Gaze-based Multimodal Interaction:
Methods for Active and Passive Gaze-based Interaction

Von der Fakultät für Informatik, Wirtschafts- und Rechtswissenschaften der Carl von Ossietzky Universität Oldenburg zur Erlangung des Grades und Titels

Doktor der Ingenieurswissenschaften (Dr.-Ing.)

angenommene Dissertation
von **Herrn Michael Johannes Barz**
Geboren am 05.01.1991 in Saarbrücken

Gutachter
Prof. Dr.-Ing. Daniel Sonntag
Prof. Dr. Antonio Krüger
Prof. Dr. Susanne Boll

Tag der Disputation
11.04.2025

Acknowledgement

Writing a dissertation is like climbing a mountain. You need certain prerequisites, the right equipment, and, above all, fellows who motivate and support you along the way. I am very grateful for the fellows on my dissertation journey and the support I have received from several sides over this long time. I would like to thank you all!

My research journey began with my computer science studies at Saarland University in 2010. At that time, the courses of Prof. Antonio Krüger's Ubiquitous Media Technology Lab aroused my interest. During this time, I also became acquainted with the German Research Center for Artificial Intelligence (DFKI), as Prof. Krüger was also co-head of the Intelligent User Interfaces department with Prof. Wolfgang Wahlster. Eventually, I wrote my Bachelor's thesis there, supervised by Denise Kahl, and my Master's thesis, co-supervised by Florian Daiber and Andreas Bulling (then at the Max Planck Institute for Informatics). At the end of 2015, I started a position as a researcher at DFKI in Daniel Sonntag's research group, which was part of the IUI department, and became the Interactive Machine Learning department at the end of 2019. Within the last (almost) 10 years, I worked together with many great colleagues. I want to especially thank Alexander Prange, Omair Shahzad Bhatti, and Hans-Jürgen Profitlich for the countless helpful discussions on research and research administration. I also want to thank the rest of our team for their support. Last but not least, I want to thank Prof. Daniel Sonntag for his guidance and advice throughout my scientific career.

During my journey, I also received immense support from my family and my friends from Turnverein St. Ingbert. A big endeavor like writing a doctoral thesis cannot be done without such support, so this is as much your success as it is mine. I want to particularly thank my parents, Winfried and Petra, my sister, Claudia, and my wife, Nathalie. I am also very happy about the newest member of our family, my daughter Amelie.

Thank you all for your guidance, for sharing your experiences, for providing necessary distractions, and for your support in various situations, whether in connection with research or at home, e.g., when renovating a house. You were great fellows on my journey to completing my dissertation. With your help, I was able to climb this mountain.

Abstract

Humans use their sense of vision to perceive their environment and make situation-aware decisions. The eyes naturally coordinate with, for instance, hand movements and speech generation to focus on relevant visual information for that task. People tend to look at objects when they aim to grasp or refer to them in a dialogue. Modern eye tracking technology allows developers to incorporate real-time gaze information as input to multimodal human-computer interfaces. It can be used as an active or passive input modality in multimodal interfaces: a user can influence a system via explicit eye movements (active), and a system can implicitly derive information about the user and their environment by observing the eye movement behavior and fixated objects in the environment (passive). However, many issues remain underexplored, so the technology cannot be widely deployed in interactive intelligent systems. This thesis aims to address related challenges, following the main research question "How can we enable effective and efficient gaze-based user interfaces and their development?". This thesis presents new approaches and methods with the goal of addressing the challenges when using gaze as input in interactive systems. We contribute by developing two methods for active gaze-based interaction and three approaches for passively interpreting the human gaze signal. Further, we outline a framework for gaze-based multimodal interaction, relating to multimodal-multisensor and intelligent user interfaces, addressing the question of how such systems can be designed and developed.

As the first part of the main research question, I investigate how gaze can be used as an active input modality. The first techniques for the corresponding **active gaze-based interaction** date back to the 1990s. Advances in eye tracking technology and methods for analyzing the gaze signal have improved ever since, enabling the development of a broad range of interaction techniques. Many approaches address how absolute eye gaze can be used as a primary input modality in direct manipulation interfaces. A prominent example is gaze-based object selection using the absolute gaze position and an additional selection trigger, like a dwell or a button click. However, errors in gaze estimation can severely hamper the usability and performance of such interfaces because selection targets can be missed, which leads to no or wrong selections. Calibration-free interaction techniques exist that circumvent the problem by, e.g., correlating smooth pursuit movements of the eye with animated on-screen objects or detecting gaze gestures in relative eye movements. This thesis addresses the problem of gaze estimation errors when using gaze as an active input modality. The corresponding partial research question concerning active gaze-based interaction is: "How can the negative impact of gaze-estimation errors on gaze-based interaction be reduced when gaze is used as an active input modality?" I investigate how the gaze-estimation error can be modeled and handled in real-time interaction and how interaction can be realized without

user calibration and accurate gaze estimation.

As the second part of the main research question, I investigate how gaze can be used as a passive input modality. Corresponding **passive gaze-based interaction** is based on observing and interpreting a user's gaze signal and is closely related to research on human vision in psychology and neuroscience. Research has found that a person's eye movements depend on their task and the interaction context, and that eye movements are an important factor for the efficiency of human-human dialogue and collaboration. This inspired researchers to investigate novel methods for analyzing the gaze signal to enable more efficient, situation-aware human-machine interfaces. The main challenges include building effective and generalizable models and applying these models in interactive systems. The partial research question concerning passive gaze-based interaction is: "How can we build effective machine learning models for interpreting human eye movements in the context of passive gaze-based interaction?" In this thesis, I investigate novel approaches for interpreting the human gaze signal using machine learning for three use cases: inferring the search target of a visual search, estimating the perceived relevance of a text that has been read by a user, and semi-automatic detection of visual attention to areas or objects of interest.

As the final part of the main research question, I discuss how individual techniques could be integrated into gaze-based interfaces, following the research question "How can we effectively design and develop gaze-based interaction systems?" In this thesis, I outline a **framework for building gaze-based multimodal interaction** systems, based on a generic architecture for intelligent user interfaces (IUI) with a strong relation to multimodal-multisensor interfaces (MMI). First, I focus on enabling gaze input in multimodal-multisensor interfaces. This thesis presents the multisensor-pipeline (MSP), a lightweight, flexible, and extensible framework for prototyping multimodal-multisensor interfaces based on real-time sensor input. Second, I discuss how the methods and approaches presented in this thesis relate to the main building blocks of a generic IUI software architecture. Eventually, I provide an outlook on future research directions and open challenges concerning gaze-based multimodal interaction.

Zusammenfassung

Kurzzusammenfassung in Deutsch

Eye Tracker ermöglichen die Nutzung von Echtzeit-Blickinformationen als Eingabe für Benutzerschnittstellen. Sie kann als aktive oder passive Eingabemodalität in multimodalen Systemen eingesetzt werden: Ein Benutzer kann ein System durch explizite Augenbewegungen beeinflussen, und ein System kann durch die Analyse von Eye-Tracking-Daten implizit Informationen über den Benutzer und die Umgebung ableiten. Allerdings sind viele Fragen ungeklärt, sodass die Einsatzmöglichkeiten eingeschränkt sind. Diese Arbeit befasst sich mit den damit verbundenen Herausforderungen der zentralen Forschungsfrage folgend: „Wie können wir effektive und effiziente blickbasierte Benutzerschnittstellen und deren Entwicklung ermöglichen?“. Die Arbeit stellt neue Ansätze und Methoden vor, die Blickinformationen als Eingabe in interaktiven Systemen ermöglichen. Darüber hinaus wird ein Framework für blickbasierte multimodale Interaktion im Kontext multimodaler Multisensor- und intelligenter Benutzerschnittstellen.

Short Summary in English

Modern eye tracking technology allows developers to incorporate real-time gaze information as input to multimodal human-computer interfaces. It can be used as an active or passive input modality in multimodal interfaces: a user can influence a system via explicit eye movements, and a system can implicitly derive information about the user and the environment by analyzing eye tracking data. However, many issues remain underexplored, so the technology cannot be widely deployed in interactive intelligent systems. This thesis aims to address related challenges, following the main research question: “How can we enable effective and efficient gaze-based user interfaces and their development?”. It presents new approaches and methods with the goal of addressing the challenges when using gaze as input in interactive systems. It further outlines a framework for gaze-based multimodal interaction, relating it to multimodal-multisensor and intelligent user interfaces.

Contents

I Introduction **1**

1 Motivation **3**

1.1 Gaze-based Multimodal Interaction 3

1.2 Overview of the Thesis and its Contributions 11

2 Background **19**

2.1 The Human Eye, Eye Movements, & Visual Attention 19

2.2 Eye Tracking Technology 21

2.3 Ethics and Privacy 30

3 Related Work **31**

3.1 Human Gaze as an Active Input Modality 31

3.2 Human Gaze as a Passive Input Modality 41

II Human Gaze as an Active Input Modality **51**

4 Error-aware Gaze-based Interaction **53**

4.1 Software Architecture 54

4.2 Prototype Implementation 56

4.3 User Study 59

4.4 Advanced Compensation Methods 63

4.5 Discussion 67

4.6 Conclusion 69

5 Calibration-free Gaze-based Interaction **71**

5.1 Gaze-based Authentication 72

5.2 Evaluation 74

5.3 Discussion 79

5.4 Conclusion 82

III Human Gaze as a Passive Input Modality 83

6 Inferring Visual Search Targets 85
6.1 Target Inference in Constrained Settings . . . 86
6.2 Target Inference in Natural Interaction Settings . . . 90
6.3 Conclusion . . . 99

7 Estimating Document Relevance 101
7.1 Data Collection: gazeRE Dataset . . . 102
7.2 Gaze-based Relevance Estimation . . . 107
7.3 Discussion . . . 111
7.4 Conclusion . . . 121

8 Visual Attention Modelling 123
8.1 Fixation-to-AOI Mapping with Pre-trained Models . . . 125
8.2 Interactive Fixation-to-AOI Mapping . . . 140
8.3 Conclusion . . . 162

IV Towards a Gaze-based Multimodal Interaction Framework 165

9 The Multisensor-Pipeline (MSP) 167
9.1 Related Systems and Frameworks . . . 168
9.2 Implementation of the Multisensor-Pipeline . . . 169
9.3 Discussion . . . 172

10 Towards a Framework for Eye Tracking in IUIs 175
10.1 Relation of the Presented Methods to IUIs . . . 175
10.2 Using MSP for Gaze-based Multimodal Interaction . . . 178

V Conclusion 181

11 Contributions & Outlook 183
11.1 Human Gaze as an Active Input Modality . . . 184
11.2 Human Gaze as a Passive Input Modality . . . 185
11.3 Gaze-based Multimodal Interaction Framework . . . 188
11.4 Relation to Research Projects . . . 189
11.5 Dissemination & Impact . . . 198

References 199

Part I

Introduction

Chapter 1

Motivation

Developing gaze-based multimodal interaction systems is connected to many challenges, including handling the error in gaze estimation for active gaze-based interaction, building effective machine learning models for passive gaze-based interaction, and designing and developing gaze-based interaction systems. In the following, I introduce essential background on gaze-based multimodal interaction in section 1.1 and provide an overview of the thesis, detailing the research questions and contributions, in section 1.2.

1.1 Gaze-based Multimodal Interaction

This thesis presents novel approaches and methods in the field of gaze-based user interfaces. Hereby, we consider an eye tracker as one sensor or gaze as one modality in multimodal-multisensor interfaces (Oviatt et al., 2017b). Gaze can be used as an active or passive input modality in multimodal interfaces: a user can influence a system via explicit eye movements (active), and a system can implicitly derive information about the user and its environment by observing the eye movement behavior and fixated objects in the environment (passive). Consequently, gaze-based interfaces are closely related to intelligent user interfaces that "aim to improve the efficiency, effectiveness, and naturalness of human-machine interaction" by incorporating benefits of, among others, multimodal interaction and artificial intelligence technologies (Maybury and Wahlster, 1998, pp. 2-3). Qvarfordt (2017) introduced the design space for gaze-informed multimodal interaction to classify gaze-based, multimodal, and intelligent user interfaces along two axes. We use this design space to classify the contributions of this thesis into active and passive gaze-based interfaces (we will refer to this as the awareness level). In the following, we briefly introduce the concepts of multimodal-multisensor interfaces and intelligent user interfaces. Further, we describe the design space of gaze-informed multimodal interaction and how it relates to the two aforementioned concepts.

1.1.1 Multimodal-Multisensor Interfaces

A brief definition of multimodal interfaces is provided by Oviatt and Cohen (2015, p. 5): "*Multimodal interfaces* support input and processing of two or more modalities, such as speech, pen, touch and multi-touch, gestures, gaze, and virtual keyboard, which may be used simultaneously or alternately. User input modes can involve recognition-based technologies (e.g., speech) or discrete input (e.g., keyboard, touch). Some modes may express semantically rich information (e.g., pen, speech, keyboard), while others are limited to simple selection and manipulation actions that control the system display (e.g., gestures, touch, sensors)." Gaze, like any other input modality, can be used in an active or passive input mode: a user can influence a system via explicit eye movements (active), and a system can implicitly derive information about the user and its environment by observing the eye movement behavior and fixated objects in the environment (passive). "*Active input modes* are ones that are deployed by the user intentionally as explicit input to a computer system (e.g., speaking, writing, typing, gesturing, pointing) [and] *passive input modes* refer to naturally occurring user behavior or actions that are recognized and processed by the system (e.g., facial expressions, gaze, physiological or brain wave patterns, sensor input such as location). They involve user or contextual input that is unobtrusively and passively monitored, without requiring any explicit user command to a computer" (Oviatt and Cohen, 2015, p. 5). The class of multimodal-multisensor interfaces extends multimodal interfaces by incorporating sensor information as additional contextual cues: "*Multimodal-Multisensor Interfaces* combine one or more user input modalities with sensor information that involves passive input from contextual cues (e.g., location, acceleration, proximity, tilt) that a user does not need to consciously engage" (Oviatt and Cohen, 2015, p. 6). In this paper, we classify gaze-based interfaces as multimodal-multisensor interfaces, since data streams from eye trackers can be counted as passive input and sensor-based cues. However, we use the terms passive input and sensor-based cues interchangeably in many parts. For instance, in foveated rendering (Duchowski, 2018), i.e., when real-time gaze input is used to determine which parts of a virtual reality scene should be rendered in great detail, gaze can be counted as a pure sensor-based cue that serves as an input to optimize the efficiency of the rendering pipeline, but also as a naturally occurring user behavior used for the same purpose.

Why is it interesting to strive towards multimodal rather than unimodal interaction? A basis for answering that question can be found in the theoretical foundations of human multisensory processing. For a detailed introduction to this topic, we refer to Oviatt (2017) and Oviatt and Cohen (2015, Chapter 4 to 5). In the following, we briefly summarize the main motivations for and advantages that can be expected when building multimodal-multisensor interfaces (Oviatt and Cohen, 2015, pp. 17-25):

- User preference and natural interaction patterns: People prefer to interact multimodally for many tasks and application domains.
- Flexible interaction patterns: Multimodal interfaces can enable users to choose the best input modality that fits the task at hand and the possibly dynamic interaction context.
- Accommodation of individual differences: Individual differences in ability and preference can be addressed, making multimodal interaction technology an important building block

for accessible human-computer interaction.

- Efficiency: Efficiency can be improved, particularly concerning the input of spatial information and when counting recovery from errors of, e.g., recognition systems.
- Superior error handling: Multimodal interfaces may be advantageous for error avoidance and recovery. For instance, through "*mutual disambiguation* [which] involves disambiguation of signal- or semantic-level information in one error-prone recognition modality by using partial information supplied by another modality" (Oviatt and Cohen, 2015, p. 106).
- Minimization of cognitive load: User's cognitive load can be reduced by allowing them to convey information using the most efficient modality. For instance, pointing with a pen more efficiently conveys a spatial position than speech. Further, users' working memory is limited, but its effective size can be expanded when multiple modalities are used.
- Expressive power and simulation of cognition: Expressively powerful interfaces, including, e.g., digital pen input, can stimulate cognition and consequently improve users' performance in, e.g., idea generation, problem-solving, and inferential reasoning.

The advantages mentioned above cannot be taken for granted. Oviatt and Cohen (2015, Chapter 6) highlights the human-centered design methodologies' essential role in creating successful multimodal interfaces. Nowadays, the smartphone is likely the most prominent and widespread device facilitating multimodal-multisensor interfaces. Modern mobile devices allow users to use, e.g., touch-based gesturing, handwriting via pens, and speaking in an active input mode, and carry proximity and geolocation sensors for context-aware features (cf. Oviatt and Cohen (2015, pp. 12-16)). For a complete guide to and overview of multimodal-multisensor interfaces and related technologies, we refer to the book series by Oviatt et al. (2017a, 2018, 2019).

1.1.2 Intelligent User Interfaces

Maybury and Wahlster (1998, p. 2) defined Intelligent User Interfaces (IUIs) as "human-machine interfaces that aim to improve the efficiency, effectiveness, and naturalness of human-machine interaction by representing, reasoning, and acting on models of the user, domain, task, discourse, and media." According to their vision, benefits for users will arise from the support of multimodal input analysis and multimodal output generation, semi- or fully automated completion of tasks, and advanced interaction management. They argue that a successful realization of "intelligent human-computer interaction requires a synergistic integration of" enabling technologies from fields like artificial intelligence (including text processing, spoken language processing, knowledge representation, and planning) and human-computer interaction (Maybury and Wahlster, 1998, pp. 2-3). A similar perspective is adopted in the context of interactive intelligent systems, i.e., "interactive systems that incorporate some sort of AI technology (or technology that at one time was viewed as belonging to AI)" (Jameson et al., 2009, p. 11). Jameson et al. (2009) introduces the metaphor of a *monocular view* versus a *binocular view* when designing such systems. He proclaims that "when creating algorithms or systems that are supposed to be used by people, we should adopt a 'binocular' view of users' interaction with intelligent systems: a view that

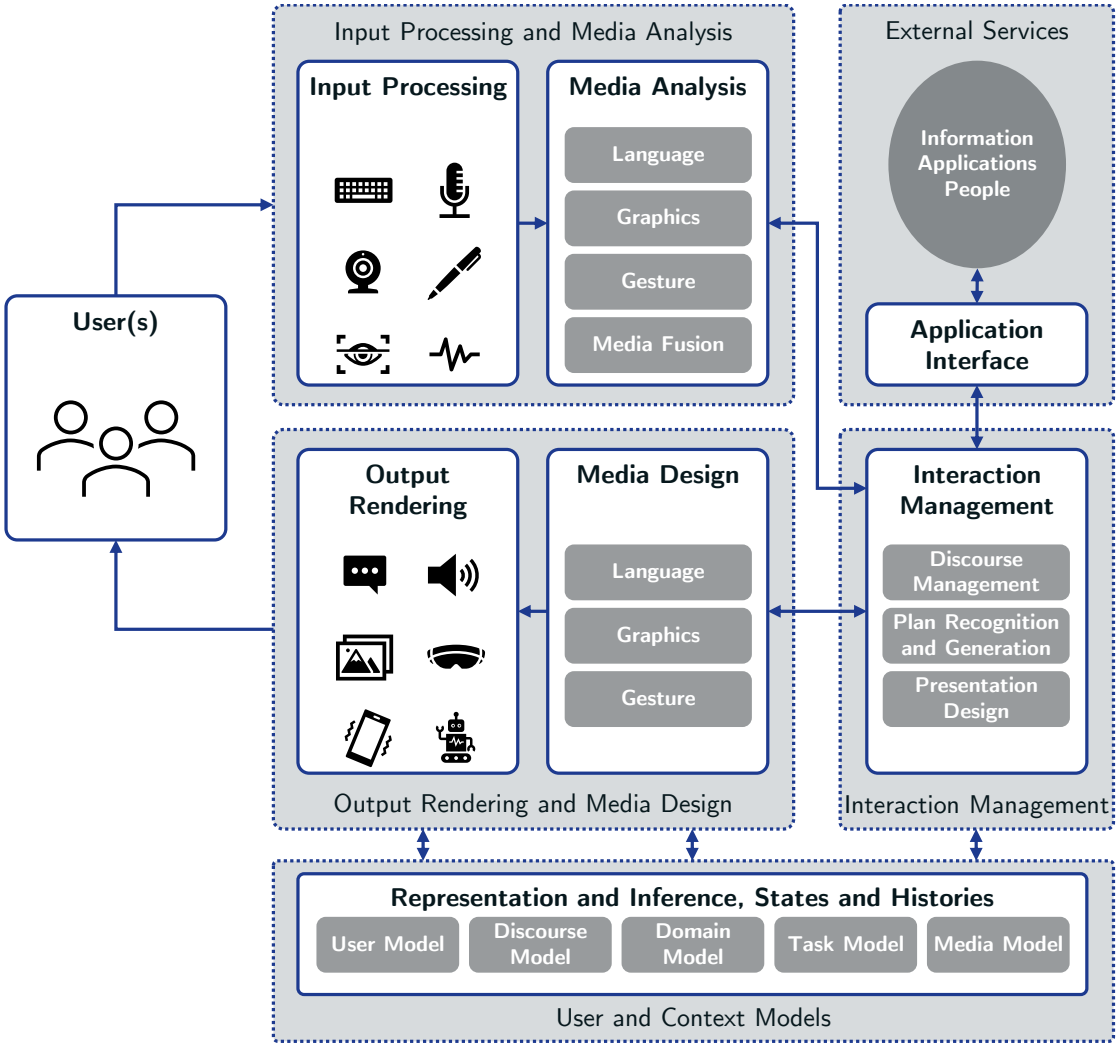


Figure 1.1: Architecture of Intelligent User Interfaces adopted from Maybury and Wahlster (1998).

regards the design of interaction and the design of intelligent algorithms as interrelated parts of a single design problem“ (Jameson et al., 2009, p. 11). Conversely, the *monocular view* refers to researchers' focus on only one aspect of intelligent interaction, either developing an (AI) algorithm or designing user interaction. Maybury and Wahlster (1998, p. 7) conclude that the “principal areas of intelligent interface research include *analysis* of input (e.g., spoken, typed, and handwritten language; gestures, including hand, eye, and body states and motion), *generation* (planning or realization) or coordinated output, [and] *modeling* of the user, discourse, task, and situation and *interaction management*, including possible tailoring of interaction to the user, task, and/or situation.” Figure 1.1 shows an adapted version of the long-acknowledged conceptual architecture by Maybury and Wahlster (1998, p. 3), which indicates the interrelation of these areas. In this thesis, we consider the functional coherent elements *Input Processing & Media Analysis*, *Interaction Management*, *Output Rendering & Media Design*, *User & Context Models*, and *External Services* as the main building blocks of an intelligent user interface. We briefly explain the role of each building block and, in part IV, we relate the methods and approaches presented in this thesis to them.

1.1.2.1 Input Processing & Media Analysis

One of the building blocks is located on the input side of an IUI, namely *Input Processing and Media Analysis*. The focus of related methods is to facilitate human-like signal processing and interpretation capabilities in human-machine interaction, such as multimodal input interpretation (Maybury and Wahlster, 1998, p. 8). This shows the strong relation between IUIs and multimodal-multisensor interfaces (see section 1.1.1). The main goals include increasing the interaction's efficiency, effectiveness, and naturalness.

1.1.2.2 Interaction Management

Interaction Management is the center part between *Input Processing & Media Analysis* and *Output Rendering & Media Design*. Components in this building block consume pre-processed active and passive user inputs, manage the state of the interaction or discourse, and decide on whether and how to act or respond to a user.

1.1.2.3 Output Rendering & Media Design

Output Rendering and Media Design refers to technologies concerning the “generation of coordinated multimedia output” (Maybury and Wahlster, 1998, p. 9), including multimedia presentation design, automated graphics design, and automated layout. Components in this building block are triggered by components from the *Interaction Management* block.

1.1.2.4 User & Context Models

User and context models are key in enabling adaptive and situation-aware interaction in IUIs. Components in the *User & Context Models* building block focus on model acquisition, i.e., on how to build effective user, discourse, or other context models, tracking the state of these models, and how to use them (Maybury and Wahlster, 1998, pp. 9-10). This can be applied in most parts

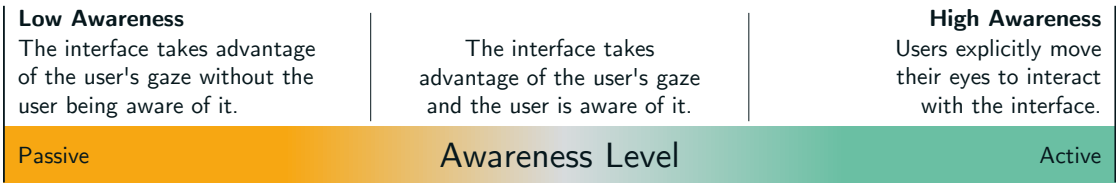


Figure 1.2: Awareness level dimension of gaze-informed multimodal interfaces.

of an IUI, including *Input Processing & Media Analysis*, *Interaction Management*, and *Output Rendering & Media Design*.

1.1.2.5 External Services

Via application programming interfaces (APIs), IUIs can use any external service like knowledge bases, crowd-sourcing platforms for human computation systems, and, meanwhile, many AI-based services like image classification, automatic speech recognition, natural language understanding, and language generation. While *External Services* are mainly connected to *Interaction Management*, it can also help in media analysis and design.

1.1.3 Design Space of Gaze-informed Multimodal Interaction

Gaze can be used in many ways in multimodal and intelligent systems. We adopt the term gaze-informed (or gaze-based) multimodal interaction from Qvarfordt (2017) to refer to such interfaces. Qvarfordt (2017, pp. 382-383) suggests a differentiation of gaze-based interfaces based on two dimensions depicting "how actively the users control their gaze, and how stationary the user is while interacting with the system". They span the design space of gaze-informed multimodal interaction: interaction can be rated on a continuum from active to passive and from stationary to mobile, i.e., according to users' level of awareness and mobility. We name these dimensions the *awareness level* and the *mobility level*. The awareness level directly relates to the concept of active vs. passive input modes in multimodal-multisensor interfaces (see section 1.1.1). In addition, gaze-based interaction is related to intelligent user interfaces on multiple levels (see section 1.1.2). Active gaze-based interaction technologies can be used in input processing and media analysis to improve the effectiveness and efficiency of interaction. Further, passive gaze-based interaction technologies can be used to model the user and context, and to guide the interaction management.

1.1.3.1 Awareness Level

The distinction between using gaze in an active or passive input mode defines the range of the awareness level dimension of the design space. Interfaces or interactive systems would be sorted along this axis depending on how much "the user is consciously deploying their actions with the intent of providing input to a computer system" (Oviatt and Cohen, 2015, p. 2). As illustrated in figure 1.2, low awareness refers to using gaze in a passive input mode while the user is not aware of that fact. High awareness refers to using gaze in an active input mode while the user is fully aware

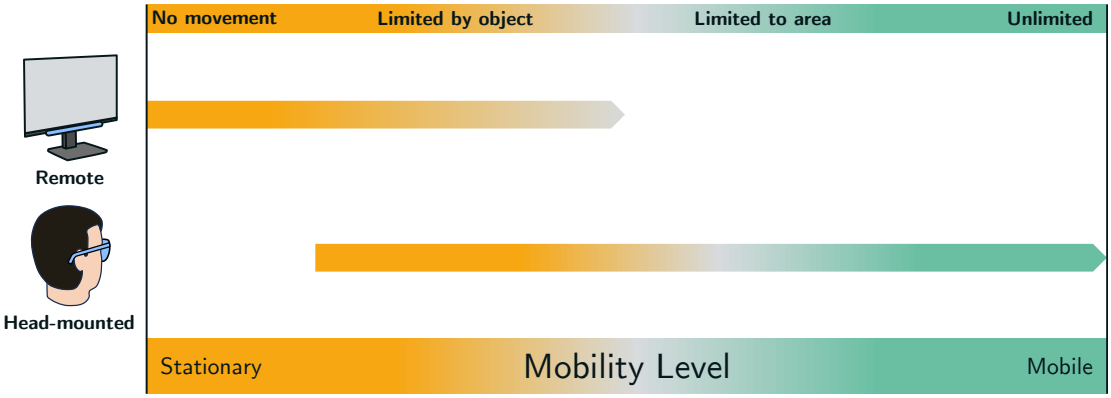


Figure 1.3: Mobility level dimension of gaze-informed multimodal interfaces.

of that fact. When the gaze is used in an active input mode, a user's intentional eye movements directly impact the interaction. We call these *active gaze-based interfaces*. Examples include gaze-based selection (Stellmach and Dachselt, 2012b) and text input (Feng et al., 2021). Such interfaces have an awareness level, i.e., the users know that their gaze impacts the interaction. Section 3.1 provides a more detailed overview of active gaze-based interfaces. When the gaze is used in a passive input mode, naturally occurring eye movement behavior is monitored and can indirectly impact the interaction. We call them *passive gaze-based interfaces*. Typically, the gaze signal is used to infer information about the user, such as the ongoing activity (Li et al., 2018a) or the induced cognitive load of an activity (Klingner, 2010), to tailor the interaction to the user and its current state. In such cases, the users are less or not aware that their gaze is used to influence the interaction. Consequently, these interfaces would rank low in terms of awareness level. A more detailed overview of related approaches can be found in section 3.2. The awareness level is a continuous scale. For instance, users might be aware that their gaze is used to resolve deictic references in speech (Elepfandt and Grund, 2012) but may forget about this fact if the interaction runs smoothly.

1.1.3.2 Mobility Level

The mobility level is defined by the level of freedom a user has when interacting with the system. Concerning eye tracking, this ranges from settings where the user's head is fixed using a chin rest, i.e., no movements are possible, to mobile settings where users can move freely (see figure 1.3). Some constraints are hereby induced by the eye tracking device (see section 2.2.1). For instance, table-mounted eye trackers have a fixed perspective and enable eye tracking in a limited area only, also called the headbox. Head-mounted eye trackers allow users to move around, sometimes limited by the range of wireless data transmission. Still, the interface might be presented on a screen with a fixed position. The mobility level is also a continuous scale accounting for various degrees of mobility as illustrated in figure 1.3. For instance, interaction with a desktop computer using a remote eye tracker counts as stationary, as movements are limited by the object at which the eye tracker is mounted. However, interaction with fixed head positions, such as through a chin rest, restricts mobility more than a headbox. Similarly, interaction based on head-mounted

eye trackers can vary in mobility. For instance, a user interface shown on a fixed screen requires the screen to be in the field of view of the eye tracker and the user to be close enough to enable interaction (movement is limited by an object, i.e., the screen). Also, range limitations of wireless data transmission can reduce mobility (movement is limited to an area). The interaction would be fully mobile if no movement restrictions existed in a completely independent setup.

1.2 Overview of the Thesis and its Contributions

Eye trackers have evolved from pure diagnostic sensors to input devices for real-time interactive systems (Duchowski, 2018). This was partially driven by advances in eye tracking hardware over the last decade concerning the devices' affordability, performance, and form factor. Also, the characteristics of the human visual system, including the eyes, are essential to this development. Directing the gaze to a specific spot is very efficient, and the gaze direction implicitly reveals where a human allocates its limited attentional resources (Holmqvist and Andersson, 2017, p. 26). This visual attention can be attracted by generally appealing, bottom-up factors and goal- or task-oriented, top-down factors (Borji and Itti, 2013). Modern eye trackers can capture and stream a user's gaze in real-time, allowing developers to use the human gaze as input to interactive systems. The gaze signal can be captured in stationary settings, typically using remote eye trackers, and in mobile settings, using head-mounted eye trackers. Remote eye trackers have a fixed position in the world, usually attached to a screen, and allow little to no movement by a user. Head-mounted eye trackers are worn by the user like glasses and allow more freedom of movement. Still, head-mounted eye trackers can also be used in a stationary setting. From stationary to mobile interaction settings, gaze can be employed as an active input modality or as a passive sensor-based cue, which spans the design space for gaze-based multimodal interaction (cf. section 1.1). Although the human gaze is considered to be a valuable input modality for interactive systems, only very specific user interfaces are available, like gaze-based typing applications for disabled people in the domain of assistive technologies (Feng et al., 2021) and foveated rendering, which enables fast scene rendering through gaze-based control of the rendering quality (Duchowski, 2018), commonly used in high-resolution virtual reality headsets.

Many challenges that hinder a wide deployment of gaze-informed multimodal interfaces remain underexplored. A key problem of active gaze-based interaction is the gaze estimation error: the difference between the estimated and true gaze position can be substantial (Holmqvist et al., 2012), in particular, if the user moves in front of a display (Cerrolaza et al., 2012; Mardanbegi and Hansen, 2012). Moreover, the longstanding Midas Touch problem persists, i.e., the problem of deciding when a gaze should be interpreted as input and when it should not be (Jacob, 1990). Other prohibitive factors include low user acceptance and privacy issues (Steil, 2019), as well as the often mandatory and lengthy calibration process (Vidal et al., 2013). In addition, methods for passive gaze-based interfaces are underexplored and often suffer from low effectiveness and a low generalizability to other users or application domains (Plopski et al., 2022). This thesis addresses key challenges of gaze-based interaction concerning active gaze-based interaction in part II, passive gaze-based interaction in part III, and discusses how to design and develop gaze-based interaction systems in part IV. The following sections provide a detailed motivation for each part in sections 1.2.1 to 1.2.3 and a summary of the research questions and contributions in section 1.2.4.

1.2.1 Human Gaze as an Active Input Modality

Using eye gaze as an active input modality has received a lot of interest during the last three decades (Duchowski, 2018). Many approaches have been presented to use human gaze as an active input modality in multimodal interfaces (Qvarfordt, 2017). Common applications of remote and mobile eye tracking include gaze-based selection and manipulation of objects and gaze-based typing.

> "When gaze is employed as an active input mode, users can directly impact the system by explicitly changing their gaze behavior"
> – Qvarfordt (2017, p. 383)

Key issues that have to be addressed include the Midas touch problem, i.e., the human gaze is always active and it has to be decided when to use the signal and when not to use it, errors in the gaze estimation process, and the usually required and often tedious calibration process of eye tracking devices. A prominent solution to overcome the Midas touch problem is to combine the absolute gaze position with an additional selection trigger like a dwell (Jacob, 1991) or a button click (Zhai et al., 1999). Still, approaches based on the absolute gaze position for selection and manipulation have to cope with the inevitable gaze estimation error, i.e., the problem that the real gaze position deviates from the estimated one (Holmqvist et al., 2012). Common countermeasures include using gaze-to-object mapping algorithms (Špakov, 2011), gaze signal filters (Špakov, 2012), or optimized interface designs (Feit et al., 2017). Other solutions include interacting using gaze gestures (Drewes and Schmidt, 2007) or smooth pursuit eye movements (Vidal et al., 2013), which rely on relative eye movement patterns. These calibration-free interaction techniques do not require accurate gaze estimates and user calibration but introduce other limitations. For instance, gaze gestures can quickly become complex and difficult to remember, and interaction based on smooth pursuits requires constantly moving interface elements that the eyes can follow.

In part II of this thesis, I aim to address the problem of gaze estimation errors that can severely hamper the effectiveness and usability of active gaze-based interaction. I present a framework for handling the gaze estimation error in head-mounted eye tracking and develop a calibration-free interaction technique for remote eye trackers that does not rely on accurate gaze estimates. We present our developments on error-aware gaze-based interfaces in chapter 4 and on calibration-free interaction in chapter 5.

Error-aware Interaction This thesis presents a new class of gaze-based interfaces in chapter 4, namely error-aware gaze-based interfaces that incorporate the inevitable gaze estimation error. We implement and evaluate methods for modeling the error via machine learning and use it for real-time error-adaptive object selection with a monocular head-mounted eye tracker. We demonstrate the positive effects of incorporating error estimates in an active gaze-based interface. The gaze estimation error can severely hamper the usability and performance of mobile gaze-based interfaces given that the error constantly varies for different interaction positions (Barz et al., 2015, 2016a, 2018; Lander et al., 2015; Mardanbegi and Hansen, 2012). We explore error-aware gaze-based interfaces, i.e., interfaces that estimate and adapt to the gaze estimation error on-the-fly. We develop a user interface prototype for gaze-based selection and evaluate it

using three different error compensation methods. First, we implement and evaluate the error compensation method that increases the size of interface components directly proportional to absolute error estimates from four different error models (*NaïveScaling*). Based on our findings, we implement two further compensation methods: one method scales components based on a two-dimensional error distribution (Feit et al., 2017) (*Feit2dDist*) and another one shifts gaze points by a predicted directional error estimate (*PredictiveShift*). We evaluate these compensation methods in a 12-participant user study and show that our *PredictiveShift* method significantly outperforms the others in terms of selection rate, particularly for small gaze targets. These results underline both the feasibility and potential of next-generation error-aware gaze-based user interfaces.

Calibration-free Interaction We implement and evaluate an approach for calibration-free authentication based on saccadic eye movements in chapter 5. Our approach circumvents the requirement of accurate gaze estimates. Concretely, we demonstrate a PIN entry method for public displays that is based on saccadic eye movements. The technology is demonstrated with a remote eye tracking device. We chose this demo usecase because the usage of interactive public displays has increased, including the number of sensitive applications and, hence, the demand for user authentication methods. In this context, gaze-based authentication was shown to be effective and more secure but significantly slower than touch- or gesture-based methods. We implement a calibration-free and fast authentication method based on saccadic eye movements for situated displays. In a user study (n=10), we compare our new method with *CueAuth* from Khamis et al. (2018b), an authentication method based on smooth pursuit eye movements. The results show a significant improvement in accuracy from 82.94% to 95.88%. At the same time, we found that the entry speed can be increased enormously with our method, on average, from 18.28 s down to 5.12 s, which is comparable to touch-based input.

1.2.2 Human Gaze as a Passive Input Modality

Eye tracking was initially used as a diagnostic tool to understand the human visual system (Duchowski, 2002, 2018). Typically, eye movements were recorded during a user study and analyzed post-hoc. Research has found that the human gaze can be guided by salient bottom-up factors and task-related top-down factors in a scene (Borji and Itti, 2013). When humans perform a task, the number of fixations to irrelevant but salient objects drops, while the fixations to task-relevant objects, i.e., top-down factors, increase (Yarbus, 1967; Land and Hayhoe, 2001; Holmqvist and Andersson, 2017; DeAngelus and Pelz, 2009; Rothkopf et al., 2016). Passive gaze-based interfaces observe and interpret such eye movements and react to them during the interaction. In contrast to active gaze-based interfaces, the eye movements are not explicitly controlled by the user to generate a system response. One example can be found in gaze-contingent displays that render contents with a high level of detail in the foveal region to match the high visual acuity and with reduced details in the outer regions of the visual field of view to speed up the rendering (Duchowski, 2018). Other approaches observe eye movements to infer information about users, such as their preferences and ongoing activities, to adapt the interface to them (Qvarfordt, 2017). For instance, Buscher (2010) investigated whether gaze can be exploited

to enhance information retrieval systems. Toyama (2015) presented approaches for analyzing the visual content viewed by a user for building attention-aware interactive systems.

In part III of this thesis, I introduce novel methods for interpreting the human gaze signal and depict how they can drive the development of attention-aware user interfaces. I implement and evaluate approaches for inferring visual search targets, modeling the relevance of text passages for information retrieval systems, and identifying viewed objects in a scene for attention-aware computing. This thesis presents developments on inferring search targets during visual search processes in chapter 6, on inferring the relevance of read paragraphs in chapter 7, and on detecting visual attention to ambient objects in chapter 8.

Search Target Inference Visual search is a perceptual task in which humans aim at identifying a search target object, such as a traffic sign, among distractor objects. Search target inference subsumes computational methods for predicting this target by tracking and analyzing overt behavioral cues of that person, e.g., the scanpath and fixated visual stimuli. We investigate whether eye tracking can be used to infer search targets during visual search process in chapter 6. We develop two novel encodings for scanpaths based on the fixated visual stimuli in a scene and investigate their impact on the effectiveness of search target inference models in section 6.1. The *Bag of Deep Visual Words* encoding is based on an approach from the literature that uses *Bag of Visual Words*, a common method to encode a set of images in computer vision. It encodes image sequences from scanpaths using a pre-trained convolutional neural network to enable search target inference based on machine learning. We evaluate it using an available dataset that includes visual search trials for collages of book covers in comparison to a re-implementation of the *Bag of Visual Words* method. The results show that our new scanpath encoding outperforms the baseline from the literature, in particular, when excluding fixations on the search target. We also present an approach for inferring search targets in natural scenes in section 6.2. We aim to predict the class of the image segment surrounding the search target determined by SegNet, a deep learning image segmentation model (Badrinarayanan et al., 2017). We develop a novel method for encoding scanpaths from a visual search as *Histograms of Fixated Image Segments* and compare it to our previous approach. We create a new search target inference dataset for this purpose. The results show that our segmentation-based sequence encoding outperforms the *Bag of Deep Visual Words* method and enables target inference in natural environments in exchange for less spatial precision.

Implicit Relevance Feedback Eye movements were shown to be an effective source of implicit relevance feedback in constrained search and decision-making tasks. Recent research suggests that gaze-based features, extracted from scanpaths over short news articles, can reveal the perceived relevance of read text with respect to a known trigger question (Bhattacharya et al., 2020b,a). We aim to confirm this finding and investigate whether it generalizes to multi-paragraph Wikipedia documents requiring readers to scroll down to read the whole text in chapter 7. We conduct a user study (n=24) in which participants read single- and multi-paragraph articles and rate their relevance at the paragraph level with respect to a trigger question. We model the perceived relevance using machine learning and features from the literature. Our results confirm that eye movements can be used to effectively model the relevance of short news

articles, especially when we exclude difficult cases. These documents are on the topic of the trigger questions but are irrelevant. However, our results do not clearly show that the modeling approach generalizes to multi-paragraph document settings. We publish our data and code for feature extraction under an open-source license to enable future research in gaze-based implicit relevance feedback.

Visual Attention Modeling Processing visual stimuli in a scene is essential for the human brain to make situation-aware decisions, and mobile eye tracking can track a user's visual attention during that process. For instance, by tracking which ambient objects are fixated. Such information can benefit real-time interactive systems for, e.g., reference resolution in conversational interfaces (Barz et al., 2017; Prasov and Chai, 2008). It can also accelerate the analysis of human behavior through diagnostic eye tracking studies. Important stimuli are typically encoded as rectangular areas of interest (AOIs) per video frame, and hypotheses relate to them. This involves tedious manual annotation, as each participant has an individual egocentric video. In this thesis, we implement two methods for automatically mapping fixations to AOIs using pre-trained deep learning models for image classification and object detection in chapter 8. We develop an evaluation framework based on the VISUS dataset (Kurzhals et al., 2014a) and performance metrics from activity recognition (Ward et al., 2006, 2011). We evaluate our methods and discuss their potential and limitations in section 8.1. Further, we address the limited flexibility of this approach by implementing *eyeNotate*, a web-based annotation tool that enables semi-automatic data annotation and learns to improve from corrective user feedback in section 8.2. Users can manually map fixation events to areas of interest (AOI) in a video-editing-style interface (baseline version). Further, our tool can generate fixation-to-AOI mapping suggestions based on a few-shot image classification model (IML-support version). We conduct an expert study with trained annotators (n=3) to compare the baseline and IML-support versions. We measure the perceived usability, annotations' validity and reliability, and efficiency during a data annotation task. We asked our participants to re-annotate data from a single individual using an existing dataset (n=48). Further, we conducted a semi-structured interview to understand how participants used the provided IML features and assess our design decisions. In a post-hoc experiment, we investigate the performance of three image classification models in annotating the data of the remaining 47 individuals.

1.2.3 Towards a Gaze-based Multimodal Interaction Framework

In part IV of this thesis, we describe our efforts toward building a framework for gaze-based multimodal interaction. First, we demonstrate how we enable gaze input in multimodal-multisensor interfaces (Oviatt et al., 2017b, p. 3) by implementing the Multisensor-Pipeline package for Python. Second, we discuss how the contributions of this thesis can be used in intelligent user interfaces (Maybury and Wahlster, 1998).

Multisensor-Pipeline (MSP) In chapter 9, we present the multisensor-pipeline (MSP), a lightweight, flexible, and extensible framework for prototyping multimodal-multisensor interfaces based on real-time sensor input like gaze from eye trackers. This directly results from our devel-

opments and is published under an open-source license. It enables the integration of gaze-based interaction technologies presented in this thesis from active gaze-based input approaches presented in part II to methods for passive gaze-based interaction presented in part III. We describe the framework, showcase how it can be used to implement gaze-based multimodal interaction and discuss its relation to the methods and approaches presented in this thesis.

Eye Tracking in Intelligent User Interfaces In chapter 10, we discuss how gaze can be used in intelligent user interfaces. We briefly recap the generic architecture introduced by Maybury and Wahlster (1998) and point out how the eye tracking methods and approaches presented in this thesis relate to its main building blocks. In addition, we outline how MSP could be extended toward a framework for implementing gaze-based intelligent user interfaces, i.e., how it can be extended towards a framework for gaze-based multimodal interaction.

1.2.4 Overview of the Research Questions & Contributions

This thesis aims to address some of the essential challenges of gaze-based interaction with the vision of enabling more effective and efficient user interfaces. The main research question of this thesis is: *"How can we enable effective and efficient gaze-based user interfaces and their development?"*

- In part II, I address the problem of gaze estimation errors when using gaze as an active input modality, following the partial research question *"How can the negative impact of gaze-estimation errors on gaze-based interaction be reduced when gaze is used as an active input modality?"* In chapter 4, I develop a methodology for modeling the gaze estimation error in head-mounted eye trackers and demonstrate the positive effects of incorporating error estimates in an error-aware gaze-based interface for object selection. Further, in chapter 5, I present a method for calibration-free authentication via PIN entry based on saccadic eye movements, which does not require accurate gaze estimates, using a remote eye tracker.

- In part III, I address the problem of building effective and generalizable machine learning models in the context of passive gaze-based interaction. The corresponding partial research is *"How can we build effective machine learning models for interpreting human eye movements in the context of passive gaze-based interaction?"* I investigate novel approaches for interpreting the human gaze signal using machine learning for three use cases: inferring the search target of a visual search in chapter 6, estimating the perceived relevance of a text that has been read by a user in chapter 7, and semi-automatic detection of visual attention to areas or objects of interest in chapter 8.

- In part IV, I discuss how individual techniques could be integrated into gaze-based interfaces, following the research question *"How can we effectively design and develop gaze-based interaction systems?"* I outline a framework for building gaze-based multimodal interaction systems, based on a generic architecture for intelligent user interfaces (IUI) with a strong relation to multimodal-multisensor interfaces (MMI). In chapter 9, I present the multisensor-pipeline (MSP), a lightweight, flexible, and extensible framework for prototyping multimodal-multisensor interfaces based on real-time sensor input. In chapter 10, I discuss how the methods and approaches presented in this thesis relate to the main building blocks of a generic IUI software architecture.

Eventually, in part V, I conclude by briefly summarizing the results of this thesis, stating the limitations, and providing an outlook on future research directions. I relate the contributions of this thesis and corresponding future directions to past and ongoing research projects, namely the EU project MASTER and the BMBF projects GeAR, SciBot, and No-IDLE. We also report on dissemination activities related to this thesis and its impact so far.

Chapter 2

Background

Eye tracking is an important tool in various fields, ranging from psychology and neuroscience to human-computer interaction and marketing research. Eye tracking provides valuable insights into human visual perception, attentional processes, and cognitive behavior by analyzing the movements and gaze patterns of the human eye. This work aims to incorporate eye movements and their characteristics into gaze-based interfaces and interaction technologies. This section provides background information on key concepts and technologies for the systems presented in this work. The section introduces basic information on the human eye, eye movements, and visual attention. Further, I report on eye tracking hardware, common gaze estimation methods, and typical applications of eye tracking. Related work on active and passive gaze-based interfaces is provided in the respective parts II and III.

2.1 The Human Eye, Eye Movements, & Visual Attention

The human eye is a complex sensory organ that allows us to perceive the visual world around us. Its outer layer consists of the transparent cornea and the sclera, a dense, white, opaque, fibrous tissue with mainly protective functions (Atchison and Smith, 2023). A horizontal section illustrating important parts of the human eye is shown in figure 2.1. Holmqvist and Andersson (2017, pp. 12-13) describe the basic function of the eye as follows: light passes through the cornea and pupil, is flipped upside down by the lens, and projects an image onto the retina. The retina, located at the back of the eye, contains specialized photoreceptor cells known as rods and cones, which convert light stimuli into electrical signals that are transmitted to the brain via the optic nerve. The rods are sensitive to light and support vision at low brightness. The cones enable color vision and are very densely arranged in the area of the fovea, which is located in the back part of the eye and covers only two degrees of the visual field. The cones are less densely arranged in the periphery of the retina. Therefore, humans have a high-acuity vision in the area of the fovea only. The line between a viewed object of interest and the fovea is called the visual axis. It deviates from the optical axis, i.e., "the line joining the centers of curvatures of the refracting surfaces" including the cornea and the lens (Atchison and Smith, 2023, p. 10). The iris regulates the pupil diameter (Atchison and Smith, 2023, p. 4).

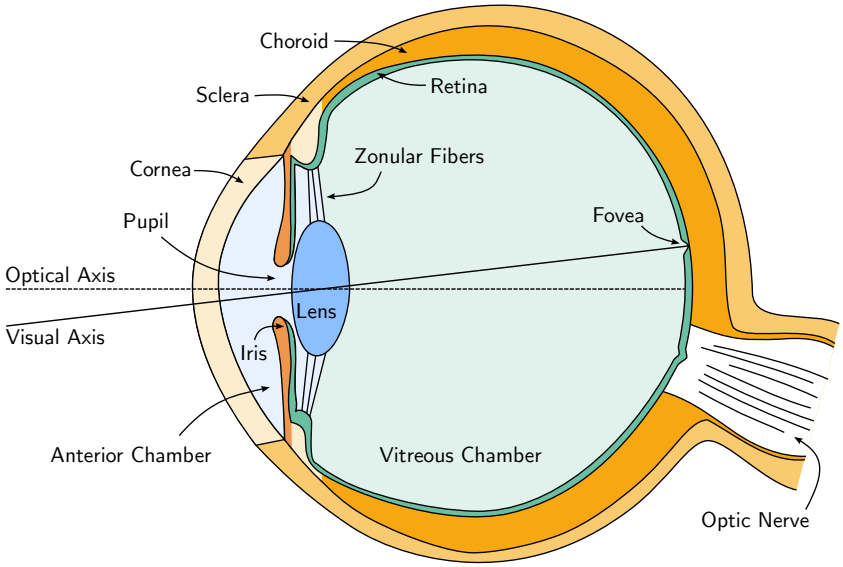


Figure 2.1: Horizontal section of the human eye (based on figure 1.1 in Atchison and Smith (2023, p. 4)), adapted from Talos (2008)).

The human eye can perform several movements, also referred to as oculomotor events, with the aim to, e.g., direct the foveal region to a specific part of a visual scene or maintain high visual acuity on a specific part of it. Three pairs of muscles enable the horizontal (yaw), vertical (pitch), and limited torsional (roll) movements for repositioning the eye in three-dimensional space (Holmqvist and Andersson, 2017, p. 13). We provide an overview of important eye movements. The most prominent oculomotor event is a **fixation**. During fixations, the eye remains (relatively) still to focus the visual axis on an object of interest. Micromovements that occur during a fixation include tremor, drift, and microsaccades. The duration range of fixations is typically reported to lie between 200 and 400 ms (Holmqvist and Andersson, 2017, p. 15) or 150 and 600 ms (Duchowski, 2007, p. 44). A common simplification is to assume that, for the time of a fixation, the human pays attention to the fixated object. This assumption is known as the *eye-mind hypothesis* (Just and Carpenter, 1980) and a sufficient approximation for many applications of eye tracking. However, it neglects that, besides attending to the object hit by the visual axis (overt attention), a person can attend to another object using their peripheral vision (covert attention) (Findlay and Gilchrist, 2003, p. 3). An important role of covert attention and peripheral vision is to collect relevant information for planning re-directions of the visual axis (Findlay and Gilchrist, 2003, p. 9). These re-directing eye movements are called **saccades**, i.e., rapid eye movements between two fixations that take around 30 to 80 ms. With up to 500 degrees per second, saccades are the fastest body movement (Holmqvist and Andersson, 2017, pp. 14-15). **Smooth pursuit** events or pursuit movements are slower, with a velocity of around 10 to 30 degrees per second. They occur when a person tracks a moving target with their eyes. Unlike saccades, the visual scene can be processed during a pursuit movement. Further eye movements include torsion, which is a rotation around the direction of gaze, and vergence,

which is the movement of the eyes in opposite directions to adjust the focus to a stimulus at a different depth (Holmqvist and Andersson, 2017, pp. 14-15). Nystagmus is, in principle, a series of smooth pursuit movements followed by saccades in the opposite direction (Holmqvist and Andersson, 2017, p. 216). The closing of the eyelids is commonly referred to as a blink event (Holmqvist and Andersson, 2017, p. 16). Blinks are typically removed from the raw gaze signal because pupil tracking might be erroneous before closing and after opening the eye, i.e., when the eyelid partially occludes the pupil.

2.2 Eye Tracking Technology

Human gaze and eye movements can be captured and analyzed using modern eye tracking technology. In this section, we provide an overview of common types of eye tracking hardware and corresponding methods for gaze estimation, algorithms for detecting eye movements, and methods for visualizing and analyzing these eye movements.

2.2.1 Eye Tracking Devices and Gaze Estimation

Modern hardware and software enable precise and non-intrusive tracking of the gaze direction. The most common device types include video-based remote and head-mounted eye trackers, also known as video-oculography systems (see figure 2.2). In such systems, the video cameras are placed in the environment, e.g., on a table or at the edge of a screen, or on the head of a participant, respectively. Other types exist but typically suffer from specific drawbacks making them unsuitable for gaze-based interaction, which is the focus of this thesis. Examples include eye-mounted systems and electro-oculography. Eye-mounted systems are invasive and suffer from low comfort and hygiene problems (Holmqvist and Andersson, 2017, p. 95). For instance, scleral coil systems are based on contact lenses that have to be attached to the eyeball. A coil is mounted on the lens and influences an electromagnetic field upon eye movements which can be measured and used to estimate the gaze direction (Duchowski, 2007, p. 51). Electro-oculography is based on differences in the electrical potential of the skin around the human eye. Electrodes placed around the eye can be used to measure these differences, which indicate horizontal and vertical eye movements as well as blinking but cannot be used to estimate the gaze direction (Holmqvist and Andersson, 2017, p. 70). Nevertheless, wearable electro-oculography systems could have applications in context-aware mobile interaction (Bulling et al., 2008). In this thesis, we focus on video-based eye tracking systems in the context of human-machine interaction. Video-based eye tracking systems include two essential processing steps: eye detection and gaze estimation (Hansen and Ji, 2010). Eye detection refers to the localization of the eye in the video stream of one or multiple video cameras. The gaze estimation step determines the gaze direction or point of regard (PoR) based on the outcome of this process. Typically, a calibration procedure is required to generate a function that maps the outcome of the eye detection step to a direction or point in a target space, e.g., a two-dimensional coordinate system of a computer screen. We introduce concepts and techniques for eye detection and gaze estimation, which are commonly used for remote and head-mounted eye trackers. We then describe these device types and highlight important differences.

2.2.1.1 Eye Detection

Video-based eye trackers capture one or multiple eyes of a human. Their presence and position must be detected to enable the subsequent gaze estimation step. However, this task remains challenging due to several aspects that can greatly alter the appearance of the eye region. Occlusions of the eye by the eyelids, the condition of the eye (i.e., whether it is open or closed), ethnicity, viewing angle, light conditions, and the variability in size, reflectivity, or head pose are influential factors and lead to a high variance problem domain. A taxonomy for eye detection techniques by Hansen and Ji (2010) includes shape-based, appearance-based, and hybrid methods, i.e., methods that combine several approaches, that address this problem.

Shape-based Methods

Shape-based methods leverage the shape characteristics of the face and eye region to locate and track the position of the eye or the gaze direction in a sequence of eye images. For that, features or contours extracted from the images are matched to an eye model. The basic building blocks of shape-based methods include a geometric eye model and a similarity measure for matching the structures of the image to that model. Models can be classified into simple fixed-shape models, more complex geometric models such as deformable-template models, and feature-based models that rely on local image features. A common simple model is the elliptical shape model that aims at matching a set of image features to an elliptic shape that may represent the elliptic boundary of the iris (i.e., the limbus) or the pupil. In fact, these shapes are circular, but depending on the camera position, they appear elliptic. Usually, such approaches are efficient and robust under various viewing angles but face problems with variations in eye features. More complex models, like the deformable-template model (Yuille et al., 1992), enable a more detailed representation of the state of the eye. It models the human eye with two parabolas for the eyelids and a circle for the limbus. In general, such models are accurate and generic. However, they suffer from several limitations, including high computational costs, a need for high-contrast images, problems with model initializations far from the eye position and re-initializations after large head movements (for remote eye tracking only), and problems in handling eye occlusions. Feature-based models are based on local image features instead of holistic image features describing the whole eye or head. These are typically more robust against variations of ambient illumination and camera viewpoints. Common local features include the limbus, the pupil, and corneal reflections. The most prominent approaches in current eye tracking hardware include pupil detection based on local feature-based or holistic shape models and active infrared (IR) illumination. Commonly, near IR light emitters with a wavelength of around 780 to 880 nm are used, which can be captured by many default camera modules and are mostly invisible to humans. The light source can be closely aligned with the optical axis of the camera, which results in a bright appearance of the pupil. In this case, the camera can capture the IR light that enters the eye through the pupil, is reflected from the retina, and leaves the eye through the pupil again. The pupil appears brighter than the rest of the eye and head because it reflects more IR light than the rest of the eye and head. The most common case is that the light source is not aligned with the optical axis. The head and the eye reflect more IR light than the pupil because the incident IR light reflected from the retina cannot leave the pupil toward the camera: the pupil appears darker than the rest.

Appearance-based Methods

Appearance-based eye detection and tracking methods, also known as image template methods or holistic methods, rely on the visual appearance of the eye. The appearance is captured using, e.g., color distributions or filter responses for the camera image showing the eye region. Approaches can be based on image templates that take into account spatial and intensity information of the image, i.e., camera images are matched against image-based templates for eye detection, or on statistical models that interpret the intensity distribution independent from spatial information. Modern approaches rely on machine learning methods like convolutional neural networks to detect and track the eyes (Zhang et al., 2015b). Appearance-based methods are considered to work well in realistic settings. However, large amounts of template or training data must be collected to be robust to illumination and facial orientation variations.

2.2.1.2 Gaze Estimation

The gaze estimation step determines the gaze direction or point of regard. A gaze direction is typically reported as a three-dimensional gaze ray in a reference coordinate system anchored in the real world. The point of regard can refer to a point in 2D space, e.g., from a computer screen, or in 3D space, such as the mentioned world coordinate space. In principle, gaze estimation methods model the relation between the image data of an eye tracker (by using the outcome of the eye detection step) and the gaze direction or point of regard. One of the challenges is that users move their heads when interacting with eye tracking devices. For that reason, many remote eye tracking systems use chin rests, bite bars, or similar structures to fix the user's head with the goal of preventing such distorting head movements. Modern approaches for stationary eye tracking rely on additional head tracking, either in a direct way or implicitly, e.g., as part of a mapping function. Another option is to mount the eye tracking hardware on the user's head: cameras for eye detection move with the head, and no more compensation is required. Head-mounted devices are considered to be more invasive, but modern tracking devices based on glasses-like mounts enable comfortable mobile eye tracking (Kassner et al., 2014). One drawback is that gaze estimates are always relative to the user's head pose unless additional external head tracking or surface detection is employed. Typically, the point of regard is determined as a 2D point in the coordinate system of an egocentric (i.e., front-facing) camera that is also mounted on the same glasses-like device (Holmqvist and Andersson, 2017, p. 95). Independent of the device type, all gaze estimation methods require prior calibration. Important parameters that need to be determined are intrinsic camera parameters (camera calibration), the geometric setup of the tracking device, including its cameras and light sources (geometric calibration), individual user characteristics (personal calibration), and parameters of the gaze mapping function (gaze mapping calibration). In most cases, the related mapping functions are feature-based, i.e., based on local image features like shape-based eye detection methods. These are either model-based geometric approaches or interpolation-based/regression-based approaches. Despite significant advances in eye tracking technology, gaze estimation is erroneous. More details on gaze estimation errors, their sources, related metrics, and methods for compensation are given in section 3.1.1.1.

Regression-based

Regression- or interpolation-based methods rely on a function that maps image features to 2D gaze coordinates or 3D gaze directions. This mapping function can have a parametric form like a polynomial, or a non-parametric form like neural networks. In this case, the calibration routine has the goal of finding the best parameters (or weights in the case of neural networks) that approximate the gaze position or direction.

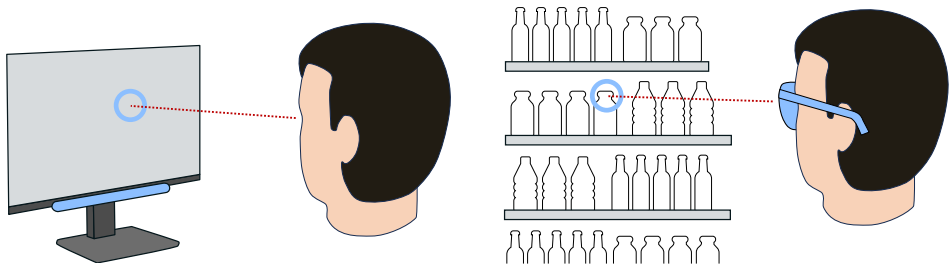
Model-based

Model-based or geometric methods rely on a computational model that directly infers the gaze direction from the eye features extracted in the eye detection step. These approaches model the geometry of the eye tracker and human physiology, which allows them to calculate a 3D gaze direction. The point of regard can be obtained by intersecting this vector with the closest object in the scene, such as a computer screen.

2.2.1.3 Remote and Head-mounted Video-based Eye Trackers

The two most prominent types of video-based eye trackers are remote and head-mounted devices (Duchowski, 2007). **Remote eye tracking devices** as illustrated in figure 2.2a track the eyes of a user from a distance (cf. Holmqvist and Andersson (2017, pp. 100-103)). The devices are placed in the environment, typically close to a stimulus, such as a computer screen. The gaze directions or points of regard are reported in a fixed, world-anchored coordinate system, e.g., as points on the computer screen. Remote eye trackers do not require the user to be instrumented and allow head movements to a certain degree. Device manufacturers typically report a headbox for that, i.e., a 3D volume relative to the device in which the quality of eye tracking does not deteriorate. These advantages are traded off against lower data quality compared to more restricted stationary setups like tower-mounted systems for which the head of a participant is usually fixed at the tower mount using a forehead and chin rest to prevent distorting head movements. Still, applications of remote eye trackers are typically restricted to stationary settings like observing users when interacting with a computer. The range of commercially available remote eye trackers goes from affordable consumer devices like the Tobii Eye Tracker 5[1] for gaming (around 300 €) to professional devices like the Tobii Pro Fusion[2], which costs several thousands of Euros. In addition, several approaches based on usual webcams, i.e., cameras capturing the visible light spectrum, or depth cameras, are available, for instance, WebGazer.js[3] enables webcam-based eye tracking in a browser. **Head-mounted eye trackers** as illustrated in figure 2.2b are wearable devices, i.e., all active parts like cameras and light sources are mounted on a head-worn frame (cf. Holmqvist and Andersson (2017, pp. 95-99)). Typically, the gaze estimation process reports points of regard as 2D points in the local coordinate system of an egocentric, front-facing camera that captures parts of the human field of view. This coordinate system is relative to the head-mounted device. Head-mounted devices allow users to move around freely, enabling many applications in mobile interaction settings like observing gaze

[1] `https://gaming.tobii.com/product/eye-tracker-5/` (accessed on 12 Dec 2024)
[2] `https://tobii.com/products/eye-trackers/screen-based/tobii-pro-fusion` (accessed on 12 Dec 2024)
[3] `https://webgazer.cs.brown.edu/` (accessed on 12 Dec 2024)



(a) Remote Eye Tracker **(b)** Head-mounted Eye Tracker

Figure 2.2: Common eye tracking devices include (a) remote eye trackers and (b) head-mounted eye trackers. The blue circle indicates the point of regard, and the dotted red line is the visual axis.

behavior in a supermarket. Transferring points to another coordinate system, e.g., the one of a computer screen or a shelf in a supermarket, requires an additional mapping step, like head-pose tracking or marker tracking (see chapter 4). Head-mounted devices are considered to be more invasive, but modern devices like Pupil Invisible[4] are mostly indistinguishable from usual glasses and offer a higher wearing comfort than, e.g., meanwhile obsolete helmet-based systems. Another aspect in favor of head-mounted eye trackers is that recent virtual and augmented reality headsets incorporate eye tracking technology such as Microsoft's HoloLens 2[5].

2.2.2 Eye Tracking Data Analysis

Data visualization and analysis techniques are important for almost any use case of eye tracking. For instance, diagnostic applications can be driven by quantitative measures, like completion times or accuracy rates concerning visual tasks in user experiments, or qualitative assessments based on visualizations of the raw gaze data that help in understanding human visual behavior (Blascheck et al., 2017). One prominent example is the experiment by Yarbus (1967), who investigated the influence of different tasks on the viewing strategy of participants by visually inspecting the eye tracking data as an overlay to the image-based stimulus. Active and passive gaze-based interfaces also use techniques for data analysis ranging from eye movement detection to machine learning-based analysis techniques taking features extracted from the raw gaze signal as input. In this section, we introduce the term raw gaze, describe common eye movement detection algorithms and the terms area of interest and scanpath, and provide an overview of visualization techniques.

2.2.2.1 Raw Gaze

The term *raw gaze* refers to the real-time signal or the recording of this signal of an eye tracking device. This signal originates from the gaze estimation module of an eye tracker, which delivers a series of timestamped 3D gaze directions, 2D points of regard, or both (see section 2.2.1.2). Most applications use the 2D points of regard that can be represented as a triple (t, x, y) with t

[4] `https://pupil-labs.com/products/invisible/` (accessed on 12 Dec 2024)
[5] `https://www.microsoft.com/en-us/hololens/hardware` (accessed on 12 Dec 2024)

indicating the capturing time of this sample and x, y defining the 2D point in a certain coordinate system, e.g., a corresponding video frame of a head-mounted eye tracker or a coordinate of a computer screen at which a remote eye tracker is mounted. Simple visualizations of the raw gaze signal include line graphs plotting x and y against time on the x-axis and plotting x and y coordinates as an overlay on the visual stimulus in question. The timestamps reveal an eye tracker's temporal resolution or sampling rate, typically reported in Hertz. It is computed as $1/t$ Hz, typically averaged over the whole recording or a certain window size because the time deltas might slightly differ. The actual sampling rate of an eye tracking setup might also differ from the sampling rate reported by the manufacturer of the eye tracking device. For video-based devices, sampling rates range from 30 Hz to 2000 Hz (Holmqvist and Andersson, 2017, p. 91). It defines the type of eye movements that can be detected: the Nyquist-Shannon sampling theorem implies that to capture events occurring at a maximum frequency of B, a sampling rate greater than $2B$ is sufficient (Shannon, 1949). In practical terms, this means that a 30 Hz eye tracker should be able to capture all fixations, but might miss quick saccades with a length of around 30 ms. Further characteristics that determine the quality of the eye tracking signal include the offset of the gaze estimates to the actual points of regard, also known as spatial accuracy, the dispersion of the signal around the actual points of gaze, also known as spatial precision, and the ratio of invalid samples, also known as data loss (see section 3.1.1.1).

2.2.2.2 Eye Movement Detection

Several movements of the human eye can be detected in the raw gaze signal of eye tracking devices (see section 2.1). These events, also called oculomotor events, include fixations, saccades, smooth pursuits, and blinks. Detecting and characterizing oculomotor events are essential steps in eye tracking data analysis. For instance, the eye-mind hypothesis by Just and Carpenter (1980) suggests that fixation times correspond to the time a participant pays attention to the viewed point of regard, which requires accurate fixation detection methods. However, noise from imperfect tracking and natural micromovements of the eye, like tremor or microsaccades, can distort the detection of eye movement events, which makes prior signal pre-processing mandatory in many cases. Typical compensation approaches, particularly for gaze-based interaction, include real-time signal filters (Špakov, 2012). A more detailed overview of compensation methods is provided in section 3.1.1.1. In the following, we describe prominent methods for detecting fixations. Fixation detection algorithms label each sample of a raw gaze signal as belonging to a fixation event or not. In many cases, the resulting gaps are considered to be saccadic eye movements. Gaps due to loss of pupil tracking are usually not considered in this classification or may be interpreted as blink events: the eyelids occlude the pupil, resulting in a temporary loss of tracking. Salvucci and Goldberg (2000) introduced a taxonomy that classifies fixation detection algorithms based on spatial and temporal characteristics. An algorithm can be velocity-based, dispersion-based, or area-based in terms of spatial characteristics. Velocity-based algorithms take advantage of saccades' high velocities compared to fixations during which the eyes remain relatively still. One example is the *velocity-threshold fixation identification* method (I-VT). I-VT computes velocities between successive samples. If the velocity for two samples (s_i, s_{i+1}) stays below a certain threshold, s_i is classified as a fixation sample. Otherwise, it is marked

as a saccade sample. Fixation events are derived by grouping fixation samples. It is reported as quadruple (t, x, y, d): t is the starting time of the event, i.e., the timestamp of the first sample of the group, x, y is calculated as the centroid of all samples, and the duration d is the difference between the last and the first timestamp of a group. Dispersion-based algorithms rely on low dispersion values of consecutive gaze samples that belong to one fixation in contrast to high dispersion values during a saccade. A prominent algorithm is the *dispersion-threshold identification* method (I-DT). It starts by computing a dispersion measure for a window of a pre-defined minimum size. If the measure is greater than a certain threshold, no fixation is found, and the window is moved one sample ahead. A fixation is identified if the measure is lower or equal to the threshold. Then the window is extended by one sample at a time until the dispersion threshold is exceeded again. The fixation event is represented by the previous window, i.e., when the dispersion threshold was not yet exceeded. Area-based algorithms define fixations based on fixed target regions, i.e., consecutive samples that lie within a given target region are counted as a single fixation event; samples outside these regions are considered saccade samples. One example is the *area-of-interest identification* method (I-AOI), which is based on rectangular regions of interest that depict relevant parts of a considered visual scene. However, velocity-based and dispersion-based algorithms are the more common options for fixation detection. For instance, area-based methods are more suitable for high-level analysis when applied to a sequence of previously identified fixation events rather than the raw gaze signal (Salvucci and Goldberg, 2000). Temporal characteristics include the criteria duration sensitive and locally adaptive. An algorithm is duration sensitive if duration information is used, like the minimum duration threshold for I-DT. An algorithm is locally adaptive if samples that are close together, temporally, can influence the detection. For instance, I-DT computes dispersion for a window of variable size. It is locally adaptive. In contrast, I-VT takes into account pairwise velocity information only.

2.2.2.3 Areas of Interest (AOIs)

Eye tracking studies often consider visual attention to specific areas of interest (AOIs) to analyze and understand how people process visual information. AOIs are specific regions in a scene or interface that are defined by a researcher to collect related gaze data (Holmqvist and Andersson, 2017, pp. 254-301). Visual attention refers to the time a person pays attention to these regions. By measuring visual attention to and transitions between AOIs during a study, researchers can gain insights into which elements of a scene are relevant to an activity and how interventions of an experiment influence the participant's eye movement behavior. This is usually done based on fixation events as they are assumed to approximate a person's allocation of cognitive resources through the time they spend processing a visual scene (Just and Carpenter, 1980).

AOI Events

Basic events that are commonly extracted include AOI hits, dwells, and transitions. They correspond to fixations and saccades but are defined over semantically meaningful areas in a scene, i.e., over AOIs. They can be calculated from the raw gaze signal or based on previously detected fixation events. For instance, an AOI hit is reported if a sequence of gaze samples or a

fixation lies within an AOI. It is a simple count-based measure. Extracting AOI hits based on the raw gaze signal shares similarities with area-based fixation detection. It may also take into account a minimum time threshold. The AOI dwell is similar to a fixation event: it aggregates a sequence of consecutive gaze samples that hit the same AOI. AOI transitions refer to gaze shifts from one AOI to another. They can be seen as saccades between AOI-based fixations (or dwells). Many other AOI events exist but are typically based on these basic events.

Event Extraction

Extracting AOI events depends on their definition and type. Questions that arise include, what areas are considered as important, how are they formally defined, and how can gaze samples or fixations be mapped to them? In remote eye tracking, AOIs typically refer to specific regions in the targeted visual stimulus and are defined using, e.g., rectangular bounding boxes in the simplest case. AOIs can be static, for stimuli like static images or text, or dynamic, e.g., for video-based stimuli in which the relevant area moves over time. The complexity increases for head-mounted eye trackers. In mobile settings, AOIs are usually defined based on relevant objects or areas in the scene as captured by the front-facing camera. In remote eye tracking with static stimuli, an AOI can be defined once and reused for every participant. Dynamic AOIs in video-based stimuli can be annotated using keyframe-based annotation techniques, i.e., AOIs are marked via bounding boxes for keyframes, and interpolation is used to annotate intermediate frames (Kurzhals et al., 2014b). However, these efficient fixation-to-AOI mapping techniques from remote eye tracking do not scale for mobile eye tracking applications. Accurately annotating mobile eye tracking data remains a challenging and time-consuming task because scene videos taken with a head-mounted eye tracking device are unique for every participant. In mobile eye tracking practice, one or more annotators decide per fixation whether an AOI was hit or not (Uppal et al., 2022; Kurzhals et al., 2014a).

2.2.2.4 Scanpaths

The term scanpath can be defined as "the route of oculomotor events through space within a certain timespan" (Holmqvist and Andersson, 2017, p. 327). This definition refers to the sequence of actual eye movements of a person, i.e., eye trackers can only approximate the scanpath of a person through their raw gaze signal. Another common definition describes the scanpath as "a sequence of alternating fixations and saccades" (Blascheck et al., 2017, p. 262). In many cases, scanpaths are plotted to inspect an eye tracking recording. For instance, to check the quality of individual scanpaths or to identify differences in viewing patterns of different user groups or for different conditions of an eye tracking experiment (Holmqvist and Andersson, 2017, p. 330). A typical visual scanpath representation shows points of gaze or fixation points as dots connected by edges in the stimulus space.

2.2.3 Applications of Eye Tracking

Duchowski (2007, p. 247) classifies eye tracking applications into two broad categories: diagnostic and interactive. Diagnostic applications refer to offline data analysis in the context of user studies,

i.e., eye movements and gaze directions are recorded under specific experimental conditions and analyzed post-hoc. Each type of application can occur in a range of application domains. Common domains include neuroscience & psychology, industrial engineering & human factors, marketing & advertisement, and computer science (Duchowski, 2007, pp. 249-339). All domains, besides computer science, focus on diagnostic applications. The goal is to understand general perceptual and cognitive processes and people's perceptions of user interfaces, such as in aviation and media. Gaze-based interaction is subsumed under computer science. However, findings from diagnostic studies can also be used to build interactive eye tracking applications, e.g., passive ones that react to a user's cognitive state. More applications of eye tracking in user interfaces are described in the respective related work sections for active and passive gaze-based interaction in sections 3.1 and 3.2.

2.3 Ethics and Privacy

In many cases, eye trackers serve as a tool for conducting experimental research with human participants. Common ethical principles from the target domain should be applied. For instance, in psychology, six aspects are typically named: informed consent, protection from harm, deception, freedom from coercion, debriefing, and confidentiality & anonymity. These are detailed in the *Ethics Code*[6] published by the American Psychological Association (APA), but are also reflected in different contexts, e.g., in a guidance note by the European Commission on ethics in social science and humanities (Rauhala and Kalokairinou, 2021). *Informed consent* means that participants are informed about the purpose of the study and the associated risks before performing any tasks, and their explicit consent is obtained. Participation should be voluntary, meaning that the participant can withdraw at any time without giving a reason. *Protection from harm* means that physical and psychological harm caused by the study should be minimized. In some cases, harm must be weighed against an expected benefit. For instance, eye-mounted eye trackers, like the scleral coil system, are invasive and uncomfortable. Still, they have been used to capture human eye movements with high spatial accuracy, precision, and temporal resolution. Also, explicit eye movement can lead to eye strain and fatigue. This should be minimized, e.g., by reducing the number of trials per participant, but it can usually not be avoided completely. *Deception* should be avoided, i.e., information for participants should not be misleading unless the expected benefit outweighs the cost. *Coercion* must be avoided. It means that participants are pressured, e.g., by threats, to participate in or stay in a study, or perform another activity against their will. *Debriefing* refers to a phase after the study is over in which participants are allowed to ask questions and potential deceptions are revealed. The last principle, *confidentiality & anonymity*, concerns data privacy and security. For instance, participants' identities should not be revealed and be kept secure. In the case of eye tracking, this may affect the way how raw gaze data and video data from head-mounted eye trackers are handled.

Privacy and ethical considerations are also important when developing eye tracking technology with the goal of deploying it, e.g., as part of a product. Eye trackers record a person's eye movements and gaze directions, which contain extensive information about their perceptual and cognitive processes (Gressel et al., 2023; Bulling and Roggen, 2011). In addition, most head-mounted eye trackers capture a gaze-synchronized video of the environment. This motivates the development of gaze-based interfaces enabling efficient and situation-aware interaction based on that information. However, access to such sensitive information raises several ethical, legal, and privacy issues that need to be addressed (cf. Gressel et al. (2023)), particularly because eye trackers are now integrated into more and more end-user devices like head-mounted extended reality headsets and gaming laptops[7]. Recently, the topic of privacy-aware eye tracking technology gained more interest in the research community (see, e.g., Steil (2019)). Also, since 2021, the International Workshop on Privacy and Ethics in Eye Tracking (PrEThics)[8] provides a discussion platform for related topics.

[6] `https://www.apa.org/ethics/code` (accessed on 12 Dec 2024)
[7] `https://www.tobii.com/products/integration` (accessed on 12 Dec 2024)
[8] `https://prethics.perceptualui.org/` (accessed on 12 Dec 2024)

Chapter 3

Related Work

This thesis presents new approaches and methods with the goal of addressing key challenges of gaze-based interaction concerning active gaze-based interaction, passive gaze-based interaction, and the design and development of gaze-based interaction systems. It aims to contribute by developing two methods for active gaze-based interaction and three approaches for passively interpreting the human gaze signal. In the following, I present related work on using gaze as an active input modality in section 3.1 and as a passive input modality in section 3.2.

3.1 Human Gaze as an Active Input Modality

This thesis presents novel techniques for active gaze-based interfaces, i.e., interfaces that use human gaze in an active input mode, in part II. In the following, we provide an overview of interfaces that use the absolute gaze position as input and background information on the gaze estimation error that can severely hamper the usability and performance of such interfaces. We also present calibration-free interaction techniques that do not rely on absolute gaze estimates and are, therefore, mostly robust to gaze estimation errors.

3.1.1 Absolute Gaze Position as Input

A straightforward way to use eye tracking in multimodal interfaces is to use the absolute gaze position as a spatial pointer. This enables, for instance, gaze-based object selection and manipulation based on the principle "what you look at is what you get" (Jacob, 1990). This has also led to eye tracking becoming an essential tool for people with motor disabilities: absolute gaze position tracking was used to enable communication via gaze-based text entry and environmental control (Bates et al., 2007). A key problem is that gaze estimation errors directly affect interaction techniques that rely on absolute gaze positions.

3.1.1.1 Gaze Estimation Error

Gaze estimation error refers to the discrepancy between the intended gaze direction and the estimated direction of gaze in eye tracking systems. Various factors, including hardware limi-

tations, software algorithms, and participant-related factors, can cause this error. It can have implications for the validity and reliability of eye tracking data in diagnostic studies and severely hamper the usability of gaze-based interaction systems. The importance of errors in gaze estimation has long been recognized. Still, previous work has focused mainly on error source reduction or post-hoc error compensation for diagnostic eye tracking studies in stationary settings.

Sources of Error

Sources of error are typically found in humans, i.e., participants and operators, in the experimental setup, including characteristics of the tasks and the recording environment, and in the tracking device, including its hardware and software (Holmqvist et al., 2012). Human factors include the presence of visual aids, characteristics of the eye's physiology, like the direction of eyelashes, eye color, pupil diameter, and the experience of operators (Nyström et al., 2013; Drewes et al., 2012). Typical factors concerning the experimental setup include illumination changes and infrared distortion (Holmqvist et al., 2012; Kassner et al., 2014), the arrangement of visual stimuli (Nyström et al., 2013), and factors that may deteriorate the calibration like long recording times and tasks that require participants to move (Nyström et al., 2013; Holmqvist et al., 2012). The latter two sources, in particular, also apply to head-mounted devices, as they can lead to displacements of the tracking device (John et al., 2012). An inherent issue with monocular head-mounted eye trackers is the parallax error: the gaze estimation error increases as the participant moves away from the calibration position (Mardanbegi and Hansen, 2012). Also, the gaze signal needs to be mapped to screen coordinates for on-screen interaction with head-mounted eye trackers, which is another error source. However, it remains largely unexplored what sources of error occur in head-mounted eye trackers and how they affect gaze-based interaction.

Error Metrics

Several metrics have been used to quantify the gaze estimation error. The most prominent ones include spatial accuracy and spatial precision, which are typically reported in terms of degrees of visual angle (Akkil et al., 2014; Blignaut and Beelders, 2012; Holmqvist et al., 2012; Kassner et al., 2014; Nyström et al., 2013; Hornof and Halverson, 2002; John et al., 2012). We follow the definitions by Holmqvist and Andersson (2017, pp. 165-189). **Spatial accuracy** A reflects the offset between estimated and true gaze estimates. It is a measure of central tendency and is calculated as the mean angular offset θ_i between n estimated and true fixation or gaze points.

$$A = \frac{1}{n} \cdot \sum_{i=1}^{n} \theta_i \tag{3.1}$$

Spatial precision P reflects the distortions in the gaze signal. It is a measure of statistical dispersion and is calculated as the Root Mean Square (RMS) of the angular distance θ_i between successive fixation or gaze points. However, other measures of statistical dispersion, like standard deviation, can also be used.

$$P = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} \theta_i^2} \tag{3.2}$$

Instead of angular differences, the length or distance between two points can be more relevant regarding gaze-based interaction. In this case, θ would be the Euclidean distance between two points using a linear measure like centimeters. This can be a distance between two 2-dimensional points for screen-based interfaces or two 3-dimensional points for 3D interfaces such as in virtual reality. Another common metric is **data loss**, also known as robustness (Nyström et al., 2013; Akkil et al., 2014; Blignaut and Beelders, 2012). It is typically measured as the ratio between the number of valid estimated gaze points n_{valid} and the theoretical maximum amount of samples that could have been captured n_{max}. Data loss or low robustness is usually caused by loss of pupil tracking.

$$n_{loss} = \frac{n_{valid}}{n_{max}} \tag{3.3}$$

Low-pass signal filters can improve gaze signals with low spatial precision, i.e., smoothing filters that reduce noise in the signal (Špakov, 2012). Methods to cope with low spatial accuracy include, for instance, gaze-to-object snapping algorithms (Špakov, 2011). Data loss can be minimized by optimizing boundary conditions like illumination and the configuration of the eye tracking setup Holmqvist and Andersson (2017, p. 167) and by addressing software issues like low robustness of pupil tracking algorithms for highly off-axis perspectives (Świrski et al., 2012). Other quality measures concern temporal aspects of the gaze signal. Latency or temporal accuracy indicates the difference between the time of the eye movement and the time reported by the eye tracker; the variance in temporal accuracy is called temporal precision (Holmqvist et al., 2012).

Error Compensation

Many approaches have been presented for compensating the gaze estimation error. Common techniques for coping with low spatial accuracy and precision include gaze-to-object mapping algorithms and gaze signal filters. Gaze-to-object mapping methods draw gaze points towards or directly map them on nearby objects, compensating for low accuracy and precision (Špakov, 2011; Zhai et al., 1999; Špakov and Gizatdinova, 2014). Gaze signal filters are filters from the signal processing domain for noise reduction, mainly addressing low precision (Špakov, 2012). Other approaches concern calibration and recalibration methods. Hornof and Halverson (2002) proposed a method for monitoring the spatial accuracy of the gaze signal during remote eye tracking studies. They introduce the notion of implicit required fixation locations (RFL), i.e., locations that participants must look at within a certain time window to accomplish a task. The difference between the observed fixation position and the RFL is used to track the spatial accuracy of the signal and to trigger a recalibration of the eye tracking system. A similar method is to correct a systematic error based on the distances between the centers of fixation clusters and the respectively nearest objects (Zhang and Hornof, 2011, 2014). Lander et al. (2016) investigated how the time required for a recalibration can be reduced. Cerrolaza et al. (2012) showed that head movements perpendicular to the screen cause errors in gaze estimation. They proposed a new calibration procedure and gaze estimation function to incorporate the eye-to-screen distance that compensates for this error. Blignaut and Wium (2013) compared different calibration methods and found that the arrangement and number of calibration targets significantly affect the quality of gaze estimation. Further, they developed an addition to calibration procedures using a gaze-based selection task and regression to improve the quality of the gaze estimation (Blignaut

et al., 2014; Blignaut and Wium, 2013). Feit et al. (2017) investigated the gaze estimation error for remote tracking devices at a constant interaction distance. They modeled the error for different positions on a display based on the mean deviation of gaze points to their true on-screen positions and the corresponding standard deviation (SD). They showed that the SD could be used to optimize signal filters for the gaze signal and the mean offset to inform gaze-based interface design. We include their findings in designing our proposed gaze-based interface that is aware of the error inherent in mobile eye trackers. Many other methods for addressing the gaze-estimation error are presented as part of an interaction technique.

3.1.1.2 Interaction Techniques

In his seminal work, Jacob (1990) explored how the human gaze can be used as an input to computer systems. Based on the principle "what you look at is what you get," he developed the first techniques for active gaze-based interaction, including approaches to object selection and manipulation, gaze-controlled information visualization, and text scrolling. A key finding was that the gaze signal alone is unsuitable as input because it cannot be differentiated whether the user wants to issue an action at the current point of gaze. This is known as Midas' Touch problem. Proposed solutions include using a dwell time, i.e., an action is triggered when the user fixates an element for longer than a predefined period of time, or using an additional action trigger, such as pressing a button. All interaction techniques that have been presented later relate in some way to these basic principles. In the following, we present an overview of active gaze-based interaction techniques ranging from unimodal approaches using dwell times to multimodal systems that combine gaze with additional selection triggers like a key or button press.

Unimodal Interaction

Unimodal active gaze-based interfaces are user interfaces that rely solely on the user's gaze as an input modality. This technology allows users to interact with digital devices, such as computers or smartphones, using only their eyes to control the interface. Unimodal gaze-based interfaces have the potential to offer an accessible and hands-free way for people with disabilities to interact with technology (Bates et al., 2007). Typically, unimodal gaze-based interfaces are restricted to object selection with a dwell time as a selection trigger. More complex interactions are possible but rely on mode-switching, which can be confusing. Nevertheless, interfaces that rely only on gaze as an input modality are important for accessible user interfaces, especially when users can only move their eyes. Miniotas et al. (2004) investigated the impact of virtually increasing the target size on the error rate and the time required for making a dwell-based selection. They found that larger target sizes reduce the error rate and the selection time. Further, they introduced a gaze-to-object mapping algorithm that allows the gaze signal to leave the target area without immediately interrupting the dwell counter. It decreases the error rate but increases the required time for selection. Istance et al. (2008) developed Snap Clutch, which enables switching between four gaze-based interaction modes by glancing to one of four sides outside a screen. All modes rely on dwell times to enable gaze-only interaction for handicapped people. They include a click mode, a cursor parking mode that allows users to trigger a certain action at the cursor position, a drag-and-drop mode for object manipulation, and a gaze control off mode. Zhang et al. (2008)

investigated ways to stabilize the eye cursor when the precision of the gaze signal is low. They presented three gaze-to-object mapping methods, of which two improve the effectiveness of dwell-based object selection: Speed Reduction and Force Field, which both drag the gaze point toward a virtual force field, i.e., the previous gaze point or the center of the selection target, respectively. Gizatdinova et al. (2018) implemented a text entry system based on gaze or head directions and a dwell for character selection. In particular, they investigated the impact of the key size on the speed and error rate of the typing process. They found that the error rate drastically increased when the key size was lower than 0.8 degrees of visual angle. Isomoto et al. (2018) developed a method based on Fitts' law for reducing the dwell time required for successful object selections. Minakata et al. (2019) compared various dwell-based object selection methods in a virtual reality setting with a Fitts' law experiment. They found that gaze-based selection was slower than a mouse, foot-mouse, and head-based selection. Choi et al. (2020) introduced the bubble gaze cursor, a circular gaze-based cursor that adapts its size such that it points at a single target only. They found that using the bubble gaze cursor with a dwell as a selection trigger outperformed a usual gaze-based cursor in performance, usability, and workload. Also, they showed that using an additional magnification lens (bubble gaze lens) leads to further performance gains. Krishna Sharma et al. (2020) implemented a proof-of-concept robot control system based on gaze input and dwells. It enables users with severe speech and motor impairments to execute simple pick-and-place and navigation tasks. Ahn et al. (2021) developed the StickyPie marking menu, a pie menu optimized for unimodal gaze-based input in mixed reality: submenus pop up at the new fixation position of the user when an option was selected to avoid false activations. A submenu is typically shown immediately when crossing the border of the outer circle at that position. However, saccades might also pass the border of that submenu in a single movement. Similar to that, Kim et al. (2022) introduced the Lattice Menu, which prevents overshoots by displaying a lattice of visual anchors to align eye movements with menu positions. Choi et al. (2022) introduced a menu that places items in the Kuiper Belt, i.e., a region in the user's field of view that is not frequently hit by fixations, to reduce false activations. Best and Duchowski (2016) introduced a PIN entry method that relies on a spatial interaction design, similar to the concept of marking menus: they show digits in a circular design that imitates a rotary dial phone. A digit is entered by moving the gaze from the center of the screen to a digit-specific area and back. The average entry time is low, with less than five seconds (M=4.62), but the accuracy is too low (M=71.16%) for safety-related applications. It would be hard to differentiate between valid but erroneous inputs and invalid inputs from a brute-force attack.

Multimodal Interaction

Multimodal gaze-based interfaces are user interfaces that combine gaze-based input with other input modalities, such as touch, voice, or gesture recognition. By integrating multiple input modalities, such interfaces can provide users with more flexible and efficient ways to interact with technology (Oviatt and Cohen, 2015). To overcome Midas' touch problem, the gaze signal is often used as a spatial pointer and combined with an additional input modality as a selection or manipulation trigger (Qvarfordt, 2017). Zhai et al. (1999) introduced the MAGIC pointing method, which positions the mouse cursor on a fixated object or in its proximity. The

user confirms the selection using the mouse. Monden et al. (2005) implemented the SemiAuto method combining gaze-based cursor positioning and mouse-based selection confirmation similar to Zhai et al. (1999) with gaze-to-object mapping, i.e., with a mouse click, the nearest object is selected. Drewes (2010, pp. 79-86) developed the MAGIC touch system, which extends the MAGIC pointing approach by Zhai et al. (1999). They use touch events from a touch-sensitive computer mouse to position the mouse cursor and click events as selection triggers. Stellmach et al. (2011) designed gaze-based interaction techniques to explore large image collections. They found that combining gaze with a touch-enabled and tilt-sensitive mobile phone enables versatile inputs to a multimedia retrieval system, like controlling a fish-eye lens to magnify thumbnails at a certain position or panning and zooming in to navigate through a 2D multimedia item space. They also designed and evaluated several input methods for navigating in 2D map applications (Klamka et al., 2015) and steering in 3D virtual environments based on 2D user interface overlays (Stellmach and Dachselt, 2012a). Zhang et al. (2019) used a similar approach to control remote telepresence robots using a virtual reality headset. In another work, Stellmach and Dachselt (2012b) focused on object selection for distant displays: they extend the MAGIC pointing principle using touch input from a mobile device. For instance, their MAGIC tab technique allows users to select one of the objects close to the current fixation point using sliding gestures on a mobile device. Further combinations of gaze direction, head direction, and touch input via handheld devices have been explored for object selection and manipulation on distant displays (Stellmach and Dachselt, 2013). Zhang et al. (2015a) investigated the effectiveness and efficiency of using gaze in combination with hand gestures for this purpose. Similarly, Chatterjee et al. (2015) introduced a gaze-based object selection method using hand gestures as an activation trigger. Turner et al. (2013) investigated different interaction techniques for cross-device content transfer using head-mounted eye trackers. They considered content transfer between public displays and private handheld devices, a concept they coined Eye Pull, Eye Push. They combined gaze with touch gestures on a mobile device to move contents in a cut-and-paste, drag-and-drop, or summon-and-cast manner. The latter combines gaze and swipe up or down gestures to push contents to a distant display or pull it from there. They also considered gaze in combination with mouse clicks as an activation trigger in another study (Turner et al., 2014). Pfeuffer et al. (2014) presented the Gaze-touch system. It combines gaze input with multitouch gestures to interact with multimedia content on a single device. For instance, they combine gaze with multitouch gestures for indirect rotation, scaling, and translation of an object. Further, they developed the Gaze-shifting method that enables automatic switching between such indirect and direct input modes when the human gaze is combined with absolute manual input: Touch or digital pen input is direct when the user's gaze is on the input location, or indirect otherwise (Pfeuffer et al., 2015). With CursorShift, the authors presented an effective method for using gaze with indirect touch input on handheld tablets (Pfeuffer and Gellersen, 2016). The interaction design is based on the idea that thumbs typically have a limited interaction range when holding a display but can be used for indirect touch input in combination with gaze-based pointing. They followed a similar approach when building the GazeButton system, which accepts indirect touch input via a separate button only (Rivu et al., 2019). Creed et al. (2020) implemented the Sakura tool that enables people with physical impairments to create multimedia content. They investigated the efficiency and usability of their interaction design which is based on gaze and a mechanical switch

as a selection trigger. With Gaze + Hold, Ramirez Gomez et al. (2021) presented a system that enables object selection and manipulation by looking at them while performing a blink gesture. A single blink is used for selection; keeping the eye closed allows a user to choose a target location with the second eye. Kumar et al. (2020) developed the TAGSwipe interface for gaze-based text entry. It combines touch input from a mobile device to trim the real-time gaze signal over a virtual keyboard by setting the start and end times. The trimmed gaze signal is used to infer the intended word, similar to modern touch-based text entry systems. The Hummer interface is a direct extension that uses different kinds of humming to trim the gaze signal or to select special characters (Hedeshy et al., 2021). The same group investigated whether dwell-based gaze input and gaze with speech commands can be used to correct speech-based text input (Sengupta et al., 2020). Feng et al. (2021) introduced the HGaze Typing system, which works similarly to TAGSwipe (Kumar et al., 2020) and Hummer (Hedeshy et al., 2021) but relies on head gestures to start and stop gaze-based word entry phases. Many approaches have also been implemented and tested in virtual reality (VR) or augmented reality (AR). For instance, the combination of gaze and indirect manual input has also been tested in VR settings for object selection and manipulation tasks. Pfeuffer et al. (2017) presented the Gaze + Pinch interaction technique that combines gaze for pointing with one-handed and two-handed pinch gestures for selecting, rotating, scaling, and translating objects in VR. Later, they presented the Look & Turn interaction method (Reiter et al., 2022): gaze selects a menu option, and rotating the wrist changes a continuous parameter like hue, saturation, or value for defining colors. Using the alignment of gaze and hand directions for object selection (Lystbæk et al., 2022b) and text entry (Lystbæk et al., 2022a) has been proposed for interaction in AR. Luro and Sundstedt (2019) investigated the difference in performance and usability between pure controller-based and gaze-based object selection in an aim-and-shoot task in VR. For the gaze condition, a button click of the controller was used for confirmations. Yu et al. (2021) investigated the effectiveness and efficiency of an interaction technique that combines gaze and hand gestures for object rotation, scaling, and translation, similar to Pfeuffer et al. (2017). Rajanna and Hansen (2018) compared different keyboard layouts for text entry in VR using gaze with a dwell or button click on a controller as a selection trigger. He et al. (2022) introduced the TapGazer system for text entry in VR. They combine touch- or glove-based blind typing with gaze-based word selection to resolve ambiguities. Sidenmark et al. (2020b) proposed a gaze-based object selection method for VR with error correction by additional head movements. Their system can automatically switch between a normal gaze mode and a head-based correction mode, i.e., head movements are used for a fine-grained correction of the gaze-based pointer. They employ a similar concept in their radial menu for AR, Radi-Eye: gaze is used for pointing, and a head movement from the center of the radial menu toward the gaze position triggers a selection of the fixated menu item (Sidenmark et al., 2021). The gaze signal has also been used in combination with speech input. Elepfandt and Grund (2012) investigated explicit gaze pointing combined with speech commands for moving objects on a distant screen. They found that participants preferred short commands over longer ones when combined with explicit gaze pointing. Mardenbegi and Qvarfordt (2015) implemented a system for annotating images taken with a head-mounted AR headset. The system allowed users to take an image with the world camera and annotate parts of it by looking at a certain position while verbalizing the annotation. A few multimodal gaze-based interfaces have been presented

in the context of user authentication, which typically boils down to a text entry task. Kumar et al. (2007) introduced "EyePassword," a system for authentication by gaze-based password or PIN entry using a virtual keyboard or number pad via gaze in combination with a dwell or a key press. The system achieves low error rates comparable to interaction with a physical keyboard. Still, the entry time is higher, and the method requires prior eye tracker calibration. Abdrabou et al. (2019) presented three PIN-entry methods that allow users to enter digits using their gaze plus a dwell, a mid-air gesture, or a combination of both, i.e., the gaze is used for pointing and an additional hand gesture for confirmation. Besides the efficiency of the digit entry, they investigated the resistance to shoulder-surfing attacks for each method.

3.1.2 Calibration-free Interaction Techniques

Calibration-free gaze-based interaction methods eliminate the need for users to go through a time-consuming and tedious calibration procedure before the actual interaction can start. Most techniques are, by design, robust against gaze estimation errors. They offer a promising direction for developing user-friendly and efficient interaction techniques and enable spontaneous interaction with pervasive gaze-based interfaces (Bulling, 2016) and public displays (Khamis et al., 2015). Two prominent directions for calibration-free gaze-based interaction are interfaces based on gaze gestures (Drewes and Schmidt, 2007) and smooth pursuit eye movements (Vidal et al., 2013). Other approaches include appearance-based gaze estimation (Zhang et al., 2015b) and methods based on corneal reflections and pupil detection (Lander et al., 2018).

3.1.2.1 Gaze Gestures

Gaze gestures are sequences of consecutive relative eye movements that can be used to trigger certain actions upon recognition (Drewes and Schmidt, 2007), similar to touch-, hand-, or pen-based gestures. Since recognition is based on relative movement patterns, no device calibration is required. Still, some techniques require prior calibration because, for instance, the gesture relates to a fixated object (Heikkilä and Räihä, 2012). We report on gesture-based interfaces demonstrated with calibrated systems if the calibration had not been required for the recognition part. A disadvantage of gesture-based interaction is that gesture patterns can quickly become complex to learn and remember (Drewes, 2010, p. 123) and cause fatigue (Bulling et al., 2008). Several researchers investigated how gaze gestures can be used to build active gaze-based interfaces. Drewes and Schmidt (2007) implemented a recognition method for gaze gestures based on an existing method for mouse-based gesture recognition. They encode a sequence of saccade directions as a string of characters and recognize a gesture via string matching. They consider eight directions (horizontal, vertical, and diagonal). Further, only saccades between short fixations are interpreted based on the assumption that users perform gestures in one piece. Bulling et al. (2008) implemented a similar gesture recognition system for wearable electrooculography goggles and evaluated it in a gaming context. Wobbrock et al. (2008) introduced EyeWrite, a gesture-based text entry system. Characters are entered, one by one, by separate gestures that approximate the character's shape. Istance et al. (2010) investigated the effectiveness and efficiency of users performing gaze gestures of different lengths and using different basic directions in a computer game. They found that gestures based on diagonal eye movements caused more

false recognitions than vertical and horizontal movements. Also, they compared gesture-based input with dwell-based input in gaming and found that gestures caused fewer errors but similar or better completion times than dwells (Hyrskykari et al., 2012). Kangas et al. (2014) presented an interface for gesture-based interaction on mobile phones. They found that augmenting the gesture with haptic feedback reduces the completion times. They recently suggested detecting gaze gestures from gaze patterns when viewing static drawings (Majaranta et al., 2019). The drawings guide the eye movements following the principle that recognition should be chosen over recall (Shneiderman et al., 2016). Bâce et al. (2016) developed a system based on a head-mounted eye tracker and a smartwatch that allows users to attach annotations to viewed objects in a real-world environment. A gaze gesture triggers the object selection. However, the pointing requires a calibrated eye tracking device. Zhang et al. (2017) developed the GazeSpeak system for people with motor impairments. GazeSpeak is a low-cost communication system based on a mobile phone that shall facilitate communication between patients and caregivers more easily. The system enables patients to enter text via gaze gestures recorded by the phone's camera. Gaze gestures have also been used to realize authentication interfaces that are more robust to shoulder-surfing attacks than a keyboard- or touch-based approach in public display settings. De Luca et al. (2009) implemented two methods for gaze-based authentication in stationary settings, EyePIN and EyePassShapes. The EyePIN system enables users to enter 4-digit PINs using pre-defined gaze gestures similar to Drewes and Schmidt (2007) while pressing an activation key (De Luca et al., 2007). EyePassShapes allows users to authenticate by performing a predefined gesture (De Luca et al., 2009).

3.1.2.2 Smooth Pursuits

A special gesture that facilitates calibration-free gaze input is based on smooth pursuit eye movements, i.e., our eye movements when following a moving object. Vidal et al. (2013) have been the first to present a user interface based on smooth pursuit eye movements. The interface triggers an action when eye movements correlate with an element's trajectory in a dynamic user interface, which enables spontaneous interaction with ambient displays. Pfeuffer et al. (2013) investigated the effectiveness of smooth pursuits recognition for eye tracker calibration. They leveraged the fact that upon recognition of a smooth pursuit eye movement, the gaze position on the screen is known in terms of the coordinates of the moving object. Esteves et al. (2015) implemented the Orbits interaction technique for smartwatches. It is based on pupil tracking from a head-mounted eye tracker and smooth pursuits recognition for dynamic watch faces. Velloso et al. (2016) presented AmbiGaze for gaze-based control in smart environments. The AmbiGaze system recognizes if a user follows a moving, windmill-like appliance in a real scene to trigger an action. A systematic comparison of different recognition algorithms is also presented by Velloso et al. (2018). Khamis et al. (2018a) showed that smooth pursuits can be effectively used in virtual reality applications. Mattusch et al. (2018) also showed that recognizing smooth pursuits is possible if the moving object is hidden for some parts of its trajectory, as it could happen in games and virtual reality applications. Hassoumi et al. (2019) implemented EyeFlow as a proof-of-concept for smooth pursuits recognition based on an off-the-shelf RGB camera and optical flow estimation. Similarly, Bâce et al. (2020) combine appearance-based gaze estimation

using an RGB camera with optical flow estimation to recognize smooth pursuits. Sidenmark et al. (2020a) presented Outline Pursuits, an interaction method for virtual reality that allows users to select an object from a group by following a visual cue that moves along the outline of this object. Like usual gaze gestures, gaze-based interaction based on smooth pursuits is well-suited for authentication at public displays. Cymek et al. (2014) implemented a user interface for calibration-free PIN entry based on a number pad. Each digit moves, following a simple trajectory to enable smooth pursuits recognition. They achieve a high detection rate of 91.55%, but the minimum entry time is high (25 s). Liu et al. (2015) presented a similar approach for mobile phones based on the front-facing camera. Four objects, each assigned to a number between one and four, are located in the center of the screen. To enter a PIN, the user repeatedly follows a digit's trajectory to the screen edge. The input time is low (9.6 s), but also the accuracy (77.1%). Further, the password space is ten times smaller than for a typical four-digit PIN. Rajanna et al. (2017) proposed a system where users must follow three pre-defined shapes, moving along complex trajectories, out of 36 to authenticate. They observed a high accuracy of 96%, but the user can select from 12 shapes only, which yields a smaller password space than a four-digit PIN. A PIN entry is rather slow, lasting at least 15 s. Khamis et al. (2018b) introduced *CueAuth*, which is based on a digital number pad, similar to the approach by Cymek et al. (2014). However, instead of moving buttons, they use small moving circles within each button that differ in their trajectory, including linear, circular, and zigzag movements. On average, the entry time is 26.35 s with minimal entry times around 18 s.

3.1.2.3 Other Approaches

Other approaches enable calibration-free gaze-based interaction as well. One direction can be found in appearance-based gaze estimation methods that map input features like the head direction and close-up eye images to gaze vectors (Zhang et al., 2015b). Examples include the SideWays (Zhang et al., 2013) and GazeHorizon (Zhang et al., 2014) systems that enable spontaneous interaction with ambient displays based on off-the-shelf webcams. Another approach for mobile eye tracking is gaze estimation based on corneal reflections and pupil detection. Lander et al. (2018) introduced the hEYEbrid system that estimates the point of gaze in a scene based on the reflection of that scene in the user's corneal limbus. The system extracts a scene video from the reflections in the user's eye, and the pupil center depicts the point of gaze. Since the scene video and the pupil position stem from the same video stream, no calibration is required.

3.1.3 Summary

This section provided an overview of methods and interaction techniques in active gaze-based interaction. All presented methods address the question of how intentional eye movements can facilitate interaction with computer systems. However, gaze estimation errors from various sources can severely hamper the effectiveness and usability of interfaces that rely on the absolute gaze position as input. Low spatial accuracy and precision can be addressed using compensation methods like signal filters, gaze-to-object mapping algorithms, or special interface designs. Most of these methods only alleviate the symptoms but do not incorporate the gaze estimation error in

the interaction design. We propose the concept of error-aware gaze-based interfaces that incorporate real-time estimates of the gaze estimation error in chapter 4. We implement and evaluate methods for modeling the error via machine learning and use them for real-time error-adaptive object selection. We demonstrate the positive effects of incorporating error estimates in an active gaze-based interface. Another option is to use calibration-free interaction techniques based on gaze gestures or smooth pursuit eye movements. These are, by design, robust against gaze estimation errors. However, they typically do not provide high-fidelity interactions due to the lack of accurate on-screen gaze positions or require dynamic interfaces, which are only suitable for some applications. One such application can be found in gaze-based authentication at public displays. We implement a novel calibration-free authentication system that addresses flaws of existing gaze-based authentication systems in chapter 5. Common flaws include low accuracy in recognizing the entered PIN, high entry times, or the need for prior calibration. We use a circular design, similar to the system introduced by Best and Duchowski (2016), but without needing a prior calibration. Further, we implement *CueAuth* as a baseline system and compare the performance in terms of accuracy and entry time with our novel authentication method.

3.2 Human Gaze as a Passive Input Modality

This thesis also presents novel methods for interpreting the human gaze signal in the context of passive gaze-based interaction. Our contributions in part III relate to work on search target inference during visual search processes, on implicit relevance feedback from reading behavior and its applications, and on visual attention modeling with a focus on methods for mapping fixations to AOIs. We briefly introduce each topic and background information on shared computer vision topics, including image classification, object detection, few-shot learning, and image patch encoding based on convolutional neural networks (CNN).

3.2.1 Search Target Inference

Visual search is a perceptual task in which humans aim at identifying a search target object, such as a traffic sign, among distractor objects. Search target inference subsumes computational methods for predicting this target by tracking and analyzing overt behavioral cues of that person, e.g., the scanpath and fixated visual stimuli. In the following, we present related approaches and findings from the literature. Wolfe (1994) provided an extensive review of visual search processes and introduced a computational model of the visual search process, the *Guided Search 2.0* model, enabling inference of the search target. Their model distinguishes two visual processing stages: a preattentive stage that processes basic visual features like color and motion and a follow-up step that focuses on a higher-level semantic understanding of the scene. They aim to predict the search target by calculating an activation map from bottom-up features like color and orientation via feature maps and user-driven, task-related input. Newer approaches to search target inference are similar in that they model the visual search process using bottom-up features for a given stimulus in combination with user-driven input. User input is typically implicitly acquired using eye tracking technology. Zelinsky et al. (2013) showed that objects fixated during a visual search are likely to share similarities with the target's appearance. They presented a method that infers

the search target taking SIFT features (Lowe, 1999) and local color histograms from fixation-based image patches as input. Borji et al. (2015) presented methods for identifying a 3×3 sub-pattern, i.e., a search target, in a QR-code-like image. Their approaches include a simple distance function and a voting-based ranking algorithm that compares fixated sub-patches with available sub-patches for the considered stimulus. They showed that increasing the number of fixations used as input leads to an increase in classification accuracy. Sattar et al. (2015) encoded scanpaths from visual search trials by applying the *Bag of Visual Words* (`BoVW`) method on fixated image patches. In follow-up work, the authors combine gaze information and CNN-based features to infer the category of a user's search target instead of a particular object instance or image region (Sattar et al., 2017a). Targets have been chosen from the DeepFashion dataset (Liu et al., 2016). They also present a method to generate visual representations of the user's search target using fixated image patches and generative image models (Sattar et al., 2017b).

In chapter 6, we use the concepts introduced in Sattar et al. (2015) to develop our *Bag of Deep Visual Words* encoding for scanpaths. Further, we extend the basic idea to support applications in less constrained settings: we classify an automatically extracted segment class to localize the search target instead of predicting one out of five defined target classes.

3.2.2 Implicit Relevance Feedback

Previous research on implicit relevance feedback addressed how a user's eye movements can be associated with the relevance of a text in relation to a task or a trigger question.

3.2.2.1 Estimating Text Relevance

Several approaches have been proposed for estimating text relevance from reading behavior. Salojärvi et al. (2003, 2004, 2005a) investigated whether eye tracking can be used to estimate a user's perceived relevance of a document. They used machine learning to predict the relevance using the eye movements from reading the document titles as input. The authors organized a related research challenge (Salojärvi et al., 2005b). Loboda et al. (2011) presented an approach for gaze-based estimation of sentence relevance using fixations to sentence-terminal words, i.e., words at the end of a sentence, as there is empirical evidence that these words are fixated longer on average. This is known as the sentence wrap-up effect, a manifestation of the integrative process in reading. Buscher et al. (2008a) investigated the relation between reading behavior and document relevance using eye tracking technology. They found that the ratio of skimming is higher in irrelevant documents, and the ratio of continuous reading behavior is higher for relevant documents. Further, they introduced the concept of attentive documents that keep track of the perceived relevance based on eye movements (Buscher et al., 2012; Buscher, 2010). Gwizdka (2014a,b) modeled the relation between eye movements and perceived document relevance and investigated the cognitive effort involved in the relevance judgment. They introduced the g-REL corpus, a collection of short news stories and corresponding questions, which they used to collect ground truth relevance ratings and corresponding gaze data. The authors could confirm the findings from Buscher et al. (2012) that relevant documents tend to be read continuously, while irrelevant documents are rather skimmed (Gwizdka, 2014a). Akuma et al. (2016) compared gaze-based relevance feedback with implicit relevance feedback from more common

sensors such as mouse movements. They found a high correlation between both feedback options and a relationship between gaze-based features and perceived document relevance. Li et al. (2018b) investigated the reading behavior for relevant and irrelevant documents for factual and intellectual tasks. Based on data from a user study, they suggested a two-staged reading model for explaining the cognitive processes inherent in relevance judgments. Jacob et al. (2018) investigated whether eye movements can be used to infer the interest of a reader in a currently read article rather than the perceived relevance. Bhattacharya et al. (2020b) encoded fixations from participants' scanpaths over documents from the g-REL corpus and trained a convolutional neural network (CNN) with the perceived relevance as a prediction target. This approach is limited to small texts of similar lengths. Further, they suggested novel features based on the convex hull of scanpath fixations to model the participants' perceived relevance (Bhattacharya et al., 2020a). In addition, they simulated the user interaction to investigate whether their approach can be used in real-time scenarios by cumulatively adding fixations of the scanpath and normalizing the convex hull features with the elapsed interaction time. Hienert et al. (2019) developed a generic method for mapping gaze data to HTML documents at the word level. The tool was used by Davari et al. (2020) to investigate the role of fixations to words in query term prediction. Feit et al. (2020) modeled the user-perceived relevance of information views in a graphical user interface for decision-making. They showed room advertisements in a web-based interface via multiple viewports to users and asked them what information was perceived as relevant to their decision to book a room.

3.2.2.2 Query Expansion Methods

Other work focused on generating or expanding search queries based on the user's gaze behavior. Miller and Agne (2005) presented a system that extracts relevant search keywords from short texts based on eye movements. Hardoon et al. (2007) and Ajanki et al. (2009) proposed methods for implicitly generating search queries from eye movements during an information retrieval task. The system proactively retrieves relevant documents in the background using the generated query and content-based ranking. Buscher et al. (2008b) proposed a technique for automatic query expansion and re-ranking for document retrieval. They use relevance estimates to identify recently read paragraphs relevant to the user and, eventually, reformulate the search query. Chen et al. (2015) presented a query expansion method based on eye tracking and topic modeling. They identified fixated terms and modeled the user's latent intent using the Latent Dirichlet Allocation (LDA) for topic modeling.

3.2.2.3 Factors that Influence Eye Movements

Some researchers have focused on investigating factors that influence eye movements in the context of information retrieval and reading. Buchanan et al. (2017) surveyed work in gaze-based implicit relevance feedback. They identified several factors that might influence gaze patterns and should be considered when building gaze-enhanced information retrieval systems. Key factors include the task type, the task complexity, individual differences such as expertise, and the presentation of the search results. For instance, Cole et al. (2013) showed that "the user's level of domain knowledge can be inferred from their interactive search behaviors". Bhattacharya

and Gwizdka (2018) modeled the knowledge change while reading using gaze-based features: a high change in knowledge coincides with significant differences in the scan length and duration of reading sequences and in the number of reading fixations. Gwizdka (2017) investigated the task-related differences in reading strategies between a word search and relevance decisions during an information search. Eickhoff et al. (2015) studied the relationship between the user's visual attention to tokens in a search engine result page (SERP) or document and the corresponding search query: users fixate terms, which are part of their current query more often and longer than others. Further, they found that the semantic proximity of the search query to the user's attention increases for different reformulation strategies such as specialization, generalization, and reformulation.

In chapter 7 of this thesis, we investigate whether the perceived relevance can be estimated for paragraphs of long Wikipedia-like documents in contrast to sentences or short articles. This requires compensating for the scrolling activity, which may distort the gaze signal and fixation extraction, and developing a method for effectively extracting consecutive gaze sequences to individual paragraphs.

3.2.3 Visual Attention Modelling

Human gaze from eye trackers can be considered a proxy for human visual attention, i.e., "the allocation of limited attentional resources to certain information in the visual field, while ignoring other information" (Holmqvist and Andersson, 2017, p. 26). Hence, tracking human gaze via eye tracking is a very important tool in research and interaction design. Still, this is connected to tedious and costly human annotation, particularly in mobile eye tracking. In this thesis, we investigate whether pre-trained computer vision models and interactive machine learning (IML) approaches can be used to improve the annotation process. Next, we provide an overview of existing approaches for the annotation of mobile eye tracking data and video annotation in general. Further, we provide a brief overview of methods for real-time interpretation of eye tracking data that can be used to develop wearable attention-aware user interfaces (Toyama, 2015). Using unobtrusive modern eye tracking head-gear (see, e.g., Tonsen et al. (2017); Lander et al. (2018)) or augmented reality headsets like Microsoft's HoloLens 2 that come with integrated eye tracking sensors, our system for interactive annotation and model training can enable developers to easily create custom computer vision models for attention-aware mobile interaction.

3.2.3.1 Annotation of Data from Mobile Eye Trackers

Head-mounted eye trackers allow researchers to investigate human behavior in mobile settings. However, efficient methods for mapping fixations to AOIs from remote eye tracking cannot be used because the video of the front-facing scene camera differs for each participant. Instrumenting the experiment scene with fiducial markers is an option to cope with this issue (Yu and Eizenman, 2004; Pfeiffer et al., 2016). Software that accompanies modern head-mounted eye trackers typically integrates marker tracking, like the marker-based surface tracking in Pupil Capture (Kassner et al., 2014). However, the instrumentation of the experiment area comes with certain limitations. Marker tracking might be lost due to low camera quality or due to occlusion through other objects in the scene. In Augmented Reality (AR) settings, which allow learners to

see digital objects embedded in reality by looking through the camera of smartphones or tablets, supposedly unique markers might appear twice, causing ambiguity. Consequently, objects can no longer be distinctly identified by markers. Another disadvantage of marker-based surface tracking is that the numerous markers needed to reliably recognize objects in information-rich learning environments might impair the instructional design by claiming cognitive resources for the marker processing and distracting from learning-relevant visual stimuli. Therefore, this work focuses on an approach to facilitate and support the time-consuming and challenging procedure of mapping human gaze or fixations to objects or AOIs in *non-instrumented environments*. Commercial tools like Tobii Pro Lab[1] exist that offer automatic mapping of the gaze signal to AOIs defined in a reference image. However, the *assisted mapping* function works for static scenes only, is error-prone in cases of fast head movements and distorted image frames, and, hence, requires additional manual effort for correcting wrong assignments or annotating missing samples (Kumari et al., 2021). Further, the software is very expensive and does not support the annotation of eye tracking data from other devices like Pupil Core head-worn device that we used. Previous research also addressed this problem in the context of data analysis for diagnostic eye tracking studies. However, these approaches come with certain limitations.

Most approaches rely on pre-trained computer vision models that do not support an adaptation of the underlying models to the target domain. Sümer et al. (2018) investigated the problem of automatic attention detection in a teaching scenario. They extract image patches for all student faces in the egocentric video feed and cluster them using a ResNet-50 model (He et al., 2016) trained on VGGFace2 data (Cao et al., 2018). They assign student IDs to each cluster, allowing them to map the teacher's gaze to individual students. Chong et al. (2017) developed a system for measuring eye contact in adult-child social interactions using mobile eye trackers. Callemein et al. (2019) presented a system for detecting when the participant's gaze focuses on the head or hands of another person without the possibility of differentiating between interlocutors. Machado et al. (2019) matched fixations with bounding boxes from an object detection algorithm. They used a sliding-window approach with a MobileNet model (Howard et al., 2017), pre-trained on ImageNet data (Russakovsky et al., 2015). Venuprasad et al. (2020) used unsupervised clustering with gaze and object locations to detect visual attention to an object or a face. They used a Faster-RCNN model (Ren et al., 2015), pre-trained using the MS COCO dataset (Lin et al., 2014). Barz and Sonntag (2021) compared two approaches for automatic fixation-to-AOI mapping using pre-trained deep learning models: two ResNet models pre-trained with ImageNet data and a Mask R-CNN model pre-trained using MS COCO data. In an evaluation based on the VISUS dataset (Kurzhals et al., 2014a), they found that pre-trained models have severe drawbacks in realistic scenarios like AOIs not being represented by the training data. Deane et al. (2022) also presented an annotation system based on a pre-trained Mask R-CNN model (He et al., 2020). They found high agreements between manual and automatic annotations for AOIs that match the MS COCO classes. These can be applied in very constrained settings only, i.e., if the dataset used for training the machine learning model matches the target domain.

Other approaches suffer from a lack of flexibility. Wolf et al. (2018) developed an algorithm that maps fixations to object-based AOIs using the Mask R-CNN object detection model (He

[1] `https://connect.tobii.com/s/article/how-to-perform-manual-and-assisted-mapping` (accessed on 12 Dec 2024)

et al., 2020). They conducted a controlled lab study to record data in a healthcare setting with two AOIs: a bottle and five syringes. An evaluation has shown that using 72 training images with 264 annotated object masks, their system can closely approximate the AOI-based metrics compared to manual fixation-wise annotations as a baseline. Batliner et al. (2020) presented a similar system for simplifying usability research with mobile eye trackers for medical screen-based devices. Kumari et al. (2021) investigate the effectiveness and efficiency of three object detection models for annotating mobile eye tracking data from students participating in STEM lab courses. These methods are based on a single, a priori model training or fine-tuning step with no possibility of adapting the model during the annotation process.

Some approaches include promising interaction concepts but use outdated computer vision methods. Pontillo et al. (2010) presented SemantiCode, an interactive tool for post-hoc fixation-based annotation of egocentric eye tracking videos. It supports semi-automatic labeling using a distance function over color histograms of manually annotated fixations. Brône et al. (2011) proposed to use object recognition with mobile eye tracking to enhance the analysis of customer journeys. In follow-up work, they compared different feature extraction methods (De Beugher et al., 2012) and evaluated their approach in a museum setting (De Beugher et al., 2014). Evans et al. (2012) reviewed methods for mobile eye tracking in outdoor scenes ranging from pupil detection and calibration to data analysis. They presented an early overview of methods for automating the process of analyzing mobile eye tracking data. Fong et al. (2016) presented a semi-automatic data annotation approach. An annotator assigns video frames with a gaze overlay to AOIs, and as the annotation process advances, the system learns to classify AOIs via instance-based learning. Kurzhals et al. (2017) used bag-of-SIFT features and color histograms with unsupervised clustering to sort fixation-based image patches by their appearance. They offer an interactive visualization for manual corrections. Panetta et al. (2019) presented an annotation method based on bag-of-visual words as features and a support vector classification model (SVC) that is trained a priori. In follow-up work, they present a system that automatically segments objects of interest using two state-of-the-art neural segmentation models (Panetta et al., 2020). They used pre-trained models to showcase and evaluate new data visualization methods, but they did not assess the performance of their automatic annotation approach.

Recently, Kurzhals et al. (2020) described an interactive approach for annotating and interpreting egocentric eye tracking data for activity and behavior analysis. They implement an iterative time sequence search based on eye movements and visual features. They aim to annotate high-level activity events instead of AOI-hit events like we do. In follow-up work, Kurzhals (2021) presented an approach for annotating the objects viewed by study participants wearing mobile eye trackers. They propose to crop image patches around each point of gaze, segment the resulting image patches similar to the fixation detection method by Steil et al. (2018a), and present representative gaze thumbnails to annotators as image clusters in 2D. Annotators interact with this cluster representation to annotate and analyze the mobile eye tracking data. In contrast, our method is based on interactive few-shot image classification. Our system learns to recognize the type of fixated objects or regions based on human feedback during the interaction.

This work aims to accelerate and objectify research on visual attention with mobile eye tracking using technologies from the field of computer vision and interactive machine learning.

3.2.3.2 Video Annotation in General

The annotation of mobile eye tracking data requires the interpretation of the video feed from the front-facing scene camera. Hence, systems and methods for video annotation are closely related to our approach. An important difference is that general tools for video annotation do not take the gaze signal or fixation events into account. In fact, video annotation based on the definition of bounding boxes around relevant objects, a respective interpolation for intermediate frames, and a mapping of gaze or fixation points to these areas is the state-of-the-art for annotating video stimuli used with remote eye tracking devices (Kurzhals et al., 2014b). Even though these methods do not scale when it comes to the annotation of mobile eye tracking with individual video feeds for each participant, we briefly review recent approaches and tools for video annotation, as they can provide guidance for the design of similar systems. With LabelMovie, Palotai et al. (2014), presented a tool for collaborative video annotation. They proposed machine learning-based quality assurance and automation of the annotation process. In more recent work, the research group presented a method for the semi-automatic annotation of videos for analyzing the behavior of laboratory animals (Kopácsi et al., 2021). The Multimodal Multisensor Activity Annotation Tool (MMAAT) offers similar functionalities for multichannel data streams from multiple sensors, like depth channels from 3D cameras and accelerometers from wrist-worn devices (Barz et al., 2016b). The VGG Image Annotator (VIA)[2] is a stand-alone tool that enables manual annotation of images, audio, and video data in a web browser (Dutta and Zisserman, 2019). The Computer Vision Annotation Tool (CVAT) is an open-source system for interactive image and video annotation[3]. It integrates functionalities for scaling video annotation, like automatic pre-annotation based on computer vision models and keyframe-based interpolation of manual annotations, in an easily deployable online platform for large-scale projects. A general overview of interaction methods for video content was presented by Schoeffmann et al. (2015).

3.2.3.3 Methods for Attention-aware Interfaces

Human gaze can be considered a proxy for human visual attention and thus can enhance gaze-based multimodal interaction (Qvarfordt, 2017). We provide a brief overview of such real-time interactive systems because they can benefit from our presented approach for interactive annotation of mobile eye tracking data. Related work includes approaches for building user interfaces that are aware of the current context or situation (Bulling, 2010), including conversational interfaces (André and Chai, 2013). For instance, Bulling et al. (2013) presented an approach for inferring high-level contextual cues from eye movements to facilitate behavioral monitoring and life-logging. Similarly, Steil and Bulling (2015) used topic modeling to detect everyday activities from eye movements in an unsupervised fashion. In a later work, the authors presented an approach for visual attention forecasting in mobile interaction settings, which takes the visual scene and device usage data as additional inputs (Steil et al., 2018b). Toyama et al. (2012) implemented the Museum Guide that uses SIFT (scale-invariant feature transform) features (Lowe, 2004) with the nearest neighbor algorithm and a threshold-based event detection to recognize user attention to one of 12 exhibits. They extended their approach to detecting read texts and

[2] `https://www.robots.ox.ac.uk/~vgg/software/via/` (accessed on 12 Dec 2024)
[3] `https://github.com/opencv/cvat` (accessed on 12 Dec 2024)

fixated faces with the goal of building artificial episodic memories to support dementia patients (Toyama and Sonntag, 2015). Other approaches combine visual features of a scene with gaze information to detect actions recently performed by a user (Fathi et al., 2012; Li et al., 2015, 2018a; Shiga et al., 2014). Prasov and Chai (2008) developed a system that combines speech and passive gaze input to enhance reference resolution in conversational interfaces. Baur et al. (2015) implemented NovA, a system for analyzing and interpreting social signals in multimodal interactions with a conversational agent, which integrates eye tracking technology. Thomason et al. (2016) developed a gaze-based dialog system that enables the grounding of word meanings in multimodal robot perception. Uppal et al. (2022) presented a method for segmenting the fixated object using an end-to-end computer vision model. Chang et al. (2021) developed the MemX system that detects human visual attention based on mobile eye tracking and automatically extracts important video sequences that can be used for, e.g., lifelogging. Meyer et al. (2022) proposed to use head- and eye movement in combination with other sensor data to recognize human activities for building context-aware smart glasses.

3.2.4 Computer Vision

Computer vision is the algorithmic equivalent of human visual perception and subsumes image classification and object detection methods. Image classification refers to assigning a single label to an image; object detection refers to localizing and classifying multiple objects in one image (Russakovsky et al., 2015). Recent methods experienced a performance boost due to advances in deep learning technology and the availability of large datasets for model training. Popular examples are the ImageNet dataset for image classification (Russakovsky et al., 2015) and the MS COCO dataset for object detection (Lin et al., 2014). A good overview of object detection was presented by Zhao et al. (2019). Research has shown that image representations from hidden neural network layers of image classification models can be reused for different tasks, including image clustering and transfer learning. Razavian et al. (2014) developed scene recognition and object detection methods based on the L2 distance between the vector-based image representations from pre-trained CNN models. Donahue et al. (2014) showed that using image representations from a CNN model can be used to build models for a label prediction task that outperforms previous state-of-the-art approaches. They also showed that CNN-based image representations create semantically coherent clusters: images with similar content are located close to each other. Jiang and Canny (2017) presented an intelligent user interface for fine-tuning models based on a pre-trained AlexNet (Krizhevsky et al., 2012) model.

We use CNN-based image features to model the relation between the fixation history of a visual search and the visual representation of the target or a target's segment class in chapter 6. We include different layers of a pre-trained AlexNet[4]. In addition, we propose a scanpath encoding method based on the neural image segmentation model SegNet (Badrinarayanan et al., 2017), which is trained on the SUN RGB-D dataset (Song et al., 2015) with 10,000 images of indoor scenes annotated for object detection, classification, and segmentation. We encode a scanpath as the sequence of fixated segment classes.

In chapter 8, we use a residual network model (He et al., 2016) pre-trained on ImageNet

[4] `https://github.com/happynear/caffe-windows/tree/ms/models/bvlc_alexnet` (accessed on 12 Dec 2024)

and a Mask R-CNN model for object detection (He et al., 2020) pre-trained on MS COCO for fixation-to-AOI mapping. Further, we suggest using few-shot image classification for interactive model training in the context of mobile eye tracking data annotation. Our approach is based on the idea of reconstruction (Zhang et al., 2020; Lee and Chung, 2021; Li et al., 2023) where the class membership task is framed as a problem of reconstructing feature maps. We have used a Feature Map Reconstruction Network (FRN) (Wertheimer et al., 2021), which classifies a target image by reconstructing class associate feature maps of the image using a set of support features.

Related approaches also include methods for egocentric activity recognition without gaze data. For example, Ma et al. (2016) use hand segmentations, object localizations, and the optical flow from egocentric videos to infer ongoing activities. Another example is EgoNet by Bertasius et al. (2016), which determines the action-object in egocentric videos.

Part II

Human Gaze as an Active Input Modality

Chapter 4

Error-aware Gaze-based Interaction

Head-mounted eye trackers are promising for mobile interaction as they provide information about the user's intentions. Human gaze conveys the user's interest (Shell et al., 2003), which is already used for interaction with one or multiple displays (Lander et al., 2015; Stellmach et al., 2011; Turner et al., 2014). A key problem of mobile gaze-based interaction is that the gaze estimation error, i.e., the difference between the estimated and true on-screen gaze position, can be substantial, particularly if the user moves in front of a display (Lander et al., 2015; Mardanbegi and Hansen, 2012). Besides user position and orientation, factors specific to the eye tracker and display, e.g., parameters of the calibration routine and the display detection algorithm, can significantly impact the gaze estimation error (Barz et al., 2015, 2016a). Some interaction techniques omit the need for accurate point of gaze (POG) estimates, e.g., by correlating raw eye movements with animated on-screen targets (Vidal et al., 2013; Esteves et al., 2015), but introduce the need for dynamic user interfaces. Other methods aim to address this problem by filtering gaze jitter (Špakov, 2012), snapping gaze to on-screen objects (Špakov and Gizatdinova, 2014) or by optimizing interface layouts for gaze estimation error at the design stage (Feit et al., 2017). Although such methods can improve user experience, they only alleviate the symptoms and do not embrace the inevitable gaze estimation error in the interaction design. Also, the gaze signal from head-mounted eye trackers is provided as a sequence of scene camera coordinates. Accordingly, it must be mapped to the coordinate system used for interaction (Kassner et al., 2014). A key challenge for mobile gaze-based interfaces is that the tracker's pose relative to that coordinate system needs to be tracked. For this purpose, Bardins et al. (2008) attached infrared LEDs to an eye tracker and used a stereo camera at the screen to track the 3D pose of the headset. Several approaches are based on visual markers at the screen to reconstruct the eye tracker pose and estimate the user's on-screen gaze position (Breuninger et al., 2011; Hales et al., 2013; Yu and Eizenman, 2004). Mardanbegi and Hansen (2011) implemented an algorithm that detects rectangular displays in the eye tracker's field of view and maps gaze points using a homography matrix. Kassner et al. (2014) developed an open-source head-mounted eye tracker which, in its most recent version, supports surface tracking with visual markers[1]. Lander et al. (2015)

[1] `https://docs.pupil-labs.com/core/software/pupil-capture/#surface-tracking` (accessed on 12 Dec 2024)

proposed GazeProjector, a system for mobile gaze-based interaction with ambient displays. They seamlessly integrate multiple displays based on natural feature tracking to estimate the user's pose. We use marker tracking similar to Kassner et al. (2014) to map gaze from scene camera coordinates to display coordinates to enable on-screen interaction with mobile eye trackers.

We present the first ever gaze-based user interface that is "aware" of the ever-changing gaze estimation error and can adapt to the error on the fly. We implement a prototype for error-aware gaze-based selection that scales targets proportional to real-time error estimates. In total, we develop four models, which include two simple models based on a fixed angular error, a predictive model, and a baseline using a fixed target size (Barz et al., 2015, 2016a). We evaluate our prototype in a user study with 12 participants. The best selection rates are achieved with the predictive model on the cost of large target sizes: it is unclear to what extent the advantage in selection rate originates from the target size. Based on our experiences from this first study, we develop two more advanced error compensation methods and investigate their effect on selection rates and target sizes using our study corpus (Barz et al., 2018). The results of our analysis suggest that the selection rate can be improved using directional error estimates without increasing the average size of targets. However, this advantage vanishes with increasing target sizes. A great improvement can be achieved by training personalized directional error models: with this approach, high selection rates can be achieved even with low target sizes. All in all, our architecture and error models enable a new class of gaze interaction that incorporates gaze estimation errors and is suitable for many fields of application in mobile and ubiquitous computing. We contribute as follows:

- Section 4.1: We present a generic software architecture for error-aware gaze-based interaction (see figure 4.1).
- Section 4.2: We implement the first error-aware gaze interface that scales selection targets proportionately to real-time error estimates (see figure 4.2). We integrate four gaze estimation error models (Barz et al., 2016a).
- Section 4.3: We systematically evaluate our prototype and the gaze estimation error models in a mobile interaction user study with 12 participants.
- Section 4.4: Informed by the findings of the first study, we develop and evaluate two advanced adaptation techniques: a method based on Feit et al. (2017) that scales gaze selection targets according to the 2D error distribution in the data and a novel method that combines scaling targets and shifting gaze by directional error estimates.

4.1 Software Architecture

We propose an approach for gaze-based interaction that, in contrast to existing methods, incorporates the gaze estimation error of the eye tracker in real-time. This section describes our proposed software architecture for error-ware gaze-based interfaces (see figure 4.1). It includes three main components. The *Gaze Estimation and Input Data Acquisition* component connects an eye tracker to receive gaze information with respect to any pre-defined display and further

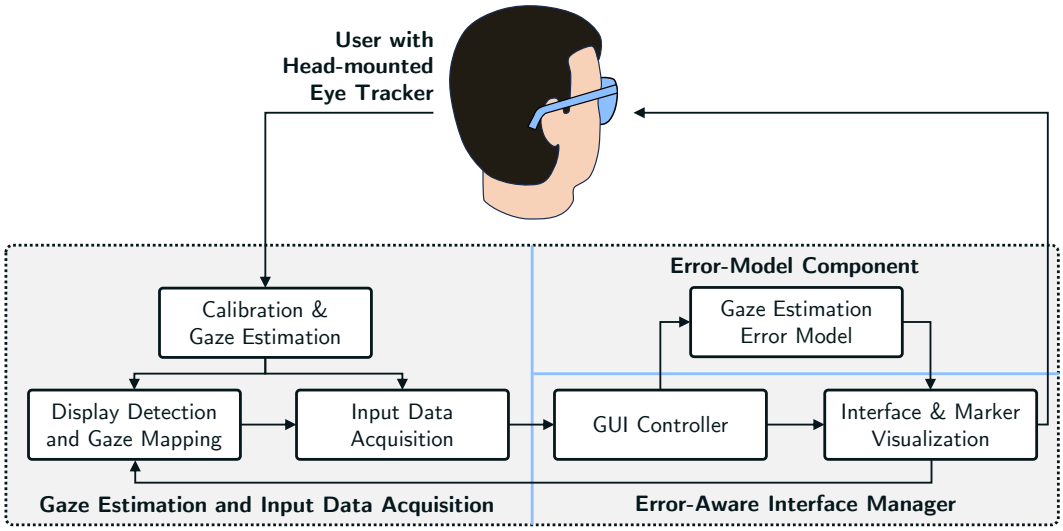


Figure 4.1: Generic architecture of an error-aware gaze-based interface.

inputs required for predicting the gaze estimation error. In most cases, this part will integrate with the software package accompanying the eye tracking hardware. In principle, this component could also connect remote eye tracking devices, but we focus on mobile, head-mounted eye trackers in this section. All parameters are sent to the *Error-Aware Interface Manager* that adjusts the interface by compensating the gaze estimation error based on a real-time estimate of an interchangeable *Error-Model Component* and presents it to the user. In the following, we describe the individual components of our architecture and their interplay.

4.1.1 Gaze Estimation and Input Data Acquisition

This component has three major tasks. First, it connects an eye tracking device to handle calibration and to receive gaze data. Second – if not fulfilled by the eye tracker's API – it is responsible for detecting displays and mapping gaze on these displays. Third, the component acquires all necessary input parameters for the *Error-Model Component*. The marker visualization, as well as executing the error estimation, is part of the *Error-Aware Interface Manager*.

4.1.2 Error-Aware Interface Manager

The interface manager is the core component of our framework and accomplishes several tasks. First, it handles the interface elements comprising their properties and events, similar to any user interface framework. Any module, e.g., containing the application logic, can add elements, influence its position, and register for events. Second, it handles the presentation of markers used by the *Gaze Estimation and Input Data Acquisition* to identify the display and compute the eye tracker pose. Third, and most importantly, this component interfaces the *Error-Model Component* to gather essential data for, e.g., adjusting the size of interface elements and shifting the input signal. In this work, we implement and evaluate different compensation methods

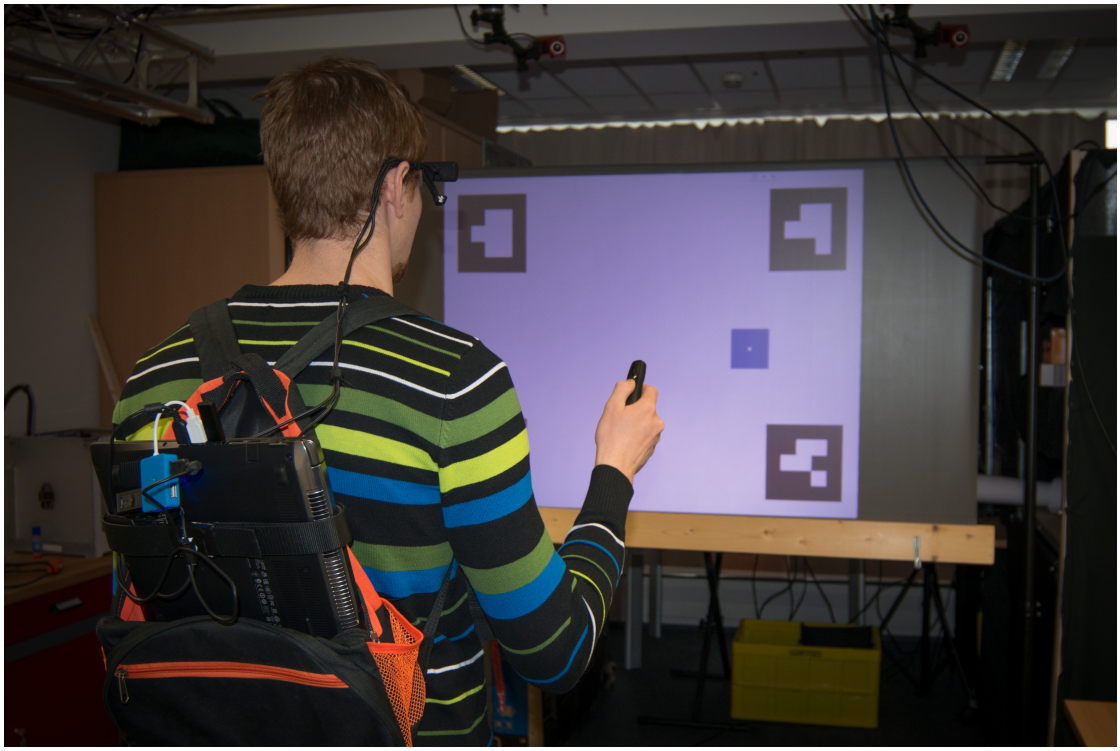


Figure 4.2: Study participant interacting with our error-aware gaze-based interface using a mobile eye tracker and a presenter as selection trigger.

summarized in figures 4.3 and 4.7.

4.1.3 Error-Model Component

The error-model component encapsulates all calculations and algorithms for predicting the gaze estimation error via a shared interface. It is important that the computation time for the error inference is suitable for real-time applications: it should be on par with the sampling rate of the gaze estimation (e.g., between 30 and 200 Hz for mobile devices). The concrete implementation should allow developers to exchange the error model for testing different approaches or updating models if newer revisions are available. For instance, when a model can be personalized for each user. We investigate different error models that serve as input for error-compensation methods in the *Error-Aware Interface Manager*.

4.2 Prototype Implementation

We implement an error-aware interface based on the proposed software architecture to show the feasibility of error-aware gaze-based interfaces and to validate our architecture. Our prototype allows users to select a button by looking at it while pressing a mobile presenter's button (see figure 4.2). Our goal is to increase the selection rate through real-time optimization of the

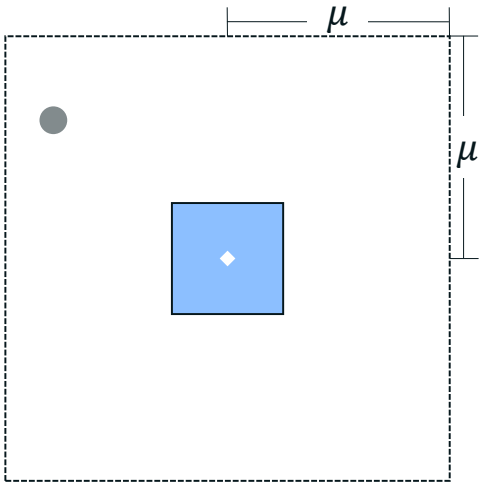


Figure 4.3: The *NaïveScaling* method adapts the interface by scaling selection targets likewise for two dimensions (horizontal and vertical). Scaling is based on an interchangeable error model that provides absolute error estimates in real-time.

button size, i.e., to increase the edge size as much as required for successful selections based on the current gaze estimation error. Next, we describe each component of our prototype in detail.

4.2.1 Gaze Estimation and Input Data Acquisition

Our prototype is based on a monocular Pupil Labs Pro eye tracker, which captures gaze at 30 Hz (Kassner et al., 2014). The resolution of the scene camera is 1280×720 pixels and 640×480 pixels for the eye camera. The device is connected to a laptop inside a backpack worn by a user (see figure 4.2). This component is based on Pupil Capture, an extensible software package provided by the eye tracking manufacturer for device calibration and gaze estimation. We extend Pupil Capture by a marker-based display detection component to map gaze from the scene camera coordinate system to the coordinate system of any display or interactive surface in the environment. Further, our plugin collects all data required for other components of our interface prototype, especially for the *Error-Model Component*. A detailed description of the required input data can be found in Barz et al. (2016a).

4.2.2 Error-Aware Interface Manager

Our prototype can visualize a single clickable button with a dynamic edge size at a given position. Buttons are shown as blue rectangles with a white dot at their center. To select a target, a user can press a button on a wireless presenter while fixating on it. This additional modality solves the Midas problem inherent to gaze-based interfaces. When the user has performed a button click, we check if there was a recent fixation within the area of a button and raise its trigger

event accordingly. It flashes green upon a successful selection and red otherwise. The edge size can be bound to real-time error estimates from the *Error-Model Component* via an error compensation method. We implement a *NaïveScaling* approach: the size of the selection target is calculated as two times the gaze estimation error μ as shown in figure 4.3. We double the error estimates because they are absolute and, hence, do not contain directional information. It is possible to change the active model on-the-fly, also during runtime. To prevent jumpy changes in an element's shape, we smooth the edge size using the 1-Euro-Filter (Casiez et al., 2012) with $\beta = 0.01$ and $f_{c_{min}} = 0.3$ as parameters. In our demo setup, we use a back-projection screen with a resolution of 1024×768 pixels and a pixel density of 8.88 px/cm.

4.2.3 Error-Model Component

A key building block of our error-aware interface is a model that estimates the gaze estimation error for head-mounted eye trackers. We implement four models, two simple models based on a fixed angular error, a predictive model, and a baseline using a fixed target size.

4.2.3.1 Simple and Baseline Error Models

We implement two simple models based on Barz et al. (2015, 2016a): the models *Best* and *Measured* calculate the gaze estimation error based on the distance d of the user's head to the current on-screen target and a constant angular error e_c. This error is either 0.6° as stated by the hardware manufacturer (best-case) or 1.23° as measured for the actual setting in (Barz et al., 2015). The function to approximate the gaze estimation error μ with the distance d and the error $e_c \in \{0.6°, 1.23°\}$ as input is $\mu = d \cdot tan(e_c)$. The baseline model *None* reports a constant error, which is computed once with 1.23° and a static distance of $d_{cal} = 175$ cm as input (center of the interaction space).

4.2.3.2 Predictive Error Model

Monocular head-mounted eye trackers are typically equipped with two cameras: a scene camera that captures part of the user's current FOV and an eye camera that records a close-up video of the user's eye (Kassner et al., 2014). The problem of gaze estimation can be defined as mapping 2D pupil positions in the eye camera coordinate system to 2D gaze positions in the scene camera coordinate system (Majaranta and Bulling, 2014). The mapping is usually established in a calibration process. Pupil positions and corresponding scene camera positions are then typically mapped to each other using a first or second-order polynomial. Suppose these gaze positions are meant to be used for interacting with a display placed somewhere in the environment. In that case, they must be mapped further to the corresponding display coordinate system, e.g., by using visual markers attached to the display (Yu and Eizenman, 2004) or by detecting the display itself (Mardanbegi and Hansen, 2011). This indicates two main components where errors can arise (see figure 4.4): 1) the mapping of 2D pupil positions in eye camera coordinates to 2D scene camera coordinates (*Pupil Position Mapping*), as well as 2) detecting interactive displays in the environment and mapping gaze from scene camera coordinates to display coordinates (*Display Detection and Mapping*).

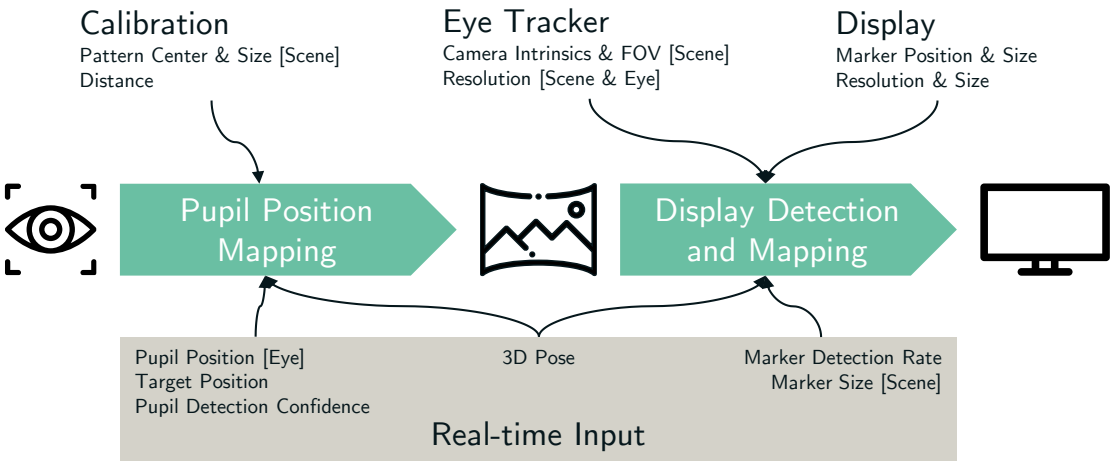


Figure 4.4: Gaze estimation error model for head-mounted eye trackers comprising two main building blocks: Pupil Position Mapping and Display Detection and Gaze Mapping. Model inputs include parameters for Calibration, Eye Tracker, and Display, as well as the real-time gaze, visual marker, and 3D head pose, some specific to the Eye or Scene camera.

4.3 User Study

The proposed error-aware interface is evaluated for mobile gaze interaction in a public display setting. The interaction with the interface consists of a gaze-based selection task on a large display. Hereby, the *Error-Aware Interface Manager* uses estimates of the *Error-Model Component* for scaling selection targets in real-time: The larger the predicted gaze estimation error, the larger the targets (see figure 4.3). We invited 12 participants (six female) aged between 20 and 53 ($M = 28.68, SD = 10.84$).

Conditions We investigate the performance of our *Naïve* target scaling approach using different error models for the *Error-Model Component*. The error models introduced above correspond to our four conditions (*Best*, *None*, *Measured*, and *Predictive*).

Tasks For each condition, a calibration is performed at the center of a 3×3 grid with 50×50 cm cells starting 100 cm in front of the display (approximately 175 cm and orthogonal to the screen). Then, the selections are performed using six on-screen targets (radially arranged with one at the display center) from all positions of the 3×3 grid, totaling 216 selections per participant. Stimuli are shown at the same positions in randomized order.

Design In the user study, we consider four different methods for predicting the gaze estimation error in our error-aware interface (within-subject design). The order of conditions is counterbalanced between participants.

Procedure Participants are introduced to the experiment and asked to complete a general questionnaire. Then, per condition, each participant calibrates the eye tracker and performs all selections for the currently considered error prediction method (counterbalanced order). We instruct the participants to be as accurate as possible. The grid position is shown to the user prior to each selection. On average, one run lasted 967 s.

Apparatus We use the gaze-based prototype for button selection described in section 4.1 with the *NaïveScaling* error compensation. Our setup uses a back-projection screen with 1024×768 pixels and a pixel density of 8.88 px/cm. We mark a 3×3 grid in front of the display with adhesive tape to coarsely position the participants without restricting their mobility. Also, we collect the 3D head pose of the user based on the marker tracking component to get more fine-grained location data (continuous values for distance and angle).

Independent and Dependent Variables Independent variables include the error prediction method, the user position, and the on-screen target. The models correspond to the conditions outlined above. The 3×3 grid enforces the nine different user positions. Additionally, we record the 3D pose to get the distance between the user and the target as well as the angle of the user to the display. The dependent variable is the selection rate and the size of the target area.

Hypotheses We hypothesize that the *Naïve* compensation approach achieves the best selection results with the *Predictive* model with respect to the selection rate (H1). This method will especially outperform the other approaches for varying distances (H2) and orientations (H3) of the participants in front of the display with reasonable target sizes (H4).

4.3.1 Results

Averaged over all on-screen targets and grid positions, the selection rate is 22.47% ($SD = 15.56$) for *Best*, 48.92% ($SD = 29.09$) for *None*, 53.4% ($SD = 22.53$) for *Measured* and 81.48% ($SD = 17.99$) for *Predictive* (see figure 4.5). A repeated measures ANOVA (N=12) shows that the differences are significant ($F(3, 9) = 56.294, p < 0.001$). All pairwise differences (Bonferroni-corrected) are significant, besides the ones between *Measured* and *None*. In addition, all but one participant judged *Predictive* as their favorite method.

We further analyze the effect of distance and angle of the user's head to the corresponding on-screen target on the selection rate. We cluster the data into three groups for both variables based on the respective histogram (visually inspected). The resulting intervals are $[80, 150]$ cm (near), $[150, 215]$ cm (mid), and $[215, 290]$ cm (far) for the distances and $[-50, -12.5]^\circ$ (left), $[-12.5, 12.5]^\circ$ (center) and $[12.5, 50]^\circ$ (right) for the angles to the selection target. For *Measured*, we observe a significant drop in selection rate when moving from the calibration position towards the display (from mid to near) of 43.18% ($F(2, 10) = 14.127, p = 0.001$). Results for *Best* also decrease by 34.92%, but not significantly ($F(2, 10) = 1.448, p = 0.28$). For *None*, we find an inverse effect, i.e., the selection rate increases by 30.39% ($F(2, 10) = 9.988, p = 0.004$). The results for the *Predictive* model reveal a similar effect as for *Measured* and *Best*, but the selection

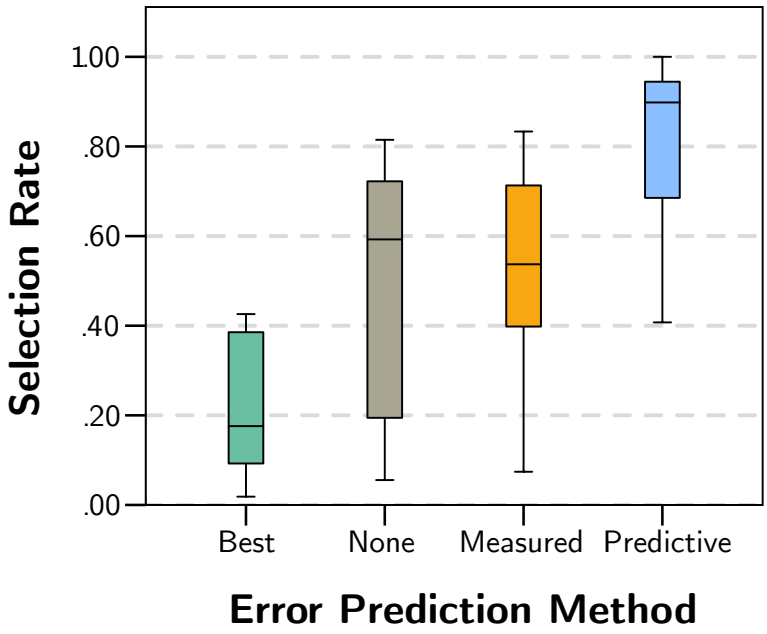


Figure 4.5: Mean selection rate of different error estimation models averaged over on-screen targets and grid positions.

rate only decreases by 20.64% ($F(2, 10) = 17.298, p = 0.001$). We find no significant differences in the selection rate considering the intervals for the angles.

Concerning the size of generated targets, we compute the angular error $e_c = 1.98°$ for *Predictive* that would generate, on average, the same target sizes when using the error estimation function of *Best* and *Measured*. The inverse of this function is used with the distance and error estimation result of each selection as input. Further, we average the selection rate and the target area over all participants, which maintains the variance for different user positions and targets for each error model (see figure 4.6). Hereby, the performance of the *Predictive* model is measured with reduced and increased edge lengths to see whether it systematically over- or underestimates the gaze estimation error using $\{0.8, 0.9, 1.1, 1.2\}$ as factors for scaling the edge length. Concerning the *Predictive* method, the selection rate improves with a regressive slope, whereas the target area grows quadratically. The mean target size for the *Predictive* model is 168.15 cm^2 ($SD = 84.67$) which is larger than 76.15 cm^2 ($SD = 41.2$) for *Measured*, 60.38 cm^2 with zero variance for *None* and 17.09 cm^2 ($SD = 9.27$) for *Best*.

4.3.2 Analysis

The evaluation confirms that our compensation method *Naïve* performs best with the predictive model (Barz et al., 2016a), significantly outperforming the two simple models as well as the baseline method for gaze error estimation (supports H1). On average, the method *Best* performs significantly worse, and *Measured* performs as well as *None*, which uses no error compensation.

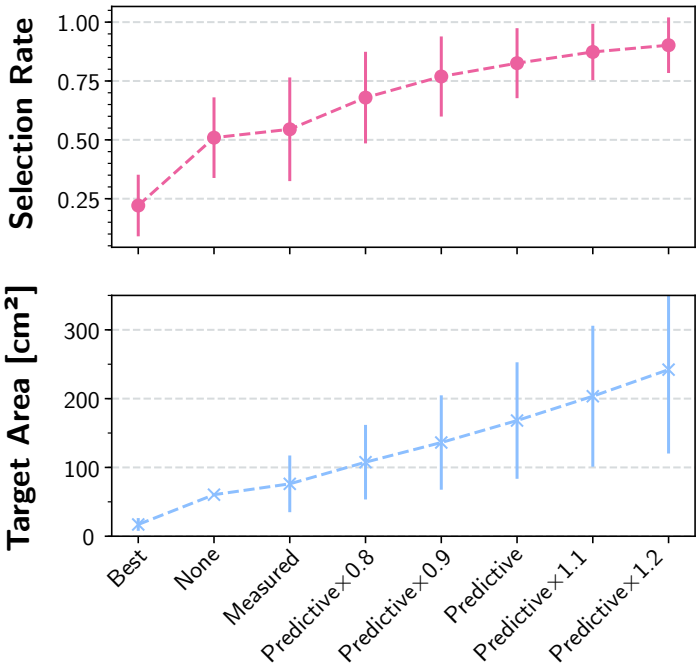
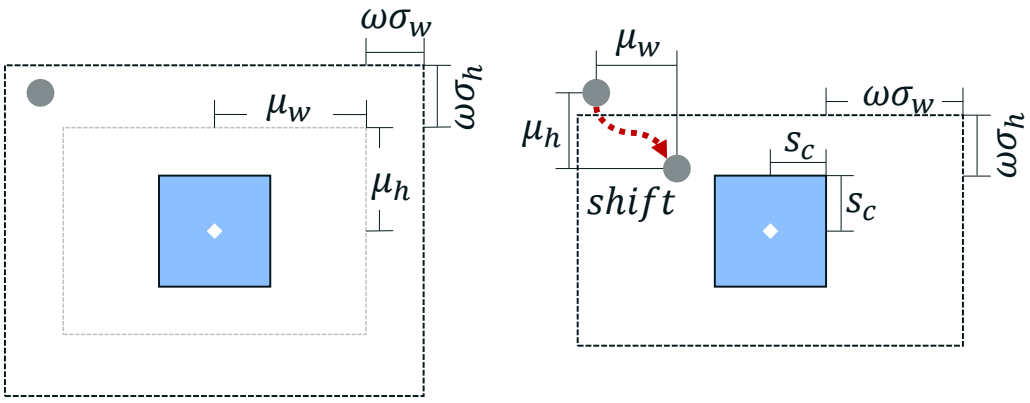


Figure 4.6: Mean selection rate and mean target area for different error estimation models averaged over all participants. The error bars indicate the standard deviation (±) of n=54 selections per model.

However, the performance for far user positions significantly increases for *Measured* whereas it decreases for *None*, and the same holds for *Predictive* and *None*. This inverse behavior confirms H2. We could not find a dependency between the selection rate and the angle of the user to the display and thus could find no evidence for H3. Our evaluation shows that the gaze estimation error of 0.6° (*Best*), as reported by the manufacturer, is inapplicable for mobile settings. *Measured* assumes a more realistic error than *Best*, which yields better selection rates. The model-based approach *Predictive* outperforms all baseline methods regarding the selection rate but would presume the highest error of 1.98° if it was a distance-dependent method.

The distribution of its target areas ($M = 168.15$ cm²) as shown in figure 4.6 is broader compared to *Measured* and *Best* indicating a higher degree of adaptation; accordingly, the variance of *None* is zero. Systematically increasing and decreasing the estimated target size shows how the selection performance can be traded against the target size. Also, it shows that *Predictive* seems to be a reasonable compromise. However, we cannot confirm H4 because it is unclear which portion of the improvement stems from choosing the target sufficiently large and to what extent from the adaptive behavior of the *Predictive* model. Another reason might be the high variance in performance of different participants, which these models cannot explain, i.e., the *Predictive* model might predominantly explain the system error of the eye tracking device. Using the *Naïve* scaling approach results in undesirable selection rates or relatively high target sizes, which might be up to the compensation method or the error models. For this reason, we develop a more sophisticated method that incorporates directional error information, enabling



(a) *Feit2dDist* (Feit et al., 2017) **(b)** *PredictiveShift*

Figure 4.7: We implement and evaluate two additional methods for adapting the interface and for compensating the gaze estimation error: (a) the method *Feit2dDist* (Feit et al., 2017) that scales targets based on the 2D error distribution extended for mobile settings and (b) our novel method *PredictiveShift* that shifts gaze based on a directional error estimate.

not only naïve scaling of interface controls but also shifting gaze toward the user's actual point of regard. Further, we investigate user-specific error modeling to cover personal differences in tracking quality and interaction style. We evaluate our new compensation method and compare it to a recent approach by Feit et al. (2017) in a post-hoc analysis with our recorded data.

4.4 Advanced Compensation Methods

Based on our findings from the user study, we develop two additional compensation methods: *Feit2dDist* that is based on a recent model by Feit et al. (2017) that is based on the 2D distribution of the gaze estimation error and a novel model *PredictiveShift* that shifts gaze by a directional error estimate. Both methods implement the compensation part of the *Error-Aware Interface Manager* and their specific counterpart in the *Error-Model Component*.

The compensation method *Feit2dDist* is based on a work by Feit et al. (2017). They introduce an approach for modeling the eye tracking error using a two-dimensional Gaussian. For each on-screen position, they compute the spatial accuracy as the mean offset μ and the spatial precision as the standard deviation σ of the respective gaze samples in x and y direction. μ and σ which define the 2D Gaussian are used for approximating the gaze error distribution. The target size for a specific on-screen position in terms of width and height is computed as $S_{w/h} = 2 \cdot (\mu_{w/h} + 2 \cdot \sigma_{w/h})$. Using the doubled standard deviation to either side includes around 95% of all samples assuming a normal data distribution. By definition, this approach will achieve

a selection rate of around 95%, assuming that the error distribution does not differ significantly during the interaction. However, this approach was neither meant nor used for real-time error estimation in adaptive user interaction. It is limited to a static setting where only the on-screen targets can vary. We extend their approach for mobile settings: we consider different positions in front of the display using a look-up table with pre-computed means and standard deviations for all combinations of grid and screen positions for inference. Further, we add the factor ω to control the influence of the precision estimate σ, allowing us to investigate the effect of changing target sizes. Our adapted version computes the target size as $S_{w/h}(\omega) = 2 \cdot (\mu_{w/h} + \omega \cdot \sigma_{w/h})$ depending on the grid position where the user is standing and the location of the on-screen target (see figure 4.7a).

Our novel approach *PredictiveShift* is based on modeling the directional information of the gaze estimation error (i.e., the offset), which enables shifting the estimated gaze point to the actual point of regard, thus reducing the systematic error of the eye tracking setup and personal characteristics in focusing gaze targets. The error model is a multivariate ElasticNet regression model implemented in scikit-learn (Pedregosa et al., 2011) using a preceding standard scaler for standardizing input features by removing the mean and scaling them to unit variance. We use the following predictor variables from our user study for estimating μ and σ (for x and y): the coordinates of the on-screen target, the distance to the fixated display region, the horizontal angle to the display and the estimated gaze position in world camera coordinates. The distance, angle, and on-screen position are similar to the input of the distribution-based approach by Feit et al. (2017). We add the gaze estimates in world coordinates to cover potential systematic weaknesses of the tracking device, e.g., in regions close to the border of the camera's field of view. In contrast to all other methods, the spatial accuracy is modeled, including its directional information, which allows for shifting the measured gaze toward the actual point of regard (see figure 4.7b). Like the model above, we consider a weight factor ω to scale the estimated spatial precision. The target size is computed as $S_{w/h}(\omega) = 2 \cdot (s_c + \omega \cdot \sigma_{w/h})$ with $2 \cdot s_c = d_{cal} \cdot \tan(1.23°)$ being the target size as computed for the baseline method *None* for the *Naïve* approach. We use this static base size as a lower bound because we observed a significant plus in selection rates for near-user positions. The participant's gaze is corrected as follows: $\mathit{shift}(g_{x/y}, \mu_{w/h}) := g_{x/y} + \mu_{w/h}$.

We hypothesize that compensation with *PredictiveShift* achieves higher selection rates than *Naïve* and Feit et al. (H1.1), particularly enhancing the ratio of target size and selection rate (H4.1); personalized error models yield a further improvement (H4.2).

4.4.1 Model Training and Evaluation

We use the recorded interaction sessions from our user study to train and evaluate the compensation methods. First, we pre-process the data extracting all relevant information and excluding outliers. The subsequent steps are based on this cleaned dataset.

4.4.1.1 Pre-Processing

The recorded dataset includes the raw and metadata of all selection trials from our study. We extract the relevant parts of each selection sequence by cropping the signals starting half a second before the selection was triggered (presenter click) and stopping at that event. Outliers

	Best	None	Measured	Predictive
Selection Rate [%]	21.37	54.19	52.99	83.76
Target Size [cm²]	16.33	55.24	68.62	170.36

Table 4.1: Mean selection rate and mean target size from the offline simulation using the error compensation methods from the user study.

are removed based on the mean gaze offset to the actual target center and the distribution of the standard deviations of the gaze signal for the cropped intervals. We drop a selection trial if the mean offset is greater than five degrees (see Kassner et al. (2014)) and if the standard deviation is greater than the 95^{th} percentile of all standard deviations. High offsets commonly appear when the marker tracking fails or when the eye tracking headset is displaced. High variance values can occur when the user's pupil cannot be tracked well. We drop 8.80% of the data so that 2364 selections from 12 users remain. On average, the dataset contains 197 selection trials for each user ($SD = 13.93$).

4.4.1.2 Model Training

The error estimation of the applied compensation methods is data-driven and requires a training phase described here. We split the data into a train (75%) and a test set (25%). Concerning the user-specific models for *PredictiveShift(personal)*, we split the data per user with the same ratio. For *Feit2dDist*, we compute the means μ and standard deviations σ on the train-set, as described above, and store them in a look-up table. For our new predictive model, we conduct a 5×5 nested cross-validation (k-fold) on the train set where the inner loop performs a grid search for optimizing model parameters of the ElasticNet algorithm. The search space includes the degree of polynomial features $\in \{1, 2\}$, the factor alpha$\in \{10, 1, .1, .001\}$ weighting the penalty terms and the l1_ratio $\in \{0, .5, 1\}$ trading off the L1 and L2 regularisation.

4.4.1.3 Evaluation Procedure

We use the selection trials from the test set for validating all error compensation methods, including *Naïve* with all error models, *Feit2dDist*, *PredictiveShift* and *PredictiveShift(personal)*. For each selection and compensation method, we infer the gaze error using the recorded signals as input. The error estimate is used for computing the target size and, in the case of *PredictiveShift*, for shifting the point of gaze (see figure 4.7). The selection success and the respective target size are logged. We repeat the evaluation cycle with 20 values of ω, weighting the influence of the gaze precision estimate on the target size. The interval range and step size are chosen, starting with $\omega = 0$ (no influence), such that selection rates of models converge to 100%. For all methods, this results in a maximum target size between 300 and 350 cm² (see figure 4.8). We consider $\omega \in \{0 \leq i < 3\}$ with step-size 0.15 for *Feit2dDist* and $\omega \in \{0 \leq i < 20\}$ with step-size 1 for *PredictiveShift* and *PredictiveShift(personal)*.

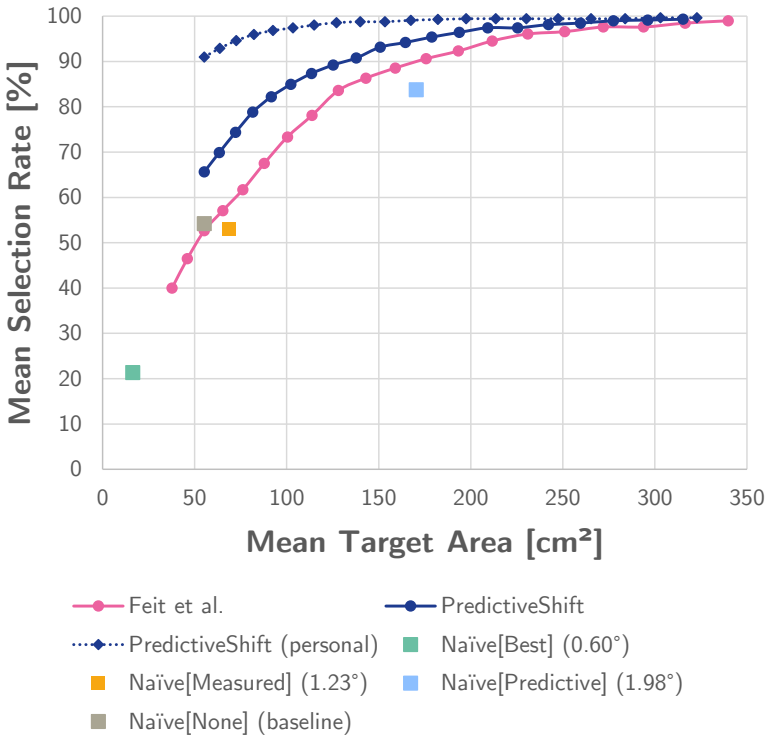


Figure 4.8: Mean selection rate off all methods in relation to the average target sizes. We include the results for varying ω for the new compensation methods.

4.4.2 Results

We average the selection rate and the target area over all on-screen targets and grid positions. Figure 4.8 shows the mean selection rates in relation to their mean target size per method. This allows a direct comparison with our previous results: the four squares represent the offline simulation results of the *Naïve* scaling using different error models, confirming our previous findings (see table 4.1 and figure 4.5). The results of the other compensation methods are plotted as curves due to the variable parameter ω. Tests for statistical significance are conducted using McNemar's test for selection rate and the Wilcoxon signed-rank test for target sizes.

For $\omega = 0$, the compensation method of Feit et al. (2017) achieves a selection rate of 40% with an average target size of 37.74 cm². When increasing ω by two steps to 0.3, it achieves a similar selection rate and target size than the baseline method *Naïve[None]*: 52.65% at 55.27 cm². The selection rate further improves with greater values for ω, but the ratio to the target area decreases (i.e., the slope of the curve is regressive). Hereby, this method reaches a similar selection rate than *Naïve[Predictive]* at $\omega = 1.2$ with 83.59%, but with significantly smaller target sizes of, on average, 128 cm² ($Z = -14.6$,$p < .001$). In the same way, it outperforms *Naïve[Predictive]* with a similar target size (175.7 cm²) in terms of selection rate: 90.6% at $\omega = 1.65$ ($p < .001$).

With $\omega = 0$, *PredictiveShift* generates targets with the same size as the baseline model *Naïve[None]*, which is used for computing its base size s_c (see figure 4.7b). However, shifting the gaze by the predicted directional gaze estimation error increases the selection rate by 11.45% to 65.64% ($p < .001$). Similar to the *Feit2dDist* method, with increasing ω, our novel method achieves higher selection rates with a regressive slope. Particularly for small targets, our method *PredictiveShift* achieves higher selection rates.

The personalized version *PredictiveShift(personal)* achieves the best selection rates and target sizes. For $\omega = 0$, the selection rate is 90.98%, i.e., it significantly improves by 25.34% compared to its corresponding non-personalized version and is 36.79% better than the baseline *Naïve[None]* ($p < .001$). In addition, it exceeds the selection rate of *Naïve[Predictive]*, despite significantly smaller target sizes by 67.57% ($Z = -20.45$,$p < .001$). Like the other approaches, the selection rate increases as ω grows with a regressive slope. For $\omega = 10$, the personalized compensation method achieves a selection rate of 98.01% with target sizes around 165.82 cm^2 that are comparable to *Naïve[Predictive]* ($p < .001$).

4.5 Discussion

In this work, we show that the performance of mobile gaze interaction can be significantly improved if the interface is aware of and can adapt to the inevitable gaze estimation error. The error compensation method *PredictiveShift* consequently achieves better selection rates than our initial *Naïve* scaling approach and excels the method based on Feit et al. (2017) (supports H1.1). The results of our evaluation show that, given a specific selection rate, all new compensation approaches generate smaller targets (supports H4.1). Especially, the user-specific training with our shift-based compensation method, *PredictiveShift(personal)*, increases the performance measures tremendously without increasing the target size (supports H4.1 and H4.2). We conclude that personalized models better explain the variation between users, e.g., covering differences in tracking quality and how they interact using gaze. The decreasing slope can be explained by the fact that the error might be normally distributed. Increasing the selection rate beyond a certain point comes at the cost of larger selection targets that grow quadratically in their area. To put our results into context, we approximately measured the area of common controls of the Windows 10 user interface for the display setting as shown in figure 4.2 with a size of 115.32×86.49 cm: Selecting taskbar icons (19 cm^2) and small tiles in the start menu (32 cm^2) would still be difficult; mid-sized tiles (170 cm^2) and larger controls would achieve high selection rates. Next, we discuss the use cases, advantages, and shortcomings of our approach.

4.5.1 Advantages and Use Cases

The key advantage of our error-aware interface is its application to enhance mobile gaze-based interfaces. The evaluation shows the benefits of using sophisticated compensation methods with error models in selection performance and generated target sizes. To the best of our knowledge, this is the first attempt to apply such a model in error-aware real-time interaction. A direct application of our methods is the extension of the interaction framework by debugging components that visualize the gaze error just in time, and a simulation component that provides a

priori information on the expected gaze estimation error for early user interface prototypes. We suggest two data visualization techniques and describe the simulation capability with our error models similar to Feit et al. (2017), but extend it to mobile interaction settings with varying user positions. Both visualization methods are implemented and informally tested for the purpose of real-time interface analysis.

4.5.1.1 Uncertainty Indicator

This approach comprises the augmentation of the gaze pointer by a transparent ellipse indicating the estimated distribution of the gaze error. The two-dimensional spatial accuracy and spatial precision estimates of any of the presented error models can be used to span an ellipse around a recent fixation position.

4.5.1.2 Heatmap Overlay

In this visualization, a regular grid ($N \times M$) is laid over the display space and mapped back to the world camera space (with an inverse display gaze mapping). Then, the error of the tracking device is estimated for each position. A texture with the resolution $N \times M$ is generated where each pixel corresponds to one point of the grid. For each point, the colors are assigned based on the error value. Finally, this texture is mapped to the display region on the world camera stream, enabling immediate feedback for a considered interface from the user's egocentric perspective.

4.5.1.3 Gaze-interface Design Tool

Simulations have significant potential for interaction designers to optimize interfaces, interaction techniques, and visualizations without testing each variant through actual human studies. Our framework enables the assessment of gaze-based interfaces at the design stage, similar to Feit et al. (2017). For this, each interactive control of an interface can be compared to target sizes generated by our compensation methods for identifying potential issues. The advantage of our approach is that different user positions in a mobile interaction setting with given display positions and sizes can be covered.

4.5.2 Limitations

Despite its novelty in terms of error-awareness, our gaze-based interface has some limitations. We currently restrict the number and type of interface elements to a single button that can be triggered. To showcase an error-aware gaze-based interface, this button expands in size for high-error situations and optionally shifts the gaze to ease selection. Currently, there is no efficient method for collecting required training data, but our personalized compensation approach achieves the best selection rate and target size ratio.

4.6 Conclusion

We introduced an error-aware gaze interaction framework. This framework enables a new class of gaze-based interfaces that are aware of the gaze estimation error. Driven by real-time error estimation, this approach has the potential to outperform state-of-the-art gaze selection methods in terms of selection performance with competing target sizes. We presented a first implementation of the framework and evaluated a naïve target scaling approach with four methods to estimate the gaze error in a user study. Elaborating on our findings, we developed and compared advanced error compensation methods. The results show both the real-time capability as well as the advantages of an error-aware interface, relying on gaze shifting and personalized training with a predictive model in terms of selection performance and target sizes.

Some approaches have been transferred to the open source Pupil Capture platform (Kassner et al., 2014), like estimating the 3D position of the user relative to marker-based surfaces and using the 1-Euro-Filter (Casiez et al., 2012) for smoothing the gaze signal for visualization purposes[2].

We plan to further extend and evaluate the concept of error-aware gaze-based interaction in future work. As a first step, we want to add and compare further mechanics to adapt the interface according to the gaze estimation error. One idea would be to move small objects to low-error regions. This could, for example, help to automatically assign more fine-grained interfaces with small objects to close-by displays and coarse-grained interfaces with larger objects to more distant displays. Error models can also help in assessing non-adaptive gaze-based user interfaces by, e.g., inferring expected miss rates for object selection of all UI elements based on their on-screen position and potential user positions. Further, exploring techniques for a seamless collection of training samples for training new error models or fine-tuning them for new users, e.g., as part of a calibration routine or fully automatic, will be interesting.

[2]See `https://github.com/pupil-labs/pupil/commits?author=MichaelBarz` (accessed on 22 Nov 2024)

Chapter 5

Calibration-free Gaze-based Interaction

Calibration-free gaze-based interaction techniques like systems based on gaze gestures or smooth pursuit eye movements can circumvent the issues caused by gaze estimation errors because, by design, they do not require accurate gaze estimates. We introduce a novel calibration-free interaction technique based on saccadic eye movements for user authentication at public displays. An increasing number of use cases require users to be authenticated prior to interaction with a public display. Typical situations include making purchases, retrieving sensitive information, or for identification. A common class of authentication methods are knowledge-based approaches, e.g., PIN, password, or patterns are entered via touch input on the public screen or via an external keyboard. These methods are prone to residues- or observation-based attacks. Smudges (Aviv et al., 2010) or thermal residues (Abdelrahman et al., 2017) can reveal partial to full information of a PIN or password. Attackers might also observe the input by shoulder surfing attacks (Brudy et al., 2014) or more sophisticated attacks involving cameras (Ye et al., 2017). An alternative to knowledge-based approaches are biometric authentication methods such as fingerprint and iris scans. In this work, we concentrate on knowledge-based authentication. Prior research proposed authentication mechanisms based on different modalities to be more robust against attackers. E.g., Kim et al. (2010) used the pressure signal of touch-based input to overcome shoulder surfing. Particularly, gaze-based methods have been found to be more secure than, e.g., touch-based input (De Luca et al., 2008; Khamis et al., 2018b). Further, it allows hands-free authentication and interaction, which is more hygienic than touch input. This is important due to the high contamination of public displays (Gerba et al., 2016). A major drawback of many gaze-based authentication methods is a mandatory calibration of the eye tracking device (Kumar et al., 2007; Best and Duchowski, 2016). These approaches are unsuitable for public displays because it is time-consuming and perceived as cumbersome (Majaranta and Bulling, 2014). However, interaction methods for public displays shall be designed for immediate usability (Khamis et al., 2015). Calibration-free methods exist, but tend to be slow (Khamis et al., 2018b; De Luca et al., 2007; Cymek et al., 2014; Rajanna et al., 2017) or suffer from high error rates (De Luca et al., 2009; Khamis et al., 2018b).

We implement a novel gaze-based and calibration-free authentication method, *EyeLogin*, that addresses the limitations of prior approaches (Bhatti et al., 2021). Our system uses the direction of saccadic eye movements in a radial interaction design, similar to the system by Best and Duchowski (2016), that facilitates accurate and fast PIN entry (see figure 5.1b). We use a low-cost remote eye tracking sensor that allows broad integration into public displays and spontaneous user interaction. Our method is calibration-free, unlike the system proposed by Best and Duchowski (2016). We implement the state-of-the-art method *CueAuth* as a baseline system, described in Khamis et al. (2018b). In a user study (n=10), we compare both authentication methods by their PIN entry accuracy and time and concerning usability and the perceived workload. In this chapter, we contribute as follows:

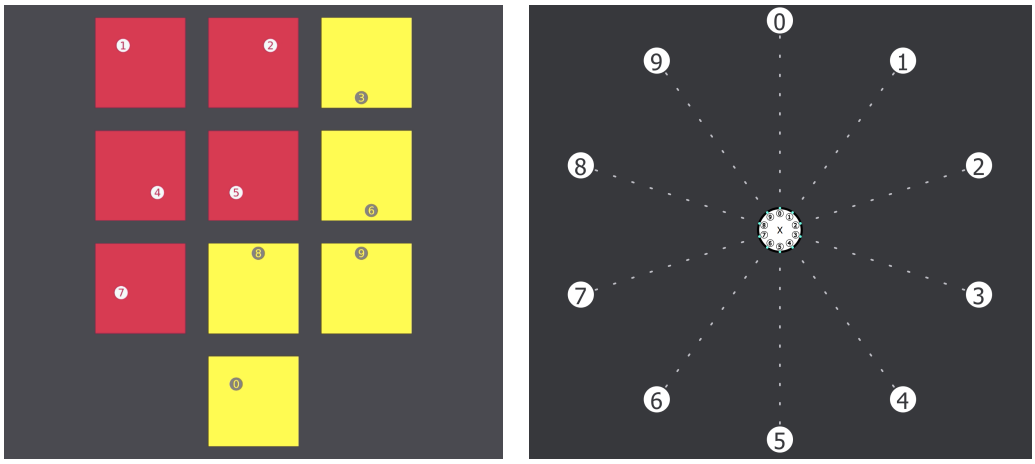
- Section 5.1: We implement two methods for calibration-free gaze-based authentication via PIN-entry: the state-of-the-art method *CueAuth* as a baseline system (Khamis et al., 2018b) and a novel approach based on saccadic eye movements, *EyeLogin*.
- Section 5.2: We conduct a user study (n=10) to compare both methods in terms of accuracy, efficiency, usability, and perceived workload.

5.1 Gaze-based Authentication

Next, we describe the design and implementation of the gaze-based *CueAuth* method (Khamis et al., 2018b) and our novel authentication method *EyeLogin* based on saccadic eye movements. We implement *CueAuth* as a baseline system because it is the most recent calibration-free method that implements the same knowledge-based authentication method, i.e., a four-digit PIN entry for public displays. Khamis et al. (2018b) implemented and compared three authentication methods based on touch, gesture, or gaze input. Unless stated otherwise, we refer to the gaze-based *CueAuth* version.

5.1.1 CueAuth

We implement the *CueAuth* PIN entry method introduced by Khamis et al. (2018b). It matches the smooth pursuit eye movements of a user with the trajectory of moving digits (0-9) in the interface (see figure 5.1a). The interface shows a virtual number pad. Each digit is presented in a small circle that moves with a pre-defined unique trajectory. The trajectories are either linear, circular, or zigzag-shaped. A user must follow the movement of four digits in a row to enter a PIN. The interface provides visual feedback in a separate text view: an asterisk symbol is added when an input is detected. Addressing the known limitations of *CueAuth*, we add a one-second break after the trajectory-based animation ends to allow the user to re-focus. Further, we provide feedback when the matching process begins and ends. The matching begins after the animations of the digits stop (see algorithm 1). We compare the trajectories of the interface controls with the relative eye movements of the same time frame. We calculate the Pearson correlation for two axes (x and y) and average the correlation coefficients in the `Correlate` function. If the mean correlation $c \geq 0.8$, the digit is stored, and the user receives immediate feedback of the match (asterisk). If more than one trajectory reaches the threshold detection threshold of 0.8, we



(a) Interface of *CueAuth* **(b)** Interface of *EyeLogin*

Figure 5.1: We compare two gaze-based authentication methods. (a) *CueAuth* is based on a dial pad layout and smooth pursuit eye movements. (b) *EyeLogin* uses a radial digit pattern and saccadic eye movements for PIN entry.

choose the digit with the higher correlation. We call `Correlate` with two different time windows: [2 s-4 s] and [1.5 s-4 s] that start 2 s or 2.5 s before the animation stops. The digit with the highest correlation coefficient is appended to the stored PIN.

5.1.2 EyeLogin

We propose a novel algorithm for calibration-free authentication based on saccadic eye movements. *EyeLogin* shows the digits 0 to 9 in a radial design (see figure 5.1b), similar to Best and Duchowski (2016). At the center, we present feedback on the progress by adding one asterisk per entered digit. Further, we show miniaturized digits as directional cues for the user to prevent errors. A dashed line connects the inner and outer digits to guide the user's gaze. The user starts the authentication process by pressing the space bar while fixating the center area. This trigger is required to overcome Midas' touch problem inherent in gaze-based interaction. It could be replaced by any trigger in the future, e.g., presence detection in combination with a long fixation or speech-based hotwords known from digital assistants. When the authentication process is started (trigger), the initial gaze position is stored as reference point $gaze\_c$ and provided as input to *EyeLogin* (see algorithm 2). The user enters a digit by fixating on it and returning the focus to the center position. We leverage the quick saccadic eye movement between two fixations to determine the relative direction of the eye movement and to detect the digit of choice: we determine the farthest point max_p from the reference point $gaze\_c$ and calculate the direction vector dir_n with $gaze\_c$ as the origin and max_p as the destination point. The angle between the y-axis and the direction vector allows us to infer the fixated digit: each digit is assigned to a certain sector. Upon detection, the system gives feedback by displaying an additional asterisk in the center area. Showing the feedback at the center region ensures that the user returns its

ALGORITHM 1: CueAuth - PinDetection

```
Function ObserveInput
    while (PIN.size() < 4) do
        OnAnimationStop += CalcDigit(gaze, trajectories);
    end
end
Function CalcDigit (List gaze, List[] trajectories)
    (corr_a, digit_a) = Correlate(gaze, trajectories,2.0);
    (corr_b, digit_b) = Correlate(gaze, trajectories,2.5);
    if (corr_a > 0.8 OR corr_b > 0.8) then
        if (corr_a > corr_b) then
            PIN.Add(digit_a);
        else
            PIN.Add(digit_b);
    end
end
```

focus to this point as the algorithm expects. This process is repeated four times to complete the PIN entry. One limitation is that users might gaze to the next digit before returning to the center area. This would cause an erroneous input because the next digits would be recognized only. This error type rarely occurred in our study.

5.2 Evaluation

We conduct a user study (n=10) to compare our novel authentication method *EyeLogin* to the existing *CueAuth* method. We investigate the effectiveness, efficiency, usability, and perceived workload of both methods in a public display setting. We adopt the experimental design from Khamis et al. (2018b) to ensure comparability with their results. Hence, our study has the same limitation: a consecutive PIN entry is no realistic scenario and might negatively impact our results. However, this should not cause problems due to our within-subjects design. The study is designed as a repeated measures experiment with the *authentication method* as the independent variable. We recruited 10 students (two females and eight males) with normal or corrected to normal vision aged between 25 and 31. One participant with weak vision refrained from wearing eye correction but did not report any problems. Two of the participants had prior experience with eye tracking.

5.2.1 Conditions & Metrics

We investigate the performance and usability of the authentication methods *EyeLogin* and *CueAuth*. For each method, we ask participants to enter 11 PINs in the training phase and 17 PINs in the main phase, totaling 28 PINs per method and user. The instructor vocalizes

ALGORITHM 2: EyeLogin - PinDetection

Function ***ObserveInput*** *(Point $gaze_c$)*
  while *($PIN.size() < 4$)* **do**
    if *(Saccadic_Movement_Recognized())* **then**
      CalcDigit(*saccade*, $gaze_c$);
    end
  end
end
Function ***CalcDigit*** *(List saccade, Point $gaze_c$)*
  max_p = GetFarthestPoint(*saccade*,$gaze_c$);
  dir_n = CalculateDirection($gaze_c$, max_p);
  angle = CalculateAngle_To_Y_Axis(dir_n);
  digit = CalculateDigit(*angle*);
  PIN.Add(*digit*);
end

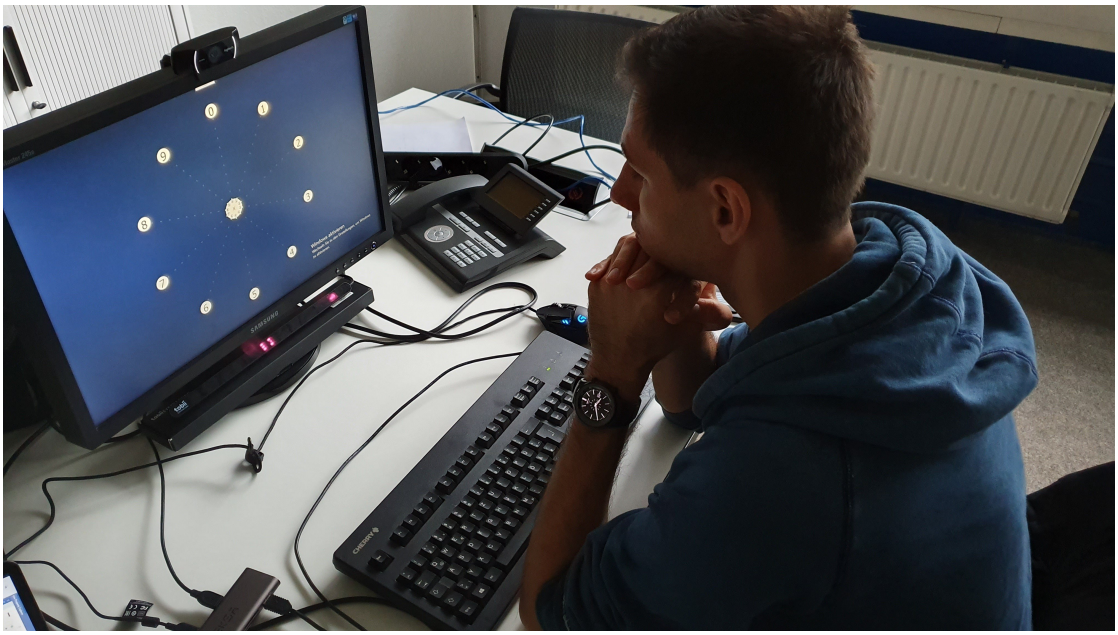


Figure 5.2: Setup of the user study.

the randomly selected four-digit PIN before the participant starts the authentication procedure by pressing the space key. They receive automatic feedback about the progress, as described above, but not whether a digit or the complete PIN was recognized correctly. To measure the performance of both methods, we use the following metrics: the PIN entry time, the accuracy in entering a PIN, the System Usability Scale (SUS) as a usability measure, and the NASA TLX score as a measure for the perceived workload. The PIN entry time is the time between pressing the space bar and recognizing the fourth digit. The PIN accuracy is the average number of correct PIN entries. A PIN is considered correct if all digits are entered correctly. A false entry is counted if one or more digits are incorrect.

5.2.2 Procedure

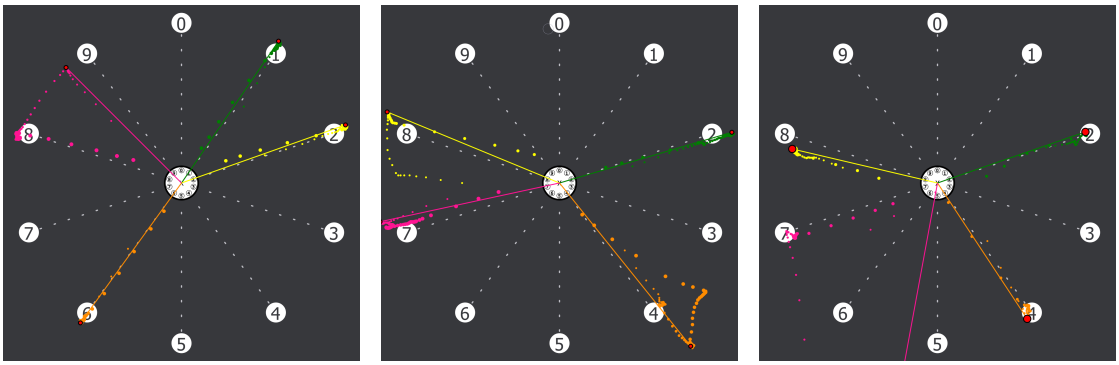
First, each participant signs an informed consent form. Then, the instructor introduces the topic and procedure of the study. The instructor presents one of the considered authentication methods in a counterbalanced order by demonstrating the interface and explaining how a digit or password is entered. In the training phase, each participant enters 11 PINs. Participants can familiarise themselves with the interaction method by entering three simple PINs and eight random PINs. The instructor provides additional support if major problems are detected. In the main phase, the participant enters 17 PINs. After completing all PINs, the participant fills in a NASA TLX form (Hart and Staveland, 1988) for assessing the perceived workload and a System Usability Scale (SUS) questionnaire (Brooke, 1996) for measuring usability. This cycle is repeated with the remaining authentication method. At the end of the study, participants fill in a questionnaire including demographic questions and open-ended questions on their experience with each authentication method. The order of the authentication methods is counterbalanced to avoid ordering effects.

5.2.3 Apparatus

A 27-inch widescreen monitor with a resolution of 1920×1080 pixels is used to display the authentication interfaces. We use the binocular Tobii 4C remote eye tracker (Tobii Help, 2016) with a 60 Hz sampling rate, which is attached below the screen (see figure 5.2). Prior to all sessions, the eye tracker is calibrated once for the study instructor: we use the same calibration for all participants. Participants are seated approximately 60 cm in front of the display. A keyboard is provided to start the authentication trials. For analysis, we store the participant ID, the timestamped eye movements, and synchronized movements of all smooth pursuits stimuli (*CueAuth* only) for every PIN entry attempt. Further, we store the recognized PIN, the correct PIN, and the answers to the questionnaire and the NASA TLX items.

5.2.4 Hypotheses

We hypothesize that users are more accurate in entering PINs with *EyeLogin* compared to *CueAuth* because directed saccadic movements are seemingly less error-prone than more complex smooth pursuits (H1). In addition, we expect that authentication is faster using our saccade-based method *EyeLogin* than using *CueAuth*, which is bound to long animation times (H2).



(a) Incorrect PIN (expected: 1628; detected: 1629)

(b) Correct PIN (expected: 2487)

(c) Incorrect PIN (expected: 2487; detected: 2485)

Figure 5.3: Visualizations of the raw gaze signal of individual participants from our study, including three trials using *EyeLogin*. Trials a) and c) result in a wrong PIN entry. Trial b) shows the sequence of a successful trial. The colors indicate the input order (green, orange, yellow, pink), the dot size relates to the order of gaze samples (increasing radius corresponds to increasing timestamps), and the red dot marks the point max_p from our algorithm.

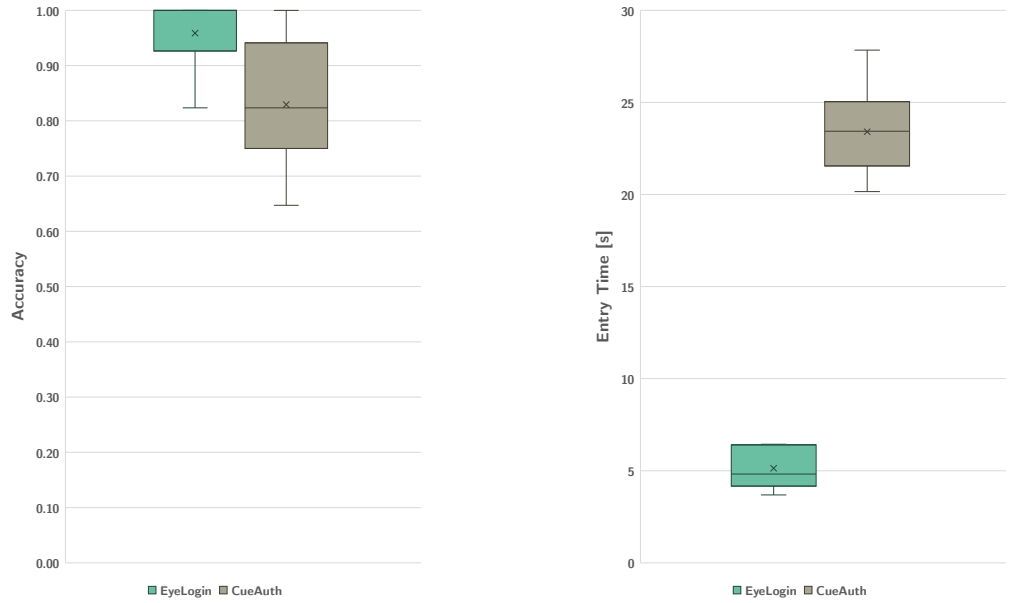
Further, we expect that the gains in effectiveness (accuracy) and efficiency (time) have no negative impact on usability and the perceived workload (H3).

5.2.5 Results

For both methods, we observe the accuracy, the entry time, the NASA TLX, and the SUS score for entering PINs. We report the results of the main phase (17 PINs). We found no significant differences between our training and main phase measurements. We use the 2-tailed paired samples t-test with an alpha level of 5% in SPSS to test for statistical significance. The Shapiro-Wilk test is used to check whether the differences of the paired samples are from a normal distribution, i.e., to check that no assumption of the dependent t-test is violated. We also checked whether the order of methods affects our dependent variables. We found no significant differences using an independent t-test and the order as a between-groups factor ($p > .05$).

5.2.5.1 Accuracy

Using our implementation of *CueAuth*, the users achieve a mean accuracy of 82.94% ($SD = 11.58$). This is close to the results from Khamis et al. (2018b), who reported 82.72% ($SD = 38.53$). On average, the accuracy is lower during the training phase ($M = 71.25\%, SD = 21.28$), but not significantly ($t(9) = -1.491, p = .17$). For our proposed method *EyeLogin*, we observe an accuracy of 95.88% ($SD = 6.23$), which is 12.94% better than the *CueAuth*-baseline (see figure 5.4a). This difference is statistically significant with $t(9) = 3.18$, $p = .012$. In addition, our gaze-based method performs better than the best method of Khamis et al. (2018b) based on touch interaction ($M = 93.38\%, SD = 26.05$).



(a) Accuracy in PIN entry

(b) PIN Entry time [s]

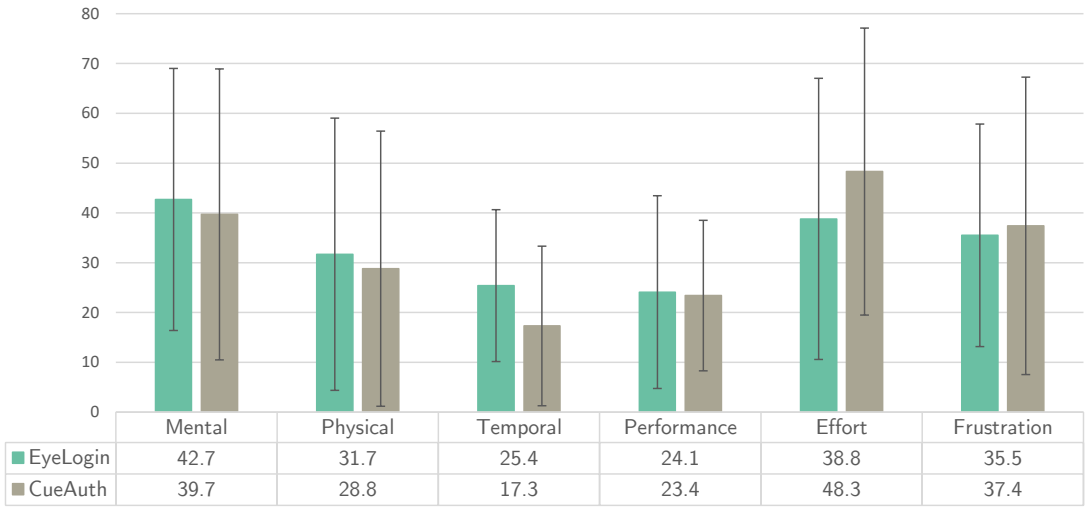
Figure 5.4: Results from our user study (n=10) including a box-plot for the accuracy of PIN entry (a) and for the PIN entry time (b) for the main phases of *EyeLogin* and *CueAuth*.

5.2.5.2 Entry Time

On average, we measure entry times of 23.41 s ($SD = 2.28$) for *CueAuth*, which is similar to the result of 26.35 s reported in the literature (Khamis et al., 2018b). Their reported standard deviation of 22.09 is much higher than ours. Using *EyeLogin*, we observe average pin entry times of 5.12 s ($SD = 1.09$). The absolute time saving of 18.28 s compared to the baseline (see figure 5.4b) is statistically significant ($t(9) = 24.063, p < .001$). The touch-based method from Khamis et al. (2018b) is reported to be the fastest and is, with an average of 3.73 s ($SD = 1.07$), slightly faster than our proposed gaze-based approach.

5.2.5.3 Perceived Workload and Usability

We use the NASA TLX questionnaire to evaluate the perceived workload as suggested by Khamis et al. (2018b). The mean scores for all dimensions of the test are reported in figure 5.5. None of the differences are significant as determined by a paired samples t-test per dimension ($df = 9; p > .05$). Also, we ask the participants to fill in a SUS questionnaire, which results in a subjective usability score for both methods. We receive an average score of 66.5 ($SD = 18.72$) for *CueAuth* and 75.75 ($SD = 15.28$) for *EyeLogin* (higher is better). However, the difference of 9.25 points is not statistically significant ($t(9) = -1.075, p = .31$).



	Mental	Physical	Temporal	Performance	Effort	Frustration
EyeLogin	42.7	31.7	25.4	24.1	38.8	35.5
CueAuth	39.7	28.8	17.3	23.4	48.3	37.4

Figure 5.5: Bar chart diagram visualizing the NASA TLX results (mean ± SD) for the main phases of *EyeLogin* (blue) and *CueAuth* (yellow, dotted).

5.2.5.4 Qualitative Feedback

We collect qualitative feedback via open-ended questions asking participants to note the pros and cons of each method and to provide suggestions for improvements. Analyzing the answers, we find that *EyeLogin* is perceived as fast (7/10) and easy to use (7/10). Three participants criticized that blinks are likely to cause errors during pin entry. For *CueAuth*, participants stated the advantages that the layout is familiar (4/10) and easy to use (2/10). The participants perceived *CueAuth* as slow (7/10) and tiring (4/10). Two participants criticized that the system was not sensitive enough to recognize their input.

5.3 Discussion

The results of our evaluation show that our authentication method *EyeLogin* is significantly more accurate than the baseline system *CueAuth*. Users succeed in entering a PIN in 95.88% of all trials, which is 12.94% better than the baseline and confirms H1. In addition, our gaze-based method achieved a similar accuracy to the touch-based version of *CueAuth* as reported in Khamis et al. (2018b). Further, our method performs more accurately than the method by Best and Duchowski (2016) with 71.16% accuracy, which is also based on a circular design but requires user calibration. The PIN entry times for *EyeLogin* are tremendously lower than for *CueAuth* (significant). On average, users need 5.12 s to enter a four-digit PIN which is 18.28 s faster than measured for the baseline *CueAuth* and confirms H2. The measured PIN entry times for our implementation of *CueAuth* are similar to the reported results by Khamis et al. (2018b). Further, the entry times for *EyeLogin* are comparable to their touch-based version (3.73 s on average). We did not assess whether *EyeLogin* achieves the same level of security as *CueAuth*. Only 0.05% of all attacks were successful if the eyes and the display content were visible. We used

the SUS and the NASA TLX questionnaires to measure the usability and the perceived workload of both authentication methods. The results do not reveal significant differences between the two considered methods, suggesting we can confirm H3. We observe a higher average SUS for *EyeLogin* (75.75) than for *CueAuth* (66.5). This might indicate that our authentication method yields better usability than the baseline system. For comparison, other works using the SUS questionnaire achieve, on average, a score of 69.5 (n=273) (Bangor et al., 2009).

5.3.1 Limitations and Future Work

EyeLogin enables fast and reliable input for authentication at public displays. It achieves the same level of performance as more common touch-based methods. However, a few limitations remain, including a required start trigger to overcome the Midas touch problem, some error types that cause avoidable authentication fails, and potential vulnerabilities to camera-based attacks. We address each of these limitations and suggest how they could be solved.

5.3.1.1 Midas Touch Problem

Our system requires users to press the space bar to capture the reference gaze position $gaze_c$ and start the authentication. Using an additional modality for disambiguating the gaze-based input is common practice (Qvarfordt, 2017; Oviatt et al., 2017b). However, this trigger should be replaced with more suitable alternatives for public displays, like touching a button. An alternative could be speech-based input: An instruction can be shown at the central area of the user interface, asking the user to start the authentication by vocalizing a trigger word. A pure gaze-based method can be realized using a dwell-based trigger to start the authentication. The presence of a user can be detected by the presence or absence of gaze data from the eye tracking sensor or using mmWave radar-based presence sensors (Cui and Dahnoun, 2021). Other approaches include leveraging smooth pursuit eye movements like the CueAuth baseline, which was significantly slower (Khamis et al., 2018b), or gesture-based authentication using gaze-enabled lock patterns, similar to the touch-based version known from Android devices, as shown in figure 5.6.

5.3.1.2 Error Types

We observe two common error types for *EyeLogin* that cause a wrong PIN entry. Figure 5.3a) shows the raw gaze signal of a user who moved their gaze to the wrong digit (9), corrects the gaze position (8), and returns to the center. However, *EyeLogin* detects 9 as input resulting in a wrong digit sequence. Figure 5.3b shows a similar case, but the correction (for 4 and 8) is done earlier, and *EyeLogin* detects the correct sequence. Figure 5.3c shows a trial that failed due to a blink before the fixation to the last digit: 7. The gaze signal is highly imprecise just before the blink, causing the algorithm to choose the wrong digit (5). A further limitation is that users might turn their gaze to the next digit before returning to the center area, which is unsupported. Future extensions to improve the system should consider a less strict interpretation of users' gaze input. Concretely, the system should allow to directly jump from one digit to the next without fixating the center region in between. Further, the system should be more robust in cases of data loss due to blinks. A blink detection with a corresponding signal correction before and

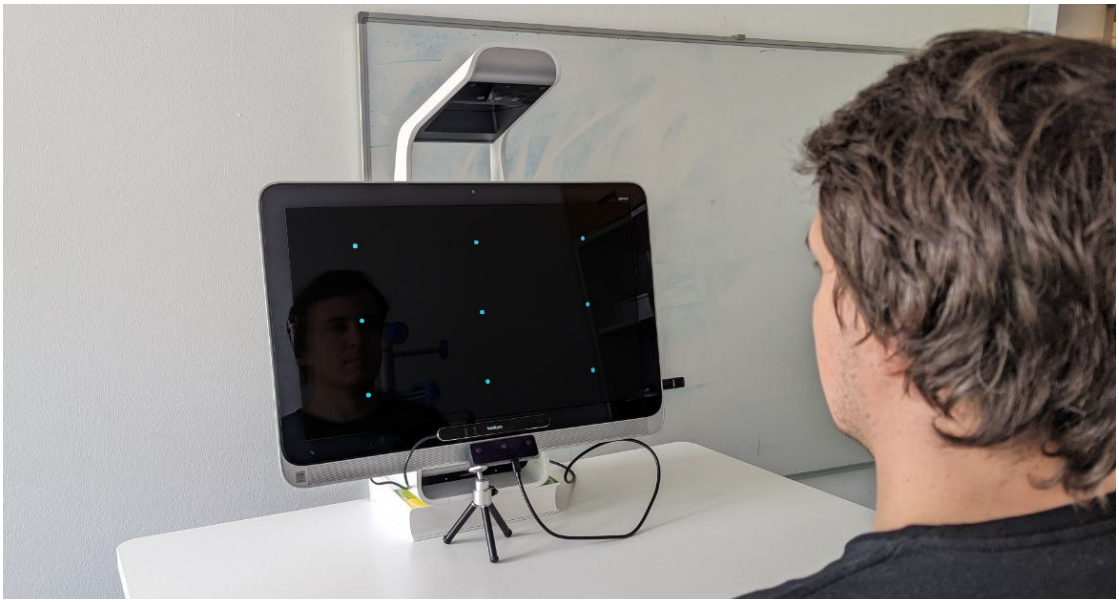


Figure 5.6: Prototype of a gaze-based lock pattern interface for authentication at public displays.

after blink events could reduce false positive detections. Eventually, users should be enabled to recover from errors more easily. They should be able to proactively clear a partial entry if they recognize a wrong input themselves.

5.3.1.3 Camera-based Attacks

EyeLogin is robust against traditional shoulder-surfing attacks because an attacker must observe the display and the eyes of a user during PIN entry. More sophisticated attackers might attach a camera to the public display and infer the password from a video stream that captures the user's face and eye movements. The collected video feeds and display contents from our study can be used to evaluate the vulnerability of our PIN entry method as suggested by Khamis et al. (2018b). Randomizing the arrangement of digits for *EyeLogin* could further increase the robustness against attackers in case of vulnerabilities. All our participants perceived this as more secure. Better security through randomly arranged digits probably needs to be traded off against usability and entry time.

5.4 Conclusion

We presented a calibration-free and gaze-based authentication method for public displays. In a user study, we could show that our method *EyeLogin*, which leverages saccadic eye movements, performs significantly faster and significantly more accurate than *CueAuth*, a state-of-the-art gaze-based authentication system from the literature (Khamis et al., 2018b). With this work, we presented the first calibration-free authentication method using gaze that is on par with less secure input modalities such as touch- and gesture-based input in terms of effectiveness and efficiency. To further improve our system, future research should investigate how to avoid erroneous inputs and ways to recover from errors if they happen. Further, future systems should aim to overcome the Midas touch problem. We identified gaze-enabled lock patterns as a promising direction. In addition, it would be interesting to investigate whether the gaze signal from a calibration-free authentication procedure, which naturally occurs at the beginning of a user-system interaction, can be used to calibrate an eye tracker for more fine-grained gaze-based interaction, including selection and manipulation of objects. It could also be investigated whether the input of a calibration routine could be used to calibrate an error model for error-aware gaze-based interaction as described in chapter 4.

Part III

Human Gaze as a Passive Input Modality

Chapter 6

Inferring Visual Search Targets

Human gaze behavior depends on the task in which a user is currently engaged (Yarbus, 1967; DeAngelus and Pelz, 2009) and visual search is a task in which humans aim at identifying a search target among other objects using their visual perception. Predicting the target of a visual search with computational models and the overt gaze signal as input is commonly referred to as search target inference (Borji et al., 2015; Sattar et al., 2015, 2017a). This provides implicit insight into a user's intentions and allows external observers or intelligent user interfaces to make predictions about the ongoing activities (Flanagan and Johansson, 2003; Rotman et al., 2006; Bader and Beyerer, 2013; Haji-Abolhassani and Clark, 2014; Akkil and Isokoski, 2016; Rothkopf et al., 2016). The ability to infer visual search targets can help to construct and improve intelligent user interfaces in many sub-tasks, e.g., gaze-supported media retrieval (Sattar et al., 2015), anticipatory user interaction with mobile phones (Steil et al., 2018b), or collaborative robots (Huang and Mutlu, 2016). Further, it allows for more fine-grained activity recognition for situation-aware assistance of mentally impaired people (Toyama and Sonntag, 2015), other applications of cognitive assistance (Sonntag, 2015), and to model users in complex multimodal communication (Sonntag, 2012; Oviatt et al., 2017a, 2018), to name a few. Recent work investigates algorithmic principles for search target inference on generated dot-like patterns (Borji et al., 2015), target class prediction using *Bag of Visual Words* (Sattar et al., 2015) on image collages, and target category prediction using a combination of gaze information and CNN-based features (Sattar et al., 2017a). A common disadvantage is that these approaches are constrained to artificial data sources with a specific format: dot-like patterns and image collages for which target regions are clearly defined and which are restricted to a certain amount of classes. Another problem is that current methods only achieve low accuracy.

We aim to address this issue by implementing a new encoding method for scanpaths, the *Bag of Deep Visual Words* encoding. It extends the *Bag of Visual Words* approach (Sattar et al., 2015) but uses a pre-trained CNN model's activations to encode fixations to visual stimuli. Another challenge lies in moving towards ubiquitous and mobile interaction settings which comprise natural scenes for which target classes and respective target regions are not known in advance. With natural scenes, we refer to naturally occurring everyday scenes in indoor and outdoor environments, e.g., offices and kitchens as indoor settings and a street view as an outdoor setting.

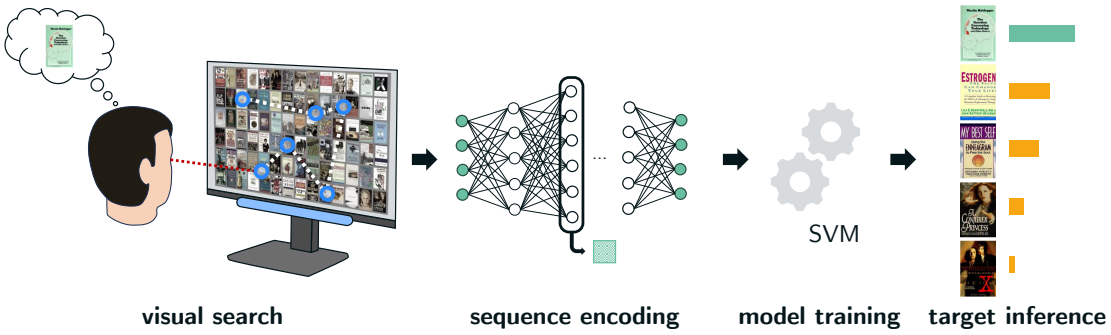


Figure 6.1: Search target inference takes a fixation sequence from a visual search as input for target prediction. The pipeline we implement encodes sequences using a *Bag of Words* approach with features from a CNN for model training and inference.

For this purpose, we propose to infer the image segment class that includes the search target instead of directly inferring the search target. Our approach is based on a pre-trained model of the SegNet image segmentation method (Badrinarayanan et al., 2017): we segment each visual stimulus, encode scanpaths as a *Histograms of Fixated Image Segments*, and infer the segment class that includes the visual search target. In this chapter, we contribute as follows:

- Section 6.1: We implement the novel *Bag of Deep Visual Words* encoding for scanpaths from visual search trials (Stauden et al., 2018). We compare its performance in inferring visual search targets from a constrained search target inference task (*Amazon book cover* dataset) to the *Bag of Visual Words* baseline (Sattar et al., 2015).
- Section 6.2: We develop and evaluate a novel method for encoding visual search scanpaths based on image segments, the *Histograms of Fixated Image Segments* encoding (Barz et al., 2020b). We evaluate its performance using the publicly available visual search dataset VIU that includes natural indoor and outdoor scenes (Koehler et al., 2014).

6.1 Target Inference in Constrained Settings

We extend the idea of using a *Bag of Visual Words* (`BoVW`) for classifying search targets (Sattar et al., 2015): we implement a *Bag of Deep Visual Words* model (*BoDVW*), based on image representations from a pre-trained CNN, and investigate its impact on the estimation performance of search target inference (see figure 6.1). We reproduce the results of Sattar et al. (2015) by re-implementing their method as a baseline and evaluate our novel feature extraction approach using their published *Amazon book cover* dataset (Sattar et al., 2015).

6.1.1 Sequence Encoding Methods

The Bag of Words algorithm is a vectorization method for encoding sequential data to histogram representations. The encoding is commonly used in natural language processing for, e.g., document classification (Goldberg, 2017), and was extended to the *Bag of Visual Words* method

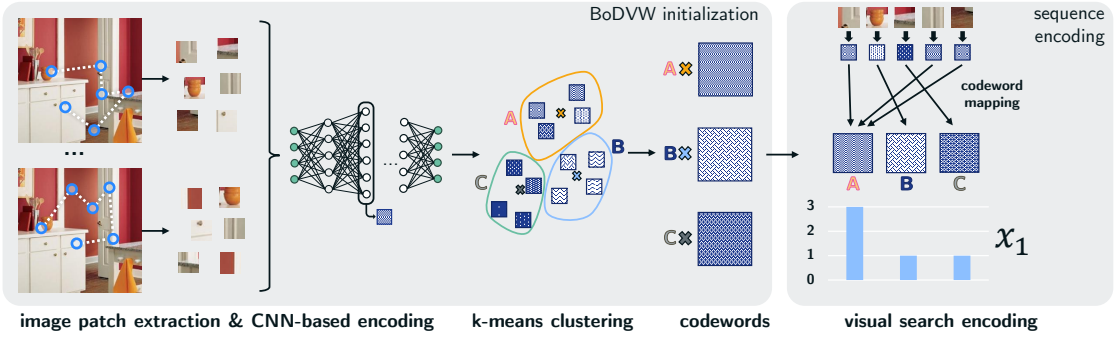


Figure 6.2: Image patches from fixation histories are extracted and encoded using a pre-trained CNN to initialize the *Bag of Deep Visual Words* encoding. The activations from a certain hidden layer are used for a k-means clustering that identifies deep codewords (cluster centers) to encode visual search sequences into a vector x_1 with a fixed size for model training.

in the computer vision domain for, e.g., scene classification (Yang et al., 2007). Bag of Word encodings are initialized with a set of k low-level feature vectors (=codewords) with a fixed size. The k codewords represent meaningful higher-level features for the underlying data and task. The algorithm for identifying these codewords is an essential part of the approach and influences the performance of classifiers that use Bag of Word histograms as input. Each sample is assigned to the most similar codeword for encoding a sequence, resulting in a codeword histogram of size k. We implement two methods based on this concept: a `BoVW` baseline similar to Sattar et al. (2015) and the CNN-based `BoDVW` encoding.

6.1.1.1 Bag of Visual Words

Sattar et al. (2015) were the first to use a *Bag of Visual Words* encoding for search target inference. They encode scanpaths of visual search trials on image collages, e.g., using their publicly available *Amazon book cover* dataset that includes fixation sequences of six participants. They trained a multi-class SVM that predicts the search target from a set of five alternative covers using the encoded histories as input. We re-implement their algorithm for search target inference as a baseline, including the `BoVW` encoding and the SVM target classification. Following their descriptions, we implement methods for image patch extraction from fixation sequences, a `BoVW` initialization for extracting codewords from these patches, and the histogram generation for a certain sequence. We test our algorithms using their *Amazon book cover* dataset.

6.1.1.2 Bag of Deep Visual Words

Our *Bag of Deep Visual Words* approach extends the *Bag of Visual Words* method by Sattar et al. (2015): we encode RGB patches using a CNN before codeword generation and mapping (see figure 6.2). The `BoDVW` method includes components for image patch extraction from fixation sequences, a `BoDVW` initialization for extracting codewords from these patches, and the codeword mapping for histogram generation. Figure 6.2 shows the processes involved in codeword initialization and sequence encoding. The *image patch extraction & CNN-based encoding* is required

for the initialization and the sequence encoding process. We crop squared image patches from a static search image centered around each fixation of a sequence. Each patch is split into nine sub-patches with equal edge length l_e. The eight outer patches shall balance inaccuracies inherent in eye tracking (Sattar et al., 2015). This is particularly important for mobile settings where the gaze-estimation error can be substantial (see chapter 4). All extracted image patches of a visual search sequence are fed to the publicly available AlexNet model pre-trained with the ImageNet dataset (Russakovsky et al., 2015) for image classification. The flattened activation tensor of a specified hidden layer is used as a low-level feature vector for initialization and sequence encoding. We consider the layers `conv1`, `pool2`, `conv4`, `pool5`, `fc6` and `fc8` which represent different stages of the network. The *codeword initialization* is done using a k-means clustering with the flattened activation tensors of all fixation patches contained in a training set as input. The resulting k cluster centers (mean of vectors) serve as the deep visual codewords. For *encoding a fixation sequence* – a variable number of fixations from a single visual search on a static scene image – we extract image patches and activation tensors as described above and map them to our k codewords using the Euclidean distance measure. Each patch is assigned to the codeword with the smallest distance. Finally, we generate a histogram of length k that encodes the frequency of matches to each of the k deep visual codewords. The idea of this encoding is that participants fixate on top-down features related to the target more frequently than on others. E.g., when searching for a particular pen, humans might predominantly fixate on other pens, other pen-shaped objects, or common locations like a pen holder. We hypothesize that the k-means clustering can find deep visual codewords that represent such search strategies and allow classification of the search target of the user.

6.1.2 Experiment

We conduct a simulation experiment to compare the performance in predicting the search target of a visual search using our re-implementation of Sattar et al. (2015). We compare the prediction accuracy using their `BoVW` encoding to our novel `BoDVW` encoding. We closely follow the evaluation procedure of Sattar et al. (2015) for reproducing their original results using the *Amazon book cover* dataset. For this, fixations of a visual search trial are encoded for model training and target inference. This includes fixations on the target after it has been found. However, this conflicts with the goal of inferring the search target (Zelinsky et al., 2013; Borji et al., 2015). Therefore, we exclude all fixations at the tail of the signal (target fixations) and repeat the experiment, keeping all other parameters constant.

6.1.2.1 Dataset

Sattar et al. (2015) published a dataset containing eye tracking data of participants performing a search task. They arranged 84 (6×14) different book covers from Amazon in collages as visual stimuli. Six participants were asked to find a specific target cover per collage within 20 seconds after it was displayed for a maximum of 10 seconds. Fixations were recorded for 100 randomly generated collages in which the target cover appeared exactly once and was taken from a fixed set of five covers. Participants were asked to press a key as fast as possible after they found the target. We manually annotated each collage with a bounding box for the target cover.

6.1.2.2 Conditions and Procedure

In our experiment, we compare the target prediction accuracy using the `BoVW` method against our `BoDVW` encoding (using different layers). For the `BoDVW` approaches, we train multiple models, each using a different neural network layer for image patch encoding as stated in section 6.1.1.2. First, we use the *Amazon book cover* dataset with all available fixations for training and inference as proposed in (Sattar et al., 2015). Second, we repeat the experiment without the target fixations at the end of the signal. Using a train set, we initialize the respective `BoW` method for each condition. We encode the fixation histories (with or without target fixations) and train a support vector machine to classify the output label. The codeword initialization and model training are performed separately for each user (within-user condition). This procedure yielded the best results in Sattar et al. (2015). We extract patches around all fixations in the train set to initialize the codewords for both approaches. We crop squared fixation patches with an edge length of 80 px and generate $k = 60$ codewords. We train a One-vs-All multiclass SVM with $\lambda = 0.001$ for L1-regularization and feature normalization using Microsoft's Azure Machine Learning Studio[1]. We measure the prediction accuracy using a held-out test set, as specified by Sattar et al. (2015) (balanced 50/50 split per user).

6.1.2.3 Hypotheses

We hypothesize that using our *BoVW* implementation, we can reproduce the prediction accuracy of Sattar et al. (2015) (H1.1) and that our `BoDVW` encoding improves the target prediction accuracy concerning the *Amazon book cover* dataset (H1.2). Further, we expect a severe performance drop when excluding target fixations, i.e., when using the filtered *Amazon book cover* dataset (H2.1). In contrast, the `BoDVW` encoding should still perform better than the `BoVW` method (H2.2).

6.1.2.4 Results

Averaged over all users, our `BoVW` re-implementation of the method of Sattar et al. (2015) achieved a prediction accuracy of 70.67% (20% chance) for search target inference on their *Amazon book cover* dataset with target fixations. We could reproduce their findings, even without an exhaustive parameter optimization. Concerning our *Bag of Deep Visual Words* encoding, applied in the same setting, we observe higher accuracies for all layers. The `fc6` layer performed best with an accuracy of 85.33% (see figure 6.3), which is 14.66% better compared to the baseline. When excluding the target fixations at the tail of the visual search history, the prediction accuracy of both approaches decreases: the `BoVW` implementation achieves an accuracy of 35.96% and our novel `BoDVW` encoding achieves a prediction accuracy of 43.56% using the `fc8` layer. In this setting, the `fc8` layer yields better results than the `fc6` layer with 38.26% (see figure 6.3).

6.1.3 Discussion

Our implementation of the `BoVW`-based search target inference algorithm introduced by Sattar et al. (2015) achieves, with a prediction accuracy of 70.67%, a comparable performance than

[1] `https://azure.microsoft.com/en-US/products/machine-learning` (accessed on 12 Dec 2024)

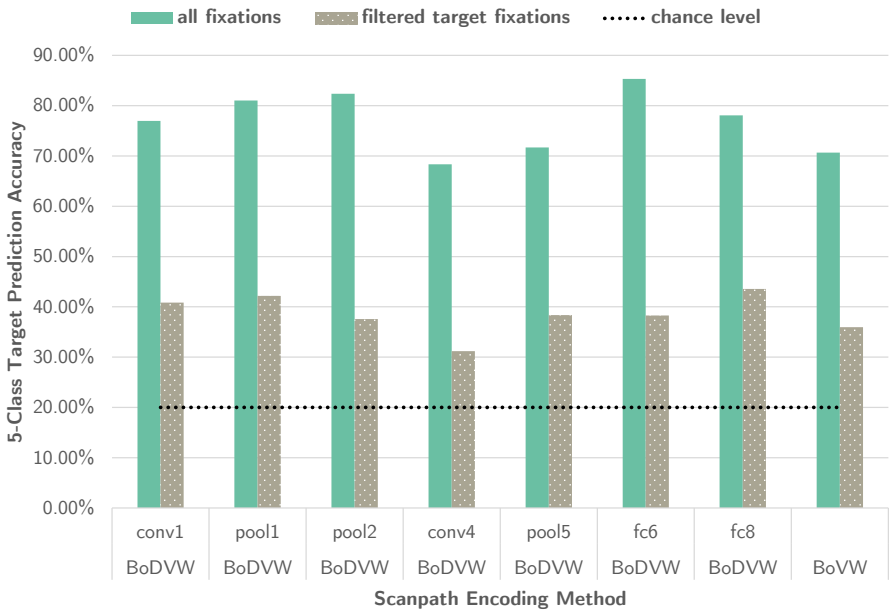


Figure 6.3: Search target inference accuracy of 5-class SVM models using the `BoDVW` encoding with different layers and the `BoVW` encoding (baseline) on complete fixation sequences and filtered fixation sequences.

stated by the authors, for the same settings (confirms H1.1). Our novel `BoDVW` encoding achieves an improvement of 14.66% with the `fc6` layer: an SVM can better distinguish between classes when using CNN features which suggests that H1.2 is correct. In the second part of our experiment, we observed a severe drop in prediction accuracy for both approaches (confirms H2.1). A probable reason is that fixation patches at the end of the search history, which show the target object, greatly impact the prediction performance: the task is simplified to an image comparison. The RGB-based codewords still yield a prediction accuracy above the chance level (20%). Our `BoDVW` approach performs 7.6% better than this baseline with the `fc6` layer (improvement of 21.13%), which suggests that H2.2 is correct. Excluding the fixations on the target is particularly important for investigating methods for search target inference due to the introduced bias. Hence, the procedure and results of the second part of our experiment should be used as a reference for future investigations.

6.2 Target Inference in Natural Interaction Settings

We present a new approach for search target inference that overcomes the limitation of constrained target classes. This enables applications in natural interaction scenarios. We propose a novel search target inference method based on the neural image segmentation model SegNet (Badrinarayanan et al., 2017). Instead of estimating the object class of the search target, we predict the segment class that contains the search target (see figure 6.4). We use the histogram

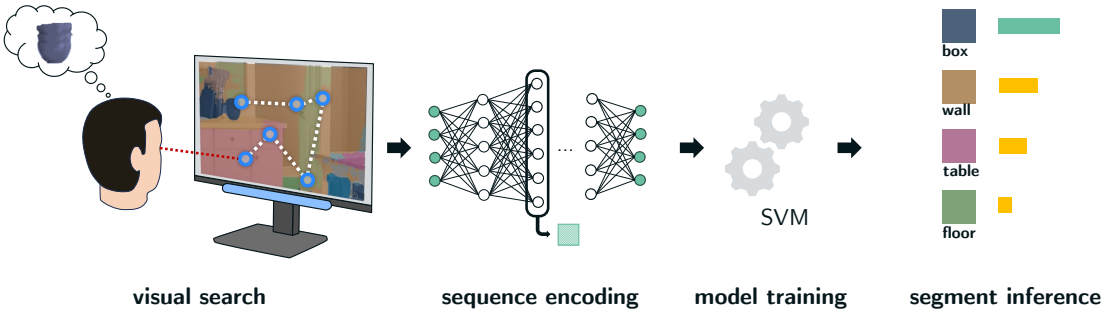


Figure 6.4: Search target inference takes a fixation sequence from a visual search as input for target prediction. We predict the segment class that contains the user's search target to enable applications in ubiquitous settings and natural scenes. Our pipeline encodes sequences using a *Bag of Words* approach with features from pre-trained CNNs for model training (SVM) and inferencing target segment classes.

of segment classes fixated during a visual search as a novel encoding of fixation sequences. This enables the localization of search targets in environments covered by the data used for training SegNet. We evaluate our approach using a subset of the VIU dataset by Koehler et al. (2014): we manually annotated the static scene images with bounding boxes for the ground truth region and automatically extracted image segments using SegNet. In total, we compare three different methods for encoding scanpaths that serve as input to our inference model: (1) we use the *Bag of Deep Visual Words* (`BoDVW`) algorithm as a baseline, (2) we implement and evaluate the *Histogram of Fixated Image Segments* (`HoFIS`), and (3) we include a combined encoding using `BoDVW` and the `HoFIS` encoding. We assume that the histogram of fixated segment classes encodes relevant semantic cues of the scene, which can increase the robustness in natural environments. All methods consider fixation sequences of participants performing a visual search as input and, opposite to the literature, the target segment class as the output label. The image patches are cropped from the search image around each fixation of a visual search with a pre-defined size and get encoded using one of the described methods. The resulting feature vectors are used to train a machine learning model (SVM) to infer the target segment class, including the search target.

6.2.1 Sequence Encoding Methods

Visual search is a perceptual task in which humans aim to identify a search target among other objects in a visual scene. A visual search is performed as a series of fixations guided by visual cues and the user's experiences. The fixation sequence can be observed with eye tracking technology. A scene with corresponding search tasks and the resulting fixation sequence are typically used to investigate how the human brain perceives visual stimuli. Search target inference subsumes methods and algorithms that aim to forecast the search target of a visual search. The idea is that the subtle visual cues that guide the human search process can be modeled, e.g., using machine learning techniques.

Our approach to search target inference includes two major components that need to be trained and applied subsequently: the visual search encoding (sequence encoding) and the target

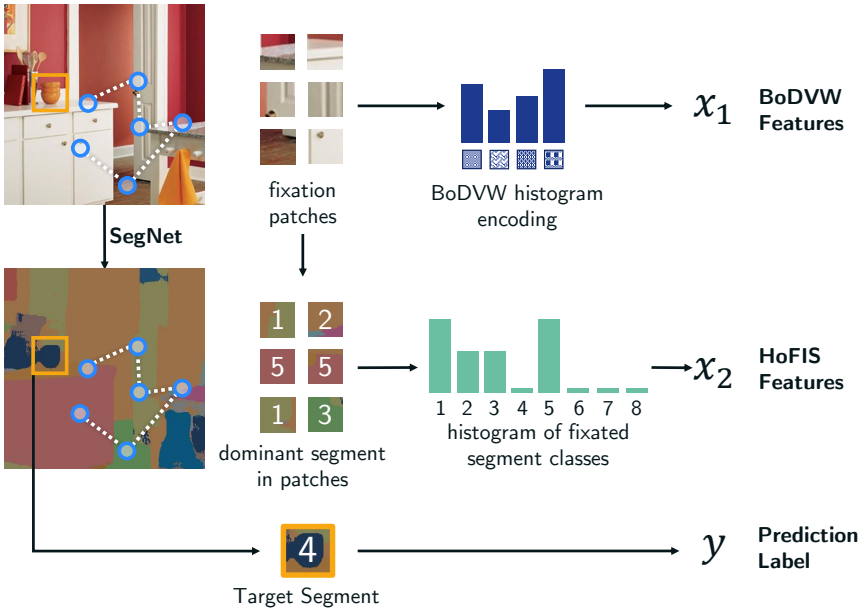


Figure 6.5: The `HoFIS` encoding is based on image segments: scanpaths are encoded as the sequence of fixated image segments. We compare it to the `BoDVW` encoding.

classification module that is based on these encodings (see figure 6.4). We implement three sequence encodings based on fixated image regions: the *Bag of Deep Visual Words* method (see section 6.1.1.2), a novel encoding based on fixated image segments, and a combined version. The module for target classification uses a support vector machine (SVM) similar to Sattar et al. (2015) and our approach presented in section 6.1. These approaches are constrained to a fixed set of target classes which strongly restrict the utility in more realistic settings that comprise natural scenes. Therefore, we suggest an extension based on the neural image segmentation model SegNet (Badrinarayanan et al., 2017). The SegNet model[2] is trained on the SUN RGB-D dataset (Song et al., 2015) which includes 10,000 images of indoor scenes annotated for object detection, classification, and segmentation. Instead of inferring the exact search target, our segmentation-based approach aims to predict the segment class in which the target object is located (see figure 6.5). At first sight, this might be a disadvantage because the target area cannot be inferred with the same spatial precision. But, it comes with the great advantage that search target inference can be used with various natural scenes. Similar to Sattar et al. (2015) and our previous approach presented in section 6.1, we train multi-class SVM models using the sequence encodings as input but the segmentation classes as output labels. Next, we describe the approaches considered for encoding the visual search sequences. The training procedure we use for all methods is described in detail in the experiment section.

[2] `https://github.com/alexgkendall/SegNet-Tutorial` (accessed on 12 Dec 2024)

6.2.1.1 Histogram of Fixated Image Segments Encoding (HoFIS)

We implement an encoding of visual search sequences based on fixated image segments (see figure 6.5). Our algorithm takes a visual scene and the fixation sequence of a user as input. In the first step, we segment the scene image using SegNet (Badrinarayanan et al., 2017), which assigns each pixel to exactly one segment class, i.e., one of the 37 semantic classes that were considered for training SegNet. We applied this as a pre-processing to all images. In the second step, we extract image patches for each fixation of the visual search process and determine the most frequent segment class in each patch. The predominant segment classes of a visual search trial are aggregated into a segment histogram vector that is used as input to our classification model. We hypothesize that segment encoding can improve the classification accuracy of search target inference. We assume the model training benefits from the encoding because it conveys the semantics of fixated segment classes. For instance, tables might be fixated more often than the floor or walls when searching for a pen. However, it is an open question whether the `HoFIS` encoding is beneficial for in- and outdoor scenes contained in the VIU dataset (Koehler et al., 2014) because SegNet is trained using indoor scenes.

6.2.1.2 Bag of Deep Visual Words Encoding (BoDVW)

An extension to *Bag of Deep Visual Words*, which showed improved performance in classifying the search target class, was presented in section 6.1. We use the *Bag of Deep Visual Words* encoding in our experiments.

6.2.2 Experiment

We conduct a simulation experiment to compare the performance of three approaches that predict the target segment class containing the search target of a visual search. We consider inference models using the `BoDVW` encoding x_1 as a feature, the `HoFIS` encoding x_2 and a combined version that concatenates x_1 and x_2 (see figure 6.5). The fixations of a visual search trial are encoded using one of these methods and serve as input for model training and target inference. The visual search trials are taken from the VIU dataset by Koehler et al. (2014), which we labeled with the ground truth region of the search targets. We consider the accuracy in predicting the correct target class as a performance metric (dependent variable). In the following, we describe the dataset and the evaluation procedure and report our results.

6.2.2.1 Extending the VIU Dataset

We use the VIU dataset published by Koehler et al. (2014) to test our segmentation-based approach for search target inference. It includes natural scenes, including indoor and outdoor environments, other than artificial scenes, such as image collages. The dataset includes 800 natural scene images with recorded gaze data from 19 participants for several tasks—free viewing, search for highest saliency, and cued object search. We use gaze recordings from the cued object search task in which the written description of an object was shown to participants, who were then asked whether the target was present in a scene image presented next for 2000 ms. However, only 400 of 800 images contained the cued object. Only the 400 images containing the cued object

are included in our visual search experiment. We filter, pre-process, and additionally annotate this part of the original VIU dataset to investigate the performance of search target inference algorithms in ubiquitous computing settings. We manually annotate the 400 remaining images with bounding boxes for the ground-truth region: one or multiple rectangles that approximate the region of at least one instance of the target object class, i.e., the cued object. Further, we process all images using the pre-trained SegNet model, mapping image regions to segment classes (pixel-based). The target segment class – the segment class that contains the search target – is chosen by selecting the most frequent segment class within the ground-truth region. This dominant segment class serves as prediction label y for all 19 search trials per scene image. As VIU contains natural scenes, the predicted segment classes and, thus, the labels for classification are highly unbalanced. Many of the 37 segment classes occur rarely or not at all. We compensate for this by including segment classes that occur at least 21 times: 8 target segment classes and 168 scene images remain. The segment classes include wall, table, chair, window, floor, picture, box, and door (in order of decreasing frequency with respect to our dataset). This totals 3,192 visual search trials from 19 participants with 168 different scenes. With 21 scenes per target class, we have a balanced model training and evaluation dataset with a chance level of 12.50% (8 classes). An additional pre-processing step that we apply is to cut all fixation sequences after the target object was fixated for the first time, including the initial fixation on the target area, assuming that the object was found. Subsequent fixations are not driven by the intended task, i.e., finding the cued object. This step reduces the average number of fixations per visual search trial from 7.98 ($SD = 2.43$) to 4.33 ($SD = 2.91$). Our extensions to the VIU dataset can be downloaded from our GitHub[3].

6.2.2.2 Evaluation Procedure

Our experiment compares three inference approaches based on image segmentation using our modified version of the VIU dataset (Koehler et al., 2014). Our approaches are based on the `BoDVW` and the `HoFIS` encoding. We initialize multiple instances for the `BoDVW` encoding, each using a different neural network layer for image patch encoding. We initialize the `BoDVW` method using a train set (we use cross-validation) and encode the fixation sequences into vectors x_1. The `HoFIS` encoding x_2 is generated from the SegNet-based image annotations of our preprocessing step. We train a support vector machine for all methods that predict the output labels y, i.e., the segment classes. Similar to Sattar et al. (2015), we consider a training that includes the data of all users (*cross-user*) and a *user-specific* training. For the latter, the data-driven codeword initialization of the `BoDVW` encoding and the model training is performed separately for each user. In addition, we investigate the performance of our approach for indoor and outdoor scenes separately. SegNet is trained using indoor scenes only. Hence, it is likely that the segmentation of outdoor scenes is semantically less meaningful, which might impact the accuracy of search target inference. We perform this experiment using the *user-specific* `BoDVW+HoFIS`-based model. We select 48 balanced training samples from each participant, including in- and outdoor scenes, and test separately on 8 held-out indoor or outdoor scenes. On the one hand, it would be better to include only indoor samples for training. On the other hand, the training and test sets would

[3] `https://github.com/DFKI-Interactive-Machine-Learning/STI-Dataset` (accessed on 12 Dec 2024)

be even smaller. This experiment shall provide a first insight into the impact of the scene type used for training SegNet but is limited due to the low amount of balanced training samples that can be extracted from the extended VIU dataset.

Codeword Initialization

For initializing the codewords for the `BoDVW` approach, we extract image patches around all fixations of the train set. We crop squared fixation patches with an edge length $l_e = 45$ px and generate $k = 10$ codewords for the k-means clustering. This is less compared to Sattar et al. (2015) and our previous experiment (see section 6.1) because our considered dataset contains images with a smaller size and significantly shorter fixation sequences.

Model Training

We train a one-vs-all multi-class SVM for all conditions defined by the sequence encoding and, in the case of `BoDVW`, by the considered network layer for image patch encoding. We use a grid search for parameter optimization. For the *cross-user* setting, we perform a 2-fold cross-validation for each condition based on users: we split users into two groups, one for training and the other for validation, respectively. Concerning the *user-specific* setting, we conduct a 10-fold cross-validation for estimating the accuracy per user (n=19). We estimate the overall accuracy for each condition by averaging the accuracy values over all users.

Conditions

In our evaluation, we consider three feature encoding methods for predicting the target segment class of a visual search and a corresponding scene image: `BoDVW`, `HoFIS`, and their concatenation `BoDVW+HoFIS`. For all methods that depend on `BoDVW`, we report performance values using seven different layers for the image patch encoding step. Further, we differentiate between models that are trained across users (*cross-user*) and specifically for each user (*user-specific*).

Hypotheses

We hypothesize that our segmentation-based inference approach enables search target inference for natural interaction scenarios using one of the considered sequence encodings (H1). We expect that using the `HoFIS` encoding improves the model performance (H2). The underlying assumption is that `HoFIS` captures semantic cues of the scene based on the SegNet segmentation, which complements the `BoDVW` feature representation and eventually improves the model performance. In addition, we investigate the difference in performance when applying our model to indoor and outdoor scenes separately. The resulting image segments for outdoor scenes might be semantically less meaningful because SegNet is pre-trained using indoor scenes. As this might deteriorate the model performance, we hypothesize that our approach works better for indoor scenes (H3). Similar to Sattar et al. (2015), we expect higher accuracies for the *user-specific* model training in all cases (H4). Further, we are interested in identifying the layer of the pre-trained AlexNet that provides the most suitable image patch encodings for search target inference. We expect

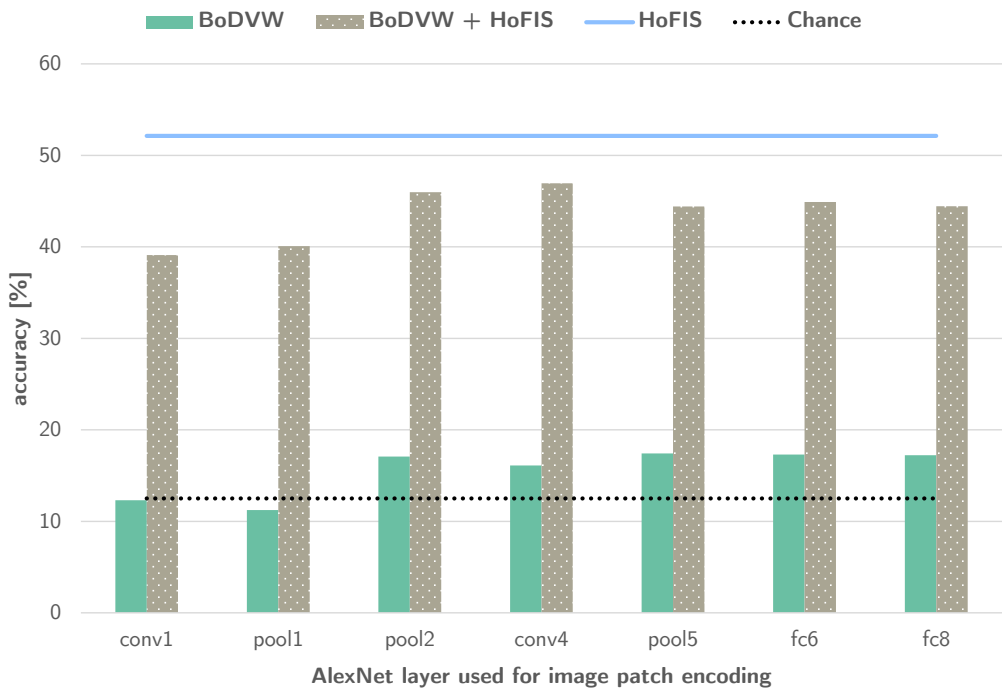


Figure 6.6: Accuracy in predicting the target segment class of a visual search for the *cross-user* model training and all considered sequence encoding methods.

to confirm our previous findings (see section 6.1) that the `fc8` layer works best for inferring the search target (here: target segment class) of users (H5).

6.2.2.3 Results

Figure 6.6 summarizes the results for all sequence encoding methods using the *cross-user* setting for model training: data of multiple users are included. The target inference models based on the `BoDVW` encoding achieve an average accuracy of 15.53%. The best performance of 17.42% is achieved using the activations of the `pool5` layer. Concatenating the `BoDVW` and the `HoFIS` feature vector for training yields a better average accuracy of 43.70% in classifying the correct target segment class. With 46.95%, the `conv4`-based `BoDVW` feature vector performs best. The model trained using the `HoFIS` encoding only achieves the best accuracy for the *cross-user* setting of 52.13%. The results for the *user-specific* training setting are shown in figure 6.7. With the `BoDVW` encoding, the target inference models achieve an average accuracy of 18.93% (mean of the accuracy of participants). The best performance of 20.60% is achieved using the `fc8` layer activations for codeword generation, closely followed by the `pool5` (20.30%) and the `conv4` layer (19.77%). Similar to the *cross-user* setting, the concatenated feature vector `BoDVW+HoFIS` achieves a better average accuracy of 41.30%, the best accuracy of 43.96% is observed for the `conv4`-based encoding. Overall, the model based on the `HoFIS` encoding achieved the best accu-

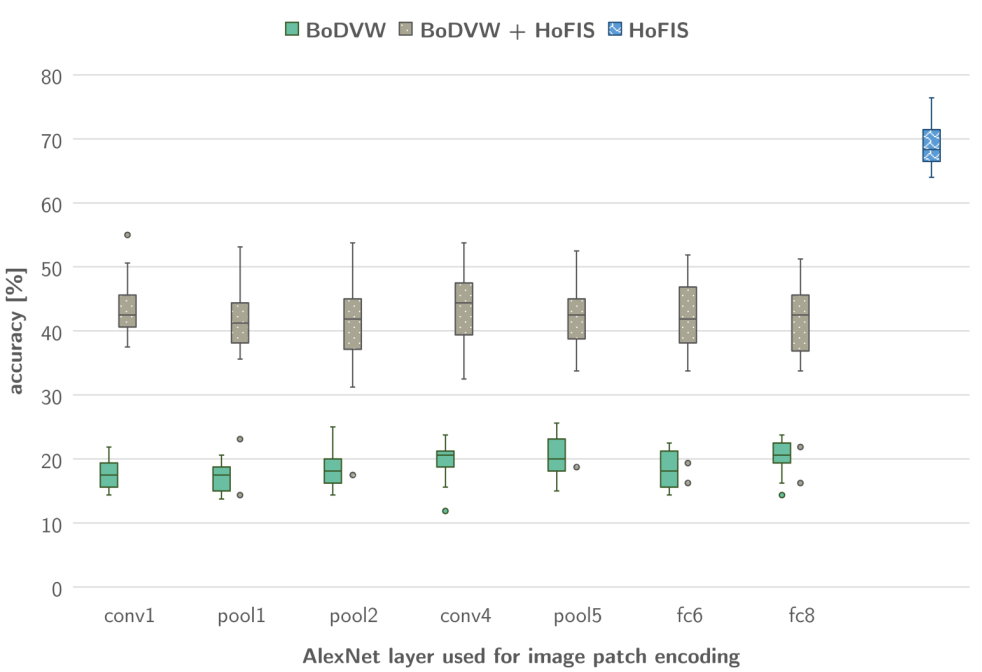


Figure 6.7: Boxplot of mean accuracy scores of all participants (n=19) in predicting the target segment class of visual search trials with *user-specific* model training, including all sequence encoding methods.

racy of 68.81%. A repeated measures ANOVA with Greenhouse-Geisser correction (Mauchly's test, $\chi^2(2) = 6.04, p = .049$, indicates a violation of sphericity) determined that the accuracy in classifying the correct target segment differs statistically significantly between the two `conv4`-based models and the `HoFIS`-based model ($F(1.54, 27.71) = 891, p = .000$). Post hoc tests using the Bonferroni correction revealed that all pairwise differences are significant with $p < .001$. We get similar results for other layers but restrict ourselves to the `conv4` layer because it achieved the best or close to the best results for both methods. We do not test statistical significance for the *cross-user* setting because we have only two accuracy values from the 2-fold cross-validation across all users. Concerning our additional model experiment for comparing the performance for in- and outdoor scenes, our *user-specific* `BoDVW+HoFIS`-based model achieves an accuracy of 28.95% ($SD = 17.70$) for outdoor scenes and 36.45% ($SD = 8.79$) on indoor scenes. Even though a 2-tailed exact Wilcoxon signed-rank test showed that the difference is not significant ($Z = -1.773, p = .077$).

6.2.3 Discussion

The results of our evaluation show that our approach to search target inference, using automated image segmentation with different sequence encoding methods for visual search, achieves accuracies above the chance level when applied to unstructured natural scenes from the extended VIU dataset. In particular, the `HoFIS` sequence encoding achieves a good classification accuracy of

68.81% for *user-specific* training, which suggests that H1 can be confirmed. Including the *Histogram of Fixated Image Segments* in model training is very beneficial for the *cross-user* and the *user-specific* setting. Concatenating it to the `BoDVW` encoding, the accuracy increases by 28.17% to 43.70% for the *cross-user* setting and by 22.37% to 41.30% for the *user-specific* setting. In both settings, the `HoFIS` encoding performs best overall with 52.13% for the *cross-user* setting and 68.81% for the *user-specific* setting. The differences in means for the *user-specific* setting are statistically significant, suggesting that H2 can be confirmed. But, the `HoFIS` encoding achieves a better performance without the `BoDVW` encoding. At the same time, we expected that the two feature encodings complement each other, e.g., through interaction effects or because additional semantic cues are encoded. This rather indicates that we have to reject H2 and might have several reasons. One possible explanation is that the increased number of features for `BoDVW+HoFIS` leads to an over-fitting to the training data and, hence, a drop in the reported test accuracy. Further, the model training might not converge to optimal weights due to the relatively small dataset. Our comparison of indoor and outdoor scenes revealed that our approach performs slightly better on the given indoor scenes. However, the difference is not significant, which might be related to the small amount of available training data. We assume this trend (and hence H3) could be confirmed if sufficient data samples were available for training. A question that remains open is if `HoFIS` encodes semantically meaningful cues for in- and outdoor scenes and how this relates to the observed accuracies. Follow-up experiments should clarify whether the data-driven selection of segment classes is meaningful for unseen indoor scenes and how they apply in detail to outdoor scenes: are the indoor segment classes approximating meaningful segments for outdoor scenes? Further directions include the development of methods including multiple segmentation models with scene recognition for improving the semantic meaning of segments and similarity-based approaches that abstract from the pre-defined semantics of SegNet's segmentation classes.

Our expectation that *user-specific* models perform better, e.g., due to differences in personal gaze behavior, can be confirmed in the case of the `HoFIS` encoding, which performs better when trained individually for each participant. However, the results for `BoDVW` only increase slightly, and we observe a marginal performance drop for `BoDVW+HoFIS`. One reason for this observation might be that a model using the `HoFIS` encoding better generalizes across users and superposes the effect of user-specific model training when combined with the `BoDVW` encoding. Another reason could be that the *user-specific* training includes fewer trials for training than the *cross-user* training and, hence, be related to the potential over-fitting problem mentioned above. Eventually, our results suggest that H4 can be confirmed for the `HoFIS`-based model only. We could reproduce our previous findings (see section 6.1) that using the `fc8` layer with the *BoDVW* method and *user-specific* training yields the best classification accuracy. However, this does not hold for all conditions and training settings. For that reason, but also because our `HoFIS`-based model performs better than all `BoDVW`-based methods, we reject H5.

Replacing object classes with segment classes as the prediction target facilitates search target inference independent of the target object. Also, the nature of scene images in the VIU dataset indicates that search target inference in natural interaction scenarios is feasible. However, a prediction accuracy of 68.81% is too low for robust applications, yet it is comparable to or even better than results from the literature, which concern more artificial scenarios. Using target inference for, e.g., pervasive activity tracking and extending artificial episodic memories, requires

further improvements of algorithms. In particular, our approach must be transferred to real-time applications involving dynamic scenes. In this direction, future work should be concerned with developing and evaluating methods for applying search target inference in real-world scenarios and identifying related challenges. Examples include the detection of visual search patterns among non-related gaze patterns and the impact of inference methods and their accuracy on the overall performance of users. Previous works on search target inference are exclusively based on data from visual search experiments that contain visual search trials by experiment design. One exception can be found in Dietz et al. (2017): they present a binary classifier for visual search activities during indoor navigation using features based on human gaze and head motion. Future experiments should also focus on identifying additional factors that improve prediction accuracy. Examples include knowledge about the user's context. For instance, a user's location, activity, and cognitive state could be considered.

6.3 Conclusion

We investigated whether eye tracking can be used to infer the target in a visual search process. We developed two novel encodings for scanpaths based on a sequence of fixated visual stimuli in a scene and investigated their influence on the effectiveness of models for inferring the search target. First, we introduced the *Bag of Deep Visual Words* method for integrating learned features for image classification in the popular *Bag of Words* sequence encoding algorithm for the purpose of search target inference. An evaluation showed that our approach performs better than similar approaches from the literature Sattar et al. (2015), in particular, when excluding fixations on the visual search target. Further, we proposed a segmentation-based approach that enables search target inference in natural interaction scenarios. For this, we infer the image segment that contains the search target instead of the target itself. We investigated the performance of three different methods for visual search encoding using this principle: one using the previously introduced `BoDVW` encoding as a baseline, the `HoFIS` encoding that reflects segment classes fixated during a visual search, and a concatenation of them. An evaluation using our manually extended dataset showed that our new encoding significantly improves classification accuracy. We conclude that semantic dependencies in search tasks provide a high potential for future research to improve the stability and precision of target inference systems.

Chapter 7

Estimating Document Relevance

Searching for information on the web or in a knowledge base is pervasive. However, search queries to information retrieval systems seldom represent a user's information need precisely (Carpineto and Romano, 2012). At the same time, a growing number of available documents, sources, and media types further increase the required effort to satisfy an information need. Implicit relevance feedback, obtained from users' interaction signals, was proposed to improve information retrieval systems as an alternative to more accurate but costly explicit feedback (Agichtein et al., 2006). Behavioral signals that were investigated in this regard include clickthrough data (Joachims et al., 2017; Agichtein et al., 2006), dwell time of (partial) documents (Buscher et al., 2009), mouse movements (Akuma et al., 2016; Eickhoff et al., 2015), and eye movements (Buscher et al., 2012). This data may originate from search logs, which can be used to tune the ranking model of a search engine offline, or from real-time interaction data to extend search queries during a search session or to identify relevant text passages. This work aims to identify relevant paragraphs using real-time eye tracking data as input.

Eye movements play an important role in information acquisition (Gwizdka and Dillon, 2020) and were shown to be an effective source of implicit relevance feedback in search (Buscher et al., 2008a) and decision-making (Feit et al., 2020). However, eye movements highly depend on user characteristics, task, and content visualization (Buchanan et al., 2017). Related approaches use eye tracking to infer the perceived relevance of text documents concerning previously shown trigger questions (Salojärvi et al., 2003, 2004, 2005a; Loboda et al., 2011; Buscher et al., 2008a; Gwizdka, 2014a; Bhattacharya et al., 2020b,a), and to extend (Buscher et al., 2008b; Chen et al., 2015) or generate search queries (Hardoon et al., 2007; Ajanki et al., 2009). A common disadvantage of approaches for gaze-based relevance estimation is that they are tested using documents with constrained layouts and topics such as single sentences (Salojärvi et al., 2003, 2004, 2005a) or short news articles that fit on the screen at once (Gwizdka, 2014a; Bhattacharya et al., 2020b,a; Loboda et al., 2011; Buscher et al., 2008a). Hence, it is unclear whether related findings generalize to more realistic settings such as those that include Wikipedia-like web documents.

We investigate whether eye tracking can be used to infer the perceived relevance of read documents concerning previously shown trigger questions in a less constrained setting. We include multi-paragraph documents that exceed the display size and require scrolling to read

the whole text. For this, we conducted a user study with n=24 participants. Participants read single- and multi-paragraph articles and rated their relevance at the paragraph level while their eye movements were recorded. Single-paragraph documents with corresponding trigger questions originate from the g-REL corpus (Gwizdka, 2014a). Multi-paragraph documents with related questions are selected from the Google Natural Questions (GoogleNQ) corpus (Kwiatkowski et al., 2019). We assemble a corresponding dataset, the *gazeRE* dataset, which is available to the research community under an open-source license via GitHub (see section 7.1.5). Using the gazeRE dataset, we aim to confirm the findings from the literature on short news articles and investigate whether they can be generalized to multi-paragraph documents from Wikipedia. We model the perceived relevance using machine learning and the features from Bhattacharya et al. (2020a) as input. In this chapter, we contribute as follows:

- Section 7.1: We created the *gazeRE* dataset through a user study with n=24 participants. Participants read single- and multi-paragraph articles and rated their relevance at the paragraph level.

- Section 7.2: We investigated whether eye tracking features can be used to model perceived relevance through a machine learning experiment. We investigated whether findings from the literature on short news articles generalize to longer multi-paragraph documents (Barz et al., 2022).

7.1 Data Collection: gazeRE Dataset

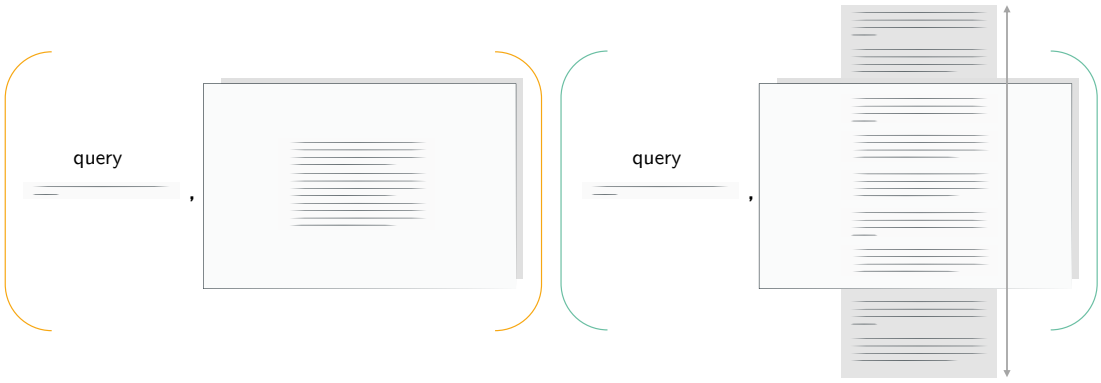
We conduct a user study (n=24) to collect eye movement data during relevance estimation tasks. The participants are asked to read documents of different lengths and judge, per paragraph, whether it answers a previously shown trigger question. We use this data to model the relation between the recorded gaze data and the perceived relevance using machine learning in section 7.2.

7.1.1 Participants

For our study, we invited 26 students (15 female) with an average age of 27.19 years ($SD = 5.74$). Data from two participants had to be discarded because they withdrew their participation. The remaining participants reported having normal (11) or corrected to normal (13) vision, of which 11 wore eyeglasses and 2 wore contact lenses. Ten of them participated in an eye tracking study before. The participants rated their language proficiency in English for reading texts as native (1), fluent (18), or worse (5). Each participant received € 15 as compensation.

7.1.2 Stimuli

The stimuli data used in our study are pairs of trigger questions and documents with one or multiple paragraphs (see figure 7.1). We use a subset from the g-REL corpus (Gwizdka, 2014a) with single-paragraph documents that fit on one page and selected pairs from the Google Natural Questions (NQ) corpus, which includes multi-paragraph documents that require scrolling



(a) Example pair of the g-REL corpus **(b)** Example pair of the Google NQ corpus

Figure 7.1: We sample stimuli from 7.1a the g-REL corpus, which includes pairs of questions and short English news articles, and 7.1b from the Google NQ corpus, which includes pairs of questions and English Wikipedia articles.

(Kwiatkowski et al., 2019). Both corpora include relevance annotations per paragraph, which we call system relevance.

7.1.2.1 g-REL Corpus

The g-REL corpus includes a set of 57 trigger questions and 19 short English news texts that fit on one page. Questions include, for instance, "Where is the headquarters of OPEC located?" and "What was Camp David originally named?". The news texts are either irrelevant, topically relevant, or relevant concerning these questions: the corpus includes three questions per document. If a document is irrelevant, it is off-topic and does not contain an answer to the question. Topically relevant and relevant documents are on topic, but only the relevant texts contain an answer to the question. The original news texts were selected from the AQUAINT Corpus of English News Texts (Graff, 2002) as used in the TREC 2005 Question Answering track[1]. The questions and judgments (system relevance) from TREC data were further revised and tested by Michael Cole and Jacek Gwizdka. Prior results for this corpus have been published in work by Gwizdka (2014a,b, 2017); Bhattacharya et al. (2020b,a). Like Bhattacharya et al. (2020b,a), we consider a binary relevance classification. Hence, the topically relevant document-question pairs are counted as irrelevant ones. For our user study, we select a balanced subset of 12 distinct documents, of which four are relevant, four are topical, and four are irrelevant concerning the accompanying trigger question. We selected two additional documents for the training phase, one of which is relevant and one that is topical. We select the news texts such that the length distribution is similar to the whole corpus. The mean number of tokens of the selected news texts is 170.500 (SD = 14.211). If all documents were included, the mean number of tokens would have been 176.404 (SD = 12.346). We used a simple whitespace tokenizer, which segments each

[1] `https://trec.nist.gov/data/qa.html` (accessed on 12 Dec 2024)

document into a list of words to determine the number of tokens in each document.

7.1.2.2 Google Natural Questions Corpus

The Natural Questions (NQ) corpus[2] by Google includes 307k pairs of questions and related English Wikipedia documents (Kwiatkowski et al., 2019). Example questions include "What is the temperature at bottom of ocean?" and "What sonar device let morse code messages be sent underwater from a submarine in 1915?". Each document includes multiple HTML containers, such as paragraphs, lists, and tables. Each container that answers the accompanying question is listed as a *long answer*. We consider this container to be relevant (system relevance). In addition, the corpus provides a *short answer* annotation if a short phrase exists within a container that fully answers the question. The Google NQ questions are longer and more natural than other question-answering corpora, including TREC 2005. For our user study, we select a subset of 12 pairs of documents and questions (plus one for training) from the NQ training data using a set of filters followed by a manual selection. Our filter removes all documents with at least one container different than a paragraph because we focus on continuous texts in this work. It selects documents with exactly one long and one short answer, i.e., all but one paragraph per document can be considered irrelevant. Also, it removes all documents that have very short or very long documents. We drop documents with less than 20 or more than 200 tokens. Finally, our filter selects all documents with five to seven paragraphs, which results in a set of 355 of 307k question-document pairs. We manually select documents from the remaining ones guided by two factors: the average number of tokens and the position of the relevant paragraph. The remaining documents have an average length of 420.083 ($SD = 54.468$) tokens. This means a document's height corresponds to two times the display's height, and participants must scroll through the document to read all paragraphs. The position of relevant paragraphs is balanced: we select two documents with an answer at position i with i ranging from 0 to 5. On average, each paragraph contains 72.55 tokens.

7.1.3 Tasks & Procedure

At the beginning of the study, each participant is asked to sign an informed consent form and to fill in a demography questionnaire. The remainder of the study is divided into two blocks, which follow the same pattern (see figure 7.2). In each block, stimuli from one of the two corpora are presented (within-subjects design). The starting order is alternating to avoid ordering effects. At the beginning of each block, the experimenter provides block-specific instructions and asks the participant to calibrate the eye tracking device. Next, the participant completes a training phase to get familiar with the task, the user interface, and the characteristics of the stimuli from the current corpus. We include two training examples for g-REL and one for Google NQ. The participant is encouraged to ask questions about the system and the task in this phase. Subsequently, the participant completes the main phase of the block, which includes 12 stimuli of the respective corpus. After both blocks are finished, participants receive the compensation payment. The participant's task is to mark all paragraphs of a document as relevant that contain

[2] `https://ai.google.com/research/NaturalQuestions` (accessed on 12 Dec 2024)

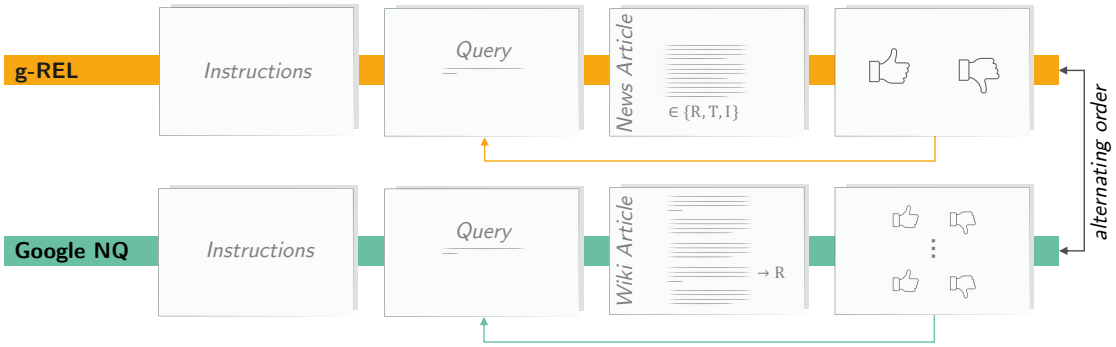


Figure 7.2: Procedure of our user study with one block of tasks per corpus of stimuli: g-REL and Google NQ.

an answer to the previously shown trigger question (query). Participants read the query and navigate to the corresponding document, either a *news article* or a *wiki article*. There is no time constraint for reading the article. Then, participants move to the rating view, which enables them to enter a binary relevance estimate (perceived relevance) per paragraph. At this stage, the query and the text of the paragraph are available to the participant. For stimuli from the g-REL corpus, participants must provide one relevance estimate for the whole text. For stimuli from the Google NQ corpus, participants must provide five to seven relevance estimates, depending on the number of paragraphs.

7.1.4 Apparatus

The study is conducted in a separate room of our lab. We use the Tobii 4C eye tracker[3], a non-intrusive remote eye tracker, which is attached to the lower bezel of a 27-inch screen. This monitor has a resolution of 2560×1440 pixels, and the attached eye tracker collects the gaze data with a sampling rate of 90 Hz. The monitor and eye tracker are connected to an experimenter's laptop running the study software and a monitoring tool. The participants are seated approximately 60 cm in front of the connected display. A mouse is provided to scroll through documents, navigate between views, and rate each paragraph's relevance. The text-based stimuli are displayed in black, 38-points Roboto font[4] on a white background. We calibrate the eye tracker before the user executes the tasks using the built-in 9-point calibration procedure. During the calibration process, the user is asked to look at calibration dots on the connected display until they vanish. We use the multisensor-pipeline (see chapter 9), our Python-based framework for building stream processing pipelines, to implement the study software that is responsible for showing the stimuli and recording the interaction signals according to our experiment procedure.

[3] `https://help.tobii.com/hc/en-us/articles/213414285-Specifications-for-the-Tobii-Eye-Tracker-4C` (accessed on 12 Dec 2024)

[4] `https://fonts.google.com/specimen/Roboto` (accessed on 12 Dec 2024)

Figure 7.3: Setup of our user study: A user is seated approximately 60 cm from a 27-inch display with the remote eye tracker mounted at its lower bezel.

7.1.5 gazeRE Dataset

We assembled the stimuli and the recorded interaction signals into the *gazeRE* dataset, a dataset for **gaze**-based **R**elevance **E**stimation. It includes relevance ratings (perceived relevance) from 24 participants for 12 stimuli from the g-REL corpus and 12 stimuli from the Google NQ corpus. Also, it includes participants' eye movements per document in terms of 2D gaze coordinates on the connected display. We use the gazeRE dataset for modeling the perceived relevance based on eye tracking in this work. It is publicly available under an open source license on GitHub[5].

Processing of Eye Tracking Data

The gaze data included in the gazeRE dataset is pre-processed and cleaned. We correct irregular timestamps by resampling the signal with a fixed sampling rate of 83 Hz, corresponding to the ratio of samples per recording length. Further, we use the `gap_fill` algorithm, which linearly interpolates the gaze signal to close small gaps between valid gaze points (Olsen, 2012). This may happen due to a loss of tracking. In addition, we use the Dispersion-Threshold Identification (I-DT) algorithm to detect fixation events (Salvucci and Goldberg, 2000).

Dataset Format

The gazeRE dataset includes synchronized time-series data per document and user. Each record includes a column for timestamps, gaze coordinates (x and y), a fixation ID if the gaze point belongs to a fixation event, the scroll position, and the ID of the paragraph that is hit by the current point of gaze. The origin of the gaze and fixation coordinates is the lower-left corner of the display $(0, 0)$ while $(2560, 1440)$ denotes the upper-right corner. The scroll position reflects the status of the scrollbar and lies between 0 and 1. The position is 1 if the document head is visible or the document is not scrollable. It is 0 if the end of the document is visible. We

[5] `https://github.com/DFKI-Interactive-Machine-Learning/gazeRE-dataset` (accessed on 12 Dec 2024)

Corpus	Subset	Relevant	Irrelevant	Total
g-REL	*agree*	86 (48%)	95 (52%)	181 (63%)
	topical	20 (20%)	76 (80%)	96 (33%)
	all	**107 (37%)**	**181 (63%)**	**288 (100%)**
Google NQ	*agree*	248 (17%)	1190 (83%)	1438 (86%)
	all	**450 (27%)**	**1230 (73%)**	**1680 (100%)**

Table 7.1: Number of samples in our dataset per corpus and subset. The topical subset includes samples for irrelevant paragraphs that are on the topic of the trigger questions. The agree subset includes samples for which the participant's relevance rating matches the system relevance and is not topical. Each trial corresponds to one paragraph that was either perceived as relevant or irrelevant.

provide the perceived relevance per document and user: *True* is used for positive ratings, i.e., if a paragraph was perceived as relevant, *False* otherwise.

Descriptive Statistics

We report descriptive statistics and agreement statistics of the relevance ratings in our dataset. We use Fleiss' κ to determine if there was an agreement in our participants' judgment on whether paragraphs are relevant with respect to a trigger question. If the agreement among participants is low, the rating task might have been too difficult, or participants might have given inadequate ratings. Further, we compute Cohen's κ to determine the level of agreement between each participant's relevance rating (perceived relevance) and the ground-truth relevance (system relevance). We report the mean agreement, averaging over all participants. We expect that the ratings of our participants moderately differ from the system relevance, similar to the findings in Bhattacharya et al. (2020a). For the g-REL corpus, we include a total of 288 trials, i.e., eye movements and a corresponding relevance estimate per paragraph (see table 7.1). The 12 documents, each being a single paragraph, include 4 relevant paragraphs (system relevance). On average, the participants rated 4.46 ($SD = 1.04$) paragraphs as relevant: they perceived 107 (37%) as relevant and 181 (63%) as irrelevant. Fleiss' κ reveals a good agreement for perceived relevance ratings with $\kappa = 0.641$. The mean of Cohen's κ of 0.769 (SD=0.197) indicates a substantial agreement between the participant and ground-truth relevance ratings. We obtained a total of 1680 trials using the Google NQ corpus. The 12 stimuli include 12 relevant paragraphs out of 70. On average, the participants rated 18.75 ($SD = 4.361$) paragraphs as relevant: they perceived 450 (27%) as relevant and 1230 (73%) as irrelevant. Fleiss' κ reveals a moderate agreement for perceived relevance ratings with $\kappa = 0.576$. Also, the mean of Cohen's κ of 0.594 ($SD = 0.126$) indicates a moderate agreement between the perceived and the system relevance.

7.2 Gaze-based Relevance Estimation

We investigate different methods for predicting the perceived relevance of a read paragraph based on a user's eye movements. We consider relevance prediction a binary classification problem

because paragraphs are either relevant or irrelevant. Each classification model takes a user's eye movements from reading a paragraph as input to predict the perceived relevance of this paragraph. The explicit user ratings are used as ground truth. In the following, we describe our method for extracting gaze-based features at the paragraph level, depict our model training and evaluation procedure, and report the results based on the gazeRE dataset.

7.2.1 Extraction of Gaze-based Features

To encode a user's eye movements for a certain paragraph p, we have to extract coherent gaze sequences within the paragraph area. A user might visit a paragraph multiple times during the relevance judgment process. We refer to these gaze sequences as visits $v_p^i \in V_p$ where i indicates the order of visits. We implement an algorithm that extracts all visits to a paragraph with a minimum length while ignoring short gaps. It identifies consecutive gaze samples within the given paragraph's area and groups them into a visit instance each. As long as there is a pair of two subsequent visits with a gap shorter than 0.2 s, these are merged. All visits with a minimum length of 3 s are returned as a list. We found that this duration ensures that at least 3 fixations are contained in each visit, which is required to compute the convex hull features. We use the longest visit per paragraph v_p^* to encode eye movements.

We implement a set of 17 features that were successfully used to model the perceived relevance of short news articles in Bhattacharya et al. (2020a). This requires selecting one visit or to merge them. We decided to use the longest visit, assuming that the largest consecutive sequence of gaze points is most likely to capture indicative eye movements. Our feature extraction function f returns a vector of size 17 per visit: $f(v) \rightarrow \mathbb{R}^{17}$. Four of these features are based on fixation events, eight are based on saccadic movements, and five are based on the area spanned by all fixations. Table 7.2 provides an overview of all features and describes how they are computed. Some features are normalized by a width factor `w` or a height factor `h`. In Bhattacharya et al. (2020a), these correspond to the display width and height, respectively. We set `w` and `h` to the width and height of the current paragraph because the display size does not respect the different paragraph sizes and the scrolling behavior. The absolute reading time of a visit (`scan_time`) is used to compute velocity-based or time-normalized features. The `hull_area`, i.e., the area of the convex hull around all fixations, is used to compute two area-based features.

7.2.2 Model Training and Evaluation

We build and compare several machine learning models that take an encoded paragraph visit v_p^* as input and yield a binary relevance estimate as output. The models are implemented using the scikit-learn machine learning framework (Pedregosa et al., 2011). Model training and testing are done using our gazeRE dataset, which includes eye movements and relevance estimates for documents from the g-REL corpus and the Google NQ corpus. We refer to these partitions as g-REL data and Google NQ data.

	Feature	Description
fixation -based	`fixn_n`	Number of fixations
	`fixn_dur_sum`	Sum of fixation durations
	`fixn_dur_avg`	Mean of fixation durations
	`fixn_dur_sd`	Standard deviation of fixation durations
saccade -based	`scan_dist_h`	Sum of horizontal amplitudes of all saccades, normalized by a factor `w`
	`scan_dist_v`	Sum of vertical amplitudes of all saccades, normalized by a factor `h`
	`scan_dist_euclid`	Sum of Euclidean distances of normalized amplitudes of all saccades
	`scan_hv_ratio`	Ratio of horizontal to vertical amplitudes: `scan_dist_h`/`scan_dist_v`
	`avg_sacc_length`	Average saccade amplitude: `scan_dist_euclid`/(`fixn_n` − 1)
	`scan_speed_h`	Horizontal saccade velocity: `scan_dist_h`/`scan_time`
	`scan_speed_v`	Vertical saccade velocity: `scan_dist_v`/`scan_time`
	`scan_speed`	Saccade velocity: `scan_dist_euclid`/`scan_time`
area -based	`box_area`	Area spanned by summed saccade amplitudes: `scan_dist_h` ∗ `scan_dist_v`
	`box_area_per_time`	The `box_area` normalized by the scan time: `box_area`/`scan_time`
	`fixns_per_box_area`	Number of fixations per scanned area: `fixn_n`/`box_area`
	`hull_area_per_time`	The `hull_area` normalized by the scan time: `hull_area`/`scan_time`
	`fixns_per_hull_area`	Number of fixations per convex hull area: `fixn_n`/`hull_area`

Table 7.2: Overview of the 17 features adapted from Bhattacharya et al. (2020a) based on fixation events, saccadic eye movements, and the scanned area, which we use to encode paragraph visits.

7.2.2.1 Model Training Conditions

We largely replicate the conditions for model training and evaluation from Bhattacharya et al. (2020a) because we aim to confirm their findings: we group all visits $v \in V^*$ by their relevance rating into three subsets, train each model on 80% of the data of each subset, and evaluate it on the remaining 20% of the data. The grouping yields an *agree* subset, a *topical* subset, and the complete data denoted as *all*. Table 7.1 depicts the number of relevant and irrelevant samples in our dataset per subset. The *agree* subset includes all visits for which the perceived relevance rating agrees with the system relevance. All visits to topical articles, i.e., visits to on-topic articles that are irrelevant, are excluded as well. The *topical* subset includes visits to topical articles only, which are expected to be more difficult to classify. This subset is empty for the Google NQ corpus because its paragraphs are marked as either relevant or irrelevant. We report the model performance metrics averaged over 10 random train-test splits to estimate the generalization performance. We use the `train_test_split()` function of scikit-learn to split the visits stratified with prior shuffling.

7.2.2.2 Metrics

We include the same metrics as Bhattacharya et al. (2020a): the F1 score, i.e., the harmonic mean of precision and recall, the area under the curve of the receiver operator characteristic (ROC AUC), and the balanced accuracy. In addition, we report the true positive rate (TPR) and the false positive rate (FPR), which allow us to estimate the suitability of our models for building adaptive user interfaces similar to Feit et al. (2020).

	Model	F1 Score	ROC AUC	Balanced Accuracy	TPR	FPR
agree	RF	0.674	0.748	0.680	0.694	0.333
	RF*	0.677	0.747	**0.689**	0.688	**0.317**
	SVC*	**0.702**	**0.787**	0.683	**0.782**	0.417
topical	RF	0.119	**0.546**	**0.527**	0.100	**0.047**
	RF*	0.247	0.528	0.518	0.250	0.213
	SVC*	**0.270**	0.460	0.509	**0.325**	0.307
all	RF	0.458	0.650	0.594	0.405	**0.217**
	RF*	0.495	0.652	0.594	0.505	0.317
	SVC*	**0.506**	**0.652**	**0.605**	**0.510**	0.300

Table 7.3: Scores for all relevance prediction models using the g-REL corpus.

7.2.2.3 Model Configurations

We use the random forest classifier of scikit-learn with default parameters (`n_estimators` = 100) as our baseline model (`RF`), which worked well in Bhattacharya et al. (2020a). Further, we investigate the effect of using two pre-processing steps with a random forest classifier (`RF`*) or a support vector classifier (`SVC`*) with default parameters in an estimator pipeline. The default parameters of the `SVC`* model are `kernel` = `rbf` and `C` = 1. As a pre-processing step, we apply the oversampling technique SMOTE (Chawla et al., 2002) from the imbalanced-learn package (Lemaître et al., 2017) because visits to relevant paragraphs are underrepresented in our dataset (see table 7.1). Also, we apply a standard feature scaling method that removes the mean and scales features to unit variance. We train separate models for g-REL data and Google NQ data.

7.2.2.4 Hypotheses

We hypothesize that our models can effectively estimate the perceived relevance of short news articles as shown in Bhattacharya et al. (2020a) but using our newly assembled gazeRE dataset (H1). Confirming this hypothesis would also serve as a validation of our dataset. Further, we assume that the visit-based scanpath encoding enables the prediction of a participant's perceived relevance for individual paragraphs of long Wikipedia articles, i.e., if the participant must scroll through the document to read all contents (H2).

7.2.3 Results

We compare the performance of three models in predicting a user's perceived relevance using our gazeRE dataset. The performance scores for each model and subset are shown in table 7.3 (g-REL) and table 7.4 (Google NQ). For the g-REL data, we observe the best performance for the *agree* subset. Models trained on the *topical* subset achieve the worst results. Models for the *all* subset, which includes both other subsets, rank second. Across all subsets, the SVC* model performs best, or close to best, for most metrics. For the *topical* subset, the RF model without over-sampling and feature scaling achieves better ROC AUC and FPR scores. However,

	Model	F1 Score	ROC AUC	Balanced Accuracy	TPR	FPR
	RF	0.052	0.54	0.502	0.03	**0.027**
agree	RF*	0.246	0.543	**0.543**	0.278	0.229
	SVC*	**0.297**	**0.563**	0.54	**0.467**	0.388
	RF	0.189	0.552	0.517	0.129	**0.095**
all	RF*	0.331	0.552	0.527	0.343	0.289
	SVC*	**0.428**	**0.596**	**0.57**	**0.552**	0.412

Table 7.4: Scores for all relevance prediction models using the Google NQ corpus.

we observe a very low TPR and F1 score in this case. For the Google NQ data, models trained on the *all* subset rank best compared to their counterparts trained on the *agree* subset. Similar to our experiment on the g-REL data, the SVC* model performs best, or close to best, for both subsets. Also, the RF model achieves the best FPR score but the worst TPR and F1 scores.

7.3 Discussion

The results of our machine learning experiment for short news articles (g-REL data) are similar to those in Bhattacharya et al. (2020a) (see table 7.3). Our results indicate that we can effectively predict the perceived relevance for the *agree* subset, i.e., if the user's relevance rating agrees with the actual relevance of a paragraph and if irrelevant articles are not on topic. The *topical* trials are the most difficult to classify: our models fail in differentiating between relevant and irrelevant paragraphs if they are on topic. Including *all* samples for training, our models perform better than chance with an F1 score greater than 0.5. The best-performing model pipeline, on average, is SVC*, a support vector classifier with over-sampling and feature scaling. Bhattacharya et al. (2020a) reported results for the RF model based on the original g-REL corpus using the same features for training but with data from other participants. For the *agree* subset, their best model achieved an F1 score of 0.82, an ROC AUC of 0.92, and a balanced accuracy of 0.84. For the *topical* subset, they observed an F1 score of 0.3, an ROC AUC of 0.77, and a balanced accuracy of 0.59. Using *all* data samples results in an F1 score of 0.65, an ROC AUC of 0.85, and a balanced accuracy of 0.73. Even though we observed worse results per subset, we found the same overall pattern: the best performance is observed for models trained on the *agree* subset, followed by models for the *all* subset, and model for the *topical* subset rank last. This similarity is a good indicator of the validity of our gazeRE dataset, and, eventually, it suggests that we may confirm our hypothesis H1. The differences in model performance may have several reasons. For instance, it is likely that the higher amount of training data in Bhattacharya et al. (2020a) yields better models. They used 3355 trials from 48 participants compared to 288 trials from 24 participants in our experiment. Further, our user study was conducted at a university in Germany with participants being, besides one, non-native English speakers, while the studies reported in Bhattacharya et al. (2020a) were conducted at two universities in the United States and predominantly included native English speakers. This may lead to a higher degree of variance

in eye movements from our study. Another aspect may be that we used another eye tracking device; hence, the data quality and pre-processing steps likely differ.

For the Google NQ data in our machine learning experiment, we observe better scores when training on *all* data than when training on the *agree* subset (see table 7.4). However, the best-performing model, which is also the SVC* model, achieves F1 scores less than 0.5 in both cases, although we have access to a higher number of training samples (see table 7.1). The area under the ROC curve indicates classification performances better than chance, but we do not see enough evidence to confirm our hypothesis H2. A potential reason for the low performance might be that irrelevant paragraphs, in fact, belong to the same Wikipedia article as the relevant ones: the *agree* subset is rather a *topical* subset for which all user ratings agree with the system relevance. This would explain why models for the *agree* subset perform worse than models trained on *all* data. Also, the individual paragraphs in the Google NQ corpus are smaller than the ones in the g-REL corpus. This means that we aggregate less information per scanpath, which may deteriorate the model performance. Further, having multiple paragraphs allows the participants to revisit paragraphs. As we decided to encode the longest visit to a paragraph, we may miss indicative gaze patterns from another visit, which would have a negative impact on model training. In addition, the gaze estimation error inherent in eye tracking (Cerrolaza et al., 2012) may lead to a higher number of incorrect gaze-to-paragraph mappings: gaze-based interfaces should be aware of this error and incorporate it in the interaction design (Barz et al., 2018; Feit et al., 2017).

7.3.1 Feature Importance

We use 17 features as input to model the perceived paragraph relevance. In the following, we assess the importance of individual features to our best-performing model, the SVC* model. We use the permutation feature importance[6] method of the scikit-learn package (Pedregosa et al., 2011) to estimate feature importance because SVCs with an RBF kernel do not allow direct feature analysis. This method randomly shuffles the values of one feature at a time and investigates the impact on the model performance. The loss in model performance reflects the dependency of the model on this feature. We report the mean loss in the f1 score from 30 repetitions per feature as a measure of importance. We only analyze the feature importance for the *all* and *agree* subsets of the g-REL corpus because we observed f1 scores lower than 0.5 for all other conditions. The importance is reported on the training and test set of a single train-test split (80/20 split). We include both because features that are important for the training data but not for the test data might cause the model to overfit. The f1 test scores are 0.714 for the *agree* subset and 0.682 for *all* samples. However, this method might return misleading values if two features correlate. A model would still have access to nearly the same amount of information if one feature was permuted but could be represented by another one. Hence, we perform a hierarchical clustering on the feature's Spearman rank-order correlations and use one feature per cluster to asses its importance. For this, we follow the scikit-learn manual for handling multicollinearity[7]. The pairwise correlations and a grouping of our features based on correlation-

[6] `https://scikit-learn.org/stable/modules/permutation_importance.html` (accessed on 12 Dec 2024)

[7] `https://scikit-learn.org/stable/auto_examples/inspection/plot_permutation_importance_multicollinear.html` (accessed on 12 Dec 2024)

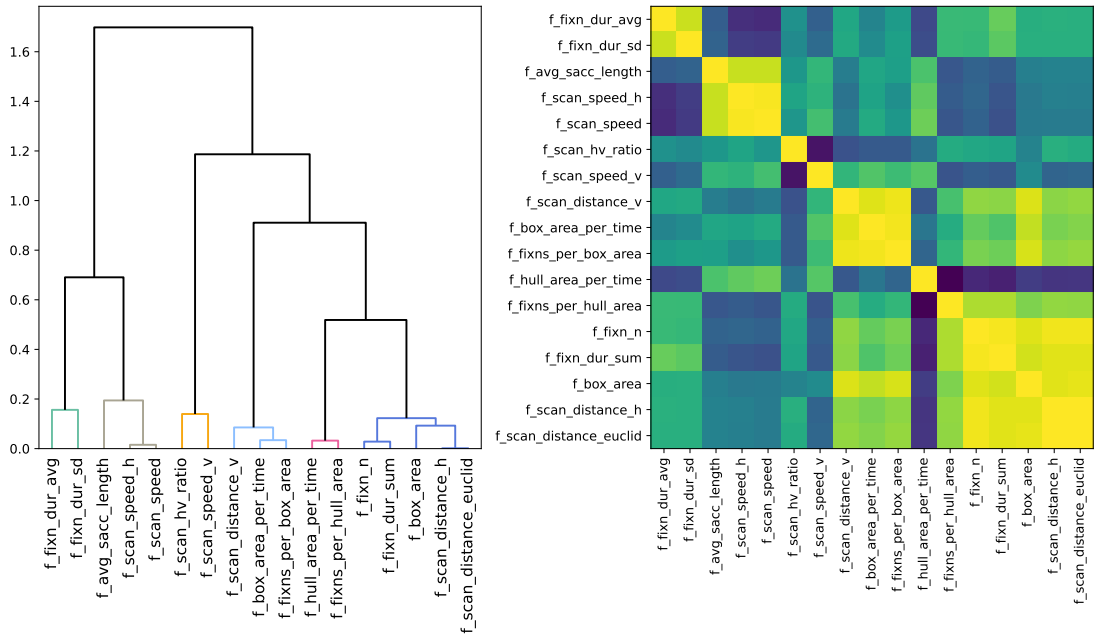


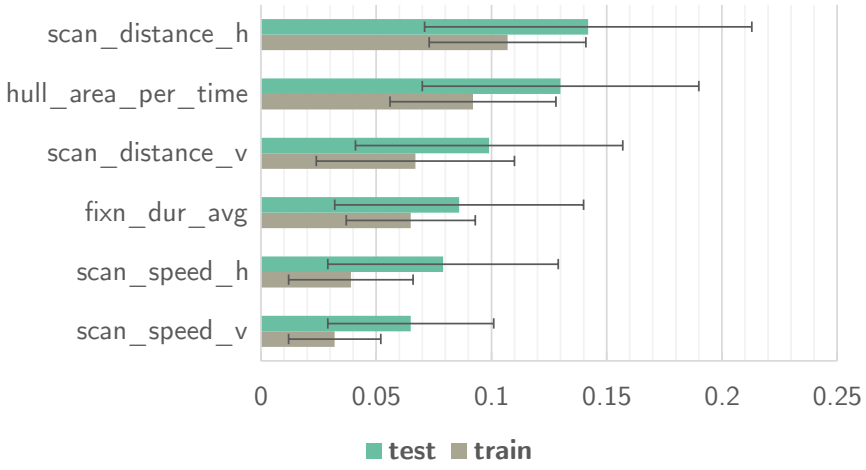
Figure 7.4: The heatmap (right) shows the pairwise Spearman rank-order correlations for our features using all samples of the g-REL data. The dendrogram (left) shows feature groupings based on their correlation-based distances. Setting the distance threshold to $t = 0.3$ yields six feature clusters (see colored leaves).

based distances are visualized in figure 7.4 (*all* samples of the g-REL data). We set the distance threshold to $t = 0.3$ for the feature importance analysis, for which we obtain six feature clusters as indicated by the colored leaves of the dendrogram. We obtain the same feature clusters for the *agree* subset and for both subsets of the Google NQ data. Using one feature per cluster to train and evaluate the SVC* model, we observe a drop in f1 scores of 0.015 for the *all* subset and no decline for the *agree* subset. These representative features include `fixn_dur_avg`, `scan_speed_h`, `scan_speed_v`, `scan_distance_v`, `scan_distance_h`, and `hull_area_per_time`. We remain at $t = 0.3$ because higher thresholds lead to substantially lower f1 scores and to differences in the resulting feature clusters between subsets and corpora.

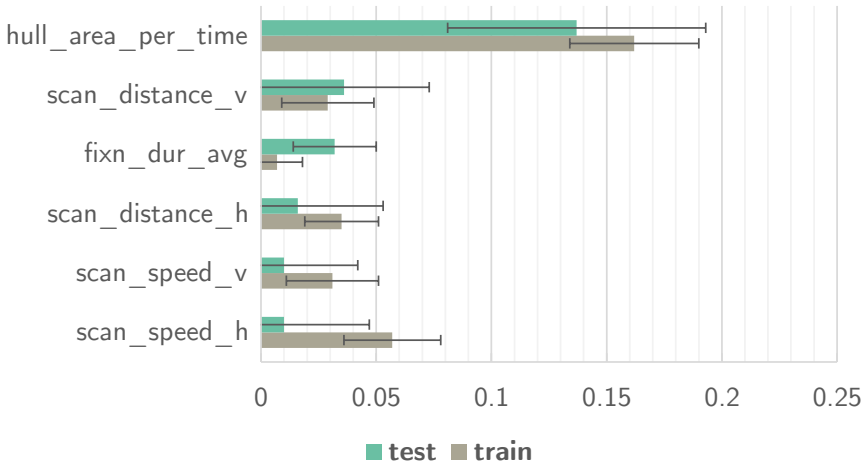
The importance of feature clusters is visualized in figure 7.5. For the *all* subset, we observe f1 losses ranging from 0.065 for `scan_speed_v` and 0.142 for `scan_distance_h` for the test set. We observe slightly lower losses for the train set but the same importance ranking. Eventually, the features `scan_distance_h` and `hull_area_per_time` are most important when using *all* samples. For the *agree* subset, `hull_area_per_time` is by far the most important feature with an f1 loss of 0.162 on the train set and 0.137 of the test set. The features `scan_distance_v` and `fixn_dur_avg` are somewhat important with losses of 0.036 and 0.032. For `scan_speed_h`, we observe a higher importance on the train set (0.057) than on the test set (0.01), which may indicate that this feature causes the model to overfit the training data. Overall, the `hull_area_per_time` feature introduced by Bhattacharya et al. (2020a) is of high importance for modeling the perceived paragraph relevance and stable when including *topical* samples and samples for which the user rating disagrees with the ground truth. The remaining five features are important when including all samples, particularly the `scan_distance_h`. This result suggests that, in the first stage, these five features could be used to identify *topical* (irrelevant) samples and, in the second stage, the `hull_area_per_time` can predict paragraphs perceived as relevant among the remaining, non-topical samples.

7.3.2 Application to Adaptive User Interfaces

Our relevance estimation method can enable the development of adaptive user interfaces (UIs) that emphasize relevant contents or suppress irrelevant ones similar to Feit et al. (2020). Over time, their system detects relevant and irrelevant elements of a UI that show different records of flat advertisements: a certain UI element always shows the same type of information, which depends on the currently viewed flat record. Our use case differs in that we want to highlight relevant text passages of a document or hide irrelevant ones. Adaptations may be based on perceived relevance estimates from recent eye movements and could, e.g., ease revisiting of relevant paragraphs in a document by immediately highlighting them or by hiding irrelevant passages. Alternatively, collecting relevant and irrelevant text passages in the pass of a search session may allow an adaptive UI to properly format text passages of documents hitherto unseen by the user. An adaptation method requires a precise recognition of relevant (true positive) or irrelevant (true negative) paragraphs to emphasize or suppress them, respectively. Misclassifications would lead to incorrect adjustments and, subsequently, to usability problems. Emphasizing irrelevant content (false positive) or suppressing relevant content (false negative) is likely to have a stronger negative impact on the user interaction than failing to suppress irrelevant content or to highlight

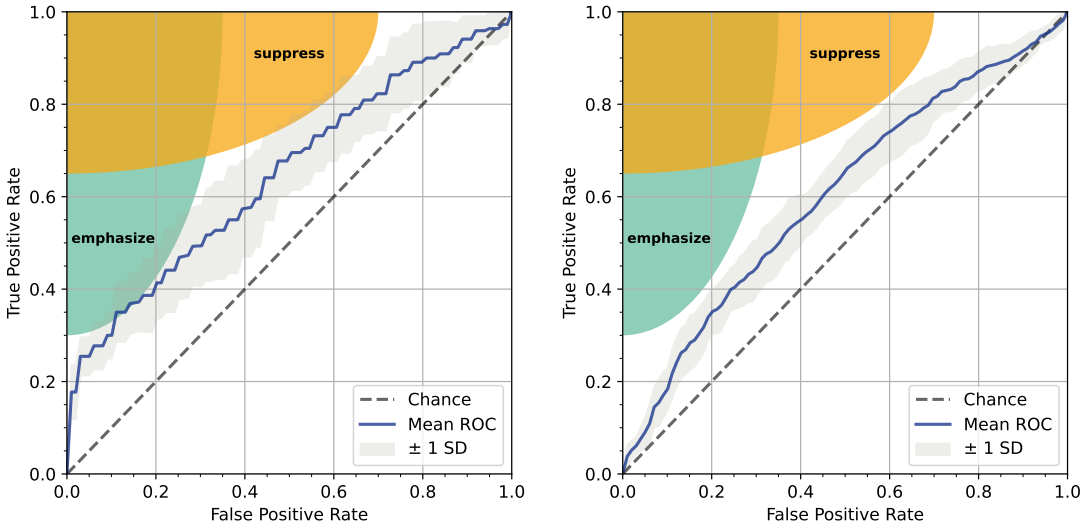


(a) *all* subset



(b) *agree* subset

Figure 7.5: Permutation feature importance in terms of the f1 loss for the *all* subset 7.5a and the *agree* subset 7.5b of the g-REL data. We show the mean loss in f1 scores from 30 permutation iterations ± the standard deviation. We include the feature importance estimates for the train and test split.



(a) g-REL **(b)** Google NQ

Figure 7.6: Receiver Operator Characteristic curves for the SVC* relevance prediction models trained and tested using *all* samples from the g-REL data 7.6a and the Google NQ data 7.6b. The curves are averaged over the 10 training and test cycles; the gray area indicates the standard deviation (± 1 SD). The elliptic areas indicate acceptable regions for true and false positive values for emphasizing or suppressing contents based on Feit et al. (2020).

a relevant one (Feit et al., 2020). To avoid strong negative impacts, adjustments by accentuation require a relevance model with a low false positive rate (FPR), and adjustments by suppression require a model with a high true positive rate (TPR), i.e., with a low number of false negatives. Depending on the type of adjustment, the TPR and FPR could be traded against each other by using different decision thresholds. We show possible trade-offs for our SVC* models using ROC curves. One model is trained on *all* g-REL data and one on *all* Google NQ data (see figure 7.6). We do not consider other subsets for realistic application scenarios because we would not be able to determine whether a user agreed with the actual (system) relevance of a paragraph or whether a text passage was on topic but irrelevant (*topical*). This differentiation, which is aligned to the approach by Bhattacharya et al. (2020a), requires prior knowledge about the paragraphs and was meant to identify *topical* samples as being the most challenging cases for classification algorithms. Analogous to Feit et al. (2020), the shaded areas in our ROC plots in figure 7.6 indicate acceptable true and false positive rates for emphasizing or suppressing contents. For g-REL data, the ROC curve of the SVC* model hits the *emphasize* area, indicating that it could be used to emphasize short news articles perceived as relevant if the decision threshold is tuned accordingly. But, many relevant contents would be missed, indicated by the low true positive rate (recall). The shaded areas also reveal that our models are unsuitable for other adjustments.

7.3.3 Reading Model Assessment Tool (ReMA)

We develop the interactive **Re**ading **M**odel **A**ssessment tool (ReMA), an interactive tool for analyzing scanpaths from relevance judgment tasks and for assessing gaze-based relevance estimation models (Valdunciel et al., 2022). Our tool allows experimenters to easily browse recorded trials from the gazeRE dataset, compare the model output to the ground truth, and visualize gaze-based features at the token- and paragraph-level that serve as model input. Our goal is to facilitate an understanding of the relation between eye movements and the human relevance estimation process and to understand the strengths and weaknesses of our presented model. Our data exploration tool is related to others in the context of reading behavior analysis. For instance, GazePlot enables reading performance analysis for children (Špakov et al., 2017). EyeMap allows researchers to analyze fixations and saccades at the word level (Tang et al., 2012). Both offer a scanpath visualization as text overlays: eye tracking data is presented as a gaze plot in which fixations are depicted as circles and saccades as lines. However, "actual scanpath records are usually quite complex, and can be difficult to interpret and compare" (Goldberg and Helfman, 2010). Other tools implemented more intuitive visualizations which, for example, aggregate the gaze data at the word level by mapping the gaze data to objects of the Document Object Model (DOM) of a web page (Reeder et al., 2001; Beymer and Russell, 2005; Hienert et al., 2019). WebEyeMapper and WebLogger (Reeder et al., 2001) and WebGazeAnalyzer (Beymer and Russell, 2005) introduced this approach. Hienert et al. (2019) use a similar mapping approach in the Reading Protocol tool. It allows experimenters to analyze eye movements more effectively on arbitrary areas of interest. They use this gaze-to-object mapping to generate a heat map that visualizes the summed fixation durations at the word level. Davari et al. (2020) use this tool to investigate the role of word fixations in query term prediction. Buscher et al. (2012) introduced the concept of attentive documents that keep track of a user's perceived relevance

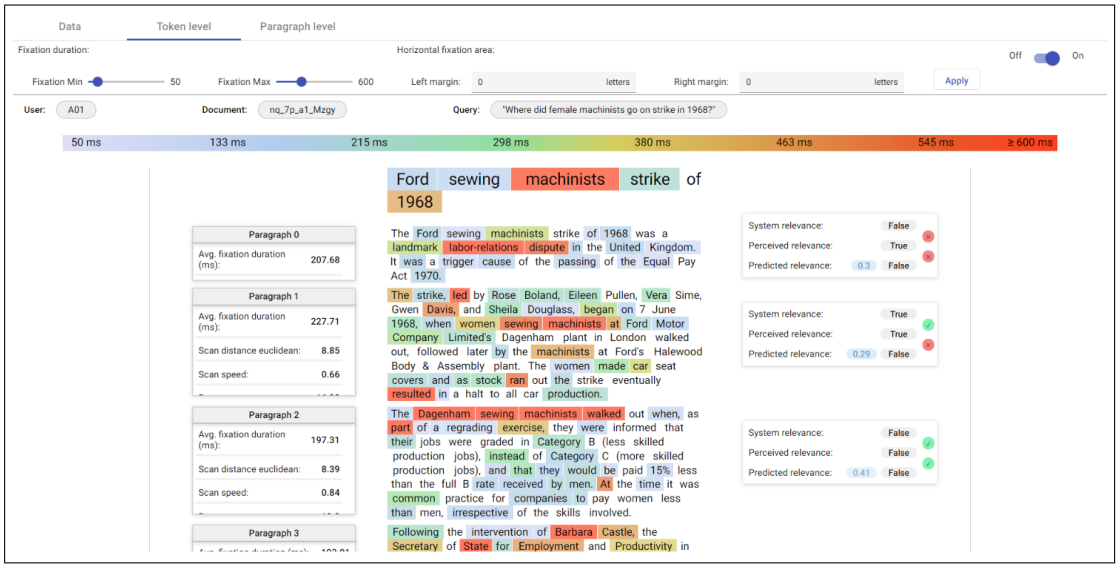


Figure 7.7: Screenshot of ReMA, our interactive reading model assessment tool. It shows a Wikipedia article with the token-level heat map, paragraph-level features, and relevance information per paragraph.

based on its eye movements. Their system may highlight text passages that were previously read and not skimmed. Eyekit is a recent Python package that supports the analysis of reading behavior (Carr et al., 2021). It provides different visualization techniques like gaze plots and character-based heat maps, which work similarly to the word-based heat map from Hienert et al. (2019). Moreover, it offers gaze-based feature extraction from scanpaths originating from a reading activity.

Our goal was to create a tool for interactively exploring the recorded trials of the gazeRE dataset and to assess our relevance-judgment models that take a scanpath as input to predict the perceived relevance. The recorded trials include a text-based stimulus, a user's scanpath, and the system relevance and perceived relevance. As it is rarely helpful to visualize the fixation and saccade sequences to understand and compare scanpaths (Goldberg and Helfman, 2010), we show the text-based stimulus along with paragraph-level features and a token-level heat map. Our tool shows the predicted relevance estimate and highlights whether it agrees or disagrees with the perceived relevance (ground truth), which allows us and other researchers to assess the model more efficiently and understand its strengths and weaknesses. In the long run, such a tool may enable researchers to build more effective models for gaze-based relevance estimation. The user interface shows a single trial with the text-based stimulus as its central element (see figure 7.7). A trial can be selected via the *Data* tab by providing a participant's acronym and a document ID. The corresponding query is shown below the tab area. In the *Token level* and *Paragraph level* tabs, users can configure the visualization, including the token-level heat map, the paragraph-level feature display, and the relevance ratings. The size and position of each token correspond to the original layout from the user study. However, we scale the layout depending on the resolution of the connected display. The web-based front end is developed

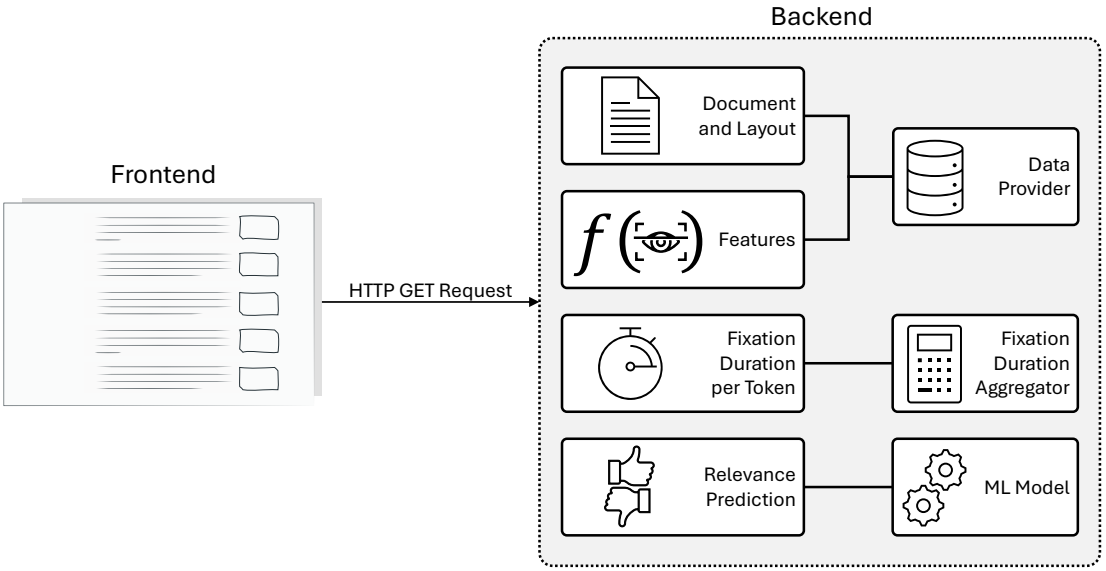


Figure 7.8: Architectural overview of our interactive model assessment tool. The backend serves all information to the front end, including the stimuli, the extracted features, and the model predictions.

using AngularJS, HTML, CSS, and Typescript. It is responsive and can adapt its layout to different screen sizes. All data can be queried from a single backend that connects the example dataset (see figure 7.8). It offers functions for loading a document, the token-level heat map, and extracted features per paragraph. Also, it integrates the pre-trained machine learning model to provide relevance estimates for a selected trial. The backend REST API is implemented using the Python framework Flask.

7.3.3.1 Token-level Heat Map

We integrate the token-level heat map, which encodes a scanpath by accumulating fixation durations per token, as proposed in Hienert et al. (2019). We extend the heat map generation by a configurable perceptual span setting, which allows experimenters to estimate the information that was actually processed by a reader more accurately. The heat map is generated by coloring the background of each token based on the duration of all fixations (sum) that hit the token area. The range of considered fixation durations can be set using two sliders in the *Token level* tab as shown in figure 7.7. The resulting color legend is shown above the stimulus and replicates the coloring in Hienert et al. (2019). Tokens that were fixated shorter than the minimum threshold are not colored. Fixation durations longer than the maximum threshold result in a red background color. However, the region from which useful information is acquired during a fixation, also known as the perceptual span, is wider than a single character (Rayner et al., 2010). Research using the classical paradigm of the gaze-contingent moving window has shown that in English and other alphabetic languages read from left to right, the perceptual span extends 3-4 letters to the left and up to 14-15 letters to the right for a given fixation point (McConkie and Rayner,

1976). But, the extent of the perceptual span is not constant: its size is influenced by linguistic parameters such as the readability of the text (Rayner, 1986), the frequency of words (Rayner et al., 2003), and the linguistic ability of the reader (Choi et al., 2015). We extend the token-level heat map to account for the perceptual span (see figure 7.7): Our tool allows researchers to configure a perceptual span by setting a left margin m_l and a right margin m_r, which defines how many letters to the left and right of fixation are considered for accumulating the fixation durations per token. The current fixation duration is added for all tokens which overlap with this region. Hereby, one letter translates to a fixed number of pixels based on the font size.

7.3.3.2 Paragraph-level Features

Per paragraph, we display a box with features extracted from the longest partial scanpath for this paragraph to its left. To encode a user's eye movements for a certain paragraph, we have to extract coherent gaze sequences within the paragraph area. We refer to these partial scanpaths as visits. A user might visit a paragraph multiple times during the relevance judgment process. We extract all visits to a paragraph with a minimum length while ignoring short gaps: As long as there are two subsequent visits with a gap shorter than 0.2 s, these visits are merged. All visits that satisfy a minimum length of 3 s are kept. A common way to encode a scanpath in a meaningful way is to extract handcrafted features like the `scan_hv_ratio`: the horizontal to vertical ratio of saccade amplitudes (Holmqvist and Andersson, 2017, p. 442). We extract a set of 17 features, including the `scan_hv_ratio`, which have been used to model the perceived relevance of short news articles in Bhattacharya et al. (2020a). Four features are based on fixation events, eight are based on saccadic movements, and five are based on the area spanned by all fixations. The source code for feature extraction is available on GitHub[8]. In the *Paragraph level* tab, researchers can select features that shall be displayed.

7.3.3.3 Relevance Model Assessment

Per paragraph, our ReMA tool shows the system relevance, the perceived relevance collected in the user study, and the predicted relevance using the pre-trained model. The relevance values can be either *True* or *False*, indicating whether this paragraph is deemed relevant to the corresponding query. We display the model's certainty in terms of its probability estimate $p \in [0, 1]$ for the *True* class (relevant): 1 indicates that the model is certain that the trial includes a relevant paragraph, 0 indicates certainty for a non-relevant instance. The certainty is shown in blue with an opacity proportional to its value, i.e., the closer p is to 0 or 1, the more opaque the value is shown. We also show circular badges indicating whether the participant's perceived relevance agrees with the system relevance and whether the model correctly predicted the perceived relevance. This enables a more efficient assessment of the model performance in context.

[8] `https://github.com/DFKI-Interactive-Machine-Learning/gazeRE-dataset/tree/main/features` (accessed on 12 Dec 2024)

7.4 Conclusion

In this work, we investigated whether we can confirm the findings from Bhattacharya et al. (2020a) that gaze-based features can be used to estimate the perceived relevance of short news articles read by a user. Further, we investigated whether the approach can be applied to multi-paragraph documents that require the user to scroll down to see all text passages. For this, we conducted a user study with n=24 participants who read documents from two corpora, one including short news articles and one including longer Wikipedia articles in English, and rated their relevance at the paragraph-level with respect to a previously shown trigger question. We used this data to train and evaluate machine learning models that predict the perceived relevance at the paragraph-level using the user's eye movements as input. Our results showed that, even though we achieved lower model performance scores than Bhattacharya et al. (2020a), we could replicate their findings under the same experiment conditions: eye movements are an effective source for estimating the perceived relevance of short news articles if we leave out articles that are on topic but irrelevant. However, we could not clearly show that the approach generalizes to multi-paragraph documents. In both cases, the best model performance was observed when using over-sampling and feature scaling on the training data and a support vector classifier with an RBF kernel for classification. Future investigations should aim to overcome the limited estimation performances. A potential solution could be to use higher-level features such as the *thorough reading ratio*, i.e., the ratio of read and skimmed text lengths (Buscher et al., 2012), or the *refixation count*, i.e., the number of re-visits to a certain paragraph (Feit et al., 2020). Another solution could be found in using scanpath encodings based deep learning (Castner et al., 2020; Bhattacharya et al., 2020b). We envision the gaze-based relevance detection to be a part of future adaptive UIs that leverage multiple sensors for behavioral signal processing and analysis (Oviatt et al., 2018; Barz et al., 2020b,a). We published our new gazeRE dataset and our code for feature extraction under an open-source license on GitHub to enable other researchers to replicate our approach and to implement and evaluate novel methods in the domain of gaze-based implicit relevance feedback.

Chapter 8

Visual Attention Modelling

Eye tracking studies often consider visual attention to specific areas of interest (AOIs) to analyze and understand how people process visual information. AOIs are specific regions in a scene or interface that are defined by researchers (Holmqvist and Andersson, 2017). Visual attention refers to the time a person pays attention to these regions. By measuring visual attention to and transitions between AOIs during a study, researchers can gain insights into which elements of a scene are relevant to an activity and how interventions of an experiment influence the participant's eye movement behavior. This is usually done based on fixation events as they are assumed to approximate a person's allocation of cognitive resources through the time they spend processing a visual scene (Just and Carpenter, 1980). Further, advances in modern head-worn eye tracking technology (Tonsen et al., 2017) can enable attention-aware mobile human-computer interfaces. In remote eye tracking with static stimuli such as images, an AOI can be defined once and reused for every participant. Dynamic AOIs in video-based stimuli can be annotated using keyframe-based annotation techniques, i.e., AOIs are marked via bounding boxes for keyframes, and interpolation is used to annotate intermediate frames (Kurzhals et al., 2014b). However, these efficient fixation-to-AOI mapping techniques from remote eye tracking do not scale for mobile eye tracking applications. Accurately annotating mobile eye tracking data remains a challenging and time-consuming task because scene videos taken with a head-mounted eye tracking device are unique for every participant. In mobile eye tracking practice, one or more annotators decide per fixation whether an AOI was hit or not (Uppal et al., 2022; Kurzhals et al., 2014a). This fixation-wise annotation approach reduces the annotation effort compared to a video frame-based annotation because fixations last around 200-400 ms (Holmqvist and Andersson, 2017), which corresponds to 2-2.5 events per second. Videos are typically recorded with a sampling rate of at least 30 Hz. Still, it does not remedy the need to annotate AOIs in every single recording and hinders the development of attention-aware mobile interfaces.

Attaching fiducial markers to target stimuli was proposed as a solution in research (Yu and Eizenman, 2004; Pfeiffer et al., 2016; Mehlmann et al., 2014) and was adopted in modern commercial software solutions like Pupil Cloud[1]. However, markers are obtrusive and may impact the visual scanning behavior. Therefore, the present research aims at a solution for non-instrumented

[1] `https://pupil-labs.com/blog/pupil-cloud-projects-enrichments/` (accessed on 02 Feb 2024)

environments. Existing approaches for automatic or semi-automatic analysis of head-mounted eye tracking data use computer vision models to map fixations to AOIs. Most of these approaches rely on pre-trained computer vision models that do not allow for adapting the underlying model to a certain target domain (Sümer et al., 2018; Machado et al., 2019; Deane et al., 2022; Venuprasad et al., 2020; Uppal et al., 2022). These can be applied in very constrained settings only, i.e. if the dataset used for training the machine learning model matches the target domain. Some approaches support a single, a priori model training or fine-tuning step for adaptation to a target domain (Wolf et al., 2018; Kumari et al., 2021; Panetta et al., 2019). These approaches offer no possibility of adapting the model during the annotation process and, hence, suffer from a lack of flexibility. Further, not all methods are evaluated quantitatively (Barz and Sonntag, 2016; Kurzhals et al., 2017; Brône et al., 2011) or evaluation metrics are not properly described (Pontillo et al., 2010; Machado et al., 2019) or inadequate, e.g., ignoring temporal aspects (Panetta et al., 2019). Some commercial tools offer automatic mapping of the gaze signal in world video coordinates to a reference frame that defines AOIs, such as the assisted mapping function of Tobii Pro[2]. However, this is only possible for a limited number of reference frames.

We aim to develop a method for semi-automatic mapping of fixations to AOIs, which enables efficient analysis and interpretation of humans' complex interaction behavior. This bears the potential to boost the efficiency in research based on eye tracking by automating the time-consuming and expensive data annotation process (Panetta et al., 2019) and to facilitate novel real-time adaptive human-computer interaction (Huang and Mutlu, 2016; Barz et al., 2021b). In the first step (see section 8.1), we develop and evaluate two methods for automatically detecting visual attention to ambient objects based on pre-trained computer vision models (Barz and Sonntag, 2021). Our goal is to systematically assess the ability of pre-trained models to map fixations to AOIs and, by that, establish a robust baseline for this task. In the second step (see section 8.2), we aim to break the limitations of using pre-trained models, i.e., the issue of lacking flexibility and quality assurance through humans-in-the-loop. We implement and evaluate *eyeNotate*, a user interface that enables semi-automatic annotation of mobile eye tracking data (Barz et al., 2025). Our tool allows mobile eye tracking practitioners to manually annotate their recordings fixation-wise, reflecting the current state-of-the-art and representing our baseline approach. Further, we implement an extension offering fixation-to-AOI mapping suggestions using a few-shot image classification model, which was shown to be successful in another use case (Desmond et al., 2021). This model can learn from user feedback, i.e., when users accept or reject/correct suggestions, following the interactive machine learning (IML) paradigm. IML combines frequent human input and feedback with machine learning technologies without requiring background knowledge in machine learning (Dudley and Kristensson, 2018; Sonntag et al., 2024). Domain knowledge from end-users, like eye tracking practitioners, can be integrated more effectively into complex applications. However, it is important to thoroughly design such systems to achieve better user experiences and more effective learning systems (Amershi et al., 2014). We conduct a case study with n=3 trained annotators to compare the baseline version and the IML-supported approach. We measure the perceived usability, annotation validity and reliability, and efficiency during a data annotation task using an existing mobile eye tracking dataset with ground-truth annota-

[2] `https://connect.tobii.com/s/article/how-to-perform-manual-and-assisted-mapping` (accessed on 12 Dec 2024)

tions (n=48). We ask participants to re-annotate data for one individual in this dataset. After task completion, we conducted a semi-structured interview (SSI) to understand how participants used the provided IML features. In addition, we investigate the performance in automatically annotating the remainder of the dataset using our resulting machine learning models. In this chapter, we contribute as follows:

- Section 8.1: We develop and evaluate two methods for automatically mapping fixations to AOIs based on pre-trained computer vision models.
- Section 8.2: We implement and evaluate *eyeNotate*, a user interface that enables semi-automatic annotation of mobile eye tracking data.

We develop two methods for automatically detecting visual attention to ambient objects using head-mounted eye trackers in combination with pre-trained computer vision models in chapter 8. We investigate their effectiveness for the automatic annotation of mobile eye tracking data from diagnostic user studies, i.e., for automatic mapping of fixations to areas of interest of that study. Further, we develop an interactive machine learning interface that enables semi-automatic annotation of mobile eye tracking data based on few-shot image classification. It addresses the limited flexibility and accuracy when using pre-trained models.

8.1 Fixation-to-AOI Mapping with Pre-trained Models

We implement two methods for automatically detecting visual attention to a visual stimulus in a scene. Both take the video feed and the corresponding gaze or fixation signal as input and predict if the participant paid attention to an AOI for each frame (see figure 8.1). We contribute by (i) implementing two methods for detecting visual attention using eye tracking data and pre-trained deep learning models for image classification and object detection; and (ii) evaluating the performance of our methods using the VISUS dataset (Kurzhals et al., 2014a) and fine-grained activity recognition metrics in a systematic way (Ward et al., 2011).

8.1.1 Method

The first method, *IC*, aggregates classifications of image patches cropped around the gaze signal using a pre-trained image classification model similar to the gaze-guided object classification system by Barz and Sonntag (2016). The second method, *OD*, matches fixation events with the result of a pre-trained object detection model similar to Wolf et al. (2018) and Machado et al. (2019). In this work, we concentrate on pre-trained computer vision models, similar to Barz and Sonntag (2016) and Machado et al. (2019), to explore when models without a training overhead can be applied effectively and when they reach their limits. We leave fine-tuning of the models as a task for future work because it is outside the scope of this paper. Both methods are implemented in Python using the *multisensor-pipeline*[3] package for flexible streaming and processing of signals from one or multiple sources (see chapter 9). It allows us to easily set

[3] `https://github.com/DFKI-Interactive-Machine-Learning/multisensor-pipeline` (accessed on 12 Dec 2024)

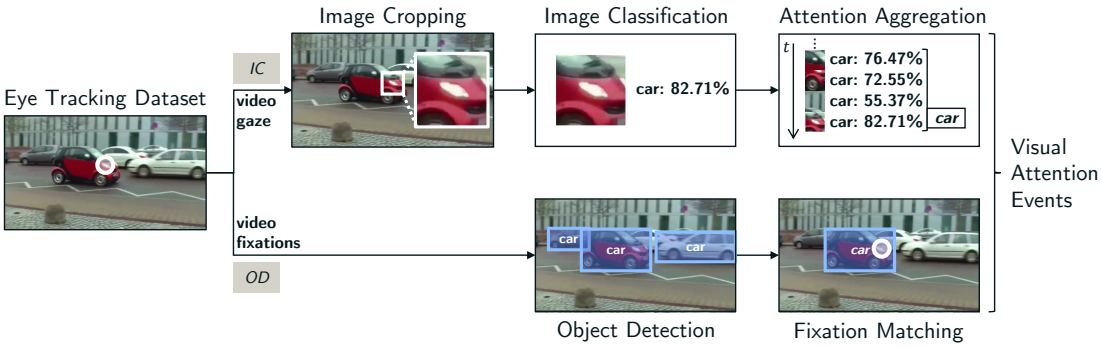


Figure 8.1: Processing workflow of the two proposed methods for automatic attention detection: *IC* is based on image classification and gaze samples, *OD* uses object detection and fixation events. Both methods support visual attention detection in real-time.

up real-time applications using source modules for connecting sensor input, processor modules for manipulating or aggregating incoming data streams and events, and sink modules for, e.g., storing and visualizing the output. In the following, we describe the implementation of both methods and their adjustable parameters.

8.1.1.1 Detect Attention Using Gaze-Guided Image Classification (IC)

Our method based on image classification includes four subsequent steps. First, we re-sample the gaze signal to 5 Hz and crop an image patch of 200×200 pixels from the egocentric video feed (1920×1080 pixels) per remaining sample. We use this crop size because it turned out to perform well in real-time applications (see Barz and Sonntag (2016); Barz et al. (2021b)), and the size fits well to the AOIs in the VISUS dataset (manual inspection). Second, each patch is classified using a pre-trained version of the ResNet image classification model (He et al., 2016), which is trained on the ImageNet dataset with 1001 object classes (Russakovsky et al., 2015). The prediction result includes the top-5 class candidates and their probability. In the third step, we aggregate similar class labels by accumulating their probabilities. We merge similar object classes based on a manually defined lookup table. For example, if the top-5 output includes the ImageNet classes `passenger car`, `streetcar` and `limousine`, we replace the probability of `passenger car` by the sum of all three probabilities and remove the remaining class labels from the output. In the last step, we implement a working memory- and threshold-based attention detection algorithm similar to Barz and Sonntag (2016) and Toyama et al. (2012) using the top-1 predictions of the previous step as continuous input: An update routine is called for each incoming prediction, i.e., a tuple including a unique class label and the corresponding probability output of the model: $(c, p(c))$. If $p(c)$ exceeds the minimum probability T_p, we increase the duration counter C_{dur} at the index c by the number of milliseconds that passed since the last run of the update loop (circa 200 ms). We increase the noise counter C_{noise} by the same amount of time for all other classes with a non-zero duration count. If the aggregated duration $C_{dur}[c]$ exceeds the duration threshold T_{dur}, we send an *attention started event* including c, $p(c)$ and the timestamp of the latest prediction. In addition, we store c as the currently attended class c_{active} and reset both counters for it: we

set $C_{dur}[c]$ and $C_{noise}[c]$ to zero. If c_{active} is not empty and not equal to c, we consider the prior attention event to be over and send an *attention ended event*. We send an *attention confirmed event* is c is equal to c_{active}. Finally, we check whether the aggregated noise duration in C_{noise} exceeds the noise threshold T_{noise} for all remaining classes. In this case, we reset both counters for this class, and if this class is equal to c_{active}, we send an *attention ended event*. We subtract T_{dur} from the event timestamp to match better the actual start and end times of the attention events. We offer a parameter for setting the image classification *model*: we include pre-trained ResNet-50[4] and ResNet-152[5] models via Tensorflow Hub. Any other model from this platform that was trained using ImageNet can also be used by providing a corresponding link. The default setting is $T_{dur} = T_{noise} = 300$ ms, $T_p = 40\%$, and the *model* is set to ResNet-152. We refer to this setting as IC-152-300-40 (in general: IC-*model*-$T_{dur/noise}$-T_p).

8.1.1.2 Detect Attention Using Object Detection (OD)

Our second method is based on an object detection model that can detect multiple object instances in an image from a set of candidate classes. To detect visual attention, we match the position of fixation events from the eye tracker with detected object regions. For each fixation, we extract an image frame from the video feed that is closest to the start of the fixation event. The object detection takes longer per image than the image classification algorithm. However, this method can still be applied in real-time because it is applied once per fixation. During a fixation, the eye is relatively still and should point to the same location in the world space. However, fixation detection is imperfect, e.g., in the presence of smooth pursuit movements, which makes this method dependent on the quality of the applied fixation detection algorithm. Next, we detect all object instances in the current image frame: we use a Mask R-CNN model (He et al., 2020) that is pre-trained on the MS COCO dataset (Lin et al., 2014) with the Detectron2 framework (Wu et al., 2019)[6]. For each instance, it provides a class label with a probability value, as well as a rectangular bounding box and a pixel-wise segmentation mask depicting the object area. Finally, we check whether the fixation position lies within the object area, either using the bounding boxes (*bbox*), similar to Machado et al. (2019), or the more fine-grained segmentation masks (*mask*), similar to Wolf et al. (2018), as reference. This can be configured via the *object mask* parameter that defaults to *bbox*. If a hit is detected, we send an *attention started event* using the start time of the fixation and an *attention ended event* using its end time. We choose the one with the highest probability if two object areas are hit. We refer to the two possible settings as OD-bbox (default) and OD-mask.

8.1.2 Dataset

We use the VISUS dataset for our evaluation (Kurzhals et al., 2014a), which contains eye tracking data from 25 participants for 11 video stimuli, totaling 275 sessions[7]. The gaze data was recorded

[4] `https://tfhub.dev/google/imagenet/resnet_v2_50/classification/4` (accessed on 12 Dec 2024)

[5] `https://tfhub.dev/google/imagenet/resnet_v2_152/classification/4` (accessed on 12 Dec 2024)

[6] Model weights: `https://dl.fbaipublicfiles.com/detectron2/COCO-InstanceSegmentation/mask_rcnn_R_101_FPN_3x/138205316/model_final_a3ec72.pkl` (accessed on 12 Dec 2024)

[7] `https://www.visus.uni-stuttgart.de/publikationen/benchmark-eyetracking` (accessed on 12 Apr 2021)

using a Tobii T60 XL remote eye tracker at 60 Hz. The authors did not report the spatial accuracy and precision as measured during their recordings. The video stimuli have a resolution of 1920×1080 pixels at 25 frames per second and have an average length of 75.55 s ($SD = 59$). Each video is manually annotated with axis-aligned rectangular bounding boxes from two annotators for 1 to 6 AOIs per video (see table 8.1). Bounding boxes were set at keyframes and interpolated for intermediate frames. The main purpose of the dataset is to serve as a benchmark for visualization and analysis techniques in the field of eye tracking. We use the dataset as a benchmark dataset for automatic detection of visual attention to dynamic AOIs. We treat the fixation events reported in the dataset that hit the manually defined bounding boxes as ground truth attention events to the respective AOIs. If two AOIs in a single frame are hit, we select the AOI that yields the longer event. While the VISUS dataset is acquired with a remote tracking device, we use it to approximate mobile eye tracking recordings: we do not leverage that the videos are the same for each participant. In the following, we describe the ground truth extraction, the scenarios (video stimuli) and AOIs, and we describe the related challenges for gaze to AOI mapping.

8.1.2.1 Scenarios & Challenges

The dataset includes 11 scenarios, each with a different kind and number of AOIs. They pose multiple challenges to attention detection methods. In the simplest case, a method must map gaze to AOIs representing distinct concepts (challenge I). This applies to, e.g., `01-turning car` in which a single AOI, a "red car", is shown, and to `07-kite` with two distinct AOIs: a "person" flies a "kite". The difficulty increases if two AOIs in a scenario refer to the same concept (challenge II). For instance, the scenario `01-car pursuit` shows a "red car" driving through a turning area, with a "white car" on the opposing lane and multiple parking cars in the background. The challenge is not only to detect that a car is fixated but to differentiate between the two prominent cars (AOIs) and the background cars, which are multiple instances of the same concept. Similarly, scenarios 03, 08, 09, and 11 require the ability to differentiate between multiple instances of the concept person, for instance, a "hooded" person, a person wearing a "red shirt and hat", and several distractor "persons" in scenario `11-person search`. The problem becomes more complex if two AOIs share a concept and their appearance (challenge III). An example can be found in scenario `04-thimblerig`. It includes three cups with identical appearance. Distinguishing them requires object tracking for multiple instances and an initial assignment of each instance to an AOI by hand. The aforementioned cases do not cover the scenario `05-memory`. It shows a memory game: in the beginning, all 16 "cards" look the same, while, until the end of the game, we see 8 pairs of cards with different visual appearances per pair. Yet, all "cards" count toward the same AOI. The challenge is if the appearance of an AOI changes over time (challenge IV).

8.1.2.2 Mapping Class Labels to AOIs

Our methods aim to solve the aforementioned challenges using pre-trained computer vision models. For this, AOIs need to be mapped to class labels of ImageNet for the *IC* method and of MS COCO for the *OD* method. We assume that the performance per scenario depends on the type of AOIs and whether they are represented in the model's training data. If no matching class label

Scenario	AOI	ImageNet Labels	MS COCO Labels
`01-car pursuit` (25 s)	red car	streetcar, sports car, minivan, cab, minibus, limousine, car mirror, racer, passenger car	car
	white car	–	–
`02-turning car` (28 s)	red car	streetcar, sports car, minivan, cab, minibus, limousine, car mirror, racer, passenger car	car
`03-dialog` (19 s)	left face	ear	person
	right face	–	–
	shirt	sweatshirt	–
`04-thimblerig` (30 s)	cup1	cocktail shaker, coffee mug, cup	cup
	cup2	–	bowl
	cup3	–	–
`05-memory` (148 s)	cards	desk	dining table
`06-UNO` (121 s)	left hand	–	person
	right hand	–	–
	stack covered	desk	dining table
	stack uncovered	–	–
`07-kite` (97 s)	person	lab coat, poncho, cardigan, cloak, sweatshirt, trench coat	person
	kite	balloon, kite, parachute	kite
`08-case exchange` (27 s)	persons	sombrero, cowboy hat	person
	textbox	–	–
	case	mailbag, packet, plastic bag, shopping basket, backpack, bucket, crate	handbag, suitcase
	suspects	lab coat, poncho, cardigan, cloak, sweatshirt, trench coat	–
`09-ball game` (31 s)	ball	baseball, basketball, rugby ball, tennis ball, volleyball, soccer ball	sports ball
	player white	ballplayer	person
	player red1	–	–
	player red2	–	–
	player red3	–	–
`10-bag search` (133 s)	red bag	plastic bag	handbag
	yellow bag	–	–
	blue bag	–	–
	red-white bag	–	–
	brown bag	mailbag	–
	persons	lab coat, poncho, cardigan, cloak, sweatshirt, trench coat	person
`11-person search` (172 s)	hooded	lab coat, poncho, cardigan, cloak, sweatshirt, trench coat	person
	red shirt and hat	sombrero, cowboy hat	–
	persons	–	–

Table 8.1: Overview of scenarios and AOIs in the VISUS dataset and the corresponding mappings of class labels to AOIs. Class labels originate from ImageNet in case of *IC* methods and from MS COCO in case of *OD* methods.

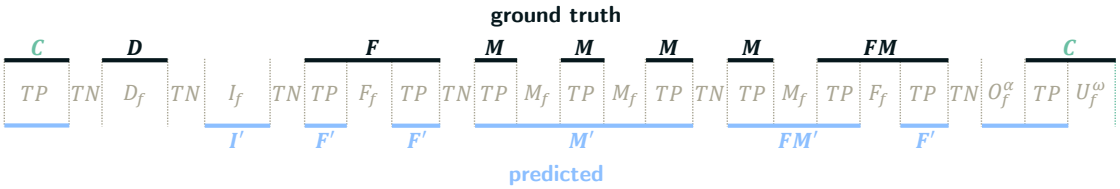


Figure 8.2: Example of segmented ground truth events and predicted events with annotations for event error and frame error classes. The vertical bars depict the segment boundaries. The frame error classes are given per segment.

exists for an AOI, none of the methods can detect respective attention events. If a class label matches multiple AOIs of a scenario, i.e., if they share a concept, we can only assign the label to one of them. This probably leads to an increase in false positives. The performance might also suffer from inadequate matches. This experiment uses a separate mapping from class labels to AOIs for each method and scenario, as shown in table 8.1. For *IC* methods, we identified ImageNet labels for 19 AOIs including adequate matches like *passenger car* for the AOI "red car", but also weak matches like *sweatshirt* as a proxy for the AOI "person". Similarly, we found MS COCO labels for 17 AOIs for the *OD* methods. For instance, *car* is an adequate match for the AOI "red car", while *dining table* is a weak match for the "stack covered" in `06-UNO` (the stack is located on a table).

8.1.3 Metrics

To quantify the performance of our methods, we need evaluation metrics that depict how well our detected attention events match the ground truth events. We reviewed the metrics proposed in closely related works, but none of them was fully satisfactory: Panetta et al. (2019) compared their system to manual ground truth annotations by calculating the distance between two histograms that aggregate the duration of fixations from predicted or ground truth AOI regions, respectively. However, their metric does not punish if detected AOI fixations are shifted in time or if they occur in the wrong order, which puts the validity of their metric into question. For instance, the histogram would be equal if the predicted events were reported reversely. Machado et al. (2019) reported accuracy and precision, but whether they compute the metrics frame-wise or event-based is unclear. Toyama et al. (2012) reported event-based precision and recall for each method: precision reflects how many of the detected attention events were classified correctly, and recall indicates the proportion of detected attention events to all attention events. Similarly, De Beugher et al. (2014) reported precision and recall, but at the frame-level. Wolf et al. (2018) and Batliner et al. (2020) reported the recall (true positive rate) and the specificity (true negative rate) at the frame-level, including one frame per fixation for the analysis. The specificity reflects the ratio of frames that are correctly classified as not showing human attention to an AOI (negative class) in relation to all frames with a negative class label. Sümer et al. (2018) compared the absolute number of predictions for each class, i.e., four individual students, to the ground truth count. In addition, they use a confusion matrix to show the performance of their face recognition system that is used to assign fixations to students' faces. Callemein et al.

(2019) used measures for inter-rater agreement like Cohen's κ to show the performance of their gaze-to-face and gaze-to-hand mapping. Venuprasad et al. (2020) reported precision, recall, and accuracy for frames and event metrics based on detection events: first looks, extra looks (i.e., revisits), false positive and false negative events are counted. Other works reported qualitative results only or did not evaluate their method. In this work, we report fine-grained frame- and event metrics per AOI from the field of activity recognition (Ward et al., 2011). They were shown to be effective for evaluating event detection methods in the field of mobile eye tracking (Ward et al., 2011; Steil et al., 2018a). The metrics are based on a segmentation of the ground truth and prediction signal at the frame level per AOI (see figure 8.2). A segment ends if the ground truth or the prediction changes, i.e., both signals are constant within a segment. Each segment can now be rated as one of true positive, true negative, false positive, or false negative. The event and frame metrics are derived from these segments. Prior to feature computation, we remove events with a duration smaller than the frame time and merge adjacent events.

8.1.3.1 Event Metrics

Ward et al. (2011) define a set of error classes for events that are meant to characterize the performance of a single-class event detection method. For multi-class problems, each class is handled separately. Error classes include the insertion (I') and deletion (D) errors, which are commonly used in event detection. An insertion error indicates that a detected event is not present in the ground truth (false positive), and a deletion error indicates a failure to detect a ground truth event (false negative). Additional error classes include fragmentation and merge errors: a ground truth event is fragmented (F) if multiple fragmenting events (F') are detected in the output. Similarly, multiple ground truth events of the same class can be merged (M) by a single merging event (M') in the output. Both errors can appear together, e.g., if a ground truth event is fragmented by three event detections of which the third is merging an additional ground truth event. In this case, the first ground truth event is marked as fragmented and merged (FM), and the third event detection is marked as fragmenting and merging (FM'). The apostrophe indicates whether an error class is assigned to a ground truth event or a predicted event in the output. If none of the error classes can be assigned, a detected event is counted as correct (C), i.e., as a true positive. According to Ward et al. (2011), we visualize the metrics by means of an event analysis diagram (EAD). It shows the number and ratio of error classes in relation to the number of reference events, i.e., to the number of ground truth events $|E| = D+F+FM+M+C$, the number of predicted events (or returns) $|R| = M' + FM' + F' + I' + C$, or both in case of correct predictions C. Also, we can compute event-based precision and recall as a ratio between $|R|$ or $|E|$ and the error class counts. We compute a conservative precision as $Pr = \frac{C}{|R|}$ and recall as $Re = \frac{C}{|E|}$. Counting F, FM, M and M', FM', F' as correct, similar to Toyama et al. (2012), we calculate a more progressive precision as $Pr^* = \frac{|R|-I'}{|R|}$ and recall as $Re^* = \frac{|E|-D}{|E|}$.

8.1.3.2 Frame Metrics

For extracting the frame metrics, Ward et al. (2011) project error classes to frames per segment. Similar to event-based error classes, a frame can be rated as insertion (I_f), deletion (D_f), merge

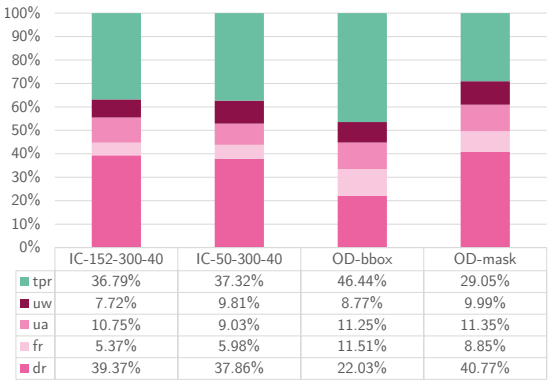
(M_f), or fragmentation (F_f). Merge errors are assigned to false positive frames from merging events, and fragmentation errors are assigned to false negative frames between fragmenting events. Further, if a neighboring segment is classified as true positive, frames of a false positive segment are marked as overfill (O_f) and frames of a false negative segment are marked as underfill (U_f). In other words, an overfill occurs if a detected event starts early or ends late, and an underfill occurs if a detected event starts late or ends early. A superscript indicates whether an underfill or overfill occurs at an event's start (α) or end (ω). Frames of true positive (TP) and true negative (TN) segments are classified likewise. Ward et al. (2011) define the frame metrics as ratios of the error class counts and the total positive frames P or negative frames N in the ground truth, with $P = D_f + F_f + U_f^\alpha + U_f^\omega + TP$ and $N = I_f + M_f + O_f^\alpha + O_f^\omega + TN$. The resulting ratios (lowercase equivalents to error classes) can be used to express the false positive rate as $fpr = ir + mr + o^\alpha + o^\omega$, and one minus the true positive rate as $(1 - tpr) = dr + fr + u^\alpha + u^\omega$. We use a set of two stacked bar charts to visualize the frame metrics (compared to pie charts in Ward et al. (2011)).

8.1.4 Evaluation

We evaluate the performance of the two methods described above regarding their ability to detect time intervals in which a participant fixates a certain AOI. Our evaluation procedure utilizes the VISUS dataset (Kurzhals et al., 2014a), including eye tracking data from 25 participants for 11 scenarios and manual AOI annotations, which we use for ground truth extraction. To measure the performance, we use a set of frame- and event-based metrics by Ward et al. (2011) from the field of activity recognition. We report the metrics per scenario and for each of the 34 AOIs to identify effective applications and limitations when using state-of-the-art pre-trained computer vision models.

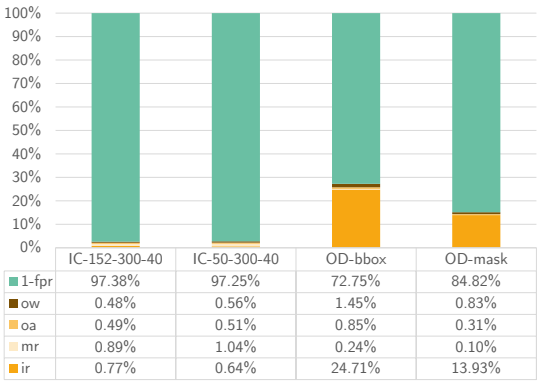
8.1.4.1 Experiment Conditions & Procedure

We compare two methods for visual attention detection: *IC* based on gaze-guided image classification and a threshold-based event detection, and *OD* based on object detection and fixation mapping. We generate predictions for the VISUS dataset using each method and analyze their results. We start with default parameters to identify AOIs that are not supported. We define cases with a zero recall as failing: this corresponds to a deletion rate of $dr = 100\%$ (frame metrics) or if all ground truth events are marked as deletions D. By design, we expect AOIs without a matching class label to fail (see dashes in table 8.1). For the remaining AOIs, we investigate the impact of different methods and parameters on the performance metrics. We compare two *IC* methods using the classification *models*, ResNet-50 and ResNet-152, and two *OD* methods using the *object mask* options bbox and mask. The other parameters are set to their defaults, which results in the following set of parameterized methods: IC-152-300-40, IC-50-300-40, OD-bbox, and OD-mask. For *IC*, we additionally test different values for T_{dur}, T_{noise}, and T_p using the ResNet-152 *model*, with $T_{dur} = T_{noise} \in \{100, 300, 500, 700\}$ ms and $T_p \in \{20\%, 40\%, 60\%\}$. Changing these parameters might have an effect on the performance of the *IC* method. Per method, we compute the frame and event metrics for each AOI and per participant. We sum the metrics over participants if we report the performance per AOI and over participants and



	IC-152-300-40	IC-50-300-40	OD-bbox	OD-mask
tpr	36.79%	37.32%	46.44%	29.05%
uw	7.72%	9.81%	8.77%	9.99%
ua	10.75%	9.03%	11.25%	11.35%
fr	5.37%	5.98%	11.51%	8.85%
dr	39.37%	37.86%	22.03%	40.77%

(a) false negative errors



	IC-152-300-40	IC-50-300-40	OD-bbox	OD-mask
1-fpr	97.38%	97.25%	72.75%	84.82%
ow	0.48%	0.56%	1.45%	0.83%
oa	0.49%	0.51%	0.85%	0.31%
mr	0.89%	1.04%	0.24%	0.10%
ir	0.77%	0.64%	24.71%	13.93%

(b) false positive errors

Figure 8.3: Frame metrics with respect to positive 8.3a and negative 8.3b ground truth frames across all AOIs.

AOIs if we report the overall performance of a method. Summing the metrics corresponds to concatenating the recordings of all participants per AOI because the metrics are based on absolute counts. Ratios are computed using the total number of positive and negative ground truth frames or events, which we also add up.

8.1.4.2 Results

Using default parameters, we observe a recall of zero for all AOIs without a matching class label for a method, but also for other AOIs: for the *IC* method, this includes "left face", "cup1", "cards", "stack covered", "case", "player white", "red bag", "brown bag", and "hooded". For the *OD* method, this includes "cup1", "cup2", "cards", "stack covered", "case", "red bag", and "persons" (for `10-bag search` only). We count one additional AOI for *OD* ("ball") and three AOIs for *IC* ("suspects", "ball", and "persons" in `10-bag search`) as failing because they yield a recall close to zero ($dr \geq 90\%$). The remaining six AOIs for *IC* and nine AOIs for *OD* are analyzed in detail (AOIs are listed in section 8.1.4.2). The AOIs for *IC* include $|E| = 2438$ ground truth events corresponding to $P = 71{,}911$ positive frames and $N = 111{,}467$ negative frames. The AOIs for *OD* include $|E| = 4328$ events with $P = 154{,}783$ positive and $N = 270{,}750$ negative frames.

Overall Performance

We compare the metrics of two *IC* and two *OD* methods that vary in terms of the *model* or *object mask* setting (see section 8.1.4.1). The frame metrics for the remaining AOIs are summarized in figure 8.3. It shows the ratios of false negative errors with respect to P in figure 8.3a and of false positive errors with respect to N in figure 8.3b. Concerning the false negative errors, deletion is the most prominent class across all methods: they account for 22.03% (OD-bbox) and 40.77% (OD-mask) of the errors for *OD*, and *dr* is 39.37% for ResNet-152 and 37.86% for ResNet-50 for the *IC* methods. On average, the *tpr* does not differ between *OD* (37.75%) and *IC* (37.05%).

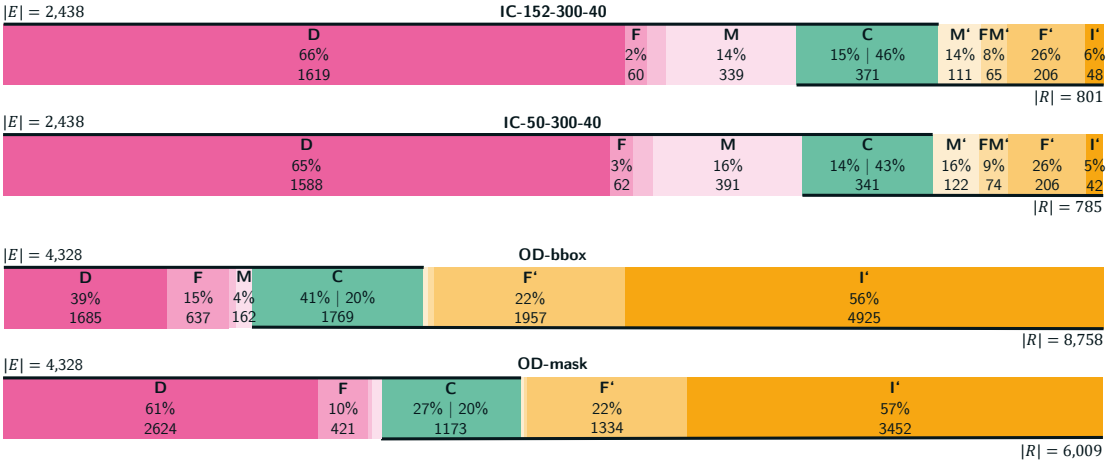


Figure 8.4: EAD diagrams visualizing the event-based error classes with respect to ground truth events $|E|$ and returned events $|R|$.

However, OD-bbox yields the best *tpr* with 46.44%, which is 9.39% better than the average of both *IC* methods and 17.39% better than OD-mask. The remaining error classes account for 30.86% for *OD* and 24.33% for *IC*: on average, *IC* faces 6.53% less false negatives through fragmenting events and underfills than *OD*. Concerning the false positive errors (see figure 8.3b), insertions are most prevalent for *OD* with $ir = 24.71\%$ for OD-bbox and $ir = 13.93\%$ for OD-mask. We observe fewer insertions for *IC*, averaging to 0.71%. Errors from merging event detections and overfills account for 1.98% (*IC*) and 1.89% (*OD*). Hence, the *fpr* adds up to 2.69% for *IC* and to 21.21% for *OD*, which means that the *OD* methods cause 18.53% more false positive errors at the frame level, on average.

Further, we report event metrics, which are normalized by the number of ground truth events $|E|$ or the number of retrieved events $|R|$ (see figure 8.4). Both *IC* methods show a similar distribution of error classes. For IC-152-300-40, we observe a high fraction of deletions, $\frac{D}{|E|} = 66.41\%$, and a low fraction of insertions, $\frac{I'}{|R|} = 5.99\%$, which is consistent to frame metrics. 371 predictions are correct which corresponds to $Re = 15.22\%$ (conservative recall) of the ground truth and $Pr = 46.32\%$ (conservative precision) of all retrieved events. The more progressive recall and precision are higher with $Re^* = 33.59\%$ and $Pr^* = 94.01\%$. The distribution of the remaining error classes shows, e.g., how many fragmenting events F' (206 $\rightarrow$ 25.72%) cause the fragmentations F (60 $\rightarrow$ 2.46%) in the ground truth.

The two *OD* methods have a similar distribution of error classes for retrieved events: the rate of fragmenting events is $\frac{F'}{|R|} \approx 22\%$, the rate of insertions is $\frac{I'}{|R|} \approx 56\%$, the counts for M' and FM' are very low, and the conservative precision Pr is similar with $\frac{C}{|R|} \approx 20\%$. However, OD-bbox predicts 2749 events more than OD-mask, resulting in a higher absolute number of correct events for OD-bbox ($C = 1769$) compared to OD-mask ($C = 1173$). With $|E|$ being constant for both *OD* methods, $Re = \frac{C}{|E|}$ is higher for OD-bbox (40.88%) than for OD-mask (27.1%). Consequently, the fraction of deletions for OD-bbox (38.94%) is lower than the fraction for OD-mask (60.63%), which is close to the level of the *IC* methods. Further, the *OD* methods

Method	AOI	|E|	D	F	FM	M	C	M'	FM'	F'	I'	|R|	Re*	Pr*
IC-152-300-40	01-car pursuit → red car	302	43.38%	4.30%	4.64%	34.77%	12.91%	23.35%	20.36%	10.78%	31.74%	13.77%	167	56.62%	86.23%
	02-turning car → red car	305	41.31%	4.26%	4.92%	28.52%	20.98%	36.78%	13.79%	13.22%	31.61%	4.60%	174	58.69%	95.40%
	03-dialog → shirt	42	71.43%	0.00%	0.00%	4.76%	23.81%	83.33%	8.33%	0.00%	0.00%	8.33%	12	28.57%	91.67%
	07-kite → kite	1316	75.53%	1.75%	1.44%	8.89%	12.39%	52.75%	12.62%	7.44%	23.95%	3.24%	309	24.47%	96.76%
	08-case exchange → persons	201	68.66%	2.99%	0.50%	7.96%	19.90%	63.49%	11.11%	1.59%	20.63%	3.17%	63	31.34%	96.83%
	11-person search → red shirt & hat	272	73.53%	1.84%	0.00%	4.41%	20.22%	72.37%	7.89%	0.00%	14.47%	5.26%	76	26.47%	94.74%
OD-bbox	01-car pursuit → red car	304	36.84%	18.42%	4.61%	7.89%	32.24%	23.90%	1.71%	2.44%	53.90%	18.05%	410	63.16%	81.95%
	02-turning car → red car	311	15.43%	36.33%	6.11%	7.40%	34.73%	16.67%	0.62%	2.01%	71.14%	9.57%	648	84.57%	90.43%
	03-dialog → left face	193	7.77%	16.06%	1.55%	3.63%	70.98%	35.13%	0.51%	0.77%	17.95%	45.64%	390	92.23%	54.36%
	06-UNO → left hand	1157	26.71%	15.56%	1.64%	2.16%	53.93%	31.23%	0.35%	0.55%	21.02%	46.85%	1998	73.29%	53.15%
	07-kite → kite	1321	68.96%	8.33%	0.45%	2.73%	19.53%	39.21%	2.28%	0.91%	55.62%	1.98%	658	31.04%	98.02%
	07-kite → person	290	52.07%	2.07%	0.00%	0.69%	45.17%	89.12%	0.68%	0.00%	8.16%	2.04%	147	47.93%	97.96%
	08-case exchange → persons	199	16.08%	20.60%	1.01%	4.02%	58.29%	29.44%	0.25%	1.52%	25.63%	43.15%	394	83.92%	56.85%
	09-ball game → player white	338	26.04%	8.28%	1.18%	8.88%	55.62%	24.29%	1.68%	0.26%	8.01%	65.76%	774	73.96%	34.24%
	11-person search → hooded	214	8.88%	33.64%	3.27%	3.27%	50.93%	3.26%	0.03%	0.15%	7.31%	89.25%	3339	91.12%	10.75%

Figure 8.5: *EAD* table for AOIs with a non-zero recall ($dr < 90\%$; $\frac{D}{|E|} \ll 100\%$) for IC-152-300-40 and OD-bbox.

report a higher level of fragmented events F than merged events M. We observe the opposite for *IC*. The progressive precision and recall values are $Pr^* = 43.77\%$ and $Re^* = 61.06\%$ for OD-bbox, and $Pr^* = 42.55\%$ and $Re^* = 39.37\%$ for OD-mask.

AOI Performance Breakdown

For IC-152-300-40 and OD-bbox, we report the event metrics per AOI in a table (see figure 8.5). For *IC*, we observe a difference between the two "red car" AOIs and the remaining AOIs. On average, we see a lower level of deletions for "red car" with $\frac{D}{|E|} = 42.34\%$ and an increased level of merged events with $\frac{M}{|E|} = 31.65\%$, compared to the other AOIs which average to 72.29% and 6.51%, respectively. Consequently, we observe the best progressive recall for "red car" with $Re^* = 57.66\%$ on average, compared to $Re^* = 27.71\%$ for the other AOIs. The conservative precision $\frac{C}{|R|} = 30.07\%$ is lower than the average of 67.99% for the other AOIs. For "red car", M is higher, and C is lower for `01-car pursuit` than for `02-turning car`. Hence, the conservative recall $\frac{C}{|E|} = 12.91\%$ for `01-car pursuit` is relatively lower by 38.47%, while the Re^* is lower by 3.53% only. Further, we observe the highest relative number of insertions with $\frac{I'}{|R|} = 13.77\%$ (others average to 4.92%). The *OD* methods result in more diverse error class distributions. We observe a low level of D in relation to ground truth events $|E|$ for "red car" (`02-turning car`), "left face", "persons", and "hooded", averaging to 12.04%. The AOIs "kite" and "person" result in the highest level for deletions D with 68.96% and 52.07%, respectively. For these AOIs, the low and high levels for D coincide with the highest and lowest Re^* values. Overall, we see a high level of fragmented events F with highest values for "hooded" with 33.64% and "red car" in `02-turning car` with 36.33%, and lowest values for "person" in `07-kite` with 2.07%. The AOI "left face" results in the best conservative recall, $Re = 70.98\%$, due to the low level of deletions D, with 7.77%. For insertions I', we observe the lowest levels for the AOIs in `kite` with an average of 2.01%, followed by "red car" with 9.57% in `02-turning car` and 18.05% in `01-car pursuit`. All other AOIs average to 58.13% with a peak for "hooded" with 89.25%. These four AOIs with the lowest insertion rate $\frac{I'}{|R|}$ have the best progressive precision with, on average, $Pr^* = 92.09\%$. The remaining five AOIs average to $Pr^* = 41.87\%$ with the minimum for "hooded" with $Pr^* =$

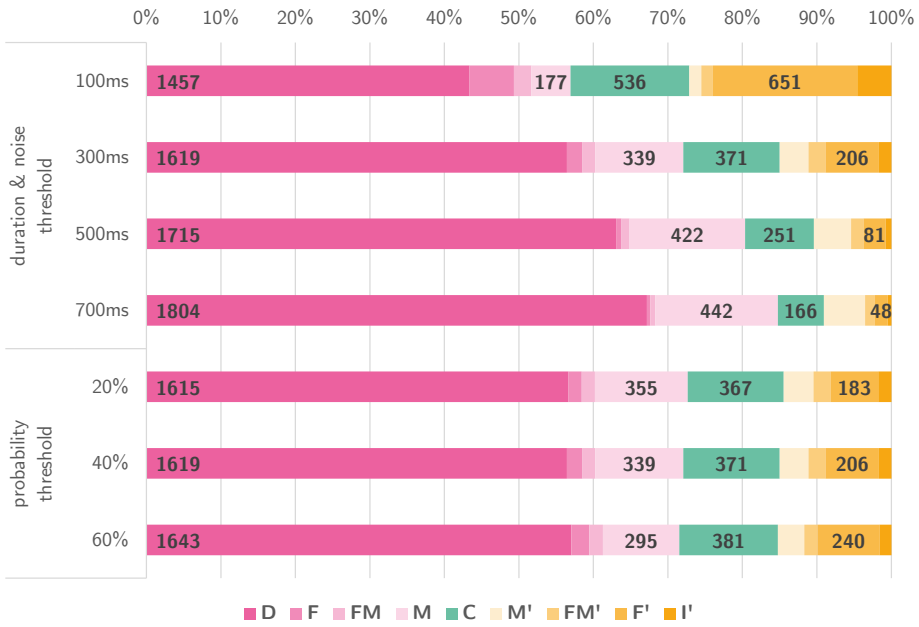


Figure 8.6: Simplified EAD diagrams for *IC* methods with varying parameters aggregated over AOIs with non-zero recall. We vary the probability threshold T_p or the duration and noise threshold $T_{dur} = T_{noise}$. We use default values for other parameters. The results for 300 ms and 40% refer to the default setting IC-152-300-40 as shown in figure 8.4.

10.75%. The highest levels of fragmenting events are observed for "red car" (53.9% and 71.14%) and "kite" (55.62%). To compare the results of AOIs that remain for *IC* and *OD* methods, we calculate an event-based f_1 score as $f_1 = 2 \cdot \frac{Pr^* \cdot Re^*}{Pr^* + Re^*}$. For the AOIs "red car" (x2), "kite", and "persons" (`08-case exchange`), we receive f_1 scores of $68.36\%, 72.67\%, 39.06\%, 47.36\%$ for IC-152-300-40 and $71.34\%, 87.4\%, 47.15\%, 67.78\%$ for OD-bbox. For this selection of AOIs, OD-bbox yields the respectively better performance.

Impact of IC Parameters on Performance

The *IC* method offers multiple parameters for tuning the outcome, besides the classification *model*. We investigate the impact of T_{dur} & T_{noise} and T_p on the frame and event metrics. With varying T_p, we observe no changes in the distribution of event error classes (see figure 8.6). In addition, Pr^* ranges between 32.69% and 33.76%, and Re^* ranges between 93.58% and 94.39% for the different settings of T_p. When increasing the duration and noise thresholds, we observe a monotonic increase in the number of deletions D: the ratio $\frac{D}{|E|}$ ranges from 59.49% for 100 ms to 70.87% for 700 ms. At the same time, $\frac{M}{|E|}$ increases from 7.23% to 18.1% and $Re = \frac{C}{|E|}$ decreases from 21.89% to 6.8%. Concerning the error classes of retrieved events, we observe a monotonic decrease in the number of insertions I' ranging from 10.64% for 100 ms to 3.18% for 700 ms. Similarly, $\frac{F'}{|R|}$ decreases from 44.99% to 11.74%, as well as the absolute number of retrieved events $|R|$ which ranges from 1447 to 409. The level of merging events M' increases

from 3.73% to 36.19% which corresponds to the increase of merged events M. In addition, we see a trend in the progressive precision and recall values: with increasing duration and noise threshold, Re^* decreases from 40.51% to 26.13% and Pr^* increases from 89.36% to 96.82%. The highest f_1 score of 55.74% is reached for 100 ms.

8.1.5 Discussion

Our results show that using our methods with default parameters and the AOI configuration from table 8.1 does not support all AOIs. In particular, our observations confirm that they fail to detect visual attention for AOIs without a mapping. This affects 15 AOIs (44.12%) for *IC* and 17 AOIs (50%) for *OD*. Our results reveal 13 additional AOIs for *IC* and 8 AOIs for *OD* with weak matches that result in zero or close to zero recalls ($dr \geq 90\%$). Effectively, we count 28 AOIs (82.35%) for *IC* and 25 AOIs (73.53%) for *OD* as failing. We attribute these fails to challenge I, because the concepts of the AOIs have no adequate match to any class label of the underlying computer vision model. And, if there is a matching class label, the instances might differ from what the model has learned, i.e., from the training samples.

8.1.5.1 Overall Performance

The frame and event metrics for the remaining AOIs show that deletions are the most frequent false negative error across all methods. The frame-based deletion rates dr are lower than the respective deletion events level D. For instance, in IC-152-300-40, $\frac{D}{|E|} = 66.41\%$ of the ground truth events correspond to $dr = 36.79\%$ of the positive ground truth frames. This may indicate that our methods delete more short events than long ones. The high level of deleted events might be caused by false negatives from the computer vision model (related to challenge I). Another problem could be that our models failed to map the gaze signal, although the prediction was correct. To investigate this issue further, we generated videos showing the manual annotations, the gaze and fixation events, and our prediction output. We noticed that the eye tracking signal frequently suffers from low accuracy and, hence, the gaze point does not hit an AOI object even though it is obvious that the participant followed that object, e.g., a "kite". The manual annotations (bounding boxes) in the VISUS dataset are bloated to include such erroneous gaze signals, better capturing human behavior than exact annotations. However, assuming that the gaze signal was accurate, this data annotation style results in many false positive ground truth events (see figure 8.7a). Our methods are not robust against such cases because they rely on local image classifications (*IC*) or fixation to object mask mapping (*OD*). Consequently, our methods report no attention events, which might be a major reason for the high deletions. We investigate this issue in detail in section 8.1.5.4. This could also explain the difference between OD-bbox, using bounding boxes, and OD-mask, using exact object masks, i.e., OD-bbox better resamples the manual annotations and, to some degree, compensates the inaccurate gaze signals. OD-bbox shows the best progressive recall with $Re^* = 61.06\%$, while the other methods average around 35.91%. Also, our results show that *OD* yields more insertion errors than *IC* in terms of frame and event metrics: the insertion rate is $\frac{I'}{|R|} \approx 56\%$ for *OD* methods and 6% for *IC* methods. Consequently, with an average of $Pr^* = 94.33\%$, *IC* results in better progressive precision than *OD* with $Pr^* = 43.16\%$. This suggests that the *IC* method may be the better choice for use

(a) `07-kite` (b) `03-dialog`

Figure 8.7: Example frames from two scenarios of the VISUS dataset (Kurzhals et al., 2014a) showing the recent fixation (white circle), ground truth annotations (green), object masks and bounding boxes for *OD* (blue), and the cropping area for *IC* (white rectangle).

cases with a good object-to-class label match and if false negative errors are not severe. In addition, the relation of FM, F, F' to fr and FM', M, M' to mr can reveal more about the error characteristics. For instance, if we see many event errors and a low ratio of corresponding frame errors, the fragmenting or merging predictions approximate the ground truth well (see figure 8.2). For instance, for IC-152-300-40, merge errors M make up 14% of the event errors with respect to the ground truth but result in a low frame error rate of $mr = 0.89\%$.

8.1.5.2 Performance per AOI

The results for default parameters at the AOI level show that OD-bbox performs best for the four overlapping AOIs (see figure 8.5). However, all other AOIs for OD-bbox suffer from high insertion levels of more than 40%. A reason might be that these AOIs match to "person" (see table 8.1) and, at least, a second AOI shares this concept (related to challenge II). For instance, we map the MS COCO class label "person" to the AOI "left face" in `03-dialog`, but "person" would also fit "right face" and "shirt". The generated debug videos show that both *OD* methods detect attention events for "right face" and "shirt" based on the "person" class label. However, these are wrongly mapped to "left face" which results in many false positives (see figure 8.7b). This problem of the remaining AOIs is likely to cause the high level of insertion errors and the low progressive precision for *OD* overall.

8.1.5.3 Impact of IC Parameters

Our investigation with different parameters for *IC* reveals that T_p will likely have no impact on event metrics. Our assumption is that the subsequent aggregation of image classification results is a harder criterion than a high T_p. For example, an incorrect classification with low probability might be dropped anyway due to reaching T_{noise} because it alternates with other wrong classifications. The parameters T_{dur} and T_{noise} have a clear impact on the performance: increasing the threshold results in decreasing values of Re and Re^*. $T_{dur} = T_{noise} = 100$ ms yields the best overall performance by means of the f_1 score, followed by the default setting,

which results in a better Pr^*, but worse Re^*.

8.1.5.4 Impact of Re-Annotating the Ground Truth Data on Deletions

In many cases, the gaze recordings from the VISUS dataset suffer from low spatial accuracy, which resulted in coarse manual annotations. For instance, in figure 8.7a, the manual annotations for "person" and "kite" (green bounding boxes) are much larger than the actual object to catch the point of gaze that, when looking at the video, obviously follows the kite. In contrast, the bounding boxes and exact object masks generated by Mask R-CNN (blue rectangles and polygons) closely frame the "person" and the "kite". We hypothesize that this kind of annotation is responsible for many deletion errors (false negative events) because the ground truth reports a false attention event that cannot be captured by our detection methods. To verify our assumption, we re-annotate AOIs without a close to zero recall (see figure 8.5) and repeat our analysis using the new ground truth annotations, but the same event predictions from IC-152-300-40 and OD-bbox that we have gathered in our main experiment. The videos are annotated by a single annotator and reviewed by an eye tracking expert using the Computer Vision Annotation Tool CVAT[8]. We use the polygon-based annotation feature: a polygon that closely frames an object at keyframes with interpolation for intermediate frames is created. The results show that the ratio of deletion events $\frac{D}{|E|}$ decreases by 16.3% to 50.11% for the *IC* method and by 10.3% to 28.64% for the *OD* method. Consequently, the progressive recall values Re^* increase by the same amounts to 49.89% for *IC* and to 71.36% for *OD*. Thus, we can confirm our hypothesis that coarse AOI annotations increase the level of deletions. This emphasizes the importance of accurate gaze estimation methods to avoid such errors. Further, it raises the need for error-aware gaze-to-object mapping methods to compensate the impact of the gaze estimation error, similar to those presented chapter 4. For instance, we could detect an AOI hit by checking whether the distance of a fixation point to the boundary of an AOI is smaller than a defined threshold.

8.1.5.5 Limitations & Future Work

Our evaluation revealed several limitations related to the challenges identified in section 8.1.2.1 and to accuracy issues with the gaze signal in the VISUS dataset. The main limitation of our methods is related to challenge I: many AOIs are not supported because the concepts are not included with the pre-trained computer vision models. A promising solution to address it is to collect new samples for unsupported AOIs and AOIs with weak matches for fine-tuning the computer vision models (Käding et al., 2017; Sonntag et al., 2017). We want to investigate the effectiveness of interactive machine learning methods for this purpose (Simard et al., 2017; Dudley and Kristensson, 2018) compared to randomly annotating a small portion of the data as suggested in Wolf et al. (2018). Further, our methods offer no solution for challenge II, i.e., when AOIs share the same concept. This could be solved using similarity models with interactive training. For instance, we could iteratively train a model to differentiate between "left face" and "right face", reducing the number of insertion errors for `03-dialog`. Using multiple object tracking algorithms (Li et al., 2019) with humans-in-the-loop is a promising approach to support

[8] `https://github.com/openvinotoolkit/cvat` (accessed on 12 Dec 2024)

challenges III and IV, i.e., when AOIs share a concept and have a similar appearance or AOIs change their appearance over time.

We aim to address these challenges in the next section on interactive fixation-to-AOI mapping by leveraging interactive model training and humans-in-the-loop (see section 8.2). In addition, we showcase a real-time application of the *IC* method in an augmented reality setting with objects that are well represented in the training data of the image classification model (see section 8.2.4). Further issues might arise from the fixation detection algorithm used in Kurzhals et al. (2014a). The authors mentioned that, e.g., smooth pursuit movements are not supported well, which make up a large portion of the data. This is known from recent work on fixation detection algorithms for head-mounted eye tracking headset (Steil et al., 2018a).

8.2 Interactive Fixation-to-AOI Mapping

To address the challenges in annotating head-mounted eye tracking data, we implement *eyeNotate*, a user interface that enables semi-automatic annotation. Our tool allows mobile eye tracking practitioners to manually annotate their recordings fixation-wise (baseline) and semi-automatically using fixation-to-AOI mapping suggestions based on a few-shot image classification model (IML-support). We contribute by (i) implementing the *eyeNotate* tool for semi-automatic annotation of head-mounted eye tracking data based on few-shot image classification, and (ii) evaluating our *eyeNotate* in a case study with n=3 trained annotators to compare the baseline version and the IML-supported approach, measuring the perceived usability, annotation validity and reliability, and efficiency during a data annotation task.

8.2.1 Method

We implement *eyeNotate*, a web-based tool for fixation-to-AOI mapping, an essential data processing step in research based on mobile eye trackers. Our tool allows practitioners to annotate recordings manually fixation-wise, reflecting the current state-of-the-art (baseline). We designed the user interface to enable efficient navigation through videos based on fixation events aligned to common video-editing interfaces. Further, we integrate an IML component that can provide AOI label suggestions for fixations and learn from user feedback, i.e., when they accept or reject/correct suggestions, based on a few-shot image classification model (IML-support). User annotations and model-based suggestions are stored in a database. Figure 8.8 shows the basic user interface and an overview of the IML-support features.

8.2.1.1 Baseline Annotation Tool

The *baseline* tool offers a video-editing-like interface for fixation-wise data annotation (see figure 8.8 a). It includes three main elements: A top bar displays information on the selected recording and the annotation progress, a list on the left that shows all fixations and their annotation state, and a video view on the right with a fixation overlay and buttons for manual annotation. Selecting a fixation from the list causes the video view to show the respective image frame with a circular overlay at the fixation position, indicating the currently assigned AOI. An

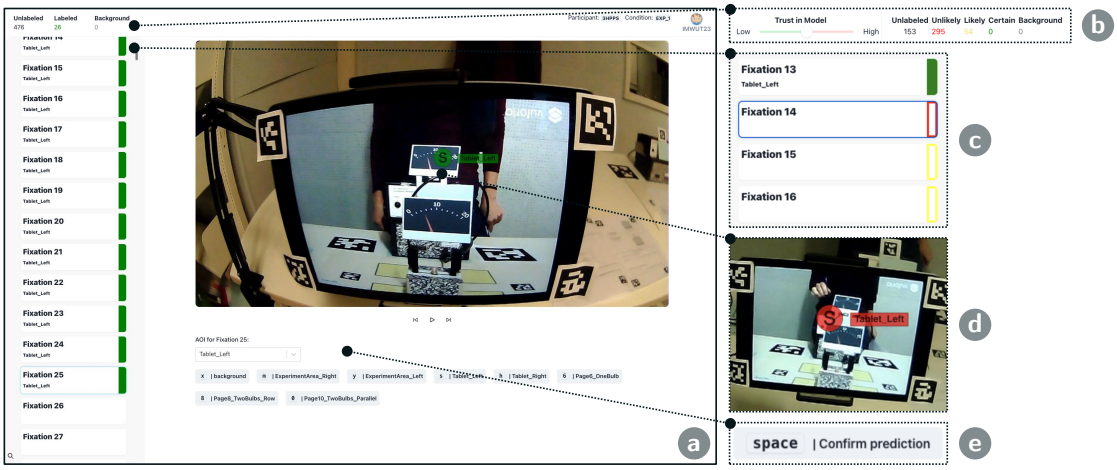


Figure 8.8: (a) Screenshot of the user interface of our *baseline* annotation tool, and (b-e) an overview of the *IML-support* features. It extends the baseline by (b) a status bar indicating the number of AOI suggestions grouped by model certainty and a trust-level slider for adjusting certainty intervals, (c) indicators for AOI suggestions in the fixation list, (d) adjusted fixation overlays for the video, and (e) an option to confirm AOI suggestions.

AOI can be assigned to the fixation by clicking one of the AOI buttons or pressing the corresponding shortcut on the keyboard. This is visually confirmed by a green badge that appears next to the fixation's list entry, and the overlay in the video view that turns green and shows the newly assigned AOI label. Navigation through fixations is possible via arrow keys and on-screen video controls. When multiple fixations hit the same AOI, they can be annotated simultaneously by selecting all fixations from the list using shift and arrow keys, consistent with multi-item selection in common list views.

8.2.1.2 Interactive Machine Learning Support

The *IML-support* version of our tool integrates an IML component based on a few-shot image classification model, which is initialized with a small set of images per AOI. This model generates AOI label suggestions for each fixation by cropping an image patch from the corresponding video frame around the fixation point. Manual annotations and confirmative or corrective feedback are used to re-train the image classification model, aiming to improve its performance over time. The model training and inference run in parallel to enable flexible and quick adaptations of the model to the target domain. Figure 8.9 shows a high-level overview of the components of our system and how the inter-relate.

User Interface

The user interface of the *IML-support* version is extended to display and interact with model-based label suggestions (see figure 8.8 b-e). A non-filled badge at a fixation's list item indicates that a suggestion is available (see figure 8.8c). The outline color of the badge encodes the model's

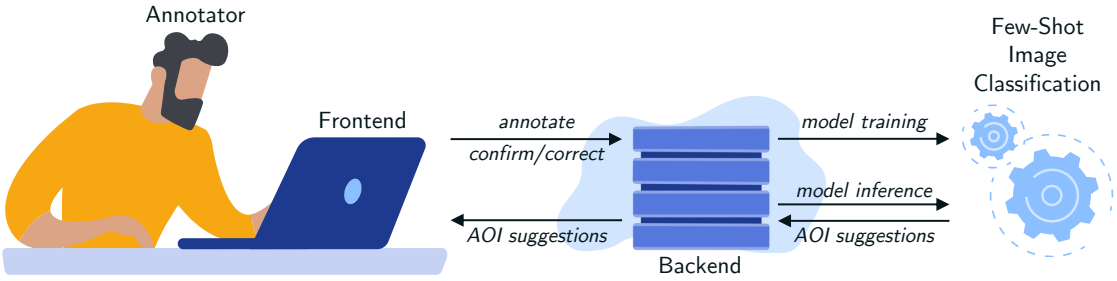


Figure 8.9: Overview of the architecture of our interactive annotation system including a web-based user interface (frontend), a backend for managing data storage, and an IML service that enables label suggestions and model retraining for the *IML-support* version of our tool.

confidence which is either high (green), medium (yellow), or low (red). The color is also reflected in the fixation overlay in the video view (figure 8.8d). Users can set their perceived trust in the model using a slider in the top bar (figure 8.8b). Moving the slider towards high trust decreases the confidence thresholds: more suggestions appear in green. Next to the slider, an overview displays the distribution of suggestions across confidence levels. A suggestion can be confirmed or corrected by users. They press the space key to confirm a suggestion for one or multiple selected fixations (figure 8.8e). To correct it, they assign another class.

Image Classification Model

The *IML-support* version adopts a few-shot learning strategy based on the Feature Map Reconstruction Network (FRNet) (Wertheimer et al., 2021) to generate AOI label suggestions. An overview of the training and inference for this model is illustrated in figure 8.10. The FRNet is a convolutional neural network (CNN) architecture that performs classification via a class-agnostic distance function: The image classification task is framed as a reconstruction problem in latent space, i.e., predicting class membership relies on measuring the distance between a query point and reference points in latent space representing our target classes (i.e., AOIs). For any query image x, the convolutional block of the network outputs a feature map $Q \in \mathbb{R}^{r \times d}$, where r is spatial resolution ($h \times w$) and d is the number of channels. The network is trained in an N-shot-K-way manner to learn support feature maps $S_k \in \mathbb{R}^{Nr \times d}$ for each AOI class $k \in K$ from a pool of N training images per class. During inference, the model aims to reconstruct the best-fit query feature map Q_k for each class category as a weighted sum of rows of S_k such that $WS_k \approx Q_k$, where W is the model weights optimized during model training. By examining the negative reconstruction error, which represents the disparity between the original feature map Q and each AOI-wise reconstructed feature map Q_k, FRNet assigns a class score. Smaller reconstruction errors indicate a higher likelihood that the query image belongs to the same class as the support features. We train our classification model using N=10 images and for K=7 AOIs (initial labeled data pool). Following Wertheimer et al. (2021), we combine the classification loss with an auxiliary loss L_{aux} that optimizes support features from different classes to span the

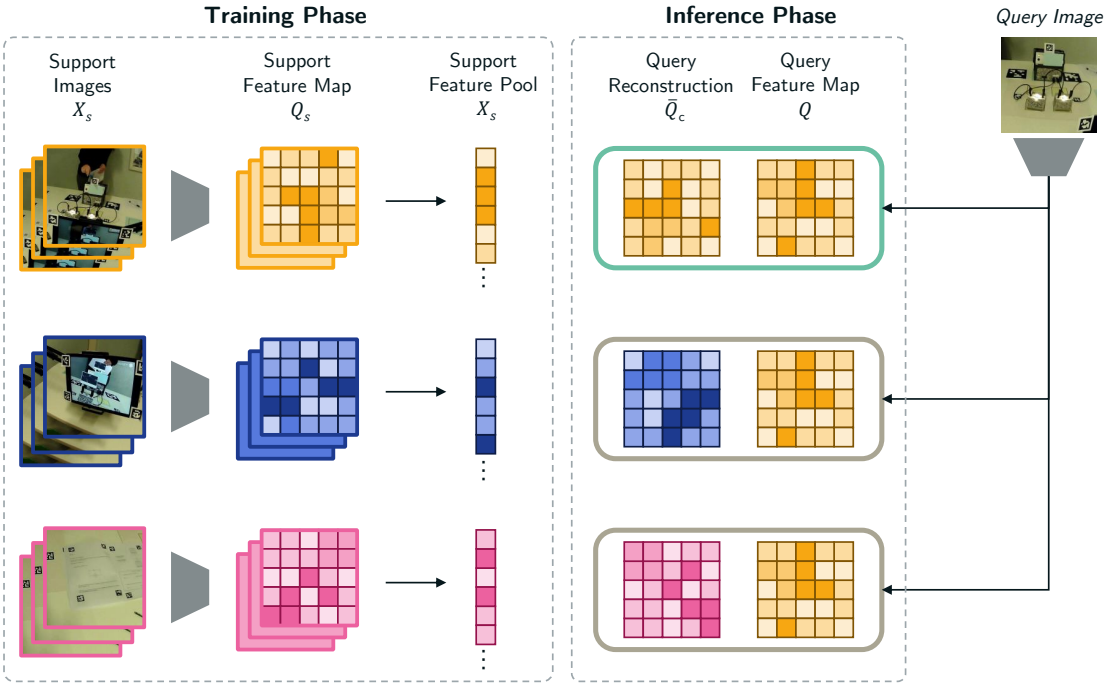


Figure 8.10: Overview of the FRNet classification workflow for a few-shot classification problem.

latent space to train FRNet:

$$L_{aux} = \sum_{i \in K} \sum_{j \in K, j \neq i} \|S_i S_j^T\|^2 \tag{8.1}$$

The annotation tool uses this pre-trained FRNet model to infer AOI labels for each fixation in the selected dataset. Label suggestions are displayed if the threshold exceeds a minimum confidence value (0.4) that the user can adjust through the trust level slider. Manual annotations and confirmed or corrected AOI labels are added to the labeled data pool. For every 10 new samples, a model re-training is started in the background. The model weights used for inference are updated upon completion. The models are trained for 30 epochs at each iteration with weights initialized from the previous steps. On an NVIDIA RTX 3080 GPU (24GB), the model training takes 2-4 s per epoch.

8.2.2 Evaluation

We evaluate our approach in two ways: we conduct a small case study with n=3 trained annotators to quantitatively and qualitatively compare the *baseline* version of our tool with the *IML-support* version. Annotators have been asked to annotate a small portion (ca. 2%) of an existing dataset with ground-truth annotations. In a post-hoc experiment, we assess the performance of three machine learning models in automatically annotating the remaining part of the dataset. In the following, we describe the usecase and the corresponding dataset, and we report

AOI	Visibility	
	exp_1	exp_2
Tablet_Left → T_L	✓	✓
Tablet_Right → T_R		✓
Experiment_Area_Left → E_L	✓	✓
Experiment_Area_Right → E_R		✓
Page6_OneBulb → P_6	✓	✓
Page8_TwoBulbs_Row → P_8		✓
Page10_TwoBulbs_Parallel → P_10		

Table 8.2: List of AOIs indicating their (intended) visibility per experiment phase.

on the case study and the machine learning experiment.

8.2.2.1 Usecase and Dataset from Educational Research

The evaluation focuses on educational research as an important eye tracking usecase. Most digital and analogous learning environments are based on visual information. Hence, gaze behavior is an important indicator of learning processes. Jarodzka et al. (2017) specify three main research aims for using eye tracking in educational sciences: The first aim is the improvement of instructional designs by investigating the waste of cognitive resources on ineffective instructional material (see, e.g., Malone et al. (2020)). Second, eye tracking can be used to investigate visual expertise leading to superior performance (see, e.g., Reingold and Sheridan (2011)). Third, eye tracking can be used to model learners' eye movements to promote visual expertise (see, e.g., Jarodzka et al. (2013)). Some further educational studies also used eye tracking to investigate learners' gaze behavior in testing situations before and after learning phases (see, e.g., Thees et al. (2022)). Recent mobile eye tracking devices are convenient to wear and enable learners to move freely and naturally in dynamic and interactive real-world learning environments, e.g., classrooms or science laboratories (Salminen-Saari et al., 2021; Fleischer et al., 2023). This is especially beneficial for eye tracking recordings with children, as they can easily be distracted by intrusive measurements and have difficulties sitting still for long periods of time.

The case study (n=3) and machine learning experiment described below use recordings from an existing mobile eye tracking dataset (n=48). It was recorded and annotated at Saarland University to investigate the impact of AR-support in a lab work-based learning scenario about electrical circuits on learning outcomes and learning processes of elementary school children[9]. Tablet-based AR was used to visualize measured values of current in different electric circuits in real-time during several experiment and observation tasks. The tablet-based AR condition was compared to a separate tablet display presentation format displaying the same values without using AR. The data to be annotated in the current case study originate from a single individual (child) who was assigned to the separate display condition. All children wore a Pupil Core head-mounted eye tracker (version for children with a smaller frame) (Kassner et al., 2014). The lab work started after a short introduction and the calibration of the mobile eye tracker

[9]Pre-registered at Open Science Framework: `https://osf.io/gwhu5` (accessed on 12 Dec 2024)

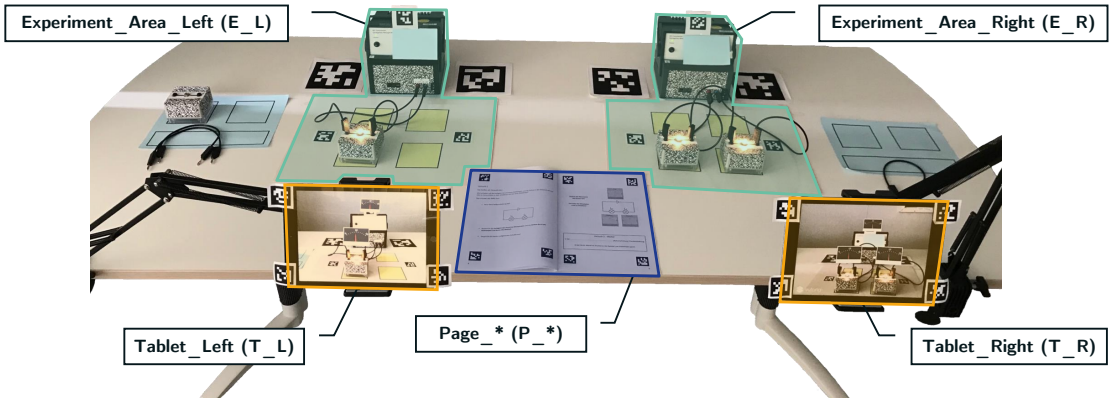


Figure 8.11: Overview of the experiment setup illustrating considered AOIs.

through physical markers. The Pupil Capture tool was used to record eye tracking data and a video from the world camera. The experiments aimed at the children learning scientific laws on current in series and parallel circuits. In the first experiment, children built a simple electrical circuit with one bulb. While the current at the power supply was manipulated, the children answered questions on the bulb's brightness and current measurements. After building up a series circuit with two bulbs for the second experiment, children again observed the current and brightness of bulbs while the current at the power supply changed and answered some questions. Subsequently, the children were asked to compare the brightness and current of the simple circuit they built up for the first experiment and the series circuit. The children also carried out a third experiment on parallel circuits, which is not part of the present study. For the current case study, the comparison process within the simple circuit (experiment phase 1 → exp_1) and the comparison process between the simple and the series circuit (experiment phase 2 → exp_2) are examined. An overview of considered AOIs can be found in table 8.2. Figure 8.11 shows an overview of the scene with overlays for each AOI. Experiment phase 1 includes five AOIs of the simple circuit setup with one bulb placed on the left side of an experimentation table: left tablet with measurement values (Tablet_Left → T_L), left voltage source and electric components (Experiment_Area_Left → E_L), and a double page in a workbook (Page6_OneBulb → P_6). Experiment phase 2 includes additional AOIs of a series circuit placed on the right side of the same table: right tablet with measurement values (Tablet_Right → T_R), right voltage source and electric components (Experiment_Area_Right → E_R), and another double page in a workbook (Page8_TwoBulbs_Row → P_8). The voltage source and electric components AOIs per side were merged into a single AOI for analysis. A third double page for phase three was sometimes visible as children scrolled through the workbook (Page10_TwoBulbs_Parallel → P_10). This results in a total of seven AOIs: three for experiment phase 1, three for experiment phase 2, and one additional for the workbook pages of phase 3. However, the AOIs could have also been visible when not intended because the scene was set up completely, and children might have looked to non-relevant AOIs. Nevertheless, fixations to these AOIs have been annotated. It is important to note that the tablets, experiment areas, and workbook AOIs have similar appearances, which relates to challenge III outlined in the previous section 8.1. Following the

completion of the data collection, the Pupil Player tool was used for detecting fixation events and annotating the eye tracking data fixation-wise: it offers an option to jump between successive fixations and supports hotkey-based annotation. All recordings have been annotated by four student assistants employed by Saarland University. They received intensive training before the annotation took place. The manual annotation of the full dataset took many days, which led to fatigue, frustration, and, eventually, inadvertent errors in the annotations that were difficult to fix. We recruited three of these student assistants for the present expert study, the fourth did not reply to our invitation.

8.2.2.2 Case Study

We measured the perceived usability, annotation validity and reliability, and the efficiency of the annotation process during an annotation task (within-subjects design). Further, we conducted a semi-structured interview to understand how the IML-support version was used and how that might impact the efficiency and validity of the annotation process. For this case study, we focused on the usecase of educational research and the existing dataset described above.

Procedure

We conducted the user study online via video calls, recorded for post-hoc transcription. First, we introduced the study procedure and obtained a signed informed consent via email. Then, we asked annotators to complete an annotation task with both *eyeNotate* versions. For each, we showed a short instruction video explaining the features. We allowed participants to familiarize themselves with the tool in a 5-minute training phase and ask clarification questions. Subsequently, participants performed an annotation task and completed the system usability scale (SUS) questionnaire (Brooke, 1996). Two participants started with the *baseline* version, one with the *IML-support* version of the tool. After both annotation tasks were completed, we conducted a semi-structured interview to retrieve further qualitative feedback for our tools, particularly for the distinct features of the *IML-support* version. The study took around one hour; each participant received a 10 EUR compensation payment.

Annotation Task

We asked participants to annotate 870 fixations from the dataset described above with ground-truth annotations. The dataset stems from a mobile eye tracking study in the educational sciences with the goal of investigating the impact of AR-support in a lab work-based learning scenario about electrical circuits on learning outcomes and learning processes of elementary school children (n=48). To reduce the workload in our study, we constrained the annotation task to data from a single child and two experiment phases (exp_1: 646 fixations; exp_2: 224 fixations). Fixations could be mapped to one of seven AOIs or a background class (see figure 8.8). The task ended when the participant annotated all fixations. For the *IML-support* version, the participants could stop early if all fixations had highly confident ("green") label suggestions, while the confidence level depends on the trust level slider.

Methods & Measures

We measure the perceived usability, annotation validity and reliability, and efficiency during the annotation task to assess the two annotation tool versions. We expect the IML-support version to be more efficient than the baseline, with the perceived usability and annotation validity and reliability remaining stable.

Validity & Reliability. We measure the validity of the participants' annotations for each tool version. We report their accuracy in mapping fixations to AOIs compared to the ground-truth annotations from the dataset used in this study. Further, we assess the reliability as the level of agreement among all participants for each version of our tool by calculating Fleiss' κ (Fleiss, 1971). We consider both measures to be control variables: we expect to observe a high accuracy for both versions of the tool ($\geq$ 95%) and an almost perfect inter-rater agreement ($\kappa > .8$) (Landis and Koch, 1977).

Efficiency. We measure the time required for completing the annotation tasks in seconds (task completion time) for each tool version. We expect the IML-support version to be more efficient than the baseline, according to findings in prior research, i.e., that the availability of label suggestions leads to easier and faster decision-making (Desmond et al., 2021).

Usability. We assess the usability of both versions of our annotation tool using the System Usability Scale (SUS) questionnaire (Brooke, 1996). Scores can range between 0 and 100, with high scores indicating better usability. We interpret the SUS scores according to the adjective rating by Bangor et al. (2009). We consider this a control variable, i.e., we do not expect a difference in perceived usability between the two versions of our tool, but we expect a high SUS score for both versions. Further, we conduct a semi-structured interview (SSI) to gain further qualitative insights about our annotation tool and specific IML features. The transcribed interview was analyzed using a reflective thematic analysis (Braun and Clarke, 2012).

Results

In the following, we present the results for each tool version, i.e., the baseline and *IML-support* versions. In some cases, we report the individual values per participant because we only considered three trained annotators for our case study: A1, B1, and B2. Participants started with the IML-support (A) or the baseline version (B).

Validity & Reliability. We assess the validity of annotations in terms of their accuracy compared to the ground truth. We report the mean over all three participants for each version of the annotation tool per phase and combined (see table 8.3). For phase *exp_1*, we observe an accuracy of 97.32% for the *baseline* version and 97.78% for the *IML-support* version. We observed slightly lower values for phase *exp_2*: the accuracy is 89.58% for the *baseline* version and 88.24% for the *IML-support* version. The weighted average over both phases results in an accuracy of 94.76% for the *baseline* version and 94.55% for the *IML-support* version. This weighted average considers the unbalanced number of fixations in each phase. We calculate Fleiss κ as a measure

	Accuracy		Fleiss' κ	
	baseline	*IML-support*	*baseline*	*IML-support*
exp_1	97.32%	97.78%	0.954	0.963
exp_2	89.58%	88.24%	0.941	0.920
mean	94.76%	94.55%	0.951	0.952

Table 8.3: Annotations' validity (Accuracy) and reliability (Fleiss' κ) per experiment phase and as a weighted mean.

for the inter-rater agreement. It is calculated per condition and phase based on the ratings from all three participants. Table 8.3 shows agreement values that range between 0.919 to 0.963 (almost perfect). On average, we observed no deviations in validity or reliability comparing the two versions of our annotation tool.

Efficiency. We analyze the time required for completing the annotation task per tool and user. Overall, the slowest participant was A1, who completed the tasks for the baseline condition in 1999 seconds and 2095 seconds for the IML-support condition. Participant B1 was faster, with 1,189 seconds for the baseline condition and 1251 seconds for the IML-support condition. With 980 seconds for the baseline condition and 966 seconds for the IML-support condition, participant B2 was the fastest annotator. While the individual differences in the task completion times are large, we found only small differences in the completion times between the two conditions. On average, our participants required 1389 s to complete the tasks for the baseline condition and 3.44% longer (1437 s) for the IML-support condition. The high rater agreement indicates that there is no relation between task completion time and the validity of the generated annotations.

Usability. We measured perceived usability using the SUS questionnaire. The *baseline* version is consistently rated as "excellent" with values ranging from 87.5 to 95 (91.6 on average). For the *IML-support* condition, we observed a high variance in SUS scores: the ratings range from 50 for B1 ("poor") over 67.5 for B2 ("OK") to 97.5 for A1 ("excellent"), averaging to 71.6. The reflexive thematic analysis of the SSI revealed two themes: (a) The tool's design facilitates the annotation of mobile eye tracking data, and (b) The constrained model performance limits IML-based benefits. Details are provided in the discussion section below.

8.2.2.3 Machine Learning Experiment

In a post-hoc experiment, we assess the performance of three machine learning models in automatically annotating the part of the dataset that was not annotated during our study, i.e., all test data remains unseen. This includes around 230k fixations from 47 individuals. The automatic fixation-to-AOI mapping includes all 7 classes from our experiment plus a background (BG) class. However, the models are not trained to directly classify the background class. The background class BG is assigned if the probability is lower than a threshold $t_{BG} = 0.4$. This means fixations are assigned to one of the 7 AOIs if the probability for this classification is greater or equal to

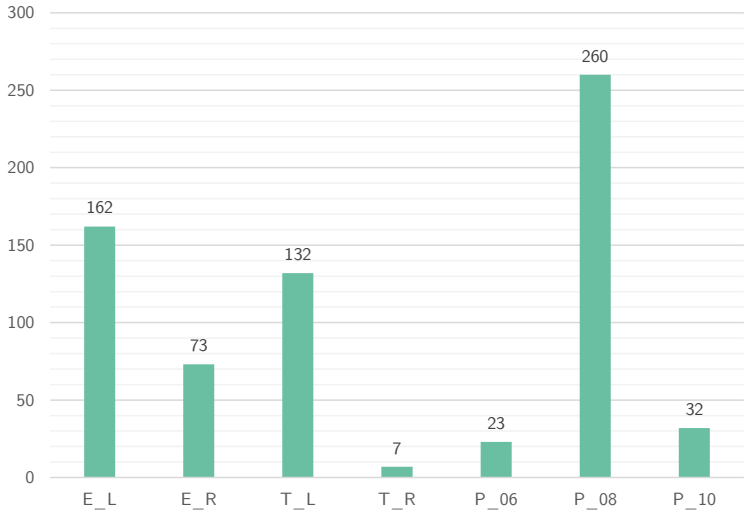


Figure 8.12: Class distribution on the training set for the post-hoc machine learning experiments. *Tablet_Right (T_R)* has the lowest and *Page8_TwoBulbs_Row (P_08)* the highest number of samples.

t_{BG}. The three considered models include the few-shot learning model (FRNet) (Wertheimer et al., 2021) that was used in our *IML-support* version, ResNet50 (ResNet) (He et al., 2016), a well-established foundation model for image classification tasks, and MobileNetV2 (MobileNet) (Sandler et al., 2018), a lightweight architecture model suitable for resource-constrained environments. We consider two data settings for model training: *base* and *final*. For the *base* setting, we use the initial labeled data pool with 10 images per class as the train set, i.e., the 70 images that were used to pre-train the FRNet model for the *IML-support* version of our tool. For the *final* setting, models are trained using ground truth labels for the 870 fixations from the annotation task. Figure 8.12 shows the class distribution for the 7 AOIs in the train set. However, by that, we assume that a participant correctly annotates all fixations, which is not exactly true but sufficient for our experiment: the average accuracy of our participants in annotating these 870 fixations was 94.55%. In the *final* setting, we train FRNet in a 100-shot-7-way manner, upsampling images for classes with less than 100 training images because the model requires an equal number of samples per class (random oversampling). Instead of upsampling, we use *weighted cross-entropy* classification loss to train ResNet and MobileNet, which addresses the class imbalance. As described in section 8.2.1.2, an additional loss with a scaling faction of 0.03 is used to train FRNet. All models are trained for 30 epochs using an SGD optimizer with a learning rate of 0.0001. We report the accuracy and f1 scores of all models.

Results

Table 8.4 reports the accuracy for each model and train setting. FRNet outperforms MobileNet and ResNet: it achieves an accuracy of 57.57% in the *base* setting and 58.78% in the *final* setting, which is 6.64% and 7.39% better than the second-best models, respectively. The model performs marginally better when taking the annotations of our participants into account for training in

test samples	*base* setting			*final* setting		
	MobileNet	ResNet	FRNet	MobileNet	ResNet	FRNet
230.3k	50.93%	39.60%	57.57%	49.28%	51.39%	**58.78%**

Table 8.4: Accuracy for each model and train setting.

class	test samples	*base* setting			*final* setting		
		MobileNet	ResNet	FRNet	MobileNet	ResNet	FRNet
E_L	10771	0.207	0.180	0.323	0.153	0.224	**0.384**
E_R	7780	0.001	0.481	**0.556**	0.006	0.457	0.463
T_L	26167	0.077	0.002	0.662	0.057	0.001	**0.687**
T_R	11407	0.183	0.316	0.570	0.144	0.087	**0.575**
P_6	14725	0.317	0.256	0.334	0.310	0.320	**0.375**
P_8	10242	0.003	0.198	0.209	0.014	0.120	**0.329**
P_10	11392	0.151	0.044	0.093	0.133	**0.153**	0.146
BG	137852	0.676	0.569	0.678	0.663	**0.681**	**0.681**
Macro Average		0.202	0.256	0.428	0.185	0.255	**0.455**
Weighted Average		0.460	0.409	0.579	0.445	0.471	**0.593**

Table 8.5: Class-wise f1-scores for each model and train setting.

the *final* setting (+1.21%). MobileNet ranks second for the *base* setting with an accuracy of 50.93%. The accuracy slightly decreases to 49.28% for the *final* setting. ResNet performs worst for the *base* setting with 39.60% and benefits most from using more training samples in the *final* setting. The accuracy increases by 11.78% to 51.39%, now slightly outperforming MobileNet.

Table 8.5 reports the class-wise and averaged f1-scores for each model and train setting. In both train settings, FRNet performs best in terms of the macro and weighted average of the f1-score. The best performance is achieved in the *final* setting with a macro average f1-score of 0.455 and a weighted average of 0.593. In the *base* setting, the macro average is 0.428, and the weighted average is 0.579. MobileNet and ResNet achieve considerably worse average f1 scores in both settings. For the *base* setting, the macro average is 0.202 for MobileNet and 0.256 for RestNet, the weighted average is 0.460 for MobileNet and 0.409 for ResNet. MobileNet does not benefit from taking more training samples into account in the *final* setting: the macro average f1 score slightly drops to 0.185, and the weighted average f1 score to 0.445. For ResNet, the macro average f1 score stays similar, while the weighted average f1 score improves by 0.062 to 0.471. However, this is still 0.122 worse compared to FRNet in the same setting and 0.107 worse than FRNet in the *base* setting. It is noteworthy that the difference between FRNet and the other two models is larger for the macro average f1 score (difference $\geq$0.172) than for the weighted average f1 score (difference $\geq$0.119). Also, the macro average f1 score is always clearly worse than the weighted average f1 score, indicating that all models perform better for classes with many samples than for classes with a small number of samples. A class-wise analysis shows that all models perform best for the background class (BG) with f1-scores starting from 0.569 for

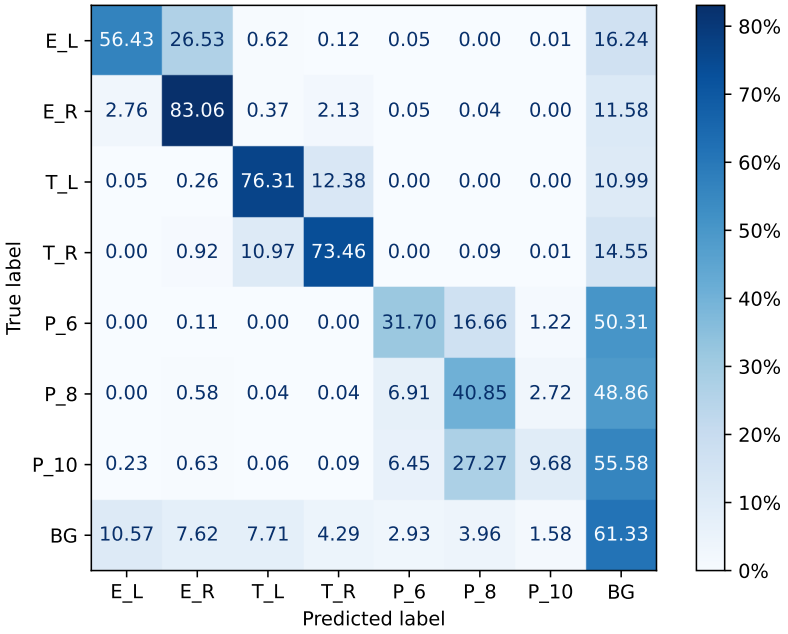


Figure 8.13: Confusion matrix for the test set for FRNet in *final* setting (normalized over rows).

ResNet in the *base* setting and larger than 0.663 for all other conditions. The best performance for the background class was observed for ResNet and FRNet in the *final* setting with an f1 score of 0.681. We only observed a single better f1 score of 0.687 for the tablet class *T_L* for FRNet in the *final* setting. As the background class covers more than half of all samples (137.9k of 230.3k samples), it has a large impact on the weighted average. For MobileNet and ResNet models, we observed low f1-scores of less than 0.5 for all seven classes other than *BG* in both settings. FRNet shows a more balanced performance. In the *base* setting, only four out of eight classes achieve an f1 score below that threshold. Further, for FRNet, we observed the best performance for each class besides *P_10* for which MobileNet was better. In the *final* setting, FRNet improves for all classes besides the experiment area *E_R* (-0.094), which is why five out of eight classes have an f1 score lower than 0.5. Still, the model performs best for all classes besides *P_10*. For *BG*, ResNet performs equally well in this setting. The best f1-scores for FRNet are observed for the background class *BG* and the two tablet classes *T_L* and *T_R*.

Figure 8.13 shows the confusion matrix of the best-performing condition: FRNet in the *final* setting. It is normalized over the true conditions (i.e., over rows): the values on the diagonal correspond to the recall of a respective class. Other values in the same row correspond to false-negative errors and sum up to the miss-rate of that class. For instance, for the background class *BG*, the recall is 61.33%, and the false negatives sum up to a miss rate of 38.66%. The background is often misclassified as one of the experiment area classes (18.19%) or as one of the tablet classes (12%). The confusion matrix shows that classes with similar appearances are frequently confused. This can be observed for the two experiment area AOIs, the two tablet

class	Precision	Recall
E_L	0.291	0.564
E_R	0.321	0.831
T_L	0.625	0.763
T_R	0.473	0.735
P_6	0.459	0.317
P_8	0.275	0.409
P_10	0.295	0.097
BG	0.765	0.613
Macro Avg.	0.438	0.541
Weighted Avg.	0.633	0.588

Table 8.6: Class-wise precision and recall for the FRNet model in the *final* setting.

AOIs, and the three workbook AOIs. For instance, for *E_L*, the recall is 56.43%, and, with 26.53%, the majority of the false negatives were classified as *E_R*. The recall of *T_L* is 76.31% while 12.38% of the false negatives were classified as *T_R*. A similar pattern was observed for the workbook AOIs *P_**. All AOI classes are frequently misclassified as background. Hereby, the false negative errors for the experiment area and tablet AOIs range between 10.99% and 16.24%. The three workbook AOIs are affected more severely: the false negative errors range between 48.86% and 55.58%. This results in a precision of 0.765 for *BG*, which is the best precision among all classes. Precision and recall for all classes are reported in table 8.6.

8.2.3 Discussion

Next, we discuss the results of our case study and the post-hoc ML experiment.

8.2.3.1 Validity & Reliability

The validity of users' annotations is high and alike for both versions of our annotation tool. We observed an accuracy of 94.76% for the baseline version and 94.55% for the IML-support version (weighted mean). Further analysis revealed 14 errors (1.6%) in the ground truth. We identified errors in cases when all three annotators agreed on an AOI that deviated from the ground truth. With a corrected ground truth, accuracy increases to 96.29% for the baseline version and 96.07% for the IML-support version, respectively. This suggests we met our goal of achieving an accuracy of at least 95%. However, annotators might have worked particularly conscientiously during the study. Our results further suggest that the *exp_2* was more difficult to annotate because accuracy values consistently dropped for both versions of the tool from more than 97% accuracy to less than 90%, and we observed a higher ratio of ground-truth errors. A reason might be that the second phase included more AOIs and a more complex scene. The inter-rater agreement was almost perfect with $\kappa \geq .9$ in all cases, i.e., the reliability of annotations from both versions of our tool is high.

8.2.3.2 Efficiency

On average, task completion times for both tool versions were similar: annotators were 3.44% (48 seconds) slower when using the IML-support version. Likewise, the difference in task completion times between versions per participant is small. On the other hand, the differences between participants are large. A1 required around 2000 seconds to solve the task per tool, while B1 and B2 required around 1200 seconds and below 1000 seconds, respectively. This is almost twice as fast without compromising accuracy, which indicates that B1 and B2 had a more efficient strategy in using our tools. During the study, we observed that all participants used shortcuts for annotation and confirmation, but A1 did not use the multi-select feature, which could explain the high difference. Given the 870 fixations in our annotation task, our interactive annotation tool achieves a worst-case annotation rate of 2.41 seconds/fixation (A1, IML-support version) and a best-case annotation rate of 1.11 seconds/fixation (B2, IML-support version). When using an ML model to map the remaining 230k fixations, this means the time-saving potential lies between 70 and 150 hours. Eventually, we could not confirm our hypothesis that providing label suggestions would accelerate the labeling process. This is likely because all annotators manually checked and confirmed label suggestions for all fixations in the IML-support version. We observed corresponding annotation behavior during the study, and theme b of our SSI analysis concerning the constrained model performance confirms this: annotators did not trust the model sufficiently and felt highly responsible for doing the job well. Hence, they did not benefit from automatic label suggestions as found in Desmond et al. (2021). A follow-up study could investigate how lay users perform in the annotation task. Can lay users achieve the same validity as trained annotators? Do lay users benefit more from label suggestions in terms of efficiency?

8.2.3.3 Usability

The usability of our tool's baseline version was consistently rated as excellent: the basic features and general interaction design of our annotation tool were perceived very positively, which is supported by theme (a) of our thematic analysis concerning the tool's interaction design: "the tool's design facilitates the annotation of mobile eye tracking data." However, B1 and B2 rated the IML-supported version drastically lower, which contradicts our assumption that both tools achieve a similar usability rating. Looking into individual SUS items, B1 and B2 majorly penalized an increased inconsistency of the IML-support version and that it was more cumbersome to use. Both felt less confident using the IML-support version and thought it was less easy to use. This can be attributed to the integration of IML-support features and relates to theme (b) of our thematic analysis concerning the constrained model performance: "the constrained model performance limits IML-based benefits." The two themes originate from a reflexive thematic analysis of the SSI that was conducted with each participant.

(a) The tool's design facilitates the annotation of mobile eye tracking data. Our case study participants liked our tool's basic functionality and interaction design. In particular, they highlighted the clean design that allowed them to focus on the annotation task throughout the study. They reported high usability and learnability. Quick reaction times and visual feedback were highly appreciated. Particularly, the video overlay immediately displaying updates after

manual annotation or confirmation was considered very helpful because they had to check the video frame to decide on the AOI class anyway. All participants reported a high perceived performance due to the clean, focused interaction design and the ability to use shortcuts for navigation and annotation. Also, the multi-select feature for annotation and confirmation seems to impact annotation efficiency positively. The video playback function was not used by our participants but might have supported understanding the video-editing-like interface metaphor. Upon asking them, participants reported they understood the trust-level slider but did not use it often, although it was considered useful. High-certainty suggestions (green highlight) were also considered helpful. However, certain but wrong label suggestions were frustrating as they could lead to wrong confirmations. Also, the red color of uncertain suggestions was reported to interrupt the interaction flow, in case the predictions were correct. In summary, color-coding of the model certainty for label suggestions might cause frustration in the case of certain but wrong predictions and can interrupt the interaction flow in the case of uncertain but correct predictions. An implication could be to restrict label suggestions to highly certain suggestions. Our participants suggested two interesting features that will be considered in future versions of our tool. They proposed a feature that enables jumping to non-annotated fixations or uncertain suggestions. Further, they proposed a feature to batch-accept all certain predictions (depending on the state of the trust-level slider).

(b) The constrained model performance limits IML-based benefits. All participants reported a perceived model performance of 30-40% accuracy, although the true value is much higher (62%). This indicates that our participants had low trust in the underlying model generating the AOI label suggestions and could explain why they checked all suggestions manually. This is also in line with their reports on problems with certainty-based color coding. All participants specified that the model suffered from a left-right-weakness: Some AOIs with the same appearance were present on the left and right sides of the experiment scene, but the model could not properly differentiate between them. We intentionally investigated this challenge by including experiment phase 2. One example is T_L and T_R, referring to two instances of the same tablet mounted on the left or right side of the experiment scene. This is evident in the confusion matrix for FRNet in figure 8.13: T_L is wrongly classified as T_R in 12.38% of the cases. The false negative errors concerning all other classes besides BG sum up to 0.31%. We observe similar problems for *experiment area* and *workbook page* AOIs. If objects look very much alike, our IML-support version has limitations. Addressing the left-right weakness is essential because AOIs with similar appearances are common. Future research should investigate whether object-tracking or position-aware models can help to address this challenge. Another option can be found in meta-models that iteratively learn for which classes a model performs well and activate suggestions for those only.

8.2.3.4 Post-hoc ML Experiment

We observed the best average f1-scores and accuracy scores when using the FRNet model architecture in the final setting, i.e., when using the 870 annotated fixations for training (see tables 8.4 and 8.5). However, using more training data for the FRNet model only slightly increases the

performance, e.g., +1.21% in accuracy and +0.015 concerning the weighted average f1-score. With +11.78% for accuracy and +0.062 for the weighted f1 score, ResNet showed the greatest improvement when more training samples were added. MobileNet performs slightly worse for all metrics. Either way, the results show that the models are not good enough for most applications such as automatic or semi-automatic annotation with humans-in-the-loop. This is in line with the user's feedback from the SSI as summarized in theme (b).

The best f1 score of 0.687 was observed for the *T_L* class for the FRNet model in the *final* setting, followed by an f1 score of 0.681 for the *BG* class. The precision is highest for *BG* with 0.765 (see table 8.6), so labeling support only for the *BG* class could have been effective. Since almost 60% of all labels belong to this class, this could already save a lot of time without raising usability issues like the ones mentioned in theme (b). The high ratio of *BG* samples in the test set also means that summary statistics like accuracy and the weighted f1-score are biased through the relatively high performance for this class. This is visible in the large deviation between the weighted and macro average f1-scores for all models. Overall, FRNet shows the most balanced performance across all classes: it performs best for all classes besides *P_10*. This also explains the greater relative difference in the macro-average values and the weighted average values for f1 for MobileNet and ResNet.

The confusion matrix in figure 8.13 shows the strengths and weaknesses of the FRNet model (final) on the class level in more detail. As counts are normalized over the true condition, i.e., over rows, the diagonal shows the recall scores for the true condition or class of that row, while the remaining values of that row sum up to the corresponding miss rate. For *BG*, we observed a recall of 61.33% with a precision of 76.53%. This means that, when limiting suggestions to the *BG* class, labels for more than one-third of all instances (61.33% of 59.88% of all 230.3k instances) could have been provided, of which around three-quarters would have been correct. Still, one-quarter would have been wrong. So, limiting suggestions to *BG* alone would likely not solve the usability issues mentioned in theme (b). These scores were observed for the default setting when *BG* is assigned if the model's classification probability for an AOI class is lower than $t_{BG} = 0.4$. Lowering t_{BG} would increase the precision for the *BG* class but at the cost of a lower recall. Likewise, increasing the threshold for assigning one of the 7 AOI classes, we call it t_{AOI}, would increase the precision for these classes. Eventually, a class-specific batch-accept feature for accepting label suggestions for a certain class with manually tuned t_{BG} and t_{AOI} could be useful. The user should be able to configure the probability threshold t_{BG} and the classification thresholds t_{AOI} for each class, which would allow annotators to accept labels based on their own experiences of how the model performs per class. However, most f1-scores and all precision scores for AOI classes are lower than the scores for the *BG* class (see table 8.6), which indicates that tuning the thresholds for a batch-accept feature might be difficult. We conduct and report on a follow-up experiment that investigates how changes in t_{BG} and t_{AOI} affect the classification performance and relate to the number of fixations without a label suggestion. By that, we aim to estimate the potential of a class-wise batch-accept feature.

The confusion matrix also indicates that a reason for the low f1 scores is the similar appearance of the AOI classes, including the two experiment areas *E_**, the two tablets *T_**, and the three workbook pages *P_**. These three groups can be clearly identified along the diagonal as three squares based on the high number of false negative errors within each group. Further, it shows

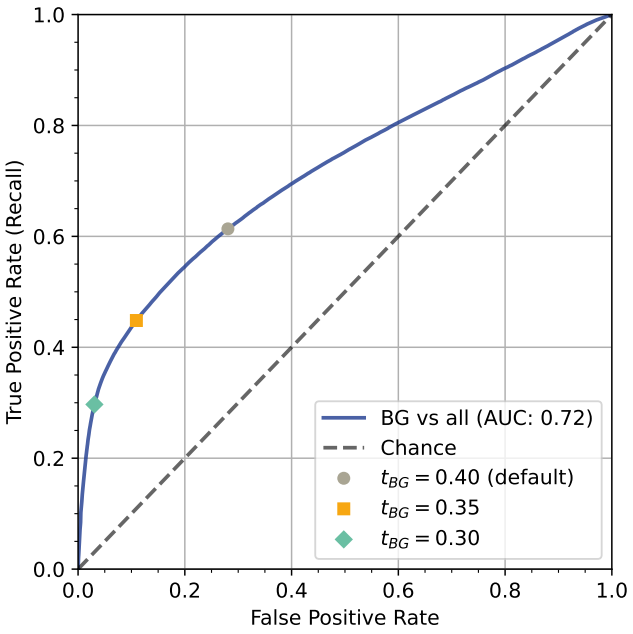


Figure 8.14: ROC curve for the background class *BG* for the FRNet model in the *final* setting. The decision boundary corresponds to the threshold $t_{BG} = t_{AOI}$.

that many AOI classes are frequently misclassified as belonging to the background class *BG*, particularly the three workbook AOIs. Confusion of AOI classes with the *BG* class could be reduced by increasing the classification threshold t_{AOI}. This could be realized, e.g., through a class-based trust-level slider. Confusion of similar-looking AOI classes can only be solved by using more suitable approaches like multi-object tracking, i.e., once an AOI was manually labeled or confirmed by a user, the system could track this instance to reveal wrong classifications or auto-confirm true classification, or graph neural networks that consider the spatial location of an object for classification (Le et al., 2025). An option to increase the utility of the FRNet model would be to provide label suggestions at a higher semantic level. For instance, eyeNotate could identify all tablets and ask the user which instances belong to the left (T_L) or right (T_R) class. Similarly, this could be done for the two experiment areas and the three workbook pages. Classification performance would likely be higher for this 4-class problem because it is a less complex classification problem. We investigate this aspect in another follow-up experiment. Further, a 2-level decision task (left vs. right) or 3-level decision task in the case of the workbook pages is less difficult for users than the 8-level decision task, which includes all AOIs and the separate background class.

Next, we report on the two mentioned follow-up experiments: one for estimating the utility of a class-wise batch-accept feature and one for investigating how the model would perform for the 4-class classification problem.

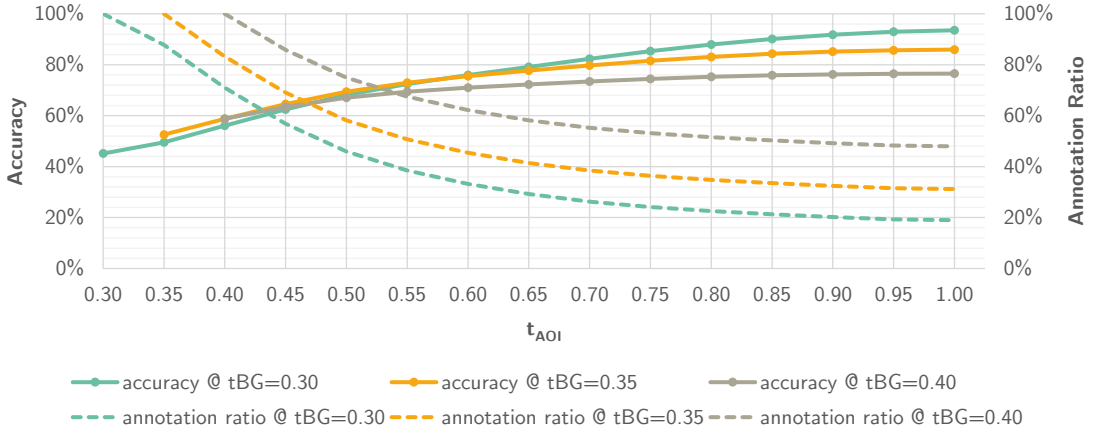
Estimating the Utility of a Class-wise Batch-accept Feature

To estimate the utility of a class-wise batch-accept feature, we investigate the impact of adjusting the classification thresholds t_{BG} and t_{AOI} on the model performance in an additional experiment. In the current setting, eyeNotate suggested *BG* as a label when the probability was below a threshold of $t_{BG} = 0.4$ and the highest-ranked AOI class otherwise. In this post-hoc experiment, we add a second threshold t_{AOI} that determines the minimum classification probability p before we assign an AOI class. The higher the gap between these two thresholds, the higher will be the number of instances without a label suggestion. Hence, there will be a trade-off between the number of instances with a label suggestion and the precision of those.

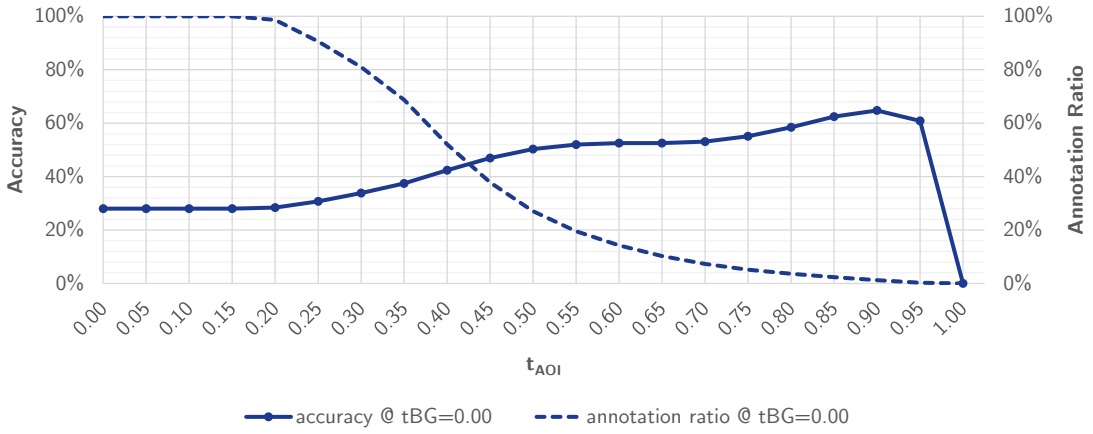
In the first step, we assess whether the default threshold for classifying the background class $t_{BG} = 0.4$ was a good choice. For this, we plot a ROC curve that illustrates the trade-off between the true positive rate (recall) and the false positive rate for classifying the *BG* class (versus all other AOI classes) depending on t_{BG} (see figure 8.14). Note that in the default setting, $t_{AOI} = t_{BG}$. The ROC curve shows that false positive rate for $t_{BG} = 0.4$ is quite high: 28.07% of non-*BG* instances are wrongly classified as *BG*. Reducing t_{BG} to 0.35 or 0.30 improves the false positive rate: only 10.92% or 3.06% are wrongly classified as background. The recall would drop to 44.83% and 29.96%, respectively. A recall of 29.96% still corresponds to 17.94% of all samples (41.3k) because 59.88% of all 230.3k samples belong to the *BG* class.

However, simultaneously reducing t_{BG} and t_{AOI} optimizes the false positive rate for the background class but will also lead to an increase in false positive rates for all other classes. Hence, we investigate the impact of increasing t_{AOI} in 5% steps on accuracy with constant t_{BG} for $t_{BG} \in \{0.3, 0.35, 0.4\}$. At the same time, we investigate the impact on the number of samples that will not be annotated. The results are presented in figure 8.15a. It shows the model accuracy and the annotation ratio, i.e., the portion of samples that received an annotation suggestion, as a function of t_{AOI}. Using the default parameters $t_{BG} = t_{AOI} = 0.4$, we observe an accuracy of 58.78% as reported in table 8.4 for FRNet in *final* setting. The annotation ratio is 100% because $t_{BG} = t_{AOI}$. For $t_{BG} = t_{AOI} = 0.3$, the curve starts with an accuracy of 45.15%. For $t_{BG} = t_{AOI} = 0.35$, accuracy starts with 52.58%. In all three cases, the accuracy increases and the annotation ratio decreases with increasing t_{AOI}. Setting $t_{AOI} = 1$ means, we do not consider annotations for any class besides *BG*. For $t_{BG} = 0.4$, the accuracy reaches 76.53% and the annotation ratio 57.96% in this setting. We observe that the lower t_{BG}, the lower the accuracy, and the higher the annotation ratio. Consequently, the maximum accuracy is reached for $t_{BG} = 0.3$ with 93.54% as well as the minimum annotation ratio of 18.97%. However, for $t_{AOI} = 1$, prediction labels would be limited to *BG*. This indicates that a batch-accept feature for *BG* could be effective. For a batch-accept feature that includes other classes than *BG*, t_{AOI} must be smaller than 1. To assess how well the model would perform for AOI classes only, i.e., for all classes besides the background class *BG*, we ran the experiment for $t_{BG} = 0$ and $0 \leq t_{AOI} \leq 1$. The corresponding diagram is shown in figure 8.15b. Up to $t_{AOI} = 0.15$, all samples are classified as one of the AOI classes. This means that the minimum model certainty lies between 0.15 and 0.2. With increasing t_{AOI} the accuracy also increases until it reaches its maximum for $t_{AOI} = 0.9$ with 64.75%. However, with these parameters, only 1.24% of all samples would be annotated.

Overall, the results of this additional experiment indicate that a batch-accept feature for the



(a) $t_{BG} \in \{0.3, 0.35, 0.4\}$



(b) $t_{BG} = 0$

Figure 8.15: Accuracy and annotation ratio as a function of t_{AOI} for the FRNet model in *final* setting for $t_{BG} \in \{0.3, 0.35, 0.4\}$ (a) and for $t_{BG} = 0$ (b).

AOI	# Samples	precision	recall	f1 score
E	18551	0.380	0.842	0.524
T	37574	0.661	0.874	0.753
P	36359	0.598	0.479	0.532
BG	137852	0.765	0.613	0.681
macro avg		0.601	0.702	0.622
weighted avg		0.691	0.653	0.656

Table 8.7: Class-wise precision, recall, and f1-scores for the FRNet model in *final* setting for a reduced set of four target classes.

background class *BG* could add value to eyeNotate. Since the parameters are optimized over the test set, the results can only serve as an upper bound of the performance. In a realistic scenario, the performance with a human optimizing the parameters would lie below this upper bound, but it would, in theory, be reachable for the considered usecase, dataset, and model. However, the results also show that the classifier is not good enough for providing label suggestions for AOI classes, even under the assumption that users could tune the decision thresholds. A reason is likely the high similarity between some of the AOI classes.

Simulating Model Performance in a 4-class Classification Setting

Another option to increase the utility of eyeNotate using the FRNet model is to tread the classification as a 4-class problem, i.e., to only differentiate between the background class *BG* and three further AOI classes: experiment area *E*, tablet *T*, and workbook pages *P*. For our usecase, the human annotator would still need to decide whether, e.g., the identified tablet is the left or right version. But this decision is less complex than assigning one out of all eight classes. Also, this investigation can reveal the potential benefit of eyeNotate for other, more simple usecases. Hence, we assess the overall accuracy and the precision, recall, and f1-scores under the assumption that only four target classes exist, i.e., *E*, *T*, *P*, *BG*, using the FRNet model in the *final* setting. For this, we replace the true and predicted class labels with the corresponding summary class, e.g., *E_L* and *E_R* are replaced with *E* before computing scores. The *BG* labels do not change.

In the 4-class setting, FRNet achieves an accuracy of 65.30% which is 6.52% better than in the original 8-class setting. Table 8.7 shows the corresponding precision, recall, and f1-scores. As expected, the scores for summary classes are better compared to the original classes. For instance, for *E*, we observe an f1 score of 0.524, while the f1 scores for *E_L* and *E_R* are 0.384 and 0.463, respectively. This also holds for *T* and *P*. The results do not change for *BG* because there were no changes concerning the background class. Consequently, the macro average and weighted average f1 scores are also higher. The macro average f1 score increases by 0.167 and the weighted average f1 score by 0.063.

In summary, reducing the complexity of the classification problem has a positive effect on all observed scores. However, to enable effective annotation support we will need to further improve the model performance. Promising directions that should be investigated include methods like

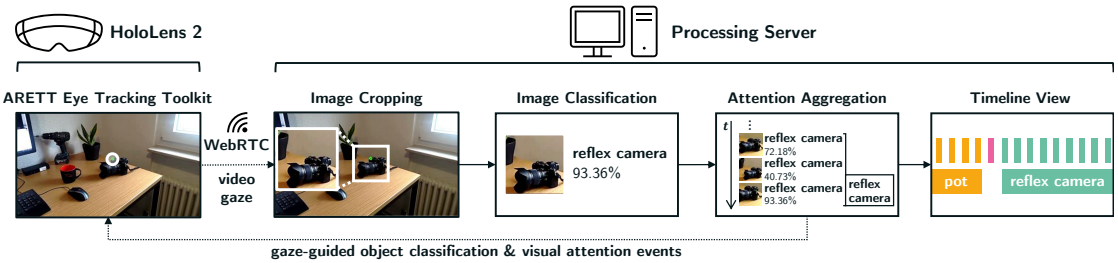


Figure 8.16: Architectural overview of our real-time attention-aware interactive prototype system.

multi-object tracking and graph neural network models.

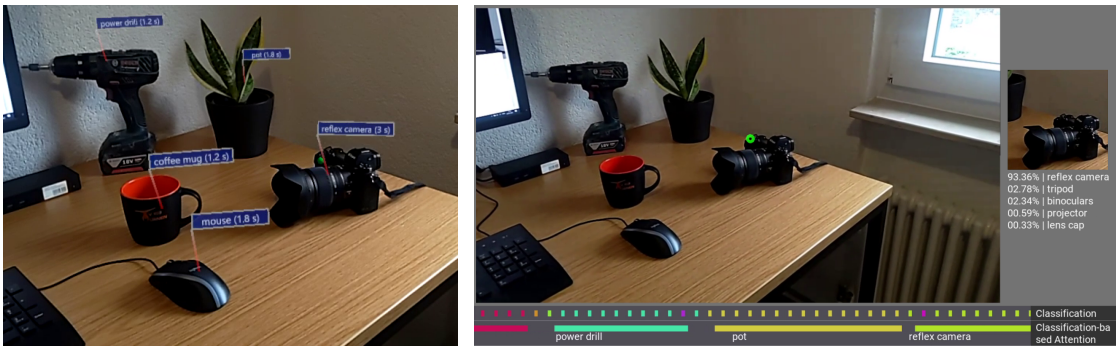
8.2.4 Towards Real-time Attention-aware Interfaces

We showcase that our methods for detecting visual attention can facilitate the development of real-time attention-aware user interfaces. For this, we develop a prototype that identifies and visualizes objects fixated by a user in real-time using a head-mounted eye tracker, particularly using the HoloLens 2 which comes with an integrated eye tracking sensor. We extend our approach for visual attention detection presented in section 8.1 to implement a real-time prototype for situation-aware interaction in AR. We use our ARETT toolkit (Kapp et al., 2021) to connect the HoloLens 2 as an eye tracking and video source and to augment the attended objects. To the best of our knowledge, we are the first to present an AR system that augments fixated objects with their class in real-time based on the user's gaze. This technology can enable cognition-aware mobile user interaction in AR settings. It bears the potential to provide benefits in several application domains, e.g., healthcare, industry 4.0, research & education, and entertainment. This is also approved by our two ongoing research projects that build up on top of this technology (see part V).

Our prototype is based on the HoloLens 2, a head-mounted device for AR systems with integrated eye tracking capabilities, and a GPU-accelerated processing server for visual attention detection (see figure 8.16). The eye tracking signal and the video stream data of an egocentric camera are acquired using the ARETT toolkit (Kapp et al., 2021) for HoloLens 2. We implement an extension that offers both sensor streams to a local network via WiFi using the WebRTC protocol. On the processing server, we run an instance of the presented fixation-to-AOI mapping algorithms (see sections 8.1 and 8.2). Our system supports real-time video and gaze streaming via WebRTC from the HoloLens 2 device. The output of the visual attention detection is visualized on the processing server using the event timeline monitoring tool and our prototype application on the HoloLens 2 (see figure 8.17).

8.2.4.1 HoloLens 2 Application

We implement our prototype using the Microsoft HoloLens 2 and the Unity 3D game development engine (Barz et al., 2021b). It integrates an extension of the Augmented Reality Eye Tracking Toolkit for Head Mounted Displays (ARETT) (Kapp et al., 2021) for eye tracking data acquisition



(a) Snapshot from the HoloLens 2 application **(b)** Timeline monitoring tool

Figure 8.17: Our prototype classifies and augments fixated objects in real-time. It displays classification labels and the duration of recent attention events as a hologram in HoloLens 2 (a). The classification and attention detection pipeline runs on a separate server with a real-time monitoring tool (b).

and streaming using the WebRTC protocol. The ARETT toolkit is extended to include mapping of gaze data to a video feed and real-time streaming of gaze data using the WebRTC protocol. ARETT utilizes the APIs provided by the device to reliably acquire gaze data in terms of gaze origin and direction. The raw data is processed to identify the gaze position by casting a gaze ray into the virtual environment. However, the data from this toolkit does not provide a gaze mapping to the world camera. Our extension accurately maps the gaze data to the camera image of the front-facing webcam. For this, we use a virtual camera in the virtual scene that matches the location, projection and resolution of the real camera. This enables the projection of the 3D gaze position to the virtual 2D camera image and to the webcam image. The video and gaze data is streamed to the processing server in real-time using the WebRTC protocol via the MixedReality-WebRTC library[10]. Besides video streaming, it provides data channels for streaming the gaze signal and retrieving classifications and attention events from the processing server. Our HoloLens 2 demo application offers two options for data visualization that correspond to the output of the visual attention detection system: *real-time image classifications* and *visual attention events*. The *visual attention events* view shows the last five attention events (see figure 8.17a): for each attention *start* event, we add a tooltip with the class label and the event duration at the object location which is continuously updated. The *real-time image classifications* view shows a tooltip at the latest gaze position, indicating the top-1 classification candidate and its probability. Both views can be activated and deactivated using a virtual menu attached to the user's hand.

8.2.4.2 Visual Attention Detection

Our method for visual attention detection includes three subsequent steps. First, we re-sample the gaze signal to 5 Hz and crop an image patch of 200×200 pixels from the egocentric video feed (960×540) around the point of gaze for each remaining sample. This re-sampling ensures our

[10] `https://github.com/microsoft/MixedReality-WebRTC/tree/v2.0.1` (accessed on 12 Dec 2024)

system's real-time capability. Second, each patch is classified using a pre-trained version of the ResNet image classification model (He et al., 2016) which is trained on the ImageNet dataset with 1001 object classes (Russakovsky et al., 2015). Third, we use a working memory- and threshold-based attention detection algorithm that uses the top-1 predictions as continuous input (Barz and Sonntag, 2016). The top-5 class candidates and the visual attention events are forwarded to the timeline monitoring tool (see figure 8.17b) and sent to the HoloLens 2 demo applications for visualization (see figure 8.17a). The visual attention events consist of *start*, *confirmation*, and *end* events which allow us to display the event with a low delay and to count up the event duration until it ends. The timeline monitoring tool allows a joint visualization of all outputs from our visual attention detection. It shows the video stream of the front-facing camera and the raw gaze signal (green circle) from the HoloLens 2 in the upper left view. The top right view shows an image patch cropped around the gaze point and the corresponding image classification output five times per second (see *Image Classification* module in figure 8.16). Two timelines are shown at the bottom, one showing the classification output as color-coded bars (corresponds to the top-1 result of the image classification output) and another showing the reported attention events. The bars move from right to left.

8.3 Conclusion

In this chapter, we investigated approaches for semi-automatic mapping of fixations to AOIs, which can enable efficient analysis and interpretation of humans' complex interaction behavior.

In the first step, we developed and evaluated two methods for automatically detecting visual attention to ambient objects based on pre-trained computer vision models. We systematically assessed their ability to map fixations to AOIs and identified several limitations. For this, we defined an evaluation framework based on the VISUS dataset by Kurzhals et al. (2014a) and identified four challenges for methods that map gaze to AOIs. We used a set of fine-grained metrics by Ward et al. (2011) from the field of activity recognition to evaluate our visual attention to AOI mapping methods. Our methods performed well for AOIs with distinct concepts that have a strong match to the pre-trained model classes. However, several limitations impede our goal of accelerating and objectifying AOI annotation in eye tracking research. For instance, our methods drop in performance when a concept is not supported, when two instances of the same concept cannot be disambiguated, or when gaze estimation errors occur.

In the second step, we aimed to overcome these limitations, i.e., the lack of flexibility and quality assurance through humans-in-the-loop. Toward this goal, we presented eyeNotate, an interactive annotation tool for mobile eye tracking data based on few-shot image classification. The results of a case study confirmed that eyeNotate effectively enables fixation-to-AOI mapping: users liked the basic functionality and interaction design, and the validity and reliability of users' annotations were high. However, we observed that providing AOI label suggestions in the *IML-support* version did not increase the efficiency, likely because of performance issues of the model that led to low trust in the trained annotators. Still, our results suggested that FSL bears great potential for initiating interactive data annotation. Overall, the task completion times were low, with 1.11 s per annotation (best-case) to 2.41 s (worst-case). We identified constrained

model performance as the main hindering factor, especially problems with similar-looking AOIs. Future research should focus on the development of more sophisticated approaches that can cope with the dynamic and complex nature of mobile eye tracking data, for instance using multi-object tracking, 3D reconstruction methods (Kopácsi et al., 2023), and graph neural networks (Le et al., 2025). Further, it is important to investigate the role of humans in interaction with machine learning algorithms (cf. section 11.4.4).

Eventually, we demonstrated that our technology can be used in real-time interactive systems. We presented a real-time AR system that augments the user's field of view with information on recently attended objects. We implemented our system using Microsoft's state-of-the-art head-mounted display HoloLens 2 with an integrated eye tracking sensor. In the healthcare domain, the recognized sequence of visual attention events can be used to support patients with mental diseases such as dementia by complementing missing parts of their episodic memory (Orlosky et al., 2014; Sonntag, 2015) or to provide AR-based decision support for doctors (Sonntag, 2014). The education domain can also benefit from this technology, for example it could be used for eye tracking-based learning support based on multimedia effects (Alemdag and Cagiltay, 2018). In the context of experiment-based learning augmented reality has recently gained increasing interest (Thees et al., 2020; Kapp et al., 2020). However, current approaches are limited to stationary settings and are unable to integrate eye tracking data for real-time feedback and support. Another usecase in education is Multimodal Learning Analytics (MLA). The goal of MLA is to track the learning process using multiple sensor streams and interaction modalities as input to better understand the complex nature of how students learn (Blikstein, 2013; Oviatt, 2018). The technology can also be used to drive innovation in other domains. Examples include but are not limited to, gaze-based symbol grounding (Mehlmann et al., 2014; Ijuin et al., 2019; Barz et al., 2017) and language learning (Matuszek, 2018) in speech-based human-robot interaction or pervasive attentive user interfaces (Bulling, 2016).

Part IV

Towards a Gaze-based Multimodal Interaction Framework

Chapter 9

The Multisensor-Pipeline (MSP): A Framework for Prototyping Multimodal-Multisensor Interfaces

In this chapter, we present the multisensor-pipeline (MSP), a lightweight, flexible, and extensible framework for prototyping multimodal-multisensor interfaces (see section 1.1.1) based on real-time sensor input like gaze from eye trackers. We describe the Python-based framework, showcase how it can be used to implement gaze-based multimodal interaction, and discuss the relation to methods and approaches presented in this thesis. MSP integrates our experiences from developing gaze-based multimodal interaction in parts II and III and, as such, is a direct result of this thesis.

For many applications, multimodal interfaces have long been recognized to be more robust, accurate, and preferred by users than unimodal ones (Oviatt and Cohen, 2015). This is the case because users can choose the individually most suitable modality or modality combination for solving the problem at hand. Major challenges in research on multimodal interfaces include integrating new sensors and the development of effective algorithms for multimodal signal analysis (Oviatt et al., 2017b; Sebe, 2009). This requires a suitable software infrastructure that allows researchers to prototype and adapt or extend novel interaction systems effectively and efficiently (Oviatt et al., 2019; Serrano et al., 2008). We present the MSP as a solution that is published on GitHub under an open-source license and available via the Python Package Index (PyPI) repository. It enables researchers and developers to easily build and adapt stream processing pipelines by connecting existing or custom modules: it allows them to easily integrate multiple sensors or other data streams via source modules, to add stream and event processing capabilities via processor modules, and to connect user interfaces or databases via sink modules in a graph-based processing pipeline. Our framework introduces a minimal set of concepts and provides convenience functions for building and running processing pipelines. Our framework is implemented in Python with a low number of dependencies, which enables a quick setup process, execution across multiple operating systems, and direct access to cutting-edge machine learning libraries and models. We showcase the functionality and effectiveness of MSP through a sample applica-

tion. It connects a mobile eye tracker to classify image patches surrounding the user's fixation points using a pre-trained deep learning model and visualizes the top-5 classification results in real-time. Related work and details on the implementation are described in sections 9.1 and 9.2. In section 9.3, we outline the strengths and limitations of MSP and outline how the methods and approaches presented in this thesis relate to MSP.

9.1 Related Systems and Frameworks

MSP is related to real-time stream and event processing frameworks. This includes modern streaming systems for distributed event processing which are used to implement, e.g., mobile and web-based applications at scale. One example is Apache Flink[1], a framework for batch and stream processing based on dataflow graphs (Katsifodimos and Schelter, 2016). In Apache Flink, a dataflow graph is represented as a directed acyclic graph that defines operators (nodes) and data streams (edges). These systems are designed for scalable, high-throughput data analysis with fault tolerance which introduces overheads for, e.g., the installation and development process. Other frameworks support the development of multimodal dialogue systems. For instance, Advisor is a modular and extensible framework for developing socially engaged dialogue systems (Li et al., 2020). It can be used to create conversational agents with available modules for, e.g., social signal processing and speech processing. Likewise, the EEVA framework enables the development of interactive social agents for the web (Polceanu and Lisetti, 2019). SiAM-dp is a platform for developing multimodal dialogue systems. It has a focus on integrating distributed input and output devices in cyber-physical environments (Neßelrath, 2016), based on SmartWeb's iHub platform (Reithinger and Sonntag, 2005; Reithinger et al., 2005; Sonntag, 2010). With MSP, we introduce a middleware (Feld et al., 2019) that eases the development of such dialogue systems or similar applications that require real-time multimodal signal processing.

More closely related frameworks aim for synchronized processing of real-time sensor inputs in the domain of multimodal user interaction. The Social Signal Interpretation (SSI) framework[2] enables real-time recognition of social signals (Wagner et al., 2013). It supports a range of sensors and provides a set of ready-to-use modules for, e.g., signal filtering, feature extraction, and machine learning. Developers can add new components written in C++ or Python using their application programming interface. The framework is actively maintained and licensed as open source. NOVA is built on top of SSI and enables interactive data annotation using semi-supervised active learning techniques (Heimerl et al., 2019; Baur et al., 2020). SSJ is an android port of the SSI framework that enables real-time signal processing on mobile phones in the wild (Damian et al., 2018). The open source Platform for Situated Intelligence (\psi) aims to support the rapid development and study of multimodal, integrative AI systems (Bohus et al., 2021). Similar to SSI, it provides a set of components for capturing and analyzing data streams from multiple sensors that can be connected to versatile processing pipelines. Also, they offer a set of tools for debugging, data visualization, annotation, and analysis. \psi is implemented in C# and licensed as open source[3]. Unlike SSI and \psi, MSP comes with a concise set of concepts

[1] `https://flink.apache.org/` (accessed on 12 Dec 2024)
[2] `https://hcai.eu/projects/ssi/` (accessed on 12 Dec 2024)
[3] `https://github.com/microsoft/psi` (accessed on 12 Dec 2024)

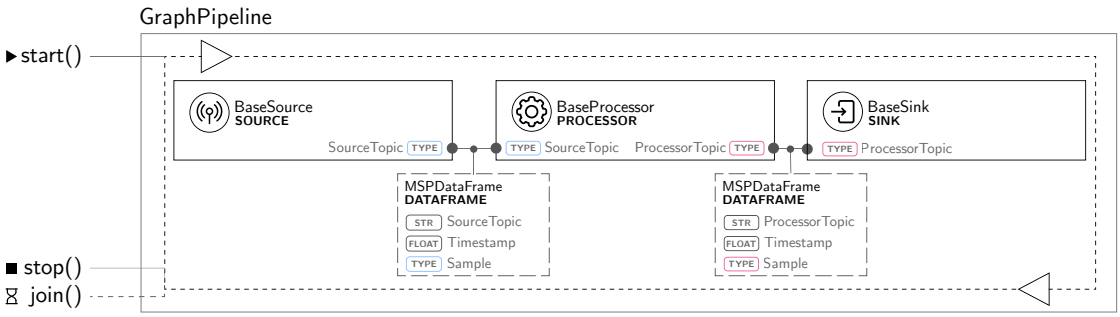


Figure 9.1: Architectural overview and programming interface of a general pipeline.

and functionalities only that ease the creation and execution of complex processing pipelines. We leave the integration of sensors and the implementation of algorithms to the researchers and developers of multimodal interfaces.

9.2 Implementation of the Multisensor-Pipeline

The multisensor-pipeline (MSP) is a framework for prototyping sensor-based user interfaces based on real-time sensor input (Barz et al., 2021a). It enables researchers and developers to build complex processing pipelines from existing or custom modules with a low overhead. A pipeline consists of at least one source module, one sink module, and an arbitrary number of processor modules which form a weakly connected, directed graph. A generic example is shown in figure 9.1). Data frames originating from sources or processors flow along the directed edges to connected processors or sinks. We released MSP as open-source software on GitHub[4] which can be installed from source or the official Python package index (PyPI) using the pip package manager. Next, we describe the two main concepts of our framework, namely the module and the pipeline, and a sample application based on MSP.

9.2.1 Modules

The module is a core concept of MSP, which is represented by the `BaseModule` class. A module can be a source, processor, or sink by inheriting from the `BaseSource`, `BaseProcessor`, or `BaseSink` class, respectively, each inheriting from the `BaseModule` class. The communication between modules is realized using the observer pattern. As an example, figure 9.2 shows the general architecture of a processor module, which combines the functionality of a sink and a source. Source modules integrate sensors or other data sources, such as a camera or an eye tracking device, and notify registered processors or sinks of new data samples. Processor modules can subscribe to source modules and other processors. Processors consume and process incoming real-time signals or events and notify their observers of the result. Sink modules can subscribe to source modules and processors. Sinks only consume incoming data samples and, for instance,

[4] `https://github.com/DFKI-Interactive-Machine-Learning/multisensor-pipeline` (accessed on 12 Dec 2024)

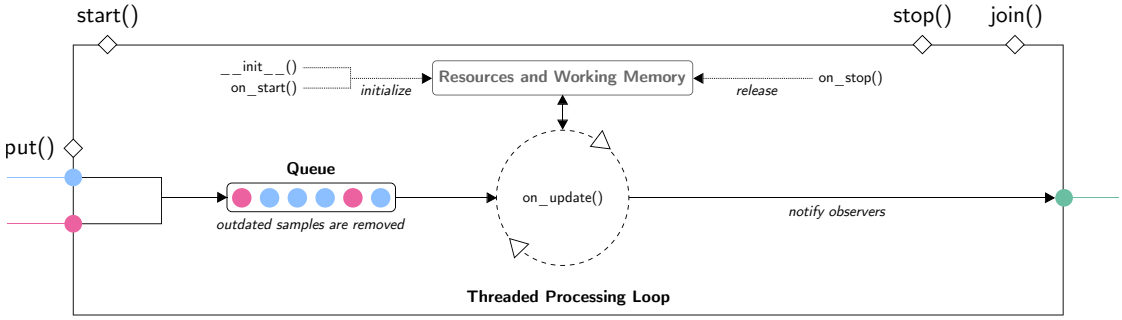


Figure 9.2: Architectural overview and programming interface of a processor module.

visualize or store the received information. Each module offers a `start()` method, which calls the `on_start()` method and starts the processing loop in a separate thread. The loop frequently calls the `on_update()` method, which implements the event-handling logic of the module. It may return `None` if no further action shall be triggered or an instance of `MSPDataFrame` to transmit a data sample or an event. `MSPDataFrame` adds metadata to each data sample which can be used by connected processors and sinks down the line: an obligatory timestamp, a unique name describing the sample, and a data type. Received instances are buffered in a queue. If a sample becomes obsolete because too many samples arrive or processing takes too long, it is removed from the queue. Stopping a module can be initiated using its `stop()` method. It interrupts the processing loop and triggers a call to `on_stop()`. The methods `on_start()` and `on_stop()` can be used to manage the resources required at runtime. To avoid unresponsive behavior, large resources like pre-trained deep learning models should be loaded in the `__init__()` method of a module. A call to `join()` allows the calling thread to wait until the processing loop is stopped and all resources are released. To implement a custom source, processor, or sink module, developers must override the abstract methods of the respective base class.

9.2.2 Pipeline

Another core concept of MSP is the pipeline implemented by the `GraphPipeline` class. A `GraphPipeline` instance is a handle to manage a processing pipeline and its modules. The modules and connections between those are represented as a weakly-connected, directed graph. The pipeline class offers functions to add modules in this graph as nodes and connections as edges. This makes it easy to replace, reorder, or add modules to update the functionality of a pipeline. For instance, a processor module that classifies images using a machine learning model can be replaced by another module using a more advanced model. Likewise, a processor module could be replaced during experiments to investigate the corresponding effect on the utility and usability of a multimodal interface. The pipeline class also offers convenience functions to start, stop, and join a processing pipeline. A call to the `start()` method triggers the `on_start()` methods of all modules and starts their threaded processing loops. The pipeline runs until its `stop()` method is called or until all subscribed sources send an *end of stream* control message. In both cases, the main thread can wait until the pipeline stops using its `join()` method.

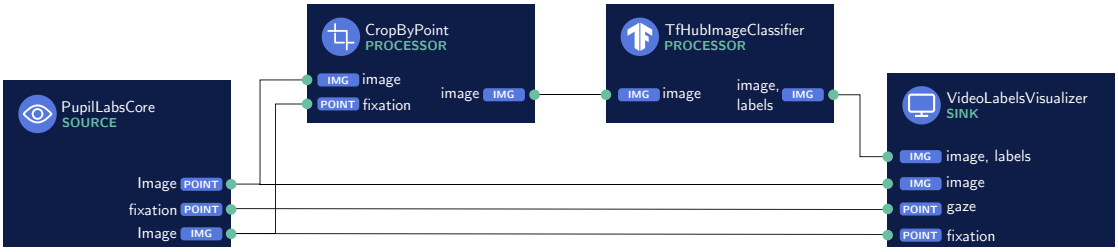


Figure 9.3: Pipeline of our sample application with a source, two processors, and one sink module.

9.2.3 Sample Application

We demo the functionality of MSP by implementing a sample application using a mobile eye tracker. This worked example is similar to the systems in chapter 8: It classifies objects fixated by a user and shows the results as a video overlay in real-time. We implement four modules for this application: an eye tracking source, an image cropping processor, an image classification processor, and a video visualization sink. The corresponding graph is shown in figure 9.3.

The eye tracking source `PupilLabsCore` inherits from our `BaseSource` class to connect a Pupil Labs Core mobile eye tracker. We use the networking API[5] of their recording tool Pupil Capture (Kassner et al., 2014) which provides real-time access to the eye tracking data. We connect the video feed of the front-facing camera (30 Hz), the raw gaze signal (200 Hz), and recognized fixation events. Typically, fixations last around 200 to 400 ms (Holmqvist and Andersson, 2017). The image cropping processor `CropByPointer` inherits from the `BaseProcessor` class. It fuses two input modalities, a video stream and a 2-dimensional signal that lies within the coordinate system of the video frames. The processor crops an image patch of a configurable size around each new point using the latest available image frame. In our example, both signals originate from the eye tracking source. We crop quadratic image patches with an edge length of 200 pixels which turned out to work well in a similar scenario (Barz et al., 2021b). The image classification processor `TfHubImageClassifier` is a wrapper around TensorFlow Hub[6] which provides access to a range of pre-trained machine learning models. It connects image classification models that are pre-trained on the ImageNet dataset (Russakovsky et al., 2015). Inference runs on the local CPU or GPU using a downloaded structure and weights of a model. The model can be set using the URLs from TensorFlow Hub. We use a pre-trained[7] ResNet-50 (He et al., 2016) model in our example. Eventually, the module annotates image patches with the top-k class labels and their probabilities (k is configurable and set to 5 in our example). The images originate from the image cropping processor. The video visualization sink `VideoLabelsVisualizer` consumes four input streams: a video stream, a 2D signal for gaze samples and one for fixation events, and the image classification output, including an image patch and corresponding class labels with probabilities. For each incoming video frame, we visualize the most recent eye movement data and classification results as an overlay and render the new image to a window on the screen (see figure 9.4). The input comes from the eye tracking source and the image classification processor.

[5] `https://docs.pupil-labs.com/core/software/pupil-capture/#network-plugins` (accessed on 12 Dec 2024)
[6] `https://www.tensorflow.org/hub` (accessed on 12 Dec 2024)
[7] `https://tfhub.dev/google/imagenet/resnet_v2_50/classification/5` (accessed on 12 Dec 2024)

Figure 9.4: Screenshot of the visualization sink of our sample application. The user's recent gaze (red dot) and fixation point (green circle) are on a loudspeaker, which is the top-1 prediction for the corresponding image patch (upper left).

9.3 Discussion

Our sample application shows that MSP can be used to prototype multimodal-multisensor interfaces effectively. Our framework is flexible and extensible due to its module-based architecture and because it is implemented in Python, which provides cross-platform support and access to a broad range of data science and machine learning tools maintained by an active community. A processing pipeline can easily be changed by rearranging existing modules and extended by adding new ones. For instance, we developed four new modules for the example above: one integrates an eye tracking sensor, and one facilitates access to image classification models from the tensorflow-hub model zoo. A web-based frontend could easily replace the current visualization sink. Likewise, other researchers might contribute to the module ecosystem of the open-source MSP framework. MSP is lightweight because it has only a few dependencies, which enables a quick installation process and simplifies the integration into other software projects. We limit MSP to a concise but powerful set of concepts and functionalities: module instances can be connected to a processing pipeline with convenient functions for starting and stopping them. Implementing modules and connecting them to a functioning pipeline are left to researchers and developers. We offer a selection of basic modules, such as a webcam source, a video output sink, and sample pipelines in our test suite.

The development of MSP incorporates our experience from implementing several real-time gaze-based interfaces presented in this thesis. For instance, our experience with implementing error-aware gaze-based interfaces in chapter 4 was a cornerstone for the development of MSP.

Further systems were based on MSP. For instance, in section 8.1, it was used to implement a system for automatic detection and annotation of visual attention to areas of interest (Barz and Sonntag, 2021). In section 8.2.4, we implemented an augmented reality system that classifies fixated objects and augments the real objects with virtual labels that stick to them in real-time (Barz et al., 2021b). We integrated Microsoft's HoloLens 2 with an integrated eye tracking sensor as an MSP source module using the augmented reality eye tracking framework ARETT (Kapp et al., 2021). In chapter 7, MSP was used to implement the study software for collecting gaze data from participants during a stationary reading task (Barz et al., 2022).

Nevertheless, some limitations remain, mainly related to multiprocessing, networking capabilities, and data synchronization. For instance, MSP offers a wrapper that starts a module in a separate process, enabling the execution of a processing pipeline across multiple physical CPU cores. However, the current method for inter-process communication is not efficient and is limited to transferring basic datatypes and numpy arrays (Harris et al., 2020). Another limitation arises from MSP's networking support: dataframes can be streamed via a network using its networking sink and source modules. However, each pipeline must be started separately to run a processing pipeline across multiple computers in a local network. Like multiprocessing, the transfer of dataframes via a network suffers from the inefficient serialization approach. Another limitation is the simplistic approach to data stream synchronization. Currently, modules drop outdated samples to avoid the propagation of delayed messages. We plan to address these limitations in future MSP releases. For instance, we plan to enable better support for multiprocessing by integrating efficient and fast inter-process communication for dataframes. Further, we aim to integrate more advanced data synchronization methods and multimodal data fusion approaches. This could be realized through custom processor modules or by extending the `BaseSink` class accordingly. Eventually, we aim to extend MSP so developers can effectively develop intelligent user interfaces as outlined in section 10.2.

Chapter 10

Towards a Framework for Eye Tracking in Intelligent User Interfaces

Intelligent user interfaces (IUIs) (see section 1.1.2) are "human-machine interfaces that aim to improve the efficiency, effectiveness, and naturalness of human-machine interaction by representing, reasoning, and acting on models of the user, domain, task, discourse, and media" (Maybury and Wahlster, 1998, p. 2). Building an IUI means not only making use of technologies and methods from human-computer interaction, multimodal-multisensor interfaces, and artificial intelligence to reach that goal, but also considering "the design of interaction and the design of intelligent algorithms as interrelated parts of a single design problem" (Jameson et al., 2009, p. 11), also known as the *binocular view*. In this section, we relate the contributions of this thesis, i.e., the developed eye tracking methods and technologies, to the main building blocks of an IUI. We briefly explain the role of each building block and discuss how our contributions relate to them. In addition, we outline how MSP could be extended toward a framework for implementing gaze-based intelligent user interfaces, i.e., how it can be extended towards a framework for gaze-based multimodal interaction.

10.1 Relation of the Presented Methods to IUIs

All methods and approaches presented in this thesis can be used in at least one of the main building blocks of an IUI. The main building blocks include *Input Processing & Media Analysis*, *Interaction Management*, *Output Rendering & Media Design*, *User & Context Models*, and *External Services* (see figure 1.1, (Maybury and Wahlster, 1998; Zacharias et al., 2018)). In the following, we discuss how the contributions from this thesis relate to these main building blocks. We developed two methods for active gaze-based interaction and three approaches for passively interpreting the human gaze signal. In this section, we briefly recap our contributions and explain how the resulting methods and approaches can be used in IUIs. This includes two

contributions concerning gaze being used as an active input modality presented in part II and three contributions concerning gaze being used as a passive input modality presented in part III.

10.1.1 Human Gaze as an Active Input Modality

We addressed the problem of gaze estimation errors that can severely hamper the effectiveness and usability of active gaze-based interaction in part II.

- We presented a new class of gaze-based interfaces in chapter 4, namely error-aware gaze-based interfaces that incorporate the inevitable gaze estimation error. We implemented and evaluated methods for modeling the error via machine learning and used it for real-time error-adaptive object selection with a monocular head-mounted eye tracker. The gaze estimation error models relate to the *User & Context Models* block. They provide real-time information about how precise and accurate the gaze signal is. This was used to adjust the gaze signal based on the predicted offset in our *PredictiveShift* method and relates to the *Input Processing & Media Analyis* block. Using a presenter to trigger the selection of a fixated button counts as a modality fusion and can also be assigned to this block. Further, we used the error information to scale buttons in a user interface in real-time to improve the selection rate related to the presentation design part of the *Interaction Management* block and the *Output Rendering & Media Design* block. Accordingly, the software architecture for error-aware gaze-based interfaces presented in section 4.1 can be seen as an implementation of the generic IUI architecture introduced in figure 1.1.
- We implemented and evaluated an approach for calibration-free authentication (PIN entry) based on saccadic eye movements, called *EyeLogin*, in chapter 5 using a remote eye tracker. Using relative eye movements is less dependent on accurate and precise gaze estimation and enables spontaneous interaction with situated displays. In this case, the intelligence lies on the input side and, hence, can be assigned to the *Input Processing & Media Analyis* block. However, it also requires a specific interface design and layout, which counts towards *Output Rendering & Media Design*.

10.1.2 Human Gaze as a Passive Input Modality

We presented several techniques for interpreting eye movements in the context of passive gaze-based interaction in part III. We investigated approaches for three scenarios: visual search, information retrieval by reading texts, and general scene perception.

- We investigated whether eye tracking can infer the target in a visual search process in chapter 6. We developed novel encoding methods for scanpaths based on a sequence of fixated visual stimuli in a scene and investigated their influence on the effectiveness of models for inferring the search target. Our work mainly relates to the *User and Context Models* block because we focused on the modeling aspect, i.e., we aimed to create a model that keeps track of which object a user was looking for. It also connects to the *Input Processing & Media Analyis* block because it analyzes the gaze signal of users for that purpose. However, we did not address the utilization of this model. This requires further

investigations, including detecting when a visual search occurs to activate search target inference. Further, how knowledge about the search target can improve the interaction should be investigated. This would affect the blocks *Interaction Management* and *Output Rendering & Media Design*.

- We investigated whether the perceived relevance of a paragraph can be inferred using a user's eye movements when reading it as input in chapter 7. We conducted a user study (n=24) in which participants read single- and multi-paragraph articles and rated their relevance at the paragraph level concerning a trigger question. Our models were trained using data from this study. This work also focused on the modeling aspect and, hence, mainly relates to the *User and Context Models* block and the *Input Processing & Media Analyis* block because the gaze signal is analyzed. Similar to our approaches for search target inference, we did not focus on the utilization aspect, which leaves investigations on how to use the relevance signal in *Interaction Management* and *Output Rendering & Media Design* as tasks for future work.

- We developed two methods for automatically detecting visual attention to ambient objects using head-mounted eye trackers in combination with pre-trained computer vision models in chapter 8. We investigated their effectiveness for the automatic annotation of mobile eye tracking data from diagnostic user studies, i.e., for automatic mapping of fixations to areas of interest of that study. Further, we developed an interactive machine learning interface called *EyeNotate* that enables semi-automatic annotation of mobile eye tracking data based on few-shot image classification in section 8.2. It addresses the limited flexibility and accuracy when using pre-trained models. First of all, *EyeNotate* is an IUI that supports users in annotating eye tracking data. It interprets user annotations through the web frontend as a supervision signal and uses it to train an image classification model. This model is then applied to provide annotation suggestions to the user. The architecture of the *EyeNotate* system is shown in figure 8.9. *EyeNotate* uses a web-based frontend in the *Input Processing & Media Analyis* block to collect initial training data for an AI-based support system. Model training and inference can be seen as part of the *User and Context Models* or the *External Services* block. Interpreting the human input as a supervisory signal, managing when re-training of models is triggered, and when inference is applied belongs to the *Interaction Management* block. The data and annotation visualization aspects of the frontend count towards the *Output Rendering & Media Design*. The fixation to object mapping models that result from interacting with the *EyeNotate* system can also be used in other contexts, such as attention-aware interfaces. One example is shown in section 8.2.4: we display which objects have been fixated for how long in an augmented reality setting. This example shows how to use the models in real-time for input processing (*Input Processing & Media Analyis*) with immediate feedback (*Output Rendering & Media Design*). However, it is still an open question how the information can facilitate an attention-aware user interface that increases the efficiency, usability, or utility for users.

10.2 Using MSP for Gaze-based Multimodal Interaction

In this section, we describe how MSP can be extended towards a framework for gaze-based multimodal interaction. The contributions of this thesis revolve around the concept of gaze-based multimodal interaction, which has been introduced in section 1.1. It draws inspiration from the design space for gaze-informed multimodal interfaces introduced by Qvarfordt (2017). It uses two dimensions for classifying gaze-based interactive systems: interaction can be rated on a continuum from active to passive and from stationary to mobile. We named these dimensions the *awareness level* and the *mobility level*, respectively (see section 1.1.3). We concretized that the *awareness level* is strongly related to the concept of multimodal-multisensor interfaces because it rates whether gaze is used in an active or passive input mode (see section 1.1.1). The *mobility level* rates the level of freedom during the interaction, which also depends on the type of eye tracker used. It can be either remote or head-mounted (see section 2.2.1). Gaze-based interaction technologies can also be used in the context of intelligent user interfaces, which have been introduced in section 1.1.2. In the previous section, we outlined how the methods and approaches presented in this thesis relate to IUIs. This shows that the generic architecture of IUIs is also a suitable tool for planning and implementing gaze-based multimodal interfaces. However, this architecture is an abstract and theoretic framework that leaves many open questions about how to implement a concrete system. With MSP, we presented a lightweight, flexible, and extensible framework for prototyping multimodal-multisensor interfaces based on real-time sensor input like gaze from eye trackers (see chapter 9). It enables researchers and developers to easily build and adapt stream processing pipelines by connecting existing or custom modules: it allows them to easily integrate multiple sensors or other data streams via source modules, to add stream and event processing capabilities via processor modules, and to connect user interfaces or databases via sink modules in a graph-based processing pipeline. However, it does not fulfill all the requirements for implementing an IUI.

Next, we describe how MSP can be used as is to implement an IUI and identify the current shortcomings. We then outline how these can be overcome to extend MSP to a framework for implementing IUIs and gaze-based multimodal interfaces. Hereby, we refer to the main building blocks of an IUI, including *Input Processing & Media Analysis*, *Interaction Management*, *Output Rendering & Media Design*, *User & Context Models*, and *External Services* (see figure 1.1).

10.2.1 Input Processing & Media Analysis

MSP is optimized for the real-time acquisition and processing of sensor data. MSP offers the tools for integrating any input device or sensor via source modules and options for pre-processing, fusing, and analyzing data streams using custom processor modules in its directed graph-like pipeline structure (see section 9.2). Some standard input devices and sensors are supported by default, including a keyboard, mouse, microphone, and camera. Further devices or sensors can be integrated by inheriting from the base class for source modules `BaseSource`. Data processing functionality and signal interpretation logic must be implemented in custom processor modules, which can be achieved by inheriting from the corresponding base class for processors `BaseProcessor`. These modules can then be composed into a pipeline that manages the life-

cycle of each module during runtime and the order in which data from source modules passes through processing modules. In summary, MSP provides a good basis for implementing the *Input Processing & Media Analysis* block of an IUI. Specific functionalities like connecting new sensors or integrating specific modality fusion algorithms must be implemented. However, this was the intention behind MSP, which is an open-source middleware for prototyping multimodal-multisensor interfaces that allow the research community and practitioners to contribute additional functionality by implementing and sharing new modules. Examples of such new modules are described in the sample application in section 9.2. These include the `PupilLabsCore` source module that connects a head-worn eye tracker, providing real-time gaze and camera streams, and the `CropByPoint` processor module that crops small image patches from the video stream around the most recent point of gaze. Known limitations that affect this building block are the simplistic approach to datastream synchronization and the unavailability of ready-to-use modality fusion algorithms. To ease the implementation of multimodal data processing in IUIs, we aim to add this functionality in future revisions of MSP.

10.2.2 Interaction Management

MSP has no dedicated concept to implement the *Interaction Management* block. Currently, interaction management can be implemented through custom processor or sink modules that retrieve all required input and sensor information to control the visualization and, by that, interaction options. MSP is also decoupled from established UI frameworks such that direct manipulation of traditional UIs would be difficult. Future versions of MSP should offer better integration with established UI frameworks like Qt for Python[1] for desktop-based interfaces or Unity[2] for 3D and mixed reality interfaces.

10.2.3 Output Rendering & Media Design

MSP modules are connected in a directed graph-like structure, which allows developers to add an arbitrary number of sink modules for various presentation purposes. Each sink module is implemented by inheriting the corresponding base class `BaseSink`. Processor modules that are connected upstream of sink modules allow the output to be prepared and scheduled according to the demands of the interactive system. With this basic functionality, MSP enables versatile options for output rendering and media design. However, these would be hardwired once a pipeline is started, limiting flexibility. Currently, MSP does not support hot-swap options, i.e., it does not allow the exchange of modules during execution. Further, there is no global timer that would enable a global synchronization of presentations. However, this would be essential for multimodal output generation. Future work should investigate how global synchronization could be realized, which could also facilitate better data stream synchronization on the input side. An example sink module for the *Output Rendering & Media Design* block can be found in the `VideoLabelsVisualizer` in section 9.2 which renders the video stream of the head-mounted eye tracker and overlays it with a pointer for the gaze signal and the output of an image classification model that predicts what the user is looking at.

[1] `https://doc.qt.io/qtforpython-6/` (accessed on 21 Nov 2024)
[2] `https://unity.com/` (accessed on 21 Nov 2024)

10.2.4 User & Context Models

MSP enables real-time processing of sensor and input device data streams. Individual modules that analyze and interpret data streams can build up local histories about, e.g., identified events in the data, and further processing can depend on this history. Modules can also save and load such histories. However, each module is responsible for its own local history, and there is no good option to share histories with other processors in a pipeline. To realize effective user and context models, a global option for creating, storing, and sharing models is required but is currently not supported by MSP. This is an important requirement for the next version of MSP because all building blocks besides *External Services* might benefit from or even require additional context information. An option to realize global user and context models would be to hand over a reference to a central context component to every module when starting a pipeline. This component would then be responsible for model acquisition, tracking, and utilization. It should offer a corresponding programming interface such that each module in a pipeline can provide data for creating or improving models, and make use of available models. This would, for instance, allow the implicit creation of user preference models from sensor data on the input side and their application for individualized visualizations on the output side.

10.2.5 External Services

External services can be integrated into any kind of module, including source, processor, and sink modules. One example is the `TfHubImageClassifier` processor module that integrates image classification models from the tensorflow-hub platform (see section 9.2). In this case, the connected library downloads and caches a model locally. This comes with the advantage that model inference runs locally, which causes no issues with network delays or data synchronization. On the other hand, this option requires GPU hardware for fast inference on the client side. Cloud-based execution of the inference step would not require local GPU hardware but might come with delays because image data must be passed to the server. In a similar way, almost every web service could be connected to an MSP module. Examples include knowledge graphs like ConceptNet[3]. Meanwhile, there is also a broad range of AI-powered APIs to consider, including, e.g., recently launched services based on large language models like OpenAI's ChatGPT[4] and Google's Gemini[5].

[3] `https://conceptnet.io/` (accessed on 21 Nov 2024)
[4] `https://openai.com/index/introducing-chatgpt-and-whisper-apis/` (accessed on 21 Nov 2024)
[5] `https://ai.google.dev/` (accessed on 21 Nov 2024)

Part V

Conclusion

Chapter 11

Contributions & Outlook

In this thesis, I developed new methods to enable effective and efficient gaze-based multimodal interfaces. In part I chapter 1, I motivated the use of gaze as active and passive input modality for intelligent user interfaces, outlined the related challenges, and presented the considered research questions around the main research question *"How can we enable effective and efficient gaze-based user interfaces and their development?"* In section 1.1, I drew the connection between eye tracking, gaze-based interaction, multimodal-multisensor interfaces (MMI), and intelligent user interfaces (IUI): I introduced *gaze-based multimodal interaction* as a superclass of all presented interaction techniques in this thesis. It is based on the design space for gaze-informed multimodal interfaces by Qvarfordt (2017), which declares two dimensions for classifying gaze-based interfaces. Interaction can be rated on a continuum from active to passive and from stationary to mobile. I named these dimensions the *awareness level* and the *mobility level* (see section 1.1.3). I concretized that the *awareness level* is strongly related to the concept of multimodal-multisensor interfaces because it rates whether gaze is used in an active or passive input mode (see section 1.1.1). The *mobility level* rates the level of freedom during the interaction, which also depends on the type of eye tracker used. It can be either remote or head-mounted (see section 2.2.1). I also drew the connection to IUIs, which, by definition, can incorporate multimodal and gaze-based interaction techniques, to improve human-machine interaction. In chapter 2, I introduced the relevant background, including important details on the human eye and eye movements and their relation to human visual attention processes. Further, I introduced how eye trackers work and basic information on related data analysis techniques, including signal preprocessing and eye movement detection. I also raised awareness for ethics and privacy issues that come with processing data from eye trackers in section 2.3. I reported about our contributions in parts II to IV. I contributed by developing two methods for active gaze-based interaction and three approaches for passively interpreting the human gaze signal. Next, I summarize the main contributions and our efforts towards creating a framework for gaze-based multimodal interaction, i.e., towards enabling effective and efficient gaze-based user interfaces and their development.

11.1 Human Gaze as an Active Input Modality

In part II, I investigated the research question *"How can the negative impact of gaze-estimation errors on gaze-based interaction be reduced when gaze is used as an active input modality?"*. I addressed the problem of gaze estimation errors that can severely hamper the effectiveness and usability of **active gaze-based interaction**. I presented a framework for handling the gaze estimation error in head-mounted eye tracking and developed a calibration-free interaction technique for remote eye trackers that does not rely on accurate gaze estimates:

11.1.1 Error-aware Gaze-based Interaction

We presented a new class of gaze-based interfaces in chapter 4, namely error-aware gaze-based interfaces that incorporate the inevitable gaze estimation error. We implemented and evaluated methods for modeling the error via machine learning and used it for real-time error-adaptive object selection with a monocular head-mounted eye tracker. This approach has the potential to outperform state-of-the-art gaze selection methods in terms of selection performance with competing target sizes. The results of our experiments show the advantages of an error-aware gaze-based interface, relying on gaze shifting and personalized training with a predictive model in terms of selection performance and target sizes.

We plan to further develop the concept of error-aware gaze-based interaction in future work, because our findings are currently limited to a single eye tracking device and an adaptive user interface with a single button. We plan to develop and evaluate new adaptive user interfaces that dynamically distribute interface elements based on their size and on the error to be expected in each area of the interface. For instance, small buttons could be located in low-error regions, while large buttons could be located in more distant regions. This could be particularly effective for mixed reality interfaces that allow more dynamic interfaces that settings with a single display. In addition, we want to investigate to what extent error models can help to evaluate static gaze-based interfaces without involving users. Pre-trained error models could be used to infer potential success rates for object selection and manipulation based on the respective interface layout for all possible user positions. Such simulation-based interface assessments are quick and cheap and would allow front-end developers to improve gaze-based interfaces before going into expensive user testing. This also requires effective and efficient training of error models for new eye trackers. We aim to automate the model training based on data from standard calibration routines and from gaze-based interaction. For instance, selection and manipulation via gaze and detection of smooth pursuits caused by moving on-screen objects could be used for this purpose.

11.1.2 Calibration-free Gaze-based Interaction

We implemented and evaluated *EyeLogin*, an approach for calibration-free authentication (PIN entry) at public displays based on saccadic eye movements from a remote eye tracker in chapter 5. It is less affected by gaze estimation errors because it is based on the interpretation of relative eye movements similar to gesture recognition. In a user study, we could show that our method *EyeLogin* performs significantly faster and significantly more accurate than *CueAuth*, a state-

of-the-art gaze-based authentication method based on smooth pursuit eye movements (Khamis et al., 2018b). *EyeLogin* is the first calibration-free authentication method using gaze that is on par with less secure input modalities such as touch- and gesture-based input in terms of effectiveness and efficiency.

However, *EyeLogin* still has limitations that would affect usability and utility in real applications. To avoid Midas' touch problem, users must press a key to start the interaction. While this is possible in many scenarios, such as withdrawing money from ATMs, some advantages, like better hygiene, vanish. Future research should investigate which additional input modalities can help to overcome this problem. Another limitation is the lack of options to recover from errors such as entering a wrong PIN. We identified gaze-enabled lock patterns as a promising direction. In addition, it would be interesting to investigate whether the gaze signal from a calibration-free authentication procedure, which naturally occurs at the beginning of a user-system interaction, can be used to calibrate an eye tracker for more fine-grained gaze-based interaction, including selection and manipulation of objects. It could also be investigated whether the input of a calibration routine could be used to calibrate an error model for error-aware gaze-based interaction as described in chapter 4.

11.2 Human Gaze as a Passive Input Modality

In part III, I investigated the research question *"How can we build effective machine learning models for interpreting human eye movements in the context of passive gaze-based interaction?"* to address the challenge of building effective and generalizable machine learning models in the context of **passive gaze-based interaction**. I presented several techniques for interpreting the human gaze signal using machine learning for three use cases: inferring the search target of a visual search, estimating the perceived relevance of a text that has been read by a user, and semi-automatic detection of visual attention to areas or objects of interest.

11.2.1 Inferring Visual Search Targets

We investigated whether eye tracking can be used to infer the target in a visual search process in chapter 6. We developed two novel encoding methods for scanpaths based on a sequence of fixated visual stimuli in a scene and investigated their influence on the effectiveness of models for inferring the search target. Our experiments were based on two datasets, including eye movements from visual search tasks captured using remote eye trackers and video stimuli in constrained and natural interaction settings. First, we introduced the *Bag of Deep Visual Words* method for integrating learned features for image classification in the popular *Bag of Words* sequence encoding algorithm for the purpose of search target inference. An evaluation showed that our approach performs better than similar approaches from the literature (Sattar et al., 2015), in particular, when excluding fixations on the visual search target, which would be essential for real applications. The findings of this experiment are limited to the domain of the dataset, which includes collages of book covers similar to online shopping scenarios. In a second experiment, we investigated whether search target inference could be used in less constrained settings. We proposed a segmentation-based approach that infers the image segment that contains the search

target instead of the target itself. We compared the performance of three different methods for visual search encoding using this principle: one based on the previously introduced `BoDVW` encoding as a baseline, the `HoFIS` encoding that reflects segment classes fixated during a visual search, and a concatenation of them. Our evaluation showed that our new encoding achieves a significantly better classification accuracy than the compared methods.

Our research confirmed that the inference of visual search targets using machine learning approaches works better than random guessing in constrained settings. With our segmentation-based approach, we showed that spatial accuracy can be traded against inference performance. However, there are still several limitations that require further development and investigation until search target inference can be applied in real applications. The scenarios considered are still limited: We used pictures with gaze recordings from a lab study rather than videos with gaze recordings from real use cases. Further, our approaches take visual search sequences as input, i.e., they assume that the start time and duration of a visual search process are known, which is not the case in realistic interaction scenarios. Hence, future research should investigate how the onset of visual search processes could be detected. Lastly, the prediction performance is too low to enable effective end-user applications. Future work should also focus on building more effective models. This likely includes collecting more and more diverse data for model training to improve prediction performance and generalizability.

11.2.2 Estimating Document Relevance

We investigated whether the perceived relevance of a paragraph can be inferred using a user's eye movements when reading it as input in chapter 7. We investigated whether we can confirm the findings from Bhattacharya et al. (2020a) that gaze-based features can be used to estimate the perceived relevance of short news articles read by a user. Further, we investigated whether the approach can be applied to multi-paragraph documents that require the user to scroll down to see all text passages. For this, we conducted a user study with n=24 participants who read documents from two corpora, one including short news articles and one including longer Wikipedia articles in English, and rated their relevance at the paragraph level with respect to a previously shown trigger question. We used this data to train and evaluate machine learning models that predict the perceived relevance at the paragraph level using the user's eye movements as input. Our results showed that even though we achieved lower model performance scores than Bhattacharya et al. (2020a), we could replicate their findings under the same experiment conditions: eye movements are an effective source for estimating the perceived relevance of short news articles if we leave out articles that are on the topic but irrelevant. However, we could not show that the approach generalizes to multi-paragraph documents. For both document types, the best model performance was observed when using over-sampling and feature scaling on the training data and a support vector classifier with an RBF kernel for classification, in particular for cases where the users' perceived relevance matched the ground truth from the document corpora. We published our new gazeRE dataset and our code for feature extraction under an open-source license on GitHub[1] to enable other researchers to replicate our approach and to implement and evaluate novel methods in the domain of gaze-based implicit relevance feedback.

[1] `https://github.com/DFKI-Interactive-Machine-Learning/gazeRE-dataset` (accessed on 25 Nov 2024)

Future research should aim to overcome the low estimation performance and the restriction to a single text layout to enable intelligent user interfaces that take into account whether a read text was perceived as relevant or not. Researchers could investigate the effectiveness of scanpath encodings based on graph neural networks (GNNs) (Mohamed Selim et al., 2024) or CNNs (Castner et al., 2020; Bhattacharya et al., 2020b).

11.2.3 Visual Attention Modelling

In this thesis, we investigated approaches for semi-automatic mapping of fixations to AOIs, which can enable efficient analysis and interpretation of humans' complex interaction behavior in chapter 8. In the first step, we developed and evaluated two methods for automatically detecting visual attention to ambient objects based on pre-trained computer vision models in section 8.1. We investigated their effectiveness for the automatic annotation of mobile eye tracking data from diagnostic user studies, i.e., for automatic mapping of fixations to areas of interest of that study. For this, we defined an evaluation framework based on the VISUS dataset by Kurzhals et al. (2014a) and identified four challenges for methods that map gaze to AOIs. Overall, our methods performed well for AOIs that directly matched a class of the pre-trained model. However, we also identified limitations that prevent accelerating and objectifying AOI annotation in eye tracking research: mapping performance decreases when AOIs have no direct match with a class of the pre-trained model, when two instances of the same concept cannot be disambiguated, or when gaze estimation errors occur. Further, we developed the interactive machine learning interface *eyeNotate* that enables semi-automatic annotation of mobile eye tracking data based on few-shot image classification with the goal of addressing the limited flexibility and accuracy when using pre-trained models. The results of a case study confirmed that eyeNotate effectively enables fixation-to-AOI mapping: users liked the basic functionality and interaction design, and the validity and reliability of users' annotations were high. However, we observed that providing AOI label suggestions in the *IML-support* version did not increase the efficiency, likely because of performance issues of the model that led to low trust in our target users. Still, our results suggested that FSL bears great potential for initiating interactive data annotation. We identified constrained model performance as the main hindering factor, especially problems with similar-looking AOIs. We also demonstrated that our technology can be used in real-time interactive systems: We developed a real-time AR system that augments the user's field of view with information on recently attended objects using Microsoft HoloLens 2.

The main limitation of our fixation-to-AOI mapping systems is that the object classification and image classification models at the basis cannot differentiate two instances of the same concept. Hence, future research should focus on the development of more sophisticated approaches that can cope with the dynamic and complex nature of mobile eye tracking data, for instance, using multi-object tracking, 3D reconstruction methods (Kopácsi et al., 2023), and graph neural networks (Le et al., 2025), that allow for better disambiguation of object instances. Further, it is important to investigate and better understand the role of humans in interaction with machine learning algorithms (see section 11.4.4). The interaction design will differ and, hence, should be optimized depending on the individual preferences and the use case at hand. Another focus on future research should lie on applications of fixation-to-AOI mapping algorithms. In the health-

care domain, the recognized sequence of visual attention events can be used to support patients with mental diseases such as dementia by complementing missing parts of their episodic memory (Orlosky et al., 2014; Sonntag, 2015) or to provide AR-based decision support for doctors (Sonntag, 2014). The education domain can also benefit from this technology. For example, it could be used for eye-tracking-based learning support leveraging multimedia effects (Alemdag and Cagiltay, 2018). In the context of experiment-based learning, augmented reality has recently gained increasing interest (Thees et al., 2020; Kapp et al., 2020), but current approaches are limited to stationary settings and are unable to integrate eye tracking data in real-time feedback and support. Another use case in education is Multimodal Learning Analytics (MLA). The goal of MLA is to track the learning process using multiple sensor streams and interaction modalities as input to better understand the complex nature of how students learn (Blikstein, 2013; Oviatt, 2018). The technology can also be used to drive innovation in other domains. Examples include but are not limited to, gaze-based symbol grounding (Mehlmann et al., 2014; Ijuin et al., 2019; Barz et al., 2017) and language learning (Matuszek, 2018) in speech-based human-robot interaction or pervasive attentive user interfaces (Bulling, 2016).

11.3 Gaze-based Multimodal Interaction Framework

In part IV, I described our efforts toward building a framework for gaze-based multimodal interaction, corresponding to the partial research question *"How can we effectively design and develop gaze-based interaction systems?"* The methods presented in parts II and III play a vital role because every contribution adds more knowledge to the field of gaze-based interaction and, hence, gaze-based interfaces closer to the consumer market. With the multisensor-pipeline (MSP), I introduced a practical framework for prototyping related multimodal-multisensor interfaces based on eye trackers (see chapter 9). Further, I discussed how eye tracking can be used in intelligent user interfaces and proposed to extend MSP towards a framework for gaze-based multimodal interaction that also supports features from intelligent user interfaces (see chapter 10).

11.3.1 Multisensor-Pipeline (MSP)

In chapter 9, we presented the multisensor-pipeline (MSP), a lightweight, flexible, and extensible framework for prototyping multimodal-multisensor interfaces based on real-time sensor input like gaze from eye trackers. Its modular and graph-based architecture allows researchers and developers to intuitively build, adapt, and extend stream and event processing pipelines in Python. We demonstrated the effectiveness of MSP by implementing a sample application that crops image patches around a user's fixation point, classifies these patches using a pre-trained deep learning model, and visualizes the results as a real-time video overlay. Our framework can be used to drive future research in multimodal-multisensor user interaction. We plan to enhance the utility of our framework by finalizing the networking and multiprocessing modules, integrating more advanced data synchronization techniques, and adding a dashboard for intuitive debugging and monitoring of processing pipelines. We further discussed the relation to methods and approaches

presented in this thesis. MSP was published under an open-source license on GitHub[2]. We described the framework, showcased how it can be used to implement gaze-based multimodal interaction and discussed the relation to methods and approaches presented in this thesis.

11.3.2 Eye Tracking in Intelligent User Interfaces

In chapter 10, we discussed how the human gaze could be used in intelligent user interfaces by relating the eye tracking methods and approaches presented in this thesis to its main building blocks, including *Input Processing & Media Analysis*, *Interaction Management*, *Output Rendering & Media Design*, *User & Context Models*, and *External Services* (see section 1.1.2). Further, we outlined how MSP can be extended toward a framework for implementing gaze-based intelligent user interfaces, i.e., how it can be extended towards a framework for gaze-based multimodal interaction. We discussed, for each of the building blocks, how MSP can be used in its current form or how MSP would need to be extended to fulfill the corresponding requirements.

11.4 Relation to Research Projects

This thesis is related to past and ongoing research projects of the Interactive Machine Learning department[3] at the German Research Center for Artificial Intelligence (DFKI) supported by national or European funding agencies. These include two completed projects sponsored by the German Federal Ministry of Education and Research, the Bundesministerium für Bildung und Forschung (BMBF): The projects GeAR[4] (Oct 2018 to Sep 2022) and SciBot (Jan 2019 to Dec 2020). Another related project that is currently sponsored by BMBF is No-IDLE (Sonntag et al., 2024) (Apr 2023 to Mar 2026). One ongoing related project, which is sponsored by the European Union (Jan 2023 to Jun 2026), is MASTER[5] (Barz et al., 2024b). In the following, I briefly introduce each project and describe how the contributions of this thesis are related.

11.4.1 GeAR

GeAR was a joint research project on empirical educational research with three partners: Saarland University, Kaiserslautern University of Technology, and DFKI. The aim was to analyze how using AR changes the receptive and productive learning behavior of students. This included identifying requirements for a successful application of AR technology in schools. Studies took place in student laboratories where learners of different ages (primary, lower, and upper secondary education) were performing experiments in the area of electricity. Tablets and smart glasses were applied to realize AR from a technical point of view: the goal was to augment the learners' visual reality with supplementary digital information (e.g., symbols, measuring data) based on AR technology. Multimodal assessments of learning processes and products, as well as multisensory assessments of cognitive load, were used to compare AR learning conditions to

[2] `https://github.com/DFKI-Interactive-Machine-Learning/multisensor-pipeline` (accessed on 22 Nov 2024)

[3] `https://www.dfki.de/iml` (accessed on 16 Dec 2024)

[4] `https://digi-ebf.de/gear` (accessed on 16 Dec 2024)

[5] `https://cordis.europa.eu/project/id/101093079` (accessed on 27 Nov 2024)

control groups. Finally, proven experts were invited to assess the overall results of the research project regarding the opportunities to implement AR in future school learning.

Our role in the project was to develop novel technologies for multimodal learning analytics based on eye tracking and digital pens. In this context, we developed (interactive) fixation-to-AOI mapping algorithms that can help in analyzing and understanding data from studies based on head-mounted eye trackers. These technologies were presented in this thesis in part III, chapter 8 on visual attention modeling. We further investigated how visual search targets could be inferred with the goal of supporting learners in lab-based learning environments when they are searching for learning materials. This work was presented in part III, chapter 6 on inferring visual search targets. We also developed the novel calibration-free active gaze-based interaction method *EyeLogion* based on saccadic eye movements that could reduce the extraneous cognitive load of interactive learning environments. However, we only tested its effectiveness and efficiency for the use case of authentication and did not transfer it to the educational context (see part II, chapter 5). Eventually, our developments in this project laid the ground for the multisensor-pipeline presented in part IV, chapter 9.

11.4.2 SciBot

The SciBot project was funded as a part of the Software Campus program[6], which is a qualification program for doctoral candidates in computer science with the goal of enhancing their leadership skills. In this two-year program, participants lead their own projects in collaboration with a company to get hands-on leadership experience. In addition, participants can participate in advanced training for executives offered by the partner companies. I successfully graduated from the Software Campus program[7] by completing the SciBot project with DATEV eG[8] as an industry partner. The goal of my project was the development of efficient methods for analyzing and learning to analyze user behavior during interactions with an expert system. The considered scenario included one or more expert users that interact with a multimodal-multisensor interface. In particular, the project focused on applications of eye tracking and speech-based input modalities for semantic search applications.

The result of the project was a method for estimating whether a text was perceived as relevant by a reader or not given a search query. I developed this idea together with my collaborators at DATEV eG with the goal of supporting tax accountants searching for information in the LEXinform knowledge base[9]. Our vision was to implicitly collect feedback on the relevance of retrieved documents based on eye tracking to automatically build or adjust search queries. The resulting approach for passive gaze-based interaction was presented as a part of this thesis in part III chapter 7 on estimating document relevance.

[6] `https://softwarecampus.de/en/program/` (accessed on 28 Nov 2024)
[7] `https://softwarecampus.de/certificates/17736/` (accessed 28 Nov 2024)
[8] `https://www.datev.de/` (accessed on 28 Nov 2024)
[9] `https://lexinform.apps.datev.de/` (accessed on 28 Nov 2024)

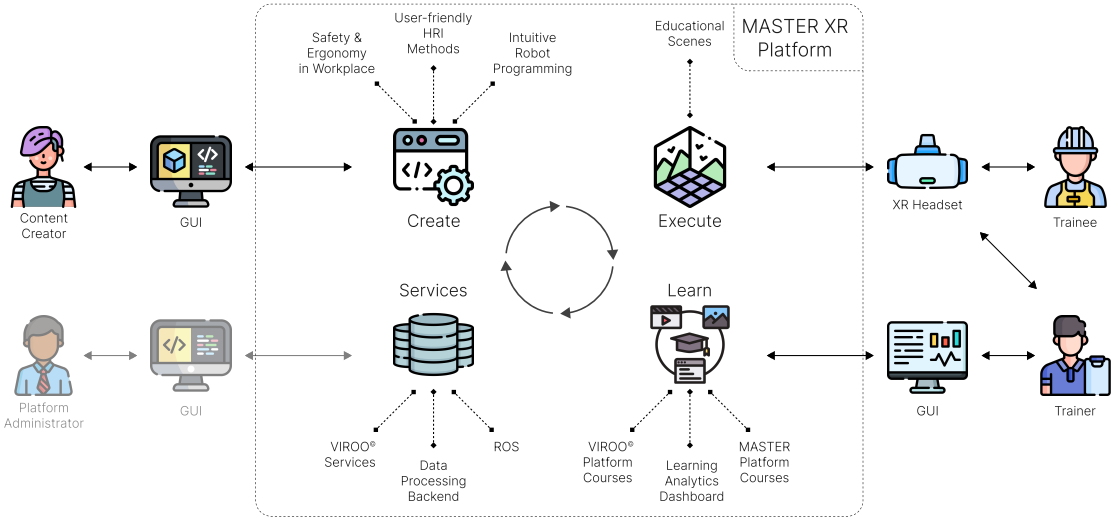


Figure 11.1: Architecture of the MASTER-XR platform.

11.4.3 MASTER

Many industries adopt robots and Extended Reality (XR) technologies to realize Industry 4.0 production models. This requires industrial workers to understand both technologies, particularly in flexible and collaborative manufacturing settings that require frequent programming of robots and safe shared workspaces. For instance, high awareness of robot actions and intentions can help mitigate stress and potential dangers. Consequently, vocational school students and professionals must learn how to program robots and collaborate with them safely. XR technologies can support the learning process (Thees et al., 2022) and may become a key technology of future manufacturing processes. According to data published in 2021, XR will create 1.2-2.4 million new jobs in the EU by 2025, and European XR markets are expected to grow between € 35 billion and € 65 billion by 2025 (Vigkos et al., 2021). Specific education and training programs will be essential for successfully transitioning to XR-based robotic workspaces. The ongoing EU project, MASTER, aims to provide an open XR platform for worker training in robotics and XR-enhanced human-robot collaboration. It is an ongoing HORIZON Innovation Action, which has a focus on technology transfer. We work together with six partners, including companies and other research facilities from Greece, Spain, and Germany, to build a mixed-reality learning platform for advancing the training of workers in robotics. A description of the project and our objectives was published as a chapter in a recently appeared Springer book (Barz et al., 2024b).

The main key exploitable result of our project will be the MASTER-XR platform. It will facilitate the creation of XR-based training scenarios and materials and provide key functionalities according to the three main technological objectives of the project via separate modules: creating safe robotic environments, programming flexible robotic applications, and integrating advanced interaction techniques based on eye tracking. The implementation requires consolidating, maturing, and unifying existing technologies from respective partners. For that purpose,

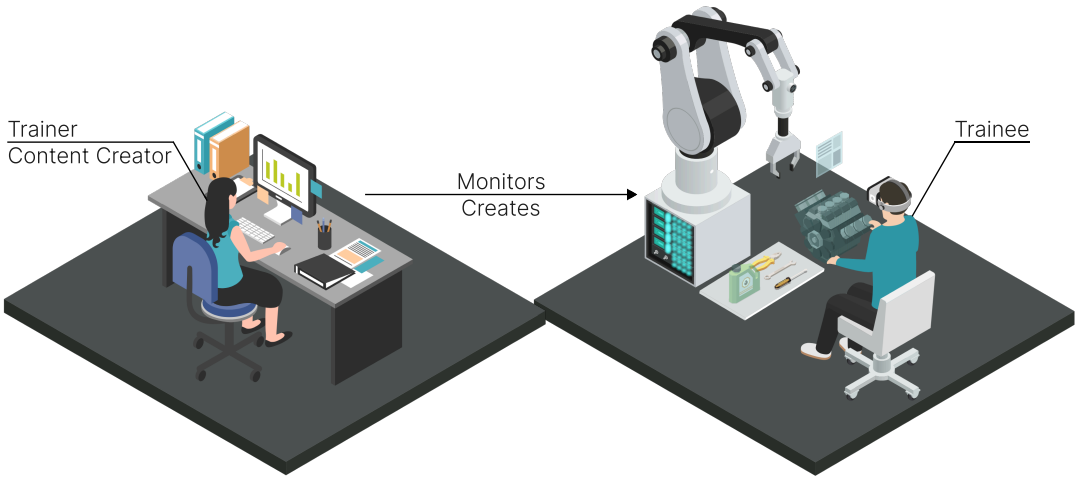


Figure 11.2: User roles of the MASTER-XR platform include content creators, trainers, and trainees.

the VIROO® platform[10] will be extended by our project partner Virtualware 2007 S.A., a virtual reality (VR) platform with a focus on industrial applications. It includes tools and services for easy creation, management, and deployment of immersive content for single-user and remote collaborative multi-user settings. For instance, VIROO® Studio enables low-code VR content creation based on the Unity 3D engine[11], also empowering users without programming experience to create VR content. In addition, it maintains simplicity in certification processes required for industrial applications. VR contents are hosted in a cloud to enable quick and easy deployment and are managed using the Portal service. The XR Platform is composed of four core components: Create, Execute, Services, and Learn (see figure 11.1). Create includes an XR development environment for content creators to integrate the developed modules, like the User-friendly HRI Methods, which will include gaze-based interaction technologies from this thesis. Execute serves the resulting Educational XR Scenes to trainees. Services manages the realized educational scenes and integrates our data processing backend, which is an external processing server that is connected via networking. It serves as an external processing unit for the used machine learning and IML systems. Finally, Learn provides an Educational Dashboard for trainers to configure the IML-based components of XR scenes and for data visualization and analysis purposes. It will enable trainers to easily configure gaze-based interaction and analysis techniques and to gain insights from tracked trainee behavior.

Integrating effective and efficient means to interact with the educational system will be essential for its success. Hence, the fourth technological objective of the MASTER project aims to improve the interaction with XR systems using multimodal interaction (TO4). We integrate gaze-based interaction techniques in virtual and augmented reality settings tailored to educa-

[10] `https://www.virtualwareco.com/viroo/` (accessed on 14 Dec 2024)
[11] `https://unity.com` (accessed 14 Dec 2024)

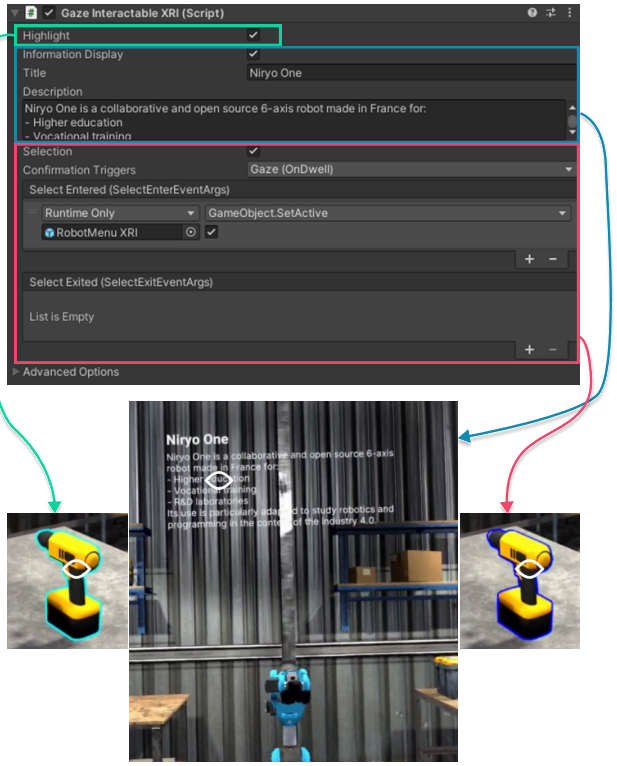


Figure 11.3: Screenshots of the menu of our module for the MASTER-XR platform enabling object highlighting, information displays, and object selection.

tional use cases. We aim to integrate active gaze-based interaction for the manipulation of, e.g., robots, their digital twins, or digital menus. In addition, we plan to integrate methods for real-time analysis of the human gaze behavior for assessing the learning progress of users. This project offers a unique opportunity to transfer the majority of the results of this work to the market. Our module targets three user groups: content creators, trainers, and trainees. Figure 11.2 illustrates the user groups in an example scenario. The content creator would use the XR platform to create XR learning scenarios. They would use our module to set up and configure gaze-based interaction technologies as part of an XR scene. This comprises IML components to, e.g., interactively adapt machine learning models to a new use case. Trainers could use the module for learning analytics purposes, i.e., to analyze learners' visual attention when learning/interacting in XR. Our module will not require any prior knowledge of eye tracking or machine learning. Trainees (or learners) are vocational students or workers in further training in the domain of robotics and make up the third user group. They are the users of XR lessons developed by content creators and provided by their trainers. They wear the XR headset and, if configured, they can interact with the XR scene using their eyes. The trainee's eye movements and visual attention to objects in the scene (also real ones) can be monitored. Visualizations and aggregations of the data can be retrieved by trainers for manual assessment of interaction strategies in robotics training. Our module particularly focuses on gaze-based menus that incorporate findings from parts II and III.

Our module will enable active gaze-based interaction (cf. part II), which is when a user (i.e., a trainee) consciously uses their eyes to interact with a system. Our module will enable content creators to add active gaze-based interaction techniques to their XR scenes, including gaze-based object highlighting, gaze-contingent information displays, gaze-based object selection, and gaze-based radial context menus which combine object highlighting and selection. Figure 11.3 shows how the object highlighting, information displays, and object selection features are realized via a menu integrated into the Unity development platform and how the result looks in VR. This part of the module builds on top of our experience on error-aware interaction presented in chapter 4 and takes advantage of our findings on calibration-free interaction presented in chapter 5. Our module will also enable passive gaze-based interaction (cf. part III), i.e., the trainee's gaze signal is monitored and can be used to implicitly affect the XR scene. The module will enable content creators to integrate visual attention monitoring to virtual and real objects at the same time. A potential extension is to enable content creators to create adaptive user interfaces, i.e., interfaces that automatically adjust themselves based on observed gaze patterns. Trainers will be enabled to explore the collected data to gain insights into the way they interact in XR and learn about robotics. Content creators or robotics trainers might need to adapt computer vision models for visual attention monitoring to their use cases, i.e., to new XR scenes and training content. However, typically these people are domain experts, i.e., experts in XR content creation or robotics, but not in machine learning. Hence, our module will integrate our technology for interactive fixation-to-AOI mapping for learning analytics purposes presented in chapter 8.

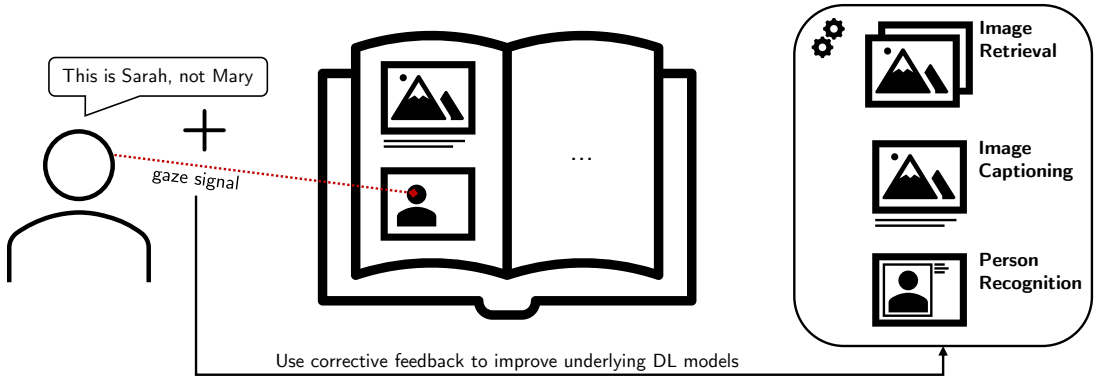
11.4.4 No-IDLE

In recent years, machines have surpassed humans in the performance of specific and narrow tasks, such as some aspects of image recognition or decision-making along clinical pathways in the medical domain (weak AI). Although it is very unlikely that machines will exhibit broadly applicable intelligence comparable to or exceeding that of humans in the next 30 years (strong AI), it is to be expected that machines will reach and exceed human performance on more and more applied tasks. To develop the positive aspects of AI, manage its risks and challenges, and ensure that everyone has the opportunity to help in building an AI-enhanced society and to participate in its benefits, in the No-IDLE project, human intelligence and ML take the center stage: Interactive Machine Learning (IML) is the design and implementation of algorithms and intelligent user interface frameworks that facilitate ML with the help of human interaction (Amershi et al., 2014; Simard et al., 2017; Dudley and Kristensson, 2018; Zacharias et al., 2018). The project's focus is to improve the interaction between humans and machines, by leveraging state-of-the-art HCI approaches, as well as solutions that involve state-of-the-art ML techniques. The No-IDLE project focuses on Interactive Deep Learning (IDL): DL approaches for IML (Sonntag et al., 2024). Computers shall learn from humans by interacting with them in natural language, for example, and by observing them. Basic and fundamental research in this corridor project should also reveal deeper insights into users' behaviors, needs, and goals. ML and, particularly, DL should become accessible to millions of end users, and be functionally more advanced than current recommender systems in online shops that provide suggestions for items that are most pertinent to a particular user. In addition, the project emphasizes the role of multimodal interaction and

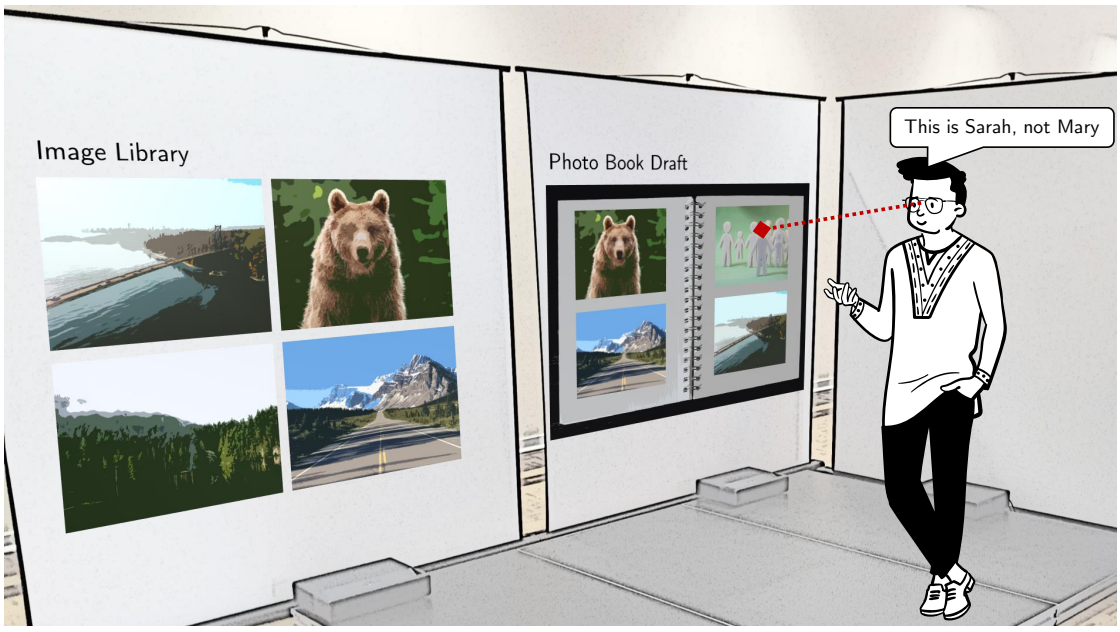
mixed-initiative interaction in this context.

No-IDLE's goals and scientific challenges center around the desire to increase the reach of DL solutions (and ML solutions in general): DL for non-experts in ML and improving DL models when not enough data is available (e.g., due to highly individualized tasks like photo book creation) or data quality is not sufficient. In addition, fully automating tasks in practical applications, such as interactive photo book creation, can be extremely difficult and even undesirable. As a consequence, the goals of No-IDLE are to find a computational and design methodology to gracefully combine automated services with direct user input or manipulation. The scientific goals of the project are investigated in the context of a specific usecase: interactive photo book creation (see figure 11.4). However, the technologies developed shall be beneficial for other domains as well such as healthcare or smart manufacturing. All methods and approaches developed in this thesis in parts II and III will be used and matured in the context of one of the four main scientific goals: Active and passive user input needs to be interpreted carefully to establish an efficient and effective interaction between humans and an AI system. The challenge includes interpreting signals from multiple input modalities (e.g., gaze and spoken instructions). It may be required to interpret the input signals according to a user or context model (e.g., reflecting a user's preferences or the interaction context). In No-IDLE, multimodal interaction techniques for incremental photo book creation will be developed with the goal of improving model training through rich multimodal user feedback and improving the user experience through robust and intuitive interfaces. Also, the limitations of human cognitive abilities shall be respected by using multimodal interaction technologies.

The example usecase, interactive photobook creation, serves as a testbed in the No-IDLE project. The task is highly individual because user preferences and occasions for creating a photobook can vary significantly, and it requires interaction with multimedia content like images, image captions, and visual stories such as event descriptions. To make an example, consider family Smith creating a photobook about their last joint trip to Vancouver, Canada, where they visited Aunt Mary. The tools and technologies to be developed in the No-IDLE project would enable them to create a photobook by sequentially describing the occasion in natural language. The vision is that the family can describe to the system how they perceived their vacation just like they would describe it to another human: "On the first day, we took the bus from the airport to Vancouver". As a response, the system creates a single page with suitable photos, i.e., from getting on the bus at the airport, a photo of the skyline of Vancouver from inside the bus, and one with Aunt Mary, who was waiting for them at the bus stop. Since this is the first time the family is using this tool, the automatic caption generation module is uncertain whether its output is suitable and, hence, actively asks for feedback. Being happy with this partial result, the family continues to describe the events, saying, "The incident with the bears was extremely funny, and the woods were so impressive.". The newly generated pages of the photo include pictures of the bear and the woods from their hiking trip, but none with Aunt Mary, so they complain about this. "Please add a picture with Mary here". As the system does not know yet how Mary looks, it shows extracted faces from the provided photos and asks to select a picture of Mary. Mrs. Smith looks at a picture and says "that's my sister Mary". The system uses the gaze signal to identify the face that was referred to and learns to recognize Mary. Eventually, family Smith reports how their vacation ended: "It was also something how Aunt Mary had to



(a) Gaze-based Multimodal Interaction in No-IDLE



(b) Example Scenario: Interactive Photobook Creation

Figure 11.4: Example of multimodal user input to the photo book application planned in No-IDLE. The user provides corrective feedback in natural language by saying "This is Sarah, not Mary". The system shall use the gaze signal to resolve the face that was referred to and, based on the new information, update the underlying deep learning models.

take us to the airport on short notice because our car broke down and we almost thought we wouldn't make it and how they welcomed us back at the airport after we landed.". One of the images shows Sarah in front of Aunt Mary's car, but the caption states, "This is Aunt Mary after carrying us to the airport.". Mr. Smith corrects the system by saying "this is Sarah, not Mary" (see figure 11.4). The system automatically corrects the caption and corrects the label for the detected face. From now on, the system will be better at differentiating between Sarah (Mrs Smith) and her sister Mary. Alternatively, Mr. Smith could edit the caption to "This is Sarah in front of her car after carrying us to the airport last minute." and the feedback contained in this post-edit would be used to update the image captioning model.

In summary, a goal of No-IDLE is to facilitate model updates in DL-based components like image retrieval, image captioning, and person recognition through gaze-based multimodal interaction of non-ML experts (see figure 11.4a). With the No-IDLE project, the BMBF funds the Interactive Deep Learning Enterprise of our Interactive Machine Learning research group at DFKI as well as the continuation of the research presented in this thesis. It enables us to follow many of the proposed future research directions: The goals of No-IDLE include realizing our future plans to extend interactive fixation-to-AOI mapping (cf. section 8.2) and extending MSP towards a framework for gaze-based multimodal and intelligent user interfaces (cf. section 10.2). The focus of these extensions will be on the integration of deep learning technologies and large language models such as on the interplay between humans and AI-driven systems.

11.5 Dissemination & Impact

The results of this work were mainly disseminated through peer-reviewed publications in relevant conference proceedings or journals also including demo and video formats. This led to a total of 11 peer-reviewed publications (Barz et al., 2016a, 2018, 2020b, 2021a,b, 2022, 2023; Barz and Sonntag, 2021; Stauden et al., 2018; Bhatti et al., 2021; Valdunciel et al., 2022), for which three awards were presented. This includes two full papers, three short papers, and four demo, video, or late-breaking results papers at conferences. Hereby, the ACM Symposium on Eye Tracking Research & Applications (ETRA) was the most frequent outlet, with a total of five publications (two full papers, two short papers, and one demonstration paper). The ETRA full paper on error-aware gaze-based interaction, presented in chapter 4, retrieved the Best Paper Award in 2018 (Barz et al., 2018). The ETRA short paper on calibration-free gaze-based interaction, presented in chapter 5, retrieved the Honorable Mention Award in 2021 (Bhatti et al., 2021). For our demo on *eyeNotate*, we received the Best Demo Honorable Mention Award at the ACM Conference on Intelligent User Interfaces (IUI) in 2023 (Barz et al., 2023). Another two peer-reviewed full papers have been published in journals. This includes one publication on automatic visual attention detection, presented in section 8.1, in the Sensors journal (Barz and Sonntag, 2021), and one on implicit relevance detection, presented in chapter 7, in the Frontiers in Computer Science journal (Barz et al., 2022). Two further publications on semi-automatic fixation-to-AOI mapping, related to section 8.2, are currently under review in the ACM Transactions on Interactive Intelligent Systems journal (Barz et al., 2025) and the Nature Scientific Reports journal (Le et al., 2025).

Another activity to disseminate and promote our research was the organization of the HumanEYEze workshop on Eye Tracking for Multimodal Human-Centric Computing (Barz et al., 2024a) together with renowned experts in the field, including Roman Bednarik, Andreas Bulling, Cristina Conati, and Daniel Sonntag. The workshop was a part of the ACM International Conference on Multimodal Interaction 2024 in San José, Costa Rica. It brought together researchers and practitioners from various fields, including eye tracking, artificial intelligence, robotics, affective computing, multimodal interaction, and human-computer interaction, to identify and discuss promising applications and key challenges related to gaze-based multimodal interaction.

The impact of this thesis is, up to now, mainly scientific. It can be measured by the number of citations related to this thesis: Based on Google Scholar, related publications had been cited 181 times[12]. It is also evident that the research results presented in this thesis have led to successful project acquisition. All four projects introduced in section 11.4 build on top of parts of this thesis. Until now, there has been little economic impact, such as through contributions to the open-source eye tracking platform by Pupil Labs[13]. However, the ongoing EU project MASTER offers a good starting point for future economic impact. As a HORIZON innovation action, it focuses on technology transfer from research to the market. In this case, the research outcomes include the active and passive gaze-based interaction technologies from this thesis.

[12] `https://scholar.google.de/citations?hl=en&user=GDTNjuMAAAAJ` (accessed on 12 Dec 2024)
[13] `https://pupil-labs.com/` (accessed on 29 Nov 2024)

Bibliography

Yomna Abdelrahman, Mohamed Khamis, Stefan Schneegass, and Florian Alt. 2017. Stay Cool! Understanding Thermal Attacks on Mobile-Based User Authentication. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 3751–3763. https://doi.org/10.1145/3025453.3025461

Yasmeen Abdrabou, Mohamed Khamis, Rana Mohamed Eisa, Sherif Ismail, and Amrl Elmougy. 2019. Just Gaze and Wave: Exploring the Use of Gaze and Gestures for Shoulder-Surfing Resilient Authentication. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. Association for Computing Machinery, New York, NY, USA, 10. https://doi.org/10.1145/3314111.3319837

Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving Web Search Ranking by Incorporating User Behavior Information. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '06)*. Association for Computing Machinery, New York, NY, USA, 19–26. https://doi.org/10.1145/1148170.1148177

Sunggeun Ahn, Stephanie Santosa, Mark Parent, Daniel Wigdor, Tovi Grossman, and Marcello Giordano. 2021. StickyPie: A Gaze-Based, Scale-Invariant Marking Menu Optimized for AR/VR. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 16. https://doi.org/10.1145/3411764.3445297

Antti Ajanki, David R. Hardoon, Samuel Kaski, Kai Puolamäki, and John Shawe-Taylor. 2009. Can eyes reveal interest? Implicit queries from gaze patterns. *User Modeling and User-Adapted Interaction* 19, 4 (Oct. 2009), 307–339. https://doi.org/10.1007/s11257-009-9066-4

Deepak Akkil and Poika Isokoski. 2016. Gaze Augmentation in Egocentric Video Improves Awareness of Intention. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1573–1584. https://doi.org/10.1145/2858036.2858127

Deepak Akkil, Poika Isokoski, Jari Kangas, Jussi Rantala, and Roope Raisamo. 2014. TraQuMe: A Tool for Measuring the Gaze Tracking Quality. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. Association for Computing Machinery, New York, NY, USA, 327–330. https://doi.org/10.1145/2578153.2578192

Stephen Akuma, Rahat Iqbal, Chrisina Jayne, and Faiyaz Doctor. 2016. Comparative analysis of relevance feedback methods based on two user studies. *Computers in Human Behavior* 60 (2016), 138–146. https://doi.org/10.1016/j.chb.2016.02.064

Ecenaz Alemdag and Kursat Cagiltay. 2018. A systematic review of eye tracking research on multimedia learning. *Computers & Education* 125 (Oct. 2018), 413–428. https://doi.org/10.1016/j.compedu.2018.06.023

Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. 2014. Power to the People: The Role of Humans in Interactive Machine Learning. *AI Magazine* 35, 4 (Dec. 2014), 105–120. https://doi.org/10.1609/aimag.v35i4.2513

Elisabeth André and Joyce Chai. 2013. Introduction to the Special Section on Eye Gaze and Conversation. *ACM Transactions on Interactive Intelligent Systems* 3, 2 (Aug. 2013), 2. https://doi.org/10.1145/2499474.2499479

David A. Atchison and George Smith. 2023. The Human Eye: An Overview. In *Optics of the Human Eye* (2 ed.). CRC Press, Boca Raton, FL, USA, 12. https://doi.org/10.1201/9781003128601

Adam J. Aviv, Katherine Gibson, Evan Mossop, Matt Blaze, and Jonathan M. Smith. 2010. Smudge Attacks on Smartphone Touch Screens. In *Proceedings of the 4th USENIX Conference on Offensive Technologies (WOOT'10)*. USENIX Association, USA, 1–7.

Thomas Bader and Jürgen Beyerer. 2013. Natural Gaze Behavior as Input Modality for Human-Computer Interaction. In *Eye Gaze in Intelligent User Interfaces: Gaze-based Analyses, Models and Applications*, Yukiko I. Nakano, Cristina Conati, and Thomas Bader (Eds.). Springer, London, UK, 161–183. https://doi.org/10.1007/978-1-4471-4784-8_9

Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. 2017. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, 12 (2017), 2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615

Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *Journal of Usability Studies* 4, 3 (May 2009), 114–123. https://uxpajournal.org/determining-what-individual-sus-scores-mean-adding-an-adjective-rating-scale/

Stanislavs Bardins, Tony Poitschke, and Stefan Kohlbecher. 2008. Gaze-Based Interaction in Various Environments. In *Proceedings of the 1st ACM Workshop on Vision Networks for Behavior Analysis (VNBA '08)*. Association for Computing Machinery, New York, NY, USA, 47–54. https://doi.org/10.1145/1461893.1461903

Michael Barz, Kristin Altmeyer, Sarah Malone, Luisa Lauer, and Daniel Sonntag. 2020a. Digital Pen Features Predict Task Difficulty and User Performance of Cognitive Tests. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2020, Genoa, Italy, July 12-18, 2020*, Tsvi Kuflik, Ilaria Torre, Robin Burke, and Cristina Gena (Eds.). ACM, New York, NY, USA, 23–32. https://doi.org/10.1145/3340631.3394839

Michael Barz, Roman Bednarik, Andreas Bulling, Cristina Conati, and Daniel Sonntag. 2024a. HumanEYEze 2024: Workshop on Eye Tracking for Multimodal Human-Centric Computing. In *Proceedings of the 26th International Conference on Multimodal Interaction (ICMI '24)*. Association for Computing Machinery, New York, NY, USA, 696–697. https://doi.org/10.1145/3678957.3688384

Michael Barz, Omair Shahzad Bhatti, Hasan Md Tusfiqur Alam, Duy Minh Ho Nguyen, Kristin Altmeyer, Sarah Malone, and Daniel Sonntag. 2025. eyeNotate: Interactive Annotation of Mobile Eye Tracking Data based on Few-Shot Image Classification. *JEMR* (2025). (under review).

Michael Barz, Omair Shahzad Bhatti, Hasan Md Tusfiqur Alam, Duy Minh Ho Nguyen, and Daniel Sonntag. 2023. Interactive Fixation-to-AOI Mapping for Mobile Eye Tracking Data Based on Few-Shot Image Classification. In *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces (IUI '23 Companion)*. Association for Computing Machinery, New York, NY, USA, 175–178. https://doi.org/10.1145/3581754.3584179

Michael Barz, Omair Shahzad Bhatti, Bengt Lüers, Alexander Prange, and Daniel Sonntag. 2021a. Multisensor-Pipeline: A Lightweight, Flexible, and Extensible Framework for Building Multimodal-Multisensor Interfaces. In *Companion Publication of the 2021 International Conference on Multimodal Interaction (ICMI '21 Companion)*. Association for Computing Machinery, New York, NY, USA, 13–18. https://doi.org/10.1145/3461615.3485432

Michael Barz, Omair Shahzad Bhatti, and Daniel Sonntag. 2022. Implicit Estimation of Paragraph Relevance From Eye Movements. *Frontiers in Computer Science* 3 (2022), 13. https://doi.org/10.3389/fcomp.2021.808507

Michael Barz, Andreas Bulling, and Florian Daiber. 2015. *Computational Modelling and Prediction of Gaze Estimation Error for Head-mounted Eye Trackers.* Technical Report. DFKI, DFKI Research Reports (RR), Vol. 1. 10 pages. https://www.dfki.de/web/forschung/projekte-publikationen/publikation/7619

Michael Barz, Florian Daiber, and Andreas Bulling. 2016a. Prediction of gaze estimation error for error-aware gaze-based interfaces. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications, ETRA 2016, Charleston, SC, USA, March 14-17, 2016*, Pernilla Qvarfordt and Dan Witzner Hansen (Eds.). ACM, New York, NY, USA, 275–278. https://doi.org/10.1145/2857491.2857493

Michael Barz, Florian Daiber, Daniel Sonntag, and Andreas Bulling. 2018. Error-aware gaze-based interfaces for robust mobile gaze interaction. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications, ETRA 2018, Warsaw, Poland, June 14-17, 2018*, Bonita Sharif and Krzysztof Krejtz (Eds.). ACM, New York, NY, USA, 10. https://doi.org/10.1145/3204493.3204536

Michael Barz, Sebastian Kapp, Jochen Kuhn, and Daniel Sonntag. 2021b. Automatic Recognition and Augmentation of Attended Objects in Real-time using Eye Tracking and a Head-mounted Display. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '21 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 4. https://doi.org/10.1145/3450341.3458766

Michael Barz, Panagiotis Karagiannis, Johan Kildal, Andoni Rivera Pinto, Judit Ruiz de Munain, Jesús Rosel, Maria Madarieta, Konstantina Salagianni, Panagiotis Aivaliotis, Sotiris Makris, and Daniel Sonntag. 2024b. MASTER-XR: Mixed Reality Ecosystem for Teaching Robotics in Manufacturing. In *Integrated Systems: Data Driven Engineering*, Mohammad-Reza Alam and Madjid Fathi (Eds.). Springer Nature Switzerland, Cham, 167–182. https://doi.org/10.1007/978-3-031-53652-6_10

Michael Barz, Mohammad Mehdi Moniri, Markus Weber, and Daniel Sonntag. 2016b. Multimodal multisensor activity annotation tool. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp Adjunct 2016, Heidelberg, Germany, September 12-16, 2016*, Paul Lukowicz, Antonio Krüger, Andreas Bulling, Youn-Kyung Lim, and Shwetak N. Patel (Eds.). ACM, New York, NY, USA, 17–20. https://doi.org/10.1145/2968219.2971459

Michael Barz, Peter Poller, and Daniel Sonntag. 2017. Evaluating Remote and Head-worn Eye Trackers in Multi-modal Speech-based HRI. In *Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, HRI 2017, Vienna, Austria, March 6-9, 2017*, Bilge Mutlu, Manfred Tscheligi, Astrid Weiss, and James E. Young (Eds.). ACM, New York, NY, USA, 79–80. https://doi.org/10.1145/3029798.3038367

Michael Barz and Daniel Sonntag. 2016. Gaze-guided object classification using deep neural networks for attention-based computing. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp Adjunct 2016, Heidelberg, Germany, September 12-16, 2016*, Paul Lukowicz, Antonio Krüger, Andreas Bulling, Youn-Kyung Lim, and Shwetak N. Patel (Eds.). ACM, New York, NY, USA, 253–256. https://doi.org/10.1145/2968219.2971389

Michael Barz and Daniel Sonntag. 2021. Automatic Visual Attention Detection for Mobile Eye Tracking Using Pre-Trained Computer Vision Models and Human Gaze. *Sensors* 21, 12 (June 2021), 21. https://doi.org/10.3390/s21124143

Michael Barz, Sven Stauden, and Daniel Sonntag. 2020b. Visual Search Target Inference in Natural Interaction Settings with Machine Learning. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*, Andreas Bulling, Anke Huckauf, Eakta Jain, Ralph Radach, and Daniel Weiskopf (Eds.). Association for Computing Machinery, New York, NY, USA, 8. https://doi.org/10.1145/3379155.3391314

Richard Bates, M. Donegan, H. O. Istance, J. P. Hansen, and K.-J. Räihä. 2007. Introducing COGAIN: communication by gaze interaction. *Universal Access in the Information Society* 6, 2 (Sept. 2007), 159–166. https://doi.org/10.1007/s10209-007-0077-9

M. Batliner, S. Hess, C. Ehrlich-Adám, Q. Lohmeyer, and M. Meboldt. 2020. Automated areas of interest analysis for usability studies of tangible screen-based user interfaces using mobile eye tracking. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 34, 4 (Sept. 2020), 505–514. https://doi.org/10.1017/S0890060420000372

Tobias Baur, Alexander Heimerl, Florian Lingenfelser, Johannes Wagner, Michel F. Valstar, Björn Schuller, and Elisabeth André. 2020. eXplainable Cooperative Machine Learning with NOVA. *KI - Künstliche Intelligenz* 34, 2 (June 2020), 143–164. https://doi.org/10.1007/s13218-020-00632-3

Tobias Baur, Gregor Mehlmann, Ionut Damian, Florian Lingenfelser, Johannes Wagner, Birgit Lugrin, Elisabeth André, and Patrick Gebhard. 2015. Context-Aware Automated Analysis and Annotation of Social Human–Agent Interactions. *ACM Transactions on Interactive Intelligent Systems* 5, 2 (June 2015), 1 – 33. https://doi.org/10.1145/2764921

Gedas Bertasius, Hyun Soo Park, Stella X. Yu, and Jianbo Shi. 2016. First Person Action-Object Detection with EgoNet. *CoRR* abs/1603.04908 (2016), 10. http://arxiv.org/abs/1603.04908

Darrell S. Best and Andrew T. Duchowski. 2016. A Rotary Dial for Gaze-Based PIN Entry. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*. Association for Computing Machinery, New York, NY, USA, 69–76. https://doi.org/10.1145/2857491.2857527

David Beymer and Daniel M. Russell. 2005. WebGazeAnalyzer: A System for Capturing and Analyzing Web Reading Behavior Using Eye Gaze. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*. Association for Computing Machinery, New York, NY, USA, 1913–1916. https://doi.org/10.1145/1056808.1057055

Nilavra Bhattacharya and Jacek Gwizdka. 2018. Relating Eye-Tracking Measures with Changes in Knowledge on Search Tasks. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. Association for Computing Machinery, New York, NY, USA, New York, NY, USA. https://doi.org/10.1145/3204493.3204579

Nilavra Bhattacharya, Somnath Rakshit, and Jacek Gwizdka. 2020a. Towards Real-Time Webpage Relevance Prediction Using Convex Hull Based Eye-Tracking Features. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3379157.3391302

Nilavra Bhattacharya, Somnath Rakshit, Jacek Gwizdka, and Paul Kogut. 2020b. Relevance Prediction from Eye-Movements Using Semi-Interpretable Convolutional Neural Networks. In *Proceedings of the 2020 Conference on Human Information Interaction and Retrieval (CHIIR '20)*. Association for Computing Machinery, New York, NY, USA, 223–233. https://doi.org/10.1145/3343413.3377960

Omair Shahzad Bhatti, Michael Barz, and Daniel Sonntag. 2021. EyeLogin - Calibration-free Authentication Method for Public Displays Using Eye Gaze. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '21 Short Papers)*. Association for Computing Machinery, New York, NY, USA, 7. https://doi.org/10.1145/3448018.3458001

T. Blascheck, K. Kurzhals, M. Raschke, M. Burch, D. Weiskopf, and T. Ertl. 2017. Visualization of Eye Tracking Data: A Taxonomy and Survey. *Computer Graphics Forum* 36, 8 (Dec. 2017), 260–284. https://doi.org/10.1111/cgf.13079

Pieter Blignaut and Tanya Beelders. 2012. TrackStick: A Data Quality Measuring Tool for Tobii Eye Trackers. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 293–296. https://doi.org/10.1145/2168556.2168619

Pieter Blignaut, Kenneth Holmqvist, Marcus Nyström, and Richard Dewhurst. 2014. Improving the Accuracy of Video-Based Eye Tracking in Real Time through Post-Calibration Regression. In *Current Trends in Eye Tracking Research*, Mike Horsley, Matt Eliot, Bruce Allen Knight, and Ronan Reilly (Eds.). Springer, Cham, 77–100. https://doi.org/10.1007/978-3-319-02868-2_5

Pieter Blignaut and Daniël Wium. 2013. The Effect of Mapping Function on the Accuracy of a Video-Based Eye Tracker. In *Proceedings of the 2013 Conference on Eye Tracking South Africa (ETSA '13)*. Association for Computing Machinery, New York, NY, USA, 39–46. https://doi.org/10.1145/2509315.2509321

Paulo Blikstein. 2013. Multimodal Learning Analytics. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge (LAK '13)*. Association for Computing Machinery, New York, NY, USA, 102–106. https://doi.org/10.1145/2460296.2460316

Dan Bohus, Sean Andrist, Ashley Feniello, Nick Saw, Mihai Jalobeanu, Patrick Sweeney, Anne Loomis Thompson, and Eric Horvitz. 2021. Platform for Situated Intelligence. *CoRR* abs/2103.15975 (2021), 29. https://arxiv.org/abs/2103.15975

Ali Borji and Laurent Itti. 2013. State-of-the-Art in Visual Attention Modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 1 (2013), 185–207. https://doi.org/10.1109/TPAMI.2012.89

Ali Borji, Andreas Lennartz, and Marc Pomplun. 2015. What do eyes reveal about the mind?: Algorithmic inference of search targets from fixations. *Neurocomputing* 149 (2015), 788–799. https://doi.org/10.1016/j.neucom.2014.07.055

Virginia Braun and Victoria Clarke. 2012. Thematic analysis. In *APA handbook of research methods in psychology, Vol 2: Research designs: Quantitative, qualitative, neuropsychological, and biological.* American Psychological Association, Washington, DC, US, 57–71. https://doi.org/10.1037/13620-004

Jurek Breuninger, Christian Lange, and Klaus Bengler. 2011. Implementing Gaze Control for Peripheral Devices. In *Proceedings of the 1st International Workshop on Pervasive Eye Tracking & Mobile Eye-Based Interaction (PETMEI '11)*. Association for Computing Machinery, New York, NY, USA, 3–8. https://doi.org/10.1145/2029956.2029960

John Brooke. 1996. SUS: A 'Quick and Dirty' Usability Scale. In *Usability Evaluation In Industry* (1st edition ed.). CRC Press, London, United Kingdom, 6. https://doi.org/10.1201/9781498710411

Frederik Brudy, David Ledo, Saul Greenberg, and Andreas Butz. 2014. Is Anyone Looking? Mitigating Shoulder Surfing on Public Displays through Awareness and Protection. In *Proceedings of The International Symposium on Pervasive Displays (PerDis '14)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/2611009.2611028

Geert Brône, Bert Oben, and Toon Goedemé. 2011. Towards a More Effective Method for Analyzing Mobile Eye-Tracking Data: Integrating Gaze Data with Object Recognition Algorithms. In *Proceedings of the 1st International Workshop on Pervasive Eye Tracking & Mobile Eye-Based Interaction (PETMEI '11)*. Association for Computing Machinery, New York, NY, USA, 53–56. https://doi.org/10.1145/2029956.2029971

George Buchanan, Dana McKay, Eduardo Velloso, Alistair Moffat, Andrew Turpin, and Falk Scholer. 2017. Only Forward? Toward Understanding Human Visual Behaviour When Examining Search Results. In *Proceedings of the 29th Australian Conference on Computer-Human Interaction (OZCHI '17)*. Association for Computing Machinery, New York, NY, USA, 497–502. https://doi.org/10.1145/3152771.3156165

Andreas Bulling. 2010. *Eye movement analysis for context inference and cognitive-awareness: wearable sensing and activity recognition using electrooculography.* Dissertation. ETH Zurich, Zurich, Switzerland. https://doi.org/10.3929/ethz-a-006208589 ISBN: 978-3-909386-34-5.

Andreas Bulling. 2016. Pervasive Attentive User Interfaces. *Computer* 49, 1 (Jan. 2016), 94–98. https://doi.org/10.1109/MC.2016.32

Andreas Bulling and Daniel Roggen. 2011. Recognition of Visual Memory Recall Processes Using Eye Movement Analysis. In *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp '11)*. Association for Computing Machinery, New York, NY, USA, 455–464. https://doi.org/10.1145/2030112.2030172

Andreas Bulling, Daniel Roggen, and Gerhard Tröster. 2008. EyeMote – Towards Context-Aware Gaming Using Eye Movements Recorded from Wearable Electrooculography. In *Fun and Games*, Panos Markopoulos, Boris de Ruyter, Wijnand IJsselsteijn, and Duncan Rowland (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 33–45. https://doi.org/10.3929/ethz-a-005705469

Andreas Bulling, Christian Weichel, and Hans Gellersen. 2013. EyeContext: Recognition of High-level Contextual Cues from Human Visual Behaviour. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 305–308. https://doi.org/10.1145/2470654.2470697

Georg Buscher. 2010. *Attention-based information retrieval - using eye tracking to capture attention evidence and personalize search.* Dissertation. Technische Universität Kaiserslautern. http://www.dr.hut-verlag.de/978-3-86853-542-6.html

Georg Buscher, Andreas Dengel, Ralf Biedert, and Ludger V. Elst. 2012. Attentive Documents: Eye Tracking as Implicit Feedback for Information Retrieval and Beyond. *ACM Transactions on Interactive Intelligent Systems* 1, 2 (Jan. 2012), 1–30. https://doi.org/10.1145/2070719.2070722

Georg Buscher, Andreas Dengel, and Ludger van Elst. 2008a. Eye Movements as Implicit Relevance Feedback. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2991–2996. https://doi.org/10.1145/1358628.1358796

Georg Buscher, Andreas Dengel, and Ludger van Elst. 2008b. Query Expansion Using Gaze-Based Feedback on the Subdocument Level. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '08)*. Association for Computing Machinery, New York, NY, USA, 387–394. https://doi.org/10.1145/1390334.1390401

Georg Buscher, Ludger van Elst, and Andreas Dengel. 2009. Segment-Level Display Time as Implicit Feedback: A Comparison to Eye Tracking. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '09)*. Association for Computing Machinery, New York, NY, USA, 67–74. https://doi.org/10.1145/1571941.1571955

Mihai Bâce, Vincent Becker, Chenyang Wang, and Andreas Bulling. 2020. Combining Gaze Estimation and Optical Flow for Pursuits Interaction. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 10. https://doi.org/10.1145/3379155.3391315

Mihai Bâce, Teemu Leppänen, David Gil de Gomez, and Argenis Ramirez Gomez. 2016. UbiGaze: Ubiquitous Augmented Reality Messaging Using Gaze Gestures. In *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications (SA '16)*. Association for Computing Machinery, New York, NY, USA, 5. https://doi.org/10.1145/2999508.2999530

Timothy Callemein, Kristof Van Beeck, Geert Brône, and Toon Goedemé. 2019. Automated Analysis of Eye-Tracker-Based Human-Human Interaction Studies. In *Information Science and Applications 2018 (Lecture Notes in Electrical Engineering)*, Kuinam J. Kim and Nakhoon Baek (Eds.). Springer, Singapore, 499–509. https://doi.org/10.1007/978-981-13-1056-0_50

Qiong Cao, Li Shen, Weidi Xie, Omkar M. Parkhi, and Andrew Zisserman. 2018. VGGFace2: A Dataset for Recognising Faces across Pose and Age. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*. IEEE, 67–74. https://doi.org/10.1109/FG.2018.00020

Claudio Carpineto and Giovanni Romano. 2012. A Survey of Automatic Query Expansion in Information Retrieval. *Comput. Surveys* 44, 1 (Jan. 2012), 50. https://doi.org/10.1145/2071389.2071390

Jon W. Carr, Valentina N. Pescuma, Michele Furlan, Maria Ktori, and Davide Crepaldi. 2021. Algorithms for the automated correction of vertical drift in eye-tracking data. *Behavior Research Methods* 54 (June 2021), 287–310. https://doi.org/10.3758/s13428-021-01554-0

Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1 € Filter: A Simple Speed-Based Low-Pass Filter for Noisy Input in Interactive Systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2527–2530. https://doi.org/10.1145/2207676.2208639

Nora Castner, Thomas C Kuebler, Katharina Scheiter, Juliane Richter, Therese Eder, Fabian Huettig, Constanze Keutel, and Enkelejda Kasneci. 2020. Deep Semantic Gaze Embedding and Scanpath Comparison for Expertise Classification during OPT Viewing. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3379155.3391320

Juan J. Cerrolaza, Arantxa Villanueva, Maria Villanueva, and Rafael Cabeza. 2012. Error Characterization and Compensation in Eye Tracking Systems. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 205–208. https://doi.org/10.1145/2168556.2168595

Yuhu Chang, Yingying Zhao, Mingzhi Dong, Yujiang Wang, Yutian Lu, Qin Lv, Robert P. Dick, Tun Lu, Ning Gu, and Li Shang. 2021. MemX: An Attention-Aware Smart Eyewear System for Personalized Moment Auto-Capture. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 2 (June 2021), 23. https://doi.org/10.1145/3463509

Ishan Chatterjee, Robert Xiao, and Chris Harrison. 2015. Gaze+Gesture: Expressive, Precise and Targeted Free-Space Interactions. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction (ICMI '15)*. Association for Computing Machinery, New York, NY, USA, 131–138. https://doi.org/10.1145/2818346.2820752

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research* 16 (June 2002), 321–357. https://doi.org/10.1613/jair.953

Yongqiang Chen, Peng Zhang, Dawei Song, and Benyou Wang. 2015. A Real-Time Eye Tracking Based Query Expansion Approach via Latent Topic Modeling. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM '15)*. Association for Computing Machinery, New York, NY, USA, 1719–1722. https://doi.org/10.1145/2806416.2806602

Myungguen Choi, Daisuke Sakamoto, and Tetsuo Ono. 2020. Bubble Gaze Cursor + Bubble Gaze Lens: Applying Area Cursor Technique to Eye-Gaze Interface. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 10. https://doi.org/10.1145/3379155.3391322

Myungguen Choi, Daisuke Sakamoto, and Tetsuo Ono. 2022. Kuiper Belt: Utilizing the "Out-of-Natural Angle" Region in the Eye-Gaze Interaction for Virtual Reality. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 17. https://doi.org/10.1145/3491102.3517725

Wonil Choi, Matthew W. Lowder, Fernanda Ferreira, and John M. Henderson. 2015. Individual differences in the perceptual span during reading: Evidence from the moving window technique. *Attention, Perception, & Psychophysics* 77, 7 (Oct. 2015), 2463–2475. https://doi.org/10.3758/s13414-015-0942-1

Eunji Chong, Katha Chanda, Zhefan Ye, Audrey Southerland, Nataniel Ruiz, Rebecca M. Jones, Agata Rozga, and James M. Rehg. 2017. Detecting Gaze Towards Eyes in Natural Social Interactions and Its Use in Child Assessment. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (Sept. 2017), 20. https://doi.org/10.1145/3131902

Michael J. Cole, Jacek Gwizdka, Chang Liu, Nicholas J. Belkin, and Xiangmin Zhang. 2013. Inferring user knowledge level from eye movement patterns. *Information Processing & Management* 49, 5 (2013), 1075–1091. https://doi.org/10.1016/j.ipm.2012.08.004

Chris Creed, Maite Frutos-Pascual, and Ian Williams. 2020. Multimodal Gaze Interaction for Creative Design. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 13. https://doi.org/10.1145/3313831.3376196

Han Cui and Naim Dahnoun. 2021. High Precision Human Detection and Tracking Using Millimeter-Wave Radars. *IEEE Aerospace and Electronic Systems Magazine* 36, 1 (2021), 22–32. https://doi.org/10.1109/MAES.2020.3021322

Dietlind Helene Cymek, Antje Christine Venjakob, Stefan Ruff, Otto Hans-Martin Lutz, Simon Hofmann, and Matthias Roetting. 2014. Entering PIN codes by smooth pursuit eye movements. *Journal of Eye Movement Research* 7, 4 (May 2014), 11. https://doi.org/10.16910/jemr.7.4.1

Ionut Damian, Michael Dietz, and Elisabeth André. 2018. The SSJ Framework: Augmenting Social Interactions Using Mobile Signal Processing and Live Feedback. *Frontiers in ICT* 5 (2018), 13. https://doi.org/10.3389/fict.2018.00013

Masoud Davari, Daniel Hienert, Dagmar Kern, and Stefan Dietze. 2020. The Role of Word-Eye-Fixations for Query Term Prediction. In *Proceedings of the 2020 Conference on Human Information Interaction and Retrieval (CHIIR '20)*. Association for Computing Machinery, New York, NY, USA, 422–426. https://doi.org/10.1145/3343413.3378010

Stijn De Beugher, Geert Brône, and Toon Goedemé. 2014. Automatic analysis of in-the-wild mobile eye-tracking experiments using object, face and person detection. In *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, Vol. 1. IEEE, 625–633. https://ieeexplore.ieee.org/abstract/document/7294867

Stijn De Beugher, Younes Ichiche, Geert Brône, and Toon Goedemé. 2012. Automatic Analysis of Eye-Tracking Data Using Object Detection Algorithms. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12)*. Association for Computing Machinery, New York, NY, USA, 677–680. https://doi.org/10.1145/2370216.2370363

Alexander De Luca, Martin Denzel, and Heinrich Hussmann. 2009. Look into My Eyes! Can You Guess My Password?. In *Proceedings of the 5th Symposium on Usable Privacy and Security (SOUPS '09)*. Association for Computing Machinery, New York, NY, USA, 12. https://doi.org/10.1145/1572532.1572542

Alexander De Luca, Roman Weiss, and Heiko Drewes. 2007. Evaluation of Eye-Gaze Interaction Methods for Security Enhanced PIN-Entry. In *Proceedings of the 19th Australasian Conference on Computer-Human Interaction: Entertaining User Interfaces (OZCHI '07)*. Association for Computing Machinery, New York, NY, USA, 199–202. https://doi.org/10.1145/1324892.1324932

Alexander De Luca, Roman Weiss, Heinrich Hußmann, and Xueli An. 2008. Eyepass - Eye-Stroke Authentication for Public Terminals. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 3003–3008. https://doi.org/10.1145/1358628.1358798

Oliver Deane, Eszter Toth, and Sang-Hoon Yeo. 2022. Deep-SAGA: a deep-learning-based system for automatic gaze annotation from eye-tracking data. *Behavior Research Methods* 55 (June 2022), 1372–1391. https://doi.org/10.3758/s13428-022-01833-4

Marianne DeAngelus and Jeff B. Pelz. 2009. Top-down control of eye movements: Yarbus revisited. *Visual Cognition* 17, 6-7 (Aug. 2009), 790–811. https://doi.org/10.1080/13506280902793843

Michael Desmond, Michael Muller, Zahra Ashktorab, Casey Dugan, Evelyn Duesterwald, Kristina Brimijoin, Catherine Finegan-Dollak, Michelle Brachman, Aabhas Sharma, Narendra Nath Joshi, and Qian Pan. 2021. Increasing the Speed and Accuracy of Data Labeling Through an AI Assisted Interface. In *26th International Conference on Intelligent User Interfaces (IUI '21)*. Association for Computing Machinery, New York, NY, USA, 392–401. https://doi.org/10.1145/3397481.3450698

Michael Dietz, Daniel Schork, Ionut Damian, Anika Steinert, Marten Haesner, and Elisabeth André. 2017. Automatic Detection of Visual Search for the Elderly using Eye and Head Tracking Data. *KI - Künstliche Intelligenz* 31, 4 (Nov. 2017), 339–348. https://doi.org/10.1007/s13218-017-0502-z

Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. In *Proceedings of the 31st International Conference on Machine Learning (ICML'14, Vol. 32(1))*. PMLR, 647–655. https://proceedings.mlr.press/v32/donahue14.html

Heiko Drewes. 2010. *Eye Gaze Tracking for Human Computer Interaction*. Dissertation. Ludwig-Maximilians-Universität München. https://doi.org/10.5282/edoc.11591

Heiko Drewes and Albrecht Schmidt. 2007. Interacting with the Computer Using Gaze Gestures. In *Human-Computer Interaction – INTERACT 2007*, Cécilia Baranauskas, Philippe Palanque, Julio Abascal, and Simone Diniz Junqueira Barbosa (Eds.). Springer, Berlin, Heidelberg, 475–488. https://doi.org/10.1007/978-3-540-74800-7_43

Jan Drewes, Guillaume S. Masson, and Anna Montagnini. 2012. Shifts in Reported Gaze Position Due to Changes in Pupil Size: Ground Truth and Compensation. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 209–212. https://doi.org/10.1145/2168556.2168596

Andrew T. Duchowski. 2002. A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers* 34, 4 (Nov. 2002), 455–470. https://doi.org/10.3758/BF03195475

Andrew T. Duchowski. 2007. *Eye Tracking Methodology: Theory and Practice* (2 ed.). Springer, London. https://doi.org/10.1007/978-1-84628-609-4

Andrew T. Duchowski. 2018. Gaze-based interaction: A 30 year retrospective. *Computers & Graphics* 73 (2018), 59–69. https://doi.org/10.1016/j.cag.2018.04.002

John J. Dudley and Per Ola Kristensson. 2018. A Review of User Interface Design for Interactive Machine Learning. *ACM Transactions on Interactive Intelligent Systems* 8, 2 (June 2018), 37. https://doi.org/10.1145/3185517

Abhishek Dutta and Andrew Zisserman. 2019. The VIA Annotation Software for Images, Audio and Video. In *Proceedings of the 27th ACM International Conference on Multimedia (MM '19)*. ACM, New York, NY, USA, 2276–2279. https://doi.org/10.1145/3343031.3350535

Carsten Eickhoff, Sebastian Dungs, and Vu Tran. 2015. An Eye-Tracking Study of Query Reformulation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 13–22. https://doi.org/10.1145/2766462.2767703

Monika Elepfandt and Martin Grund. 2012. Move It There, or Not? The Design of Voice Commands for Gaze with Speech. In *Proceedings of the 4th Workshop on Eye Gaze in Intelligent Human Machine Interaction (Gaze-In '12)*. Association for Computing Machinery, New York, NY, USA, 3. https://doi.org/10.1145/2401836.2401848

Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2015. Orbits: Gaze Interaction for Smart Watches Using Smooth Pursuit Eye Movements. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 457–466. https://doi.org/10.1145/2807442.2807499

Karen M. Evans, Robert A. Jacobs, John A. Tarduno, and Jeff B. Pelz. 2012. Collecting and Analyzing Eye-Tracking Data in Outdoor Environments. *Journal of Eye Movement Research* 5, 2 (Dec. 2012), 19. https://doi.org/10.16910/jemr.5.2.6

Alireza Fathi, Yin Li, and James M. Rehg. 2012. Learning to Recognize Daily Actions Using Gaze. In *Computer Vision – ECCV 2012 (Lecture Notes in Computer Science, Vol. 7572)*, Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid (Eds.). Springer, Berlin, Heidelberg, 314–327. https://doi.org/10.1007/978-3-642-33718-5_23

Anna Maria Feit, Lukas Vordemann, Seonwook Park, Caterina Berube, and Otmar Hilliges. 2020. Detecting Relevance during Decision-Making from Eye Movements for UI Adaptation. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 11. https://doi.org/10.1145/3379155.3391321

Anna Maria Feit, Shane Williams, Arturo Toledo, Ann Paradiso, Harish Kulkarni, Shaun Kane, and Meredith Ringel Morris. 2017. Toward Everyday Gaze Input: Accuracy and Precision of Eye Tracking and Implications for Design. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1118–1130. https://doi.org/10.1145/3025453.3025599

Michael Feld, Robert Neßelrath, and Tim Schwartz. 2019. Software Platforms and Toolkits for Building Multimodal Systems and Applications. In *The Handbook of Multimodal-Multisensor Interfaces: Language Processing, Software, Commercialization, and Emerging Directions*. Association for Computing Machinery and Morgan & Claypool, 145–190. https://doi.org/10.1145/3233795.3233801

Wenxin Feng, Jiangnan Zou, Andrew Kurauchi, Carlos H Morimoto, and Margrit Betke. 2021. HGaze Typing: Head-Gesture Assisted Gaze Typing. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '21 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 11. https://doi.org/10.1145/3448017.3457379

John M. Findlay and Iain D. Gilchrist. 2003. *Active vision: The psychology of looking and seeing.* Oxford University Press, New York, NY, US. https://doi.org/10.1093/acprof:oso/9780198524793.001.0001

J. Randall Flanagan and Roland S. Johansson. 2003. Action plans used in action observation. *Nature* 424, 6950 (Aug. 2003), 769–771. https://doi.org/10.1038/nature01861

Timo Fleischer, Ines Deibl, Stephanie Moser, Alexander Strahl, Simone Maier, and Joerg Zumbach. 2023. Mobile Eye Tracking during Experimenting with Digital Scaffolding—Gaze Shifts between Augmented Reality and Experiment during Zinc Iodide Electrolysis Set-Up. *Education Sciences* 13, 2 (2023), 20. https://doi.org/10.3390/educsci13020170

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin* 76, 5 (1971), 378–382. https://doi.org/10.1037/h0031619

Allan Fong, Daniel Hoffman, and Raj M. Ratwani. 2016. Making Sense of Mobile Eye-Tracking Data in the Real-World: A Human-in-the-Loop Analysis Approach. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 60, 1 (Sept. 2016), 1569–1573. https://doi.org/10.1177/1541931213601362

Charles P. Gerba, Adam L. Wuollet, Peter Raisanen, and Gerardo U. Lopez. 2016. Bacterial contamination of computer touch screens. *American Journal of Infection Control* 44, 3 (March 2016), 358–360. https://doi.org/10.1016/j.ajic.2015.10.013

Yulia Gizatdinova, Oleg Špakov, Outi Tuisku, Matthew Turk, and Veikko Surakka. 2018. Gaze and Head Pointing for Hands-Free Text Entry: Applicability to Ultra-Small Virtual Keyboards. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. Association for Computing Machinery, New York, NY, USA, 9. https://doi.org/10.1145/3204493.3204539

Joseph H. Goldberg and Jonathan I. Helfman. 2010. Visual Scanpath Representation. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications (ETRA '10)*. Association for Computing Machinery, New York, NY, USA, 203–210. https://doi.org/10.1145/1743666.1743717

Yoav Goldberg. 2017. *Neural Network Methods for Natural Language Processing* (1 ed.). Synthesis Lectures on Human Language Technologies, Vol. 10. Springer, Cham. https://doi.org/10.1007/978-3-031-02165-7

David Graff. 2002. The AQUAINT Corpus of English News Text LDC2002T31. https://doi.org/10.35111/pcbv-jq63

Céline Gressel, Rebekah Overdorf, Inken Hagenstedt, Murat Karaboga, Helmut Lurtz, Michael Raschke, and Andreas Bulling. 2023. Privacy-Aware Eye Tracking: Challenges and Future Directions. *IEEE Pervasive Computing* 22, 1 (2023), 95–102. https://doi.org/10.1109/MPRV.2022.3228660

Jacek Gwizdka. 2014a. Characterizing Relevance with Eye-Tracking Measures. In *Proceedings of the 5th Information Interaction in Context Symposium (IIiX '14)*. Association for Computing Machinery, New York, NY, USA, 58–67. https://doi.org/10.1145/2637002.2637011

Jacek Gwizdka. 2014b. News Stories Relevance Effects on Eye-Movements. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. Association for Computing Machinery, New York, NY, USA, 283–286. https://doi.org/10.1145/2578153.2578198

Jacek Gwizdka. 2017. Differences in Reading Between Word Search and Information Relevance Decisions: Evidence from Eye-Tracking. In *Information Systems and Neuroscience (Lecture Notes in Information Systems and Organisation, Vol. 16)*, Fred D. Davis, René Riedl, Jan vom Brocke, Pierre-Majorique Léger, and Adriane B. Randolph (Eds.). Springer, Cham, 141–147. https://doi.org/10.1007/978-3-319-41402-7_18

Jacek Gwizdka and Andrew Dillon. 2020. Eye-Tracking as a Method for Enhancing Research on Information Search. In *Understanding and Improving Information Search: A Cognitive Approach*, Wai Tat Fu and Herre van Oostendorp (Eds.). Springer International Publishing, Cham, 161–181. https://doi.org/10.1007/978-3-030-38825-6_9

Amin Haji-Abolhassani and James J. Clark. 2014. An inverse Yarbus process: Predicting observers' task from eye movement patterns. *Vision Research* 103 (Oct. 2014), 127–142. https://doi.org/10.1016/j.visres.2014.08.014

Jeremy Hales, Diako Mardanbegi, and David Rozado. 2013. Interacting with Objects in the Environment by Gaze and Hand Gestures. In *Proceedings of the 3rd International Workshop on Pervasive Eye Tracking and Mobile Eye-Based Interaction (PETMEI '13)*. 9. http://ecem2013.eye-movements.org

Dan Witzner Hansen and Qiang Ji. 2010. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 3 (2010), 478–500. https://doi.org/10.1109/TPAMI.2009.30

David R. Hardoon, John Shawe-Taylor, Antti Ajanki, Kai Puolamäki, and Samuel Kaski. 2007. Information Retrieval by Inferring Implicit Queries from Eye Movements. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 2)*, Marina Meila and Xiaotong Shen (Eds.). PMLR, San Juan, Puerto Rico, 179–186. https://proceedings.mlr.press/v2/hardoon07a.html

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature* 585, 7825 (Sept. 2020), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In *Advances in Psychology*, Peter A. Hancock and Najmedin Meshkati (Eds.). Vol. 52. North-Holland, 139–183. https://doi.org/10.1016/S0166-4115(08)62386-9

Almoctar Hassoumi, Vsevolod Peysakhovich, and Christophe Hurter. 2019. EyeFlow: Pursuit Interactions Using an Unmodified Camera. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. Association for Computing Machinery, New York, NY, USA, 10. https://doi.org/10.1145/3314111.3319820

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2020. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 2 (2020), 386–397. https://doi.org/10.1109/TPAMI.2018.2844175

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 770–778. https://doi.org/10.1109/CVPR.2016.90

Zhenyi He, Christof Lutteroth, and Ken Perlin. 2022. TapGazer: Text Entry with Finger Tapping and Gaze-Directed Word Selection. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 16. https://doi.org/10.1145/3491102.3501838

Ramin Hedeshy, Chandan Kumar, Raphael Menges, and Steffen Staab. 2021. Hummer: Text Entry by Gaze and Hum. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 11. https://doi.org/10.1145/3411764.3445501

Henna Heikkilä and Kari-Jouko Räihä. 2012. Simple Gaze Gestures and the Closure of the Eyes as an Interaction Technique. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 147–154. https://doi.org/10.1145/2168556.2168579

Alexander Heimerl, Tobias Baur, Florian Lingenfelser, Johannes Wagner, and Elisabeth André. 2019. NOVA - A tool for eXplainable Cooperative Machine Learning. In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*. IEEE, 109–115. https://doi.org/10.1109/ACII.2019.8925519

Daniel Hienert, Dagmar Kern, Matthew Mitsui, Chirag Shah, and Nicholas J. Belkin. 2019. Reading Protocol: Understanding What Has Been Read in Interactive Information Retrieval Tasks. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval (CHIIR '19)*. Association for Computing Machinery, New York, NY, USA, 73–81. https://doi.org/10.1145/3295750.3298921

Kenneth Holmqvist and Richard Andersson. 2017. *Eye tracking: A comprehensive guide to methods, paradigms and measures* (2 ed.). Lund Eye-Tracking Research Institute, Lund, Sweden.

Kenneth Holmqvist, Marcus Nyström, and Fiona Mulvey. 2012. Eye Tracker Data Quality: What It is and How to Measure It. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 45–52. https://doi.org/10.1145/2168556.2168563

Anthony J. Hornof and Tim Halverson. 2002. Cleaning up systematic error in eye-tracking data by using required fixation locations. *Behavior Research Methods, Instruments, & Computers* 34, 4 (Nov. 2002), 592–604. https://doi.org/10.3758/BF03195487

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *CoRR* abs/1704.04861 (2017), 9. http://arxiv.org/abs/1704.04861

Chien-Ming Huang and Bilge Mutlu. 2016. Anticipatory Robot Control for Efficient Human-Robot Collaboration. In *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI) (HRI '16)*. IEEE, 83–90. https://doi.org/10.1109/HRI.2016.7451737

Aulikki Hyrskykari, Howell Istance, and Stephen Vickers. 2012. Gaze Gestures or Dwell-Based Interaction?. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 229–232. https://doi.org/10.1145/2168556.2168602

Koki Ijuin, Kristiina Jokinen Jokinen, Tsuneo Kato, and Seiichi Yamamoto. 2019. Eye-gaze in Social Robot Interactions. *Proceedings of the Annual Conference of JSAI* JSAI2019 (2019), 3. https://doi.org/10.11517/pjsai.JSAI2019.0_3J3E402

Toshiya Isomoto, Toshiyuki Ando, Buntarou Shizuki, and Shin Takahashi. 2018. Dwell Time Reduction Technique Using Fitts' Law for Gaze-Based Target Acquisition. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. Association for Computing Machinery, New York, NY, USA, 7. https://doi.org/10.1145/3204493.3204532

Howell Istance, Richard Bates, Aulikki Hyrskykari, and Stephen Vickers. 2008. Snap Clutch, a Moded Approach to Solving the Midas Touch Problem. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications (ETRA '08)*. Association for Computing Machinery, New York, NY, USA, 221–228. https://doi.org/10.1145/1344471.1344523

Howell Istance, Aulikki Hyrskykari, Lauri Immonen, Santtu Mansikkamaa, and Stephen Vickers. 2010. Designing Gaze Gestures for Gaming: An Investigation of Performance. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications (ETRA '10)*. Association for Computing Machinery, New York, NY, USA, 323–330. https://doi.org/10.1145/1743666.1743740

Robert J. K. Jacob. 1990. What You Look at is What You Get: Eye Movement-Based Interaction Techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*. Association for Computing Machinery, New York, NY, USA, 11–18. https://doi.org/10.1145/97243.97246

Robert J. K. Jacob. 1991. The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look at is What You Get. *ACM Transactions on Information Systems* 9, 2 (April 1991), 152–169. https://doi.org/10.1145/123078.128728

Soumy Jacob, Shoya Ishimaru, Syed Saqib Bukhari, and Andreas Dengel. 2018. Gaze-Based Interest Detection on Newspaper Articles. In *Proceedings of the 7th Workshop on Pervasive Eye Tracking and Mobile Eye-Based Interaction (PETMEI '18)*. Association for Computing Machinery, New York, NY, USA, 7. https://doi.org/10.1145/3208031.3208034

Anthony David Jameson, Aaron Spaulding, and Neil Yorke-Smith. 2009. Introduction to the Special Issue on "Usable AI". *AI Magazine* 30, 4 (Sept. 2009), 11. https://doi.org/10.1609/aimag.v30i4.2273

Halszka Jarodzka, Kenneth Holmqvist, and Hans Gruber. 2017. Eye tracking in Educational Science: Theoretical frameworks and research agendas. *Journal of Eye Movement Research* 10, 1 (Feb. 2017), 18. https://doi.org/10.16910/jemr.10.1.3

Halszka Jarodzka, Tamara van Gog, Michael Dorr, Katharina Scheiter, and Peter Gerjets. 2013. Learning to see: Guiding students' attention via a Model's eye movements fosters learning. *Learning and Instruction* 25 (June 2013), 62–70. https://doi.org/10.1016/j.learninstruc.2012.11.004

Biye Jiang and John Canny. 2017. Interactive Machine Learning via a GPU-Accelerated Toolkit. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces (IUI '17)*. Association

for Computing Machinery, New York, NY, USA, 535–546. https://doi.org/10.1145/3025171.3025172

Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2017. Accurately Interpreting Clickthrough Data as Implicit Feedback. *SIGIR Forum* 51, 1 (Aug. 2017), 4–11. https://doi.org/10.1145/3130332.3130334

Samuel John, Erik Weitnauer, and Hendrik Koesling. 2012. Entropy-Based Correction of Eye Tracking Data for Static Scenes. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 297–300. https://doi.org/10.1145/2168556.2168620

Marcel A. Just and Patricia A. Carpenter. 1980. A theory of reading: From eye fixations to comprehension. *Psychological Review* 87 (1980), 329–354. https://doi.org/10.1037/0033-295X.87.4.329

Jari Kangas, Deepak Akkil, Jussi Rantala, Poika Isokoski, Päivi Majaranta, and Roope Raisamo. 2014. Gaze Gestures and Haptic Feedback in Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 435–438. https://doi.org/10.1145/2556288.2557040

Sebastian Kapp, Michael Barz, Sergey Mukhametov, Daniel Sonntag, and Jochen Kuhn. 2021. ARETT: Augmented Reality Eye Tracking Toolkit for Head Mounted Displays. *Sensors* 21, 6 (March 2021), 18. https://doi.org/10.3390/s21062234

Sebastian Kapp, Michael Thees, Fabian Beil, Thomas Weatherby, Jan-Philipp Burde, Thomas Wilhelm, and Jochen Kuhn. 2020. The Effects of Augmented Reality: A Comparative Study in an Undergraduate Physics Laboratory Course. In *Proceedings of the 12th International Conference on Computer Supported Education (CSEDU, Vol. 2)*. SciTePress, 197–206. https://doi.org/10.5220/0009793001970206

Moritz Kassner, William Patera, and Andreas Bulling. 2014. Pupil: An Open Source Platform for Pervasive Eye Tracking and Mobile Gaze-Based Interaction. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication (UbiComp '14 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 1151–1160. https://doi.org/10.1145/2638728.2641695

Asterios Katsifodimos and Sebastian Schelter. 2016. Apache Flink: Stream Analytics at Scale. In *2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW)*. IEEE, 193–193. https://doi.org/10.1109/IC2EW.2016.56

Mohamed Khamis, Andreas Bulling, and Florian Alt. 2015. Tackling Challenges of Interactive Public Displays Using Gaze. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers (UbiComp/ISWC'15 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 763–766. https://doi.org/10.1145/2800835.2807951

Mohamed Khamis, Carl Oechsner, Florian Alt, and Andreas Bulling. 2018a. VRpursuits: Interaction in Virtual Reality Using Smooth Pursuit Eye Movements. In *Proceedings of the 2018 International Conference on Advanced Visual Interfaces (AVI '18)*. Association for Computing Machinery, New York, NY, USA, 8. https://doi.org/10.1145/3206505.3206522

Mohamed Khamis, Ludwig Trotter, Ville Mäkelä, Emanuel von Zezschwitz, Jens Le, Andreas Bulling, and Florian Alt. 2018b. CueAuth: Comparing Touch, Mid-Air Gestures, and Gaze for Cue-Based Authentication on Situated Displays. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 4 (Dec. 2018), 22. https://doi.org/10.1145/3287052

David Kim, Paul Dunphy, Pam Briggs, Jonathan Hook, John W. Nicholson, James Nicholson, and Patrick Olivier. 2010. Multi-Touch Authentication on Tabletops. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1093–1102. https://doi.org/10.1145/1753326.1753489

Taejun Kim, Auejin Ham, Sunggeun Ahn, and Geehyuk Lee. 2022. Lattice Menu: A Low-Error Gaze-Based Marking Menu Utilizing Target-Assisted Gaze Gestures on a Lattice of Visual Anchors. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 12. https://doi.org/10.1145/3491102.3501977

Konstantin Klamka, Andreas Siegel, Stefan Vogt, Fabian Göbel, Sophie Stellmach, and Raimund Dachselt. 2015. Look & Pedal: Hands-Free Navigation in Zoomable Information Spaces through Gaze-Supported Foot Input. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction (ICMI '15)*. Association for Computing Machinery, New York, NY, USA, 123–130. https://doi.org/10.1145/2818346.2820751

Jeffrey Michael Klingner. 2010. *Measuring Cognitive Load During Visual Tasks by Combining Pupillometry and Eye Tracking*. Ph. D. Stanford University, Department of Computer Science. http://purl.stanford.edu/mv271zd7591

Kathryn Koehler, Fei Guo, Sheng Zhang, and Miguel P. Eckstein. 2014. What do saliency models predict? *Journal of Vision* 14, 3 (March 2014), 14–14. https://doi.org/10.1167/14.3.14

László Kopácsi, Michael Barz, Omair Shahzad Bhatti, and Daniel Sonntag. 2023. IMETA: An Interactive Mobile Eye Tracking Annotation Method for Semi-Automatic Fixation-to-AOI Mapping. In *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces (IUI '23 Companion)*. Association for Computing Machinery, New York, NY, USA, 33–36. https://doi.org/10.1145/3581754.3584125

László Kopácsi, Árpád Dobolyi, Áron Fóthi, Dávid Keller, Viktor Varga, and András Lőrincz. 2021. RATS: Robust Automated Tracking and Segmentation of Similar Instances. In *Artificial Neural Networks and Machine Learning – ICANN 2021 (ecture Notes in Computer Science, Vol. 12893)*, Igor Farkaš, Paolo Masulli, Sebastian Otte, and Stefan Wermter (Eds.). Springer, Cham, 507–518. https://doi.org/10.1007/978-3-030-86365-4_41

Vinay Krishna Sharma, Kamalpreet Saluja, Vimal Mollyn, and Pradipta Biswas. 2020. Eye Gaze Controlled Robotic Arm for Persons with Severe Speech and Motor Impairment. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 9. https://doi.org/10.1145/3379155.3391324

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems (NIPS 2012, Vol. 25)*. Curran Associates Inc., 1097–1105.

Chandan Kumar, Ramin Hedeshy, I. Scott MacKenzie, and Steffen Staab. 2020. TAGSwipe: Touch Assisted Gaze Swipe for Text Entry. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3313831.3376317

Manu Kumar, Tal Garfinkel, Dan Boneh, and Terry Winograd. 2007. Reducing Shoulder-Surfing by Using Gaze-Based Password Entry. In *Proceedings of the 3rd Symposium on Usable Privacy and Security (SOUPS '07)*. Association for Computing Machinery, New York, NY, USA, 13–19. https://doi.org/10.1145/1280680.1280683

Niharika Kumari, Verena Ruf, Sergey Mukhametov, Albrecht Schmidt, Jochen Kuhn, and Stefan Küchemann. 2021. Mobile Eye-Tracking Data Analysis Using Object Detection via YOLO v4. *Sensors* 21, 22 (Nov. 2021), 17. https://doi.org/10.3390/s21227668

Kuno Kurzhals. 2021. Image-Based Projection Labeling for Mobile Eye Tracking. In *ACM Symposium on Eye Tracking Research and Applications*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3448017.3457382

Kuno Kurzhals, Cyrill Fabian Bopp, Jochen Bässler, Felix Ebinger, and Daniel Weiskopf. 2014a. Benchmark Data for Evaluating Visualization and Analysis Techniques for Eye Tracking for Video Stimuli. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization (BELIV '14)*. Association for Computing Machinery, New York, NY, USA, 54–60. https://doi.org/10.1145/2669557.2669558

Kuno Kurzhals, Florian Heimerl, and Daniel Weiskopf. 2014b. ISeeCube: Visual Analysis of Gaze Data for Video. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. Association for Computing Machinery, New York, NY, USA, 43–50. https://doi.org/10.1145/2578153.2578158

Kuno Kurzhals, Marcel Hlawatsch, Christof Seeger, and Daniel Weiskopf. 2017. Visual Analytics for Mobile Eye Tracking. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 301–310. https://doi.org/10.1109/TVCG.2016.2598695

Kuno Kurzhals, Nils Rodrigues, Maurice Koch, Michael Stoll, Andres Bruhn, Andreas Bulling, and Daniel Weiskopf. 2020. Visual Analytics and Annotation of Pervasive Eye Tracking Video. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 9. https://doi.org/10.1145/3379155.3391326

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics* 7 (Aug. 2019), 453–466. https://doi.org/10.1162/tacl_a_00276

Christoph Käding, Erik Rodner, Alexander Freytag, and Joachim Denzler. 2017. Fine-Tuning Deep Neural Networks in Continuous Learning Scenarios. In *Computer Vision – ACCV 2016 Workshops (Lecture Notes in Computer Science, Vol. 10118)*, Chu-Song Chen, Jiwen Lu, and Kai-Kuang Ma (Eds.). Springer, Cham, 588–605. https://doi.org/10.1007/978-3-319-54526-4_43

Michael F. Land and Mary Hayhoe. 2001. In what ways do eye movements contribute to everyday activities? *Vision Research* 41, 25 (Nov. 2001), 3559–3565. https://doi.org/10.1016/S0042-6989(01)00102-X

Christian Lander, Sven Gehring, Antonio Krüger, Sebastian Boring, and Andreas Bulling. 2015. GazeProjector: Accurate Gaze Estimation and Seamless Gaze Interaction Across Multiple Displays. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 395–404. https://doi.org/10.1145/2807442.2807479

Christian Lander, Frederic Kerber, Thorsten Rauber, and Antonio Krüger. 2016. A Time-Efficient Re-Calibration Algorithm for Improved Long-Term Accuracy of Head-Worn Eye Trackers. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*. Association for Computing Machinery, New York, NY, USA, 213–216. https://doi.org/10.1145/2857491.2857513

Christian Lander, Markus Löchtefeld, and Antonio Krüger. 2018. HEYEbrid: A Hybrid Approach for Mobile Calibration-Free Gaze Estimation. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (Jan. 2018), 29. https://doi.org/10.1145/3161166

J. R. Landis and G. G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics* 33, 1 (March 1977), 159–174. https://doi.org/10.2307/2529310

Hoang H. Le, Duy M. H. Nguyen, Omair Shahzad Bhatti, László Kopácsi, Thinh P. Ngo, Binh T. Nguyen, Michael Barz, and Daniel Sonntag. 2025. I-MPN: inductive message passing network for efficient human-in-the-loop annotation of mobile eye tracking data. *Scientific Reports* 15, 1 (April 2025), 14192. https://doi.org/10.1038/s41598-025-94593-y

Dong Hoon Lee and Sae-Young Chung. 2021. Unsupervised embedding adaptation via early-stage feature reconstruction for few-shot classification. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*. PMLR, 6098–6108. http://proceedings.mlr.press/v139/lee21d.html

Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. 2017. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research* 18, 17 (2017), 1–5. http://jmlr.org/papers/v18/16-365.html

Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. 2019. SiamRPN++: Evolution of Siamese Visual Tracking With Very Deep Networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 4277–4286. https://doi.org/10.1109/CVPR.2019.00441

Chia-Yu Li, Daniel Ortega, Dirk Väth, Florian Lux, Lindsey Vanderlyn, Maximilian Schmidt, Michael Neumann, Moritz Völkel, Pavel Denisov, Sabrina Jenne, Zorica Kacarevic, and Ngoc Thang Vu. 2020. ADVISER: A Toolkit for Developing Multi-modal, Multi-domain and Socially-engaged Conversational Agents. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Association for Computational Linguistics, Online, 279–286. https://doi.org/10.18653/v1/2020.acl-demos.31

Xiangsheng Li, Yiqun Liu, Jiaxin Mao, Zexue He, Min Zhang, and Shaoping Ma. 2018b. Understanding Reading Attention Distribution during Relevance Judgement. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 733–742. https://doi.org/10.1145/3269206.3271764

Yuewen Li, Wenquan Feng, Shuchang Lyu, and Qi Zhao. 2023. Feature reconstruction and metric based network for few-shot object detection. *Computer Vision and Image Understanding* 227 (2023), 103600. https://doi.org/10.1016/j.cviu.2022.103600

Yin Li, Miao Liu, and James M. Rehg. 2018a. In the Eye of Beholder: Joint Learning of Gaze and Actions in First Person Video. In *Computer Vision – ECCV 2018 (Lecture Notes in Computer Science, Vol. 11209)*, Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (Eds.). Springer, Cham, 639–655. https://doi.org/10.1007/978-3-030-01228-1_38

Yin Li, Zhefan Ye, and James M. Rehg. 2015. Delving into egocentric actions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 287–295. https://doi.org/10.1109/CVPR.2015.7298625

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014 (Lecture Notes in Computer Science, Vol. 8693)*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.). Springer, Cham, 740–755. https://doi.org/10.1007/978-3-319-10602-1_48

Dachuan Liu, Bo Dong, Xing Gao, and Haining Wang. 2015. Exploiting Eye Tracking for Smartphone Authentication. In *Applied Cryptography and Network Security (Lecture Notes in Computer Science, Vol. 9092)*, Tal Malkin, Vladimir Kolesnikov, Allison Bishop Lewko, and Michalis Polychronakis (Eds.). Springer, Cham, 457–477. https://doi.org/10.1007/978-3-319-28166-7_22

Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. 2016. DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 1096–1104. https://doi.org/10.1109/CVPR.2016.124

Tomasz D. Loboda, Peter Brusilovsky, and Jöerg Brunstein. 2011. Inferring Word Relevance from Eye-Movements of Readers. In *Proceedings of the 16th International Conference on Intelligent User Interfaces (IUI '11)*. Association for Computing Machinery, New York, NY, USA, 175–184. https://doi.org/10.1145/1943403.1943431

D.G. Lowe. 1999. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Vol. 2. 1150–1157. https://doi.org/10.1109/ICCV.1999.790410

David G. Lowe. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60, 2 (Nov. 2004), 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

Francisco Lopez Luro and Veronica Sundstedt. 2019. A Comparative Study of Eye Tracking and Hand Controller for Aiming Tasks in Virtual Reality. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. Association for Computing Machinery, New York, NY, USA, 9. https://doi.org/10.1145/3317956.3318153

Mathias N. Lystbæk, Ken Pfeuffer, Jens Emil Sloth Grønbæk, and Hans Gellersen. 2022a. Exploring Gaze for Assisting Freehand Selection-Based Text Entry in AR. *Proceedings of the ACM on Human-Computer Interaction* 6, ETRA (May 2022), 16. https://doi.org/10.1145/3530882

Mathias N. Lystbæk, Peter Rosenberg, Ken Pfeuffer, Jens Emil Grønbæk, and Hans Gellersen. 2022b. Gaze-Hand Alignment: Combining Eye Gaze and Mid-Air Pointing for Interacting with Menus in Augmented Reality. *Proceedings of the ACM on Human-Computer Interaction* 6, ETRA (May 2022), 18. https://doi.org/10.1145/3530886

Minghuang Ma, Haoqi Fan, and Kris M. Kitani. 2016. Going Deeper into First-Person Activity Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 1894–1903. https://doi.org/10.1109/CVPR.2016.209

Eduardo Manuel Silva Machado, Ivan Carrillo, Miguel Collado, and Liming Chen. 2019. Visual Attention-Based Object Detection in Cluttered Environments. In *2019 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 133–139. https://doi.org/10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00064

Päivi Majaranta and Andreas Bulling. 2014. Eye Tracking and Eye-Based Human–Computer Interaction. In *Advances in Physiological Computing*, Stephen H. Fairclough and Kiel Gilleade (Eds.). Springer, London, 39–65. https://doi.org/10.1007/978-1-4471-6392-3_3

Päivi Majaranta, Jari Laitinen, Jari Kangas, and Poika Isokoski. 2019. Inducing Gaze Gestures by Static Illustrations. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. Association for Computing Machinery, New York, NY, USA, 5. https://doi.org/10.1145/3317956.3318151

Sarah Malone, Kristin Altmeyer, Markus Vogel, and Roland Brünken. 2020. Homogeneous and heterogeneous multiple representations in equation-solving problems: An eye-tracking study. *Journal of Computer Assisted Learning* 36, 6 (Dec. 2020), 781–798. https://doi.org/10.1111/jcal.12426

Diako Mardanbegi and Dan Witzner Hansen. 2011. Mobile Gaze-Based Screen Interaction in 3D Environments. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications (NGCA '11)*. Association for Computing Machinery, New York, NY, USA, 4. https://doi.org/10.1145/1983302.1983304

Diako Mardanbegi and Dan Witzner Hansen. 2012. Parallax Error in the Monocular Head-Mounted Eye Trackers. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12)*. Association for Computing Machinery, New York, NY, USA, 689–694. https://doi.org/10.1145/2370216.2370366

Diako Mardenbegi and Pernilla Qvarfordt. 2015. Creating Gaze Annotations in Head Mounted Displays. In *Proceedings of the 2015 ACM International Symposium on Wearable Computers (ISWC '15)*. Association for Computing Machinery, New York, NY, USA, 161–162. https://doi.org/10.1145/2802083.2808404

Thomas Mattusch, Mahsa Mirzamohammad, Mohamed Khamis, Andreas Bulling, and Florian Alt. 2018. Hidden Pursuits: Evaluating Gaze-Selection via Pursuits When the Stimuli's Trajectory is Partially Hidden. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA*

'18). Association for Computing Machinery, New York, NY, USA, 5. https://doi.org/10.1145/3204493.3204569

Cynthia Matuszek. 2018. Grounded Language Learning: Where Robotics and NLP Meet. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 5687–5691. https://doi.org/10.24963/ijcai.2018/810

Mark T. Maybury and Wolfgang Wahlster. 1998. Intelligent User Interfaces: An Introduction. In *Readings in Intelligent User Interfaces*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 13. https://dl.acm.org/doi/10.5555/286013.286437

George W. McConkie and Keith Rayner. 1976. Asymmetry of the perceptual span in reading. *Bulletin of the Psychonomic Society* 8, 5 (Nov. 1976), 365–368. https://doi.org/10.3758/BF03335168

Gregor Mehlmann, Markus Häring, Kathrin Janowski, Tobias Baur, Patrick Gebhard, and Elisabeth André. 2014. Exploring a Model of Gaze for Grounding in Multimodal HRI. In *Proceedings of the 16th International Conference on Multimodal Interaction (ICMI '14)*. Association for Computing Machinery, New York, NY, USA, 247–254. https://doi.org/10.1145/2663204.2663275

Johannes Meyer, Adrian Frank, Thomas Schlebusch, and Enkeljeda Kasneci. 2022. A CNN-Based Human Activity Recognition System Combining a Laser Feedback Interferometry Eye Movement Sensor and an IMU for Context-Aware Smart Glasses. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 4 (Dec. 2022), 24. https://doi.org/10.1145/3494998

Tristan Miller and Stefan Agne. 2005. Attention-Based Information Retrieval Using Eye Tracker Data. In *Proceedings of the 3rd International Conference on Knowledge Capture (K-CAP '05)*. Association for Computing Machinery, New York, NY, USA, 209–210. https://doi.org/10.1145/1088622.1088672

Katsumi Minakata, John Paulin Hansen, I. Scott MacKenzie, Per Bækgaard, and Vijay Rajanna. 2019. Pointing by Gaze, Head, and Foot in a Head-Mounted Display. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. Association for Computing Machinery, New York, NY, USA, 9. https://doi.org/10.1145/3317956.3318150

Darius Miniotas, Oleg Špakov, and I. Scott MacKenzie. 2004. Eye Gaze Interaction with Expanding Targets. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems (CHI EA '04)*. Association for Computing Machinery, New York, NY, USA, 1255–1258. https://doi.org/10.1145/985921.986037

Abdulrahman Mohamed Selim, Omair Shahzad Bhatti, Michael Barz, and Daniel Sonntag. 2024. Perceived Text Relevance Estimation Using Scanpaths and GNNs. In *Proceedings of the 26th International Conference on Multimodal Interaction (ICMI '24)*. Association for Computing Machinery, New York, NY, USA, 418–427. https://doi.org/10.1145/3678957.3685736

A. Monden, K. Matsumoto, and M. Yamato. 2005. Evaluation of Gaze-Added Target Selection Methods Suitable for General GUIs. *International Journal of Computer Applications in Technology* 24, 1 (June 2005), 17–24. https://doi.org/10.1504/IJCAT.2005.007201

Robert Neßelrath. 2016. *SiAM-dp: an open development platform for massively multimodal dialogue systems in cyber-physical environments*. Dissertation. Saarland University. https://dx.doi.org/10.22028/D291-26644

Marcus Nyström, Richard Andersson, Kenneth Holmqvist, and Joost van de Weijer. 2013. The influence of calibration method and eye physiology on eyetracking data quality. *Behavior Research Methods* 45, 1 (March 2013), 272–288. https://doi.org/10.3758/s13428-012-0247-4

A. Olsen. 2012. The Tobii I-VT Fixation Filter: Algorithm description. https://www.tobiipro.com/siteassets/tobii-pro/learn-and-support/analyze/how-do-we-classify-eye-movements/tobii-pro-i-vt-fixation-filter.pdf/?v=2012

Jason Orlosky, Takumi Toyama, Daniel Sonntag, and Kiyoshi Kiyokawa. 2014. Using Eye-Gaze and Visualization to Augment Memory. In *Distributed, Ambient, and Pervasive Interactions (Lecture Notes in Computer Science, Vol. 8530)*, Norbert Streitz and Panos Markopoulos (Eds.). Springer International Publishing, Cham, 282–291. https://doi.org/10.1007/978-3-319-07788-8_27

Sharon Oviatt. 2017. Theoretical foundations of multimodal interfaces and systems. In *The Handbook of Multimodal-Multisensor Interfaces: Foundations, User Modeling, and Common Modality Combinations*. Vol. 1. Association for Computing Machinery and Morgan & Claypool, 19–50. https://doi.org/10.1145/3015783.3015786

Sharon Oviatt. 2018. Ten Opportunities and Challenges for Advancing Student-Centered Multimodal Learning Analytics. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction (ICMI '18)*. Association for Computing Machinery, New York, NY, USA, 87–94. https://doi.org/10.1145/3242969.3243010

Sharon Oviatt and Philip R. Cohen. 2015. *The Paradigm Shift to Multimodality in Contemporary Computer Interfaces*. Morgan & Claypool Publishers. https://doi.org/10.2200/S00636ED1V01Y201503HCI030

Sharon Oviatt, Björn Schuller, Philip R Cohen, Daniel Sonntag, Gerasimos Potamianos, and Antonio Krüger (Eds.). 2017a. *The Handbook of Multimodal-Multisensor Interfaces: Foundations, User Modeling, and Common Modality Combinations*. Vol. 1. Association for Computing Machinery and Morgan & Claypool, New York, NY, USA. https://doi.org/10.1145/3015783

Sharon Oviatt, Björn Schuller, Philip R Cohen, Daniel Sonntag, Gerasimos Potamianos, and Antonio Krüger. 2017b. Introduction: Scope, Trends, and Paradigm Shift in the Field of Computer Interfaces. In *The Handbook of Multimodal-Multisensor Interfaces: Foundations, User Modeling, and Common Modality Combinations*. Vol. 1. Association for Computing Machinery and Morgan & Claypool, New York, NY, USA, 1–15. https://doi.org/10.1145/3015783.3015785

Sharon Oviatt, Björn Schuller, Philip R Cohen, Daniel Sonntag, Gerasimos Potamianos, and Antonio Krüger (Eds.). 2018. *The Handbook of Multimodal-Multisensor Interfaces: Signal Processing, Architectures, and Detection of Emotion and Cognition*. Vol. 2. Association for Computing Machinery and Morgan & Claypool, New York, NY, USA. https://doi.org/10.1145/3107990

Sharon Oviatt, Björn Schuller, Philip R Cohen, Daniel Sonntag, Gerasimos Potamianos, and Antonio Krüger (Eds.). 2019. *The Handbook of Multimodal-Multisensor Interfaces: Language Processing, Software, Commercialization, and Emerging Directions*. Vol. 3. Association for Computing Machinery and Morgan & Claypool, New York, NY, USA. https://doi.org/10.1145/3233795

Zsolt Palotai, Miklós Láng, András Sárkány, Zoltán Tősér, Daniel Sonntag, Takumi Toyama, and András Lőrincz. 2014. LabelMovie: Semi-supervised machine annotation tool with quality assurance and

crowd-sourcing options for videos. In *2014 12th International Workshop on Content-Based Multimedia Indexing (CBMI)*. IEEE, 1–4. https://doi.org/10.1109/CBMI.2014.6849850

Karen Panetta, Qianwen Wan, Aleksandra Kaszowska, Holly A. Taylor, and Sos Agaian. 2019. Software Architecture for Automating Cognitive Science Eye-Tracking Data Analysis and Object Annotation. *IEEE Transactions on Human-Machine Systems* 49, 3 (2019), 268–277. https://doi.org/10.1109/THMS.2019.2892919

Karen Panetta, Qianwen Wan, Srijith Rajeev, Aleksandra Kaszowska, Aaron L. Gardony, Kevin Naranjo, Holly A. Taylor, and Sos Agaian. 2020. ISeeColor: Method for Advanced Visual Analytics of Eye Tracking Data. *IEEE Access* 8 (2020), 52278–52287. https://doi.org/10.1109/ACCESS.2020.2980901

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12, 85 (2011), 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

Thies Pfeiffer, Patrick Renner, and Nadine Pfeiffer-Leßmann. 2016. EyeSee3D 2.0: Model-Based Real-Time Analysis of Mobile Eye-Tracking in Static and Dynamic Three-Dimensional Scenes. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*. Association for Computing Machinery, New York, NY, USA, 189–196. https://doi.org/10.1145/2857491.2857532

Ken Pfeuffer, Jason Alexander, Ming Ki Chong, and Hans Gellersen. 2014. Gaze-Touch: Combining Gaze with Multi-Touch for Interaction on the Same Surface. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. Association for Computing Machinery, New York, NY, USA, 509–518. https://doi.org/10.1145/2642918.2647397

Ken Pfeuffer, Jason Alexander, Ming Ki Chong, Yanxia Zhang, and Hans Gellersen. 2015. Gaze-Shifting: Direct-Indirect Input with Pen and Touch Modulated by Gaze. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 373–383. https://doi.org/10.1145/2807442.2807460

Ken Pfeuffer and Hans Gellersen. 2016. Gaze and Touch Interaction on Tablets. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. Association for Computing Machinery, New York, NY, USA, 301–311. https://doi.org/10.1145/2984511.2984514

Ken Pfeuffer, Benedikt Mayer, Diako Mardanbegi, and Hans Gellersen. 2017. Gaze + Pinch Interaction in Virtual Reality. In *Proceedings of the 5th Symposium on Spatial User Interaction (SUI '17)*. Association for Computing Machinery, New York, NY, USA, 99–108. https://doi.org/10.1145/3131277.3132180

Ken Pfeuffer, Melodie Vidal, Jayson Turner, Andreas Bulling, and Hans Gellersen. 2013. Pursuit Calibration: Making Gaze Calibration Less Tedious and More Flexible. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. Association for Computing Machinery, New York, NY, USA, 261–270. https://doi.org/10.1145/2501988.2501998

Alexander Plopski, Teresa Hirzle, Nahal Norouzi, Long Qian, Gerd Bruder, and Tobias Langlotz. 2022. The Eye in Extended Reality: A Survey on Gaze Interaction and Eye Tracking in Head-Worn Extended Reality. *Comput. Surveys* 55, 3 (March 2022), 39. https://doi.org/10.1145/3491207

Mihai Polceanu and Christine Lisetti. 2019. Time to Go ONLINE! A Modular Framework for Building Internet-Based Socially Interactive Agents. In *Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents (IVA '19)*. Association for Computing Machinery, New York, NY, USA, 227–229. https://doi.org/10.1145/3308532.3329452

Daniel F. Pontillo, Thomas B. Kinsman, and Jeff B. Pelz. 2010. SemantiCode: Using Content Similarity and Database-Driven Matching to Code Wearable Eyetracker Gaze Data. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications (ETRA '10)*. Association for Computing Machinery, New York, NY, USA, 267–270. https://doi.org/10.1145/1743666.1743729

Zahar Prasov and Joyce Y. Chai. 2008. What's in a Gaze? The Role of Eye-Gaze in Reference Resolution in Multimodal Conversational Interfaces. In *Proceedings of the 13th International Conference on Intelligent User Interfaces (IUI '08)*. Association for Computing Machinery, New York, NY, USA, 20–29. https://doi.org/10.1145/1378773.1378777

Pernilla Qvarfordt. 2017. Gaze-Informed Multimodal Interaction. In *The Handbook of Multimodal-Multisensor Interfaces: Foundations, User Modeling, and Common Modality Combinations*. Vol. 1. Association for Computing Machinery and Morgan & Claypool, New York, NY, USA, 365–402. https://doi.org/10.1145/3015783.3015794

Vijay Rajanna and John Paulin Hansen. 2018. Gaze Typing in Virtual Reality: Impact of Keyboard Design, Selection Method, and Motion. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. Association for Computing Machinery, New York, NY, USA, 10. https://doi.org/10.1145/3204493.3204541

Vijay Rajanna, Seth Polsley, Paul Taele, and Tracy Hammond. 2017. A Gaze Gesture-Based User Authentication System to Counter Shoulder-Surfing Attacks. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. Association for Computing Machinery, New York, NY, USA, 1978–1986. https://doi.org/10.1145/3027063.3053070

Argenis Ramirez Ramirez Gomez, Christopher Clarke, Ludwig Sidenmark, and Hans Gellersen. 2021. Gaze+Hold: Eyes-Only Direct Manipulation with Continuous Gaze Modulated by Closure of One Eye. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '21 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 12. https://doi.org/10.1145/3448017.3457381

Marjo Rauhala and Louiza Kalokairinou. 2021. *Ethics in Social Science and Humanities*. Guidance Note. European Commission, Brussels, Belgium. 25 pages. https://ec.europa.eu/info/funding-tenders/opportunities/docs/2021-2027/horizon/guidance/ethics-in-social-science-and-humanities_he_en.pdf

Keith Rayner. 1986. Eye movements and the perceptual span in beginning and skilled readers. *Journal of Experimental Child Psychology* 41, 2 (April 1986), 211–236. https://doi.org/10.1016/0022-0965(86)90037-8

Keith Rayner, Simon P. Liversedge, Sarah J. White, and Dorine Vergilino-Perez. 2003. Reading Disappearing Text: Cognitive Control of Eye Movements. *Psychological Science* 14, 4 (July 2003), 385–388. https://doi.org/10.1111/1467-9280.24483

Keith Rayner, Timothy J. Slattery, and Nathalie N. Bélanger. 2010. Eye movements, the perceptual span, and reading speed. *Psychonomic Bulletin & Review* 17, 6 (Dec. 2010), 834–839. https://doi.org/10.3758/PBR.17.6.834

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 512–519. https://doi.org/10.1109/CVPRW.2014.131

Robert W. Reeder, Peter Pirolli, and Stuart K. Card. 2001. WebEyeMapper and WebLogger: Tools for Analyzing Eye Tracking Data Collected in Web-Use Studies. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems (CHI EA '01)*. Association for Computing Machinery, New York, NY, USA, 19–20. https://doi.org/10.1145/634067.634082

Eyal M. Reingold and Heather Sheridan. 2011. Eye movements and visual expertise in chess and medicine. In *The Oxford Handbook of Eye Movements*, Simon P. Liversedge, Iain Gilchrist, and Stefan Everling (Eds.). Oxford University Press, 0. https://doi.org/10.1093/oxfordhb/9780199539789.013.0029

Katharina Reiter, Ken Pfeuffer, Augusto Esteves, Tim Mittermeier, and Florian Alt. 2022. Look & Turn: One-Handed and Expressive Menu Interaction by Gaze and Arm Turns in VR. In *2022 Symposium on Eye Tracking Research and Applications (ETRA '22)*. Association for Computing Machinery, New York, NY, USA, 7. https://doi.org/10.1145/3517031.3529233

Norbert Reithinger, Simon Bergweiler, Ralf Engel, Gerd Herzog, Norbert Pfleger, Massimo Romanelli, and Daniel Sonntag. 2005. A Look under the Hood: Design and Development of the First SmartWeb System Demonstrator. In *Proceedings of the 7th International Conference on Multimodal Interfaces (ICMI '05)*. Association for Computing Machinery, New York, NY, USA, 159–166. https://doi.org/10.1145/1088463.1088492

Norbert Reithinger and Daniel Sonntag. 2005. An integration framework for a mobile multimodal dialogue system accessing the semantic web. In *INTERSPEECH 2005*. ISCA, 841–844. https://doi.org/10.21437/Interspeech.2005-388

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc., 91–99. https://papers.nips.cc/paper_files/paper/2015/hash/14bfa6bb14875e45bba028a21ed38046-Abstract.html

Sheikh Rivu, Yasmeen Abdrabou, Thomas Mayer, Ken Pfeuffer, and Florian Alt. 2019. GazeButton: Enhancing Buttons with Eye Gaze Interactions. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. Association for Computing Machinery, New York, NY, USA, 7. https://doi.org/10.1145/3317956.3318154

Constantin A. Rothkopf, Dana H. Ballard, and Mary M. Hayhoe. 2016. Task and context determine where you look. *Journal of Vision* 7, 14 (July 2016), 16–16. https://doi.org/10.1167/7.14.16

Gerben Rotman, Nikolaus F. Troje, Roland S. Johansson, and J. Randall Flanagan. 2006. Eye Movements When Observing Predictable and Unpredictable Actions. *Journal of Neurophysiology* 96, 3 (2006), 1358–1369. https://doi.org/10.1152/jn.00227.2006

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* 115, 3 (2015), 211–252. https://doi.org/10.1007/s11263-015-0816-y

Jessica F. A. Salminen-Saari, Enrique Garcia Moreno-Esteva, Eeva Haataja, Miika Toivanen, Markku S. Hannula, and Anu Laine. 2021. Phases of collaborative mathematical problem solving and joint attention: a case study utilizing mobile gaze tracking. *ZDM – Mathematics Education* 53, 4 (Aug. 2021), 771–784. https://doi.org/10.1007/s11858-021-01280-z

Jarkko Salojärvi, I. Kojo, J. Simola, and S. Kaski. 2003. Can Relevance be Inferred from Eye Movements in Information Retrieval? *Workshop on Self-Organizing Maps (WSOM'03), Hibikino, Kitakyushu, Japan, September 11-14, 2003* (2003), 261–266.

Jarkko Salojärvi, Kai Puolamäki, and Samuel Kaski. 2004. Relevance feedback from eye movements for proactive information retrieval. In *Workshop on Processing Sensory Information for Proactive Systems (PSIPS 2004)*. 14–15.

Jarkko Salojärvi, Kai Puolamäki, and Samuel Kaski. 2005a. Implicit Relevance Feedback from Eye Movements. In *Artificial Neural Networks: Biological Inspirations – ICANN 2005*, Włodzisław Duch, Janusz Kacprzyk, Erkki Oja, and Sławomir Zadrożny (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 513–518. https://doi.org/10.1007/11550822_80

Jarkko Salojärvi, Kai Puolamäki, Jaana Simola, Lauri Kovanen, Ilpo Kojo, and Samuel Kaski. 2005b. *Inferring Relevance from Eye Movements: Feature Extraction*. WorkingPaper 82. Helsinki University of Technology, Finland. 45–67 pages. https://research.aalto.fi/en/publications/inferring-relevance-from-eye-movements-feature-extraction

Dario D. Salvucci and Joseph H. Goldberg. 2000. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the 2000 symposium on Eye tracking research & applications (ETRA '00)*. Association for Computing Machinery, New York, NY, USA, 71–78. https://doi.org/10.1145/355017.355028

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 4510–4520. https://doi.org/10.1109/CVPR.2018.00474

Hosnieh Sattar, Andreas Bulling, and Mario Fritz. 2017a. Predicting the Category and Attributes of Visual Search Targets Using Deep Gaze Pooling. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*. IEEE, 2740–2748. https://doi.org/10.1109/ICCVW.2017.322

Hosnieh Sattar, Mario Fritz, and Andreas Bulling. 2017b. Visual Decoding of Targets During Visual Search From Human Eye Fixations. *CoRR* abs/1706.05993 (2017), 9. http://arxiv.org/abs/1706.05993

Hosnieh Sattar, Sabine Müller, Mario Fritz, and Andreas Bulling. 2015. Prediction of search targets from fixations in open-world settings. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 981–990. https://doi.org/10.1109/CVPR.2015.7298700

Klaus Schoeffmann, Marco A. Hudelist, and Jochen Huber. 2015. Video Interaction Tools: A Survey of Recent Work. *Comput. Surveys* 48, 1 (Sept. 2015), 34. https://doi.org/10.1145/2808796

Nicu Sebe. 2009. Multimodal interfaces: Challenges and perspectives. *Journal of Ambient Intelligence and Smart Environments* 1, 1 (2009), 23–30. https://doi.org/10.3233/AIS-2009-0003

Korok Sengupta, Sabin Bhattarai, Sayan Sarcar, I. Scott MacKenzie, and Steffen Staab. 2020. Leveraging Error Correction in Voice-Based Text Entry by Talk-and-Gaze. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3313831.3376579

Marcos Serrano, Laurence Nigay, Jean-Yves L. Lawson, Andrew Ramsay, Roderick Murray-Smith, and Sebastian Denef. 2008. The Openinterface Framework: A Tool for Multimodal Interaction. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 3501–3506. https://doi.org/10.1145/1358628.1358881

C.E. Shannon. 1949. Communication in the Presence of Noise. *Proceedings of the IRE* 37, 1 (1949), 10–21. https://doi.org/10.1109/JRPROC.1949.232969

Jeffrey S. Shell, Roel Vertegaal, and Alexander W. Skaburskis. 2003. EyePliances: Attention-Seeking Devices That Respond to Visual Attention. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 770–771. https://doi.org/10.1145/765891.765981

Yuki Shiga, Takumi Toyama, Yuzuko Utsumi, Koichi Kise, and Andreas Dengel. 2014. Daily Activity Recognition Combining Gaze Motion and Visual Features. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication (UbiComp '14 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 1103–1111. https://doi.org/10.1145/2638728.2641691

Ben Shneiderman, Maxine Cohen, Steven Jacobs, Catherine Plaisant, Nicholas Diakopoulos, and Niklas Elmqvist. 2016. *Designing the User Interface: Strategies for Effective Human-Computer Interaction* (6 ed.). Pearson, Hoboken.

Ludwig Sidenmark, Christopher Clarke, Xuesong Zhang, Jenny Phu, and Hans Gellersen. 2020a. Outline Pursuits: Gaze-Assisted Selection of Occluded Objects in Virtual Reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376438

Ludwig Sidenmark, Diako Mardanbegi, Argenis Ramirez Gomez, Christopher Clarke, and Hans Gellersen. 2020b. BimodalGaze: Seamlessly Refined Pointing with Gaze and Filtered Gestural Head Movement. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 9. https://doi.org/10.1145/3379155.3391312

Ludwig Sidenmark, Dominic Potts, Bill Bapisch, and Hans Gellersen. 2021. Radi-Eye: Hands-Free Radial Interfaces for 3D Interaction Using Gaze-Activated Head-Crossing. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 11. https://doi.org/10.1145/3411764.3445697

Patrice Simard, Saleema Amershi, Max Chickering, Alicia Edelman Pelton, Soroush Ghorashi, Chris Meek, Gonzalo Ramos, Jina Suh, Johan Verwey, Mo Wang, and John Wernsing. 2017. *Machine Teaching: A New Paradigm for Building Machine Learning Systems.* Technical Report MSR-TR-2017-26. Microsoft. 14 pages. https://www.microsoft.com/en-us/research/publication/machine-teaching-new-paradigm-building-machine-learning-systems/

Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. 2015. SUN RGB-D: A RGB-D scene understanding benchmark suite. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 567–576. https://doi.org/10.1109/CVPR.2015.7298655

Daniel Sonntag. 2010. *Ontologies and Adaptivity in Dialogue for Question Answering*. Studies on the Semantic Web, Vol. 4. IOS Press. https://doi.org/10.3233/978-1-61499-338-4-i

Daniel Sonntag. 2012. Collaborative Multimodality. *KI - Künstliche Intelligenz* 26, 2 (May 2012), 161–168. https://doi.org/10.1007/s13218-012-0169-4

Daniel Sonntag. 2014. ERmed - Towards Medical Multimodal Cyber-Physical Environments. In *Foundations of Augmented Cognition. Advancing Human Performance and Decision-Making through Adaptive Systems - 8th International Conference, AC 2014, Held as Part of HCI International 2014, Heraklion, Crete, Greece, June 22-27, 2014. Proceedings (Lecture Notes in Computer Science, Vol. 8534)*, Dylan Schmorrow and Cali M. Fidopiastis (Eds.). Springer, Cham, 359–370. https://doi.org/10.1007/978-3-319-07527-3_34

Daniel Sonntag. 2015. Kognit: Intelligent Cognitive Enhancement Technology by Cognitive Models and Mixed Reality for Dementia Patients. In *2015 AAAI Fall Symposia, Arlington, Virginia, USA, November 12-14, 2015*. AAAI Press, 7. https://aaai.org/proceeding/02-fall-2015/

Daniel Sonntag, Michael Barz, and Thiago Gouvêa. 2024. A look under the hood of the Interactive Deep Learning Enterprise (No-IDLE). https://arxiv.org/abs/2406.19054

Daniel Sonntag, Michael Barz, Jan Zacharias, Sven Stauden, Vahid Rahmani, Aron Fóthi, and András Lőrincz. 2017. Fine-tuning deep CNN models on specific MS COCO categories. *CoRR* abs/1709.01476 (2017), 4. http://arxiv.org/abs/1709.01476

Sven Stauden, Michael Barz, and Daniel Sonntag. 2018. Visual Search Target Inference Using Bag of Deep Visual Words. In *KI 2018: Advances in Artificial Intelligence - 41st German Conference on AI, Berlin, Germany, September 24-28, 2018, Proceedings (Lecture Notes in Computer Science, Vol. 11117)*, Frank Trollmann and Anni-Yasmin Turhan (Eds.). Springer, Cham, 297–304. https://doi.org/10.1007/978-3-030-00111-7_25

Julian Steil. 2019. *Mobile eye tracking for everyone.* Dissertation. Saarland University, Saarbrücken, Germany. https://doi.org/10.22028/D291-30004

Julian Steil and Andreas Bulling. 2015. Discovery of Everyday Human Activities from Long-Term Visual Behaviour Using Topic Models. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 75–85. https://doi.org/10.1145/2750858.2807520

Julian Steil, Michael Xuelin Huang, and Andreas Bulling. 2018a. Fixation Detection for Head-Mounted Eye Tracking Based on Visual Similarity of Gaze Targets. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. Association for Computing Machinery, New York, NY, USA, 9. https://doi.org/10.1145/3204493.3204538

Julian Steil, Philipp Müller, Yusuke Sugano, and Andreas Bulling. 2018b. Forecasting User Attention during Everyday Mobile Interactions Using Device-Integrated and Wearable Sensors. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '18)*. Association for Computing Machinery, New York, NY, USA, 13. https://doi.org/10.1145/3229434.3229439

Sophie Stellmach and Raimund Dachselt. 2012a. Designing Gaze-Based User Interfaces for Steering in Virtual Environments. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 131–138. https://doi.org/10.1145/2168556.2168577

Sophie Stellmach and Raimund Dachselt. 2012b. Look & Touch: Gaze-Supported Target Acquisition. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2981–2990. https://doi.org/10.1145/2207676.2208709

Sophie Stellmach and Raimund Dachselt. 2013. Still Looking: Investigating Seamless Gaze-Supported Selection, Positioning, and Manipulation of Distant Targets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. Association for Computing Machinery, New York, NY, USA, 285–294. https://doi.org/10.1145/2470654.2470695

Sophie Stellmach, Sebastian Stober, Andreas Nürnberger, and Raimund Dachselt. 2011. Designing Gaze-Supported Multimodal Interactions for the Exploration of Large Image Collections. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications (NGCA '11)*. Association for Computing Machinery, New York, NY, USA, 8. https://doi.org/10.1145/1983302.1983303

Ömer Sümer, Patricia Goldberg, Kathleen Stürmer, Tina Seidel, Peter Gerjets, Ulrich Trautwein, and Enkelejda Kasneci. 2018. Teachers' Perception in the Classroom. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2315–2324. https://openaccess.thecvf.com/content_cvpr_2018_workshops/w47/html/Sumer_Teachers_Perception_in_CVPR_2018_paper.html

Talos. 2008. Eye scheme mulitlingual. https://commons.wikimedia.org/wiki/File:Eye_scheme_mulitlingual.svg (License: CC BY-SA 3.0 Unported, colorized by Jakov).

Siliang Tang, Ronan G. Reilly, and Christian Vorstius. 2012. EyeMap: a software system for visualizing and analyzing eye movement data in reading. *Behavior Research Methods* 44, 2 (June 2012), 420–438. https://doi.org/10.3758/s13428-011-0156-y

Michael Thees, Kristin Altmeyer, Sebastian Kapp, Eva Rexigel, Fabian Beil, Pascal Klein, Sarah Malone, Roland Brünken, and Jochen Kuhn. 2022. Augmented Reality for Presenting Real-Time Data During Students' Laboratory Work: Comparing a Head-Mounted Display With a Separate Display. *Frontiers in Psychology* 13 (2022), 16. https://www.frontiersin.org/articles/10.3389/fpsyg.2022.804742

Michael Thees, Sebastian Kapp, Martin P. Strzys, Fabian Beil, Paul Lukowicz, and Jochen Kuhn. 2020. Effects of augmented reality on learning and cognitive load in university physics laboratory courses. *Computers in Human Behavior* 108 (July 2020), 106316. https://doi.org/10.1016/j.chb.2020.106316

Jesse Thomason, Jivko Sinapov, Maxwell Svetlik, Peter Stone, and Raymond J. Mooney. 2016. Learning Multi-Modal Grounded Linguistic Semantics by Playing "I Spy". In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, 3477–3483.

Tobii Help. 2016. Specifications for the Tobii Eye Tracker 4C. `https://help.tobii.com/hc/en-us/articles/213414285-Specifications-for-the-Tobii-Eye-Tracker-4C` Publisher: Tobii Gaming.

Marc Tonsen, Julian Steil, Yusuke Sugano, and Andreas Bulling. 2017. InvisibleEye: Mobile Eye Tracking Using Multiple Low-Resolution Cameras and Learning-Based Gaze Estimation. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (Sept. 2017), 21. `https://doi.org/10.1145/3130971`

Takumi Toyama. 2015. *Towards wearable attention-aware systems in everyday environments*. Doctoral Thesis. Technische Universität Kaiserslautern, Kaiserslautern, Germany. `https://nbn-resolving.de/urn:nbn:de:hbz:386-kluedo-42260`

Takumi Toyama, Thomas Kieninger, Faisal Shafait, and Andreas Dengel. 2012. Gaze Guided Object Recognition Using a Head-Mounted Eye Tracker. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '12)*. Association for Computing Machinery, New York, NY, USA, 91–98. `https://doi.org/10.1145/2168556.2168570`

Takumi Toyama and Daniel Sonntag. 2015. Towards Episodic Memory Support for Dementia Patients by Recognizing Objects, Faces and Text in Eye Gaze. In *KI 2015: Advances in Artificial Intelligence - 38th Annual German Conference on AI, Dresden, Germany, September 21-25, 2015, Proceedings (Lecture Notes in Computer Science, Vol. 9324)*, Steffen Hölldobler, Markus Krötzsch, Rafael Peñaloza, and Sebastian Rudolph (Eds.). Springer, 316–323. `https://doi.org/10.1007/978-3-319-24489-1_29`

Jayson Turner, Jason Alexander, Andreas Bulling, Dominik Schmidt, and Hans Gellersen. 2013. Eye Pull, Eye Push: Moving Objects between Large Screens and Personal Devices with Gaze and Touch. In *Human-Computer Interaction – INTERACT 2013*, Paula Kotzé, Gary Marsden, Gitte Lindgaard, Janet Wesson, and Marco Winckler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 170–186. `https://doi.org/10.1007/978-3-642-40480-1_11`

Jayson Turner, Andreas Bulling, Jason Alexander, and Hans Gellersen. 2014. Cross-Device Gaze-Supported Point-to-Point Content Transfer. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. Association for Computing Machinery, New York, NY, USA, 19–26. `https://doi.org/10.1145/2578153.2578155`

Karan Uppal, Jaeah Kim, and Shashank Singh. 2022. Decoding Attention from Gaze: A Benchmark Dataset and End-to-End Models. In *NeuRIPS 2022 Workshop on Gaze Meets ML*. 22. `https://openreview.net/forum?id=1Ty3Xd9HUQv`

Pablo Valdunciel, Omair Shahzad Bhatti, Michael Barz, and Daniel Sonntag. 2022. Interactive Assessment Tool for Gaze-Based Machine Learning Models in Information Retrieval. In *Proceedings of the 2022 Conference on Human Information Interaction and Retrieval (CHIIR '22)*. Association for Computing Machinery, New York, NY, USA, 332–336. `https://doi.org/10.1145/3498366.3505834`

Eduardo Velloso, Flavio Luiz Coutinho, Andrew Kurauchi, and Carlos H Morimoto. 2018. Circular Orbits Detection for Gaze Interaction Using 2D Correlation and Profile Matching Algorithms. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. Association for Computing Machinery, New York, NY, USA, 9. `https://doi.org/10.1145/3204493.3204524`

Eduardo Velloso, Markus Wirth, Christian Weichel, Augusto Esteves, and Hans Gellersen. 2016. AmbiGaze: Direct Control of Ambient Devices by Gaze. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16)*. Association for Computing Machinery, New York, NY, USA, 812–817. https://doi.org/10.1145/2901790.2901867

Pranav Venuprasad, Li Xu, Enoch Huang, Andrew Gilman, Leanne Chukoskie Ph.D., and Pamela Cosman. 2020. Analyzing Gaze Behavior Using Object Detection and Unsupervised Clustering. In *ACM Symposium on Eye Tracking Research and Applications (ETRA '20 Full Papers)*. Association for Computing Machinery, New York, NY, USA, 9. https://doi.org/10.1145/3379155.3391316

Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: Spontaneous Interaction with Displays Based on Smooth Pursuit Eye Movement and Moving Targets. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*. Association for Computing Machinery, New York, NY, USA, 439–448. https://doi.org/10.1145/2493432.2493477

Alexandros Vigkos, Andreas Pauer, Davide Bevacqua, Luca Turturro, and Marta Kulesza. 2021. *XR and its potential for Europe.* Technical Report. ECORYS, Brussels. 108 pages. https://xreuropepotential.com/assets/pdf/ecorys-xr-2021-report.pdf

Johannes Wagner, Florian Lingenfelser, Tobias Baur, Ionut Damian, Felix Kistler, and Elisabeth André. 2013. The Social Signal Interpretation (SSI) Framework: Multimodal Signal Processing and Recognition in Real-Time. In *Proceedings of the 21st ACM International Conference on Multimedia (MM '13)*. Association for Computing Machinery, New York, NY, USA, 831–834. https://doi.org/10.1145/2502081.2502223

Jamie A. Ward, Paul Lukowicz, and Hans W. Gellersen. 2011. Performance Metrics for Activity Recognition. *ACM Transactions on Intelligent Systems and Technology* 2, 1 (Jan. 2011), 23. https://doi.org/10.1145/1889681.1889687

Jamie A. Ward, Paul Lukowicz, and Gerhard Tröster. 2006. Evaluating Performance in Continuous Context Recognition Using Event-Driven Error Characterisation. In *Location- and Context-Awareness (Lecture Notes in Computer Science, Vol. 3987)*, Mike Hazas, John Krumm, and Thomas Strang (Eds.). Springer, Berlin, Heidelberg, 239–255. https://doi.org/10.1007/11752967_16

Davis Wertheimer, Luming Tang, and Bharath Hariharan. 2021. Few-Shot Classification With Feature Map Reconstruction Networks. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 8008–8017. https://doi.org/10.1109/CVPR46437.2021.00792

Jacob O. Wobbrock, James Rubinstein, Michael W. Sawyer, and Andrew T. Duchowski. 2008. Longitudinal Evaluation of Discrete Consecutive Gaze Gestures for Text Entry. In *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications (ETRA '08)*. Association for Computing Machinery, New York, NY, USA, 11–18. https://doi.org/10.1145/1344471.1344475

Julian Wolf, Stephan Hess, David Bachmann, Quentin Lohmeyer, and Mirko Meboldt. 2018. Automating areas of interest analysis in mobile eye tracking experiments based on machine learning. *Journal of Eye Movement Research* 11, 6 (Dec. 2018), 11. https://doi.org/10.16910/jemr.11.6.6

Jeremy M. Wolfe. 1994. Guided Search 2.0 A revised model of visual search. *Psychonomic Bulletin & Review* 1, 2 (June 1994), 202–238. https://doi.org/10.3758/BF03200774

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. 2019. Detectron2. https://github.com/facebookresearch/detectron2

Jun Yang, Yu-Gang Jiang, Alexander G. Hauptmann, and Chong-Wah Ngo. 2007. Evaluating Bag-of-Visual-Words Representations in Scene Classification. In *Proceedings of the International Workshop on Workshop on Multimedia Information Retrieval (MIR '07)*. Association for Computing Machinery, New York, NY, USA, 197–206. https://doi.org/10.1145/1290082.1290111

Alfred L. Yarbus. 1967. *Eye Movements and Vision* (1 ed.). Springer, New York, NY, USA. https://doi.org/10.1007/978-1-4899-5379-7_1

Guixin Ye, Zhanyong Tang, Dingyi Fang, Xiaojiang Chen, Kwang In Kim, Ben Taylor, and Zheng Wang. 2017. Cracking Android Pattern Lock in Five Attempts. In *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*. The Internet Society. https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/cracking-android-pattern-lock-five-attempts/

Difeng Yu, Xueshi Lu, Rongkai Shi, Hai-Ning Liang, Tilman Dingler, Eduardo Velloso, and Jorge Goncalves. 2021. Gaze-Supported 3D Object Manipulation in Virtual Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 13. https://doi.org/10.1145/3411764.3445343

L.H. Yu and M. Eizenman. 2004. A new methodology for determining point-of-gaze in head-mounted eye tracking systems. *IEEE Transactions on Biomedical Engineering* 51, 10 (Oct. 2004), 1765–1773. https://doi.org/10.1109/TBME.2004.831523

Alan L. Yuille, Peter W. Hallinan, and David S. Cohen. 1992. Feature extraction from faces using deformable templates. *International Journal of Computer Vision* 8, 2 (Aug. 1992), 99–111. https://doi.org/10.1007/BF00127169

Jan Zacharias, Michael Barz, and Daniel Sonntag. 2018. A Survey on Deep Learning Toolkits and Libraries for Intelligent User Interfaces. *CoRR* abs/1803.04818 (2018), 10. http://arxiv.org/abs/1803.04818

Gregory J. Zelinsky, Yifan Peng, and Dimitris Samaras. 2013. Eye can read your mind: Decoding gaze fixations to reveal categorical search targets. *Journal of Vision* 13, 14 (Dec. 2013), 10–10. https://doi.org/10.1167/13.14.10

Shumin Zhai, Carlos Morimoto, and Steven Ihde. 1999. Manual and Gaze Input Cascaded (MAGIC) Pointing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. Association for Computing Machinery, New York, NY, USA, 246–253. https://doi.org/10.1145/302979.303053

Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. 2020. DeepEMD: Few-Shot Image Classification With Differentiable Earth Mover's Distance and Structured Classifiers. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 12200–12210. https://doi.org/10.1109/CVPR42600.2020.01222

Guangtao Zhang, John Paulin Hansen, and Katsumi Minakata. 2019. Hand- and Gaze-Control of Telepresence Robots. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. Association for Computing Machinery, New York, NY, USA, 8. https://doi.org/10.1145/3317956.3318149

Xiaoyi Zhang, Harish Kulkarni, and Meredith Ringel Morris. 2017. Smartphone-Based Gaze Gesture Communication for People with Motor Disabilities. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 2878–2889. https://doi.org/10.1145/3025453.3025790

Xinyong Zhang, Xiangshi Ren, and Hongbin Zha. 2008. Improving Eye Cursor's Stability for Eye Pointing Tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. Association for Computing Machinery, New York, NY, USA, 525–534. https://doi.org/10.1145/1357054.1357139

Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. 2015b. Appearance-Based Gaze Estimation in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 4511–4520. https://doi.org/10.1109/CVPR.2015.7299081

Yanxia Zhang, Andreas Bulling, and Hans Gellersen. 2013. SideWays: A Gaze Interface for Spontaneous Interaction with Situated Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 851–860. https://doi.org/10.1145/2470654.2470775

Yunfeng Zhang and Anthony J. Hornof. 2011. Mode-of-disparities error correction of eye-tracking data. *Behavior Research Methods* 43, 3 (Sept. 2011), 834–842. https://doi.org/10.3758/s13428-011-0073-0

Yunfeng Zhang and Anthony J. Hornof. 2014. Easy Post-Hoc Spatial Recalibration of Eye Tracking Data. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. Association for Computing Machinery, New York, NY, USA, 95–98. https://doi.org/10.1145/2578153.2578166

Yanxia Zhang, Jörg Müller, Ming Ki Chong, Andreas Bulling, and Hans Gellersen. 2014. GazeHorizon: Enabling Passers-by to Interact with Public Displays by Gaze. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*. Association for Computing Machinery, New York, NY, USA, 559–563. https://doi.org/10.1145/2632048.2636071

Yanxia Zhang, Sophie Stellmach, Abigail Sellen, and Andrew Blake. 2015a. The Costs and Benefits of Combining Gaze and Hand Gestures for Remote Interaction. In *Human-Computer Interaction – INTERACT 2015*. Springer-Verlag, Berlin, Heidelberg, 570–577. https://doi.org/10.1007/978-3-319-22698-9_39

Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. 2019. Object Detection With Deep Learning: A Review. *IEEE Transactions on Neural Networks and Learning Systems* 30, 11 (2019), 3212–3232. https://doi.org/10.1109/TNNLS.2018.2876865

Lech Świrski, Andreas Bulling, and Neil Dodgson. 2012. Robust real-time pupil tracking in highly off-axis images. In *Proceedings of the Symposium on Eye Tracking Research and Applications*. Association for Computing Machinery, New York, NY, USA, 173–176. https://doi.org/10.1145/2168556.2168585

Oleg Špakov. 2011. Comparison of gaze-to-objects mapping algorithms. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications (NGCA '11)*. Association for Computing Machinery, New York, NY, USA, 8. https://doi.org/10.1145/1983302.1983308

Oleg Špakov. 2012. Comparison of eye movement filters used in HCI. In *Proceedings of the Symposium on Eye Tracking Research and Applications*. Association for Computing Machinery, New York, NY, USA, 281–284. https://doi.org/10.1145/2168556.2168616

Oleg Špakov and Yulia Gizatdinova. 2014. Real-time hidden gaze point correction. In *Proceedings of the Symposium on Eye Tracking Research and Applications*. Association for Computing Machinery, New York, NY, USA, 291–294. https://doi.org/10.1145/2578153.2578200

Oleg Špakov, Harri Siirtola, Howell Istance, and Kari-Jouko Räihä. 2017. Visualizing the Reading Activity of People Learning to Read. *Journal of Eye Movement Research* 10, 5 (Nov. 2017), 12. https://doi.org/10.16910/jemr.10.5.5