGB colour values from all the pixels within that region were in a certain range (e.g. yellow), then we detected it as a marker. We discuss other methods for detection in the following sections.
3.2.5 Challenges and Future Research Directions

The first and perhaps the most significant challenge when using the Pupil eye tracker was the need for a tedious and time-consuming calibration process. Such simple regression-based eye trackers, have to go through a calibration phase that identifies a mapping between a point in the eye image (or the eye characteristics) and a point in the scene. This process is not automatic, users have to manually calibrate the eye tracker by finding several calibration points. Moreover, if users are on the move, the calibration tends to become rather inaccurate and a re-calibration of the eye tracker is necessary.

Another limitation in our implementation of ubiGaze is that it requires a computer to act as a central node for the system. The Pupil eye tracker has to be connected to a computer that processes the data, which limits the portability of the entire system. In the future, we believe that such wearable eye trackers can become self-contained devices that have sensors, connectivity, and computational capabilities, similar to the Google Glass.

Besides the limitations of the eye tracker, our gaze gesture recognition approach can also be improved. To detect such gestures, we relied on a library delivered with the Pupil SDK. Our research objective was not to develop and improve gesture recognition algorithms, but use existing technology to explore a novel interactive scenario. It is still unclear whether gaze gestures, as an authentication mechanism, are the right approach for ubiGaze. The question whether users will adapt easily to gaze gestures requires further investigation. Preliminary user evaluations have shown that users found this interaction technique rather cumbersome due to the inaccuracy of the gesture recognition algorithm.

Using a smartwatch as an input modality was straightforward. In our prototype, we developed an application with predefined messages (a set of Emoticons) that the users can select from a list. Given the small form factor, text input is difficult, as there is no keyboard available. However, this limitation can be addressed through spoken commands and voice interaction. This way, users could dictate the content of their message.

Our method is dependent on computer vision techniques for object recognition. Object detection and recognition are active research areas and there is a wide body of prior and ongoing work. Our application is only as good as the underlying methods. In our implementation, to facilitate prototyping, we have used simple colour markers for recognising the objects that users want to augment. Any other approach for object detection could be used, as long as it works in real time.
3.2.6 Conclusion

We presented *ubiGaze*, a wearable AR system that enables users to augment any real-world object with invisible messages. This idea extends the concept of AR tags or annotations to a ubiquitous messaging system based on gaze gestures, where messages are locked into a set of distinctive features of real-world objects. Our system is composed of two wearable devices, an eye tracker equipped with a scene camera (Pupil) and a smartwatch (Sony Smartwatch 3). Users are able to post messages through a combination of gaze gestures and input from their smartwatch. Similarly, users are able to read invisible messages from augmented objects by performing gaze gestures and use their smartwatch as a display. By combining different wearable devices we proposed a discrete and effortless interaction technique for embedding AR messages into any real-world object. We believe that this new technique can lead to novel interaction scenarios in wearable computing.
3.3 Wearable Eye Tracker Calibration at Your Fingertips

Figure 3.4. Users can quickly and independently calibrate a head-mounted eye tracker by simply pointing with their fingertip at locations in the scene. No dedicated marker, display, or additional assistance is required, enabling a calibration technique suitable for mobile and pervasive eye tracking.

As smart devices become ubiquitous, eye gaze can enhance the way in which we interact with objects around us [136]. Head-mounted eye trackers such as the Pupil [105] enable mobile and pervasive eye tracking [31] and we have already experimented with gaze-based interaction in situations that are not constrained to a desktop setting [12]. To infer the point of gaze or where the user is looking, most regression-based eye tracking systems have to go through an initial calibration phase.

Calibration is often a necessary and important first step in gaze estimation and is used to determine the mapping function between the eye’s characteristics (e.g. the centre of the pupil) and a point in the scene. High-end devices that rely on 3D geometric models of the eye can be used without a calibration phase, however, the high cost of such devices hinders the large-scale adoption. Cheaper, more flexible regression-based systems require a careful initial calibration or a re-calibration in case the device moves on the person’s head. Typically, users have to collect N samples of gaze points at known locations [104]. In a laboratory setting, a common approach is to have a second person assist with the calibration by manually clicking on points fixated by the user. An alternative is to use calibration markers, which can be printed or displayed on a screen (Figure 3.5). They can also be placed in the environment, thus making them reliable 3D position markers. While beneficial for some applications, there are scenarios where the environment cannot be augmented with markers. For example in-the-wild or outdoor studies are difficult or impractical to instrument with markers. When focusing on mobile settings, users might not have a screen, a dedicated marker, or assistance available. Additionally, current calibration methods are considered difficult and tedious [174].
Figure 3.5. Existing calibration methods for regression-based eye trackers require either special markers shown on a computer screen (a), mobile phone (b), or printed on paper (c), or the assistance of a second person (d).

We propose a novel method that enables users to quickly and independently calibrate a head-mounted eye tracker. Users do not need any specialised markers, they can use their own fingertips (Figure 3.4). By pointing at locations in the scene and fixating on their own fingertip, users can easily collect calibration samples in different environments. For distances of about an arm’s length, the proposed method achieves a comparable accuracy to standard marker-based calibration. We also provide an implementation of this method as a plugin for the open-source Pupil platform [274].

3.3.1 Related Work

User-dependent calibration [54] is regarded as one of the key challenges that hinders the wide adoption of eye trackers. Researchers have tried to define guidelines to help with selecting the number of points to use, their location, or the type of mapping function [104].

The Pupil open-source platform [105] supports several calibration methods: Markers displayed on a screen, printed markers, or natural features. Pursuit calibration proposes the use of moving targets with a known trajectory [34, 174] and can achieve an angular error
as low as 0.6°. A different approach is to leverage user events and possible interactions with a PC [82, 103]. Egocentric visual saliency can also be used for a continuous self-calibrating eye tracker [209]. CalibMe [193] is a marker-based calibration approach which facilitates fast and unsupervised collection of a large number of calibration points. Others have tried to reduce the number of re-calibrations necessary either by making them more time-efficient [123] or by leveraging mobile phone usage [160].

While the above methods provide good results in different scenarios, most of them rely on a display to show the markers, a second person to assist with the procedure, or an initial calibration. We aim to simplify the initial calibration step without the use of a screen or additional assistance.

### 3.3.2 Fingertip Calibration

![Fingertip Calibration Diagram](image)

Figure 3.6. Method overview: Finger calibration requires hand segmentation and fingertip detection. The hand is segmented based on colour. Finger candidates are points on the convex hull enclosing the hand contour. A hull point is a finger candidate if the angle between this point and two adjacent convexity defect points is smaller than a threshold ($\alpha = 60°$).

A gaze calibration algorithm collects sample points of known eye gaze locations in order to estimate a gaze-to-output space mapping function. We tackle this problem in a novel way. Users only have to hold their hand and fingertip in the scene camera’s field of view and fixate the tip of the finger. Once a fingertip has been detected, the system will output an audio feedback to let users know that multiple sample points are being collected from that specific location. A second sound will inform users that the sampling process has finished. Users can then move their fingertip to point towards a different location to further collect calibration samples.
Hand and Fingertip Detection

Fingertip calibration relies on computer vision methods to segment the hand and detect the tip of the fingers. In our implementation, we followed a similar approach to other prior works that used colour information or contours to segment the hand and then detect the users’ fingers [29, 66, 137] [258, 259, 262, 271]. Figure 3.6 gives an overview of our method. The first step is to apply a blur filter on the input image to remove noise. The hand can be segmented from the background with a binary mask which is obtained from colour-based segmentation in the HSV colour space. In this space, the colour is encoded in the hue channel, which makes thresholding easier. In our implementation, we use a hue range from 0° to 40°, a saturation range from 12% to 60%, and a value range from 24% to 100%. The above parameters work well with light-coloured skin, however, there are more sophisticated skin segmentation methods for general application [22]. We fill small holes in the binary mask through dilation. We can then use the resulting mask to segment the hand and remove the background (Figure 3.6A).

Once the image has been segmented, the contour with the largest area and its convex hull are computed (Figure 3.6B). Since the convex hull spans the entire contour, the edge points of the convex hull can be very dense. In order to group these dense edge points together (meaning that only a single edge point belongs to a fingertip), each edge point on the convex hull is assigned to a cluster. Two points are assigned to the same cluster, if the corresponding Euclidean distance is smaller than a certain threshold \(d\). In our implementation, we chose \(d = 50 \, \text{px}\) (frame resolution is \(1280 \times 720 \, \text{px}\)) and the assignment and union of clusters is done through disjoint sets. After assigning each edge point to a cluster, the points closest to the centre of their respective cluster are chosen to be the new edge points representing their cluster. Figure 3.6C shows the outcome of this reduction step.

To detect whether a hull point could represent a fingertip, we first calculate the convexity defects of the convex hull. Figure 3.6D shows the convex hull points in red, while convexity defects are shown in yellow. \(\alpha\) represents the angle between the two vectors which are spanned by a hull point and two neighbouring convexity defects. In our implementation, if the angle \(\alpha\) is smaller than 60°, the hull point is identified as a finger (Figure 3.6E).

The last step of the finger detection algorithm is to correct the centre of the fingertip. In its current state, the algorithm selects a convex hull point, but this point is located on the finger’s contour. The hull points, which were detected as fingers, are shifted towards the centre of the fingertip. The direction for this translation is obtained by looking at the angle bisector of \(\alpha\), which approximates the direction of the finger. Figure 3.6F shows the new positions of the hull points after translating them along the finger direction towards the fingertip.
Complete System

In addition to sampling calibration points, a complete eye tracking system requires a pupil detection algorithm and a gaze mapping function. The pupil detection algorithm locates the (centre of the) pupil in the infrared image captured by the eye camera. In our implementation, we use the 2D pupil detector provided with the Pupil software [105]. This method is robust to users who wear contact lenses or eyeglasses because it does not rely on corneal reflections. The gaze mapping or the transfer function which maps pupil positions to the scene space is estimated through calibration. It has been shown that simpler polynomial functions work better when the number of sample points is low [104]. Our prototype uses two bivariate polynomials (Equation 3.1):

\[
\begin{align*}
x_s &= A_x x_e^2 + B_x y_e^2 + C_x x_e + D_x y_e + E_x \\
y_s &= A_y x_e^2 + B_y y_e^2 + C_y x_e + D_y y_e + E_y
\end{align*}
\]

where \((x_e, y_e)\) are the \((x, y)\) coordinates in the eye space and \((x_s, y_s)\) are the target coordinates in the scene space. Mapping a two-dimensional space to a one-dimensional space can be reduced to the problem of surface fitting. Calculating the coefficients of the two polynomials is done with the Levenberg-Marquardt algorithm which solves non-linear least square problems.

3.3.3 Evaluation

We conducted several experiments to evaluate the proposed finger calibration method. The accuracy of a calibration is measured as the average angular offset between fixation locations and the corresponding ground truth targets. The more samples are collected, the better the gaze mapping function can be estimated. Besides accuracy, the usability of an eye tracker is influenced by the calibration time.

Our experiments were performed using the Pupil eye tracker from Pupil Labs in a monocular set-up. The device was equipped with one eye camera (60 Hz) and one scene camera (30 Hz). Our finger calibration method was evaluated similarly to a typical 9-point calibration where the users were instructed to collect sample points from 9 different locations in a grid like pattern. In all the experiments, users first performed a calibration and then an accuracy test and had the freedom to choose the 9 points as they wanted.
Number of samples versus duration

The number of sample points collected for the calibration can impact the tracking accuracy. In this first experiment, we analyse the number of samples necessary per location or fixation target. The Pupil open-source platform uses 30 sample points. A 9-point calibration would lead to 270 samples before removing any outliers.

![Figure 3.7. Number of samples per fixation vs. calibration duration. Increasing the number of samples per location also improves the stability of the overall calibration. However, this comes at an increased time cost for the user.](image)

The experiment was performed by an expert with more than one year experience in working with the Pupil eye tracker. The tested parameter is the number of samples per fixation. For each value of this parameter (10, 20, 30, 40, or 50), we performed three sessions. Each session involved a 9-point finger calibration followed by a 9-point finger accuracy test.

Figure 3.7 shows that varying the number of samples per fixation does not significantly influence the calibration accuracy. On average, the angular error varies between $1.3^\circ$ and $1.8^\circ$ with the minimum value being obtained for 30 calibration samples. One possible explanation for this could be the time required to collect these data points. The longer a user has to fixate a specific target, the higher the chance of a saccade or attention shift which would lead to incorrect samples. A calibration where only 10 samples per fixation are collected requires, on average, around 21 s. In contrast, collecting 50 samples increases the calibration time to around 37 s.
3.3 Wearable Eye Tracker Calibration at Your Fingertips

Preliminary user evaluation

The main goal of this work is to enable eye tracking calibration which can be performed quickly and independently without the need of a display or additional assistance. So far, we have evaluated the accuracy of the proposed method in a laboratory setting. In this experiment, we compared finger calibration to standard marker-based calibration. To make the comparison fair, the marker was displayed on a mobile device’s screen and the calibration had to be performed similarly (Figure 3.8).

![Figure 3.8](image)

Figure 3.8. The two conditions compared in the user evaluation. (a) Finger calibration (b) Marker calibration with the marker displayed on a mobile phone’s screen.

We designed a pilot user study and gathered quantitative and qualitative data. Twelve adult subjects aged between 23 and 58 years old (M=35.2, SD=13.6, 8 male and 4 female) took part in the evaluation. Nine participants were wearing vision correcting glasses. Five users had previously been exposed to eye tracking, however, only two of them had tried to calibrate an eye tracker before. Most of the participants (9 out of 12) have a technical background. Each subject was asked to perform two tasks. The first task was to calibrate an eye tracker with their finger, followed by a separate accuracy test. The second task was to use the standard manual marker calibration, again followed by an accuracy test. Users were informed to collect samples from 9 locations covering their field of view in a grid-like pattern (similar to a 9-point on-screen calibration). There was no information given on how to spread out the points or how to hold the finger/smartphone. Each calibration session was performed twice. The presentation order of the conditions was counterbalanced. After each task, each user was asked to complete a System Usability Scale (SUS) [30] questionnaire. This is a quick, reliable, and well known tool for usability testing. It consisted of ten simple questions on the Likert scale aiming to offer a high-level subjective assessment of the system’s usability. After completing both tasks, participants had to fill in an additional form which compared the two conditions. The duration of the experiment was between 15 and 20 mins per participant.
Figure 3.9 shows the mean angular error per participant. On average, the finger calibration error (M=2.68°, SD=0.820°) is comparable to the marker calibration error (M=2.49°, SD=0.673°). A student’s t-test reveals that the difference between them is not statistically significant (t(22) = 0.640, p = 0.53, Cohen’s d = 0.253).

A closer look at participant P12’s data reveals that, for a certain session, only 36 data points out of the total 240 collected have been used for finger calibration. The accuracy test revealed an error of around 4.9°. Similarly, for the same participant, only 38 data points were used in the marker calibration session. In this case, the angular error for the accuracy test was around 3.8°. One possible explanation for using so few points for the calibration is the outlier removal step from the Pupil software. Data points where the fixation was too short or the fixation target was not stable in-between measurements are discarded.

In terms of usability, the average SUS score for finger calibration was 78.8 compared to 76.3 for marker calibration. The SUS does not offer a guideline on how to compare scores, however, given its wide use, the average score is considered to be 68. A student’s t-test shows that the difference between finger (M=78.75, SD=16.93) and marker calibration (M=76.25, SD=16.63) is not statistically significant (t(22) = 0.365, p = 0.719, Cohen’s d = 0.149).

Qualitative feedback. At the end of the experiment, participants had to fill in a questionnaire in which the two calibration methods were compared in terms of preference,
perceived speed, ease of understanding, and the feeling of being in control. In terms of perceived speed, half of the participants saw no difference between the two methods, while four of them said that the finger calibration seemed faster. Nine out of 12 participants found the two experiments equally easy to understand, which was in line with our expectation. In terms of control, seven participants said that the finger gave them a better sense of being in control. Overall, seven out of 12 participants expressed that they preferred the finger over the marker, four were in favour of the marker, and one had no preference.

Hand segmentation

The proposed calibration approach relies on hand segmentation and fingertip detection. In this experiment, we quantitatively evaluated the hand segmentation on the EgoHands dataset [22]. The dataset contains 48 videos taken with a Google Glass from an egocentric perspective, similar to a head-mounted eye tracker. It focuses on the interaction between two people and contains 4800 manually annotated frames with pixel-level masks for hands present in the scene (up to four segmentation masks, one for each person’s hand).

Our algorithm was designed to identify one segmentation mask, the one with the largest contour. In 4177 images (87%), our segmentation overlapped with one of the manually segmented hands. For such cases, the overlap in pixels was, on average, 60%. For 61 images (1.3%), our method correctly identified that no annotated hands were present in the scene. For the remaining 562 images (11.7%) our algorithm failed to detect any. These results show that the proposed method is simple and good enough for this application.

Runtime analysis

We evaluated the runtime of the finger detection algorithm on a laptop with an i5 CPU @ 2.3 GHz. Small images (320 × 240 px) need around 7.7 ms (130 FPS), medium images (1280 × 720 px) need around 18 ms (55 FPS), and large images (1920 × 1080 px) need around 33 ms (30 FPS). This shows that our method is fast enough to accommodate the Pupil’s scene camera frame rate.

3.3.4 Discussion

Parallax Effect. The proposed method collects calibration samples at about an arm’s length away from the user. If the distance to the point of regard is different than the calibration distance, there will be a parallax error. This happens due to the position of the
camera relative to the eye. This type of error is found on most video-based monocular mobile gaze trackers and can also influence the results of our method. Mardanbegi and Hansen [138] studied this effect for different calibration and fixation distances. Methods like CalibMe [193] allow calibrating at varying distances but, to minimise the parallax error, require recalibration for every target fixation plane. Our method is well suited for mobile scenarios, but the calibration distance is limited. The best results are obtained when the interaction happens closer to the user (e.g. in interaction scenarios where the objects are within the user’s reach).

*Area covered during calibration.* In the user evaluation, for both the calibration and the accuracy test, participants had to sample 9 locations within their field of view in a grid-like pattern. To simulate a realistic scenario, they were not told how or where to place their finger/smartphone. An analysis of these locations has shown that points which fall outside of the calibration area will have larger errors due to extrapolation.

### 3.3.5 Conclusion

In this section, we presented a novel calibration technique which simplifies the initial calibration step for head-mounted eye trackers. Users collect calibration samples by pointing with their fingers and fixating the fingertip. No additional assistance or specialised calibration markers are necessary, thus enabling quick initialisation for pervasive and mobile eye tracking in any environment. The proposed method achieves comparable accuracy to similar marker-based calibration. Our preliminary user evaluation highlighted that the majority of the participants preferred finger calibration over traditional markers. In the future, we plan to investigate and incorporate ways to mitigate the parallax effect in situations where the fixation plane is further away from the user.
In recent years, pursuits has emerged as the first gaze interaction technique that allows for both natural and spontaneous, calibration-free interaction with dynamic user interfaces. It relies on smooth pursuit eye movements that are performed when following a target moving along a continuous trajectory at an appropriate speed. By correlating these eye trajectories with those of on-screen objects, the single target that users follow with their eyes can be robustly identified. The original work introducing pursuits [220] has therefore spurred a large number of follow-up works resulting in applications in multiple interactive settings such as public displays [112], smartwatches [55], or virtual reality [111].

However, all of these works require special-purpose eye tracking equipment, i.e. dedicated devices built and sold specifically for this task. Unlike other gaze interaction techniques such as dwelling [26] or gaze gestures [12, 52], pursuits does not require a calibrated eye tracker. Nevertheless, robust tracking of the relative movement of the eyes is fundamental to the technique. Moreover, dedicated eye trackers may not always be available or may be difficult to integrate into the small form factor of some devices, such as mobile phones. Several works have proposed methods that implement Pursuit-like interactions for other body parts [37], such as the hands or arms [33], based on off-the-shelf cameras and computer vision. However, computer vision-based techniques for Pursuit interaction using gaze have not been explored as of yet.

We propose a novel method to detect pursuits using a single off-the-shelf RGB camera in unconstrained remote settings. Our method combines appearance-based gaze estimation
and optical flow into a joint pipeline and is, therefore, able to capture both the gaze direction and the eye movement dynamics during a pursuit movement (Figure 3.10). On the one hand, the full-face appearance-based gaze estimator [253] predicts the 2D point of regard from a single image. By correlating the gaze estimates and the position of the moving target using the Pearson product-moment correlation, we obtain a first target candidate. On the other hand, our method uses dense optical flow [58] in the eye region extracted from a series of normalised face images to estimate the eye movement direction. Our method then correlates the eye movement directions and the target motions using cosine similarity to obtain a second target candidate. By combining the two different perspectives on eye movements and aggregating the outputs from both approaches, our method shows increased robustness.

The specific contributions of our work are two-fold. We first present a novel method to detect pursuits in unconstrained remote settings, which does not require any special-purpose eye tracking equipment but only a standard off-the-shelf RGB camera. Second, we evaluate our method in a 13-participant user study and show that it outperforms EyeFlow [73], the current state of the art, by a large margin. For a small number of targets, our method achieves over 90% accuracy, i.e. the percentage of correctly identified targets, and is even competitive with a consumer eye tracker. As such, our work paves the way for a new class of methods that enable spontaneous pursuits interaction in the wild.

3.4.1 Related Work

Our research relates to previous work on 1) smooth pursuit interaction, 2) gaze estimation, and 3) optical flow estimation.

Smooth Pursuit Interaction

Selecting a target from multiple user interface (UI) elements is a key task in gaze-based interaction [199]. Pursuits [220] is a recent alternative to pointing that has a wide range of applications, including interaction with public displays [108, 112], smartwatches [55], or VR [111]. Other works investigated the use of pursuits with other body parts, e.g. to control ambient devices with the hands [217] or to provide secure input of PINs [39] or passwords [4]. An advantage of the technique is that it is robust to partially hidden trajectories [146]. Further work optimised the method itself [216] and extended possible use cases to novel tasks, such as text entry [51]. Rather than using pursuits for interaction, others instead used it as an implicit calibration method for eye trackers [34, 174].
3.4 Combining Gaze Estimation and Optical Flow for Pursuits Interaction

While pursuits enables novel interactive experiences, the need for special-purpose eye tracking equipment hinders its broader applicability. In contrast, our method only requires a single RGB camera, such as a webcam. The closest work to ours is EyeFlow [73], but that work was designed for a wearable, head-mounted setting in which the camera is mounted close to the eyes, and therefore requires a high-resolution eye image. Moreover, their method assumes that the camera is rigidly attached to the user’s head – an assumption that no longer holds in remote settings. As such, we are first to propose a method that tackles the particularly challenging remote setting in which users are at a distance and the user’s eyes constitute only a small, low-resolution part of the full camera view.

Gaze Estimation

Gaze estimation is the task of estimating a user’s 3D gaze direction or 2D point of regard. While early works required pupil detection or (infrared) illumination [130, 159], more recent methods directly use the face’s and eye’s appearance leveraging large datasets and machine learning [252, 254]. For example, Zhang et al. proposed a full-face appearance-based gaze estimator trained on the MPIIFaceGaze dataset [253], while Krafka et al. introduced a convolutional neural network (CNN) trained on the large-scale GazeCapture dataset for mobile gaze estimation [118]. Learning-based methods have already outperformed feature- or model-based approaches and have shown increased robustness in unconstrained settings even without calibration. However, angular errors between 4° and 6° still prevent them from being used in high-accuracy applications.

Part of our method uses one such generic appearance-based gaze estimator [251, 253] trained on the GazeCapture dataset. User or model adaptation through calibration could have further increased its performance [248], but this contradicts the concept of spontaneous calibration-free interaction. Instead, for increased robustness, we propose to additionally capture eye movement dynamics using optical flow.

Optical Flow Estimation

Optical flow estimation is a computational task in computer vision with the goal of estimating the apparent motion of the pixels in the image plane. It is widely used in applications such as object detection and tracking [81], semantic segmentation [196], or activity recognition [200] because it serves as an approximation of the real physical motion. Methods that estimate sparse optical flow only examine a reduced number of pixels or features in an image [132]. In contrast, dense optical flow estimates the flow vectors for all pixels in the entire image [58], which leads to increased performance at the
estimates are then combined to make a joint decision about the single most likely followed on-screen target.

Figure 3.11. OVERVIEW of our method that consists of two components running gaze estimation and optical flow independently to first independently correlate eye movements to the object's motion trajectories and thereby estimate the target the user is following. These estimates are then combined to make a joint decision about the single most likely followed on-screen target.
3.4 Combining Gaze Estimation and Optical Flow for Pursuit Interaction

The current state of the art in optical flow estimation is end-to-end deep learning models such as FlowNet2 [87].

We are the first to leverage both static and dynamic information about eye motion to detect smooth pursuit eye movements. To estimate the motion of the eye, for maximum performance, we leverage a dense optical flow method [58].

3.4.2 Method

To detect whether a user is following a moving UI element, our method combines appearance-based gaze estimation and optical flow in the eye region to jointly analyse the eye movement dynamics during a pursuit (Figure 3.11). Given a sequence of images, the first step is to detect the users’ face and facial landmarks in each image individually. Face bounding boxes are detected using three multi-task CNNs [247] and an hourglass network [43] then predicts 68 facial landmarks inside the detected bounding box, which are then further used in the single components.

Gaze Estimation

The goal of the gaze estimation component is to correlate the 2D gaze estimates from a window of $N$ images to the 2D coordinates of all moving UI elements. The output of this component is either a candidate target if the correlation value is over a predefined threshold, or None if no target can be detected.

Face Image Normalisation. It is an effective preprocessing step in appearance-based gaze estimation [250, 252]. By rotating and scaling the input image, normalisation cancels out differences in user appearance caused by the user-camera distance or different hardware setups. Normalisation requires an estimate of the user’s head pose that can be defined as the 3D translation and rotation of the head relative to the camera. To estimate the 3D head pose, we opted to use the method proposed in [16] that has shown improved robustness to appearance variability across users and head pose angles. Using the head pose information, the input image is warped to a normalised space with fixed parameters and cropped to a size of 448x448 px (Figure 3.11 A1). We use these images to train the appearance-based gaze estimator.

Appearance-based Gaze Estimation. Our method uses a full face appearance-based gaze estimator [251, 253] trained on the GazeCapture dataset [118] to predict 3D gaze directions. While collected for gaze estimation on mobile devices, training on this dataset is still beneficial given its large number of participants (over 1’400) and training images.
(over two million). The gaze estimation model is therefore able to better abstract away data-specific biases caused, for example, by variability in user appearance. On GazeCapture the average angular error is around 4.3° while in a cross-dataset setting the error increases to around 5.3° on the MPIIFaceGaze dataset [253], which is state of the art for appearance-based gaze estimation.

**Motion Correlation.** For every image captured by the camera, the appearance-based gaze estimator outputs the 3D gaze direction in terms of *yaw* and *pitch* angles in the camera coordinate system. We intersect this vector with the *XY* camera plane (*z = 0*) to obtain the 2D point of gaze. Given multiple 2D gaze points over time, we use the Pearson product-moment correlation for the *X* and *Y* axis separately to identify which moving target is the most similar. We also account for cases in which it is impossible to calculate the correlation because one series has a variance of zero, i.e. when the trajectory is either a horizontal or vertical line, by rotating that line including the corresponding gaze estimates by 45°, similar to [216]. In the original approach [220], the axis with zero variance is discarded. However, this results in an unnecessary loss of information. When the correlation values for both axes are over a threshold, a candidate target is identified. In the case of multiple candidates, the method selects the one with the maximum sum of both the horizontal and vertical similarity.

**Optical Flow**

The goal of the optical flow component is to complement the gaze estimator and provide more robust correlations between the movement of the eyes and UI elements. This dual-branch approach exploits a fundamental difference between optical flow and gaze estimation. Gaze estimation only considers a single image, i.e. predicts the 3D gaze direction or 2D point of regard from one image frame. Optical flow, on the other hand, can capture the movement between consecutive frames and analyse in which direction the eyes move, and outputs a motion vector instead of an absolute point of regard.

**Face Image Normalisation.** The normalisation step is very similar to the one performed for the gaze estimation task except for one key difference. In the original method [250], the input image was rotated so that the *X*-axis of the head coordinate system is parallel to the *X*-axis of the camera coordinate system. By applying such a rotation, we would also rotate the movement direction of the eyes, which would no longer match the movement of the UI element shown on the screen. Therefore, in the face image normalisation step for optical flow, we transform the input image to a space where the normalised camera points at the centre of the face, yet without cancelling out the head roll, i.e. rotation in the *X* axis. The image is still scaled so that the resulting normalised face has a similar appearance in terms of shape and size across users. Normalising the input
image is beneficial to the optical flow estimation task since the same set of parameters (e.g. similarity threshold) will work for different users or hardware configurations.

**Dense Optical Flow and Motion Compensation.** We use dense optical flow to estimate the pattern of apparent motion between two image frames [58]. If the camera has a clear view of only the eye (e.g. EyeFlow [73]), the optical flow between two consecutive frames provides robust information on eye motion. In contrast, in remote settings, users are at a distance and the camera cannot capture the eye in high resolution. As such, calculating the optical flow between every two consecutive frames introduces too much noise relative to the little actual motion in the eye. In our method, we therefore calculate the optical flow every couple of frames, leading to more robust motion estimates. Empirically, we set the compute-rate parameter to 5.

Our method estimates dense optical flow between two normalised face images (Figure 3.11 B2). In general, optical flow is either due to object motion in the image or camera motion. Image normalisation transforms the input image to an image with fixed camera parameters. Since the position of the normalised camera depends on the detected facial landmarks and the head pose estimate, it slightly differs between frames, which causes the camera to appear as if it was moving. Directly estimating the optical flow would lead to incorrect flow vectors. To address this, we apply motion compensation to decompose visual motion into dominant motion caused by the camera or the background as well as residual motion caused by the user. Specifically, we use a method that estimates a 2D affine transformation model between two frames to cancel out the dominant motion, i.e. the normalised camera motion in our case [93]. The output of this step is the motion compensated optical flow.

**Eye Movement Direction Estimation.** To estimate the eye movement direction, we use the compensated dense optical flow in the eye region. Our method only requires one eye and, in our implementation, we chose the right one. We select the flow vectors that are inside the eye polygon defined by the six eye landmarks (Figure 3.11 B3). We then compute Shannon’s entropy to measure the amount of disagreement among the flow vectors as proposed in [73] and drop frames with an entropy exceeding a certain threshold. The eye movement direction is calculated by averaging the flow vectors in Cartesian space.

**Eye and Object Motion Correlation.** For the same pair of time points for which the optical flow vector was computed, we calculate a motion vector for each object by computing the difference vector between the objects’ positions. We can then calculate the similarity between eye motion and each object’s motion by computing the cosine similarity of the optical flow vector and the object motion vector. This value ranges from -1 for vectors pointing in opposite directions (a 180° angle) to 1 for vectors which point in the same direction (a 0° angle). To obtain similarities between a sequence of optical flow
motion vectors and a set of objects, we compute the mean similarity $\mu_o$ over the similarities of each optical flow vector to the corresponding objects’ motion vectors for each object $o$. While the object motion is a precisely known quantity, the optical flow vectors might contain outliers pointing away from the general direction, which are detrimental to the goal of a robust average similarity. To avoid this effect, we remove similarities which lie outside of the interval $[\mu_o - 2\sigma_o, \mu_o + 2\sigma_o]$, where $\sigma_o$ is the standard deviation over all similarities for object $o$. Finally, among all objects whose mean similarities exceed a certain threshold, we select the one with the maximum mean similarity to the target. If the threshold is not reached at all, we return $None$, i.e. no target was detected or selected.

### Joint Gaze Estimation and Optical Flow

To combine our two sub-components, we calculate target detections from each of them. This can either be a specific object or $None$ if no object was determined to be the target for that window. We simply merge detections by checking whether they agree or not, where $None$ counts as neutral, i.e. if one of the two detections is $None$, we use the other as the merged window detection (Figure 3.11). If both detections are $None$ or they contradict each other, we select $None$. Naturally, if the detections agree, we select that common detection. Running our method results in a sequence of merged detections representing (overlapping) windows of frames. To improve the robustness of our method, instead of simply selecting the first merged detection that is not $None$ as the target, we take several consecutive detections into account. We examined several voting schemes, both on fixed or variable length voting windows of detections. The best voting scheme we found is to select the object which is the first to be jointly detected as target three times, i.e. obtains three votes. These three detections do not have to be in direct succession.

Overall, our method has several free parameters: the window size, i.e. the number of image frames in a window; the stride, i.e. the number of frames two subsequent windows differ in; the threshold for the gaze correlation values; the threshold for the cosine similarity in the optical flow component; and the number of necessary votes for a target to be selected when combining detections.

### 3.4.3 Evaluation

#### Evaluation Dataset

For evaluation, we collected a novel dataset from 13 participants (three female, M=28.9 years old, SD=8.06, age range 23 to 50). This dataset was collected with the help of
3.4 Combining Gaze Estimation and Optical Flow for Pursuits Interaction

Figure 3.12. Experimental setup for collecting the evaluation dataset. Study participants were seated in front of a 24” computer display to which we attached (A) a Microsoft LifeCam Cinema Business RGB webcam and (B) a Tobii 4C eye tracker. The on-screen targets (C) were shown in red on a light-coloured background. Figure source: [106]

Alexander Kayed, as part of his Master’s thesis [106]. Participants were seated in front of a 24” computer display to which we attached a Microsoft LifeCam Cinema Business webcam (recording resolution: 1280x720 px at 30 fps) on the top and a dedicated eye tracker, the Tobii 4C, which we use for comparison, on the bottom edge as recommended by the manufacturer. The experimental setup is shown in Figure 3.12. We did not constrain the participants in any way in terms of position, distance to the screen, or head movement. The only constraint was imposed by the Tobii eye tracker, which has an operating distance between 50 cm and 95 cm. When users are 75 cm away from the screen, the tracking box is 40 x 30 cm, as defined by the manufacturer. Figure 3.13 shows a few sample images from one of the study participants.

Figure 3.14 shows the distribution of the head pose angles (pitch and yaw) in the normalised camera space. Figure 3.14a is a scatter plot of all the values as obtained by estimating the head pose from 351,039 image frames, while Figure 3.14b shows a power-law normalisation over the entire head pose range. In terms of head movement, the majority of the image samples in the normalised camera space cover vertical angles between 0° and −25° (min −40.83°, max 73.6°) and horizontal angles between −20° and 20° (min −37.13°, max 31.95°). Most vertical angles, i.e. the pitch, are negative due to the head’s position relative to the camera: The camera was mounted above the screen while the participants were looking at the screen. The user’s distance to the screen and
camera resolution also influence the number of image pixels available in the eye region. In our dataset, the right eye, used for the optical flow computation, filled an area of 243.3 px (SD=98.8 px).

![Sample Images](image)

Figure 3.13. A few sample images from the dataset showing one of the participants of the study (the author of this dissertation). The remaining study participants did not consent to the public release of the evaluation dataset. Images were recorded at 1280x720 px.

![Scatter Plot and Heatmap](image)

Figure 3.14. (a) Scatter plot: Distribution of pitch and yaw values of the users’ head pose in the normalised camera space. (b) Heatmap: Distribution of pitch and yaw values of the users’ head pose in the normalised camera space.

Before each session, we calibrated the eye tracker for each participant using the routine provided by the manufacturer. Following prior work on pursuits interaction [220], a single session consisted of several experiments with different trajectory types:

1. **Linear** trajectories moved back and forth on a straight line (tilted at a certain angle)
2. Circular trajectories followed a circle, with all competitors being on the same circle.

3. Orbital trajectories, which also followed a circle, but competitors moved on circles with different radii.

4. Alternating orbital trajectories, where every other object moved in the reverse direction.

5. Rectangular trajectories followed a rectangle, with all competitors on the same rectangle at an equal distance apart from each other.

For each type of trajectory, we used a single size, as both similarity measures we employ are invariant to scaling. However, except for rectangular trajectories, we collected data at two different velocities, which were determined in pilot experiments. The details are given in Table 3.1. The screen showed a single red dot following the specific trajectory that participants were asked to follow while we recorded the video from the webcam and gaze data from the Tobii eye tracker (as shown in Figure 3.12). To not distract participants while doing the study, only the actual target was shown.

For each trajectory type, we conducted different trials in which we varied the starting positions and the movement direction of the dot. We created a variable number of trajectories based on the target trajectory to simulate the presence of a certain number of competing targets in the interface. In the simulation, for any given number of competitors, we maximised the difference within the set of objects, consisting of the target and the competitors. For linear trajectories, we created a new trajectory for each competitor and rotated it around the centre point of the plane so that the angle between the trajectories was maximised. For circular and orbital trajectories, we maximised the phase shift between

<table>
<thead>
<tr>
<th>Type</th>
<th>No. of trials</th>
<th>Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>8</td>
<td>height-of-screen/π m/s</td>
</tr>
<tr>
<td>Linear fast</td>
<td>10</td>
<td>1.5x linear speed</td>
</tr>
<tr>
<td>Circular</td>
<td>7</td>
<td>0.8 rad/s (~45°/s)</td>
</tr>
<tr>
<td>Circular fast</td>
<td>4</td>
<td>1.1 rad/s</td>
</tr>
<tr>
<td>Orbits</td>
<td>11</td>
<td>1.1 rad/s</td>
</tr>
<tr>
<td>Alternating orbits</td>
<td>11</td>
<td>1.1 rad/s</td>
</tr>
<tr>
<td>Orbits fast</td>
<td>16</td>
<td>1.4 rad/s</td>
</tr>
<tr>
<td>Alternating orbits fast</td>
<td>16</td>
<td>1.4 rad/s</td>
</tr>
<tr>
<td>Rectangle</td>
<td>3</td>
<td>as for linear</td>
</tr>
</tbody>
</table>

Table 3.1. We study four types of trajectories with different characteristics: Linear, circular, orbits, and rectangular. The velocity for each trajectory type was set empirically after initial testing. The number of trials is per participant.
the objects. For example, for a circular trajectory with two competitors, there were three objects used in the evaluation, each one with a phase shift of 120°. Similarly, competitors for rectangular trajectories travel on the same shape as the target, but at a distance from each other.

Baseline Methods

To evaluate the performance and robustness of our method, we compared it to the following two baselines:

1. **Pursuits with Tobii 4C.** We implemented the original pursuits technique [220] that correlates the X and Y axes of the Tobii 4C gaze estimates and object trajectories independently. If both correlation values are above a threshold, the target with the maximum correlation sum is selected. The threshold was set to 0.9.

2. **EyeFlow.** We implemented the original method [73] that was not designed for remote settings but head-mounted cameras. This is a fundamental difference, yet it is the closest to our work. To simulate this configuration, we cropped a fixed eye patch from the original image. Using the six eye landmarks, we calculated the centre, height and width of the eye patch. The cropped image was resized to 72x120 px (similar to other works [252], where the eye patch size was 36x60 px). The location of the eye crop was calculated every ten frames to account for possible movement of the participants.

In the implementation of our method, for the gaze estimator, the correlation threshold was set to 0.6, while the cosine similarity threshold for optical flow was set to 0.8. The window size for all methods was set to 30 frames (i.e. 1 s for our webcam) with a stride of one. When combining gaze estimation and optical flow, the number of necessary votes was three. In all experiments that follow, we evaluated the different methods in terms of the mean and standard deviation of the accuracy across participants. The accuracy is calculated as the total number of correctly detected targets divided by the total number of trials. If no target was detected during a trial, it was counted as a false detection negatively influencing the accuracy.

Pursuits Detection Performance

We first use all of the recorded experiments and run the different methods for up to 15 competitors. As explained in Section 3.4.3, this means that for each number of competitors, we simulate competitor trajectories based on the displayed target trajectory and run each
3.4 Combining Gaze Estimation and Optical Flow for Pursuits Interaction

Figure 3.15 shows the results of this analysis for our method, *Pursuits with Tobii 4C*, and *EyeFlow*. The coloured bands depict the standard deviation across participants for each method and number of competitors. For zero competitors, we measured whether the methods were able to detect the target at all. As shown by the figure, for all methods, the accuracy decreases as the number of competitors increases. This is because the detection problem becomes more challenging with a greater number of objects to choose from. *Pursuits with Tobii 4C* achieves the highest accuracy for any number of competitors, with an accuracy of over 95% for up to three competitors, nearly linearly declining to 60% for 15 competitors. It also exhibits a lower standard deviation than our method, which also shows a high accuracy of over 90% for up to three competitors, however, then declines slightly steeper to 40% for 15 competitors. Nevertheless, our method clearly outperforms *EyeFlow*, which only achieves a very low accuracy overall. From six competitors onwards, our method shows a slight zig-zag trend, which is caused by performance differences for even and odd numbers of competitors for alternating orbital trajectories, as explained in the next section.
Influence of Trajectory Type and Speed

We further examined the influence of the type and speed of a trajectory on the accuracy. We ran the same evaluations as before, however, aggregated by trajectory type or velocity. Figure 3.16 shows the accuracy of the different methods per trajectory type and the coloured bands indicate the standard deviation across participants. The results for circular and orbital trajectories reflect the overall results and performance ranking among the three methods, with perfect or close to perfect detections for Pursuits with Tobii 4C for few competitors. When there are more than three competitors, our method outperforms Pursuits with Tobii 4C for linear trajectories.

Note that for alternating orbital trajectories, there are performance differences depending on whether the number of competitors is even or odd. Because we keep the phase shift consistent with the shift for circular or non-alternating orbits, the minimum phase shift between a competitor moving in the same direction as the target and the target itself is smaller for an even number of competitors than for a close odd number (e.g. Figure 3.17).
3.4 Combining Gaze Estimation and Optical Flow for Pursuits Interaction

For seven competitors, the phase shift between the target and the closest object moving in the same direction is 90°. For six competitors, however, this difference is only 51°, making discrimination more challenging although the number of competitors is smaller. Our method also outperforms Pursuits with Tobii 4C for rectangular trajectories with up to seven competitors; however, for both methods, the standard deviation is very high. As in the overall evaluation, EyeFlow achieves a poor accuracy below 20% independent of trajectory type.

For each type of trajectory, we recorded two different experiments, displaying the target at two different velocities, except for the rectangular case, which we, therefore, excluded in this specific comparison. The results depending on the velocity settings are illustrated in Figure 3.18. While the average performance for all three methods is similar, the standard deviation is smaller for Pursuits with Tobii 4C and Ours at higher velocity.

Ablation Study

Our method detects pursuits by jointly analysing the gaze direction and optical flow in the eye region. In this experiment, we evaluated the performance of each of the two independently and how much each component contributed to the joint result. Figure 3.19 shows the results of the same procedure as in Section 3.4.3 for our combined method and for single components, Gaze Estimation and Optical Flow. For the single components, we employ the same voting strategy as for the combined method. When the number of competitors is low, the results for Gaze Estimation alone are similar to Ours. With a larger number of competitors, the performance of the Gaze Estimation component declines to

![Figure 3.17. Alternating orbital trajectories for six and seven competitors. For seven competitors, the phase shift between the target and the closest object moving in the same direction is 90°. For six competitors, however, this difference is only 51°, making discrimination more challenging although the number of competitors is smaller.](image)
Figure 3.18. Pursuit detection accuracy at normal and higher velocity. We excluded the rectangular one from this evaluation since we only collected data for a single speed setting.

Figure 3.19. Ablation study comparing the pursuit detection performance of Ours to the individual components of our method, Gaze Estimation and Optical Flow.
3.4 Combining Gaze Estimation and Optical Flow for Pursuits Interaction

the level of the *Optical Flow* results. In such cases, *Ours*, the combined method, maintains a positive performance difference of about 8 to 10 percentage points.

To investigate the contribution of each component to the overall results, we removed the voting scheme in the last processing step and simply returned the first merged detection that is not *None*. This detection is then counted towards the contribution of the component that provided it, or as a common contribution if both agreed. Figure 3.20 shows these contributions. We separated the contributions into correct, below the accuracy curve, and incorrect, above. The correct contributions naturally sum up to the accuracy of the combined method. Note that since we removed the voting strategy to obtain the clear origin of the detection, the results are different from those in Figure 3.15. The results show that, overall, especially for a small number of competitors, *Gaze Estimation* has the strongest influence for both correct and incorrect detections, while for larger numbers, the contributions are balanced.

**Response Time**

Performance in terms of accuracy only highlights one aspect of a method’s capabilities. For interactive systems, another important characteristic is the response time that can be

![Figure 3.20. Ablation study quantifying the contribution of each component, i.e. *Gaze Estimation*, *Optical Flow*, or *Common*, towards the final decision. The bars below the accuracy curve, which is without the voting strategy, indicate correct contributions. The ones above show incorrect contributions, including a segment when nothing was detected.](image-url)
defined as the amount of time needed until a pursuit is detected. In our evaluations, we used a window size of 30 frames, which amounts to one second for the camera we used. As such, the lower bound for the response time is 1 s, which is the time needed to fill the first window.

We calculated the response time for each trial and then average across all trials. When the number of competitors is between 0 and 7, the average response time was 1.44 s (SD=0.01 s) for Tobii, 1.45 s (SD=0.06 s) for Gaze Estimation, 1.94 s (SD=0.07 s) for Ours, and 2.34 s (SD=0.08 s) for Optical Flow. For 8 to 15 competitors, the mean response time was 1.43 s (SD=0.01 s) for Tobii, 1.36 s (SD=0.01 s) for Gaze Estimation, 2.27 s (SD=0.11 s) for Ours, and 2.22 s (SD=0.01 s) for Optical Flow. We did not analyse the response time for EyeFlow since this method has a very low overall accuracy when used in remote settings, which makes it practically unusable.

Runtime Analysis

We evaluated the runtime of our pipeline on a desktop PC equipped with an Intel i7-4790K CPU @ 4.00GHz and an Nvidia GeForce 1080 ti GPU. Face detection and landmark localisation use state-of-the-art neural networks [43, 247], which require a GPU. Face detection takes about 70 ms for an image of 1280x720 px (like the ones used in our evaluations). Reducing the resolution to 640x360 px decreases the face detection runtime to about 20 ms per image, but we did not investigate the effect of lower resolution images on performance. Landmark localisation including face image normalisation takes around 570 ms for a window of 30 images (around 19 ms per frame). The gaze estimation CNN needs around 50 ms for a batch of 30 images (around 1.7 ms per frame). Optical flow estimation takes around 45 ms, motion compensation around 38 ms and, because of the compute-rate=5, they are calculated six times in a window of 30 frames. The two components of our method can run in parallel, gaze estimation on the GPU and optical flow on the CPU. In our current implementation, the runtime is bounded by the gaze estimation pipeline, which takes around 93 ms per image.

3.4.4 Discussion

In this work, we proposed to combine appearance-based gaze estimation and optical flow in the eye region to jointly detect pursuits without the need for any special-purpose eye tracking equipment. Our method only requires a single RGB camera, which is included in an ever-increasing number of devices [109].
Evaluations on a novel real-world dataset showed that our method could robustly detect the correct pursuit target with over 90% accuracy for up to four moving UI elements, independent of trajectory type (Figure 3.15). It not only outperforms the current state of the art EyeFlow [73] but is also competitive to a commercial, consumer-grade eye tracker. EyeFlow shows a low overall accuracy, and this could be, in part, because of the optical flow calculation: EyeFlow was proposed for head-mounted settings where the camera has a close-up, high-resolution view of the eye. In remote settings, users are at a distance, and only a few pixels are available in the eye region to calculate the flow (in our case, around 240 px per eye). Therefore, optical flow calculations across two consecutive frames will lead to increased noise and incorrect estimates of eye movement direction. By calculating the dense optical flow every five frames, our method is able to better approximate the eye motion since the change between two images will be more significant. For a large number of moving UI elements, all methods suffer from a drop in performance, including the one that uses a dedicated eye tracker. Other research-grade eye trackers such as the Tobii Pro Spectrum may achieve better performance. However, they are not only expensive but also not targeted towards end users.

In all our evaluations, to ensure reliable pursuit samples, we only showed participants a single target that they had to follow (similar to [220]) and generated all the competitors post hoc. In a real application, multiple moving elements might be present at the same time, potentially distracting users. However, in a multi-target environment, neuroscience literature suggests that the brain suppresses other non-tracked targets [128] and the perception of competitors is reduced during smooth pursuit [113]. Moreover, a field study in a real environment has shown that users can reliably select the desired target in spite of multiple visible competitors [220].

An in-depth analysis showed that there are also differences between the methods when looking at different trajectory types. As expected, when the number of moving targets is low, the Pursuits with Tobii 4C method performs best, yet ours follows closely. More interesting is that for linear trajectories, when the number of targets increases, our method shows increased robustness and even outperforms Pursuits with Tobii 4C (Figure 3.16). This could be explained by the use of the cosine similarity in the optical flow component. It appears that our method also outperforms the dedicated eye tracker for rectangular trajectories. However, given the little number of trials per participant, it is difficult to draw general conclusions. In our analysis, we also evaluated the influence of the target velocity on all the methods’ performance (Figure 3.18). The overall average accuracy is similar, yet the standard deviation is smaller, which implies that faster targets lead to more stable results across participants.

To further understand how and which of the two components contribute to the final results, we did an ablation study (Figure 3.19 and Figure 3.20). Of the two components,
Chapter 3 Eye-based Human-Computer Interaction

Gaze estimation has a higher overall accuracy than Optical Flow. However, when the number of targets is five or more, the methods perform quite similarly. Nevertheless, Figure 3.19 clearly shows that by combining both of them, the accuracy can improve by as much as 10 percentage points.

We also compared the different methods in terms of response time that is of particular practical relevance in interactive systems. Pursuits with Tobii 4C is, as expected, the fastest of all the methods with around 1.4 s. Our method is between Gaze Estimation and Optical Flow with a response time between 1.9 and 2.2 s. This represents a trade-off between accuracy and response time. Only using gaze estimates will lead to a faster, but not as accurate decision.

Based on the findings from our evaluations, given the high accuracy for up to four moving UI elements, our method could be applied in several practical applications. For example, similar to Orbits [55], a music player could have pursuit-enabled controls. On a public display, a few moving targets could show tourists interesting facts about a city. Another, less obvious application could be eye tracker calibration. Similarly to how pursuits can be used to calibrate an eye tracker [174], our method could be used to collect implicit calibration samples for personalised appearance-based gaze estimation [248].

Limitations

While our work is the first to propose a method to detect pursuits in unconstrained remote settings with a single RGB camera, it also has several limitations that we will address in future work. First, the application design space is limited by the number of targets that can be detected reliably. As such, our method is robust for up to four targets of the same kind. It would be interesting to investigate not only combinations of different trajectory types but finding optimal combinations that would maximise the overall performance for a given number of targets. Besides predefined trajectories, we can imagine users creating personalised trajectories, e.g. by drawing them by hand on a smartphone or tablet, thereby designing their own interfaces. Second, the performance may be influenced by users’ blinking. Our recordings naturally also contain such events and, while we did not investigate the effect of blinks, our results implicitly include them. Blinks may lead to incorrect gaze estimates or optical flow vectors, which, if filtered out, may increase performance. Catch-up saccades, which correct the eye’s position relative to the target, are another factor to consider when designing interactions with pursuits. This is a consequence of targets moving too fast (e.g. 120°/s), which then leads to a drop in performance [55]. Third, the fact that the participants knew they were taking part in a study might have altered their behaviour, e.g. in terms of head movement. A follow-up study could examine performance in a real-world deployment, such as playing a game on a public display [220].
Lastly, we also intend to optimise the runtime in order to obtain a real-time system. Face detection, which dominates the runtime, could be replaced with a fast object tracker such as KCF [74].

### 3.4.5 Conclusion

In this work, we proposed a novel method to detect pursuits by combining appearance-based gaze estimation and optical flow to jointly analyse the eye movement dynamics. Our method only requires images captured with a single off-the-shelf camera placed at a distance from the user. Through in-depth evaluations on the data collected from a 13-participant user study, our method shows a significant performance increase in comparison to the current state of the art. Moreover, for up to four moving UI elements, our method achieves an average accuracy of over 90%, which is competitive with the performance of a dedicated eye tracker. Taken together, these results are significant because they, for the first time, point towards a new class of methods that enable pursuit interactions with nothing more than a standard off-the-shelf RGB camera.
Quantification of Users’ Visual Attention in Mobile HCI

This chapter is based on the following publications:


4.1 Introduction

In recent years, the number of digital interfaces competing for users’ attention has rapidly increased. Consequently, actively managing users’ limited and valuable attentional resources has emerged as a fundamental research challenge in human-computer interaction (HCI). With mobile devices being pervasively used, this challenge is particularly pressing in mobile HCI where attentive behaviour has become highly fragmented [166, 206]. Despite its significance, little research has focused on managing attention during mobile interactions. This is, for one, because of a lack of a single commonly accepted definition and understanding of attention [5]. One widely accepted characterisation distinguishes between covert and overt attention. Covert attention is the cognitive process of shifting one’s mental focus of attention. Its measurement requires special-purpose hardware as well as carefully constrained settings and stimuli [69]. In contrast, shifts of overt attention are practically more useful for HCI purposes because they involve eye movements that can be measured using cameras. It is for this reason that only overt attention has been widely studied, e.g. in the context of attentive user interfaces (AUIs) [218]. Therefore, quantifying the allocation of so-called overt visual attention during mobile interactions – for example when, how often, or for how long users look at their device – is a crucial step towards AUIs that actively manage users’ limited attentional resources [31].

Previous research on this important topic has so far focused on alleviating negative effects of fragmented visual attention, e.g. by identifying opportune moments to interrupt the user [242] or by predicting distractiveness of mobile notifications [48, 150, 177]. While these tasks deal with users’ visual attention indirectly, research on quantifying attention directly is scarce [221]. The main reason for this is the lack of accurate and robust methods to study attentive behaviour during everyday mobile interactions without special-purpose and obtrusive eye tracking equipment [206]. As a consequence, prior work has instead relied on cumbersome and time-consuming manual annotation [166], analysis of application usage logs [98], or self-reported questionnaires collected through methods like experience sampling [214]. However, all of these approaches are only proxies to attention and, as such, temporally too coarse or can even negatively impact the naturalness of users’ attentive behaviour [221]. Recent advances in learning-based gaze estimation [253] and automatic eye contact detection [16, 249] point the way towards sensing and analysing user visual attention in-situ, i.e. while users use their mobile devices during everyday routine [31, 218]. In contrast to gaze estimation where the goal is to predict a precise 3D gaze direction or 2D point of regard, eye contact detection is the binary task of detecting if a user looks at a target or not. As such, as far as AUIs are concerned, eye contact is currently the most important measure of overt visual attention but the potential of such an approach for unobtrusive measurement of fine-grained and accurate (visual) attentive behaviour in mobile HCI has not yet been realised.
In this chapter, we first provide an investigation into the feasibility of quantifying visual attention during everyday interactions with mobile devices using automatic eye contact detection. We evaluate the influence of face and eye visibility on eye contact detection performance, given that the best performing methods require the face and facial landmarks but users’ face was shown to be visible only around 30% of the time [110]. We study the impact of head pose on eye contact detection performance, which is particularly challenging in mobile settings in which devices are held and being looked at in a variety of ways, including while on the go. We then demonstrate the need for more accurate gaze estimation and its importance to the eye contact detection task.

Guided by our investigation into the key challenges of sensing visual attention during mobile interactions, we present a novel method to detect eye contact with mobile devices accurately and robustly using the integrated front-facing camera (Figure 4.1). To improve face detection and facial landmarks localisation, including partial occlusions, our method leverages a multi-task convolutional neural network (CNN) [247] together with a state-of-the-art hourglass neural network [43]. The users’ head pose is robustly estimated from 68 facial landmarks and further stabilised using a Kalman filter. By introducing head pose thresholding in a normalised space, our method is able to cope with extreme head poses independent of user or device. Finally, to overcome variability across devices, users’ appearance, and environmental conditions, our method leverages an appearance-based gaze estimator trained on the large-scale GazeCapture dataset [118].

Figure 4.1. We present a novel method to detect eye contact in images captured with the front-facing camera of mobile devices. Eye contact is a fundamental measure of overt visual attention and our method enables, for the first time, the calculation of higher-level attention metrics during everyday mobile phone interactions.
Having a method to robustly detect eye contact during everyday mobile device interactions enables us to study and understand visual attentive behaviour in the wild. Towards this goal, we conducted a two-week in-the-wild data collection of video snippets using the front-facing camera of 32 mobile phone users. Our *Everyday Mobile Visual Attention (EMVA)* dataset contains 14,322 videos, totalling around 472 hours, as well as associated meta-data, sensor data, and device usage logs (Figure 4.2). Additionally, using crowd-sourcing functionality integrated into the app, we collected 10,759 annotations for eye contact with the device that were manually annotated by at least two different app users. In contrast to existing datasets that only contain images and associated sensor and meta data collected at discrete points in time [110], our dataset is the first to capture the temporal dynamics of attention allocation during mobile device use. To democratise further work in this area of research, we have made the dataset publicly available: [http://www.emva-dataset.org/](http://www.emva-dataset.org/). In the public release of the dataset, images that contained the faces of bystanders were masked given that we did not have their approval to publish their personal data.

Analysing such a large dataset manually is challenging. To gain insights into attentive behaviour during mobile interactions, we used our method for eye contact detection as a

![Figure 4.2](image-url)
4.2 Related Work

tool to quantify overt visual attention. Detecting when users look at their devices provides rich insights into attentive behaviour and is the basis for key attention metrics, such as the duration of sustained visual attention or the number of attention shifts [206]. We analysed several key attention metrics across users, applications, and contexts. Our results show, for example, that the average duration of sustained visual attention per video snippet is around 7 s. Additionally, attentive behaviour is both user and context-dependent and changes over the course of the day.

To summarise, the specific contributions of our work in this chapter are:

1. We discuss the main challenges and sources of error associated with sensing visual attention on mobile devices in the wild, including the impact of face and eye visibility, the importance of robust head pose estimation, and the need for accurate gaze estimation.

2. We present a novel method to detect eye contact in images captured with the front-facing camera of off-the-shelf mobile devices. Our method does not rely on any special-purpose eye tracking equipment, does not require user or device-specific calibration, and is fully automatic – no manual and tedious data annotation is needed. Through evaluations on two current datasets of natural mobile interactions [59, 110], we demonstrate significant improvements in performance and robustness for eye contact detection across mobile devices, users, or environmental conditions.

3. We provide EMVA, the first multimodal dataset that captures the temporal dynamics of attention allocation during mobile device interactions embedded in everyday routine.

4. Leveraging our method for automatic eye contact detection in mobile settings, for the first time, we analyse and quantify visual attention in-situ without the need for obtrusive eye trackers or tedious and error-prone manual annotations. We gather insights and provide detailed analyses of users’ attentive behaviour.

5. Lastly, we discuss key insights from our analyses that highlight the potential and inform the design of future mobile attentive user interfaces.

Our work relates to previous work on 1) gaze estimation, 2) eye contact detection, 3) user behaviour modelling on mobile devices, and (4) visual attention sensing.
4.2.1 Gaze Estimation

Estimating human gaze from natural images is a core research area with broad applications in many different domains. Early works relied on pupil detection and infrared illumination [71], which restricted them to short distances and stationary settings [158, 255]. Other image-based approaches were either feature-based [84, 233], which required feature engineering from eye images, or model-based [212, 224] that used a 3D model of the eye. While such classical methods have been used to estimate users’ gaze direction on mobile devices such as tablets [84, 122, 233], they require user, device, or pose-specific calibration. This reduces usability and makes these methods particularly impractical for scenarios in which it is important that the system interferes with the user as little as possible – such as for attention monitoring during mobile interactions. Additionally, these methods were shown to suffer severely from low gaze estimation accuracy and robustness due to their inability to cope with the large variability in facial appearance, illumination conditions, and distances between camera and user [251].

A more promising approach are learning-based gaze estimation methods that leverage large-scale datasets and machine learning to directly regress from eye images to gaze directions [118, 184, 252, 252, 253, 254]. Learning-based methods have significantly advanced in recent years and already outperform traditional approaches in terms of robustness with a gaze estimation performance of around 4-6° in calibration-free settings. User or device-specific model adaptation can bring such methods closer to the performance of special-purpose eye trackers [248], however, calibration is impractical for sensing in the wild.

4.2.2 Eye Contact Detection

In contrast to gaze estimation, eye contact detection is the binary classification task of predicting whether a user is looking at a particular target or not. This target could be the camera itself, a real-world object, another person, or the screen of a stationary or mobile device. While potentially easier than gaze estimation computationally, and therefore promising for low-accuracy applications, eye contact detection also faces two fundamental challenges [249]: First, without prior knowledge on the size and location of the target with respect to the camera as well as target-specific training data, training an eye contact detector that works also for small targets (e.g. mobile phone) or targets located close to the camera is as difficult as training a generic appearance-based gaze estimator. Second, robust and accurate eye contact detection also has to handle the large variability of different environments and preparing suitable training data to cover all the possible configurations is impractical.
Early works on eye contact detection used custom camera-equipped hardware and LEDs attached to target objects to detect whether users were looking at the camera or not [46, 197, 218]. Similarly, Selker et al. proposed a glass-mounted device which, upon detecting eye contact with the device, tried to initiate an information exchange between the user and the gaze target [195]. However, the need for custom hardware fundamentally hindered the wide applicability of these methods.

More recent works therefore focused on using only off-the-shelf cameras for eye contact detection. Smith et al. proposed GazeLocking, a supervised method for detecting eye contact with a remote camera [201]. Ye et al. used head-mounted wearable cameras and a learning-based approach to detect eye contact [241]. Such supervised approaches work well given enough training data but manually collecting and annotating large-scale datasets is impractical. Zhang et al. were the first to introduce an unsupervised approach to detect eye contact in stationary settings [249]. This approach has three key benefits. First, the method is fully unsupervised, there is no need to manually label data. Second, it only requires images captured with a standard RGB camera. Third, their method is the first to not detect eye contact with the camera itself but a target (display and another person) in close proximity to the camera, significantly broadening possible applications. Extending on this work, Mueller et al. [161] proposed an eye contact detection method which additionally correlates people’s gaze with their speaking behaviour by leveraging the fact that people often tend to look at the person who is speaking.

With the exception of [249], all of these methods were limited to stationary settings and assumed that the camera always had a full view of the user. None one them focused on interactions with a mobile device. As demonstrated in previous works [110], assumptions commonly made in stationary settings no longer hold when using the front-facing camera of mobile devices. We are first to fill this gap and to propose an eye contact detection method which address challenges specific to mobile interactive scenarios.

### 4.2.3 Modelling User Behaviour on Mobile Devices

Modern mobile devices are powerful and sensor-rich miniaturised computers capable of sensing the users’ environment and behaviour, including attention. Given the fragmented nature of mobile interactions, which can last as little as four seconds [166], a significant body of prior work has focused on predicting user interruptibility from device integrated sensors [36, 62, 165, 175]. A complementary task is concerned with attentiveness and receptivity towards messages and notifications [48, 177]. Smartphones and, more recently, smartwatches can be used to estimate boredom [178] or different levels of user engagement [45, 144, 213]. Such behavioural models can be used to adapt the possible interaction
modalities based on the users’ context [163, 179]. The Attention Meter is a software tool which calculates a score based on different behavioural traits taking into account head movements or facial expressions [125]. Mobile eye trackers can also be used to better understand mobile device interactions [168] and reveal, for example, boredom in outdoor settings [114]. A combination of device-integrated sensors and body-worn cameras can predict shifts of attention before they happen [206]. A promising approach to avoid the need for special-purpose eye tracking equipment are methods based on saliency [28, 64] that aim to predict regions of interest that draw attention in images or videos. Scene driven saliency models [90] can monitor a person’s attention on a display through saliency maps as probability distributions for the gaze locations [210]. However, methods based on saliency are not suitable for mobile device interactions, i.e. not (yet) a viable replacement for eye tracking. Besides device-integrated sensors and cameras, another current standard for modelling user behaviour is through self-reports and experience sampling [214]. Experience sampling together with questionnaires and smartphone logs can be used to understand user attentiveness to mobile notifications [150] or while consuming video content [42]. However, a significant challenge with such methods is finding the right moment to question the user without influencing the current level of attention [101].

In our work, we are first to study the dynamics of visual attention allocation in-situ from video recordings collected during everyday mobile device interactions. More specifically, our data collection did not require any bulky eye tracking equipment but only off-the-shelf smartphones with unobtrusive, integrated front-facing cameras. Another important distinction from prior work is that our data collection did not constrain the participants in any way, neither through the need for self-reports nor experience sampling approaches. Both aspects contribute significantly to the naturalness of the recorded user behaviour and the ecological validity of our findings.

4.2.4 Visual Attention Sensing

Estimating where people look is a long-standing research challenge in HCI [31]. Early works required special-purpose or custom hardware, such as EyePliances that respond to visual attention on everyday objects, such as a lamp [197]. The same concept has been extended to detect when people looked at one another [46] or to facilitate media playback when people looked at their devices [47]. The AttentivU glasses used electroencephalography as well as electrooculography sensors to measure a person’s attentiveness in real-time and provided feedback when their attention was low to increase user engagement [117]. While such approaches work well in constrained settings, the need for special-purpose equipment fundamentally limits possible use-cases.
4.3 Key Challenges in Quantifying Mobile Visual Attention

At the same time, mobile devices are equipped with ever more high-resolution cameras and powerful computational capabilities and have consequently increasingly been used as a platform for mobile attention sensing. For example, EyePhone was one of the first systems to introduce an attentive UI that tracked the user’s eye and could detect blinks [151]. The Visual Attention Detection with a Smartphone (VADS) system detected where users were looking in a scene by leveraging both cameras of a smartphone; the front-facing was used to estimate the user’s gaze direction while the rear one observed the scene [94]. SwitchBack was a system which only tracked the relative movement of the eye and, with prior knowledge of the task, detected distractions and further assisted users to continue where they left off [139]. EyeTab was an early model-based approach for gaze estimation on unmodified tablet computers [233]. ScreenGlint was also a model-based approach which exploited the reflection of the screen and, with calibration, achieved an angular error of around three degrees [83].

A study on the applicability of computer vision based gaze estimation methods highlighted that, in general, such methods have a low mean accuracy and a high error rate [77]. More promising are recent learning-based gaze estimation methods because they can learn robust gaze estimators from large-scale datasets [253] (as discussed previously in Subsection 4.2.1). A work by Sugano et al. proposed to aggregate gaze estimates obtained using such a method across multiple users, allowing them to still calculate joint attention distributions on a public display [211]. Advances in learning-based gaze estimation have also spurred activity on the related yet still different eye contact detection task (Subsection 4.2.2). In our work, we leverage these methodological advances in learning-based gaze estimation and eye contact detection to extract, for the first time, key visual attention metrics from everyday mobile device interactions.

4.3 Key Challenges in Quantifying Mobile Visual Attention

In this section, we provide a fundamental analysis of the key challenges associated with sensing visual attention in the wild. In the context of AUIs, the most important measure of overt visual attention is eye contact because it enables quantifying when, how often, or for how long users look at their mobile device. Eye contact detection is promising given that it is computationally simpler than gaze estimation but fully sufficient to analyse user attention in mobile settings. Both fully-supervised [201] and unsupervised methods [249] have been proposed, however, in our analyses, we opted for the method by Zhang et al. [249], which, besides achieving state-of-the-art performance does not require manual
annotation. The single assumption of their approach is that the camera is next to the object of interest – which is also true for common mobile devices.

**Everyday Eye Contact Detection.** The method by Zhang et al. [249] first detects the user’s face with a face detector. Afterwards, a landmark detector finds six landmarks inside the face bounding box. Given these six 2D facial landmarks, the image is normalised [250] and a state-of-the-art gaze estimation CNN [253] is used to predict the 2D gaze location. These 2D gaze locations are sampled for clustering under the assumption that each cluster corresponds to one eye contact target. The correct cluster, i.e. the one representing eye contact, is the one closest to the camera, which is in the centre of the camera coordinate system. After clustering, samples belonging to the target cluster will be labelled as positive, while all the others will be labelled as negative. These images can now be used to train a binary Support Vector Machine (SVM) as the eye contact classifier. The SVM input is a feature vector extracted from the appearance-based gaze estimation CNN. In our implementation, we use the last fully connected layer that produces 4096 features. Clustering is only necessary once, for training. For inference, the input to our method is the feature vector extracted using the CNN (i.e. the 4096-dimensional feature vector).

In the analyses that follow, we implemented the method by Zhang et al. [249]. We used the dlib [257] CNN face detector and the dlib 68 landmark detector. The full face appearance-based gaze estimator, which is part of the eye contact detection method, was trained on the MPIIFaceGaze dataset [253].

### 4.3.1 Evaluation Datasets

The evaluations presented in the sections that follow were conducted on two challenging, publicly available mobile interaction datasets with complementary characteristics in terms of users, devices, and environmental conditions: the Mobile Face Video (MFV) [59] and the Understanding Face and Eye Visibility (UFEV) dataset [110].

**Mobile Face Video Dataset (MFV).** This dataset includes 750 face videos from 50 users captured using the front-facing camera of an iPhone 5s. During data collection, users had to perform five different tasks under different lighting conditions (well-lit, dim light, and daylight). From the five tasks available in the dataset, we selected the “enrollment” task where users were asked to turn their heads in four different directions (left, right, up, and down). We picked this task because it enabled us to collect both eye contact and non-eye contact data. From this subset (1 video per task \(\times\) 3 sessions \(\times\) 50 users), we randomly sampled around 5000 images. Two annotators manually annotated these images with one of the three possible labels: eye contact, no eye contact, or unknown. The unknown class was used when both annotators were unsure whether the person in the image was looking
4.3 Key Challenges in Quantifying Mobile Visual Attention

at their device or not and such images were discarded from the evaluation subset. In the evaluations that follow, we used 4,363 images. 58% were labelled as having eye contact and the remaining as no eye contact. This dataset is challenging because it contains large variations between users, head pose angles, and illumination conditions.

**Understanding Face and Eye Visibility Dataset (UFEV).** This dataset consists of 25,726 in the wild images taken using the front-facing camera of different smartphones of ten participants. The images were collected during everyday activities in an unobtrusive way using an application running in the background. Similarly to MFV, two annotators manually labelled a random subset of 5,791 images. We only sampled images where at least parts of the face were visible (which was the case for 14,833 photos). 17% of the frames were labelled as having no eye contact, while 83% as having eye contact. In contrast to the previous dataset, these samples exhibit a class imbalance between positive (eye contact) and negative (no eye contact) labels. This dataset is relevant because it highlights a challenge specific to mobile interactive scenarios, namely the partial visibility of face and facial landmarks as captured by the front-facing camera of a smartphone.

### 4.3.2 Overall Performance

Before investigating the different factors that influence the accuracy and robustness of the method, we first evaluated performance in terms of the Matthews Correlation Coefficient (MCC). The MCC score is commonly used as a balanced performance metric for binary classification problems. It is particularly useful on imbalanced datasets, where it is more informative than the F1 score or the accuracy because it takes into account all four classes of the confusion matrix. For example, in the UFEV dataset, from the manually annotated images, 83% of them are positive eye contact and only 17% represent non eye contact. An MCC score of +1.0 indicates a perfect classifier, while -1.0 indicated total contradiction between the observations and predictions. A value of 0 is equivalent to random guessing.

Overall, on the UFEV dataset, the method achieves an MCC of 0.349 (SD=0.17) in a leave-one-person-out cross validation, i.e. the eye contact detector was trained within dataset on nine participants and evaluated on the remaining one. In comparison, Zhang et al. reported an MCC of around 0.45 in stationary desktop settings [249]. In this evaluation, the manually annotated labels were only used for testing. For training, the eye contact detector automatically labels the image samples through unsupervised clustering. In an ablation study, we further evaluated the performance of the method by replacing the automatically labelled training samples with the manually annotated ones. We directly used these images to train the SVM eye contact detector and evaluated the resulting model.
in a leave-one-person-out cross validation. This is the Human baseline and, in this case, the method’s MCC score increases to 0.499 (SD=0.17).

Within-dataset evaluations only highlight one aspect of performance. With machine learning systems, it is also interesting to assess them across datasets, which is a good indicator of real-world performance. In this experiment, we trained the eye contact detector on one dataset and evaluated its performance on the other. Training on MFV and evaluating on UFEV, the MCC score is 0.124. Using the manually annotated labels, the MCC score increases to 0.403. Training on UFEV and testing on MFV, the MCC score is 0.484. With ground truth labels, the MCC score is 0.431.

To better understand the failure cases, we then identified and studied three core challenges: Partially visible faces, the impact of different head pose angles, and gaze estimation performance as a basis for eye contact detection.

### 4.3.3 Challenge 1: Face and Eye (In)visibility

One highly relevant challenge for studies conducted using the front-facing camera of mobile devices is the face and eye visibility of the participants [110]. Nowadays, most face detection, landmark detection, and even many gaze estimation approaches require the full face to be visible. However, according to Khamis et al. [110], the full face is only visible around 30% of the time. Zhang et al’s [249] method also requires the full face.
as input given that one of the steps in their pipeline is a full-face appearance-based gaze estimator. In this section, we evaluate the impact of partially visible faces on the method’s performance.

Our evaluation is conducted on the UFEV dataset which provides annotations for several different visibility categories depending on whether the entire face or only parts of the face are visible. The categories, the number of images in which a face can be detected, and the total number of images are:

- Whole face all landmarks, 2020/2292
- Whole face some landmarks, 329/442
- Partial face 2 eyes 1 mouth, 866/1203
- Partial face 2 eyes no mouth, 534/790
- Partial face 1 eye 1 mouth, 129/373
- Partial face 1 eye no mouth, 63/659
- Partial face no eyes 1 mouth, 1/8
- No face, 5/24

On average, 45.25% (SD=29.12%) of the images are skipped and hence could not be used in the evaluation because no face has been detected.

Figure 4.3 shows the result of a within dataset leave-one-person-out per category cross validation. For each person, we trained an eye contact detector (unsupervised, no labels required) on the data from the remaining nine people and evaluated the performance per visibility category. The rightmost two categories, Partial face no eyes 1 mouth and No face, have an MCC score of 0 simply because no images could be used in the evaluation, either because no faces were detected or because all the images only belonged to a single class. Thus, it is not possible to train and evaluate a classifier. For the remaining categories, we compared the method proposed by Zhang et al. [249] to the same method when using the manually annotated labels, the Human baseline. The results, also from a leave-one-person-out cross validation, are as follows. When the full face is visible, the MCC is 0.457 (SD=0.22). In the Human baseline, the MCC is 0.613 (SD=0.24), which shows the potential for improving the unsupervised clustering approach for automatic labeling of the data. For the other categories, the MCC score degrades when fewer landmarks are visible. If two eyes are visible, the average MCC stays above 0.3, however, once only one eye or less is visible, the method simply becomes unusable.
Chapter 4 Quantification of Users’ Visual Attention in Mobile HCI

To understand real-world performance, we conducted a cross-dataset evaluation (see Figure 4.4 for a performance overview of the method). The eye contact detector was trained once on the MFV dataset and evaluated once on the entire UFEV dataset, per visibility category. In this case, it becomes even clearer that the method performs poorly and could be significantly improved when comparing its performance with the human baseline.

### 4.3.4 Challenge 2: Robust Head Pose Estimation

Head pose estimation is a computer vision task where the goal is to determine how the head is tilted relative to the camera. It is expressed in terms of six degrees of freedom, three for translation and three for rotation in 3D. For the appearance-based gaze estimation task, head pose estimation is often used as input to train a CNN or for data normalisation [250]. In mobile settings (Figure 4.5 - Head pose distribution), for both datasets, we have noticed a large variability in both the horizontal and the vertical pose angles. Because of this, we investigated the influence of such angles on the eye contact detection performance. In other words, is the performance of eye contact detection worse when the head is tilted and not frontal? Does this happen often in mobile scenarios?

Figure 4.5 shows the results of this experiment. The first column represents the distribution of the head pose angles in the normalised camera space [250] estimated from the two datasets. For the experiments, we divided the data in five horizontal and five vertical

![Figure 4.4. Performance of the two methods, the eye contact detector by Zhang et al. and the Human baseline which uses manually annotated class labels. The bars represent the MCC coefficient. The results are from a cross-dataset evaluation where the eye contact detector was trained on the entire MFV dataset and tested once on the UFEV dataset for all participants, per category.](image-url)
4.3 Key Challenges in Quantifying Mobile Visual Attention

Figure 4.5. Classification performance of the eye contact detector by head pose angles. The left most column shows the distribution of the pitch and yaw in the normalised camera space. The MCC values represent the performance of the two baselines, per bucket, from a leave-one-person-out cross validation. The Human baseline uses manual ground truth annotations rather than clustering to obtain the labels for the training samples.

buckets. A pitch and yaw value between -10° and 10° represent little rotation of the head. Between 10° and 20° is a mild turn of the head. We consider anything over 20° as a significant head rotation. As shown in the head pose distribution, in mobile settings, it is often the case that the head and face are not directly facing the camera.

The reported values represent the MCC coefficient from a within dataset leave-one-person-out per bucket cross validation. The first row highlights the result on the UFEV dataset, while the second one shows the results on the MFV dataset. On the UFEV dataset, for pitch and yaw values between -10° and 10°, the MCC score is 0.4 for Zhang et al. and 0.5 when using ground truth labels. Because of the distribution of the data, a similar MCC value is achieved when the pitch is between 10° and 20°. As the angles become more extreme, the methods become unusable. On the MFV dataset, the performance is even worse. For frontal faces, the MCC value for Zhang et al. is 0.2.
4.3.5 Challenge 3: Accurate Gaze Estimation

Recent advances in appearance-based gaze estimation bring us closer to the vision of systems that are able to accurately track human gaze from a single image [118, 249, 252, 254]. Despite these advancements, most gaze estimators are still far from practical use due to lower accuracies and the unsupervised eye contact detection method [249] builds on such an appearance-based gaze estimator trained on MPIIFaceGaze [253]. Consequently, improvements to the gaze estimation task will also benefit eye contact detection. Estimating the gaze direction in everyday settings has to cope with several challenges. Varying illumination conditions, variability across users, different screen and camera geometries, face and facial landmarks occlusions are only a few of the challenges which have to be addressed for accurate and robust gaze estimation. Figure 4.6 shows a few sample images from the UFEV dataset together with the gaze estimates and the predicted eye contact label. For some images, Figure 4.6 columns 1-4, if the gaze estimates are reasonably accurate, the method is able to overcome small estimation errors and correctly predict (no) eye contact. However, gaze estimates can also be highly inaccurate if, for example, the face and facial landmarks have been incorrectly detected (column 8). Another possible source of error is due to the head pose angles (column 6). Most current gaze estimation datasets only contain limited variability in head pose angles, but as seen in Figure 4.5, mobile settings can exhibit a wide range of head orientations. Without

![Figure 4.6](image-url)

Figure 4.6. Sample images with the corresponding gaze estimates and the predicted eye contact label (green represents eye contact, red non eye contact). While being computationally simpler, the state-of-the-art method proposed by Zhang et al. for eye contact detection builds on an appearance-based gaze estimator. Thus, the performance of the method is dependent on the performance of the underlying gaze estimates. E.g. for certain head poses (column 6), if the gaze estimates are incorrect, the eye contact label will also be incorrect.
additional training data, the predicted gaze estimates in such cases will be inaccurate as well.

4.3.6 Discussion

In our evaluations, we identified three key challenges for sensing attention in highly dynamic, mobile interactive settings.

Our first experiment quantified the impact of face and eye visibility on the eye contact classification performance and showed that current methods performed best when the full face or all the facial landmarks were visible. As soon as the eyes or parts of them, which convey most of the relevant information for attention, were not visible, the performance of the method decreased significantly. Such analyses were possible due to recent datasets such as UFEV [110], however, a limitation of this dataset is the relatively few number of images available in some visibility categories. As such, large-scale datasets with fine-grained annotations will further help to better understand the failure cases. Another reason for the reported performance on partially visible faces are methods which require the users’ full face, including the one by Zhang et al. [249]. Moreover, just as the findings from Khamis et al. [110] highlight, in mobile settings the entire face is often not visible. Methods which only use an image of the eye already exist [254], however, they rely on face and landmark detectors which usually require the full face to be visible. Therefore, future work should investigate methods which can robustly find eyes in an image without having to detect the entire face.

Our second experiment on the error distribution of the eye contact detector relative to the distribution of the head pose angles yielded several interesting findings. For one, current methods perform best when the head is oriented towards the camera. As soon as the head is turned in any direction, the performance of the method becomes worse. However, we can observe that if there is sufficient training data available for such cases, e.g. Figure 4.5 - on the UFEV dataset when the pitch is larger than 10°, the method can still perform well. Based on this, as future research directions, we believe that at least two things are important. First, the head pose angles we used are estimates (there is no ground truth available), so it is possible that some of these are incorrect or inaccurate. Future research could investigate head pose estimation in mobile settings and assess accuracy and robustness specifically. Second, there is a need for new datasets that cover a variety of not only head pose angles but gaze angles as well.

Our last experiment qualitatively addressed the need for accurate gaze estimation. As previously mentioned, eye contact detection methods, while computationally simpler, still require reasonable gaze estimates to produce usable results. As such, any improvement
in current gaze estimation methods will also benefit attention sensing on mobile devices. More concretely, we encourage future work to investigate gaze estimation methods and datasets which have been collected specifically in such mobile interactive scenarios (e.g. the large-scale GazeCapture dataset [118]).

### 4.4 Accurate and Robust Detection of Eye Contact with Mobile Devices

To detect eye contact with mobile devices, we extend and build on the method proposed by Zhang et al. [249] and address challenges specific to mobile interactive scenarios: Face and facial landmark detection including partial occlusions, extreme poses, and variability across users, devices, and illumination conditions. The key novelty of this method was unsupervised clustering, an approach to automatically label the training data without manual and tedious annotations. The only assumption of this approach is that the camera needs to be mounted next to the target object. This assumption is valid for most mobile and tablet devices since the camera is usually placed just above the device’s display.

#### 4.4.1 Method Overview

We describe the main components of our mobile eye contact detection method in the context of the identified challenges (Figure 4.7).

**Face Detection and Facial Landmarks Localisation.** During everyday mobile phone interactions, the way users hold their devices influences how much of the users’ face can be captured by the front-facing camera’s limited field of view. Previous work has shown that parts of the face are often occluded and the entire face was visible only around 30% of the time [84, 110, 225]. To address this challenge specific to mobile scenarios, we use a more robust face detection and alignment approach [43] which builds on a multi-task CNN [247]. If multiple faces are detected, we select the face with the largest bounding box and discard the rest (e.g. smaller faces of bystanders). After detecting the face bounding box, it is particularly important to accurately locate the facial landmarks since these are necessary for head pose estimation and image normalisation. We use a state-of-the-art hourglass model [43] which estimates the 2D position of 68 different facial landmarks.

**Head Pose Estimation.** To estimate the users’ 3D head position, we use the landmarks detected previously. Head pose estimation is the task of finding the rotation and translation of the head relative to the camera. In computer vision, the pose estimation problem is
4.4 Accurate and Robust Detection of Eye Contact with Mobile Devices

C3: Variability Across Users, Devices, and Environmental Conditions
C2: Head Pose Estimation
C1: Face Detection and Facial Landmarks Localisation

(a) Multi-task CNN for face detection
(b) Hourglass NN for landmark detection
(c) Head pose estimation & pose stabilization
(d) Image normalization
(e) Gaze estimation
(f) Gaze locations clustering
(g) SVM

Eye Contact Detector

Figure 4.7. Method overview. Taking images from the front-facing camera of a mobile device, our method first uses a multi-task CNN to detect an eye contact (a) and a state-of-the-art hourglass NN to detect 68 facial landmarks (b). Then, we estimate the head pose using a 68 3D-point facial landmark model and stabilise it with a Kalman filter (c). Next, we normalise and crop the image (d) and feed it into an appearance-based gaze estimator to infer the gaze direction (e). If the estimated head pose exceeds a certain threshold, we use the head pose as a proxy for the gaze direction (f). If the gaze direction is determined, we can use the head pose to estimate the gaze direction (g). The weighted SVM eye contact detector is trained with features extracted from the gaze estimation CNN.

Face feature vector
Train
Test

SVM
Eye Contact
Detector

Training labels

Gaze estimation CNN
commonly known as the Perspective-n-Point (PnP) problem. In contrast to Zhang et al. who used six 3D points (four corners of the eyes and two from the mouth), we instead used a model with all the 68 3D facial landmarks [21], which is more robust for extreme head poses, often the case in mobile settings [17]. We first estimate an initial solution by fitting the model using the EPnP algorithm [129], which we then optimise and finally stabilise with a Kalman filter (as proposed by Yin Guobing [263]). Such an approach requires a calibrated camera, however, in our implementation we approximate the focal length with the width of the image in pixels, we assume the centre of the camera is the centre of the image, and we do not consider any distortions.

**Face Image Normalisation.** In this step, we use the previously estimated head pose to normalise the input image using the method proposed by Zhang et al. [250]. In a nutshell, the method cancels out the roll component of the head rotation and scales the face image. As a result, all face images look like they were captured with the same (normalised) camera. The significant benefit of this step is not only the ability to handle variations due to different shapes and appearances of the face, but also variations in hardware setups (e.g. cameras with different properties). In our implementation of their method, we set the focal length of the normalised camera to 960 px, the distance from the centre of the face to the normalised camera was 300 mm, and the resulting face image was 448x448 px.

**Gaze Estimation.** The basis for our eye contact detection method, just as in the one proposed by Zhang et al. [249], is an appearance-based gaze estimator. We used a state-of-the-art convolutional neural network (CNN) [253] that requires the users’ full face as input to predict the two-dimensional 3D gaze direction. While the outputs of the CNN are the pitch and yaw of the gaze direction in the normalised space, the last fully-connected layers of the CNN can be used to extract meaningful representations of the image. Such representations, i.e. feature vectors with 4096 dimensions, can directly be used as input to train a binary eye contact classifier. Given that our method was designed for robustness on images captured with mobile devices, we trained our model on the large-scale GazeCapture dataset [118]. This dataset consists of 1,474 different users and around 2.5 million images captured using the front-facing camera of mobile and tablet devices in the wild. By using normalised face images to train a gaze estimation CNN on this dataset, our method can better handle variability across users, devices, or environmental conditions. Our model achieves a within-dataset angular error of 4.3° and a cross-dataset angular error of 5.3° on the MPIIFaceGaze dataset [253] (which is comparable to current gaze estimation approaches).

**Head Pose Thresholding.** To overcome inaccurate or incorrect gaze estimates caused by extreme head poses, we propose the following thresholding mechanism: Whenever the pitch of the estimated head pose is outside the range $[-\theta, \theta]$, or the yaw outside $[-\phi, \phi]$, we use the head pose instead of the estimated gaze vector as a proxy for gaze
direction. More specifically, we assume that the gaze direction is the $z$-axis of the head pose. Because this thresholding happens in the normalised camera space [250], only two threshold values are necessary, independent of users or devices: One vertical and one horizontal. In practice, we set a value of $40^\circ$ for both $\theta$ and $\phi$.

**Clustering and Eye Contact Detection.** The key novelty proposed by Zhang et al. [249] was the unsupervised clustering approach to label images automatically. In our work, we also used such an approach as follows. The appearance-based gaze estimator produces 3D gaze directions represented as pitch-yaw pairs. Using the 3D gaze vector, we calculated the 2D point of regard by intersecting the gaze direction vector with the image plane in the camera coordinate system ($z = 0$). Taking all the training images, we obtained a distribution of the 2D gaze locations in the image plane, which we then clustered using the OPTICS algorithm [6]. To improve the robustness of our approach, we filtered out images in which the confidence value reported by the face detector was below 0.9. The target cluster, i.e. the one that contains positive eye contact samples, is the one that is closest to the centre of the coordinate system. The remaining images were labelled as having no eye contact.

Once all the labels for the training images were automatically generated using the unsupervised clustering approach, we trained a weighted binary SVM classifier. Methods like SVMs usually require feature engineering, i.e. domain knowledge to extract meaningful features from the data. The method proposed by Zhang et al. [249] does not need manual features since the appearance-based gaze estimation CNN can be used as a feature detector. The last fully-connected layer of our CNN also produces a 4096-dimensional feature vector. To remove noise and reduce the number of dimensions, we projected these vectors from a higher dimensional space to a lower dimensional space using principal component analysis (PCA). Through PCA, we reduced the number of features while keeping 95% of the variance in the data. After transforming the input features using PCA, we trained a weighted SVM using the positive and negative labels obtained previously during clustering.

### 4.4.2 Evaluation

For the evaluation of our eye contact detection approach, we use the same datasets as in Subsection 4.3.1. For more details, please refer to that section.

There are different ways to detect eye contact, such as GazeLocking [201] which is fully supervised (i.e. requires tedious manual data annotation), or methods that infer the coarse gaze direction [188] or leverage head orientation for visual attention estimation [222]. However, all of these methods are inferior to the state-of-the-art eye contact detector.
Chapter 4 Quantification of Users' Visual Attention in Mobile HCI

Figure 4.8. Sample results for eye contact detection on images from the two datasets, MFV and UFEV. The first row shows the input image; the second row the detected face (in yellow) and facial landmarks (in red); the third row shows the estimated head pose and gaze direction (purple); the fourth row shows the eye contact detection result, green for eye contact, red for non eye contact. Columns (1-9) illustrate how our method works across different users, head pose angles, illumination conditions, or when the face is partially visible. Columns (10-12) illustrate how our method fails if the gaze estimates are inaccurate (10), the eyes are closed (11), or the face detector fails (12).
4.4 Accurate and Robust Detection of Eye Contact with Mobile Devices

proposed by Zhang et al. [249]. Therefore, two of our baselines were based on variants of this method. Additionally, like Zhang et al. [249], we compared performance to a state-of-the-art but generic appearance-based gaze estimator, i.e. an approach that is not geared to the eye contact detection task but directly uses the raw gaze estimates.

**Evaluation baselines:**

1. **Zhang et al. [249].** We replicated the original method proposed by the authors by using the dlib [257] CNN face detector, the dlib 68 landmark detector, and by training the gaze estimator on the MPIIFaceGaze dataset [253].

2. **Zhang et al. + FA.** We replace the dlib face and landmark detector. For face detection, we used the more robust face alignment (FA) [43] approach that leverages three multi-task CNNs [247] to also detect partially visible faces, a challenge and key requirement in mobile gaze estimation. Similarly, we replaced the landmark detector with a newer approach which uses a state-of-the-art hourglass model [43] to estimate the 2D location of the facial landmarks. The CNN architecture and trained model were the same as in the first baseline.

3. **Gaze projection.** In this baseline, we measured a current generation smartphone (the Samsung Galaxy S9) and, with some tolerance, defined a bounding box of 10 by 12 cm. We then projected this bounding box on the camera image plane (by taking the position of the camera into account). Only if the raw gaze direction intersected with the camera plane inside this bounding box, the image was classified as eye contact. We used dlib for face and landmark detection and the raw gaze directions were calculated by a full-face appearance-based gaze estimator [253].

**Eye Contact Detection Performance**

Figure 4.9 shows the performance comparison of the four methods on the two datasets. For the three baselines *Ours, Zhang et al.*, and *Zhang et al. + FA*, the MCC scores were calculated using a leave-one-person-out cross-validation in which data from $N-1$ participants was used for training and the method evaluated on the data of the remaining $N$-th participant. This was repeated $N$ times and the scores averaged at the end. For the gaze projection baseline, the MCC score was calculated across participants because this method does not involve any training on the MFV or UFEV dataset. The error bars show the standard deviation across the different cross-validation repetitions.

As can be seen from the figure, our method (MCC 0.84) significantly outperforms all the baselines on the MFV dataset (MCC 0.52, 0.43, and 0.39). The same holds for the UFEV dataset where *Ours* (MCC 0.67) shows significantly increased robustness in
Figure 4.9. Performance of the different methods on the two datasets. The bars are the MCC value and the error bars represent the standard deviation across participants. The transparent bars illustrate the potential performance improvements when assuming perfect clustering for the three baselines: Ours, Zhang et al., and Zhang et al. + FA. For the gaze projection baseline, there is no clustering involved.

comparison to Zhang et al. + FA (0.17), Zhang et al. (0.35), and Gaze projection (0.37). A one-way ANOVA shows that the difference between the four conditions is significant at the $p < 0.05$ level on both datasets ($F(3, 196) = 62.02, p < 0.01$ on MFV and $F(3, 36) = 17.66, p < 0.01$ on UFEV). A post-hoc Tukey HSD test further shows that the difference between our method and each of the other baselines is also statistically significant on both datasets ($p < 0.01$).

To better understand the limitations of the clustering and the potential for further improvements, we also analysed the impact of the unsupervised clustering approach on the eye contact classification performance. To eliminate the influence of wrong labels resulting from incorrect clustering, we replaced the estimated labels with the manual ground truth annotations. As such, this defines an upper bound on the classification accuracy given perfect labels.

The transparent bars in Figure 4.9 show the result of this analysis, i.e. the potential performance increase when using ground truth labels. These improvements are only possible on the three baselines (Ours, Zhang et al., and Zhang et al. + FA) which rely on clustering to automatically select the positive and negative eye contact labels. For the gaze
projection baselines, clustering is not necessary because of the manually defined bounding box which serves a similar purpose. This bounding box could be further optimised but this requires precise measurements of all different devices, their screens, and camera locations—rather impractical for real-world systems given the large number and releases in mobile devices. Despite the two improved Zhang et al. baselines, our proposed method is still able to outperform them (an MCC score of 0.88 in comparison to 0.73 and 0.72 on the MFV dataset and 0.69 in comparison to 0.45 and 0.50 on the UFEV dataset).

A one-way ANOVA shows that the difference between the three learning-based conditions is significant at the $p < 0.05$ level on both datasets ($F(2, 147) = 12.86, p < 0.01$ on MFV and $F(2, 27) = 5.68, p < 0.01$ on UFEV). The difference between Ours and each of the other two variations of the Zhang et al. method is also statistically significant on both datasets ($p < 0.01$ on MFV and $p < 0.05$ on UFEV). Furthermore, our proposed method is close to the upper bound performance when using ground truth information.

These performance improvements are due to our new gaze estimation pipeline. Due to the improved training steps of the gaze estimator (face and landmark detection, head pose estimation, and data normalisation) combined with the GazeCapture [118] dataset, our model can extract more meaningful features from the last fully connected layer of the CNN which, in turn, improves the weighted SVM binary classifier.

**Performance of Detecting Non-Eye Contact**

The complementary problem to eye contact detection is to identify when users look away from the device (non eye contact). In some datasets, there are only a few non eye contact samples (e.g. only 17% in the UFEV dataset). Accurately detecting non eye contact is equally, if not even more important and at the same time significantly more challenging due to the sparsity of non eye contact events. Incorrectly detecting such events or missing them completely has a significant negative impact on the quality of the attention metrics derived from eye contact information. For example, too many incorrect non eye contact events could wrongly indicate a distracted user. For this analysis, the most important performance indicators are the true negative rate (TNR) and the false negative rate (FNR). The TNR measures the proportion of non eye contact (negative) samples correctly identified, while the FNR measures the proportion of positive samples which were incorrectly identified as negative.

Table 4.1 summarises the results of comparing the TNR and the FNR for non eye contact detection of the four methods. On the MFV dataset, our method outperforms all baselines and correctly identifies around 88% of the non eye contact samples in comparison to only around 51%, 39%, or 53% for the other methods. Furthermore, our method has the lowest
Table 4.1. Classification performance as true negative rate (TNR) and false negative rate (FNR) for non eye contact detection. The number of images used in the evaluation is dependent on the performance of the face detector. On both datasets, our method is able to outperform all the baselines and can correctly detect more non eye contact events and, at the same time, make fewer errors.

FNR of only around 3% and on the UFEV dataset is able to achieve the highest TNR of around 73%. The Gaze projection baseline is the second best performing method with a TNR of around 70% but there is a significant gap in terms of FNR. While our method achieves the lowest FNR or around 5%, the Gaze projection baseline is among the worst with around 24%.

Cross-Dataset Performance

To realistically assess performance of eye contact detection with a view to practical applications and actual deployments, it is particularly interesting to evaluate the cross-dataset performance. Cross-dataset performance evaluations have only recently started to being investigated in gaze estimation research [254] and, to the best of our knowledge, never before for the eye contact detection task. To this end, we first trained on one dataset, either UFEV or MFV, and then evaluated on the other one across participants. Our experiments only considered the three baselines in which an eye contact detector has to be trained: Ours, Zhang et al., and Zhang et al. + FA. A cross-dataset evaluation on the Gaze

<table>
<thead>
<tr>
<th></th>
<th>No. of images</th>
<th>True Negative Rate (TNR)</th>
<th>False Negative Rate (FNR)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MFV dataset</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gaze projection</td>
<td>3,663</td>
<td>50.6%</td>
<td>15.4%</td>
</tr>
<tr>
<td>Zhang et al.</td>
<td>3,663</td>
<td>38.9%</td>
<td>5.7%</td>
</tr>
<tr>
<td>Zhang et al. + FA</td>
<td>3,960</td>
<td>52.8%</td>
<td>4.3%</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td>3,960</td>
<td><strong>88.3%</strong></td>
<td><strong>3.4%</strong></td>
</tr>
<tr>
<td><strong>UFEV dataset</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gaze projection</td>
<td>4,217</td>
<td>70.2%</td>
<td>23.5%</td>
</tr>
<tr>
<td>Zhang et al.</td>
<td>4,217</td>
<td>55.4%</td>
<td>15.8%</td>
</tr>
<tr>
<td>Zhang et al. + FA</td>
<td>5,622</td>
<td>44.2%</td>
<td>25.0%</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td>5,622</td>
<td><strong>72.5%</strong></td>
<td><strong>5.1%</strong></td>
</tr>
</tbody>
</table>
Figure 4.10 shows that our method outperforms the other baselines by a significant margin in this setting, both when training on UFEV and testing on MFV, and vice versa. When training on the UFEV dataset, our method (MCC 0.80) performs better than the two baselines (MCC 0.45 and 0.47).

A one-way ANOVA shows that the difference between the three conditions is significant at the \( p < 0.05 \) level \( (F(2,147) = 56.16, p < 0.01) \). A post-hoc Tukey HSD test further shows that the difference between our method and the other two is statistically significant \( (p < 0.01) \). When trained on MFV and tested on UFEV, Ours (0.45) outperforms Zhang et al. + FA (0.09) and Zhang et al. (0.12). Also in this case, a one-way ANOVA shows that the difference between them is significant at the \( p < 0.05 \) level \( (F(2,27) = 16.56, p < 0.01) \). Ours is significantly better than either of the two baselines (post-hoc Tukey HSD, \( p < 0.01 \)). Taken together, these results demonstrate that our method that is specifically geared to mobile interaction scenarios, is able to abstract away dataset specific-biases and to generalise well across datasets. As such, this result is particularly important for HCI practitioners who want to use such a method for real-world experiments on previously unseen data, e.g. obtained from new mobile devices with other technical properties (camera, display) or in new interactive settings with changed usage behaviour (e.g. where phones are held at largely varying distances away from the face).

**Robustness to Variability in Ambient Illumination**

Mobile interactive scenarios are significantly more challenging for eye contact detection than previously investigated, stationary scenarios given that they are characterised by a large variability in usage conditions and environments. One core challenge is variability in ambient illumination, for example caused by a user walking out of a dim office building into the bright sunlight. To study this challenge in more detail, we analysed how varying ambient illumination affects our method’s performance in comparison to the other baselines. The MFV dataset provides images collected in three different illumination conditions: a well-lit room (2157 images), the same room with dim light (986 images), and another room with daylight illumination (1221 images). For evaluation we manually annotated all of these images with eye contact labels. Since the Gaze projection baseline does not require any training, we directly tested it on this data. For the other three methods, including Ours, we conducted a cross-dataset evaluation by training all the methods on the UFEV dataset and testing on MFV.

The results of this analysis are summarised in Figure 4.11. Our method outperforms the
Figure 4.10. Cross-dataset classification performance of the different methods on the two datasets. The bars are the average MCC value and the error bars represent the standard deviation across participants. The transparent bars illustrate the potential performance improvements with ground truth labels. Our method is able to better abstract away data-specific biases which is important for in-the-wild studies.

baselines in all the three illumination conditions by a large margin (0.61 vs. 0.40, 0.41, or 0.32 for dim light, 0.84 vs. 0.43, 0.42, or 0.40 for well-lit, and 0.80 vs. 0.51, 0.49, or 0.34 for daylight).

A one way ANOVA for each of these three conditions shows that there are significant differences ($p < 0.01$) in all three settings: $F(3, 196) = 7.51$ for dim light, $F(3, 196) = 28.82$ for well lit, and $F(3, 196) = 25.34$ for daylight. For daylight and well lit, the pairwise differences between Ours and all the other three are significant at the $p < 0.01$ level (post-hoc Tukey HSD). For dim light, the pairwise differences between Ours vs. Zhang et al. + FA and Gaze projection are significant at the $p < 0.01$ level (post-hoc Tukey HSD). For Ours vs. Zhang et al. the difference is significant at the $p < 0.05$ level. Overall, the baselines Zhang et al. + FA and Zhang et al., perform similarly, which highlights that only improving face and landmark detection is not sufficient to yield notable performance improvements.
Figure 4.11. Robustness evaluation in three different illumination conditions: dim light, well-lit, and daylight. The three methods, Ours, Zhang et al. + FA, and Zhang et al., were evaluated across datasets showing the expected real-world performance (trained on UFEV, tested on MFV). The bars represent the average MCC value and the error bars represent the standard deviation across participants. While there is a certain performance drop in dim lighting conditions, our method is consistently more robust and outperforms the other baselines.

**Influence of Head Pose Thresholding**

To reduce the impact of incorrect or inaccurate gaze estimates on eye contact detection performance, in our method we introduced a thresholding step based on the head pose angle. Current datasets [118, 253] have improved the state of the art in appearance-based gaze estimation significantly, however, they offer limited head pose variability when compared to data collected in the wild. Like in many other areas in computer vision, this fundamentally limits the performance of learning-based methods. In our eye contact detection approach, the underlying appearance-based gaze estimator is trained on the GazeCapture dataset [118] which, currently, is the largest publicly available dataset for mobile gaze estimation. The head pose distribution in the normalised camera space on GazeCapture is roughly between $-50^\circ$ and $50^\circ$ for both the horizontal and vertical angles. Nevertheless, both the MFV and UFEV dataset show larger head pose variability, between $-70^\circ$ and $70^\circ$ for the head pose angles.

In this experiment, we quantified the impact of the head pose thresholding step on our
Table 4.2. Performance (Matthews Correlation Coefficient) of the three different head pose thresholding techniques on both datasets. Gaze only uses no thresholding, Head pose only replaces all the gaze estimates by head pose estimates, and Ours replaces the gaze estimates by head pose estimates whenever the pitch or the yaw is below or above a threshold.

<table>
<thead>
<tr>
<th>Method</th>
<th>MFV (MCC)</th>
<th>UFEV (MCC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaze only</td>
<td>0.36</td>
<td>0.65</td>
</tr>
<tr>
<td>Head pose only</td>
<td>0.76</td>
<td>0.24</td>
</tr>
<tr>
<td><strong>Ours</strong> (Pitch = Yaw = 40°)</td>
<td><strong>0.82</strong></td>
<td><strong>0.69</strong></td>
</tr>
</tbody>
</table>

method’s performance. Table 4.2 shows the results of an ablation study with two other versions of our pipeline: The Gaze only (MCC 0.36 on MFV and 0.65 on UFEV) baseline does not use any thresholding. The Head pose only (MCC 0.76 on MFV and 0.24 on UFEV) baseline replaces all the gaze estimates with head pose estimates. The results show that Gaze only or Head pose only can yield reasonable performance for individual datasets. However, only our method (MCC 0.82 on MFV and 0.69 on UFEV) is able to perform well on both datasets, outperforming both baselines. This result also shows that, since this value is set in the normalised camera space, the same threshold value is effective across datasets.

Impact of Head Pose Variability in Mobile Settings

With mobile devices being used pervasively including while on the go, the way in which devices are held during everyday interactions causes significant variability in the users’ head position relative to the camera. As such, mobile settings are fundamentally different to stationary desktop settings in which the head exhibits only a limited change in the horizontal or vertical angle. Therefore, in this evaluation, we investigated the influence of head pose on eye contact detection performance.

Figure 4.12 shows the result of this analysis. The first column shows the distribution of the head pose angles in the normalised camera space [250] on the UFEV and MFV datasets. For this evaluation, we split the data into five horizontal and five vertical buckets: (-70°, -20°), (-20°, -10°), (-10°, 10°), (10°, 20°), and (20°, 70°). If the pitch or yaw was between -10° and 10°, there was little change in the head orientation relative to the camera. Between 10° and 20° is a mild turn of the head. Anything over 20° becomes significant
4.4 Accurate and Robust Detection of Eye Contact with Mobile Devices

Figure 4.12. Classification performance of the four methods by head pose angles. The left most column shows the distribution of the horizontal and vertical head pose angles relative to the camera which highlights the significant variability – a challenge in mobile settings. The remaining four columns show the performance of the four methods in terms of MCC, per bucket, in a leave-one-person-out cross validation. Our method is able to outperform the other three baselines and shows increased robustness to extreme head angles, even on the challenging MFV dataset.

Impact of Partially Visible Faces

Current appearance-based gaze estimators and eye contact detectors typically require that the users’ face is fully visible. While this assumption might be true in desktop settings, recent work from Khamis et al. [110] showed that in mobile settings the full face as
captured by the front-facing camera of a smartphone is only visible around 30% of the time.

We therefore studied the impact of partially visible faces on eye contact detection performance. UFEV is currently the first and only dataset to provide labels for multiple visibility categories for 10 participants: Whole face all landmarks, Whole face some landmarks, Partial face 2 eyes 1 mouth, Partial face 2 eyes no mouth, Partial face 1 eye 1 mouth, Partial face 1 eye no mouth, Partial face no eyes 1 mouth, and No face. Since the number of images per person and per visibility category is rather low (Subsection 4.3.3 for the numbers per visibility category), in this evaluation, we conducted a leave-two-persons-out cross validation for more reliable results. Figure 4.13 shows the result of this analysis.

When the full face and all the landmarks are visible, our method outperforms the other baselines (0.76 vs 0.32, 0.51, and 0.53). A one-way ANOVA shows that there is a significant difference between the four conditions ($F(3, 16) = 10.64, p < 0.01$). A post-hoc Tukey HSD test found significant differences between Ours and all the other three baselines ($p < 0.01$ for Zhang et al. + FA and $p < 0.05$ for the other two). If the whole face is in the camera’s field of view, but some landmarks are covered for example by a hand or a scarf, our method is still the best performing one (0.77 vs 0.27, 0.38, and 0.18). There are significant differences between the four baselines (one-way ANOVA, $F(3, 16) = 6.89, p < 0.01$). Using a post-hoc Tukey HSD test, Ours vs. Zhang et al. + FA is significant at $p < 0.05$ and vs. Gaze Projection at $p < 0.01$. The comparison to Zhang et al. is not statistically significant ($p = 0.06$). A similar result can be observed when both eyes and the mouth are visible, but other parts of the face are out of the camera view (ANOVA, $F(3, 16) = 13.00, p < 0.01$). A pairwise comparison between them found statistical significance only between Ours and the two Zhang et al. baselines (vs. Gaze projection $p = 0.06$). If both eyes are visible but not the mouth, all methods perform poorly because all of them used, as a basis, a full-face appearance-based gaze estimator which, for best performance, requires visibility of the entire face. Our method appears to perform worse than the baselines (0.36 vs 0.28, 0.43, and 0.40) but a one-way ANOVA shows that these differences are not statistically significant at the $p < 0.05$ level ($F(3, 16) = 0.57, p = 0.65$). For partial faces where only one eye and the mouth is visible, our method significantly outperforms the other three baselines (0.60 vs. 0.02, 0.14, and 0.09). These differences are significant (ANOVA, $F(3, 16) = 19.18, p < 0.01$) and pairwise comparisons further confirm our findings (post-hoc Tukey HSD, $p < 0.01$). Lastly, if only one eye was visible, similarly to the case where only the eyes were visible, the difference between the methods is not statistically significant (ANOVA, $F(3, 16) = 2.42, p = 0.10$).

While some of the other baselines may achieve comparable results for certain visibility
4.4 Accurate and Robust Detection of Eye Contact with Mobile Devices

![Figure 4.13. Performance of the four methods on the UFEV dataset, per visibility category. The bars represent the MCC and the error bars represent the standard deviation from a leave-two-persons-out cross validation. While partially visible faces are still a challenge, particularly when only few landmarks are visible, our method is still able to outperform the other baselines in almost all the categories.](image)

Our method consistently achieves high MCC scores across most categories. These results provide interesting insights into the strengths and weaknesses of all the methods but further research and, even more importantly, new datasets are clearly necessary for this task.

**Runtime Analysis**

Our method enables a post-hoc analysis of visual attention from video recordings and is currently not suited for real-time operation on a mobile device. We evaluated its runtime on a desktop computer equipped with an Intel i7-4790K CPU and an Nvidia GeForce 1080...
ti graphics card. In our experiments, the image height was set to 600 pixels and the width was scaled according to the images’ aspect ratio. Face detection, the main bottleneck, takes around 64 ms per frame. Landmark localisation for a batch of 30 images takes around 485 ms (around 16 ms per frame). Gaze estimation takes around 1.5 ms per frame (45 ms per batch). Predicting eye contact using the SVM takes less than 1 ms per frame. Overall, our implementation achieves around 12 frames per second (FPS). Down sampling images to 640x480 pixels increases the runtime to around 17-18 FPS. The computational time can be further improved by replacing the face detection step with an object tracker and detecting the face only when the tracking confidence decreases below a threshold.

4.4.3 Discussion

Detecting if, when, and for how long users have eye contact with their mobile devices is currently the most practical and most important approach to measure overt visual attention with significant potential for a wide range of applications in mobile HCI. We proposed a method to accurately and robustly detect eye contact in images captured with the front-facing camera readily integrated into mobile devices. Our in-depth evaluations show that our method advances the state of the art in eye contact detection along the three most important challenges characteristic to mobile scenarios: Face and facial landmark detection including partial occlusions, extreme poses, and variability across users, devices, and environmental conditions.

Overall, our method significantly outperforms the state of the art in terms of eye contact detection performance both within and across datasets (Figure 4.9, Figure 4.10, and Table 4.1). This highlights the ability to cope with variability caused by users’ appearance, different devices, and shows increased robustness to changes in ambient illumination conditions (Figure 4.11). By improving head pose estimation and introducing head pose thresholding (Table 4.2) in the normalised camera space, our method is able to cope with significant changes in the users’ head position relative to the camera – a challenge highly relevant in mobile interactive settings due to the way users hold their devices (Figure 4.12). Existing methods suffer from a significant drop in performance when large parts of the face are occluded or missing due to the camera’s limited field of view. By introducing robust face detection and facial landmark localisation, our method is able to detect a larger number of images (Table 4.1), including images with partial occlusions (Figure 4.8). Our experiment on the impact of partially visible faces, further shows that our method is able to maintain high performance despite some occlusions (Figure 4.13). The UFEV dataset [110] used in this evaluation is currently the only to provide suitable annotations to study this challenge but suffers from a limited number of images for certain visibility categories. Our work therefore also stresses the need for new datasets with fine-grained
annotations of face and facial landmark visibility to enable further insights into common failure cases.

Taken together, our evaluations underline the significance of the improvements specifically over the previous state-of-the-art method for automatic eye contact detection [249], as well as common approaches that either require user augmentation [206] or manual and tedious video annotation [166]. Our evaluations also provide valuable insights into the strengths and weaknesses of this approach that serve as important guidelines for researchers and practitioners who want to use eye contact as a tool to study visual attention. Simpler approaches that directly use the raw gaze estimates or head pose estimates as a proxy for eye contact may produce reasonable results in constrained settings or on specific datasets. However, our method is the only to robustly work across datasets and improve over the state of the art with respect to all of the aforementioned challenges (Table 4.2).

**Importance of Eye Contact Detection**

One of the most important applications enabled by eye contact detection is visual attention quantification, i.e. the task of automatically analysing when, how often, or for how long users visually attend to ("look at") their devices. In contrast to previous works that relied on device interactions or other events as a proxy to user attention [48, 160, 176, 227], our method can be used to quantify attention allocation directly and unobtrusively, only requiring video recordings from the front-facing cameras readily integrated in an ever-increasing number of mobile devices. As such, our method could already be used to extend digital wellbeing or quantified self applications with visual attention metrics (Figure 4.1) and enable users to gain deeper insights into their own mobile phone usage. However, real-world systems also face challenges complementary to eye contact detection. For example, face recognition or person identification could help in situations where multiple faces are present or faces of non-owners.

Being able to accurately and robustly sense when users look at their device, or when they look away, is a key building block for several higher-level visual attention metrics [206] (Figure 4.14). One such metric is the number of glances that indicates how often a user has looked briefly at their mobile device. In contrast, the (visual) attention span measures the amount of time users have spent looking at their device. By calculating the total duration of sustained visual attention towards the device or the environment, we can estimate the users’ primary attentional focus in a certain time window. Finally, the number of attention shifts further characterise the highly dynamic nature of visual attention [166].

These metrics are fundamental in a number of applications such as assessing user interruptibility [36, 56, 175] or estimating the noticeability of user interface content [177].
Figure 4.14. Our eye contact detection method enables quantifying mobile visual attention. Knowing when users look at their device (black blocks) and when they look away (white blocks) is a key component in deriving several attention metrics such as the number of glances (in yellow), the number of attention shifts (in green from the environment to the device and in purple from the device towards the environment), the duration of attention span (total duration of attention towards the device or the environment in a time interval), or the primary attentional focus.

Moreover, visual attention metrics could further complement existing methods that currently rely on device usage logs to model user behaviour. For example, visual attention could be useful to assess user engagement [148], boredom [178], or fatigue [3].

Limitations and Future Work

Despite the significant improvements in terms of performance and robustness for mobile eye contact detection, our method also has limitations. One of the key components in our method is the appearance-based gaze estimator. Our work highlights a limitation of current gaze estimation datasets, namely the limited variability in head pose angles in comparison to data collected in the wild. As a result, despite significant advances of appearance-based gaze estimation methods in recent years, their gaze estimates are still frequently inaccurate, thereby also limiting eye contact detection performance. We partly addressed this limitation through head pose estimation and head pose thresholding which, for extreme head poses, uses the head orientation as a proxy to compensate for unreliable
gaze estimates. This approach has proven effective but there are still cases when the head is turned away but users still look at their device. To address this problem, future work has to collect new gaze estimation datasets with more realistic head pose distributions for improved model training.

Besides performance improvements, runtime improvements will broaden our method’s applicability and practical usefulness. In its current implementation, our approach is suited for offline analysis, i.e. for processing image or video data post-hoc. This limitation is due to the computational power of current-generation smartphones in comparison to desktop computers equipped with dedicated high-performance GPUs. However, other application domains also require increasingly powerful mobile GPUs (e.g. for Augmented Reality), including dedicated chips for machine learning. As such, we believe that our method could soon be used on mobile devices, enabling real-time eye contact detection and paving the way for a range of additional applications. For example, a real-time algorithm could provide moment-to-moment assessment and feedback on visual attention allocation and dynamically adapt user interface content based on the user’s attentive patterns. Finally, processing the recorded videos directly on the device would not require to store them externally, potentially even in the cloud, to address privacy concerns by users.

4.5 The Everyday Mobile Visual Attention Dataset

In order to collect a large-scale in-situ dataset that can be used to quantify attentive behaviour during everyday mobile device interactions, we developed an Android application (Figure 4.15) with three main components: (1) An Android data logging application to record video snippets using the front-facing camera together with metadata, sensor data, usage logs, and location data, (2) The video review component that allowed study participants to easily access their data and filter out private data that they did not want to share, and (3) the annotation game that enabled participants to annotate data collected by others. In the following, we describe each of these components in detail.

4.5.1 Data Logging Application

The Android application for data logging consists of two background services: (1) A data capture service which starts the video recorder and logs the associated metadata, sensor data, and usage logs and (2) a notification listener service which logged mobile notifications.
Figure 4.15. Our custom Android application recorded video snippets using the front-facing camera readily integrated into modern smartphones every time a user unlocked their device. After installing the app, participants were asked to complete a short questionnaire on demographics (a) and to define private locations in which their GPS location was not logged (b). From the main menu (c), participants could start/stop the data collection service, review existing videos, play the annotation game, view the scoreboard, or change app settings. The video review menu (d) allowed them to select which data they wanted to share. Videos were played back at twice the speed to make reviewing easier (e).
Videos were recorded every time users unlocked their device. To prevent extremely large video files which are then hard to upload, the data collection service automatically stopped and restarted video recording after 15 mins. No videos were recorded when the device was in standby or when users checked the time or their notifications. Once installed, the app asked users for the necessary permissions. Participants had the possibility to manually start or stop the data recording service from the application menu (see Figure 4.15c). This ensured privacy when they did not want to be recorded. The data collection application did not restrict users in any way except restrictions imposed by the Android operating system: If another foreground app (e.g. while taking photos) was using the camera, the background service cannot record videos at the same time.

The Android application logged the following data:

- **Video data.** Videos were recorded at 720x1280 px (portrait), 30 frames per second, and bitrate of 5 Mbit/s. If the device did not support this resolution, we selected the closest resolution with a width below 720 px.

- **Sensor data.** Depending on hardware capabilities, we also logged readings from the device-integrated accelerometer, gyroscope, magnetometer, proximity sensor, light sensor, ambient temperature, and step counter.

- **Location data.** Our application collected location data if users enabled location services on their devices. We stored the latitude, longitude, and the measurement accuracy as reported by the API. Since location data can contain private information, we transformed the raw GPS locations into place types. For this purpose, we relied on the Google Nearby Places Search Request API. A comprehensive list of these place is provided in the online documentation of the API [261].

- **Device usage logs.** Among the most important ones are the application running in the foreground, touch events, the charging state, screen orientation, ringer mode, display brightness, or connectivity state (Mobile Connectivity or Wi-Fi). For security, the Android OS only allows logging when touch events happen and not where on the screen.

- **Activity.** The current activity of the user as predicted by the activity recognition API from Google. Some of the possible classes were “STILL”, “IN VEHICLE”, “RUNNING”, or “ON FOOT” and include a confidence value.

- **Notifications.** The notification listener service keeps track of any notifications that appeared in the status bar. Our application logged the source of the notification, i.e. the application or package that triggered it, and additional metadata. We did not store any of the actual content.
Bluetooth data. To better understand the users’ surrounding, our application also collected Bluetooth data. The number of devices in the immediate vicinity may reveal whether users are in densely populated areas, which may impact their attention level. The data collection application scanned for nearby Bluetooth devices only once per session, i.e. when users unlocked their device. It logged the MAC address and the RSSI value. The MAC address (or hardware address) may uniquely identify a specific device. For additional privacy, we hashed these addresses using the SHA-256 algorithm.

In the analyses that follow, we investigated visual attention across users, applications, and different usage contexts. We leave further analyses of visual attentive behaviour for future work.

4.5.2 Video Review Component

None of the data was uploaded without the users’ explicit consent. Before uploading, users had to open the study application and go to the video review menu (Figure 4.15d). This menu allowed users to review all videos collected so far and to decide which ones to upload or delete permanently. For faster reviewing, videos were played back at 2x the normal speed. To further help participants with video reviewing, the study application also prompted users through a notification at 10 pm in the evening that new videos were available for review. For additional safety and privacy, our application also had a history menu which showed all the videos which had already been reviewed and uploaded. If participants considered they had made a mistake, they could retroactively request the deletion of files from the server. This measure was in accordance with the university’s ethics policy.

All the files, including the videos and associated metadata and logs, once reviewed, they were uploaded to a secured university server. Data was uploaded in the background, without any involvement from the users, and only over Wi-Fi to avoid consuming large amounts of the participants’ mobile data. In contrast to collecting a dataset consisting mainly of photos [110], our app also had to handle large video files (over 500 MB for 15 mins) and short-term internet connectivity. Before uploading a file, videos were therefore split in 1 MB chunks. Through a 128-bit MD5 hash-checksum appended to each chunk, the server validated them for correctness and, after receiving all the chunks, merged all of them back and reassembled the whole video file.
4.5 The Everyday Mobile Visual Attention Dataset

Figure 4.16. The study application had built-in crowd-sourced functionality that enabled participants to indicate the (approximate) location of the person’s face in the image (a) or to annotate whether that person was making eye contact with the device (b). This feature was implemented as a game and participants could see how many images they had already annotated and how they ranked in comparison to others (c).

4.5.3 Annotation Game

To annotate (parts of) the recorded data, in our application, we further implemented an annotation game that allowed participants to quickly and effortlessly annotate images with eye contact labels in a crowd-sourcing fashion (Figure 4.16). Each participant was assigned, for privacy, a random username. The images each participant had to annotate were randomly sampled from all the other participants from the dataset – participants did not annotate their own images. The annotation game had two steps: (1) Locate the face inside the image (Figure 4.16a) and (2) decide whether the person in the image was making eye contact with their smartphone or not (Figure 4.16b). To mark the location of a face, users first touched a point on the screen. An orange circle centred at that location started to grow. Once the circle covered the entire face of the person, app users touched the screen again to stop the circle from growing. In case of a mistake, users could repeat this process as many times they wanted. Afterwards, users had to indicate whether the person in the image was making eye contact with their device or not. We implemented a mechanism similar to a popular dating application: For eye contact swipe to the right, if no eye contact swipe to the left. Additionally, we implemented a third swipe towards the top.
of the screen in case users were unsure or if there was no person in the image. Participants earned one point for labelling one image and two points if they also indicated where the face was located. Based on the score, participants earned badges, which encouraged them to annotate more images and reach the next level. The annotation game included a scoreboard where participants could see how many images they annotated and how they ranked in comparison to others (Figure 4.16c).

4.5.4 Data Collection

We deployed the data collection application on the Google Play Store. Any user with a valid Google account could download, install the app, and participate in our data collection. This way, participants could use their own smartphones, therefore producing more ecologically valid behavioural data.

Participants

We first obtained ethics approval for both the application and the data collection as a whole from the ethics committee of ETH Zürich. We then advertised our data collection through university mailing lists, social networks, or advertising websites. In total, the application was downloaded and installed on 54 unique devices. Out of these 54, 32 participants (20 male, 12 female) went through the study set-up phase and agreed to participate. Based on the demographics survey, the ages ranged from 18 to 59 (M=26.78, SD=8.39). Three participants identified themselves as left-handed, while the rest were right-handed. Their professions included mostly bachelor, master, and PhD students, but we also had two accounting professionals, a service technician, musicians, a photographer, retirees, a scientist, and an entrepreneur. Self-reported ethnicity of the participants was 19 x White, 6 x Asian, 3 x Latino, 2 x Black or African American, 1 x Hispanic, 1 x Indian. 18 participants used the private location feature and set between one and five (M=1.55, SD=1.02) privacy-sensitive areas. They used a wide variety of Android devices from manufacturers such as Samsung, Xiaomi, Motorola, Huawei, HTC, Nokia, or LG, with different versions of the operating system (from Android version 6.0 to 9). 10 participants said they wore glasses and all of them stated using their own private device for the study.

Those who participated for at least two weeks in our study were compensated. The requirements for compensation were as follows: In 12 out of the 14 days, participants had to share at least 10 videos per day and a total duration of at least 15 mins for CHF 50 or 30 mins for CHF 100. Participants also had two days where they did not have to meet the minimum upload criteria. Out of 32, 25 participants were compensated and all of
them received the maximum amount. Moreover, 10 random participants were additionally compensated with CHF 30 if they participated in the annotation game and have annotated at least 300 images.

**Procedure**

After opening the application for the first time, a welcome screen explained participants the goals of the study. Afterwards, participants were asked to carefully read the information and give their informed consent by manually selecting a checkbox as approved by the Ethics Committee of the university. Then, users were asked to fill-in a short demographics questionnaire (Figure 4.15a). The questionnaire included questions on age, gender, profession, ethnicity, dominant hand, whether they wore eye glasses or contact lenses, and whether they considered themselves technologically adept. Given that the application also logged location information (latitude and longitude), the interface then prompted participants to set any number of private locations (e.g. home or work) (see Figure 4.15b). If users were within 100 m of each such location, no location data was logged by the application. In the final step, users were shown a video tutorial that explained most of the functionalities of the study app. Before data collection started, users had to grant the app all required permissions. This step was also shown and explained in the video tutorial, so that all participants could successfully start the study.

Figure 4.17. Key characteristics of the EMVA dataset. The number of videos and the total duration in hours per participant sorted by duration in decreasing order. The dashed lines represent mean values.
4.5.5 Dataset Characteristics

The resulting dataset contains video snippets from 32 participants collected over more than two weeks in-situ. Figure 4.19 shows a few sample images that highlight the variability not only in user appearance but also environmental conditions of the recorded data. As such, there are a total of 14,322 videos, with each participant contributing between 31 and 1535 videos (M=447.56, SD=370.21). The total duration of the videos is around 472 hours. The minimum per participant is 0.35 hours, while the maximum is 47.67 hours (M=14.76 hours, SD=12.63 hours). Figure 4.17 and Figure 4.18 show the number of videos and the corresponding duration per participant. 26 participants used the video review feature to delete 1,326 videos with a total duration of over 50 hours. On average, each participant deleted 51 video snippets (SD=81.83).

Through the annotation game, 15,740 images were annotated by at least two study participants with eye contact labels. Annotators agreed on 13,234 labels: 7,871 eye contact, 1,746 non eye contact, and 3,617 skipped/unsure. For the 2,506 frames with an annotation conflict, an experimental assistant assigned a third label, where possible (684 for eye contact, 458 for no eye contact). This resulted in 8,555 images (7,871 + 684) labelled as eye contact and 2,204 (1,746 + 458) labelled as no eye contact, a total of 10,759. On average, there are 501 annotated frames per participant (SD=442.94).

Figure 4.18. Key characteristics of the EMVA dataset. A histogram of the video (and, hence, interaction) duration. While the average video duration is 2 minutes, many videos (around 24%) are less than 10 s long. This shows the highly fragmented nature of mobile interactions and user attention.
4.5 The Everyday Mobile Visual Attention Dataset

Figure 4.19. Sample images from our EMVAD dataset showing the significant variability in terms of place and time of recording, face and eye appearance, as well as illumination conditions. The dataset contains 4,322 videos, totaling around 472 hours, of 32 users interacting with their mobile devices. It also contains associated meta-data, sensor data, and device usage logs as well as 10,759 eye contact labels, each manually annotated by at least two app users.
Besides eye contact labels, our EMVA dataset also provides face location annotations, i.e. the location and approximate size (bounding box) of the face in the image, for 11,442 images from the total 15,740. 9,368 images were annotated by two or more people and 2,074 by just one study participant.

The 15,740 images collected through the annotation game were also captured in different device orientations: 15,257 in portrait mode (camera at the top) and 349 in landscape mode, either with the camera to the left or to the right. Figure 4.20 shows a distribution of the face annotations for the whole dataset. It can be seen that the location and size of the face in the image depends on the device orientation. In landscape mode, faces tend to appear larger due to being closer to the device.

### 4.6 Quantification of Mobile Visual Attention

In this section, we use eye contact detection as a basis for calculating higher-level visual attention metrics. Zhang et al. were the first to propose an unsupervised method that
leverage inaccurate gaze estimates and still achieved state-of-the-art performance for eye contact detection [249]. In our work (Section 4.4), we significantly improved their approach by addressing challenges specific to mobile interaction scenarios. To understand whether any of these methods could be used to sense visual attention in-the-wild, we first evaluated both on our new dataset using the crowd-sourced eye contact annotations collected with the app.

**Eye Contact Detection Performance on EMVA**

To establish the state-of-the-art performance for eye contact detection on our new dataset, we compare our method to the one originally proposed by Zhang et al. [249]. For training we used three different datasets: EMVA, UFEV [110], and MFV [59]. For additional details on the MFV and UFEV dataset please refer to Section 3.4.3. For details on the EMVA dataset, please refer to Section 4.5. For training purposes, the images do not have to be labelled since both methods are fully unsupervised. Labels are only necessary for evaluating the accuracy of each approach.

We measured performance in terms of the Matthews Correlation Coefficient (MCC), which is a well-established metric to evaluate binary (two-class) classification tasks. We first conducted a cross-dataset evaluation: We trained both eye contact detectors on MFV, UEFV, and both (MFV+UFEV) and then evaluated on the 10,759 crowd-sourced annotations (8,555 eye contact and 2,240 no eye contact) from our dataset. Afterwards, we conducted a within-dataset evaluation on the annotations by doing a leave-one-person-out cross-validation, i.e. train on the data from 31 participants and evaluate on the remaining one. Finally, we also trained an eye contact detector on all three datasets.

Figure 4.21 shows the result of these evaluations. Our method [16] (Section 4.4) significantly outperforms the method by Zhang et al. [249] on our challenging dataset independent of the type and amount of training data. The highest MCC score was achieved, as expected, when training and testing on the same dataset (MCC=0.66). Nevertheless, we also noticed that when training on a combination of MFV, UFEV, and on the EMVA dataset, the MCC score is similar (0.64). A one-way ANOVA showed that the difference between the four conditions is significant at the $p < 0.05$ level ($F(4, 150) = 3.69, p < 0.01$). A post-hoc Tukey HSD test further showed that the difference is significant only between the two best performing ones and the worst performing condition, i.e. training within dataset or on all three datasets vs. training only on MFV. The difference between the first and second best performing method is not statistically significant (Tukey HSD $p = 0.4484$).

Given these findings, we opted to use only our method for all further analyses. Machine learning models typically benefit from large amounts of training data and can, as such,
better abstract away user and data-specific biases. Since the difference between training within dataset and training on all three datasets was not statistically significant, we decided to train the eye contact detector on all the data we had, i.e. MFV, UFEV, and EMVA. We ran the prediction on our entire dataset to label each frame with one of three possible labels: Eye contact, no eye contact, or undefined. The undefined class was used when no face was detected in the image, hence it was not possible to infer whether a user was present, or if the face detector had failed. Upon manual inspection of eye contact predictions, we found that predictions were often inaccurate for four of the 32 participants. To increase reliability of the attention analyses, we decided to exclude these four participants in the following evaluations. These participants are, nevertheless, included in the public release of the dataset.

4.6.1 Visual Attention Across Participants

On our evaluation dataset consisting of data from 28 participants, on average, 50.74% of the frames were predicted as eye contact, 10.73% as non eye contact, and 38.53% as undefined. Figure 4.22 shows the per-participant distribution of these labels. The average duration of sustained attention per video, i.e. continuous eye contact interval, was
4.6 Quantification of Mobile Visual Attention

7.23 s (SD=18.88 s). The average duration for non eye contact also per video was 1.87 s (SD=4.9 s) and the average duration of an undefined segment was 6.81 s (SD=36.13 s). Besides average values, we also analysed the duration of the first eye contact segment, i.e. when users unlocked their device and started to interact. In this case, the average duration of the first attention span was 11.28 s (SD=30.87 s). When looking at the longest visual attention span per participant and per video snippet, the average value was 22.56 s (SD=44.8 s).

Taking into account the distribution of the eye contact and no eye contact labels, we extracted the users’ primary attentional focus proposed by Steil et al. [206]. If the majority of labels in a single video were labelled as eye contact, excluding undefined sections, the primary attentional focus was defined to be on the device. If most of the labels were non eye contact, the focus was on the environment. Looking at all the videos and aggregating per participant, in 61.41% (SD=20.09%) of the cases, the primary attentional focus was on the mobile device. In 5.86% (SD=6.16%) of the videos, the users’ primary attention was towards the environment. For the remaining 32.73% (SD=18.6%), the main focus was undefined, i.e. in these videos, the face detector has failed to detect a face in more than 50% of the image frames.

Another key characteristic of attentive behaviour are attention shifts [206]. We analysed

![Figure 4.22](image_url)

Figure 4.22. The distribution of eye contact, no eye contact, and undefined – no face detected – labels sorted in decreasing order of eye contact percentage. The percentage of no eye contact varies significantly across participants with a minimum of 3.59%, a maximum of 38.99%, and an average of around 10%.
four types of attention shifts: From the device to the environment, from the environment to the device, and from the device/environment to an undefined section or the other way around. To avoid attention shifts caused by blinking, which the eye contact detector predicts as no eye contact, we empirically defined a threshold of 250 ms between attention shifts since the average duration of a human blink is between 0.1 and 0.4 s [194]. On average, per video snippet, there are 4.63 (SD=9.99) shifts of attention from the device to the environment and 4.4 (SD=9.81) from the environment to the device. When looking at attention shifts and undefined segments (e.g. eye contact followed by undefined), 7.9 (SD=15.91) were from or towards the environment, 6.45 (SD=14.2) from or towards the device.

We further analysed diurnal attentive patterns (Figure 4.23). The duration of sustained visual attention in the morning (between 7 and 12 am) varied between 8 and 12 s (average around 9 s). In the evening (after 6 pm), these durations tended to get shorter, varying between 6 and 7 s (average around 6.3 s). After midnight (from 0 am to 7 am), this duration decreased further to an average of 5.4 s. We also analysed visual attentive behaviour per day of the week, i.e. Monday through Sunday, averaged per participant. We did not find significant differences between days, with an average sustained attention duration per video snippet of 7.23 s (SD=0.44 s).

Figure 4.23. The average duration of sustained visual attention across participants per hour of the day. The bars represent the duration and the error bars represent the standard deviation. While similar throughout the day, the duration of sustained attention tend to be larger in the morning than in the evening and significantly decreases during the night.
4.6 Quantification of Mobile Visual Attention

4.6.2 Visual Attention Across Applications

During the data collection period, the study participants used a total of 420 different applications – identified based on the application package name. In this experiment, we clustered the individual applications by category (as listed in the Google Play Store) and analysed the average duration of sustained visual attention, i.e. the average attention span, and the number of attention shifts per application category and per video snippet (Figure 4.24).

Our results highlight that visual attentive behaviour strongly depends on the type of application used. For example, in the Medical category, the average duration of attention span was 13.21 s (SD=18.26). Similarly, in the Education category, the duration was 12.45 s (SD=30.34 s). In contrast, in an application showing the weather, the attention span was much shorter – 4.33 s (SD=7.45 s). Not only is the duration of attention spans different but also the number of attention shifts differed across categories. For instance, in the Beauty category, from our dataset, we extracted 19.5 (SD=24.74) shifts of attention towards the environment. Looking at the Education category where the attention span was higher than average, there were also fewer attention shifts (6.17, SD=7.99). While for certain categories it is difficult to draw generalisable conclusions due to the amount of data available (e.g. the Dating category), our results show that attentive behaviour can vary when different application types are used.

4.6.3 Visual Attention Across Usage Contexts

To gain further insights into attentive behaviour across different usage contexts, we first analysed visual attention relative to the activity (Figure 4.25). The data collection application also logged the current activity of the user as recognised by the activity recognition API from Google. The possible classes are Still, In vehicle, On bicycle, On foot, Running, Walking, or Tilting. On foot represents a user who was walking or running, In vehicle users who were in a car or on public transport, and Tilting was recognised when the angle relative to gravity had changed significantly. The recognition API also provides a confidence value that represents the probability of the most likely class. In our analysis, we only considered results where this value was larger than 50%. All results that follow are per video and per participant. As expected, the duration of sustained visual attention was the longest when users were still (M=7.71 s, SD=19.92 s). When users were walking or running, the duration of attention span was significantly shorter, between 2.4 and 3.1 s. The shortest attention span was when the recognised activity was Tilting (M=0.52 s, SD=1.76 s).
Chapter 4 Quantification of Users’ Visual Attention in Mobile HCI

Figure 4.24. Average duration of sustained visual attention (in purple) and the number of attention shifts towards the environment (in yellow) per video and application category. The error bars represent the standard deviation.

---

**No. of attention shifts towards the environment**

**Average duration of sustained attention**

Medical Education

News & Magazines

Social

Beauty

Sports

Games

Video Players & Editors

Food & Drink

Communication

Shopping

Photography

Study Application

Productivity

Travel & Local

Lifestyle

Business

Weather

Finance

Maps & Navigation

Music & Audio

Books & Reference

Tools

Health & Fitness

Entertainment

Personalization

System

Dating
4.7 Discussion

Understanding when, how often, or for how long people use their mobile devices is a fundamental problem in HCI with significant implications for tasks such as predicting interruptibility or estimating attentiveness to messages and notifications. Often, insights into users’ attention was only a by-product of these tasks, mainly because of a lack of methods to sense and quantify it in unconstrained in-the-wild settings, i.e. during natural everyday mobile device use. In this work, we used off-the-shelf smartphones to collect the EMVA dataset containing around 472 hours of video snippets as well as metadata, sensor data, and usage logs from 32 participants. Using our proposed method for automatic eye contact detection (Section 4.4), we provided, for the first time, quantifiable visual attention metrics extracted from this dataset. Our results, distilled into the following key insights, inform the design of future mobile attentive user interfaces in several ways.

Eye contact detection as a tool for analysing overt visual attention in situ. As demonstrated in our work (Figure 4.21), detecting when users have eye contact with their
device is feasible using latest methods even on challenging video data recorded in-situ, as available in our dataset. We also showed that eye contact detection provides rich insights into attentive behaviour and is the basis for key attention metrics, such as the duration of sustained visual attention on the device or the number of attention shifts to and from users’ environment. This method has two major advantages over previous approaches: 1) It does not require any special-purpose hardware, only an off-the-shelf smartphone with integrated front-facing camera, and 2) it does not constrain users in any way. Hence, eye contact detection enables, for the first time, the analysis of data collected in-situ and building of mobile user attention models. In the future, such models could, for example, be used to adapt interaction modalities based on users’ attentive state [163, 179].

**Visual attention in mobile HCI is highly fragmented.** Our analyses of attention showed that the average duration of sustained visual attention was only around 7 s. This finding supports previous works highlighting the highly fragmented nature of mobile interactions [166]. Moreover, our analysis also investigated attention shifts which are also a key characteristic of attentive behaviour. We found that, on average, users redirect their attention from the device to the environment around four times per interaction. When they do shift their overt attention, these diversions typically last for around 2 s. These findings underline the need to develop a new generation of attentive user interfaces that actively manage and protect such a valuable resource as human attention. This could be implemented, for example, by helping users through explicit feedback or stimuli, or by unconsciously increasing their level of engagement [117], to not direct their attention away from the device.

**Visual attention is user and context-specific.** While analysing the entire dataset in the form of aggregate statistics provides valuable insights into visual attention during mobile phone interactions, we observed that even more interesting insights can be gained when analysing the characteristic attentive behaviour patterns of individual users (Figure 4.22). We found that visual attention is not only highly user-specific but also highly dynamic over the course of a single day (Figure 4.23). Moreover, attentive patterns vary based on the mobile application used (Figure 4.24) as well as based on the users’ current activity and usage context (Figure 4.25). Taken together, these results provide strong evidence that proxy methods, such as Apple’s ScreenTime – which assumes attention when the screen is on, are inaccurate and indeed do not capture the fragmented and individual characteristics of attention. The same holds true for commonly used application usage logs as a proxy to user attention that, as our results show, only provide a very limited view on users’ attention.

**Face detection is an open challenge.** Current eye contact detection methods, including the one we used, requires the face to be detected initially. Face detection is an open research challenge in computer vision [244] and, sure enough, as can be seen from Figure 4.22, the
percentage of images in which the face cannot be detected – the *undefined* category – not only varies per participant but can be as high as 74.4%. This can happen either because the user is really not present nor visible or when only parts of the face are visible – often the case when using the front-facing camera of a mobile device [110]. The eye contact detection method we used incorporates a state-of-the-art face detector that managed to detect the face in 61.5% of images in our dataset (around 235 hours of video data) in comparison to only around 30% reported by Khamis et al. on their dataset [110]. The remaining 38.5% show that there is also an urgent need for better face detectors or for multimodal attention analysis systems that extend eye contact detection with additional information, e.g. obtained from user interactions [237]. Improvements in face and eye contact detection can be expected to also significantly increase robustness and accuracy of future analyses of mobile attention.

### 4.7.1 Limitations and Future Work

While our work is the first to present quantifiable visual attention metrics during mobile interactions in-situ, it also has several limitations that we plan to address in future work.

First, any analysis is only as good as the underlying eye contact detection method. Any improvements to this approach will therefore also increase the quality of the calculated statistics. In our evaluations we had to exclude four out of the 32 participants because the eye contact predictions were too inaccurate, too often. To better understand the failure cases, it will be crucial to extend the EMVA dataset with additional fine-grained eye contact annotations. Future work could then investigate how the method performs, for example, when blinking. Currently, images in which participants were blinking were predicted as no eye contact. To increase the reliability of the reported statistics and to avoid counting additional attention shifts caused by blinking, we currently used a buffer of 250 ms (an average duration of a blink [194]) between consecutive attention shifts. As such, we also hope that our dataset can serve as a challenging benchmark for future eye contact detection methods in mobile interactive scenarios.

Another limitation concerns the data collection process. As required by the university’s ethics committee, the application allowed participants to pause and resume data collection whenever they wanted. As a result, we observed two different behaviours: (1) Participants who kept data recording on most of the time and then chose which videos to delete post-hoc (up to 362 video snippets deleted per person) or (2) those who collected data only in situations which they explicitly wanted to share. This means that for the latter category, it is possible that participants were more aware of the fact that they were being recorded during their interactions and, hence, changed their behaviour.
4.8 Conclusion

In this chapter, we addressed the problem of quantifying users’ visual attention during everyday mobile device interactions. Understanding when, how often, or for how long users look at their mobile devices is a key challenge and is essential to several tasks in mobile HCI. Towards this goal, we first used a state-of-the-art method for automatic eye contact detection to identify three core challenges associated with sensing attention in the wild: Face and eye (in)visibility, robust head pose estimation, and the need for accurate gaze estimation. Based on these findings, we then proposed a novel method to detect eye contact in images captured with the front-facing camera of mobile devices. Through in-depth evaluations on two current mobile interaction datasets, we demonstrated significant performance improvements for mobile eye contact detection across devices, users, and environmental conditions. To be able to study visual attention in-situ, we proposed EMVA, a new 32-participant dataset of around 14,000 videos, totalling approximately 472 hours recorded over more than two weeks during mobile device interactions in-situ. Using our method for eye contact detection, for the first time, we extracted quantifiable visual attention metrics characterising the highly fragmented nature of mobile interactions. Taken together, these results are significant in that they provide the first-ever insights into (visual) attentive behaviour dynamics by directly looking at the user through the front-facing camera of mobile devices.
Chapter 5

Conclusions and Outlook

In this chapter, we review the contributions of this dissertation, we discuss remaining limitations, and propose future research directions.

5.1 Summary of Contributions

Nowadays, users are surrounded by an ever-increasing number of smart devices that create ubiquitous opportunities for interaction. Consequently, enabling natural and intuitive interaction with complex systems is one of the grand challenges in HCI. When people interact with one another, they use multiple modalities to convey information, both verbal and non-verbal communication through gestures or posture. However, current input modalities cover only a subset of these modalities: Touch interactions coupled with apps on our smartphones or tablets are today’s universal interaction devices and have recently been extended with voice control. In this dissertation, our overarching goal was to extend human interaction capabilities with natural gestures while in close physical proximity and eye-based human-computer interaction. The interaction techniques, methods, systems, and analyses we proposed are not only relevant for explicit interaction, i.e. enabling interactive technologies, but they allow to understand user behaviour better, which is an essential component of any HCI system. Towards this goal, we summarise our contributions as follows.

In Chapter 2, we presented a novel interaction technique that enables multiple users who are physically close to one another to interact using natural gestures. This supports the development of novel social interactive systems that bring people together. Three fundamental components enable our approach. Close physical proximity is detected
via two-way inaudible acoustic ranging and hand gestures are recognised from a single built-in accelerometer, commonly found on any mobile or wearable device. Information between devices is exchanged over Bluetooth, a communication protocol that is not only scalable but has become widely used for many applications. Our extensive evaluations on both mobile and wearable devices highlighted the feasibility of our approach using off-the-shelf commodity hardware. We then explored the design space enabled by the proposed interaction and demonstrated four real-world applications. Our first prototype was HandshakAR, a novel AR system that enabled two users who share the same greeting gesture to seamlessly and effortlessly exchange contact information. We then developed three additional applications. First, an interactive game for children: Opening a magical treasure chest in a theme park. For the chest to open, at least three children had to be within 1 m from the chest and synchronously perform the same hand gesture. The second and third demonstrators proposed collaborative fitness and a collaborative music band. Our results are significant in that they pave the way towards a new generation of multi-device (and implicitly multi-user) interactions for people who are physically collocated.

In Chapter 3, we explored different facets of eye-based human computer interaction. We first proposed ubiGaze, a novel AR system that enabled users to attach and retrieve digital information from any real-world object using gaze gestures. Our system extends the concept of AR annotations or tags to a messaging service, where messages are embedded into a set of distinctive features of real-world objects. This idea was enabled by two wearable devices: a head-mounted eye tracker and a smartwatch. Our work on ubiGaze allowed us to gain additional insights into the limitations of current regression-based wearable eye trackers. Among those, one of the most relevant to users is the tedious and cumbersome initial calibration phase. To address this limitation and simplify this step, we proposed fingertip calibration. To calibrate an eye tracker, users only have to point at locations in the scene and our method will automatically collect calibration samples. No specialised markers nor additional assistance from a second person are necessary. Preliminary evaluations highlighted that our fingertip calibration approach achieved comparable accuracy to similar marker-based calibration. While accuracy shows one side of the performance, usability shows another. The majority of participants in our study preferred fingertip calibration, which highlights the potential for simple and intuitive calibration techniques for wearable eye trackers.

Despite eye trackers, including wearable eye trackers, getting smaller, easily accessible, and more affordable, the need for special-purpose hardware is still a major hindrance for users and eye-based interaction. Therefore, as follow-up work, we investigated gaze-based interaction using a single RGB camera (e.g. a webcam). In particular, we explored Pursuits, which are based on smooth pursuit eye movements and have recently become popular because they enable spontaneous and calibration-free interaction. We proposed a
5.1 Summary of Contributions

A novel method to detect pursuits that leveraged an appearance-based gaze estimator and optical flow in the eye region to analyse the eye movement dynamics jointly. Experiments on a 13-participant dataset showed that our method significantly outperformed the current state of the art. Moreover, our approach is competitive to a commercial eye tracker for a small number of pursuit targets. As such, our contribution points the way towards a new family of methods for pursuit interaction directly applicable to an ever-increasing number of devices readily equipped with cameras.

In Chapter 4, we diverge from explicit interaction techniques, such as Pursuits, hand, or gaze gestures, to more subtle ways of understanding users. Modelling human behaviour is essential to many tasks in HCI and one fundamental component that has recently become highly fragmented [166, 206] is human attention. In this dissertation, we addressed the problem of quantifying visual attention during everyday mobile device interactions. Our focus on mobile devices and smartphones, in particular, may seem restrictive. However, they are used by millions (if not billions) of people from all over the world. While visual attention is only one part of the user’s full attention, in HCI it is the most practical and relevant because it enables understanding when, how often, or for how long users attend to their devices.

Towards this goal, we first analysed the feasibility of quantifying visual attention using the front-facing camera of mobile devices. With recent advancements in machine learning, we used an automatic method for eye contact detection and identified three key challenges: The impact of face and eye visibility, the importance of robust head pose estimation, and the need for accurate gaze estimation. Guided by our findings, we addressed challenges specific to mobile interactive settings and proposed a novel approach for eye contact detection with mobile devices. In-depth evaluations on two publicly-available datasets showed increased robustness across devices, users, and environmental conditions. Furthermore, our discussion highlighted the importance of eye contact as a fundamental basis for higher-level visual attention metrics.

Identifying the challenges and proposing novel methods to sense visual attention are only the first steps towards attention quantification in the wild. To fill this gap and inform the design of future mobile attentive user interfaces, we conducted a two-week in-the-wild data collection of video snippets using the front-facing camera of 32 mobile phone users. The Everyday Mobile Visual Attention (EMVA) dataset contains over 14,000 videos, totalling around 472 hours, as well as associated meta-data, sensor data, and device usage logs. Additionally, the dataset provides 10,759 crowd-sourced annotations for eye contact with the device, which serve as a benchmark for future research in this area. Using our proposed method for eye contact detection (Section 4.4), we extracted, for the first time, quantifiable metrics of visual attention. Our findings confirmed the highly dynamic nature of visual attention and further showed that attention allocation is not only user but also
context specific. Taken together, these results are significant because they provide the
first insights into visual attentive behaviour by directly looking at the user through the
front-facing camera of mobile devices. With the public release of the dataset, including
the annotations, we hope to further encourage work in this area of research.

5.2 Future Work

While our work makes several strides towards extending our interaction capabilities with
complex systems, it is not without limitations. Each of the three chapters presented
in this dissertation (Chapter 2, Chapter 3, Chapter 4) addresses limitations that are, in
general, specific to those contributions. Section 2.6 discusses limitations of our collocated
multi-user gestural interaction technique. The limitations, challenges, and opportunities in
eye-based interaction are presented in Subsection 3.2.5, Subsection 3.3.4, and Section 3.4.4.
Limitations specific to quantifying mobile visual attention are discussed in Section 4.4.3
and Subsection 4.7.1. In the following sections, we reflect on more general limitations of
the work proposed in this dissertation.

Advances in Sensing Methods. Almost all of our contributions rely on different algo-
rithms and methods to enable the desired interactive scenario. For example, in Chapter 2,
our collocated multi-user interaction technique uses template matching based on dynamic
time warping to detect hand gestures from inertial data. Such an approach works well and
fast in a user-dependent setting, however, in user-independent scenarios the performance
of our approach decreased significantly rendering the method unusable. Improvements to
the gesture recognition algorithm will naturally improve the user experience. For example,
learning-based methods are particularly well suited as they can learn robust models from
large datasets, which could then generalise to many users. One such recent approach is
GestEar [25], which uses a convolutional neural network for recognising simple gestures
on a smartwatch by combining motion sensing and audio data. Algorithmic improvements
will also benefit our fingertip calibration method proposed in Chapter 3. Our approach uses
computer vision methods to detect the hand and fingertip of users in real time. Methods
that work better across users or environmental conditions already exist [235], however,
they require additional computational capabilities that are not yet widely available on
wearable devices. Nevertheless, given the current technological trend where mobile GPUs
are necessary for a range of applications, future devices will be able to use increased
computational capabilities and benefit from such methodological advancements. For
example, in the future, our eye contact detection method, which currently only runs offline
due to the limited computational capabilities available on smartphones, will be able to run
directly on the device and in real time.


**Privacy Concerns over Continuous Sensing.** Mobile and wearable devices are nowadays equipped with many different sensors, from inertial sensors to environmental sensors (such as temperature or humidity), and even high-resolution cameras. Because these devices are either body-worn or continuously with the user, using them for continuous sensing may raise serious privacy concerns. For example, our collocated multi-user interaction technique uses a microphone for audio recording. Similarly, using the camera to record users all the time may violate their privacy. Moreover, cameras may record the surrounding environment, including bystanders, i.e. people who may not have the chance to give their consent for their recording. With eye trackers becoming easily and widely available, eye tracking data may also reveal information about its users. Because of this, there are a few recent approaches that try to protect the users data using differential privacy [205]. To summarise, there is a need to develop new methods that not only advance the state of the art in sensing, but also protect and guarantee user privacy.

**The Next Generation of User Interfaces.** In this dissertation, we presented several contributions that advance and enhance the way people interact with technology. More specifically, Chapter 2 and Chapter 3 present explicit interaction techniques. User interfaces built on such methods are mostly reactive, i.e. users issue a command and, if the system understands this command, it will act. We argue that the next generation of user interfaces need also to become anticipatory not only reactive. However, for user interfaces to become anticipatory, they have to model and understand human behaviour. In our work, we take the first step in this direction by quantifying visual attention, which is a core component of human attention, especially in HCI. Prior work has shown that device-integrated sensors and body-worn cameras can be used to forecast user attention [206]. As follow-up work, it would be interesting to investigate whether models of human attention, including data from additional sensors or event logs, can be used to predict higher-level user intent. For example, where will users’ look next on the screen, are they likely to interact with the UI, what kind of information are they likely to look for next, and so on.

### 5.3 Closing Remarks

In this dissertation, our goal was to enhance the way people use technology. Inspired by the way people interact with one another, we proposed contributions that extend current interaction modalities using three contextual cues: Physical proximity, natural gestures, and human eye gaze. Towards this goal, we proposed novel contributions to the following three research areas: Collocated multi-user gestural interactions, eye-based human-computer interaction, and quantification of users’ visual attention during everyday mobile device interactions. In each of the three areas, our contributions advance the state
of the art either with novel concepts, prototypes, methods, or findings. Overall, we argue that our results are significant in that they bring us closer to natural user interfaces that can understand and better help users interact with the devices that surround them. While (significant) efforts are still necessary to fulfil this vision, we hope that our insights will encourage future work in any of these areas of research.
Bibliography


International Conference on Management of Data (SIGMOD '99). ACM, New York, NY, USA, 49–60. DOI:http://dx.doi.org/10.1145/304182.304187


[23] Ryoji Ban, Katsuhiko Kaji, Kei Hiroi, and Nobuo Kawaguchi. 2015. Indoor positioning method integrating pedestrian Dead Reckoning with magnetic field and WiFi


[38] Paul Curtis, Mahesh K. Banavar, Sai Zhang, Andreas Spanias, and Vitor Weber. 2014. Android acoustic ranging. In *The International Conference on Information, Intelligence, Systems and Applications (IISA ’14)*. DOI:http://dx.doi.org/10.1109/IISA.2014.6878721


on Sensor-based Activity Recognition and Interaction (WOAR ’15). ACM, New York, NY, USA, 8:1–8:10. DOI:http://dx.doi.org/10.1145/2790044.2790046


and Machine Intelligence 37, 3 (Mar 2015), 583–596. DOI:http://dx.doi.org/10.1109/tpami.2014.2345390


[94] Zhiping Jiang, Jinsong Han, Chen Qian, Wei Xi, Kun Zhao, Han Ding, Shaojie Tang, Jizhong Zhao, and Panlong Yang. 2016. VADS: Visual attention detection with a smartphone. In *IEEE International Conference on Computer Communications (INFOCOM ’16)*. 1–9. DOI:http://dx.doi.org/10.1109/INFOCOM.2016.7524398


[114] Peter Kiefer, Ioannis Giannopoulos, Dominik Kremer, Christoph Schlieder, and Martin Raubal. 2014. Starting to Get Bored: An Outdoor Eye Tracking Study of


Workshops (BSN '14 Workshops). 46–49. DOI:http://dx.doi.org/10.1109/BSN. Workshops.2014.17


[178] Martin Pielot, Tilman Dingler, Jose San Pedro, and Nuria Oliver. 2015. When Attention is Not Scarce - Detecting Boredom from Mobile Phone Usage. In *Proceedings of the International Joint Conference on Pervasive and Ubiquitous Computing*


[206] Julian Steil, Philipp Müller, Yusuke Sugano, and Andreas Bulling. 2018. Forecasting User Attention During Everyday Mobile Interactions Using Device-integrated


[241] Zhefan Ye, Yin Li, Yun Liu, Chanel Bridges, Agata Rozga, and James M. Rehg. 2015. Detecting bids for eye contact using a wearable camera. In *IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG ’15)*, Vol. 1. 1–8. DOI: http://dx.doi.org/10.1109/FG.2015.7163095


[253] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2017. It's Written All Over Your Face: Full-Face Appearance-Based Gaze Estimation. In IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW ’17). IEEE, 2299–2308. DOI:http://dx.doi.org/10.1109/CVPRW.2017.284


Web References and Image Sources

All links were accessed on the 9th of December 2020 and were correct at the time of publication.


Web References and Image Sources


Short Curriculum Vitae

Mihai Bâce

Personal Information

Date of birth: 14th of June 1989
Place of birth: Satu Mare, Romania
Nationality: Romanian

Education

2014 – 2020: Dr. Sc. in Computer Science
Department of Computer Science, ETH Zürich, Switzerland

2012 – 2014: M.Sc. in Computer Science
École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

2008 – 2012: B.Sc. in Computer Science
Technical University of Cluj-Napoca, Romania

2004 – 2008: Romanian Baccalaureate (high school degree)
Emil Racoviță National College, Cluj-Napoca, Romania

Professional Experience

2014 – 2020: Research and Teaching Assistant, ETH Zürich, Switzerland

2014: Research Intern, ABB Corporate Research, Switzerland

2013 – 2014: Software Development Engineer Intern, Microsoft, Ireland

2010 – 2012: Research Assistant, Technical University of Cluj-Napoca, Romania