Improving Usability and Accessibility of the Web with Eye Tracking

by

Raphael Philipp Menges

Approved Dissertation thesis for the partial fulfilment of the requirements for a
Doctor of Natural Sciences (Dr. rer. nat.)
Fachbereich 4: Informatik
Universität Koblenz-Landau

Chair of PhD Board: Prof. Dr. Jan Jürjens
Chair of PhD Commission: Prof. Dr. Harald F.O. von Korflesch
Examiner and Supervisor: Prof. Dr. Steffen Staab
Further Examiners: Prof. Dr. Andreas Bulling, Prof. Dr.-Ing. Dietrich Paulus

Date of the doctoral viva: 18th February 2021

Abstract

The Web is an essential component of moving our society to the digital age. We use it for communication, shopping, and doing our work. Most user interaction in the Web happens with Web page interfaces. Thus, the usability and accessibility of Web page interfaces are relevant areas of research to make the Web more useful. Eye tracking is a tool that can be helpful in both areas, performing usability testing and improving accessibility. It can be used to understand users' attention on Web pages and to support usability experts in their decision-making process. Moreover, eye tracking can be used as an input method to control an interface. This is especially useful for people with motor impairment, who cannot use traditional input devices like mouse and keyboard. However, interfaces on Web pages become more and more complex due to dynamics, i. e., changing contents like animated menus and photo carousels. We need general approaches to comprehend dynamics on Web pages, allowing for efficient usability analysis and enjoyable interaction with eye tracking. In the first part of this thesis, we report our work on improving gaze-based analysis of dynamic Web pages. Eye tracking can be used to collect the gaze signals of users, who browse a Web site and its pages. The gaze signals show a usability expert what parts in the Web page interface have been read, glanced at, or skipped. The aggregation of gaze signals allows a usability expert insight into the users' attention on a high-level, before looking into individual behavior. For this, all gaze signals must be aligned to the interface as experienced by the users. However, the user experience is heavily influenced by changing contents, as these may cover a substantial portion of the screen. We delineate unique states in Web page interfaces including changing contents, such that gaze signals from multiple users can be aggregated correctly. In the second part of this thesis, we report our work on improving the gaze-based interaction with dynamic Web pages. Eye tracking can be used to retrieve gaze signals while a user operates a computer. The gaze signals may be interpreted as input controlling an interface. Nowadays, eye tracking as an input method is mostly used to emulate mouse and keyboard functionality, hindering an enjoyable user experience. There exist a few Web browser prototypes that directly interpret gaze signals for control, but they do not work on dynamic Web pages. We have developed a method to extract interaction elements like hyperlinks and text inputs efficiently on Web pages, including changing contents. We adapt the interaction with those elements for eye tracking as the input method, such that a user can conveniently browse the Web hands-free. Both parts of this thesis conclude with user-centered evaluations of our methods, assessing the improvements in the user experience for usability experts and people with motor impairment, respectively.

Zusammenfassung

Das Web ist ein wesentlicher Bestandteil der Transformation unserer Gesellschaft in das digitale Zeitalter. Wir nutzen es zur Kommunikation, zum Einkaufen und für unsere berufliche Tätigkeit. Der größte Teil der Benutzerinteraktion im Web erfolgt über Webseiten. Daher sind die Benutzbarkeit und Zugänglichkeit von Webseiten relevante Forschungsbereiche, um das Web nützlicher zu machen. Eyetracking ist ein Werkzeug, das in beiden Bereichen hilfreich sein kann. Zum einen um Usability-Tests durchzuführen, zum anderen um die Zugänglichkeit zu verbessern. Es kann verwendet werden, um die Aufmerksamkeit der Benutzer auf Webseiten zu verstehen und Usability-Experten in ihrem Entscheidungsprozess zu unterstützen. Darüber hinaus kann Eyetracking als Eingabemethode zur Steuerung einer Webseite verwendet werden. Dies ist besonders nützlich für Menschen mit motorischen Beeinträchtigungen, die herkömmliche Eingabegeräte wie Maus und Tastatur nicht benutzen können. Allerdings werden Webseiten aufgrund von Dynamiken, d. h. wechselnden Inhalten wie animierte Menüs und Bilderkarussells, immer komplexer. Wir brauchen allgemeine Ansätze zum Verständnis der Dynamik auf Webseiten, die eine effiziente Usability-Analyse und eine angenehme Interaktion mit Eyetracking ermöglichen. Im ersten Teil dieser Arbeit berichten wir über unsere Forschung zur Verbesserung der blickbasierten Analyse von dynamischen Webseiten. Eyetracking kann verwendet werden, um die Blicke von Nutzern auf Webseiten zu erfassen. Die Blicke zeigen einem Usability-Experten, welche Teile auf der Webseite gelesen, überflogen oder übersprungen worden sind. Die Aggregation von Blicken ermöglicht einem Usability-Experten allgemeine Eindrücke über die Aufmerksamkeit der Nutzer, bevor sie sich mit dem individuellen Verhalten befasst. Dafür müssen alle Blicke entsprechend des von den Nutzern erlebten Inhalten verstanden werden. Die Benutzererfahrung wird jedoch stark von wechselnden Inhalten beeinflusst, da diese einen wesentlichen Teil des angezeigten Bildes ausmachen können. Wir grenzen unterschiedliche Zustände von Webseiten inklusive wechselnder Inhalte ab, so dass Blicke von mehreren Nutzern korrekt aggregiert werden können. Im zweiten Teil dieser Arbeit berichten wir über unsere Forschung zur Verbesserung der blickbasierten Interaktion mit dynamischen Webseiten. Eyetracking kann verwendet werden, um den Blick während der Nutzung zu erheben. Der Blick kann als Eingabe zur Steuerung einer Webseite interpretiert werden. Heutzutage wird die Blicksteuerung meist zur Emulation einer Maus oder Tastatur verwendet, was eine komfortable Bedienung erschwert. Es gibt wenige Webbrowser-Prototypen, die Blicke direkt zur Interaktion mit Webseiten nutzen. Diese funktionieren außerdem nicht auf dynamischen Webseiten. Wir haben eine Methode entwickelt, um Interaktionselemente wie Hyperlinks und Texteingaben effizient auf Webseiten mit wechselnden Inhalten zu extrahieren. Wir passen die Interaktion mit diesen Elementen für Eyetracking an, so dass ein Nutzer bequem und freihändig im Web surfen kann. Beide Teile dieser Arbeit schließen mit nutzerzentrierten Evaluationen unserer Methoden ab, wobei jeweils die Verbesserungen der Nutzererfahrung für Usability-Experten bzw. für Menschen mit motorischen Beeinträchtigungen untersucht werden.

Acknowledgments

When using the “we” in formulations, it is not an empty phrase. My research would not have been possible without the people whom I worked with at the Institute for Web Science and Technologies, and our national and international project partners.

Foremost, I want to thank my supervisor Steffen Staab and my colleague Chandan Kumar for many fruitful discussions and rich feedback about my research. They have co-authored nearly all of my publications and brought in their points of view, which made my work so much more relevant and understandable to a broader audience.

I want to thank my colleagues Lukas Schmelzeisen and Alexandra Baier for recommendations in any machine-learning-related question I came up with. And Martin Leinberger and Daniel Janke for daily chats to distract me from research and for the LaTeX template this thesis is based on. And Korok Sengupta for his assistance in conducting the GazeTheWeb lab studies at the university campus and many discussions about using eye tracking and voice input for human-computer interaction. And Daniel Müller and Christopher Dreide for their contribution to the GazeTheWeb software.

I want to thank the people involved in the research projects I participated in. I am grateful for all people from the MAMEM project, who organized wonderful meetings all over Europe. I would like to name Amihai Gotlieb and Meir Plotnik, who supported me during my two visits to Israel. And Georgios Liaros, with whom I developed the MAMEM platform and prepared it for the two MAMEM trial phases. Spiros Nikolopoulos (Information Technologies Institute, Centre for Research & Technologies Hellas, Greece), Meir Plotnik (Center of Advanced Technologies in Rehabilitation, Chaim Sheba Medical Center, Tel Hashomer, Israel), Dimitrios Athanasiou (Muscular Dystrophy Association, MDA Hellas, Greece), Sevasti Bostantjopoulou (University Department of Neurology, Aristotle University of Thessaloniki, Greece), and their respective teams, had put great effort in communicating with patients and executing the MAMEM trials on the pilot sites. Moreover, I enjoyed working on the GazeMining project with Christoph Schaefer, Tina Walber, and Hanadi Tamimi a lot. I am grateful to all the participants who took part in the studies across my research projects and provided us with relevant feedback.

Finally, I want to thank Silke Werger for administrative support at all times, including travel suggestions for getting to and staying at international conferences and project meetings.

My work was financially supported by the MAMEM project (European Union’s Horizon 2020, grant agreement number 644780) and the GazeMining project (Federal Ministry of Education and Research of Germany, project number 01IS17095B).

Contents

CHAPTER 1

Introduction

The World Wide Web has evolved to the primary source for information and tool for communication since its invention in 1989. Mostly, users access services on Web sites through the interaction with graphical user interfaces defined by Web page documents. Thus, the design and the interaction with Web pages have a considerable impact on our daily lives. The usability and accessibility of Web pages are relevant areas of research to make the Web more useful to all people.

Tracking of the eye movements of a user has been evolved as a helpful tool in both research areas of usability and accessibility. Eye tracking allows for estimating a user's eye gaze on a stimulus, like a graphical user interface on a computer screen. As of today, video-based remote eye-tracking systems have become affordable (e. g., devices like Tobii 4C or VI myGaze) and manufacturers start to integrate eye-tracking technology into monitors, mobile computers, and virtual reality headsets. Within the next years, we can expect a high number of end-user systems with Web access to gather gaze signals through dedicated hardware. We think it is time to break out of the laboratory setting and explore how eye tracking can be efficiently combined with Web technology to improve the usefulness of the Web.

Following our motivation to improve on usability and accessibility of the Web, we categorize the application of eye tracking in the Web as following:

- Eye gaze is a strong indicator for attention [55], which provides insights into how a user perceives a Web page and supports usability experts in assessing the user experience through *gaze-based analysis*. Researchers and companies are interested in estimating the attention on a Web page, e. g., which sections are read, glanced, or skipped by users, and analyzing the usability of a Web page in general. This analysis requires an accurate association between the coordinates of the gaze signals and a representation of the Web page as it had been a stimulus to the users.
- Eye gaze is a natural means for communication between humans, and research has investigated various strategies to incorporate gaze as a communication channel in the human-computer interaction [88], allowing a *gaze-based interaction* with graphical user interfaces. Interaction with Web pages through gaze signals as input has been utilized primarily in the context of accessibility. People with motor impairment are often restricted in their control of traditional input devices like mouse and keyboard, and interaction with eye movements can provide them with access to digital media and communication.

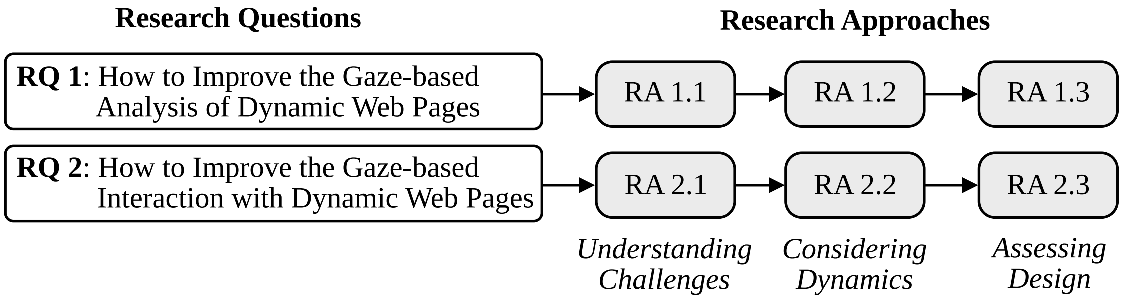
1.1 Research Questions

Gaze-based analysis and gaze-based interaction require to relate the gaze signals and the screen contents, which are acting as a stimulus to a user. Past research in eye tracking mostly relied on images or videos as a stimulus. Images or videos make it trivial to know what has been visible to a user at which point in time, allowing a straightforward interpretation of gaze signals. In contrast, graphical user interfaces usually offer multiple states, sometimes called views or modes. A user can select an interaction element like a hyperlink to navigate between these states. For example, a user can go to the home page of an online shop, which acts as the initial state of the graphical user interface. Then, the user selects the hyperlink which says *"Deal of the day."* The computer loads and displays the deal of the day from that online shop, which signifies a different state in the graphical user interface.

For gaze-based analysis, a usability expert may want to understand what led to the decision of purchasing a product. A common approach is to analyze the gaze signals just before a click on the "Buy-Button" happened — or did not happen. Gaze signals from multiple users can be aggregated such that comparisons between different users are possible. However, through the selection of other products, a user can explore further states of the graphical user interface of that online shop. Even though every user starts on the same initial state in the graphical user interface, the users may choose individual sequences through the available states, e. g., browsing various products when searching for a specific product. The interaction therefore potentially leads to a varying sequence of states per user. We need to delineate between the states in a graphical user interface to aggregate the gaze signals correctly per unique state.

For gaze-based interaction, we need to interpret the gaze signals in intelligent ways to allow rich interactions as required in the Web. A user might want to search for a product by entering a search query. In this case, we need to interpret the gaze signals as text input. Then, the user wants to select one of the found products. In this case, we need to interpret the gaze signals as a selection of a product. The intelligent interpretation of gaze signals requires an understanding of the interface semantics, i. e., the interaction elements that are available in the graphical user interface. We need to distinguish between a hyperlink, which can be clicked with a mouse, and a text input, which requires characters from keyboard input. Only then we can give users visual hints on which kind of interactions with the graphical user interface are available, let them conveniently select elements to interact with, and finally provide them with a gaze-based interaction that intelligently interprets the gaze signals as required for the specific interaction context.

In the early days of the Web, the graphical user interface as defined by a Web page document was not changed after initial loading, similar to a read-only text document in a word processor. Hyperlinks on a Web page could be selected to navigate to further Web pages via their respective address. The address of a Web page was used to delineate between the states of graphical user interfaces in the Web, enabling gaze-based analysis with a straightforward aggregation of multiple gaze signals per state. Furthermore, hyperlinks and text inputs could be extracted from the graphical user interface by parsing the initial Web page document. Then, the interaction could be adapted for eye tracking as the input method, enabling gaze-based interaction. But more elaborated dynamics on Web pages were introduced with plug-ins like Adobe Flash Player[1] in 1996, becoming widespread in the 2000s. These dynamics came in the shape of infinite scrolling lists, photo carousels with automatically changing photos, animated menus, or embedded videos. Those elements introduced further states of a graphical user interface per Web page and required sophisticated interactions. Nowadays, the concept of third-party plug-ins for contents in the Web is deprecated and dynamics are covered in the Web standards as defined by the World Wide Web Consortium (W3C) [143]. However, research in eye tracking has ignored this development so far. Dynamics have a big impact on the interactivity of a Web page and are an essential part of the modern Web. We focus our research on the user-centered application of eye tracking in the Web, comprehensively considering dynamics on Web pages. Thus, we define two research questions (RQ) that we cover with three research approaches (RA) each:


Research Questions
Research Approaches
RQ 1: How to Improve the Gaze-based Analysis of Dynamic Web Pages
RA 1.1
RA 1.2
RA 1.3
RQ 2: How to Improve the Gaze-based Interaction with Dynamic Web Pages
RA 2.1
RA 2.2
RA 2.3
Understanding Challenges
Considering Dynamics
Assessing Design

RQ 1: How to Improve the Gaze-based Analysis of Dynamic Web Pages. Analysis of unconscious human behavior has been the most prominent use case of eye tracking in the past. Besides the use of eye tracking in psychological studies about reactions of participants to manifold stimuli, eye tracking has recently become a vital tool in commercial testing of usability doing behavioral studies. Foremost the user experience on Web sites is analyzed since Web sites are the most important gateway for customers to many services. The vast amount of Web-based services leads to high competition and ongoing design changes to retain and attract customers. The ISO-9241-210 [85] also defines the design of a user-centered computer system as an inherently iterative process, requiring user-based testing at any stage of development. Thus, our first research question is *how to improve the gaze-based analysis of dynamic Web pages* for usability experts, such that they can better study the usability of Web pages while requiring less time.

RA 1.1: *What are the challenges for a usability expert in analyzing gaze data on dynamic Web pages?* We need to understand the data, techniques, and workflows that usability experts employ in behavioral studies with eye tracking on Web sites. Only then we can estimate the impact of dynamics in Web pages on their work and come up with methods that can make their tasks easier to be performed.

[1] `https://web.archive.org/web/20201115004736/https://www.adobe.com/products/flashplayer.html`, accessed on 20th May 2021.

RA 1.2: *Can we automatically detect when contents on a dynamic Web page visually change?*
We need to define which dynamics are of interest to a usability expert. Then we can investigate how to detect those dynamics on a Web page and decide how to disambiguate them from dynamics that might have no relevance for a behavioral study, e. g., underlining at hovering of a hyperlink.

RA 1.3: *How to evaluate recording and representation of dynamic Web pages for gaze-based analysis?*
We must evaluate whether the methods we introduce can support usability experts in their workflow. It is important to understand whether usability experts can save time when they employ our methods and whether the experience of the users is represented accurately with our methods.

RQ 2: How to Improve the Gaze-based Interaction with Dynamic Web Pages. Eye tracking as an input method can be especially useful for people with motor impairment. People with motor impairment may benefit from the opportunities of digital media and communication tools on the Web. However, people with motor impairment may be hindered in the operation of traditional input devices like mouse and keyboard. There have been a few prototypes for making interactions with a Web page through eye tracking as an input method more usable [123, 154, 2]. But these prototypes did not consider dynamics on Web pages and they were very limited in their functionality. Nevertheless, also ISO-9241-210 [85] defines that users with disabilities must be considered when designing user-centered computer systems. Thus, our second research question is *how to improve gaze-based interaction with dynamic Web pages* for people with motor impairment, such that the Web becomes more usable for them through eye tracking.

RA 2.1: *What are the challenges for a user in gaze-based interacting on dynamic Web pages?*
As of now, gaze-based Web browsing and interaction with Web pages is exclusively handled with the emulation of traditional input devices when it comes to end-user products [200]. The emulation approach allows users to indirectly control applications, e. g., a Web browser, as if they are operating the computer with a mouse and keyboard. We want to investigate which challenges users face when they use such an emulation approach for Web browsing, so we can improve the user experience with our methods.

RA 2.2: *Can we automatically detect interaction elements on dynamic Web pages?*
We require the interface semantics of a Web page to adapt the interaction for eye tracking as the input method. Past works that adapted interaction in the Web for unconventional input devices relied on the initial parsing of a Web page document to retrieve interaction elements. This is not sufficient for dynamic Web pages, as their contents may change after the Web page has been loaded. We require an efficient method to automatically retrieve, classify, and track interaction elements during a user session.

RA 2.3: *How to evaluate the user experience of gaze-based interaction on dynamic Web pages?*
We want to evaluate whether our approach of adaptation for gaze control leads to a better user experience for people with motor impairment. However, research in eye tracking for interaction has limited their evaluation metrics to performance indicators, i. e., how fast a user can execute a specific task. We instead take a user-centered perspective to judge the user experience in interacting with the Web as a whole.

1.2 Research Contributions

The first part of this thesis is about improving gaze-based analysis of dynamic Web pages, addressing RQ 1. We describe the background, related work, and challenges in the analysis of the usability of Web pages using eye tracking in Chapter 3, following RA 1.1. The Web page dynamics make it difficult to synchronize and aggregate gaze signals of multiple users. We divide the dynamics into those that are introduced through scrolling and those that are introduced through changing contents, either automatically or because of interaction by a user. For compensating dynamics that are caused by scrolling, we propose to take screenshots of the Web browser viewport during a user session and stitch them toward a single image. This produces a common space for all users on which their gaze signals can be aggregated. We enhance this representation of a Web page further by cropping the regions that are displayed relative to the viewport, e. g., navigation bars that stay on the top of the viewport— even when a user scrolls. We compose the stitched screenshot and those regions to an *enhanced representation* of a Web page, with the gaze signals of all users mapped accordingly. We describe the enhanced representation in Chapter 4, following RA 1.2. To cover changing contents on a Web page in the recordings, we build upon the enhanced representation and introduce the notion of *visual change* on a Web page. From the perspective of a usability expert, visual change may happen when a menu is expanded, or a carousel photo changes. We recorded a dataset of browsing on real-world Web sites and have manually labeled the recordings for these kinds of visual changes. We replicate that human-decision-based labeling process about visual change by using computer-vision features and machine learning. Finally, we merge visual coherent parts of the recordings across users into so-called *visual stimuli*. We describe the framework of visual stimuli discovery in Chapter 5, also following RA 1.2. For both methods, the enhanced representation and the visual stimuli discovery, we introduce and execute a methodology to evaluate them regarding the usefulness for usability experts, following RA 1.3.

The second part of this thesis is about interacting with Web pages through eye tracking, addressing RQ 2. We present pointing and typing methods, show how these methods can be combined to an emulation of mouse and keyboard input, and discuss challenges of this approach for users in Chapter 6, following RA 2.1. Consequently, we propose how to improve the user experience and suggest adapting the interaction on Web pages for gaze signals by considering the interface semantics of a Web page. For this, we come up with an efficient way to retrieve, classify, and track interaction elements on dynamic Web pages through introspection. We implement the introspection and adaptation in a gaze-controlled Web browser, named "GazeTheWeb," which is described in Chapter 7, following RA 2.2. We perform a comprehensive user-centered evaluation of the gaze-based interaction with GazeTheWeb. First, we report a study that compares the user experience using GazeTheWeb to an emulation approach in information retrieval tasks. Second, we show the feasibility of gaze-adapted interfaces for people with motor impairment in a lab study. Third, we present the results from a field study in which GazeTheWeb had been deployed to thirty participants with motor impairment for one month. We discuss the outcomes from the studies to assess our introspection and adaptation methods in Chapter 8, following RA 2.3.

Awards GazeTheWeb has been awarded on three distinct events. Its accessibility and user experience were signified with the TPG Accessibility Challenge Judges' Award at the 2017 Web For All conference in Perth, Australia. Its technical approach received an honorable mention at the 2017 TheWebConference in Perth, Australia. Finally, it scored third place in the first Digital Imagination Challenge by Unitymedia for its social impact and commercial potential. Moreover, we have received the Best Video Award at the 2018 ACM Symposium on Eye Tracking Research & Applications for the accompanying video about the enhanced representation method.

1.3 Software and Dataset Contributions

As part of this thesis, several software projects have been developed and were released under open-source licenses.

GazeTheWeb (`https://github.com/MAMEM/GazeTheWeb/tree/master/Browse`)
GazeTheWeb is a gaze-controlled Web browser. It supports unobtrusive gaze-based Web access with a browser incorporating unique interface design and Web engineering. Implemented with support by Daniel Müller and Christopher Dreide. See Chapter 7 for more details.

eyeGUI (`https://github.com/raphaelmenges/eyeGUI`)
eyeGUI is a framework to load, manipulate, and render graphical user interfaces for eye tracking as primary input method. See Section 6.3 for more details.

VisualStimuliDiscovery (`https://github.com/eyevido/visual-stimuli-discovery`)
The framework of visual stimuli discovery contains the tools and scripts required to process video and interaction recordings into stimulus shots and visual stimuli. See Chapter 5 for more details.

Schau genau! (`https://github.com/raphaelmenges/schaugenau`)
Schau genau! is an eye-tracking game for the State Horticultural Show in Landau, Germany, 2015. Implemented with support by Kevin Schmidt. See Section 6.2 for more details.

StupidSurvey.js (`https://github.com/raphaelmenges/StupidSurvey.js`)
StupidSurvey.js is a survey framework using JavaScript and no database. See Appendix G for multimedia form elements that we have designed for the expert survey about the visual stimuli discovery.

Moreover, we have released the dataset used in the visual stimuli discovery framework, consisting of the video and interaction recordings, the stimulus shots, and the visual stimuli as discussed in Chapter 5: `https://zenodo.org/record/4737774`.

1.4 Supporting Publications

This thesis is supported by following publications, listed in chronological order:

(1) **Raphael Menges**, Hanadi Tamimi, Chandan Kumar, Tina Walber, Christoph Schaefer, and Steffen Staab. "Enhanced Representation of Web Pages for Usability Analysis with Eye Tracking". In: *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*. ETRA '18. Warsaw, Poland: ACM, 2018, 18:1–18:9. URL: `http://doi.acm.org/10.1145/3204493.3204535`

(2) **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Improving User Experience of Eye Tracking-Based Interaction: Introspecting and Adapting Interfaces". In: *ACM Trans. Comput.-Hum. Interact.* 26.6 (Nov. 2019). Accepted May 2019, 37:1–37:46. URL: `http://doi.acm.org/10.1145/3338844`

(3) **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Eye tracking for Interaction: Adapting Multimedia Interfaces". In: *Signal Processing to Drive Human-Computer Interaction: EEG and eye-controlled interfaces*. Ed. by Spiros Nikolopoulos, Chandan Kumar, and Ioannis Kompatsiaris. Healthcare Technologies, Institution of Engineering and Technology. Institution of Engineering and Technology, 2020. Chap. 5, pp. 83–116. URL: `https://digital-library.theiet.org/content/books/10.1049/pbce129e%5C_ch5`

The contents of the first two sections of (1) are part of Chapter 3. The remaining sections of (1) are included in Chapter 4. The contents of the first section of (2) are distributed to this chapter and Chapter 6. The second and third sections of (2) are part of Chapter 6. A few paragraphs of the third section have been also used in Chapter 2 and Chapter 7. The fourth section of (2) forms Chapter 7 and is partly contained in Chapter 2. The remaining sections of (2) are included in Chapter 8. The first section of (3) is the basis for Chapter 2, whereas the second and third section are used in Chapter 6. The contents of Chapter 5 have not been published at the time of writing this thesis.

CHAPTER 2

Foundations

In this chapter, we provide the foundational knowledge for a reader to understand the remainder of this thesis. First, we introduce the terminology used throughout the thesis in Section 2.1. Second, we discuss details about the technology behind Web pages as graphical user interfaces in Section 2.2. Third, we explore the technology behind eye tracking, especially concerning its limitations that we have to consider, in Section 2.3.

2.1 Terminology

Before going into the details, we introduce the overall terminology used throughout this thesis. See Figure 2.1 for a drawing that relates the most important terms. The drawing shows a user in front of a setup that consists of a screen, an eye-tracking device attached below the screen, and a computer with an Internet connection. The user has requested a Web page from a Web server, which is transferred over the Internet as a Web page document. We define a Web page document as the textual contents, styles, scripts, and resources, like images or videos, a Web page consists of. A Web browser software on the computer renders the Web page document using its Web engine. Most Web pages exceed the available screen space in height. Therefore, a Web browser limits the rendering of a Web page to a viewport that fits on the screen. A user can move the crop of the Web page in the viewport by scrolling, i. e., with a mouse wheel or the arrow keys on a keyboard. The Web browser displays the viewport alongside its controls, e. g., an address bar, as the graphical user interface. We call the graphical user interface just *interface* in the rest of this thesis. The interface is sent to the screen as pixel data that tells the screen how to color each pixel of its pixel matrix. The pixel data from the interface on the screen acts as a visual stimulus for the user, who gazes at the screen and looks at the displayed contents. The eye-tracking device below the screen estimates the *point-of-regard* (POR) of the user within the two-dimensional space of the screen onto the stimulus. It sends the estimation of sequential PORs as gaze signal to the computer. The gaze signal consists of a series of gaze samples, each corresponding to one estimation of a POR at a point in time. The computer processes the incoming gaze signal with filtering to gaze data, which is a series of fixations and in-between saccades. We call the combination of the eye-tracking device and the gaze signal processing an *eye-tracking system*. Finally, the gaze data can be used either to analyze the attention of the user on the Web page or to enable the user to interact with the Web page.

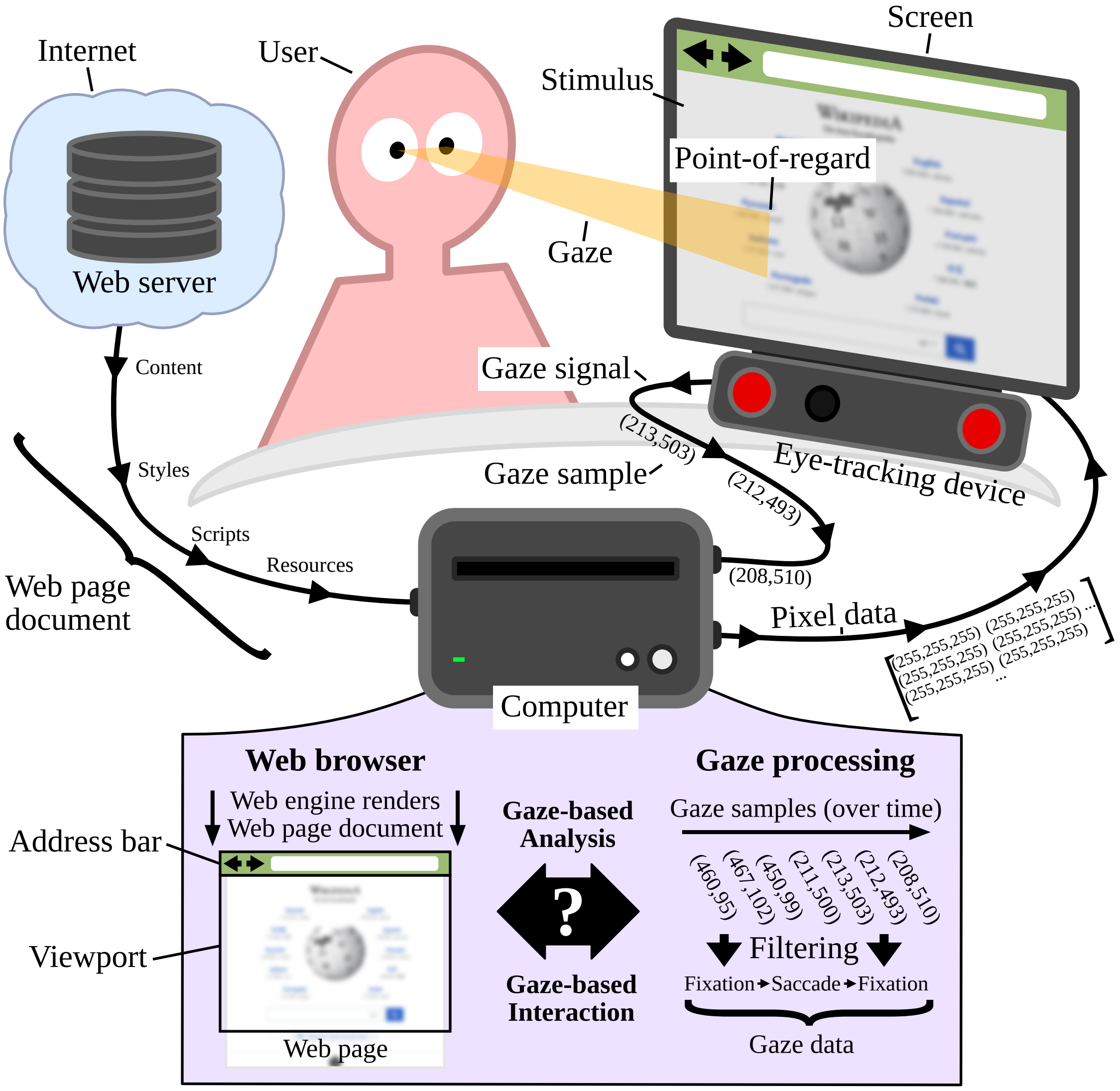


Figure 2.1: Setup of a user browsing a Web page while an eye-tracking system estimates their gaze. The question mark denotes the focus of our research: How to improve the gaze-based analysis, i. e., for better usability testing, and gaze-based interaction, i. e., for better accessibility, in this setup?

It is to be noted that the drawing omits traditional input devices like a mouse and keyboard. However, for the case of gaze-based analysis, a user usually performs interaction with the setup through mouse and keyboard input. Then, eye tracking is passively used to estimate the attention on a Web page. In contrast, when enabling interaction with eye tracking, we erase the need for mouse and keyboard and employ eye tracking actively to control a Web page.

2.2 Basics of Web Pages

Most interaction on the Internet happens through Web sites and their respective Web pages. Web sites can offer a vast variety of services to users, like providing information, allowing shopping, enabling communication, and even letting users edit text documents and excel sheets.

2.2.1 Access to Web Pages

A user instantiates a Web browser and provides a uniform resource locator (URL) in the address bar of the Web browser. A URL points to a specific resource in the Web, e. g., a Web page, such that the Web browser can query for that resource over the Internet. Such a URL is made up of several parts, as shown in Figure 2.2. (1) defines the protocol to communicate with the Web server. (2) is called the host or hostname. (3) is the subdomain. Most commonly, World Wide Web (www) is used. (4) is the domain. This is often considered as the address of a Web site. (5) is the top-level domain and indicates the purpose or the country of the service. (6) is an internal path on the Web server and specifies the Web page to be displayed from the Web site. If it no path given, a Web browser displays the `index.html` from the root directory. (7) after a question mark, parameters and their respective values might be appended, which, e. g., can be processed by scripts on a Web page. Here a specific video on YouTube is addressed.

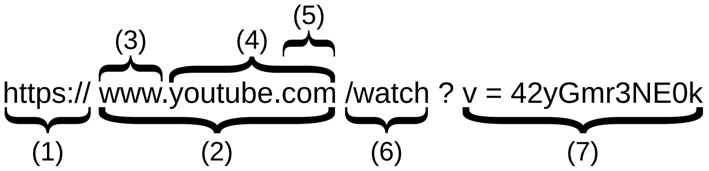


Figure 2.2: A URL that points to a video on the YouTube video platform, split into its parts.

We simplify the underlying terminology and define a Web site as a collection of Web pages that can be accessed under one domain through unique directory paths.

2.2.2 Technologies of Web Pages

A Web browser receives the contents of a requested Web page as a hypertext markup language (HTML) document. This document consists of extensible markup language (XML) elements of different types (container, hyperlink, image, etc.) with various attributes (address of a hyperlink, assigned style class, etc.). Container elements can have child elements of arbitrary type, allowing for nested XML structures. The structure is parsed into a document object model (DOM), which is a tree that consists of DOM nodes. Each DOM node corresponds to a single element from the HTML document and stores standard-conforming attributes as properties. The `<body>` element is parsed into the `document.body` node, which acts as root node of the tree that models the actual Web page contents.

The layout and design of the content are defined by cascading style sheets (CSS). One can define style classes with CSS that differ in their style attributes, e. g., color, font, position, size, or visibility. Each element in the HTML code can be assigned one or more style classes, and style attributes can even be set for each element. Cascading selectors are used per style attribute to decide which value is applied. Another core technology of the Web is JavaScript, a high-level, just-in-time compiled programming language that can be used to manipulate the contents, style, and interactivity of a Web page. CSS and JavaScript can be provided in separate files or defined within the HTML code in a `<style>` tag or a `<script>` tag, respectively. We choose the latter for the listings in this section.

2.2.3 Dynamics on Web Pages

In the past, a Web page would show the same interface for each request and would not change its contents upon interaction. A Web page might still have exceeded the available viewport height and the user could scroll the contents within a viewport. Nevertheless, every Web page could be treated as a static document, similar to a read-only text document in a word processor, which can be parsed a single time to retrieve the complete interface structure.

Modern Web pages however often change their contents automatically or via interaction after the initial loading. We call those Web pages *dynamic Web pages*. Dynamics can be achieved through JavaScript functions, which have read and write access to the DOM tree and the style classes. The JavaScript function calls can be triggered by local events (e. g., photo carousel, infinite page scrolling) and remote events (e. g., news updates from live events such as game scores). According to *"httparchive.org,"*[1] about 97% of the crawled Web pages make use of dynamic JavaScript requests (statistics from 11th September 2017). The JavaScript requests are especially utilized in the context of Asynchronous JavaScript and XML (AJAX), which allows dynamic addition and change of Web page contents via client-server communication when encountering local or remote events [139]. Hence, the visibility, position, and associated functionality of elements can change during user interaction with a Web page.

2.2.4 Example of an Interactive Photo Carousel

In the following, we walk through the example of a photo carousel made of compound elements. The carousel consists of three photos (see Figure 2.3), of which only the active one is displayed to a user. A user can switch between the available photos by clicking on buttons on the Web page. This example demonstrates how dynamic and interactive content can be implemented on a Web page. The HTML code in the following would be placed within the `<body>` element of a Web page document.

First, we provide HTML code that loads the photos into `<image>` elements in Listing 2.1. The `<image>` elements themselves are items of `<li>` elements within an unordered list `<ul>`. We add an id "carousel"

[1] `http://httparchive.org/interesting.php`

(a) Photo file `photo1.jpg` **(b)** Photo file `photo2.jpg` **(c)** Photo file `photo3.jpg`

Figure 2.3: The three photos that are used in the carousel example. The photos are under public domain.

to the unordered list element in line 1 and the style class “photo” to every image element. Furthermore, we assign the style class “show” to the active photo that will be shown in the carousel. We have not yet added any styling or scripting. Thus, the assignment of style classes has no effect. See Figure 2.4 for the Web page document as rendered by a Web browser. The Web browser displays the three photos as list items of the unordered list element. All photos are visible at once and each item of the unordered list element has a bullet in front of it.

```
1  <ul id="carousel">
2      <li>
3          <img src="photo1.jpg" class="photo show">
4      </li>
5      <li>
6          <img src="photo2.jpg" class="photo">
7      </li>
8      <li>
9          <img src="photo3.jpg" class="photo">
10     </li>
11 </ul>
```

Listing 2.1: HTML code within <body> element of the Web page document to display the three photos of the photo carousel. Note that the style classes are not yet defined and have no effect.

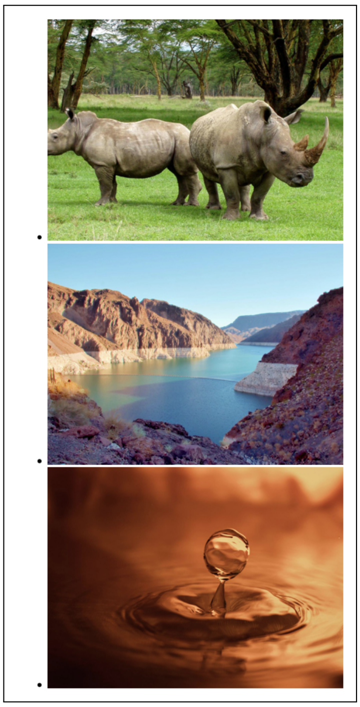


Figure 2.4: Web page rendering

Second, we define style classes to display only the active photo within the extents of the carousel. See Listing 2.2 for the respective CSS code. We select the unordered list element through its identifier in line 3. We remove the bullets in front of the items in line 4, set padding and margin to zero in line 5 and line 6, and specify the height in line 7. We set the position style attribute to `relative` in line 8, which tells a Web browser to place the unordered list element relative to the surrounding contents of the Web page in the normal flow. Furthermore, we set the position attribute of all photos to `absolute` in line 12. Such elements are placed at the position within its last ancestor element that has a position attribute of anything other than `static`. In the example it makes the photos to be positioned in absolute relation to the unordered list element. Because we do not provide any offset, all photos are displayed with zero offsets to the unordered list element. See Figure 2.5 for the Web page document as rendered by a Web browser after including the CSS code. Instead of the three photos in a vertical row, only one photo is visible at the exact position of the unordered list. The last defined photo `photo3.jpg` overlays the other photos. Now, the compound element looks like a photo carousel.

```
1  <style type="text/css">
2  
3  #carousel {
4      list-style-type: none;
5      padding: 0px;
6      margin: 0px;
7      height: 300px;
8      position: relative;
9  }
10 
11 .photo {
12     position: absolute;
13 }
14 
15 </style>
```

Listing 2.2: Definition of CSS code in the Web page document.

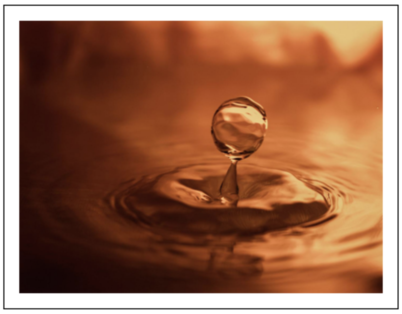


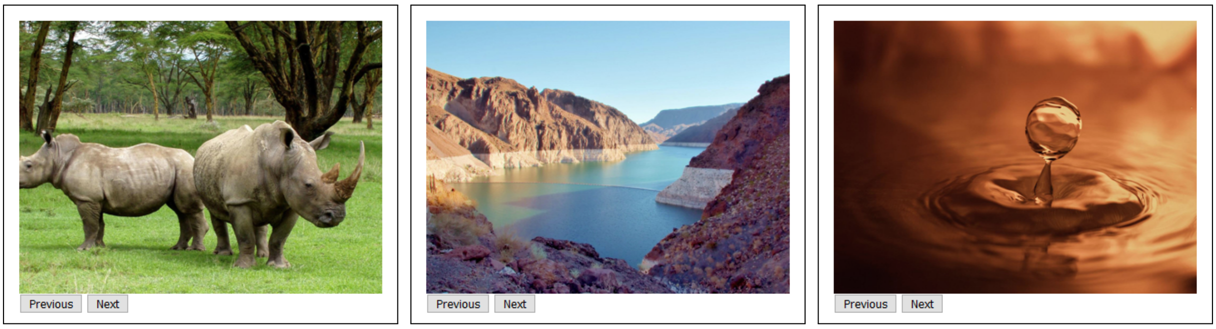
Figure 2.5: Web page rendering

Third, we add interactivity to the carousel, allowing a user to switch between the photos in the carousel. See Listing 2.3 for the HTML, JavaScript, and CSS code. The plan is to have a style class that brings an element into the front of all other elements. We assign that style class exclusively to the active photo. In the script, we first retrieve an array of all photos by querying for the style class `photo` in line 4. We remember the index of the active photo in the global variable `active`, which is defined in line 7. The function `changePhoto(offset)` in line 8 takes an offset to be applied to the value of `active`. It accepts positive and negative values, which allows for going both directions in switching through the photos. After resetting the style classes for the previously active photo in line 9, we update the index of the photo to be active in line 10. Then, we add the `show` style class to the image element that holds the active photo in line 11. In line 18 and following, we define the `show` style class within a `<style>` element. Here, we make use of the style attribute `z-index`, which indicates the order of rendering to a Web browser. Originally, all `<image>` elements are rendered at a z-index of zero. In such a case of equal z-indices, the order of definition within the HTML code decides which photo is visible to a user. By defining a higher z-index for one `<image>` element, its contained photo is rendered in front of the other photos. Finally, we add in line 24 and line 25 each a button that allows to navigate to the previous and the next photo, respectively, by executing `changePhoto(offset)` on a click event. See Figure 2.6 for screenshots of the three states of the carousel. Additionally, a periodical switch between the photos could be implemented by employing the `setInterval` functionality of JavaScript.

```
 1  <script type="text/javascript">
 2
 3  /* select carousel by id and photos by class */
 4  var photos = document.querySelectorAll(".photo");
 5
 6  /* manage active photo */
 7  var active = 0;
 8  function changePhoto(offset) {
 9      photos[active].className = "photo";
10      active = Math.min(Math.max(active + offset, 0), photos.length−1);
11      photos[active].className = "photo show";
12  }
13
14  </script>
15
16  <style type="text/css">
17
18  .show {
19      z−index: 1;
20  }
21
22  </style>
23
24  <button onclick="changePhoto(−1);">Previous</button>
25  <button onclick="changePhoto(+1);">Next</button>
```

Listing 2.3: Introduction of HTML, JavaScript, and CSS code to enable interaction with the carousel.

Defining different z-indices for the `<image>` elements is one way to implement a photo carousel. There are various other ways to accomplish the same outcome in a rendering of the Web page. One could scale all `<image>` elements but the one to be displayed to zero, move the elements out of the viewport, adapt the visibility attributes of the elements, or change the opacity of the elements. See Listing 2.4 for an implementation that utilizes the opacity instead of the z-index, while a Web browser renders the Web page identical to the variant from Listing 2.3.



(a) First photo in the carousel **(b)** Second photo in the carousel **(c)** Third photo in the carousel

Figure 2.6: Web page renderings of the carousel displaying the different photos.

The broad set of possibilities in implementing dynamics on a Web page is not exclusive to a photo carousel but applies in the same way to menus that can expand, pop-up windows, etc.

```
<style type="text/css">

.photo {
    position: absolute;
    opacity: 0;
}

.show {
    opacity: 1;
}

</style>
```

Listing 2.4: The photo switching can also be implemented by changing the opacity instead of the z-index.

2.3 Basics of Eye Tracking

The term “eye tracking” refers to the process of tracking the movement of the eyes in relation to the head, estimating the direction of eye gaze. The eye gaze direction can be related to the absolute head position and the geometry of the scene, such that a POR may be estimated. We call the sequential estimation of the POR *gaze signal*, and a single estimation *gaze sample*.

2.3.1 Anatomy of the Eye

The eyeball is covered mostly by a white protective tissue, called sclera. The colored part in the center of the eye is the iris, which encloses a hole, called pupil. The iris can shrink or grow the pupil; thus, less or more light enters the eye. Iris and pupil are covered by the transparent cornea. A flexible lens is placed directly behind the iris which refracts the light, such that it falls focused on the rear interior surface of the eye, the retina. The retina is covered by two types of photosensitive cells. The rods are sensitive to brightness. The cones are sensitive to chromatic light. There is a spot of high density in cones with a diameter of 1°– 2° on the retina, which is called fovea. The eye moves so that the light from the focused object falls into the fovea. Thus, our perceived focus of sight is sensed in the fovea region. Figure 2.7 shows a schematic illustration of the eye.

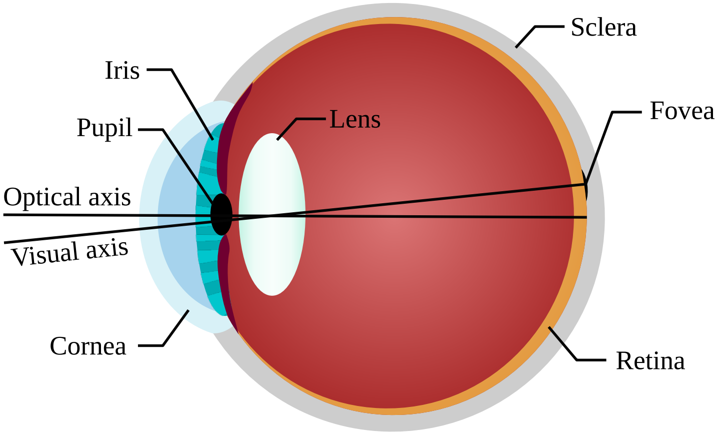


Figure 2.7: Schematic illustration of the anatomy of a human eye.

Each eye moves with the help of three antagonistic pairs of muscles that provide the eye with three degrees of freedom. One muscle pair performs horizontal movements, one muscle pair performs vertical movements, and the third muscle pair performs rotational movements around the direction of the view. The eye movements can be categorized into saccades or fixations [103]. Saccades are fast movements under 100 ms of both eyes in the same direction. These movements can be jump-like, gradual and smooth, or random. Fixations are times of about 100 – 600 ms for which an eye rests at a particular POR. The diameter of the fovea defines the portion in the visual angle of sharp sight, which is what we are interested in for eye tracking. S is the size of the object and D is the distance to the object. We can estimate the visual angle of an object [55] with

$$A = 2 \arctan \frac{S}{2D}. \tag{2.1}$$

The anatomy indicates two limitations of eye tracking which cannot be simply overcome with better

Table 2.1: Specifications of commercial video-based eye-tracking systems.

Model	myGaze n	4C	Skyle
Manufacturer	Visual Interaction GmbH	Tobii AB	eyeV GmbH
Sampling rate [Hz]	30	90	20 – 40
Operating distance	40 – 100 cm	50 – 95 cm	45 – 65 cm
Head box	50 x 30 cm at 65cm	40 x 30 cm at 75 cm	30 x 12 cm
Accuracy	0.4°	n/a	1-2°
Precision (RMS)	0.05°	n/a	n/a
Connection	USB 3.0	USB 2.0	USB 3.0
Screensize [in.]	10 – 27	max. 27	max. 24

technology. First, we need a calibration of an eye-tracking system for each individual. The eyeballs of different people have a variance in size, but more importantly, the fovea is not located directly opposite of the pupil on the optical axis, but 4°– 8° higher. Thus, the visual axis must be determined by a calibration procedure to find the POR. Second, the eye does not necessarily move such that the light of the object of interest falls in the perfect center of the fovea. The fovea diameter of about 1° limits the accuracy of an eye-tracking system to ±0.5° [52]. This is most often the best-case accuracy that manufacturers provide in their specifications, see Table 2.1. In comparison, the index fingernail also covers about 1° of the view at arm's length distance. This means at a distance of 70 cm and a visual angle of 1° an uncertainty of 1.2 cm. An uncertainty of 1.2 cm translates to about 44 pixels on a 24 in. monitor at a resolution of 1,920 × 1,080 pixels. See Figure 2.8 for an example of the uncertainty on a computer screen. Additionally, photosensitive cells in the eyes only react to changes in the amount of light they receive. In lack of changes, they quickly adapt and stop responding. Therefore, the eyes need to slightly move during long fixations, introducing micro-saccades [86].

2.3.2 Techniques to Track Eye Movements

Three techniques have become popular to measure eye movements [55]. Each technique has its motivations but also limitations, especially regarding everyday use.

Electro Oculography

In electro oculography (EOG), electrodes are attached to the skin around the eyes to measure an electric field that exists when the eyes rotate. The movement of an eye can be estimated by recording small differences in the electric potential of the skin around the eye. Specific electrode designs can record horizontal and vertical eye movements [69]. Originally, it was believed that the sensors measure the

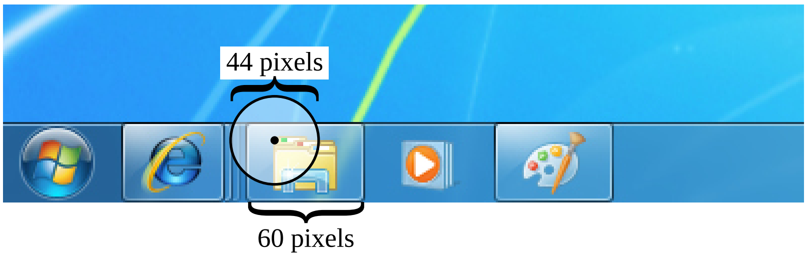


Figure 2.8: Example for the optimal accuracy of an eye-tracking device, visualized on the Microsoft Windows 7 taskbar, when considering 70 cm distance between head and screen and a 24 in. screen with a resolution of 1,920 × 1,080 pixels. The black dot signifies the true fixation of a user on the File Explorer icon. The white circle with a black border around the dot signifies the area of uncertainty as introduced by the anatomy of the human eye. Even if the eye-tracking system is perfectly calibrated, the estimated fixation may lay somewhere within the circle. In rare cases, the system might consider the fixation to be on the Internet Explorer icon, instead of the File Explorer icon.

electric potential of the eye muscles. It turned out that the sensor measures an electric field of the eye, which appears to be an electric dipole. However, the signal can change even when there is no eye movement, especially through external factors like nearby electric devices. The EOG offers a cheap and easy, but invasive technique to record large eye movements. The EOG technique is not well suited for the use-case of interaction, yet it is frequently used by clinicians. The advantage of the EOG technique is its ability to detect eye movements even when an eye is closed, e. g., while a person sleeps.

Scleral Search Coils

Using scleral search coils (SSC) [186], a coil of wire is attached to the eye. Usually, small coils of wire are embedded into a modified contact lens. A voltage is induced in the coil and movements of the coil within a magnetic field can be measured. The contact lenses are inserted into the eye under anesthetics. The advantage of SSC is the high accuracy and the nearly unlimited resolution in time. The disadvantage of SSC is that it is a highly invasive technique. Thus, this technique is mostly used in medical and psychological research. Similar to EOG, the SSC technique tracks eye movements only in relation to the head and does not indicate the POR.

Video Oculography

In video oculography, a setup with single or multiple cameras with no or multiple light sources is used to determine the movement of the eyes from captured images [149]. Mostly, infrared light is used to cause reflections on the cornea, which are visible as glints in captured images. As the cornea has a spherical shape, the glints stay in the same position independent of the eye gaze direction. A vector between pupil and glints can be calculated, which is used to estimate the direction of the eye gaze. Commercial eye-tracking systems mostly implement video-based eye tracking techniques, see Table 2.1 on page 18.

The video-based gaze estimation can be problematic for dark brown eyes for which the contrast between the brown iris and the black pupil is very low, as it makes it difficult to detect the pupil in the captured image. An additional infrared light source can be placed close to a camera, such that the emitted light reflects from the retina straight back to the camera. The reflected light makes the pupil appear white in the captured image and provides more contrast to the iris, which helps in detecting the pupil on the captured image. The effect is well known as the “red-eye” effect when photographing faces with a flash. Modern eye-tracking devices combine a single or two cameras with an array of carefully placed light sources. The light sources can flicker periodically to illuminate the eyes only for certain image captures. For example, let’s consider an eye-tracking device with one camera and three light sources. Two light sources that are placed a few centimeters away from the camera might be used to produce glints on the eyes in one image while capturing dark pupils. A third light source that is placed close to the camera produces another image of the eyes with bright pupils. The images from both setups are captured sequentially and make the system more robust regarding the color of the iris of a user. Either the image of the black pupils or the image of the white pupils would provide a sufficient contrast to detect the pupils on the image. Modern eye-tracking devices vary in the frequency to report so computed gaze coordinates from 30 Hz up to 1,000 Hz. Interaction through eye tracking requires a frequency that is comparable with the refresh rate of the screen. Thus, we recommend using systems that sample gaze at 60 Hz for interaction with eye tracking. Eye detection and tracking with video oculography remain a very challenging task due to several unique issues, including external illumination, viewing angle, occlusion of the eye, and head poses that make it difficult for the camera to detect the eyes of the user.

Eye-tracking systems using the video-oculography technique are available as table and head-mounted devices. Table-mounted eye-tracking devices, also known as remote eye-tracking devices, are designed to gather POR estimations in the coordinate system of the screen. Thus, in the context of our research, we primarily use table-mounted eye-tracking devices. See Figure 2.9 for such a system.

2.3.3 Gaze Signal Processing

Eye-tracking systems produce gaze signals, which require processing to account for their limited accuracy and precision.

Calibration of an Eye-Tracking System

A direct calculation of the POR from the captured images would not only require the spatial geometry of the eye-tracking system, the relative screen placement, and the eye position, but also the radius of the eyeball and offset of the fovea, which is specific to an individual. Commercial systems provide a

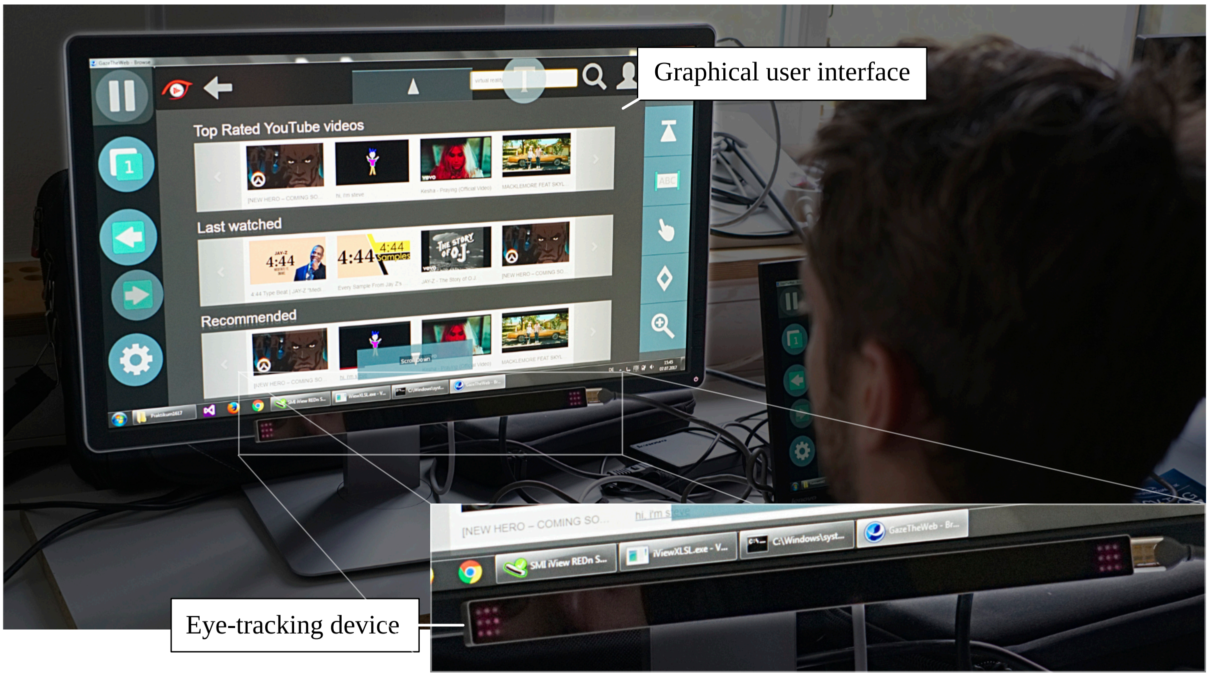


Figure 2.9: Photo of an SMI RED-n eye-tracking system with dark-pupil eye tracking.

dedicated mode for calibration, during which the screen shows a stimulus that is drawn by the software of the eye-tracking system. The stimulus usually consists of a, i. e., a dot, which moves between predefined coordinates, called *calibration points*. The dot moves sequentially from one calibration point to the next and stops at each calibration point for a short amount of time. The user is asked to follow the dot on the screen using their eyes but not their head. This calibration procedure provides reference data from the eye-tracking device for each calibration point. The reference data is used to calculate the parameters for the mapping of the glint-pupil vector on the captured images to the respective coordinates on the screen. Calibrations usually make use of five to nine calibration points, depending on the desired accuracy. The quality of the mapping can be tested using the reference data by comparing the mapped gaze signals to the true coordinates of the calibration points.

Error Modeling of Gaze Signals

Besides lighting, visual angle, and iris color, also calibration drift, tremor, and microsaccades of the eyes, which are in a magnitude of 0.1°, contribute to the inaccuracy of POR estimation [75]. Error in gaze signal can be modeled using the two measures of precision and accuracy. *Precision* is defined as the ability of the eye-tracking system to reliably reproduce the POR at a fixation. The precision error is perceived as high-frequency noise in the POR estimation. *Accuracy* is defined as the average difference between the coordinate of the focused object and the measured POR. The accuracy error is perceived as bias in the gaze signal. The body of work in *signal processing* includes techniques to filter and smooth the gaze input for improved precision [64, 106, 75]. Research in the field of *design and interaction* has proposed adaptive designs like enlarged and screen-centered interface elements or visual feedback [103, 18], to compensate for accuracy issues when it is crucial to determine the attention on a certain element of an interface that is sized below the accuracy limit of an eye-tracking system.

Filtering of Gaze Signals

Filtering of gaze signals has the purpose to improve the precision and delineate between fixations and saccades. We call the result of filtering gaze signals toward fixations and saccades then *gaze data*. Some eye-tracking systems already incorporate filtering, yet the manufacturers often do not specify their approach. There are four popular approaches to filter recorded gaze signals and identify fixations [156], as described next:

- **Velocity-threshold identification.** A point-to-point velocity is calculated between the coordinates of sequential gaze samples. Saccadic eye movements are considered to have a velocity of more than 300°/s, whereas fixations are considered to have a velocity of less than 100°/s. A fixed threshold criterion can be applied to separate fixations and in-between saccades. For each fixation, a centroid of the gaze samples is calculated. The time of the first gaze sample is the start of the fixation, while the time from the first gaze sample in the fixation until the time of the last gaze sample in the fixation is considered as the duration of the fixation. However, the static threshold is prone to over-segmentation of gaze signals into fixations. Yet, velocity-based filters are the most popular filtering approach, as they are effective and simple to implement.
- **Hidden-Markov-model identification.** Saccades and fixations can be modeled as a probabilistic state machine using a hidden-Markov model. Two states are implemented. One state represents the velocity distribution for gaze samples of saccades and the other state represents the velocity distribution for gaze samples of fixations. The hidden-Markov-model filter allows a more robust identification than fixed-threshold approaches, but the filter is more difficult to implement.
- **Dispersion-threshold identification.** Gaze samples that belong to a fixation tend to cluster spatially closely together. One can go through the gaze signal over time and calculate the spatial spread of the coordinates of gaze samples in a time window. When the spatial spread is below a preset threshold, a time window is considered a fixation. A time window in which content is considered a fixation can be expanded until the spatial dispersion of the covered gaze samples is too high. For each time window, its spatial centroid is taken as the coordinate of its contained fixation.
- **Area-of-interest identification.** The so-far described approaches to identify fixations can identify them at any location on the screen. In contrast, area-of-interest identification considers only gaze signals that occur within specified target areas. The approach uses a duration threshold to distinguish fixation in the target area from passing saccades in those areas. The target areas must

be defined manually or by another algorithm, which is the reason why the filter is not generally applicable. However, the filter can provide high-level insights into the gaze signals regarding the stimulus.

Online Filtering of Gaze Signals for Interaction Purposes

In contrast to the filtering of recorded gaze signals, filtering of gaze signals for interaction must happen on the fly. Only a small time overhead to the estimation of the raw gaze samples may be added. Fixation must be recognized in the gaze signal, while new gaze samples are coming in. The incoming gaze samples may extend the fixation or be part of a saccade toward another fixation. Usually, only the current fixation is of interest for interaction purposes. A basic attempt to smooth the gaze signal is to collect samples over time and to average their coordinates [64], toward a reliable POR for every update of the interface. This basic smoothing idea can be performed for a sliding time window, to cover, e. g., the current user fixation but not the whole gaze signal history. By declaring a gaze sample at the current time t as X_t , N as window size, and w as applied weight per sample, we can define the following formula to calculate a weighted average $\hat{X}_t$ of the gaze signal:

$$\hat{X}_t = \sum_{i=0}^{N-1} \frac{w_i}{\sum_j w_j} X_{t-i}. \tag{2.2}$$

The weight is defined by an additional kernel function, which takes the age of each sample into account. Feit et al. [64] name three common kernels to calculate a weight per sample in the window:

- **Linear kernel.** $w_i = 1$. Every sample is weighted equally.
- **Triangular kernel**. $w_i = N - i + 1$. Latest sample is weighted the highest, oldest is weighted with one — the lowest weight.
- **Gaussian kernel**., $w_i = e^{-\frac{(i-1)^2}{2\sigma^2}}$. $\sigma = \sqrt{\frac{-(N-1)^2}{2ln(0.05)}}$ is defined to assign the oldest sample in the window with $i = N - 1$ the weight of 0.05.

The Gaussian kernel is reported [64] to deliver the best results.

Saccade Detection The described filtering using a sliding time window assumes that all gaze samples inside the time window belong to a single fixation. However, the time window might contain samples from two or more fixations just after the user has shifted her attention and performed a saccade. Averaging all samples in the time window would produce a fixation somewhere in the middle of the contained fixations, possibly on a region that the user never had fixated. Therefore, the time window for filtering can be limited to the current fixation by using a spatial threshold [161]. The spatial threshold works similarly to the velocity-threshold identification and separates fixations by the spatial distance of sequential gaze samples. The current fixation is then defined as gaze samples which distance is successively below a spatial threshold, starting from the latest gaze sample.

Outlier Correction Eye-tracking devices may produce single outliers, e. g., when a reflection is shed on the camera or the data transfer suffers from an error. This may produce a single outlying gaze sample, which would prohibit proper filtering of a fixation. Therefore, when going from the latest to the oldest gaze sample within the sliding time window, for each gaze sample that is classified to belong to another fixation than the current one, the previous gaze sample is also checked if it belongs to the current fixation. If the previous gaze sample belongs to the current fixation, the looked-at gaze sample is discarded as an outlier and the consideration is continued at the previous gaze sample [106]. Otherwise, the consideration ends and the weighted average of the collected gaze samples is considered as the current fixation.

2.4 Conclusion

Nowadays, the Web and eye tracking can be brought together in one setup using a standard computer, an Internet connection, and an affordable eye-tracking device. The setup is inexpensive, making the big-scale collection of gaze signals from everyday use a realistic scenario. However, Web page interfaces and their internal structure can be complicated, as the example of a photo carousel in this chapter has shown. The same looks and interactivity of compound elements can be accomplished using manifold methods. Furthermore, additional content can be loaded during Web browsing, such that a dynamic Web page can change its looks and interactivity during the user session. The complications render it potentially difficult to know from the Web page document what is actually displayed to a user, such that the gaze signals can be correctly interpreted within the context of the stimulus. Moreover, eye tracking entails limitations in accuracy and selection, which are especially important to consider for any gaze-based interaction.

We use the term “dynamic” to refer to Web pages that alter their visual appearance after the initial load, whether automatically or triggered by user interaction. It is to notice that the term “dynamic” is used with a slightly different meaning in eye-tracking research [23]. In eye-tracking research, a dynamic stimulus is any stimulus that changes over time, e. g., a video or an interface. Furthermore, a video is called a passive stimulus because a user cannot control how the stimulus changes. Thus, the gaze data can be easily synchronized between participants. In contrast, an interface that can be manipulated by a participant is called an active stimulus. Gaze data collected on an active stimulus is not easy to synchronize across multiple participants. Therefore, a Web page is – regarding eye-tracking terminology – a dynamic and active stimulus. All stimuli that are used for analysis and interaction through eye tracking in this thesis are considered to be dynamic and active.

Part I

Analyzing the Web using Eye Tracking

CHAPTER 3

Background of Analysis with Eye Tracking

In this chapter, we provide the background about how eye tracking can support usability experts with the analysis of Web pages. First, we describe how eye tracking is used in behavioral studies by researchers and companies to estimate the attention of users on Web pages in Section 3.1. Second, we discuss why introspection and interaction recordings of Web pages do not allow aggregating the gaze data from multiple users in Section 3.2. Third, we look at methods that rely on screenshots of Web pages for usability analysis in Section 3.3. Screenshots allow aggregating the gaze data of multiple users, making the analysis potentially more efficient. Fourth, we introduce related work from shot and scene detection in Section 3.4. We later apply shot and scene detection methodology to split and merge video recordings of user sessions on Web pages toward screenshots that each cover one unique interface state.

3.1 Eye Tracking in Usability Analysis

Popular methods like clickstream analysis (e. g., Google Analytics) or A/B testing may be integrated by Web developers and aim at understanding population statistics by answering *how many and how much* is the product usage. In contrast, dedicated usability studies focus on the deep understanding of user behavior to answer *why and how to fix* design issues. In this regard, usability experts aim to understand how interfaces of Web sites stimulate user responses, such as mouse or eye movements, mouse clicks, touch or key pressings, to address shortcomings of these interfaces.

In this thesis, we focus on supporting studies that aim to assess the usability of Web pages by quantifying user attention [28]. Eye tracking has been employed to analyze attention in several application domains, such as medical, sports, commerce, and human-computer interaction studies [146, 82, 54, 153, 43, 57, 14]. Gaze data provides implicit feedback on the interaction behavior of users, which is arguably more intuitive and natural than the conventional indicators [160]. Behavioral studies with eye tracking in usability analysis can provide rich qualitative and quantitative feedback by users. However, an eye-tracking study to represent a target group in browsing a Web page requires about 34 participants [59]. Therefore, we need methods to aggregate gaze data to gather a complete picture of attention and to visualize the collected gaze data of all users in a comprehensive way.

3.1.1 Area of Interest

Commonly, gaze data is mapped onto a representation of the stimulus [60, 23], i. e., an image. Hence, usability experts can assess and correlate which elements of the stimulus have drawn attention and which have been ignored. Usability experts usually define *areas of interest* (AOIs) on a stimulus to aggregate gaze data by multiple users within specific regions. These regions might contain the face of a person, a headline, or an advertisement banner. The estimation of attention in such regions provides a usability expert with implicit user feedback to evaluate a design and compare to different designs. AOIs can be defined before or after the execution of a study. In research, especially psychology, the experimental setup is rather rigid and all stimuli – mostly photos – are known beforehand. In such a scenario, AOIs can be defined on a stimulus before the execution of the study, which allows later for an efficient analysis of the gaze data by the participants of the study. In the case of videos, AOIs can be even animated in their position and size to compensate for the movements of objects in the video. In studies on Web pages, the setup is less rigid, and the stimuli are more complex in their appearance. In the past, HTML tags like `<div>` had been chosen as AOIs before the study execution [20]. But single tags from the Web page document do not necessarily correspond to visual blocks on the Web page. Therefore, usability experts prefer to mark AOIs on the stimulus, in our case the rendered Web page, after the execution of the study, as it is more intuitive and independent of the potentially complex internal structure of a Web page. Therefore, the usability experts need special kinds of recordings and representations of the stimuli, such that they can mark meaningful AOIs.

3.1.2 Visualization of Gaze Data

Besides the marking of AOIs and the computation of eye-tracking-related metrics, another important technique is to visualize the gaze data. We divide the visualization techniques into *point-based* and *AOI-based* approaches [23].

Point-based Visualization of Gaze Data Point-based approaches visualize the gaze data in context with the stimulus. The most popular point-based visualization technique for gaze data is the scanpath. Each fixation is plotted as one circle. The index of each fixation is shown as a number within the respective circle. Longer fixations result in bigger circles. Subsequent fixations are connected through lines, which can be interpreted as saccades. The scanpath of each user is colored uniquely, such that the scanpath of different users can be distinguished. Another popular point-based visualization technique is the heatmap. In a heatmap, the sequence of fixations is discarded and the fixations of all users are aggregated. The density of fixations is taken to estimate the level of attention within a certain region. Regions of high attention are colored differently from regions of low attention. See Figure 3.1 for an example of gaze data visualized as a scanpath and as a heatmap.

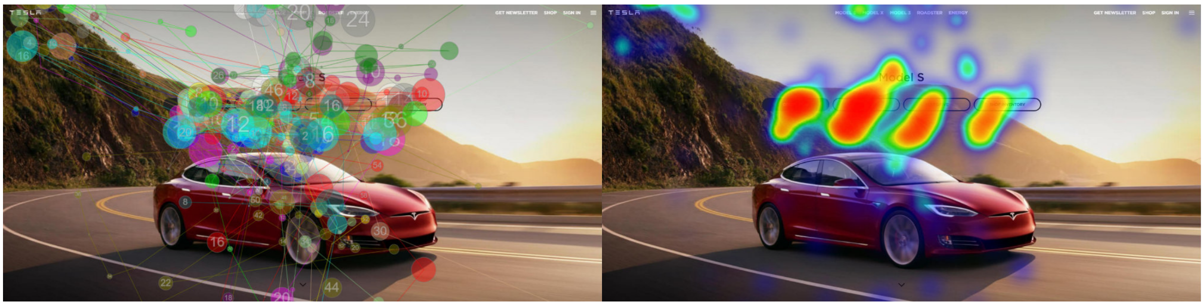


Figure 3.1: Gaze data of 20 users on the home page of Tesla, accessed in November 2018. On the left, the gaze data is visualized as scanpaths. On the right, the gaze data is visualized as a heatmap [133]. In this heatmap, warm colors correspond to high attention and cold colors correspond to low attention. No coloring corresponds to very little or no attention.

AOI-based Visualization of Gaze Data AOI-based approaches can either display the gaze data in the context of the stimulus or be used to process the attention of users on a quantitative level. For the prior, the AOIs can be rendered onto the stimulus, and transitions in the attention of users between the AOIs can be visualized as arrows. For the latter, details about the attention on AOIs like the count of users that looked at the region, the total fixation count, the time until the first fixation, or the aggregated fixation duration can be listed. Furthermore, the transitions between AOIs can be counted across users and plotted in a transition matrix. This allows high-level insights into the overall user behavior, e. g., whether customers commonly looked at the price, the photos, or the ratings of a product before clicking on the button to purchase the product.

Metrics on AOIs, point-based, and AOI-based visualization approaches require an accurate registration of gaze data to the stimulus. A valid interpretation of the gaze data is only possible when the representation of a Web page as stimulus and the gaze data is correctly aligned, both spatially and temporally. However, the accurate representation of a dynamic and active stimulus, i. e., an interface, which is critical to associate gaze data spatially, has received less attention in research and commercial tools.

3.1.3 Challenge of Interface Representation

The interface recording and later representation are critical for the accurate mapping of gaze data with respect to the elements inspected by the users. Traditionally, interfaces were comprised of a small or medium number of static visual elements, e. g., an interface might have consisted of several views which would not change their appearance upon user interaction. Then, each view in the interface would be considered a stimulus that triggered various responses of users, e. g., gaze paths or mouse traces, which might have been recorded and aggregated for each such stimulus. The usability expert who saw gaze paths or mouse traces depicted on a view would understand how users navigate through the interface elements, or which parts were rarely considered [58], and hence, she might suggest to re-order elements according to the needs of users or the priorities of the interface designer.

In modern interfaces, however, users manipulate and respond to passive or active content. For example, rich interfaces may choreograph menus and photo carousels, adjust dynamically to clicking, hovering, and scrolling triggered by a user, or exhibit other complex rendering and visual effects [23]. In this regard, Web developers may use a plenitude of styling attributes like `visibility`, `opacity`, `display`, `transform`, and `z-index` to overlay content and dynamically control the visuals of a Web page. The actual visibility of an element is not only controlled by the attributes of the element itself but also depends on the configuration of other elements on the Web page, which may overlay the element or get themselves hidden by the element. These characteristics of rich interfaces imply that it becomes difficult to align the various interactions of multiple users with well-defined visual stimuli such that the usability expert would easily understand how the users responded to which experiences. This makes the synchronization of the data non-trivial and has been identified as a challenging problem in gaze-based analysis [23].

The most commonly used video and interaction recording method records each user session in a video recording of the screen contents, which is temporally aligned with the recording of this user's interactions [182]. Thus, arbitrary dynamics of an interface are captured. However, due to differences between users' behaviors the recordings of various users are not aligned. For example, every user usually reaches a different viewport position at a certain point in time, caused by scrolling and navigation on a Web page. Hence, gaze data of multiple users cannot be simply overlaid. Therefore, a usability analyst must inspect the video and the interaction recordings of one user at a time to understand all the effects of the design, making the analysis time-consuming and tedious. This is not a scalable solution, especially because usability analysis requires feedback from a considerable number of users [59].

To further understand the challenges of analysis, we initially conducted a small anonymous online survey with usability experts regarding the issues they face while conducting behavioral studies. We sent a request to participate in the survey to a mailing list of usability experts and received 7 issues and 24 ideas for measures on interfaces with dynamic contents. A major challenge that usability experts reported is related to information overload, as they most often use the video and interaction recording method and need to inspect the videos of each user session separately. Moreover, the survey results indicate that a typical task in usability analysis is to find and map interaction recordings (gaze paths or mouse traces) on a particular element, e. g., how users interacted with a photo carousel. Our work tackles both issues: reduce the information overload and make the analysis of particular elements easier.

3.2 Introspection and Interaction Recordings

Understanding the contents of a Web page at a semantic level is a research field since the inception of the Web. The technologies of the Web are designed to allow for a flexible and rich presentation of contents, while the meaning of the contents is often not encoded in a machine-readable way. However, the context of contents, e. g., a date next to the text "day of birth," can hint at the meaning of an input field. The same goes for a combination of multiple HTML elements that can result in a compound element that is treated by users as one unit, e. g., a chat window, a product overview, or a menu. Traditionally, researchers took a Web page document and attempted to parse the HTML code toward a representation of the contents aligned with their semantic model. We call this interpretation of the HTML code *introspection*. In the following, we discuss related work that interprets the contents of a Web page at the level of HTML code, and whether they can support our purpose of mapping gaze data correctly with visual stimuli of a Web page.

Hienert et al. [81] processed a Web page at its textual level and mapped gaze data to the fixated words. Their approach allowed one to estimate which words have been read more often by users and which parts of the text on a Web page were ignored by users. They took a Web page document as input, rendered it with a Web browser, and queried for the coordinates of the words on the rendering of the Web page. Then, they stored for each word the number of fixations that fell onto it. To compensate for slight layout changes and animations, they merged the same words among different users within a context of 50 characters in the text before or after. However, their approach only considered text that was directly available from the DOM tree. Text in other media like images, videos, or canvas was ignored.

Cai et al. [30] described a "Vision-based Page Segmentation" (VIPS) algorithm, which assumed that Web designers separate semantically coherent blocks visually. The algorithm employed a top-down approach by parsing the DOM tree of a Web page. It used heuristics to identify elements that might separate blocks along horizontal or vertical axes. It first split the page into coherent blocks and then merged them back iteratively, until a threshold of visual coherence for each block was reached. See Figure 3.2 for the vision-based segmentation of a Yahoo Shopping Web page. The segmentation algorithm did not consider CSS styling, the visual impact of images, or modern HTML tags like `<video>`. There had been attempts to improve the algorithm by adding more rules to the heuristics [6], yet overlapping elements and dynamics Web pages are still not considered by the algorithm.

Kumar et al. [109] presented "Webzeitgeist," a framework that allowed for querying visual properties of Web pages and elements represented in DOM and CSS structures. The framework consisted of a database that stored the DOM tree, CSS attributes, a rendered image, and computer-vision features of each Web page. While the database provided a rendering of each Web page, the contents of the DOM tree itself were not checked for visibility in the rendered image. Furthermore, dynamics that occurred during interaction by users were explicitly excluded from the crawling process. The same goes for follow-up works about crawling Android app interfaces [47, 46, 198, 120]. These works employed both screenshots and view hierarchies of Android apps to retrieve insights into the interface designs. All of these works have in common that they did not consider scrolling of the interfaces, they did not consider layered visibility configurations, and they relied on interface state detection through an API.

Viewport responsiveness describes a technique of fitting the contents of a Web page automatically into the available horizontal space of a viewport by changing the layout of the Web page accordingly. This

[1] `http://www.cad.zju.edu.cn/home/dengcai/VIPS/VIPS.jpg`, accessed on 23rd May 2020.

makes a Web page adapt to different screen sizes, like desktop monitors, tablets, and smartphones. Lamberti et al. [114] described a method to aggregate and display the intensity of user attention on viewport-responsive Web pages with element-aware heatmaps. They injected JavaScript code into a Web page to retrieve information about elements of interest and stored data accordingly. Before recording interaction on a Web page, mouse interaction was simulated and structural changes on the Web page were rendered in the viewport. The resulting visuals were stored on a server-side recording tool. During the interaction recording, dynamic changes in the DOM tree were gathered at user events or via frequent

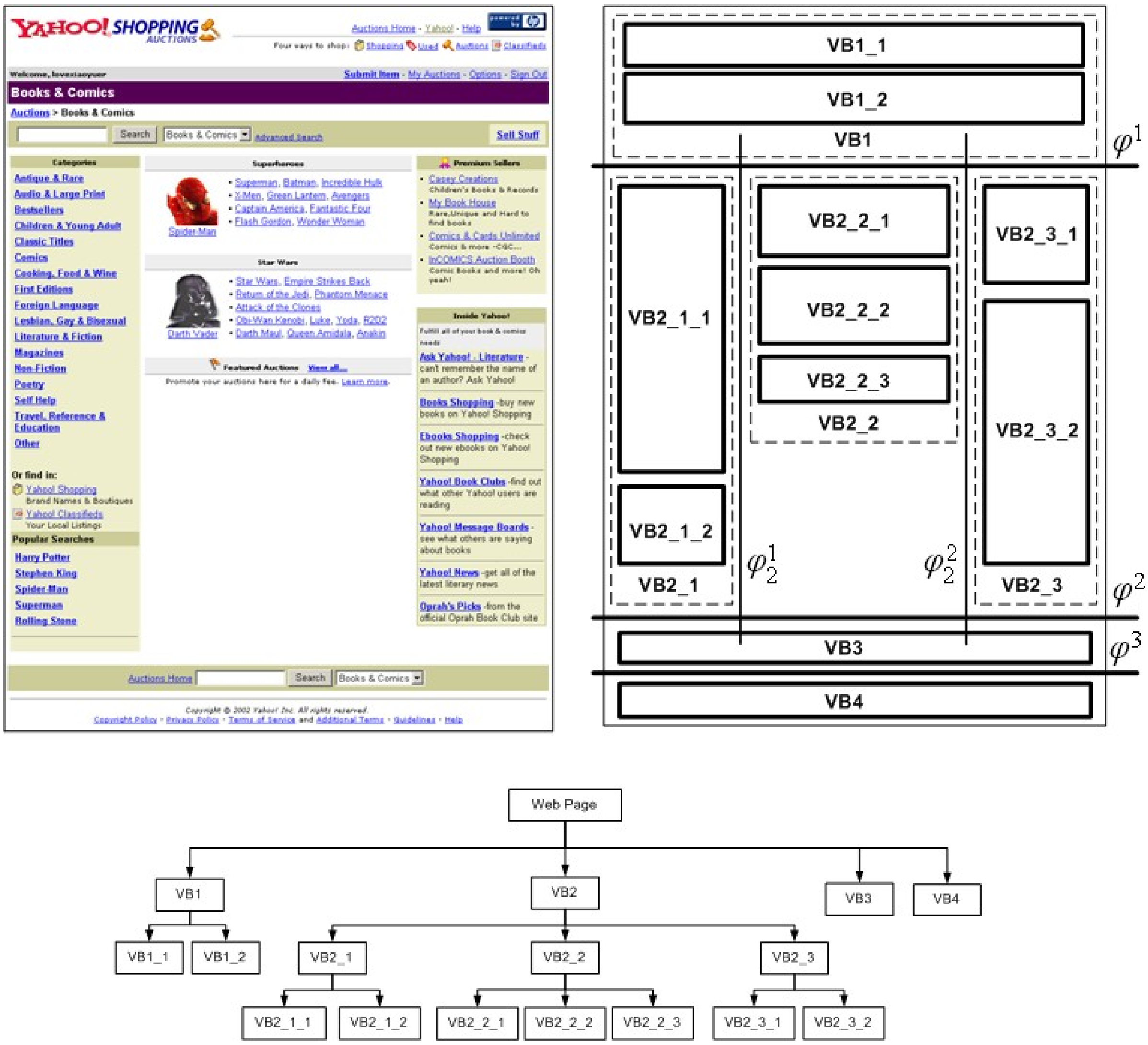


Figure 3.2: Vision-based page segmentation of a Yahoo Shopping Web page with the VIPS [30] algorithm.[1]The segmentation consists of visual blocks (VB) and separators (φ). The algorithm detected four first-level visual blocks VB1 to VB4, which were further divided into sub-content structures.

polling of the DOM tree. However, neither did the authors clarify how the changes in the visibility of elements were recognized, nor did their evaluation cover Web pages with changing contents.

Burg et al. [27] proposed a tool to support Web developers in understanding visual changes of elements on Web pages. The tools allowed for automatically tracking changes of a selected element and it revealed the code snippets that caused the visual change. A prior selection of the AOIs was required, which is not feasible for a behavioral study on a Web site. The prior selection of elements or definitions of states is a critical task, as a usability expert might not anticipate which parts of the interface will cause problems during the user interaction.

Another approach to capture the interaction with Web pages is the recording of the user actions at DOM level, called *session replay*. One can record the DOM tree on the client-side, e. g., with the Mozilla Firefox Web browser and the WebReplay[2] functionality. This technique allows for recording the DOM tree on arbitrary Web sites; however, the technique is designed to support Web developers in analyzing the behavior of individual elements when a single user interacts with them. Alternatively, there are frameworks like LogRocket[3] or mouseflow[4] which require a script to be integrated into a Web page, yet can record the sessions of every user who accesses the Web site. However, the integration of a script limits the approach to Web sites to which a usability expert has administrative access. That method faces major issues when iframes are integrated or when changes are caused by CSS animations, as those do not trigger a mutation of the DOM tree. Furthermore, session replays do not automatically detect meaningful states of a Web page and need manual intervention by usability experts.

Considering the related work, we argue that the sole introspection of the DOM tree is not feasible for a usability analysis with eye tracking. It can be (i) difficult to keep up-to-date with the interface technologies, (ii) difficult to judge the effect of structural changes on the visuals of an interface, (iii) result in a plentitude of recorded states, which hinder a meaningful aggregation of the interaction recordings from multiple users, and (iv) is limited to interfaces that allow for introspection. The introspection method has been applied in the context of the Web and Android apps, yet it is not suitable for behavioral studies of dynamic interfaces with test participants. Most approaches ignore the dynamics on Web pages and solely rely on the initial state of the interface. Recent technologies like CSS animations are ignored.

3.3 Screenshot and Interaction Recordings

In recent years, the means for defining a Web page have become more and more complex with the introduction of the living standards of HTML5[5] and CSS 3.[6] Web developers can choose between various

[2] `https://developer.mozilla.org/en-US/docs/Mozilla/Projects/WebReplay`

[3] `https://logrocket.com`

[4] `https://mouseflow.com`

[5] `https://html.spec.whatwg.org`

[6] `https://www.w3.org/Style/CSS/current-work.en.html`

ways of implementation to achieve a specific look and behavior on a Web page. The multitude of ways to implement content, design, and interactivity makes it difficult to cover and understand all aspects of a Web page with an introspection-based recording. However, we are foremost interested in the experience a user has on a Web page, regardless of the underlying implementation. Thus, what is most important for a usability analysis is what has been displayed to the user as a stimulus. Therefore, researchers have started to interpret a Web page based on pixels rendered by the Web browser.

Researchers in the field of cross-browser Web-page testing proposed methods for testing Web pages across Web browsers that make use of the HTML code and the rendering of a Web page [37, 127]. These methods detect visual differences between Web page renderings based on the pixel data, e. g., by comparing color histograms. The works either ignored dynamic Web pages or required a prior selection of changing content, which was then handled with heuristic approaches for cross-browser testing.

Methods for reverse engineering of interfaces attempted to recognize elements through interpretation of pixels and template matching alone [50, 194, 13]. However, the methods did not retrieve changes in the interface states, expected interfaces to originate from similar toolkits, required layouts without overlaps, and did usually not account for scrolling.

Cormier et al. [42] performed semantic labeling of regions on Web pages to improve accessibility through screen readers. They detected edges in the rendering of Web pages. Then, they searched for the segmentation which was best supported by the detected edges. Afterward, they attempted to classify each region with regard to its functionality or contained information. Analogously to VIPS, they assumed regions of interest within a layout to be rectangular and not to overlap each other. Their approach was also not extended to handle dynamic interfaces.

Simko and Vrba [170] presented a method to support usability analysis of app-prototype interfaces on a smartphone with eye tracking. In their method, a usability expert defined *reoccurring scenes* on the interface before the study. Each reoccurring scene was represented with one screenshot. During the study, they recorded the interactions with the app-prototype interface on a smartphone using a camcorder. For each frame from the video recording of the camcorder, they cropped the screen content and assigned the frame to one of the predefined reoccurring scenes. They employed scale-invariant feature transform (SIFT) and structural similarity (SSIM) features to map the frames to the reoccurring scenes and heuristically set a threshold for the mapping process. Their method did not discover scenes automatically but expected all scenes to be predefined by a usability expert. Scrolling prohibited the aggregation of interaction recordings because at every scroll offset the interface was treated as a unique scene. Furthermore, Simko and Vrba only evaluated a banking app prototype interface with very little dynamics in their experiment.

Kurzhals et al. [113] presented a visual analytics method for the analysis of gaze data from head-mounted eye-tracking devices based on automatic image comparison and clustering. First, they cropped thumbnails of the foveated region from the video recording at every fixation. Second, they aggregated the thumbnails of subsequent video frames into a segment until the similarity fell below a preset threshold.

Third, they computed a similarity matrix between all segments across the recordings using SIFT-feature histograms and color histograms. Fourth, they clustered the segments according to their similarity using unsupervised spectral clustering. Then, analysts could label the segment clusters semantically and might combine them toward AOIs. Their method was designed to work on the video recording of physical objects in a real environment. Similar works employed deep neural networks to identify objects in the foveated region of an ego-centric video stream [15] to automatically mark AOIs in the video stream. There have been also similar attempts to visualize data from table-mounted eye-tracking devices on a video as a stimulus by Kurzhals et al. [112, 111]. All these methods do not include the surroundings of fixations in the later representation. Therefore, a Web page would be not captured as a whole. However, a usability expert needs to work on a representation of a Web page that looks similar to what has been experienced by the users. Only then, a usability expert can interpret the attention in the context of all content that has been visible to the user.

There exist various tools for conducting usability studies with eye tracking. Many tools utilize a Webcam to estimate eye gaze and create a heatmap of user attention on screenshots of a Web page, like sticky.ai [1], eyezage [68], CoolTool [41], or realeye [148]. Tobii Pro Studio [182], a tool provided by the market leader of dedicated eye-tracking devices Tobii AB, also offers gaze mapping on a recording of a user session on a Web site. Their software either visualizes gaze data on a large screenshot of each Web page or a video recording of each user session. See Figure 3.3 for a photo of a usability expert using Tobii Pro Lab to analyze gaze data. However, Tobii explicitly recommends watching the interaction with dynamic elements on a video recording instead of the screenshots, because the dynamics are not covered in the screenshot.[7]

There also exist tools that are solely specialized in taking screenshots of Web pages without the primary intention to perform a behavioral study. For example, FireShot [178] allows for capturing an entire, potentially scrollable, Web page in a single image. First, viewport-relative elements are transformed to an absolute position on the Web page. Then, the tool stitches screenshots of the viewport, while scrolling down the Web page automatically. However, gaze data is not automatically mapped and must be manually transferred from the space of the viewport to the space of the created image.

The above-mentioned methods that interpret the rendering of a Web page either completely ignored dynamics or handled the dynamics only partially. Most methods that handled dynamics relied on a prior selection of dynamic elements or predefined states of the interface.

[7] *"...if the participants access a drop-down menu on the website the viewing patterns on that menu will be recorded and aggregated on the screenshot of the webpage, but the menu itself won't be visible on that screenshot."* `https://www.tobiipro.com/learn-and-support/learn/steps-in-an-eye-tracking-study/data/Aggregating-eye-tracking-data-across-several-participants-in-web-recordings`, accessed on 20th March 2020.

[8] `https://tobii.imagevault.media/publishedmedia/vu06e2z047t0kpxjwsn9/Tobii-Pro-mobile-testing-analysis-gazeplot.jpg`, accessed on 23rd May 2020.

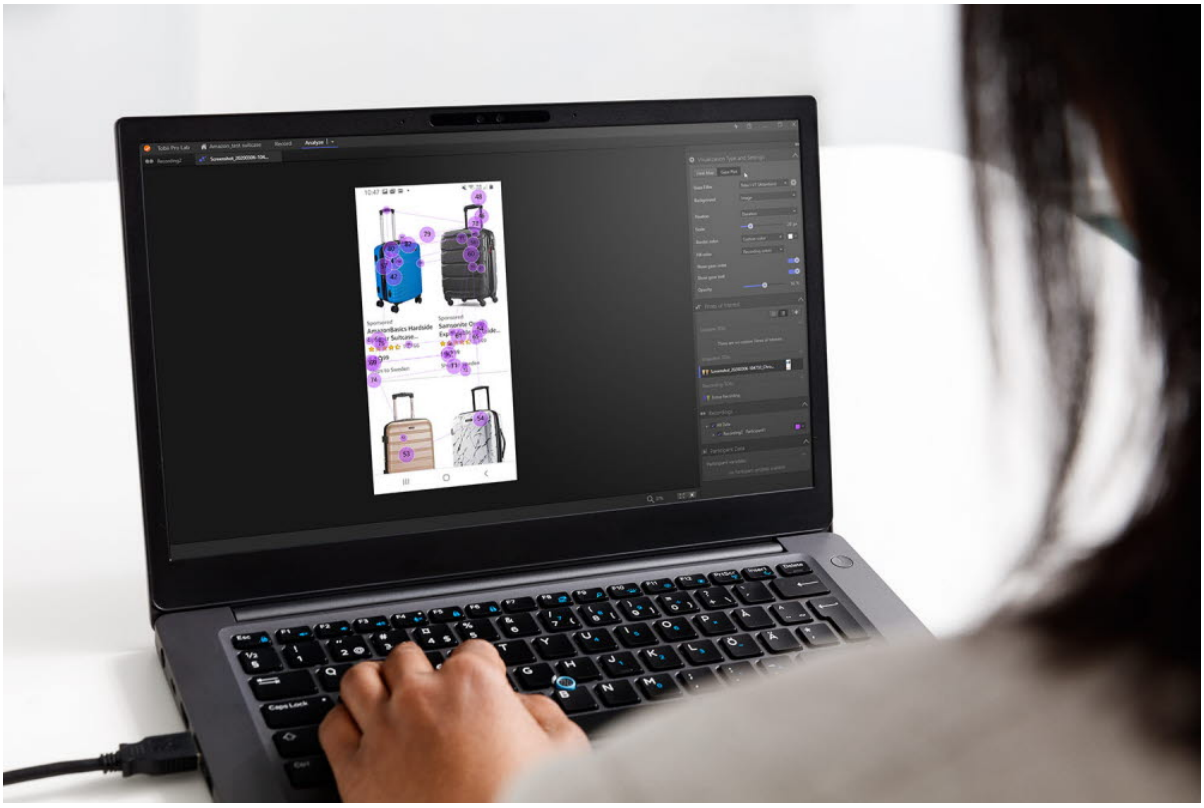


Figure 3.3: Photo of a usability expert using Tobii Pro Lab to inspect a scanpath on a product overview.[8]

3.4 Detection of Shots and Scenes in Video Recordings

Scene detection is a well-established field of research that aims to cluster different shots within a video or movie into scenes of coherent environment, time, actors, and story [141]. A video can be cut into a series of scenes, which itself can be separated into shots. A shot is the recording by one camera from a specific position and angle. A scene is composed based on a coherent environment and set of actors. There are cases in which two or more shots are interwoven, e. g., through frequent switches between shots that serve the dramatic purpose. Thus, methods for automatic shot detection and clustering of shots and scenes constitute research relevant for addressing the problem of stimuli discovery.

While Simko and Vrba [170] have applied scene detection methods for visual stimuli discovery, as described above, they did not explore the feasibility of different computer-vision features, and they manually optimized thresholds of feature values to classify frames from a video recording of the interface into before-known visual stimuli. But they did not explore the feasibility of different computer-vision features and they manually optimized thresholds of feature values to classify frames from a video recording of the interface into before-known visual stimuli. Instead, we want to investigate which features from shot and scene detection are useful for our purpose of finding visually coherent states of an interface.

Popular features for shot and scene detection like face detection, audio processing, and subtitle analysis [48] do not apply to our use-case. However, various computer-vision features that are used to detect changes in lighting, setting, and environment in videos might signify visual changes for our purpose. Multiple approaches that successfully employed color histograms [184], edge-features [195], optical-flow [77], and SIFT-features [142] in the purpose of shot and scene detection in videos are promising candidates for our purpose.

3.5 Conclusion

Knowledge about the attention of users on Web pages can help designers and developers in improving the reachability of information or make advertisements more effective. However, behavioral studies with eye tracking in the context of Web page usability require a high effort in preparation and later analysis. Especially modern Web pages with dynamic contents are challenging because gaze data cannot be simply aggregated across users with the current methods. Therefore, eye tracking is far less used as it could be, and the Web is not as usable as it might be.

CHAPTER 4

Recording of Scrolling Dynamics

In this chapter, we present a method to improve the recording and representation of Web pages with dynamics that are introduced through scrolling. Scrolling is an integral interaction with interfaces. The scrollbar was already introduced in 1981 with the Xerox Star operating system.[1] The popularity of scrolling also applies to the Web. In the past, Web pages were designed with specific screen sizes and resolutions in mind. If a viewport was not sufficiently large to show the entire layout, scrolling allowed to still browse the entire Web page [92]. Nowadays, viewport-responsive Web page layouts are designed to fit the contents horizontally into the available space of a viewport, while putting the overflowing contents into the vertical space. Thus, nearly all Web pages require users to scroll vertically. Social media Web sites like Facebook, Reddit, or Twitter go one step further and implement “infinite” vertical scrolling. As soon as a user reaches a certain scroll position, further posts are loaded and the user can continue scrolling. The Web page feels like an endless flow of postings, without any interruption through the loading of contents or manual page switches.

For usability analysis, one popular method is to create a *virtual screenshot* that comprises the complete Web page, an area that is possibly larger than what can be displayed to the user at once in the viewport, and to map all gaze data onto this virtual screenshot [182, 168]. However, the virtual screenshot method suffers from *inaccurate* gaze mapping on elements that are not affected by scrolling, on which gaze data is then misinterpreted by usability analysts. For example, gaze data at an advertisement banner that is shown on a fixed viewport position is misplaced if a user scrolls the page. The banner is rendered once on the virtual screenshot at its initial position on the Web page. When a user scrolls the viewport down such that a lower region of the Web page becomes visible and the topmost region disappears, the banner may stay in the same place relative to the viewport rendered on the screen. But the virtual screenshot method transforms *all* gaze data from the screen space to the page space. The gaze data is then registered somewhere below the initial rendering of the banner. Therefore, the attention of the users cannot be appropriately associated with the banner and a usability analyst underestimates the effect of the banner on the user experience.

We propose an *enhanced representation* method for Web pages to tackle the above-mentioned challenges of accuracy. We make use of structural information from the Web page document to identify fixed

[1] `https://www.smithjournal.com.au/blogs/history/3897-the-history-of-the-scroll-bar`, accessed on 23rd May 2020

elements and to combine them coherently with scrollable content, to portray a representation of the Web page that is simplifying the actual user interaction, yet is close enough to allow insights by a usability expert. Our method tightly integrates the eye-tracking environment with the Web browser, as it extracts information about the Web page elements from the Web page document and it considers pixel patterns rendered in the viewport of the Web browser.

First, we explain the problems of the virtual screenshot method in Section 4.1. Second, we describe our enhanced representation method in Section 4.2. Third, we evaluate the enhanced representation method against the video recording method concerning the tasks of a usability expert in a lab study with ten participants in Section 4.3.

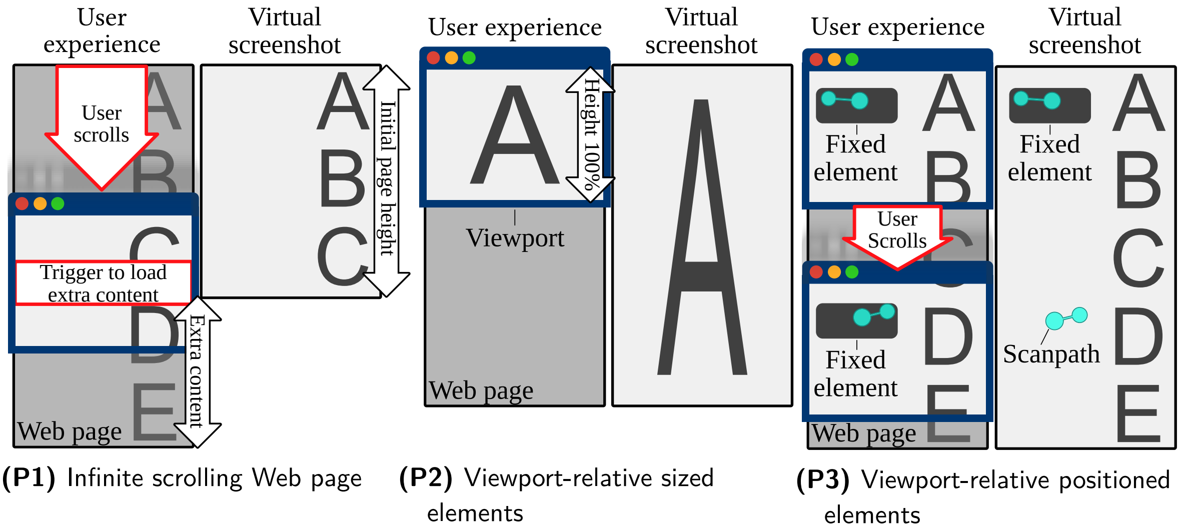
4.1 Problems of the Virtual Screenshot Method

To analyze usability with eye tracking, gaze data is visualized as an overlay on the rendering of the interface that the users experienced during their interaction with the Web page. Modern Web page usability analysis tools extend this approach for Web pages that are larger than what can be seen in a viewport by a user at a time. They assemble a virtual screenshot by capturing pixels from an off-screen viewport that is large enough to render a complete Web page at once. Gaze data, however, is collected not relative to the off-screen viewport, but relative to the viewport of the user, which is usually too small to host a complete Web page at once. Therefore, the coordinates of the gaze data must be mapped between these two reference systems. There can be no mapping that can retain the experiences of the users, as it relates to their eye movements and the actual renderings in the observed viewports. The problems are given in the following and illustrated in Figure 4.1:

- **(P1) Infinite scrolling Web pages.** The virtual screenshot method implies the assumption that at one point in time the complete Web page is loaded by the Web browser. However, this is not true or at least it is not trivial to determine the point-in-time when the complete Web page is loaded. As of today, there are infinite scrolling Web pages that use asynchronous server communication to request further Web page contents when a user reaches a certain vertical scroll offset. A user scrolls down the Web page and may reach a trigger, which causes the Web browser to request more news posts or products. Before a user reaches the bottom of the Web page, the retrieved contents are appended to the bottom of the Web page and the user can continue with scrolling, in the best case without any interruption. The scrolling interaction dynamically expands the height of the Web page during a user session. *Example: Social media news feeds.*
- **(P2) Viewport-relative sized elements.** Elements on Web pages can be sized in relation to the viewport. The height and width of elements are then provided in percentage values. When a virtual screenshot is taken, the off-screen viewport for capturing the visual content is set to the size of the complete Web page, including the contents beyond the viewport of a user. Relatively scaled

elements are scaled accordingly to the off-screen viewport that covers the complete Web page, to capture the virtual screenshot. For example, an image with a width and height of 100% at the top position of the Web page would cover wrongly the complete virtual screenshot instead of the initial viewport of a user. *Example: A welcome message filling the entire screen with further details beneath, which a user can reach through scrolling.*

- **(P3) Viewport-relative positioned elements.** Elements on Web pages can be defined to remain at a certain screen position, so they are positioned viewport-relative and are not affected by scrolling. We refer to this property as position-*fixed*. In the virtual screenshot, the gaze data on fixed elements are wrongly stored when a user scrolls the Web page. The virtual screenshot method assumes the page contents to move accordingly when a user scrolls the Web page and transforms all gaze data to the page space. Any gaze data on fixed elements is therefore wrongly transformed and cannot be associated with the viewed content in the later analysis. *Example: A fixed advertisement banner or a navigation bar at the top of the viewport.*

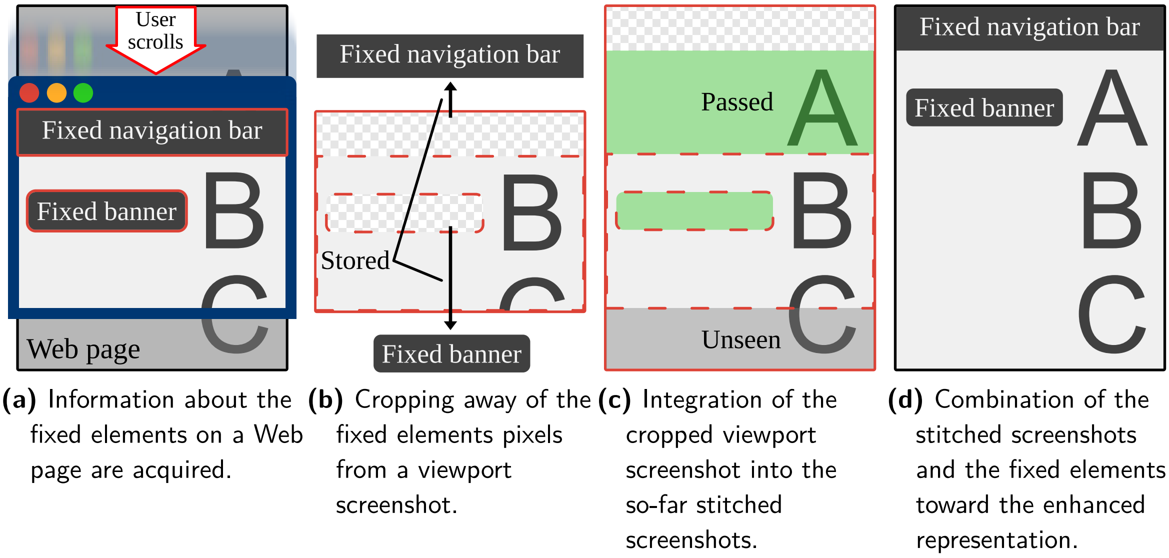


(P1) Infinite scrolling Web page **(P2)** Viewport-relative sized elements **(P3)** Viewport-relative positioned elements

Figure 4.1: The problems P1 to P3 of the virtual screenshot. The actual experience of the users on the Web page is visualized on the left. The corresponding virtual screenshot is shown on the right.

4.2 Enhanced Representation Method

In this section, we propose the enhanced representation method of a Web page, which is a composed image of viewport screenshots from a user session, while keeping the position and size of page elements consistent as per the actual experience perceived by a user. To achieve this, we first identify the fixed elements in a viewport with the approach described in step 1. Then we crop the identified fixed elements from the viewport screenshots and combine the viewport screenshots for a consistent Web page representation, as discussed in step 2. Finally, we perform a composition of the viewport screenshots with the fixed elements in step 3.



(a) Information about the fixed elements on a Web page are acquired.

(b) Cropping away of the fixed elements pixels from a viewport screenshot.

(c) Integration of the cropped viewport screenshot into the so-far stitched screenshots.

(d) Combination of the stitched screenshots and the fixed elements toward the enhanced representation.

Figure 4.2: The enhanced representation of a Web page for attention analysis with eye tracking.

Step 1: Extraction of Fixed Elements Fixed elements on a Web page are viewport-relative elements that stay visually in the same position within the viewport while a user scrolls the Web page. An example of a fixed banner and a fixed navigation bar within a viewport is shown in Figure 4.2a. For the identification of these viewport-relative positioned elements in a Web page document, we search for elements with a style position attribute value set to `fixed` [9]. The style position attribute is the standard for creating a viewport-relative positioned element and is widely used in the modern Web. To acquire the information about the fixed elements, we inject JavaScript code into the loaded Web page. The code contains a function to search recursively for fixed elements in the DOM tree and returns their position, size, and visibility via a callback to the Web browser. Furthermore, the DOM path, the path from the root node to the fixed element in the DOM tree, is used for explicit identification of the fixed elements. The identification can be used to incorporate dynamic updates of a fixed element, e. g., when a fixed element moves, resizes, or changes its visibility. When a formerly identified fixed element is no more contained in the returned list of identified fixed elements, its visibility is set to false. Gaze data that is registered within the boundaries of a visible fixed element is associated with the extracted element and stored alongside for further processing.

Step 2: Viewport Screenshot Stitching A simple stitching approach of appending viewport screenshots would replicate the fixed elements on every captured scroll offset. Therefore, we use the extracted information about the fixed elements to crop the pixel data within the boundaries of the fixed elements from each viewport screenshot, see Figure 4.2b. The cropped screenshot is transformed accordingly to the Web page scrolling and stitched together with the previously stitched screenshot of the Web page, as shown in Figure 4.2c. Areas that have been cropped away are left transparent, so they can be filled with information from other viewport screenshots at different scroll offsets.

Step 3: Composition of Stitched Viewport Screenshots and Fixed Elements The procedure described in step 1 and step 2 is triggered by `window.onload` and `window.onscroll` events and is additionally executed every 200 ms, in case the fixed elements are changed in their properties by scripts while the user is not scrolling the Web page. A lower sampling frequency has produced visual gaps in a dry run, whereas higher sampling frequencies do not add more value but only increase the computational load. Once a user session on the Web page is ended, e. g., at a change of the URL, a composition of the stitched screenshot and the collected information about the fixed elements is created. The challenge is then to place the fixed elements on the stitched screenshot as close as possible to the user experience. We employ the heuristics to align a fixed element to either the top or the bottom of the stitched screenshot, depending on which vertical half-space the element was displayed in the viewport. See Figure 4.2d for the composition of the stitched screenshot and the fixed elements from the example. Both fixed elements in the figure (navigation bar and banner) are originally placed within the upper half of the viewport and therefore aligned at the top in the final composition. Any gaze data on these fixed elements would be displayed in the visualization for analysis accordingly to the final position of these elements. Notably, for saccade analysis, we must ensure the correct fixation sequence with respect to the fixed and scrollable elements. The associated gaze data provides this information, and we propose the analysis tool to show this information by appropriate means of visualization. The method of enhanced representation can solve the problems of the virtual screenshot method as described in the following:

- The frequent stitching of viewport screenshots does include the dynamic additions on infinite scrolling Web pages, solving (P1).
- The screenshots of the user viewport guarantee that Web page elements are displayed on the same scale as they have been presented to the user, solving (P2).
- The extraction and cropping of fixed elements and their association with registered gaze data allows for accurate visualization and metrics calculation, solving (P3).

An open-source prototype based on the Qt WebEngine example using mouse data instead of gaze data developed by Hanadi Tamimi is available on GitHub.[2] The analytic tool as used in the evaluation cannot be published due to the commercial interests of the EYEVIDO GmbH.

[2] `https://www.github.com/Institute-Web-Science-and-Technologies/MTB`

4.3 Evaluation of the Enhanced Representation Method

With the enhanced representation method, we overcome the limitations of the virtual screenshot method. The enhanced representation method allows the accurate mapping of gaze data on the Web page contents for infinite scrolling pages and viewport-relative scaled elements because these mechanisms are reproduced identically to the actual user experience. We argue that it is not meaningful to evaluate the performance in analysis on virtual screenshots versus the enhanced representation. The scrollable contents of a Web page are either pixel-perfect similarly represented in both methods, or wrong on a virtual screenshot in case of an infinite scrolling Web page (P1) and a Web page with viewport-relative sized elements (P2). However, we need to assess if the mapping on the enhanced representation is perceived correctly for viewport-relative positioned elements (P3). Users have inspected the fixed elements on different scroll offsets on a Web page, which alter the contents around the fixed elements. The composition step in the enhanced representation, however, gathers fixed elements and maps associated gaze data on the top or bottom of the Web page. Similar to the P1 and P2, the virtual screenshot fails in a correct representation of viewport-relative positioned elements if a user scrolls. Hence, we compare the effectiveness of using the enhanced representation against the most often used method of video and interaction recording, which provides individual user experiences and is supported in commercial tools for the analysis of gaze data on dynamic Web pages. One major limitation of the video-based approach is the effort required by the usability experts to analyze individual videos, hence we aim to evaluate whether our method would reduce the workload on usability experts while being equally effective. In summary, we came up with the following two hypotheses to assess the analysis on fixed elements with the enhanced representation method:

- **(H1) Accuracy:** The enhanced representation method allows usability experts the analysis of gaze data on fixed elements as accurately as with a video recording.
- **(H2) Scalability:** For analyzing gaze data from multiple users, the enhanced representation method would be more efficient than a video recording.

Methodology

We report the outcomes of a lab study to verify the hypotheses. The goal of the lab study has been to quantify the effectiveness and efficiency of the two methods in supporting analysts in assessing user attention on Web pages. We recruited ten participants (two females and eight males, age = 25.4 ± 1.77 years; mean ± standard deviation), who have been trained in an analysis tool and referred to as *analysts* in the following. The analysts were randomly divided into two static groups, called A and B. The groups performed the analysis either on basis of the enhanced representation of the Web page or on the video recordings of the user sessions. The utilization of the enhanced representation or the video recordings are the independent variable of the study, see Table 4.1 for the counter-balanced distribution of the dataset among the groups. The analysts were asked to draw AOIs on the fixed elements of the chosen Web pages. The analysis tool calculated standard measures of eye tracking [153, 29] in the AOIs, of which the following were reported by the analysts: Time to first fixation (TTFF) and count of total fixations (TF).

Table 4.1: Dataset distribution among analysts. For each Web page analysis, the analysts worked either on the enhanced representation (E) or on the video recordings (V).

	Digg.com	Jimdo.com	Yelp.com	CC
Group A	V	E	V	E
Group B	E	V	E	V

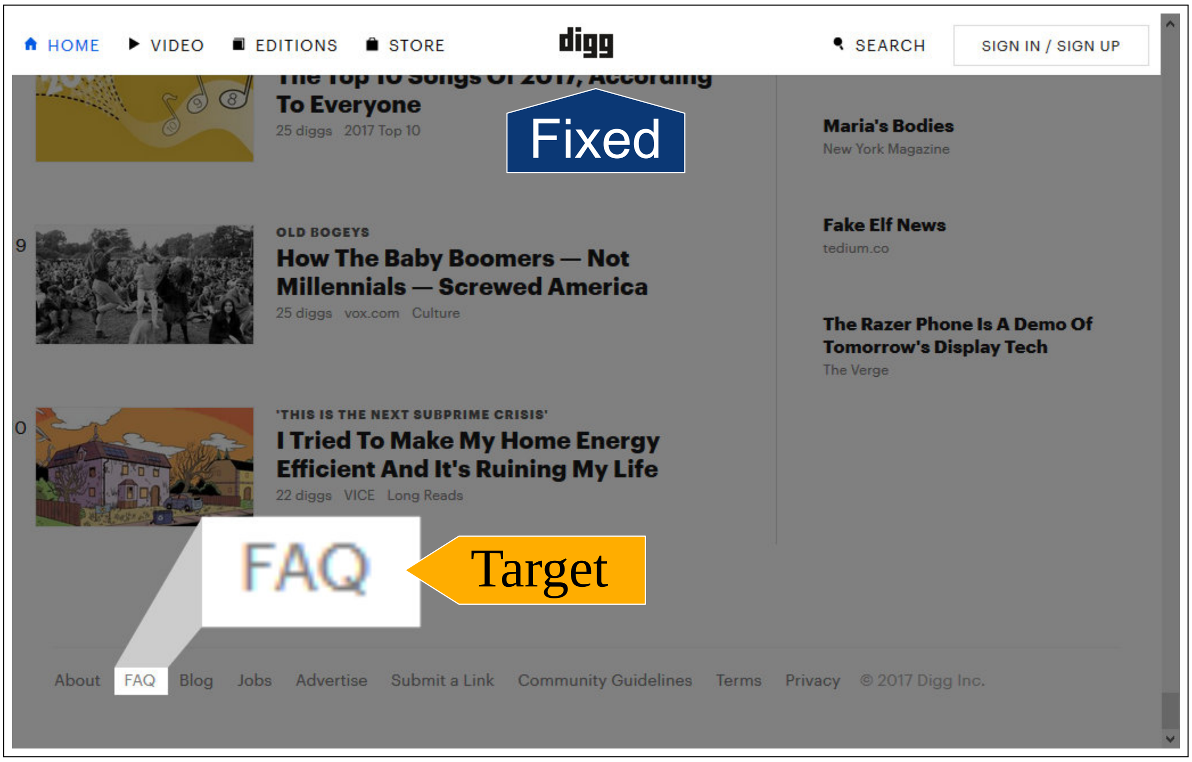
The dependent variables of the study are the AOI-based measurements of the TTFF and TF metrics, for which we created a ground truth, the completion times, the outcome of a NASA task load index (NASA-TLX) [73] with scores from one to seven, and subjective estimation by the analyst about the difficulty for the user to fulfill the task. It was formulated as the following question: *"It was challenging for the participant to solve the task."* The analysts answered the question on a five-step scale from *absolute disagreement* to *absolute agreement*.

Apparatus The methods of enhanced representation and video recording have been implemented in the EYEVIDO[3] analysis tool. The gaze data can be visualized and analyzed with the tool as scanpath in the same manner for both methods (straight line as visualization for saccades, enumerated circles for fixations, a timeline to play and skip gaze data, and the video recording, zooming, and scrolling tools for the enhanced representation, AOI marking utilities). A timeline bar can be used to play and skip through the gaze data and the video recording. However, the enhanced representation allows – in contrast to the video recording – overlaying scanpaths of multiple users at once.

Dataset The dataset for the analysts to work on was recorded with two users on four Web pages with a viewport of 1,920 × 984 pixels. We chose the following Web pages because of their visiting ranks noted on *"moz.com/top500"* and their utilization of fixed elements: (a) *"Digg.com"* (see Figure 4.3), (b) *"Jimdo.com"* (see Figure 4.4), (c) *"Yelp.com"* (see Figure 4.5),[4] and (d) *"Creativecommons.org"* (CC, see Figure 4.6).[5] The tasks on the Web pages for the users have been designed to let them explore the Web pages in their complete height and to find and click target links. The analysts were later asked to assess the attention on the fixed elements, only, which is a straightforward task for the video recording. If the analysts had to assess attention on page-relative contents, they would have had to adapt the position

[3] https://www.eyevido.de/cloud-eye-tracking/software-eyevido-lab
[4] https://www.yelp.com/search?find_loc=koblenz
[5] https://www.creativecommons.org/licenses/by/2.0

and size of the AOIs in the viewport manually to the individual scrolling behavior of each user. See the corresponding figure captions for more details about the tasks for the users (“Target”) and analysts (“Fixed”). We used a Visual Interaction myGaze n 30 Hz remote eye-tracking device for recording the dataset. Moreover, we implemented the recording of the enhanced representation in the EYEVIDO recording tool. A video recording of each user session has been performed using the Open Broadcaster Software in parallel.[6] The outcome is a dataset with a total of four enhanced representations, eight video recordings, eight interaction recordings, and the corresponding gaze data. Additionally, we have asked the users to answer *“How challenging was the task for you?”* on a five-step scale from *very easy* to *very difficult*, after viewing each Web page.



HOME VIDEO EDITIONS STORE digg SEARCH SIGN IN / SIGN UP

The Top 10 Songs Of 2017, According To Everyone
25 diggs 2017 Top 10

Maria's Bodies
New York Magazine

9 OLD BOGEYS
How The Baby Boomers — Not Millennials — Screwed America
25 diggs vox.com Culture

Fake Elf News
tedium.co

The Razer Phone Is A Demo Of Tomorrow's Display Tech
The Verge

0 'THIS IS THE NEXT SUBPRIME CRISIS'
I Tried To Make My Home Energy Efficient And It's Ruining My Life
22 diggs VICE Long Reads

About FAQ Blog Jobs Advertise Submit a Link Community Guidelines Terms Privacy © 2017 Digg Inc.

Figure 4.3: Digg: The Web page contains a fixed navigation bar on the top. The users were advised to find and click the hyperlink to the frequently asked questions (FAQ). The first user spent 31 s on the Web page, the second user spent 33 s on the Web page.

Results

First, we present the results of the accuracy for both methods. Then, we provide details about the task completion times and the feedback from the NASA-TLX questionnaire.

[6] `https://www.obsproject.com`

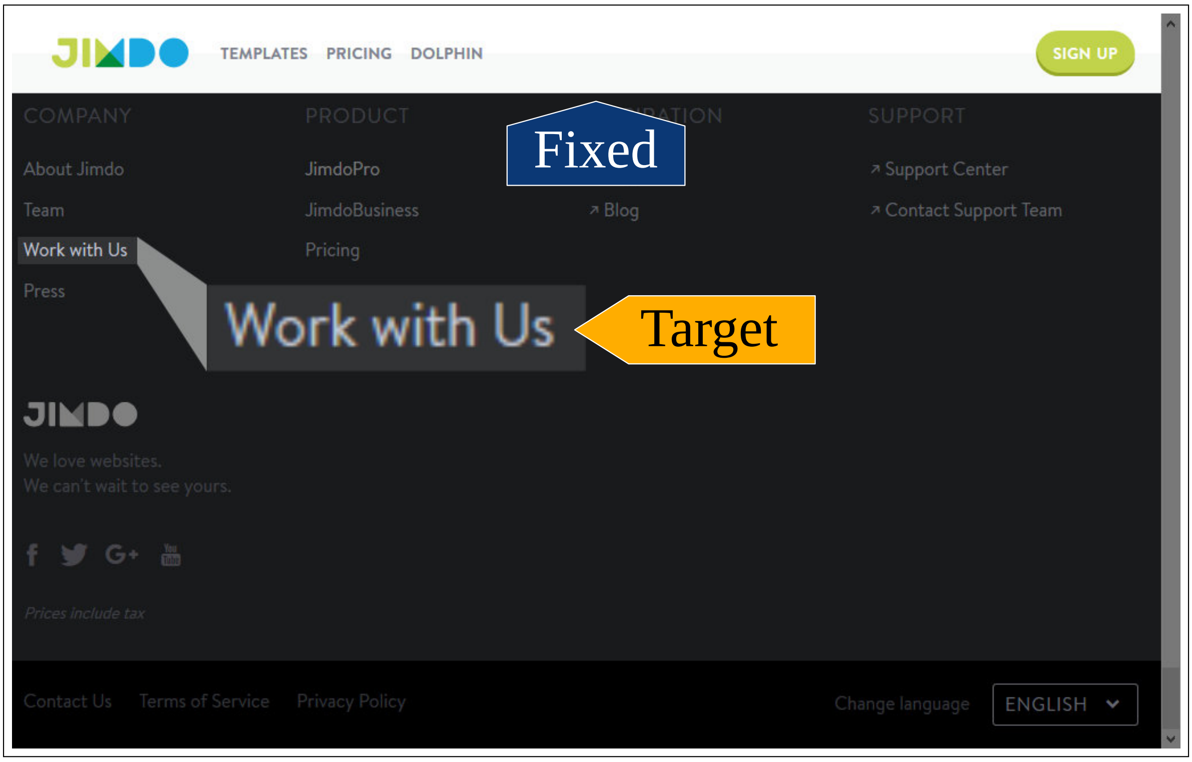
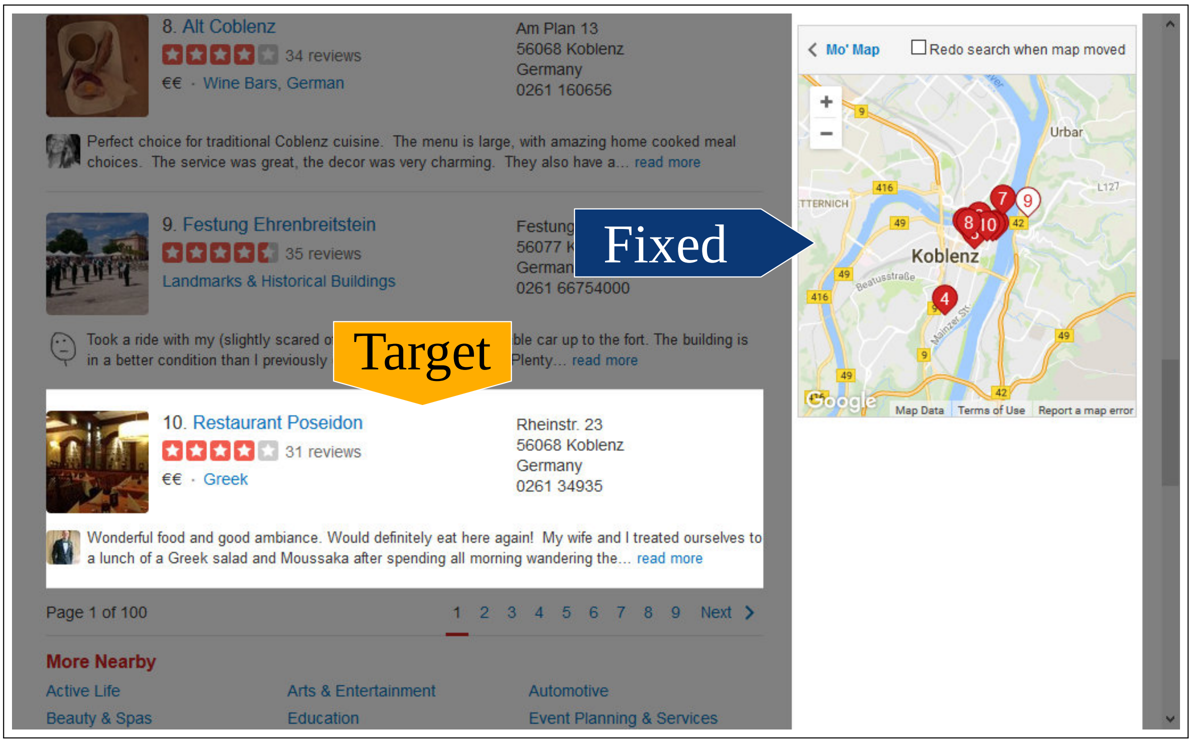


Figure 4.4: Jimdo: A fixed header is shown after a certain amount of scrolling. The users had to search for the *"work with us"* hyperlink and click on it. The first user spent 47 s on the Web page, the second user spent 29 s on the Web page.

Accuracy The accuracy is measured by the average absolute percentage errors of the two metrics TTFF and TF, within the AOIs marked by the analysts. The error is defined as the difference between the outcomes of the analysts and the ground truth by the experimenter. A lower value indicates a higher accuracy. The errors appear to be similarly low for all Web pages, as presented in Table 4.2. However, the average percentage error value of 31.3% ± 44.3% for the TF estimation on Jimdo.com, when using the enhanced representation of the Web page, appears to be high. But a two-tailed Mann-Whitney U test indicates that the TF estimation for the enhanced representation (Mdn = 0.0%) is not significantly different from the video recording (Mdn = 0.0%), U = 30, with p = 0.14 for the Jimdo.com Web page. A deeper look into the data reveals that the high error was mainly caused by one analyst, who reported correct values for TTFF but 100% deviating answers for the TF values, in the analysis of both users. There has been no systematic error, as three out of five analysts of that group have reported values with zero percent error.

Task completion time We have measured the time that the analysts required to fulfill the tasks. The times include the marking of the AOIs and the output of the statistical values. See Figure 4.7 for a box



Fixed

Target

8. Alt Coblenz
34 reviews
€€ · Wine Bars, German

Am Plan 13
56068 Koblenz
Germany
0261 160656

Perfect choice for traditional Coblenz cuisine. The menu is large, with amazing home cooked meal choices. The service was great, the decor was very charming. They also have a... read more

9. Festung Ehrenbreitstein
35 reviews
Landmarks & Historical Buildings

Festung
56077 K
German
0261 66754000

Took a ride with my (slightly scared o ble car up to the fort. The building is in a better condition than I previously Plenty... read more

10. Restaurant Poseidon
31 reviews
€€ · Greek

Rheinstr. 23
56068 Koblenz
Germany
0261 34935

Wonderful food and good ambiance. Would definitely eat here again! My wife and I treated ourselves to a lunch of a Greek salad and Moussaka after spending all morning wandering the... read more

Page 1 of 100

1 2 3 4 5 6 7 8 9 Next

More Nearby

Active Life
Beauty & Spas
Arts & Entertainment
Education
Automotive
Event Planning & Services

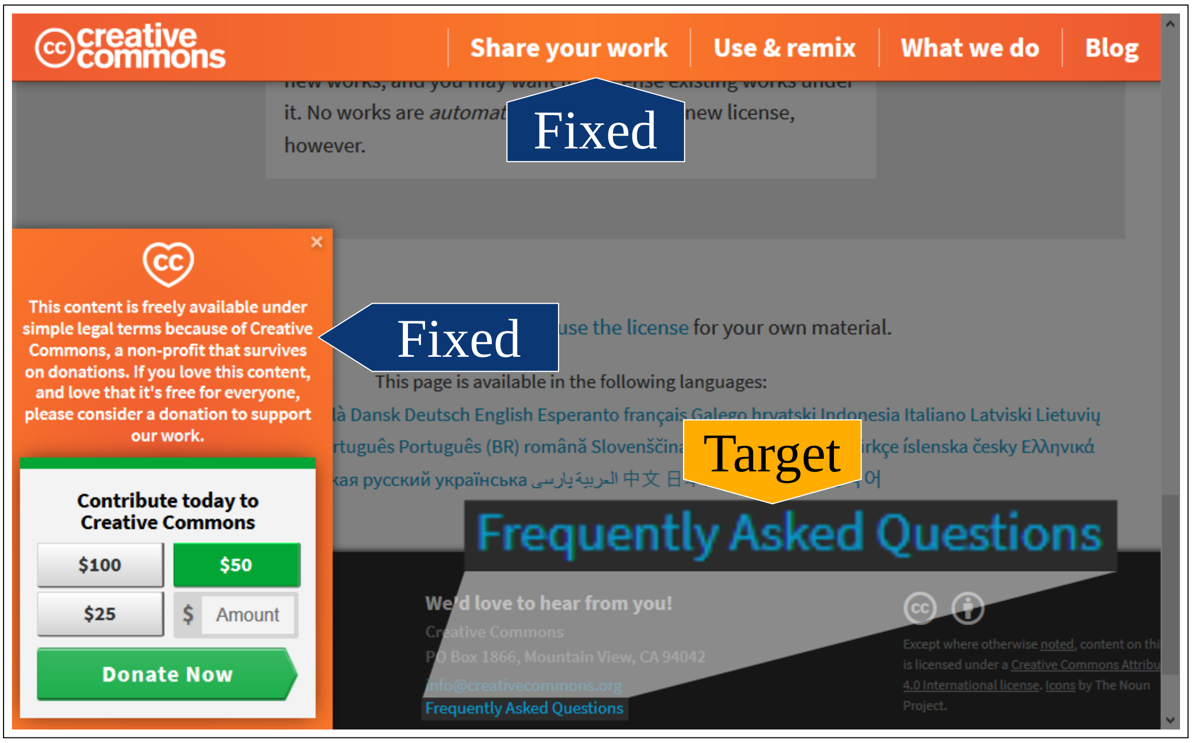
Figure 4.5: Yelp: The search page makes use of a fixed column on the right to display a map with location information about the results. The users were asked to select the 10th entry of the search results in the list. The first user spent 16 s on the Web page, the second user spent 10 s on the Web page.

plot per Web page. A Mann-Whitney U test shows no significant difference between the analysis of the first user with the enhanced representation and the analysis of the first user with the video recording for each Web page.

However, the two-tailed Mann-Whitney U test shows a significant difference in timings for analyzing the second user per Web page for both methods. In average, the analysts only needed 24.8% of the

Table 4.2: Average absolute percentage errors of the metrics estimations by the analysts for the enhanced representation (E) and the video recordings (V), compared to the ground truth.

		Digg.com	Jimdo.com	Yelp.com	CC
E	TTFF	3.0% ± 6.2%	0.0% ± 0.0%	1.2% ± 3.8%	0.0% ± 0.0%
	TF	9.4% ± 17.8%	31.3% ± 44.3%	2.5% ± 7.9%	1.0% ± 3.2%
V	TTFF	15.3% ± 31.8%	0.0% ± 0.0%	1.2% ± 3.8%	0.0% ± 0.0%
	TF	15.3% ± 17.1%	0.0% ± 0.0%	2.5% ± 7.9%	0.0% ± 0.0%



it. No works are *automat* new license, however.

This content is freely available under simple legal terms because of Creative Commons, a non-profit that survives on donations. If you love this content, and love that it's free for everyone, please consider a donation to support our work.

use the license for your own material.

This page is available in the following languages:

là Dansk Deutsch English Esperanto français Galego hrvatski Indonesia Italiano Latviski Lietuvių rtuguês Português (BR) română Slovenščina irkçe íslenska česky Ελληνικά кая русский українська العربية پارسی 中文 日 어

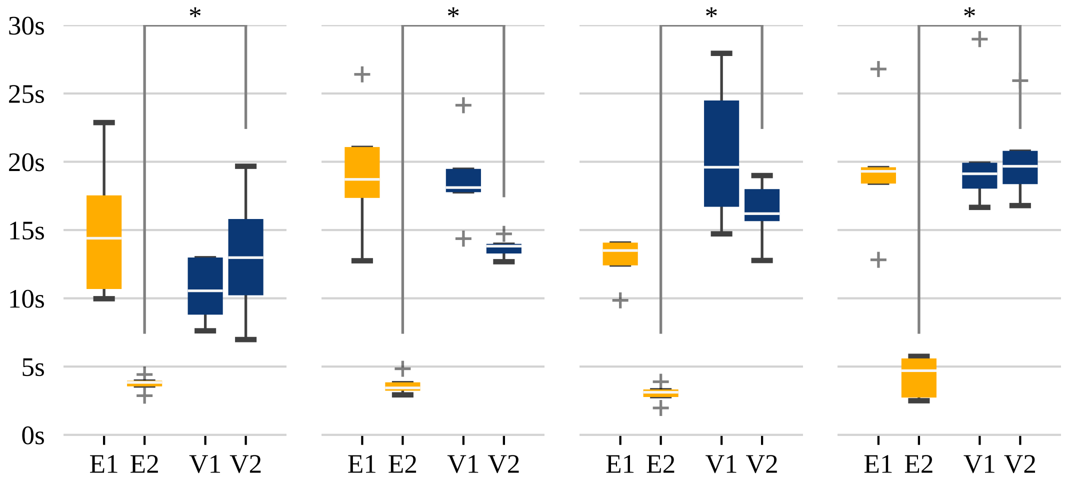
We'd love to hear from you!
Creative Commons
PO Box 1866, Mountain View, CA 94042
info@creativecommons.org
Frequently Asked Questions

Except where otherwise noted, content on thi is licensed under a Creative Commons Attribu 4.0 International license. Icons by The Noun Project.

Figure 4.6: Creative Commons: The Web page can be considered as a complex layout in regards to fixed elements. The navigation bar at the top is only shown when a user scrolls up and the donation pop-up on the left is displayed after a certain timeout. For the evaluation, only the donation pop-up was considered by the analysts. Analogously to Digg.com, users were asked to click on the FAQ. The first user spent 33 s on the Web page, the second user spent 25 s on the Web page.

time for analysis of the second user when working on the enhanced representation in comparison to the colleagues who worked on the video recording.

NASA-TLX The NASA-TLX questionnaire is designed to measure the workload of the analysts. The analysts have been asked to answer the NASA-TLX questionnaire after reporting the measurements for the user attention of each video recording, but only once for the enhanced representation of each Web page. This procedure has been defined due to the very low timings for the analysis of the second user when the enhanced representation of a Web page was utilized. If we had asked the participants to fill in the same questionnaire within a few seconds, the result would have been biased through the effort of filling the questionnaire itself. A bar plot of the outcome is shown in Figure 4.8. Most raw values are similar for both methods, though the enhanced representation receives slightly better average ratings than the analysis of the first user with the video recording. The temporal demand appears to be significantly lower. A two-tailed Mann-Whitney U test supports the impression that the temporal



(a) Digg.com **(b)** Jimdo.com **(c)** Yelp.com **(d)** Creativecommons.org

Figure 4.7: The box plot shows the times required by the analysts to solve the analysis tasks per Web page. The time for analyzing the first user with the enhanced representation is labeled with E1 and for the second user with E2. Similarly, the timings of the analysts using the video recordings are labeled as V1 for the first user and V2 for the second user. A * denotes a significant difference between the distribution of the two sets.

demand is significantly higher for the analysis with the video recording of the first user (Mdn = 2.5) than for the analysis with the enhanced representation of both users (Mdn = 1.0), U = 111, p = 0.017. The Cohens's d effect size is 0.82 and thus considered to be of large value [38]. This emphasizes that the analysts felt more rushed and hurried when using video recordings as a method than with the enhanced representation.

Difficulty Estimation We report an average overestimation in difficulty perception on the value scale from one to five of 1.34 for the enhanced representation and 0.90 for the video recording.

Discussion

The results show that the enhanced representation method allows a less time-consuming analysis than with the video recordings while enabling the analysts to reach a similar level of accuracy in estimating the attention. These results validate our first hypothesis (H1), i. e., the analysis of gaze data on viewport-relative positioned elements with the enhanced representation can be as accurate as with the video recording.

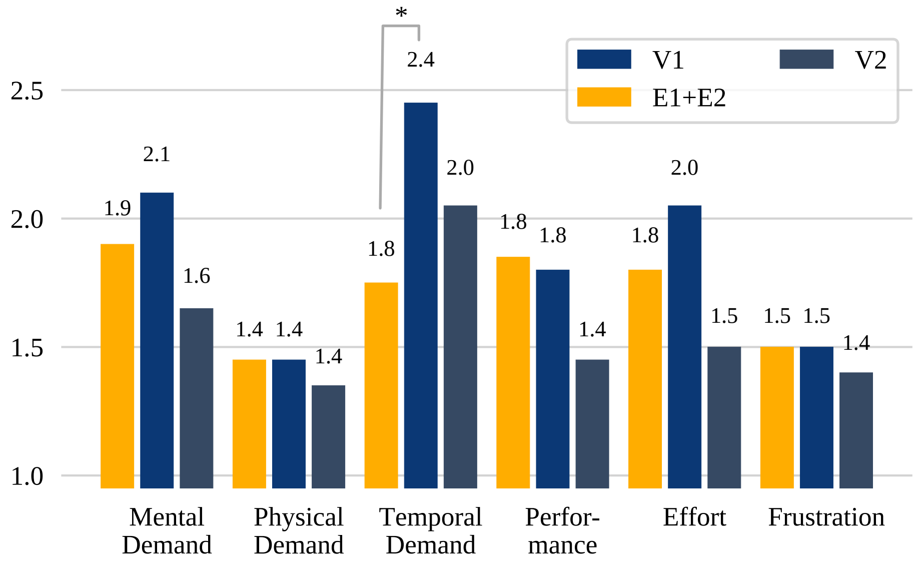


Figure 4.8: Raw NASA-TLX scores. Lower scores signify lower perceived workload. A * denotes a significant difference between the distribution of the two sets.

The times required for the analysis let us conclude that the analysts had to work on both users independently when presented with the video recordings of their sessions. This appears to be counter-intuitive, as the fixed elements should stay in the same position in the frames of each video recording from the same Web page. But fixed elements like the pop-up on Creativecommons.org have been not visible through the entire user session, but only after the user reached a certain scroll offset. The analysts had to define at which point in time the pop-up was visible in the video recording and at which it was not. However, the times required for the analysis of the first user are similar for both methods. We observe that while in the case of a video recording the time was invested to find out when the fixed element has been visible, for the enhanced representation the analysts invested time in scrolling and zooming, to find the fixed element in the enhanced representation before marking the AOIs.

The NASA-TLX results indicate that the temporal demand to analyze on basis of the first user's video recording is significantly higher than analyzing both users with the enhanced representation. However, the mental demand, physical demand, effort, and frustration metrics of the average NASA-TLX scores are lower for the second video recording than for the enhanced representation. One could argue that the analysts get into a “flow” and the task load might converge to a minimum, which is below the task load for the enhanced representation. But when an analyst just has started to analyze the video recording of the second user, another analyst using the enhanced representation of the same task is already done with the analysis, because of the low time demand on successive users. The visualization of the gaze data of multiple users at once accelerates the analysis process significantly. After the first user, the analyst can reuse existing AOIs, regardless of whether they lay upon page-relative or viewport-relative elements of the Web page.

The positive outcome for both timing and temporal demand supports our second hypothesis (H2). For multiple users, the enhanced representation method requires less time and temporal demand than the video recordings, rendering the process more efficient for them.

We report a slightly higher estimation of the difficulty by analysts with the enhanced representation. This might be introduced through the visualization of saccades from page-relative into view-port-relative content and vice versa. For example, the saccades in Figure 4.9 connect fixations on the page with fixations on the fixed element in the lower-left corner. These *virtual saccades* are visualized longer than their actual length, hence analysts might have been unaccustomed in interpretation. We argue that this is only a matter of training or visual hints for the analysts in the enhanced representation.

The evaluation covers the analysis of gaze data from two users on each Web page and already shows a clear trend in favor of the enhanced representation. Video recordings as a method would suffer significantly once the dataset becomes larger, i. e., for large-scale usability studies with many users. Recent research has shown that Web usability studies would require at least 27 users for searching tasks and 34 users for browsing tasks [59], which emphasizes the potential of the enhanced representation method in real-world applications. We show that usability experts can work more effectively when analyzing the gaze data of two users, let alone over 20 users. Furthermore, with the evolution of affordable eye-tracking devices, the user base is expanding, and there is an increasing interest for Web page usability analysis using crowdsourcing [117].

The composition of the page- and the viewport-relative contents might be displayed more interactively, e. g., allowing a usability expert to inspect only certain fixed elements or to move them on the locations as they have been perceived by the users. Moreover, the visualization of accumulated attention might be combined with video recordings of individual attention and interaction. This enables usability experts to first get an impression about the overall attention pattern and then watch specific outlying behaviors in detail in the video recording. This has the potential to merge the benefits of both methods, the scalability of the enhanced representation method and the intuitive interpretation with a video recording.

4.4 Conclusion

We argue that the recording and representation of Web pages is an imperative aspect to support the analysis of gaze data for estimating the attention of users. However, current methods of recording and representation are limited in terms of accuracy and scalability. We propose an enhanced representation method that stitches screenshots from the user viewport and extracts fixed elements from the Web page document, to tackle both challenges of accuracy and scalability. The evaluation results signify the applicability of our enhanced representation method as an accurate and scalable approach. We envision the applicability of the method in supporting large-scale quantitative studies using eye tracking.

In the future, we aim to extend the extraction approach to improve the precision in the cropping of fixed elements. For example, fixed elements may cast shadows that stay on the page after cropping of the fixed elements. Additionally, fixed elements might have overflowing children elements, which expand the bounding box or even define a shape different from a rectangular box. These challenges might be tackled by computer-vision approaches, which can be augmented with structural information from the Web page document.

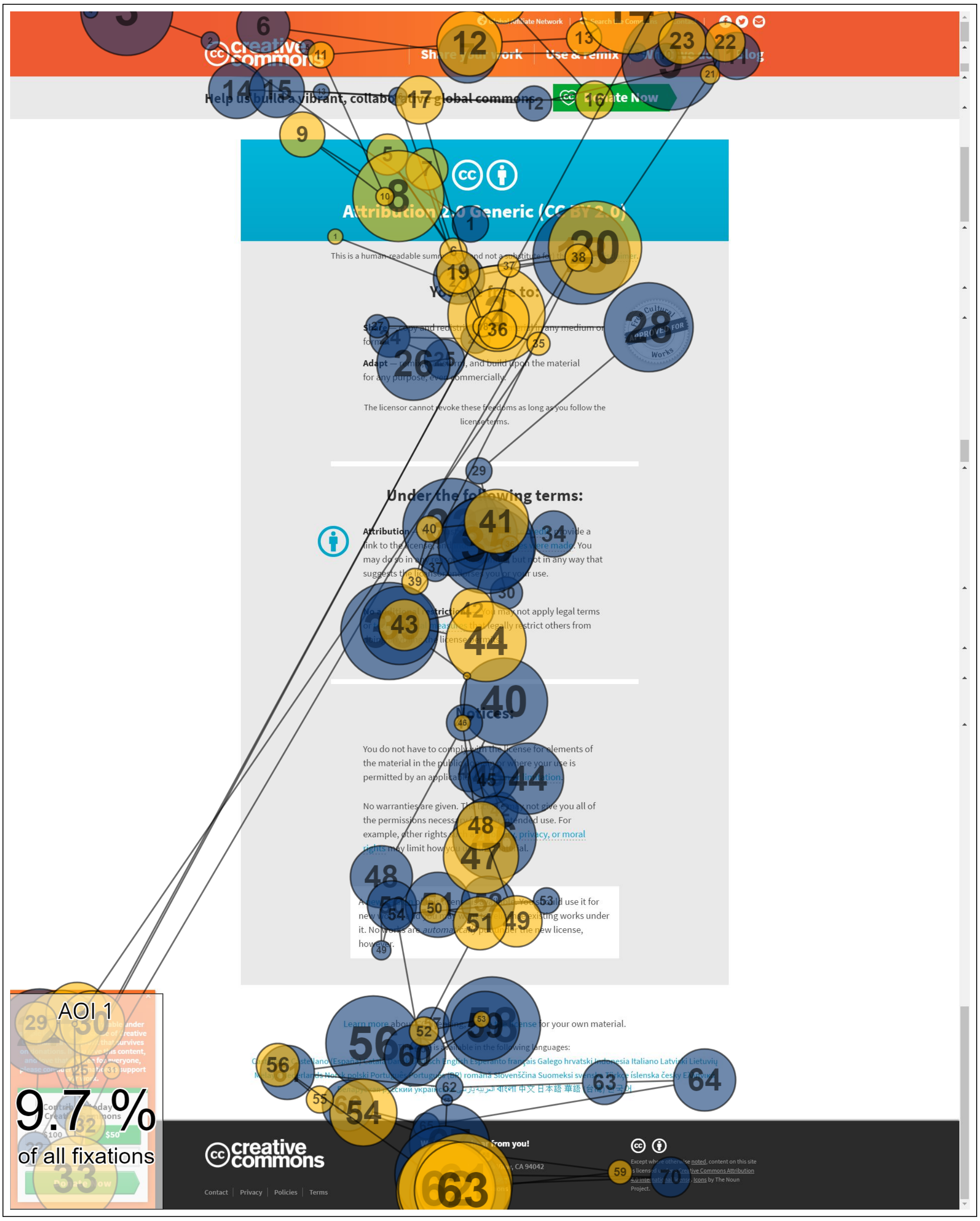

AOI 1
9.7 %
of all fixations
Attribution 2.0 Generic (CC BY 2.0)
Under the following terms:
Notices:

Figure 4.9: Enhanced representation of the Creativecommons.org Web page from the dataset. The gaze data of one user is plotted as scanpath in yellow color, the gaze data of the other user is plotted as scanpath in blue color.

CHAPTER 5

Recording of Changing Contents

In this chapter, we present a method to improve the recording and representation of Web pages with dynamics that are introduced through changing contents. Many Web pages make use of various changing contents, like photo carousels or expanding menus, causing one Web page to have many different interface states instead of only one interface state. Changing contents can be implemented in manifold ways — DOM tree manipulations, canvas, or CSS animations. This brings a challenging scenario of detecting dynamics and adapting the enhanced representation for gaze data mapping and analysis.

To deal with these challenges and the limitations, we propose a framework of *visual stimuli discovery* that creates image-based representations of a Web page like the enhanced representation method but captures the dynamics of individual interface elements like the video and introspection method. Like video and interaction recordings, we record a video and the interactions of each user session. In the bootstrapping phase, we semi-automatically label contiguous video frames as belonging to one *stimulus shot*, i. e., a partial recording of one visual stimulus, or representing the boundary between two adjacent stimulus shots. We use the labels and combine them with the analysis of computer-vision features on these frames to train a classifier that emulates the human decision as to whether two contiguous frames should be considered as belonging to the same or different visual stimuli. We call this decision process *visual change classification* and define the decision process from the point of view of a usability expert. In the application phase, we apply the learned classifier to (i) group contiguous video frames from one user session into stimulus shots and (ii) cluster stimulus shots from several user sessions into overall representations of visual stimuli that are aligned with interaction recordings by simple alignment of a time and user index.

First, we formally describe the framework of visual stimuli discovery in Section 5.1. Second, we explain the dataset that we use to apply and assess the feasibility of the formal framework in Section 5.2. Third, we define a decision process about visual change between contiguous frames and realize the reproduction of these decisions with computer-vision features and machine learning as visual change classifiers in Section 5.3. Fourth, we integrate the visual change classifiers into the framework of visual stimuli discovery and apply them to the dataset in Section 5.4. Fifth, we report a computational evaluation about the quality of the discovered visual stimuli from the dataset in Section 5.5. Sixth, we present user-centered evaluations of the discovered visual stimuli from the dataset in Section 5.6.

5.1 Formal Framework of the Visual Stimuli Discovery

In this section, we give a high-level overview of our approach and an abstract formalization of its major components. We aim to support the usability analysis of dynamic Web pages which exhibit rich interaction behaviors, choreographing dynamic elements with changing content on user responses. For such usability analysis, visual stimuli cannot be defined beforehand. We propose a framework to discover visual stimuli (semi-) automatically. Figure 5.1 depicts our framework of visual stimuli discovery in its application phase.

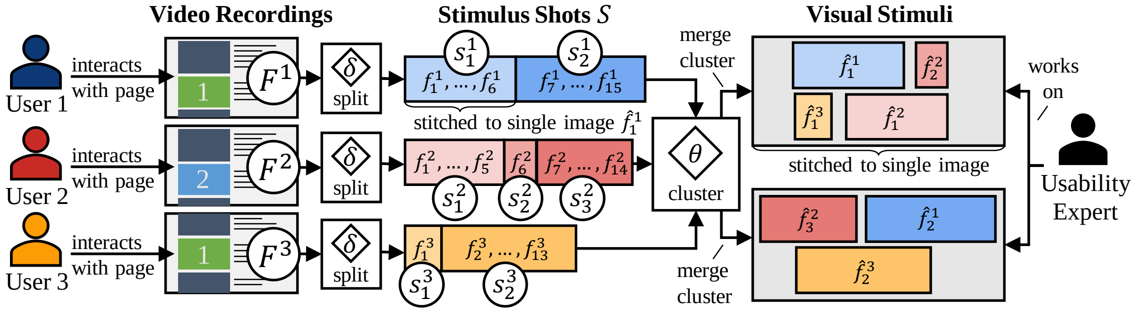


Figure 5.1: Application phase for the framework of visual stimuli discovery considering three user sessions.

Splitting Videos into Stimulus Shots We define the video recording F^i of a user session i to consist of a sequence of frames $F^i = (f_1^i, f_2^i, \dots, f_{n_i}^i)$. For this, we define a classifier δ that detects visual changes between two frames f_a, f_b. The aim is that $\delta(f_a, f_b)$ returns 0 if the two frames are visually so similar that they should be considered to belong to one visual stimulus, and 1 otherwise.

We partition a complete video recording F^i of a user session i into stimulus shots S^i, where each stimulus shot $s_j^i \in S^i$ is defined as

$$s_j^i = (f_k^i, \dots, f_{k+l}^i \mid \forall m \in [k, k+l-1] : \delta(f_m^i, f_{m+1}^i) = 0 \ \wedge \ \delta(f_{k-1}^i, f_k^i) = \delta(f_{k+l}^i, f_{k+l+1}^i) = 1).$$

We define $\hat{f}_j^i$ as a stitch of all frames in s_j^i and we define the set of all stimulus shots from all video recordings as $S = \cup_i S^i$.

In the implementation described in Section 5.4 we have decided to not stitch the frames at the very end of the splitting procedure. Instead, we stitch the frames on the fly during the splitting procedure. This allows us to compare each frame from the video recording not only with the previous frame but with the so-far stitched frame of the current stimulus shot. The stitched frame can contain areas from frames before the previous frame, such that the visual change classification has more information to decide for the current frame whether it belongs to the current stimulus shot or not. This approach does therefore further improve the outcome of the splitting procedure.

Merging Stimulus Shots to Visual Stimuli Our goal is to merge visually similar stimulus shots within and between user sessions toward a visually coherent visual stimulus. Thus, we perform agglomerative clustering on the stimulus shots S using a distance function θ.

We define θ between two stitches $\hat{f}_a$, $\hat{f}_b$ as

$$\theta(\hat{f}_a, \hat{f}_b) = (1 - \delta(\hat{f}_a, \hat{f}_b)) \cdot A(\hat{f}_a, \hat{f}_b).$$

The function A computes the area of overlap of the two stitched frames $\hat{f}_a$ and $\hat{f}_b$ in pixels. The function serves us as a similarity score of two stitched frames. It allows us to first merge stimuli shots that cover bigger portions of an interface. We do not normalize the similarity score, as there is no maximum extend of the stitched frames given. The output of clustering is a set of visual stimuli. A single visual stimulus contains a set of stimulus shots that are stitched together such that they can be rendered for usability analysis in a single visualization.

The details of this high-level approach will now be elaborated on in the following sections. To apply and assess the feasibility of the formal framework, we need a dataset representing interaction with dynamic and rich Web pages. Hence, in the next section, we describe the recording of user sessions on popular Web sites. After building on the benchmark data and labels, we discuss the visual change classification, which plays a key role in both splitting and merging of video recordings toward visual stimuli. We revisit the framework for its application on the dataset, applying the classifiers, and performing computational and user-centered evaluations of the discovered visual stimuli.

5.2 Recording a Dataset of Dynamic Web Pages

The dataset is segmented into sessions. We have recorded a video and a datacast for each session. The video contains a recording of the Web browser viewport as it has been displayed to a participant. The datacast stores information as extracted from the Web browser and DOM tree of the browsed Web pages, alongside interaction recordings. The dataset has been published under the public domain on the Zenodo platform.[1]

Apparatus We have developed a logger application based on the Qt [39] framework and its integrated Chromium-based [10] Web engine. The video recording was captured with five frames per second[2] from the Qt environment via FFmpeg [11], using lossless VP9 [180] encoding. The direct capture from the Qt environment allows us to exclude the mouse cursor from the video. We have logged cursor movements, mouse clicks, scroll events, and gaze signals. The DOM tree has been polled every 50 milliseconds for

[1] `https://zenodo.org/record/4737774`

[2] For time estimations, we assume each video frame to cover 200 ms of a session. However, the encoder skipped some frames at the recording as we noticed after the recording.

fixed elements, which are logged into the datacast, including their extent and a DOM path computed from the location of the fixed element within the DOM tree. Moreover, we have logged the creation and mutation of the DOM tree for each Web page. Scrollbars have been hidden for the recording, as they would disturb the features. The Web browser interface has been limited to a full-screen window that offers basic browsing controls like an address bar, back and forwards navigation, recording facilities, and a viewport to the Web page with a resolution of 1,024 × 768 pixels. The display had a size of 24 in. and a resolution of 1,680 × 1,050 pixels. The gaze signals have been recorded with a Tobii 4C eye-tracking device, which captures the POR at a frequency of 90 Hz. See Figure 5.2 for the setup.

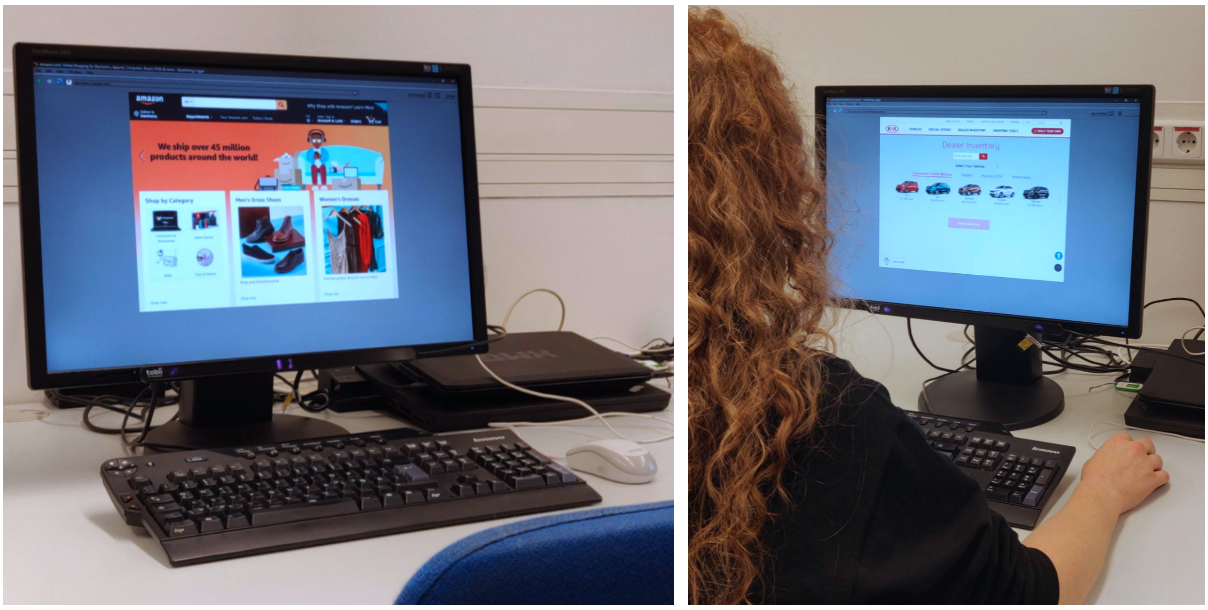


Figure 5.2: Recording setup consisting of a monitor with remote eye-tracking device mounted below the screen and mouse and keyboard for input. The photo on the right was taken during the recording.

Procedure We defined a protocol that was handed out to the participants on a sheet before the recording. Moreover, we read out loud the instructions during the recording session such that there occurred less interruption in the browsing behavior. See Appendix A for details. We have chosen twelve Web sites in the English language of four different categories, inspired by the Alexa Top 50[3] categories. The category "Shopping" consists of the sites "*walmart.com*" (Walmart), "*amazon.com*" (Amazon), and "*store.steampowered.com*" (Steam). The category "News" comprises "*reddit.com/r/pics/top/?t=month*" (Reddit), "*edition.cnn.com*" (CNN), and "*theguardian.com/international*" (Guardian). The category "Health" includes "*nih.gov*" (NIH), "*webmd.com*" (WebMD), and "*mayoclinic.org*" (MayoClinic). The category "Cars" consists of the

[3] `https://www.alexa.com/topsites`

sites *“gm.com”* (General Motors), *“nissanusa.com”* (Nissan), and *“kia.com/us/en/home”* (Kia). Our general instruction to the participants was to explore each page thoroughly, i. e., hover elements and menus, read descriptions, and click through photos of a photo carousel. For each Web site, the participants started the interaction on the landing page. Then they had to find a specific hyperlink on the landing page and navigate to the next page. On the next page, they were asked to explore the page and choose a hyperlink of their own choice as per their interest. Thus, for each Web site, they visited three pages. Two of the pages were defined by our protocol and the third page was chosen by the participants.

Participants Four male participants (age = 30 ± 2.35 years) were invited for the dataset recordings. All four participants are researchers in computer science and experienced Web users. They participated voluntarily in the dataset recording. The strategy to invite experienced Web users was motivated by the nature of sophisticated tasks, e. g., to explore the dynamics of pages thoroughly, however, not to leave the page accidentally through the activation of an outgoing hyperlink. Many menus require a mouse click for expansion, while other interface elements lead to another Web page if clicked. Only experienced Web users can make reasonable predictions of these design choices.

Results We recorded 155 minutes of Web browsing in about 1.23 GB of video recordings and datacasts. All videos contain a total of 45,310 frames. We have recorded 10,742 scrolls, 111,058 mouse movements, 893 clicks and 837,716 gaze points. The participants have browsed 172 URLs, which means on average 3.56 URLs per session. This reflects our protocol, considering that some participants clicked by accident on outgoing hyperlinks. In addition, we have recorded 140,322 DOM tree mutations, not counting the operations for the creation of the initial DOM tree of a Web page. On average, there have been 2,923 DOM tree mutations per session, with a high standard deviation of 2,735. For example, we recorded 3,423 additions of DOM nodes, 544 removals, and 797 changes for the third participant during his session on the Walmart site. We do not further use the DOM tree mutations, yet we believe it might allow further insights in the future. In the remainder of the chapter, we will refer to the participants of the dataset as “users,” if not denoted otherwise.

5.3 Visual Change Classifier

In this section, we discuss the visual change classifier δ that is required in the framework of visual stimuli discovery for both splitting and merging of video recordings toward visual stimuli. We label contiguous video frames as belonging to one stimulus shot or as representing the boundary between two adjacent stimulus shots. We base the labeling on a decision process from the point of view of a usability expert, which we define in this section. Then we introduce computer-vision features from shot and scene detection literature that are computed on the contiguous frames. Finally, we investigate the effectiveness of different classifiers using those features to reproduce the human-decision process of labeling for visual change.

5.3.1 Labeling of Visual Change

For labeling and feature computation, we have developed a tool[4] that provides a graphical interface to walk through image pairs of a user session and let a usability expert label visual change. See Figure 5.3 for a screenshot of that tool.

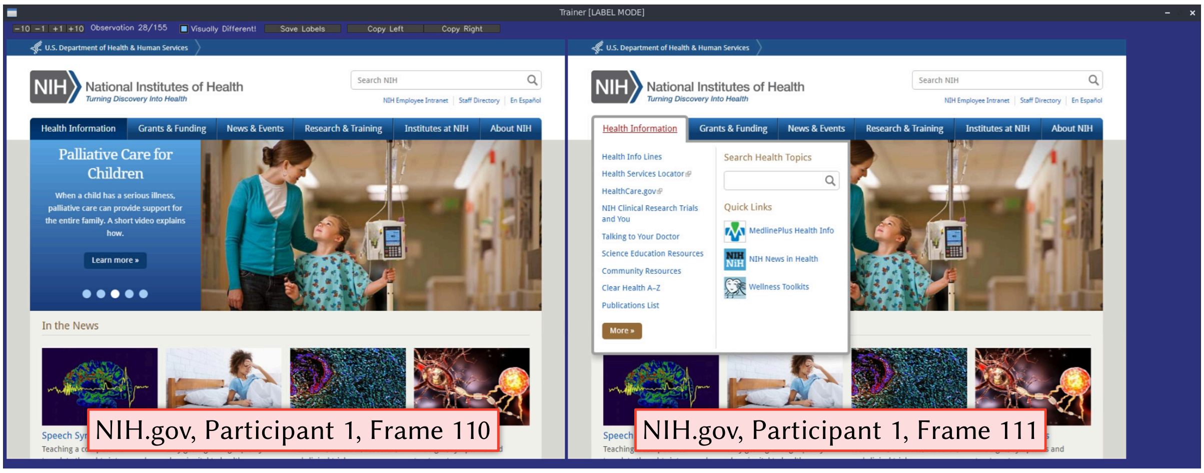

Trainer [LABEL MODE]
−10 −1 +1 +10 Observation 28/155
□ Visually Different!
Save Labels
Copy Left
Copy Right
NIH.gov, Participant 1, Frame 110
NIH.gov, Participant 1, Frame 111

Figure 5.3: Screenshot of the tool used to label visual change in each observation.

Apparatus We load the video recording and the datacast of a user session into the tool and extract image pairs as shown in Figure 5.4. For each (a) consecutive pair of frames (e. g., frames n-1 and n, frames n and n+1, ...), we (b) crop the fixed elements, as defined in the DOM tree, from the frames — similarly to the enhanced representation method [139]. Real-time observation of the DOM tree during recording lets us update the position, extent, and visibility of fixed elements according to their dynamic behavior, i. e., when a menu expands or collapses upon user interaction. This results in potentially multiple regions of fixed elements and the remaining region of each frame, which is the scrollable part of the Web page. The regions are treated separately in the following. Then, (c) each pair of regions from the two frames is treated as one *observation*. We assume that scrolling should not affect the determination of visual change, because we will later stitch together two frames that only deviate by their scroll position. Therefore, we transform the regions to match in their scroll offset and we crop only the overlapping image portions for further consideration. Each observation is (d) manually labeled for visual change (“yes”/“no”). The features (e), as later listed in Table 5.1, are also computed. Notably, we auto-

[4] `https://github.com/eyevido/visual-stimuli-discovery/tree/master/code/src/exe/Trainer`

matically skip the labeling of observations that perfectly match every pixel, as the images are visually identical and would unnecessarily consume labeling effort and feature computation time. The tool has been implemented in C++ and uses OpenCV [25] to prepare the observations and to compute features.

We have noticed that the scroll offset from the datacasts does not match perfectly with the scrolling as recorded in the videos. See Appendix B for details about our approach for estimating the true scroll offset in the video recordings, using OpenCV ORB features, and a homography estimation between the pixels from the regions of an observation.

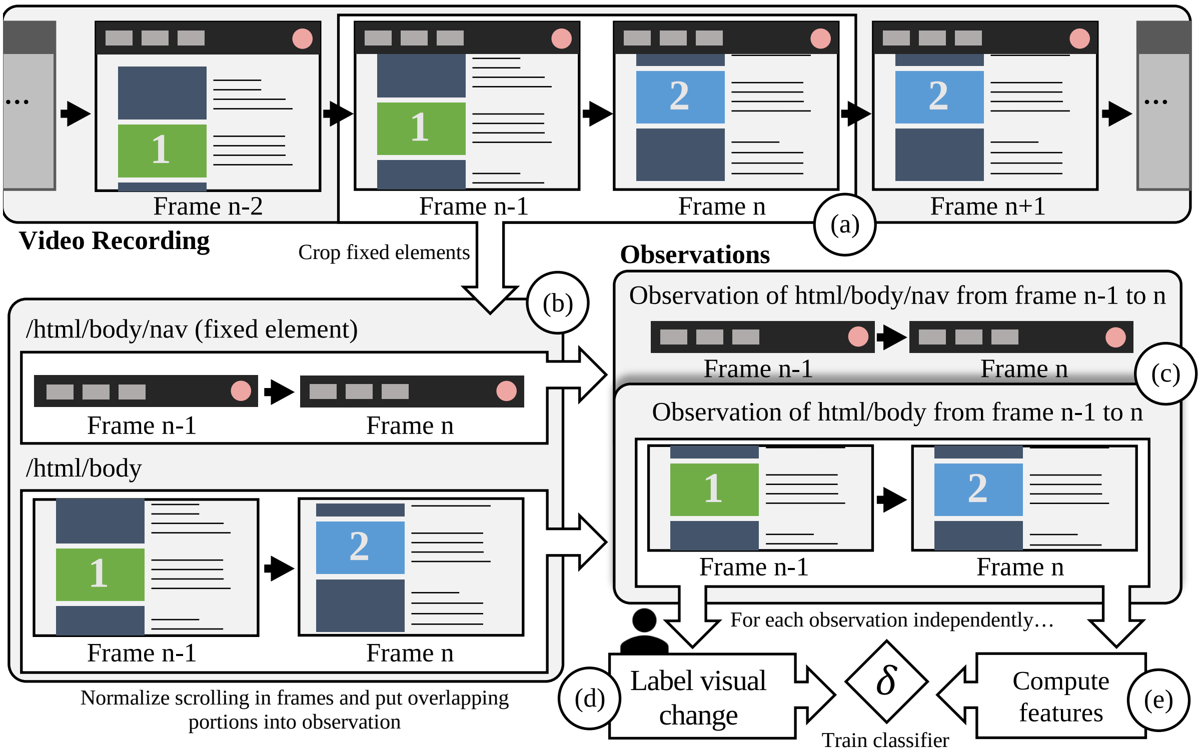


Figure 5.4: Observations are extracted from contiguous frames in a video recording, manually labeled for visual change, and computer-vision features computed.

Procedure To counter the subjectivity of labeling visual changes in observations, we have designed a decision process from the point of view of a usability expert whose objective is to aggregate interactions on distinctive visual states of an interface. See Figure 5.5 for the decision process.

Results One of the authors of this paper (an experienced researcher in usability analysis) performed the task of labeling the dataset over 16 hours within three days. In total 23,571 observations have been labeled, from which 4,446 observations have been labeled as visually different, and thus should separate stimulus shots.

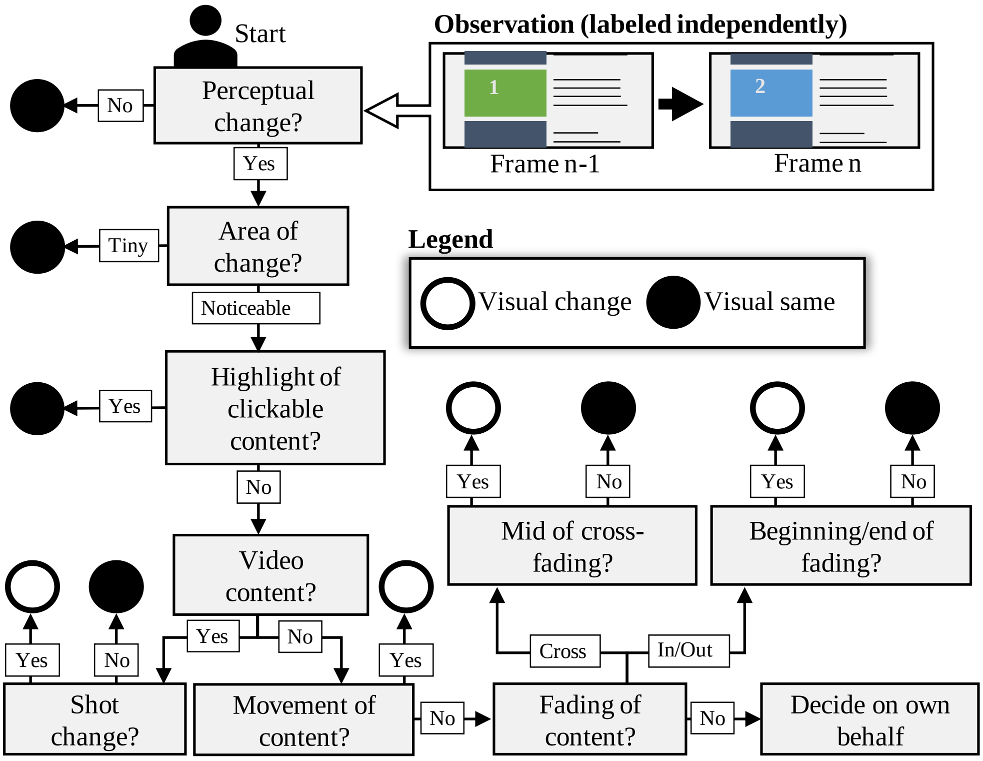


Figure 5.5: Decision process for labeling of visual change per observation.

To verify the objectivity of our decision process, we also let two students label a subset of the dataset. The students were not involved in this work. They were introduced to the task with an explanation of the decision process on a sheet of paper showing Figure 5.5 and an in-person explanation of the labeling on one exemplary user session. They labeled a subset of 12 user sessions out of the total of 48 user sessions, i. e., the shopping-related Web sites on the first user, the news-related Web sites on the second user, the health-related Web sites on the third user, and the car-related Web sites on the fourth user. We report a Fleiss' Kappa score of 0.71, which is to be considered as a substantial agreement, according to Landis and Koch [115].

Moreover, there exist well-received and ready-to-use automatic video shot segmentation tools [96]. One of the best scoring tools is the Multimedia Knowledge and Social Media Analytics Laboratory (MKLab) [140], which we have applied to our dataset. The tool uses a score function, based on local (SURF) and global (color histograms) features, to compare consecutive frames. Applied to our dataset, the MKLab tool detects only 175 shot separations through abrupt transitions, one dissolve transition, and 49 wipe transitions. In comparison, we have manually labeled 4,446 shot separations throughout the dataset. Thus, we argue that existing video shot segmentation techniques cannot be applied to the problem of visual stimuli discovery and we need to investigate choices in features and classifiers.

5.3.2 Features for Visual Change Detection

A variety of features to compare video frames is applicable [48]. We compute in total 53 computer-vision features categorized into seven different types (see Table 5.1), deemed relevant to detect visual changes in an interface. Value-based features consider the difference of pixel values between two images, either via aggregation of absolute values differences or via counting of changed pixels. This type of low-level feature accounts for every kind of visible change in an interface on a local scale. Histogram-based features consider the change in color distribution between two images on a global scale. This type of feature might support recognition of overall changes (e. g., dimming of background) on the interface, while local changes might go unnoticed. Edge-based features rely on changes in the visibility of edges between two images. Edges are an important aspect of human vision; thus, they may provide a strong indication of visual changes on an interface. Signal-based features like peak-signal-to-noise ratio (PSNR) and the mean structural similarity index [191] (MSSIM) are popular to measure image quality differences between two images. Optical-flow-based features consider the movement of content in video frames. This type of feature might be of interest for moving content like photo carousels and scrollable regions like chat windows. SIFT-based features could match descriptors of remarkable points between two images. This complex feature supports checking for changing contents at a layout level. Some interfaces predominately contain textual content; thus, we detect text using the Tesseract 4.0 optical character recognition library [173] and derive text-based features. We use the recognized texts both for bag-of-words and for character-level n-gram (with $n = 3$ [194]) similarity estimations. See Appendix C for more details about the features.

Table 5.1: Evaluated computer-vision features to estimate visual change. Postfixes b, g, and r stand for the blue, green, and red color channel, respectively. Postfixes h, s, and l stand for hue, saturation, and lightness. The postfixes of the SIFT-based matches describe the applied threshold on the SIFT-feature similarity score. The postfix spatial stands for an additional check for a similar image coordinate. See Chapter C for details.

Type	Value-based		Histogram-based
Feature	*Aggregation*	*Count*	*Correlation*
Variations	agg_bgr/b/g/r	count_bgr/b/g/r	corr_b/g/r
	agg_h/s/l	count_h/s/l	corr_h/s/l
	agg_gray	count_gray	corr_gray

Type	Edge-based	Signal-based	
Feature	*Change Fraction*	*PSNR*	*MSSIM*
Variations	change_fraction	psnr	mssim_b/g/r

Type	Optical-flow-based		SIFT-based
Feature	*Angle*	*Magnitude*	*Match*
Variations	angle_mean/std	mag_max/mean/std	match/_0/4/16/64/256/512
			match_spatial
			match_dist_min/max/mean/std

Type	Text-based	
Feature	*Bag of Words*	*n-Grams*
Variations	diff_words_count	n_grams_match_count/match_ratio/jaccard
	unique_term_count	n_grams_min_count/max_count
		n_grams_vocabulary_size

5.3.3 Investigating the Effectiveness of Visual Change Classifiers

First, the training of such a classifier should be feasible in the scope of a behavioral study by a usability expert. A usability expert may label the visual changes on one video recording and then apply the trained classifier to separate all other video recordings of users on one Web site into stimulus shots. Second, a visual change classifier should show good results in reproducing the human-decision process for both label classes, visual change and no visual change, aka "visual same." Thus, we investigate the performance of popular classifiers in the following.

Apparatus We have trained a logistic regression classifier, a support vector classifier (SVC), and a random forest classifier. The SVC uses a radial basis function kernel and balanced class weighting using the synthetic minority over-sampling technique [118]. The random forest classifier makes use of 100 decision trees and entropy as a measurement for purity. We have also evaluated different tree counts, yet variations did not yield an improvement in classification. Analogously to the SVC, a balanced class weighting has been applied. Additionally, we have implemented a baseline classifier. For every training set, a threshold of the feature `count_bgr` is optimized regarding an average weighted F_1 score. The threshold determined in the `count_bgr` feature is then used to classify the test dataset. The baseline can be intuitively described as a threshold of a number of pixels, which are different between two images. For all computations, we have removed 937 observations whose image overlap was below or equal to 32 pixels in either width or height. We consider the overlap of below or equal 32 pixels as not relevant for the training of the visual change classifier. Moreover, we have normalized all features independently with the min-max method. The min-max method normalizes all values of each feature in the training data between zero and one and applies the same transformation for the test data. The logistic regression classifier and the SVC are outperformed by the random forest classifier in almost all cases, why we only consider the results from the baseline approach and the random forest classifier in the following.

In addition to the F_1 scores per class (visual same or visual change), we report the measures of coverage and overflow for shot detection [187]. The measure of coverage indicates how much a stimulus shot from the ground truth is represented by the most overlapping computed stimulus shot. A value close to one is better, a value close to zero is worse. The measure of overflow indicates how much the neighboring stimulus shots of each ground-truth stimulus shot are overlapped with computed stimulus shots intersecting with the ground-truth stimulus shot. A value close to zero is better, a value close to one is worse. We calculate coverage and overflow by first taking the labels of visual change as boundaries of the ground-truth stimulus shots. Each classifier is then applied to compute visual changes, which are then taken as boundaries of computed stimulus shots. We have used Python 3 with scikit-learn [151] for the analysis of the classifiers.

Results The results of performing four-fold cross-validation by considering each user session once as training data are displayed in Table 5.2. We observe consistent F_1 scores above ninety percent for

Table 5.2: Classifiers of visual change. We use a four-fold cross validation. One user session acts as training data, and three user sessions act as test data. We report F_1 scores and the shot-detection measures of coverage and overflow (excluding fixed elements). *R. Forest* is a random forest classifier. Best classifier result per Web site is printed in bold font. ↑ denotes that a higher value is better. ↓ denotes that a lower value is better.

Shopping Sites	Walmart		Amazon		Steam	
Classifier	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*
↑ Visual Same [F_1]	**93% ± 01%**	87% ± 03%	**94% ± 01%**	90% ± 02%	**92% ± 01%**	72% ± 03%
↑ Visual Change [F_1]	**87% ± 02%**	78% ± 02%	**81% ± 02%**	73% ± 02%	**80% ± 02%**	55% ± 02%
↑ Agg. Coverage	**0.97 ± 0.03**	0.81 ± 0.02	**0.88 ± 0.04**	0.88 ± 0.07	**0.91 ± 0.02**	0.82 ± 0.03
↓ Agg. Overflow	0.18 ± 0.08	**0.05 ± 0.04**	**0.11 ± 0.03**	0.17 ± 0.06	**0.09 ± 0.04**	0.13 ± 0.05
News Sites	Reddit		CNN		Guardian	
Classifier	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*
↑ Visual Same [F_1]	**96% ± 00%**	84% ± 04%	**93% ± 03%**	77% ± 07%	**97% ± 00%**	81% ± 02%
↑ Visual Change [F_1]	**69% ± 01%**	42% ± 04%	**58% ± 08%**	32% ± 01%	**79% ± 01%**	42% ± 03%
↑ Agg. Coverage	**0.94 ± 0.02**	0.76 ± 0.03	**0.85 ± 0.12**	0.54 ± 0.05	**0.88 ± 0.02**	0.51 ± 0.02
↓ Agg. Overflow	0.13 ± 0.04	**0.05 ± 0.01**	0.31 ± 0.15	**0.08 ± 0.13**	0.15 ± 0.02	**0.10 ± 0.03**
Health Sites	NIH		WebMD		MayoClinic	
Classifier	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*
↑ Visual Same [F_1]	**97% ± 01%**	87% ± 02%	85% ± 12%	**86% ± 02%**	**98% ± 00%**	95% ± 00%
↑ Visual Change [F_1]	**92% ± 03%**	72% ± 01%	**55% ± 08%**	54% ± 04%	**88% ± 02%**	78% ± 01%
↑ Agg. Coverage	**0.98 ± 0.02**	0.69 ± 0.09	**0.84 ± 0.20**	0.81 ± 0.10	**0.94 ± 0.04**	0.81 ± 0.02
↓ Agg. Overflow	0.12 ± 0.05	**0.01 ± 0.01**	**0.41 ± 0.21**	0.42 ± 0.11	**0.11 ± 0.04**	**0.11 ± 0.04**
Car Sites	General Motors		Nissan		Kia	
Classifier	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*
↑ Visual Same [F_1]	**97% ± 00%**	83% ± 03%	**97% ± 00%**	92% ± 01%	**97% ± 00%**	93% ± 01%
↑ Visual Change [F_1]	**80% ± 01%**	42% ± 03%	**90% ± 01%**	80% ± 02%	**84% ± 02%**	47% ± 01%
↑ Agg. Coverage	**0.90 ± 0.05**	0.65 ± 0.05	**0.96 ± 0.02**	0.78 ± 0.03	**0.88 ± 0.04**	0.80 ± 0.03
↓ Agg. Overflow	0.25 ± 0.08	**0.15 ± 0.12**	0.15 ± 0.05	**0.05 ± 0.02**	0.11 ± 0.04	**0.07 ± 0.08**

the classification of visually same observations with the random forest classifier. The scores for visual change observations are of mixed nature. In eight out of twelve Web sites the F_1 scores with random forest classifier is above eighty percent. At first sight, the baseline classifier performs best in the overflow measure. However, one must look at both, coverage and overflow. For example, in the case of NIH, the baseline classifier produces many false-positive shot boundaries. Thus, the overflow is low (which is good) but so is the coverage (which is bad). The video would be divided into too many stimulus shots, resulting in an over-segmentation. In conclusion, a random forest classifier is a good choice for a visual change classifier.

Table 5.3: Classifier of visual change with user sessions across categories (using all features). We use a four-fold cross validation, in which the user sessions of three categories act as training data, and the user sessions of one category acts as test data. *SVC* is a support vector classifier, *R. F.* is a random forest classifier. *B.* is the baseline classifier. Best classifier result per category is printed in bold font. ↑ denotes that a higher value is better.

Category	Shopping			News			Health			Cars		
Classifier	*SVC*	*R. F.*	*B.*	*SVC*	*R. F.*	*B.*	*SVC*	*R. F.*	*B.*	*SVC*	*R. F.*	*B.*
↑ Visual Same [F_1]	92%	**93%**	74%	**95%**	94%	82%	80%	**82%**	74%	95%	**97%**	87%
↑ Visual Change [F_1]	80%	**83%**	63%	**64%**	62%	38%	51%	**61%**	50%	73%	**85%**	53%

In detail, the classification performs badly on Web sites from the News category. This might be caused by diverse multimedia content that has been very different between user sessions. Besides those, WebMD from the health category has an exceptionally low F_1 score for visual change. After manually checking which visual changes are not recalled, we found that visual changes between the bright background and white menu do not cause significant changes regarding most features.

We have also evaluated the performance in across-category classification. We consider this to be the more difficult task, as we attempt to find a general classifier that is trained by visuals on Web sites that are from a category different than the classified Web sites. For each category, we have taken all user sessions of the other three categories as the training set, and we have taken the user sessions of the chosen category as the test set. The F_1 scores are shown in Table 5.3. However, the results are worse than the same-site classification. Thus, we consider in the remainder of the chapter the training with one user session for a classifier that is constructed to work on that Web site.

Finally, we have evaluated the importance of features by their contribution to the random forest classifiers in Table 5.2. See Table 5.4 for a ranking of the top-sixteen features. The most important feature in all evaluations is the edge change fraction. It contributes nearly ten percent on average to every decision. The mean structural similarity index tends to be important across all color channels. The next important feature appears to be the SIFT-based matching. The aggregation of value differences in various color channels is important in most decision processes, too. The features from optical flow, histograms, and text recognition are not important for the classifications. We speculate that there may have been no sufficient movement on the interfaces to trigger optical flow. Histograms appear to be less useful for our purpose, as there are usually no global changes in color distribution caused by dynamics in interfaces. The text-based features have been promising, yet it might be that dynamics do not introduce enough new text to be detected in comparison to the already existing text on an interface. The optical character recognition seems also to be very sensitive to fading animations.

Table 5.4: Feature importances as provided by the random forest classifier in scikit-learn [151] when considering one user session as training set and three user sessions as test set.

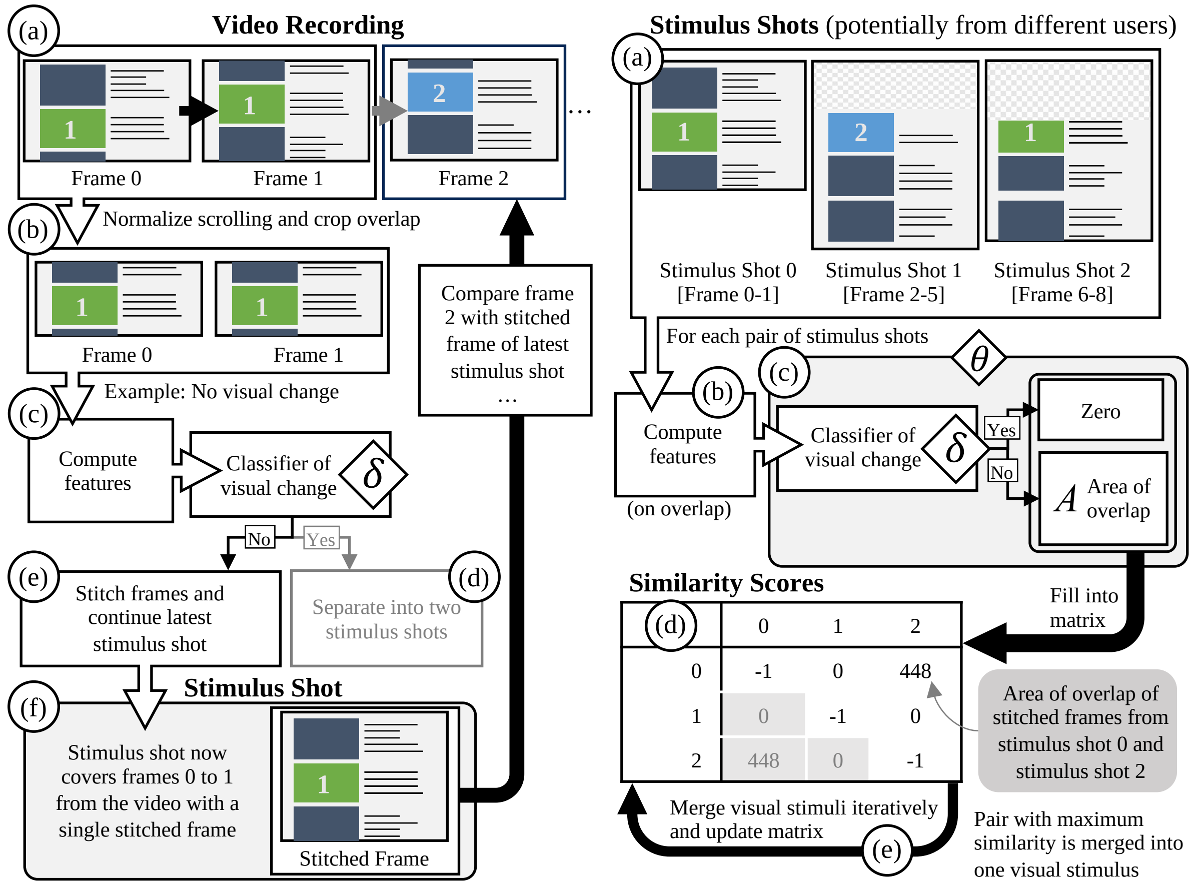
Rank	Feature	Importance	Rank	Feature	Importance
1.	`change_fraction` (edge)	9.5% ± 5.2%	9.	`psnr` (signal)	3.5% ± 3.4%
2.	`mssim_r` (signal)	5.5% ± 4.2%	10.	`agg_g` (value)	3.4% ± 2.1%
3.	`mssim_g` (signal)	5.2% ± 3.5%	11.	`agg_bgr` (value)	3.2% ± 2.2%
4.	`match_256` (SIFT)	4.3% ± 3.8%	12.	`agg_r` (value)	2.8% ± 1.7%
5.	`match_spatial` (SIFT)	4.2% ± 3.6%	13.	`agg_gray` (value)	2.7% ± 1.5%
6.	`mssim_b` (signal)	4.2% ± 3.1%	14.	`agg_h` (value)	2.5% ± 2.4%
7.	`count_b` (value)	4.1% ± 3.2%	15.	`match_64` (SIFT)	2.3% ± 1.5%
8.	`count_l` (value)	3.5% ± 1.7%	16.	`mag_max` (optical flow)	2.2% ± 2.5%

Lessons learnt The random forest classifier promises the best results for visual change detection. The training of the classifier with a single user session shows good results. Considering the above-mentioned feature importance, we only use value-based, edge-based, signal-based, and SIFT-based features for the application of visual stimuli discovery in the next sections. With considering these features exclusively, we yield a general improvement in classification compared to the F_1 scores of the random forest classifier from Table 5.2 with +0.8% ± 2.3% for visual same and +2.9% ± 5% for visual change. The high deviation is caused by the results for WebMD. Its F_1 scores for visual same increases about eight percent and visual change about eighteen percent. See Appendix D for details about the classification performance when considering only the important features.

5.4 Visual Stimuli Discovery on Web Sites

We execute the framework of visual stimuli discovery using the random forest classifiers trained per Web site with the important features and the labels of the session of the first user. This reflects a realistic scenario for a usability analysis — labeling of one user session for visual change to discover the visual stimulus across all user sessions. The following process is performed for each fixed element and the scrollable part of the Web page independently — similarly to the enhanced representation. In each frame, we crop the regions which are occupied by a fixed element. A frame is then divided into one scrollable region and potentially multiple regions of fixed elements. Thus, regions of a frame might occur in different visual stimuli, yet no region, i. e., pixel data, of a frame, is represented in multiple visual stimuli.

First, we split the video recordings into stimulus shots. See Figure 5.6a for an exemplary excerpt of the splitting procedure. For the sake of the example, there are no fixed elements present on the Web page. All frames contain scrollable regions, only. (a) Initially, the first frame is considered the current stimulus shot. Then, we take the next frame, (b) normalize scrolling, and (c) compute the features on the overlap


Video Recording
(a)
1
Frame 0
1
Frame 1
2
Frame 2
…
(b)
Normalize scrolling and crop overlap
1
Frame 0
1
Frame 1
Example: No visual change
(c)
Compute features
Classifier of visual change
δ
No
Yes
Compare frame 2 with stitched frame of latest stimulus shot
…
(e)
Stitch frames and continue latest stimulus shot
Separate into two stimulus shots
(d)
Stimulus Shot
(f)
Stimulus shot now covers frames 0 to 1 from the video with a single stitched frame
1
Stitched Frame
Stimulus Shots (potentially from different users)
(a)
1
2
1
Stimulus Shot 0 [Frame 0-1]
Stimulus Shot 1 [Frame 2-5]
Stimulus Shot 2 [Frame 6-8]
For each pair of stimulus shots
θ
(c)
(b)
Compute features
(on overlap)
Classifier of visual change
δ
Yes
No
Zero
A
Area of overlap
Similarity Scores
(d)
Fill into matrix
Area of overlap of stitched frames from stimulus shot 0 and stimulus shot 2
Merge visual stimuli iteratively and update matrix
(e)
Pair with maximum similarity is merged into one visual stimulus

	0	1	2
0	-1	0	448
1	0	-1	0
2	448	0	-1

(a) Splitting a video recording into stimulus shots. **(b)** Merging stimulus shots into visual stimuli.

Figure 5.6: Application phase of the visual stimuli discovery by example. Initially, all video recordings are split into stimulus shots. Then, the stimulus shots are merged across user sessions toward visual stimuli.

of both frames. If the classifier of visual change detects a change, (d) the current stimulus shot is closed, and a new stimulus shot is started with the currently processed frame. If the classifier does detect no change, (e) the currently processed frame is put into the current stimulus shot. (f) The frames of the stimulus shot are stitched and considered for the next frame in the detection of visual change. After the splitting, each video recording of user sessions is partitioned into a series of stimulus shots.

Second, we perform agglomerative clustering with a single linkage on the set of all stimulus shots. Analogously to the splitting process, the stitched images of stimulus shots that are found to be similar during clustering are stitched together and treated as a single stimulus shot in the further course of clustering. In the end, visually coherent stimulus shots have been merged into visual stimuli. See Figure 5.6b for an example of the merging procedure. (a) We consider each pair of stimulus shots and (b) compute features on the overlap of their stitched frames. The visual features are fed into the classifier of visual change. (c) If a visual change is detected between the stitched frames of a pair of stimulus shots, a score of zero is assigned as their similarity score. If there is no visual change, the pixel area of overlap of the stitched frames is assigned as their similarity score. (d) All similarity scores are entered into a matrix that stores pairwise similarities between entries. Initially, all entries belong to pairs of stimulus shots. When a pair of stimulus shots are merged, the merged stimulus is considered instead, and the stitched frame of the merged stimulus shots is used for the computation of further similarity scores. (e) We iteratively take the entry with the highest score and merge the corresponding pair of stimulus shots. The procedure does stop at a predefined threshold of similarity that no entry in the matrix exceeds. The remaining stimulus shots – regardless of whether they have been merged or not – are considered as visual stimuli, represented through their stitched frames.

Many Web sites contain animations of dynamic elements like in- and out-fading menus or moving photos in photo carousels. The duration of the animations is usually below one second and triggers a visual change classifier more than once during the animation. We argue that any stimulus shot that lasts below one second should not contribute to the discovery process, as those stimulus shots would produce over-segmented results. Ignoring even longer stimulus shots, however, would exclude fast menu interactions. Therefore, we merge stimulus shots below one second to the temporal closest stimulus shot that is at least one second long without stitching the pixels of the shorter stimulus shot.

We have implemented the framework[5] in C++, using OpenCV [25] and the Shogun machine learning library [175]. Applying the framework to our dataset, we discover 2,742 visual stimuli for the twelve Web sites across the four user sessions each. See Appendix E for an example of the visual stimuli discovered in the video recordings of the user sessions on NIH.

In the next two sections, we evaluate the outcome of the visual stimuli discovery on twelve Web sites from a computational perspective and a user-centered perspective. For this, we come up with two hypotheses, analogously to the previous chapter:

- **(H1) Accuracy:** The discovered visual stimuli represent a Web page as accurate as video recordings.
- **(H2) Scalability:** The discovered visual stimuli allow a more efficient behavioral analysis than the video recordings.

5.5 Computational Evaluation of the Visual Stimuli Discovery

To objectively measure the effectiveness of the visual stimuli discovery in reducing the information overload, we compare the discovered visual stimuli with the original videos they are composed of. This

[5] `https://github.com/eyevido/visual-stimuli-discovery`

Table 5.5: Overview about the discovered visual stimuli and their quality per Web site. # denotes "number of." * denotes the metrics for the visual change classifier we used to execute the framework. ↑ denotes that a higher value is better. ↓ denotes that a lower value is better.

Web Site	Walmart	Amazon	Steam	Reddit	CNN	Guardian
# Video Frames	4,524	5,064	5,466	3,645	3,855	3,823
↑ Vis. Same [F_1]*	92%	93%	93%	97%	89%	97%
↑ Vis. Change [F_1]*	85%	82%	80%	78%	50%	81%
↑ Agg. Coverage*	0.91	0.95	0.97	0.95	0.78	0.88
↓ Agg. Overflow*	0.06	0.07	0.13	0.06	0.07	0.11
# Visual Stimuli	135	189	186	37	60	32
↓ Pixel Ratio	4.22%	5.64%	4.84%	2.52%	2.9%	2.65%
↑ Correct Frames	97.18%	95.85%	83.83%	97.31%	99.18%	67.06%

Web Site	NIH	WebMD	MayoClinic	GM	Nissan	Kia
# Video Frames	2,623	3,736	2,904	3,011	4,507	2,152
↑ Vis. Same [F_1]*	95%	94%	98%	97%	98%	98%
↑ Vis. Change [F_1]*	88%	78%	91%	84%	93%	87%
↑ Agg. Coverage*	0.95	0.94	0.99	0.95	0.98	0.95
↓ Agg. Overflow*	0.08	0.14	0.05	0.11	0.15	0.15
# Visual Stimuli	30	100	38	39	82	22
↓ Pixel Ratio	1.9%	3.72%	2.32%	2.6%	3.6%	2.08%
↑ Correct Frames	99.89%	86.05%	96.70%	96.21%	89.51%	99.16%

allows us a computational point of view in the evaluation of the framework.

Methodology We have created a tool[6] with a two-column interface, where the left column displays a frame from the video recording and the right column displays the visual stimulus by which the frame is represented. The visual stimulus is cropped to the portion that corresponds to the viewport in the video. See Figure 5.7 for a screenshot of the tool. Using the tool, an annotator iterates through the frames of each video recording and decides whether a frame is correctly represented by the corresponding visual stimulus, according to our definition of visual change from Figure 5.5. See Appendix F for examples from the evaluation process.

[6] `https://github.com/eyevido/visual-stimuli-discovery/tree/master/code/src/exe/Finder`

Results We provide the results of our computational evaluation in Table 5.5. We include the metrics of F_1 score, coverage, and overflow for each visual change classifier that has been trained for the respective Web site.

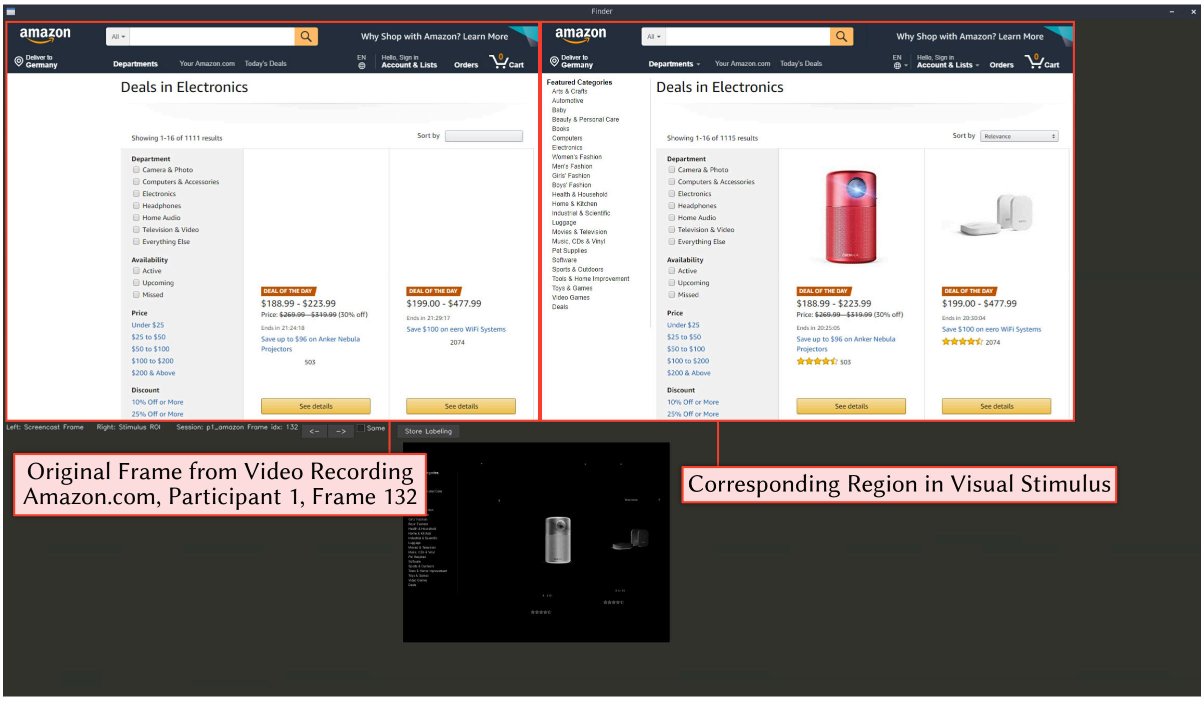
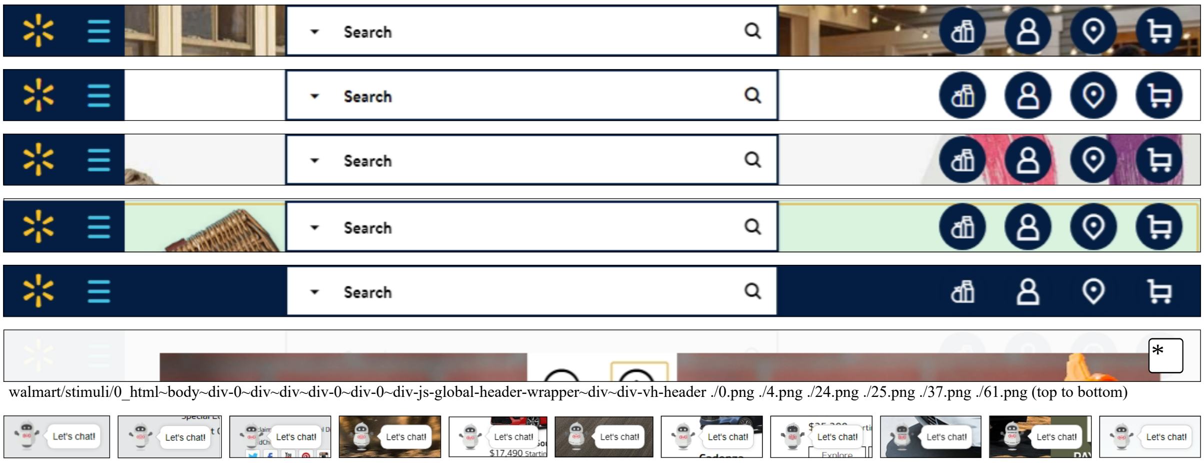

Finder
Left: Screencast Frame Right: Stimulus ROI Session: p1_amazon Frame idx: 132
<-
->
Same
Store Labeling
Original Frame from Video Recording
Amazon.com, Participant 1, Frame 132
Corresponding Region in Visual Stimulus

Figure 5.7: Screenshot of the tool that is used to check the representation of every frame from the video recordings in the corresponding visual stimulus. We have added a difference-image between both grayscale images on the bottom of the interface to support an annotator in her decision-making.

Fixed elements are treated separately from the scrollable page content in the visual stimuli discovery. This would allow us to correctly aggregate gaze data from multiple users at different scrolling offsets like in the enhanced representation. We determine fixed elements by querying the DOM tree during recording and retrieve the position and size. Then, we crop the regions from the video frames accordingly. In Figure 5.8, two common issues with fixed elements in the visual stimuli discovery are depicted. On the top, we display visual stimuli from the top menu bar of Walmart. The background of the bar is transparent, such that the content behind the bar is visible. When a user scrolls the Web page, the bar itself stays one the same position, keeps its size and its contained elements do not change. However, the scrolling contents behind the bar are visible through the bar and change the color values inside the bar dramatically. This causes our method to discover additional 105 visual stimuli just for the top menu bar of Walmart. On the bottom, we show visual stimuli from a fixed and animated chat avatar on Kia. Again, the contents behind the animated avatar are visible due to the rectangular crop. The animation and the transparency make the framework discover 669 additional visual stimuli. We think that the rectangular cropping of fixed elements is not sufficient to discover accurate visual stimuli on fixed elements. Therefore, our framework of visual stimuli discovery does not work for fixed elements on Web pages — why we exclusively focus on the page body of a Web page in the computational evaluation. In this regard, we have excluded the 1,792 visual stimuli of fixed elements from the computational evaluation that is reported in this section. Nevertheless, the visual stimuli of fixed elements are still included in the user-centered evaluations in the next section. The results of the user-centered evaluations indicate that the high number of visual stimuli from fixed elements in the visual stimuli discovery do not hinder common use cases like AOI analysis.



walmart/stimuli/0_html~body~div-0~div~div~div-0~div-0~div-js-global-header-wrapper~div~div-vh-header ./0.png ./4.png ./24.png ./25.png ./37.png ./61.png (top to bottom)

kia/stimuli/2_html-ng-app~body~div-6 ./0.png ./4.png ./15.png ./21.png ./39.png ./41.png ./99.png ./107.png ./210.png ./217.png ./263.png (left to right)

Figure 5.8: Fixed elements like menu bars or chat avatars generate too many visual stimuli because of transparency and animation. *The Walmart top menu bar can even fade out. This screenshot was taken a moment before it vanished.

Discussion Initially, we want to investigate how much information overload can be reduced by our method. We measure the reduction in information overload by comparing the number of pixels in the video recordings versus the number of pixels in the visual stimuli. The grand mean of pixel ratio across all portions is 3.25% ± 1.17%, indicating that our method can tremendously reduce the pixel numbers, which proportionally would reduce the visual inspection effort by a usability expert. This supports our second hypothesis (H2), i. e., the behavioral analysis using visual stimuli instead of video recordings can be more efficient. Assessing correctness, the results indicate that on average 92% ± 9% of the frames were represented correctly in their respective visual stimuli. This supports our first hypothesis (H1), i. e., discovered visual stimuli can represent a Web page as accurately as the video recordings.

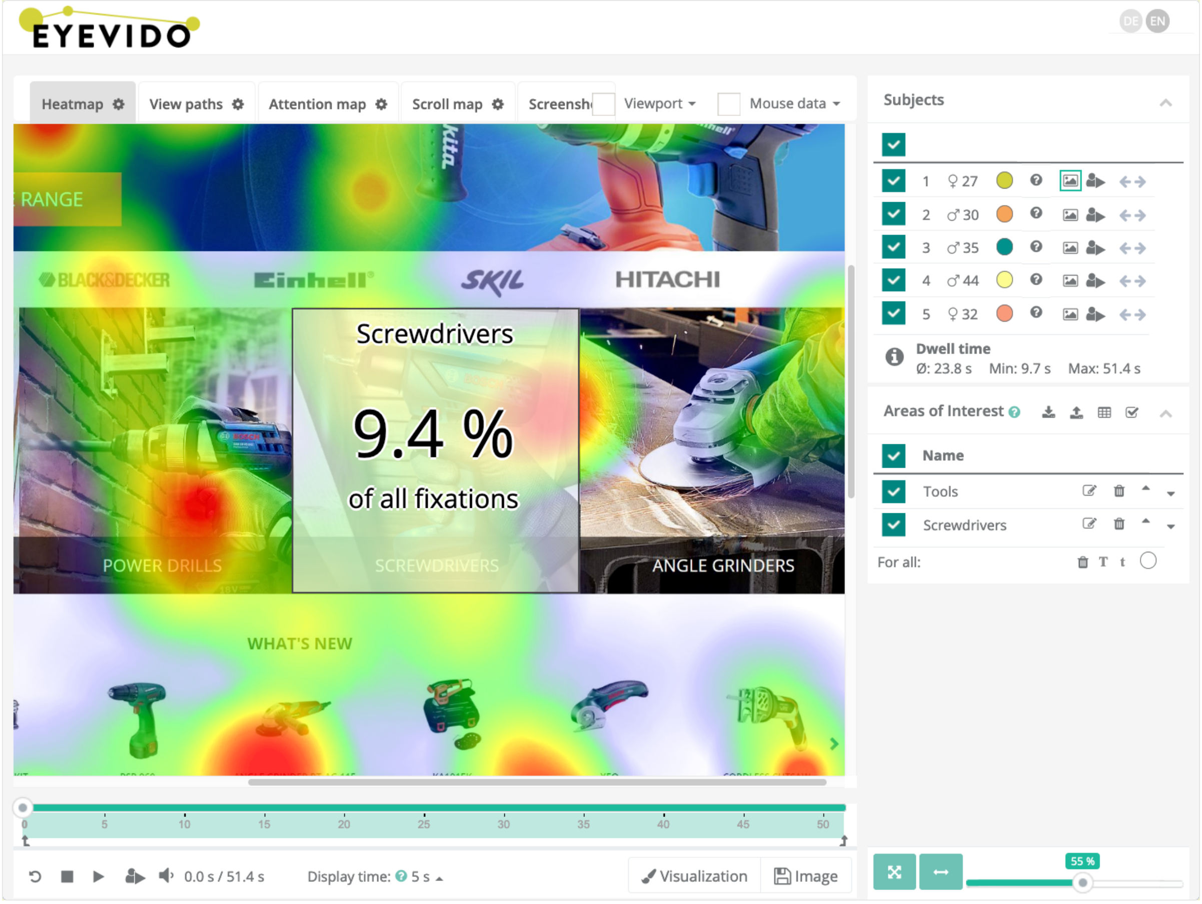
However, for Steam, Guardian, WebMD, and Nissan, the number of correct frames is below ninety percent. On Steam, we found that the order of pictures has been randomized in each user session. Thus, many frames were wrongly merged into a similar-looking layout, yet contain visually different pictures or at least shuffle the positions of the pictures in the layout. The Guardian changed articles and respective photos on the front page during the six hours of dataset recording. WebMD displayed personalized advertisements in each user session. Furthermore, contents on WebMD were shuddering during the page loading, hence the elements that varied in size impacted the entire layout of the pages on WebMD. Nissan displayed photo carousels with only a slightly different look, which the visual change classifier struggles to distinguish. There is some noise in the results across all Web sites introduced by animated advertisement banners and users who highlighted text spans with the cursor.

5.6 User-Centered Evaluation of the Visual Stimuli Discovery

In this section, we take the point of view of a usability expert on visual stimuli discovery. Thus, when we talk about a user-centered evaluation here, we mean a usability-expert-centered evaluation.

Experts perform usability analysis in software suites like Tobii Pro Studio [182], RealEye.io [148], or EYEVIDO Lab [62]. Those software suites integrate the recording of user sessions on Web sites, the data processing, and the attention analysis into a single workflow. The tight integration of the workflow is necessary due to the amount and specificity of the collected data. There are yet no standard formats to store gaze data or stimuli representations. However, this proprietary handling of data renders it impractical to plug the method of visual stimuli discovery into existing software suites. Therefore, we focus on the user-centered evaluation on the part of the workflow which may experience the highest benefit through the application of the visual stimuli discovery: The definition of AOIs [82]. An AOI is an area on a stimulus that is of special interest to a usability expert. The area might cover a headline in a newspaper article, a photo in an advertisement banner, or a button to put a product into the shopping cart. AOIs can have an arbitrary shape, but most commonly they are defined as rectangles. A usability expert can analyze gaze and mouse data within the extent of the AOI. This allows answering questions like “How long until a user has seen this area the first time?,” “How many users have looked at this area?,” “How many users have clicked on this area?,” and “How long have users looked at this area in total?” See Figure 5.9 for an AOI marked on a screenshot of a shopping Web site in the EYEVIDO lab. The AOI lays over the button that leads to the available “Screwdrivers.” The software automatically calculates that 9.4% of all fixations by the five recorded users on that Web page lay within the extents of that AOI. When multiple AOIs are marked on a stimulus, the sequence of visited AOIs can be computed and questions like “Did users first look at the product or the price tag” can be answered.

However, the definition of AOIs is a potentially tedious task. In current tools, stimuli are recorded as images or as videos. Marking of AOIs on images is a straightforward task. A usability expert can choose the shape of an AOI and mark the respective area on the image. But dynamic Web pages cannot be easily represented by images, as discussed in Section 3.3. Therefore, usability experts must mark the AOIs on the video recordings of each user session individually. As content in the interface may change its visibility throughout the video, a usability expert must manually label the visibility changes that affect an AOI. Furthermore, content may move in the viewport. Moving content can be either caused



☑	1	♀27	●
☑	2	♂30	●
☑	3	♂35	●
☑	4	♂44	●
☑	5	♀32	●

Dwell time
Ø: 23.8 s Min: 9.7 s Max: 51.4 s

☑	Name
☑	Tools
☑	Screwdrivers

Figure 5.9: An AOI on the screenshot of a shopping Web site in the EYEVIDO lab [62].

by animations of single elements or by a user scrolling the Web page such that its entire content moves up or downwards in the viewport. In such a case a usability expert must transform the AOIs on a video recording of the viewport accordingly. The precise marking of AOIs is a task that does scale linear with the number of users, as a usability expert has to perform the task on every video recording — like shown in Section 4.3.

The method of visual stimuli discovery attempts to abstract the dynamics by clustering visually coherent states of an interface, such that the generated images can correctly represent those states as visual stimuli. A usability expert might then browse through the generated images instead of the video recordings and mark AOIs on these images. We argue that this can reduce the workload of a usability expert tremendously. Therefore, we evaluate this claim in a user-centered evaluation. First, we identify exemplary AOIs on Web sites from our dataset which would be of relevance for a usability expert. The visual stimuli discovery relies on the performance of the visual change classifier, which directly influences the correctness of the visual stimuli as discussed in the previous section. Therefore, we perform a comprehensive analysis of whether the exemplary AOIs on the visual stimuli would report correct measures. Second, we have presented the exemplary AOIs as analysis tasks to usability experts in an interactive online survey. The survey results substantiate show that our method is novel and can support the workflow of usability experts. The entire user-centered evaluation also addresses our overarching research questions about the accuracy and scalability of the visual stimuli discovery.

5.6.1 Analysis of Exemplary AOIs using Visual Stimuli

The Web sites and the AOIs have been chosen with a heuristic based on the relevance of elements, i. e., elements that are an integral part of the Web sites that most users have encountered. The study aims to gather task-specific insights, rather than a comprehensive assessment of visual stimuli; hence, the outcome might depend on the choice of Web sites and the respective AOIs.

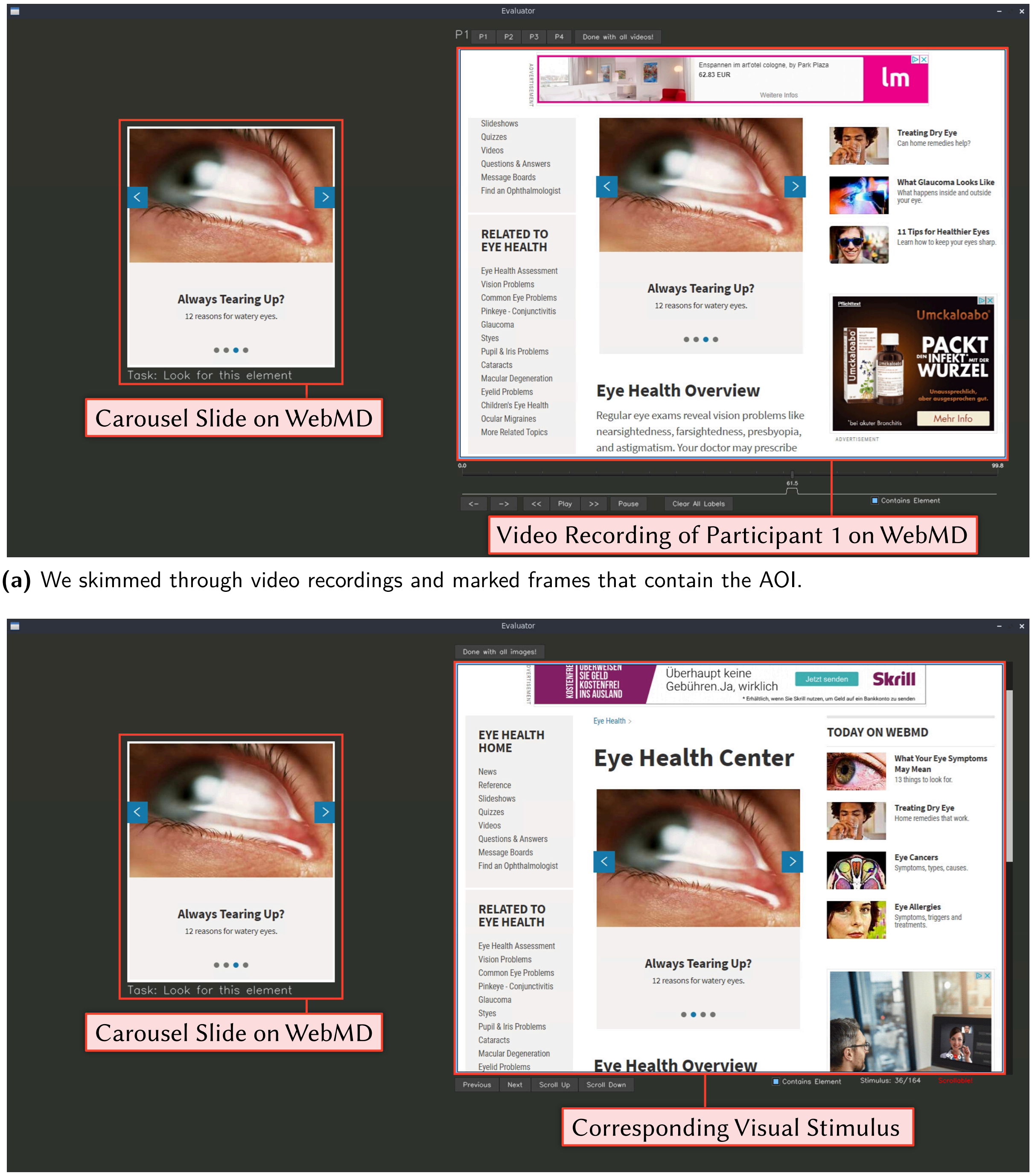
Methodology We have chosen four Web sites from the dataset, with one AOI each. First, we have labeled the occurrence of the AOIs on the frames of the video recordings and the visual stimuli independently. See Figure 5.10 for screenshots of the tool[7] for the labeling process. Second, the frames represented by each marked visual stimulus have been checked whether they show the AOI. If the visual change classifier does miss a visual change in the interface, the frames from different visual states can be incorrectly aggregated into the same visual stimulus. See Figure 5.11 for a screenshot of the tool[8] we developed for that check.

Results Figure 5.12 shows the results of our analysis. For each AOI, it includes the number of frames that are represented by the visual stimuli which contain the AOI — and indicates whether those frames show the AOI (positive contribution) or do not show the AOI (negative contribution) in the video recording. Some frames of a visual stimulus cannot contain the AOI, as a user had scrolled beneath or above the AOI in the video recording (neutral contribution). Also, we report the precision and recall values, calculated from the positively and negatively contributing frames. Next, we provide the rationale for the choice in each Web site and AOI and give a short discussion about the individual results.

(a) *Tile on CNN.* According to Table 5.5 on page 71 the CNN Web site shows a good score for overflow but a comparatively bad score for coverage. Potentially, too many visual stimuli are discovered. We have chosen a tile that is a hyperlink to the review of the “Captain Marvel” movie as the AOI. The tile occurs on the second Web page that all users browsed to on the CNN Web site. The tile is not loaded instantly, but a few milliseconds after the Web page has loaded. The minor deficits in the precision value could be due to the frames at loading and unloading the Web page since such frames do not show the tile but are still merged into the visual stimuli. The rather low coverage

[7] `https://github.com/eyevido/visual-stimuli-discovery/tree/master/code/src/exe/Evaluator`

[8] `https://github.com/eyevido/visual-stimuli-discovery/tree/master/code/src/exe/Preciser`


Evaluator
P1 P1 P2 P3 P4 Done with all videos!
Task: Look for this element
Carousel Slide on WebMD
Video Recording of Participant 1 on WebMD
<- -> << Play >> Pause Clear All Labels
Contains Element
Done with all images!
Corresponding Visual Stimulus
Previous Next Scroll Up Scroll Down
Stimulus: 36/164

(a) We skimmed through video recordings and marked frames that contain the AOI.

(b) We marked the visual stimuli, generated from the video recordings, that contain the AOI.

Figure 5.10: Screenshot of the tool used to mark the occurrence AOIs in the video recordings and in the visual stimuli. In this example, a carousel slide from WebMD is the AOI.

value from the overall evaluation does not harm the discovery, yet the tile is present on ten different visual stimuli — while being originally always on the same Web page and position.

(b) *Top Menu on Guardian.* In Table 5.5 on page 71 the Guardian Web site shows a low percentage of correct frames. We have chosen the top menu on the Web site as AOI, which can be collapsed. Here the recall is 100%. A minor dip in precision is because of a few frames in which the top menu has already been collapsed, but the frames are still merged into the visual stimuli. The difference between the good results for the Guardian Web site in this task compared to the computational evaluation is due to the task at hand. Whereas the computational evaluation looks at the entire frames of the video recordings, this task only focuses on the top menu. On the Guardian Web site, the visual change classifier does not detect a change due to a different photo behind the top menu; thus, our method merges the top menu into fewer visual stimuli. This affects the computational evaluation but works fine for this task.

(c) *Footer Menu on Walmart.* The Walmart Web site shows a very good result in Table 5.5 on page 71. We want to investigate how it correlates to the inspection of an AOI; thus, we choose a static footer menu that occurs across all pages of the Walmart Web site. That footer menu is well discovered with perfect precision and a very high recall. Even though the footer menu is part of the scrollable region, it is also contained in visual stimuli from fixed elements, as shown in Figure 5.12. Opening the fixed menu, it overlays the entire page behind the menu with a semi-transparent, dark rectangle. The rectangle is part of the fixed menu; thus, it lets us interpret the entire frame as fixed. The footer menu is still correctly discovered.

(d) *Carousel Slide on WebMD.* We report for both, the evaluation of a visual change classifier in the previous section and Table 5.5 on page 71 low performance on the WebMD Web site. We have chosen a carousel slide every user browsed to as AOI on the WebMD Web site. The precision of discovering the carousel slide is low at 42%. We found that other carousel slides and menu expansions have been mixed into the visual stimuli. This corresponds with the poor overflow value we report in the prior evaluations about the performance of the visual change classifier on the WebMD Web site.

Discussion Though the exemplary AOIs were defined in Web sites that were difficult for the visual change discovery, the errors did not affect the analysis of the AOIs too much. This outcome supports the first hypothesis (H1), i. e., the discovered visual stimuli represent a Web page as accurate as video recordings — as long as the visual change classifier works well. Similar to the enhanced representation in Chapter 4, the marking of AOIs would be faster on the visual stimuli than on the video recording. Specifically, the chosen AOIs are static on the visual stimuli but move in the image space of the frames in the video recording. A usability expert would have to compensate for different scroll offsets in the video recording by transforming the position and size of the AOI throughout the video recording. Therefore, we see our second hypothesis (H2), i. e., the discovered visual stimuli allow a more efficient behavioral analysis than the video recordings, substantiated by our assessment.

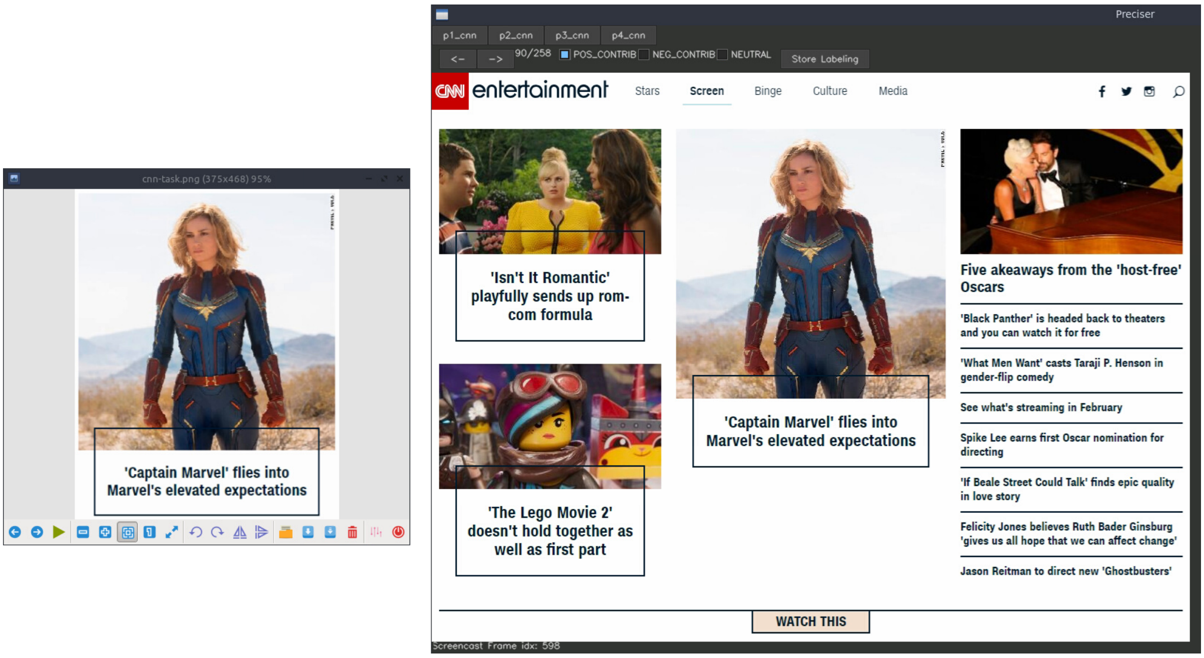

cnn-task.png (375x468) 95%
'Captain Marvel' flies into Marvel's elevated expectations
Preciser
p1_cnn
p2_cnn
p3_cnn
p4_cnn
<-
->
90/258
☑ POS_CONTRIB
□ NEG_CONTRIB
□ NEUTRAL
Store Labeling
CNN entertainment
Stars
Screen
Binge
Culture
Media
'Isn't It Romantic' playfully sends up rom-com formula
'The Lego Movie 2' doesn't hold together as well as first part
'Captain Marvel' flies into Marvel's elevated expectations
Five akeaways from the 'host-free' Oscars
'Black Panther' is headed back to theaters and you can watch it for free
'What Men Want' casts Taraji P. Henson in gender-flip comedy
See what's streaming in February
Spike Lee earns first Oscar nomination for directing
'If Beale Street Could Talk' finds epic quality in love story
Felicity Jones believes Ruth Bader Ginsburg 'gives us all hope that we can affect change'
Jason Reitman to direct new 'Ghostbusters'
WATCH THIS
Screencast Frame idx: 598

Figure 5.11: Screenshot of our tool we used to decide whether a visual stimulus correctly represents an AOI. Here, the AOI is a tile on CNN with a hyperlink to the review of the movie "Captain Marvel."

5.6.2 Expert Survey about Usefulness of Visual Stimuli

Finally, we report the results of an anonymous online survey, addressed to usability experts. See Appendix G for details about how we have integrated video recordings and visual stimuli interactively into the survey by developing a custom survey framework with JavaScript.

Methodology We hosted the survey on a Web server, consisting of eleven pages. On the first two pages, we asked for the demographics and expertise of the participant. We explained what we define as a dynamic Web page on page three. Then, we had three scenarios from the analysis of the exemplary AOIs on Walmart, WebMD, and the Guardian Web site. For each scenario, we had two pages in the survey. First, we pointed to the AOI and showed the four user sessions of the dataset as embedded video recordings. The participants were asked to skim through the video recordings and tell us, how they would tackle the analysis task for 20 users with their current workflow. Second, we had a gallery with the visual stimuli that contained the AOI from the respective Web site. The participants could zoom and pan the visual stimuli and display the scanpath and mouse traces over the visual stimuli. We asked the participant how accurate they believe an analysis would be when using our visual stimuli and whether they could save time in comparison to their current workflow. After the three scenarios, we asked for general feedback about our method on the tenth page and handled the submission of the survey on the last

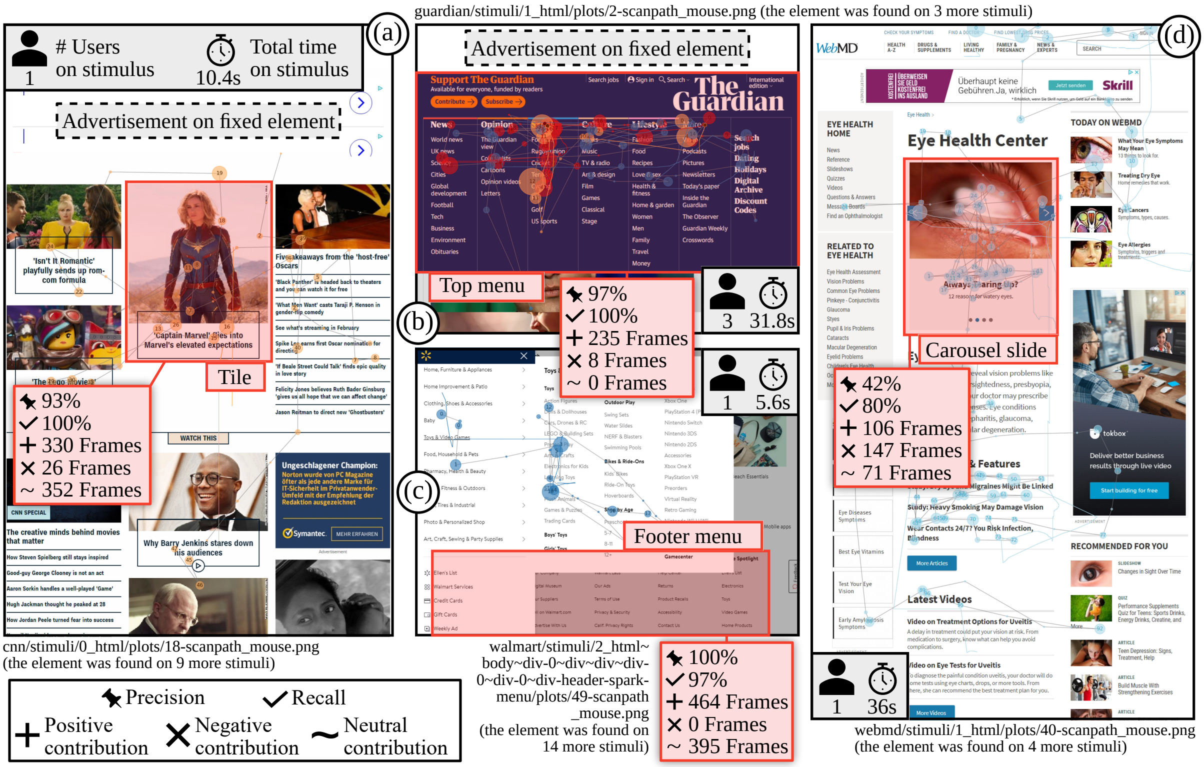

(a)
# Users on stimulus
1
Total time on stimulus
10.4s
Advertisement on fixed element
Tile
93%
100%
+ 330 Frames
× 26 Frames
~ 352 Frames
cnn/stimuli/0_html/plots/18-scanpath_mouse.png
(the element was found on 9 more stimuli)
Precision
Recall
Positive contribution
Negative contribution
Neutral contribution
guardian/stimuli/1_html/plots/2-scanpath_mouse.png (the element was found on 3 more stimuli)
Advertisement on fixed element
(b)
Top menu
97%
100%
+ 235 Frames
× 8 Frames
~ 0 Frames
3 31.8s
(c)
1 5.6s
Footer menu
walmart/stimuli/2_html~body~div-0~div~div~div-0~div-0~div-header-spark-menu/plots/49-scanpath_mouse.png
(the element was found on 14 more stimuli)
100%
97%
+ 464 Frames
× 0 Frames
~ 395 Frames
(d)
Carousel slide
42%
80%
+ 106 Frames
× 147 Frames
~ 71 Frames
1 36s
webmd/stimuli/1_html/plots/40-scanpath_mouse.png
(the element was found on 4 more stimuli)

Figure 5.12: Results of our analysis of exemplary AOIs. The AOIs (marked with a red box) are shown in one of the visual stimuli that contains them. Gaze data is plotted as scanpath and mouse cursor movements as lines. Each user is represented with a unique color.

page. We asked for participation via personal and E-mail-list invites; and public postings in respective LinkedIn groups.

Results We received eight responses, five female and three male participants (age = 35.9 ± 8.0 years). The professions of the participants ranged from three researchers (1, 2, and 7 years of experience), over a product owner (5 years), a usability expert (5 years), a sales associate (1 year), a consultant (23 years), to the head of an eye-tracking laboratory (8 years). Four of the participants have conducted more than 10 usability studies, three have conducted 4 – 10 studies, and one participant has conducted 1 – 3 studies. Conducting Web usability studies was the most common among participants, as seven votes were received for Web pages, five for mobile apps, four for desktop applications, and three for advertisements. All participants make use of gaze data for usability analysis, seven consider mouse clicks, and five touch input. Eye gaze is commonly analyzed with AOIs on the visual stimulus (among all participants), additionally, six participants also utilize heatmaps.

Given the three scenarios, we have asked the participants how they would tackle the given analysis

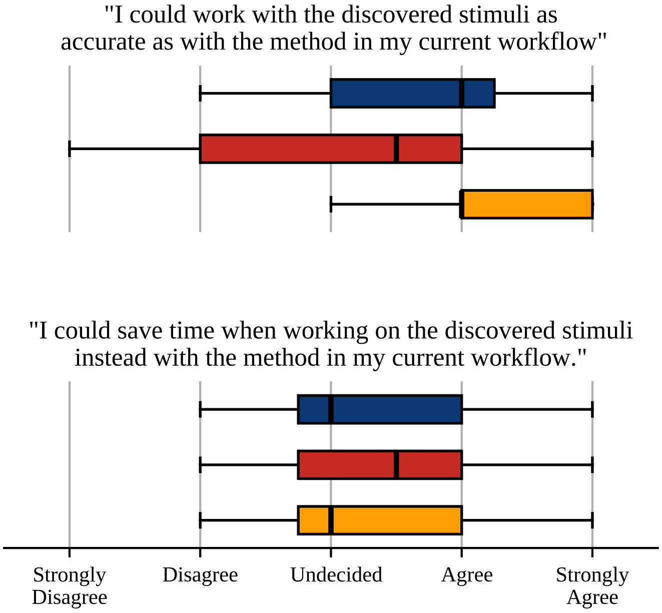


Figure 5.13: Ratings of the accuracy and potential for time saving of our method per scenario. 🟦 Footer Menu on Walmart. 🟥 Carousel Slide on WebMD. 🟨 Top Menu on Guardian.

task for 20 users with their current workflow. They have mentioned a variety of methods, i. e., four participants would use a video recording, two mention screenshots, four suggest a composed screenshot (like the enhanced representation), two would do a video recording with a camcorder, three would apply the "thinking-aloud" method, and one mentions a Live-DOM representation (recording of events and replay on the live Web site). Regarding our method, we asked the participants for each scenario about the perceived accuracy of visual stimuli and whether using them could save them time in analysis, each on a five-point Likert scale. The results are shown in Figure 5.13. The accuracy is rated high for the Walmart footer and the Guardian top menu; however, the WebMD carousel slide received a few negative votes. This might be caused by visual stimuli that show the fading process of the slides, yet are still considered to show the slide-of-interest by our definition of visual change from Figure 5.5 on page 62.

The general feedback about our method is shown in Figure 5.14 and demonstrates that experts would like to use our method for usability analysis. The acceptance of the semi-automatic mode emphasizes that our scenario of labeling one user session manually for visual change to train the visual change classifier appears to be realistic. We also asked the participants' opinions on the novelty of our method. All participants believe the method is novel, half of them state it is very novel, and half of them categorize it as relatively novel compared to the existing methods they use.

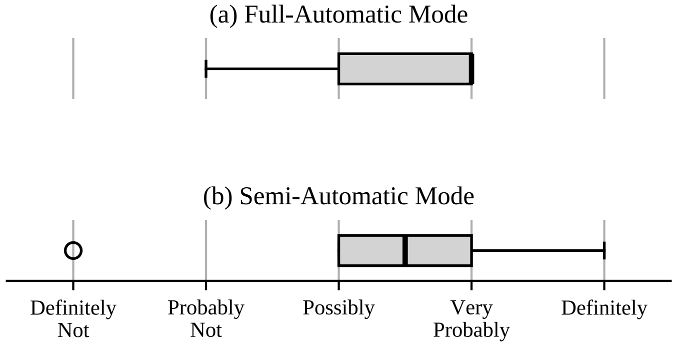


Figure 5.14: Feedback whether usability experts would employ the visual stimuli discovery in their workflow. We distinguish between (a) a full-automatic mode or (b) a semi-automatic mode. For the latter, a usability expert would have to manually label one user session, to train the visual change classifier.

Discussion The experts perceived the discovered visual stimuli as accurate as video recordings and stated that they would help them in saving time and effort, supporting our hypotheses (H1 and H2).

The experts were very welcoming about a method of visual stimuli discovery that can support their workflow. One participant left feedback that broadens the use-case beyond the aggregation of eye gaze and mouse data. According to his feedback, dynamics are *"especially challenging when synchronizing with additional measurements like GSR [Galvanic skin resistance], NIRS [Near-infrared spectroscopy], EEG [Electroencephalography] etc."*

5.7 Conclusion

Determining states in a dynamic Web page is an important aspect to allow for more efficient usability studies with a sufficient number of users. In this regard, we present a framework to discover visual stimuli in video recordings of user sessions, extending the idea of the enhanced representation to include dynamics of the contents. For this, we introduce the detection of visual changes in video recordings of user sessions on Web sites. We examine various features from scene detection literature that are deemed relevant in indicating visual changes and successfully train classifiers for recognizing visual changes on video recordings of real-world Web sites. The computational evaluation shows that the visual stimuli discovery can reduce the number of pixels significantly while preserving the correct representation of the user sessions. In addition, we estimate the usefulness in several case studies that mimic the task of a usability expert. We reinforce such findings with an online survey targeted at usability experts. The responses to the survey reveal that usability experts judge our method of visual stimuli discovery as accurate and possibly time-saving while being novel in their opinion.

Nevertheless, there is potential for improvement in the visual stimuli discovery. Aspects in the visual change classification, splitting of video recordings to stimulus shots, and merging of stimulus shots to visual stimuli might be altered, such that they are better suited to the task at hand.

First, we could investigate different approaches for a visual change classifier than a selection of computer-vision features and a random forest classifier. As of now, convolutional neural networks have become popular in image classification tasks. We could interpret the visual change detection as an image classification task, by feeding both frames to be compared into the convolutional neural network and output the decision whether there is a visual change between both frames. However, a convolutional neural network requires a huge amount of data for the training phase. One could either crowdsource the labeling process or use a synthetic dataset. The synthetic dataset could be comprised of crops from virtual screenshots from Web pages, for which the visual change decision could be assumed without manual intervention. The synthetic dataset could be used to pre-train the convolutional neural network, while a dataset with manual labels might then be used to tune the weights of the final network.

Second, we could reduce the labeling effort for a usability expert by suggesting incidents of visual change in a video recording. For this, we might employ a visual change classifier, trained on a plentitude of Web sites. We have already reported the performance of a classifier that is trained on Web sites from entirely different categories than the Web sites in the test data, see Table 5.3. The results may not be sufficient for direct application of the classifier, yet may be used to support a usability expert in the labeling effort.

Third, we could explore different approaches for merging the stimulus shots into visual stimuli. As of now, frames from the video recording, or each region of a frame when considering fixed elements, are represented in exactly one visual stimulus. However, it might be more useful for a meaningful aggregation of gaze data to have a region represented in more than one visual stimulus. In addition, the similarity score for the decision of which stimulus shots to merge is based on the overlap in their pixel area. This can be changed to another metric or a machine learning approach could be employed to decide about the merging order of stimulus shots toward visual stimuli.

Fourth, another interesting suggestion during the development of the framework had been to incorporate the interaction data, i. e., gaze and mouse data, into the process of visual stimuli discovery. As of now, the visual stimuli are only used to display gaze and mouse data onto them. But on the one hand, interaction data can also signify visual changes. For example, most visual changes do happen after a user has moved the mouse over a hoverable element or a user clicked on a menu icon. We might consider the interaction data as an addition to the video recording and perform a visual change classification in parallel on that data, e. g., by feeding the interaction data also into a convolutional neural network. On the other hand, one could argue that only visual changes that have been perceived by a user should be considered by the framework. This could be achieved by using the gaze data to divide a frame into a foveated region and a peripheral region. Features of visual change could be then weighted differently in each region.

Still, some Web sites will remain difficult to be processed such that gaze data can be aggregated across user sessions in a meaningful way. Especially premium product presentations in Web stores – like Apple did for the iPhone XS[9] – make use of advanced dynamics at scroll events. The Web page interprets the scroll offset as a time indicator for viewport-relative videos. A user can scroll to play the video, stop scrolling to pause the video, and scroll up to revert a video. In this case, the stimulus in the viewport constantly changes. But a time-wise synchronization between users remains difficult due to the scrolling-driven playback, in contrast to normal videos.

In the future, we believe the usefulness of the visual stimuli discovery framework will further benefit from integration into existing analytical tools used by usability experts. Moreover, our method is not limited to the benefit of usability experts, but psychologists can also take advantage of the method in setting up experiments that investigate rich dynamic stimulus-response scenarios.

Takeaway To cope with the exploding number of Web sites and the consistently changing design trends and interaction patterns, in the future we need even more efficient methods for usability analysis. We envision an ecosystem that considers the Web page document, the rendering of the Web page during a user session, interaction data, and prior decisions of usability experts on visually similar Web pages, to automatically judge the usability of a Web page — and even to report the issues in the design to the respective developers. This ecosystem would require a deeper understanding of the elements on a Web page and across Web pages. Especially compound elements – that are crafted from multiple elements but appear as a consistent element to a user – are required to be identified and tracked. Those compound elements appear to a user as photo carousels, expandable menus, product views, geographical maps, or chat widgets. We suggest detecting those compound elements across the discovered states of the interface on a Web site and attempt to understand their internal state-machine, e. g., the photos in a carousel or the options in a menu. This will help to find the compound elements across Web pages, as menus or product views might be consistent throughout the entire Web site. Finally, we might be able to divide a visual stimulus into portions of reoccurring layouts across Web pages and specific contents on a Web page, or even only a single state, which would allow an even more insightful aggregation of gaze data. In the end, this level of understanding of a Web interface could allow us to make the behavioral analysis on highly personalized Web sites like social media services more efficient and feasible for a bigger audience.

[9] `https://web.archive.org/web/20190104023835/https://www.apple.com/iphone-xs`, accessed on 27th June 2020.

Part II

Interacting with the Web using Eye Tracking

CHAPTER 6

Background of Interaction with Eye Tracking

In this chapter, we provide the background about how eye tracking can be used for interaction with interfaces, especially for people with motor impairment. First, we describe methods for atomic interactions like pointing and typing in Section 6.1, using eye tracking alone or in combination with other input modalities. Second, we give an overview of single-purpose interfaces that have been designed especially to be used with eye tracking in Section 6.2. Then, we present our framework for designing rich gaze-controlled interfaces in Section 6.3. Apart from our scientific contribution, we adapt the interface of Web pages and a Web browser for gaze input in the next chapter. Hence, third, we describe prominent Web browsers that comparably contribute to the field of Web accessibility in Section 6.4. Fourth, we define and explain interface semantics and how we can retrieve them in the context of Web pages in Section 6.5. Fifth, we explore how current end-user software combines pointing and typing using eye tracking toward emulation of mouse and keyboard in Section 6.6. We look at a common example of interaction through the emulation approach and investigate the issues with the emulation approach regarding the user experience.

6.1 Eye Tracking for Interaction

Eyes play a major role in our daily lives by letting us inspect the surrounding environment, perceive and understand the objects in our environment, and guide actions we want to perform. In addition, eye gaze is a natural means of communication between humans [97]. Bolt had already proposed in 1981 to extrapolate this communication toward human-computer interaction [24]:

> *"At the user/observer interface level, interactivity is extended to incorporate where the user is looking, making the eye an output device."*

In human-computer interaction, eyes are used in their natural role to inspect an interface, i. e., to perceive the contents (e. g., reading text), or to guide the interaction when performing *input actions*, i. e., when interacting with a specific interface element to achieve the associated functionality. In conventional interaction, input actions are effectively achieved with selections using motor responses through an explicit interaction channel (e. g., mouse, keyboard, or touch), and eyes are used in parallel to guide the control commands, by inspecting the interface (e. g., examining which element to select and observing the selection response). However, in gaze-based interaction, eyes are the only modality of interaction, i. e., they are additionally used to provide motor responses for desired input actions [125]; hence, the parallelism between inspection and selection cannot be achieved. The overload of inspection and selection is referred to as *Midas touch* [89]. In a naïve implementation of gaze-based interaction, the gaze triggers each interaction element that it dwells upon. In this scenario, a user would constantly select interaction elements by accident, while browsing through the available options. There are two fundamental approaches to deal with this overload and to avoid the Midas touch. Either, the conventionally parallel process of inspection and selection is sequentialized, allowing unimodal input with gaze only. More specifically, to perform any input action users sequentially need to inspect an interface to choose among the interaction elements (pre-select inspection), perform gaze-based confirmation (e. g., through a long fixation, a blink, or an eye gesture) [88] onto the desired element (selection), and inspect the effect of the selection (post-select inspection). Or, another input method is additionally used to perform selections, like a physical trigger or voice, allowing multimodal input that is potentially even more efficient and enjoyable than using an interface with traditional input devices.

However, an eye-tracking system provides gaze signals with limited precision and accuracy. As discussed in Chapter 2, the precision can be improved with filtering. But there is yet no universal approach to bring the accuracy of gaze signals on par with mouse pointing. Thus, interfaces must consider the limited accuracy of at most ±0.5°. The limited accuracy renders it infeasible to simply replace mouse pointing with gaze-controlled pointing or a keyboard with gaze-controlled typing. Both, the Midas touch and the limited accuracy make it inevitable to develop distinct interaction means to interpret gaze signals as input signals.

6.1.1 Selection Methods

The most intuitive method to select an interaction element using gaze alone might be the blinking of the eyes. A user could look at the interaction element to select, close one or both eyes for a short period, and the system would select the looked-at interaction element. However, there are various flaws with blinking as a selection method. First, it is not easy to distinguish between a natural blink and an intentional blink solely from the gaze signal. The user might blink with only one eye, which is not convenient. An intentional blink with both eyes would have to last longer than a natural blink, rendering it a tedious process. Second, even if blinking with one eye would be sufficiently convenient, the gaze signals are disturbed by blinking. Eye tracking does not work when the eyes are closed, that is why blinking has been considered infeasible for selection purposes [89].

Another simple but much more popular selection method is the so-called dwell time, i. e., if the duration of a user's fixation exceeds a predefined threshold, a selection is triggered [89]. The slightest discrepancy between a user's eye movements and what they see, feel, or hear can disrupt the experience they are engaged in. Hence, designers need to re-purpose the feedback mechanisms for sensory information from the eyes. The usage of adequate visual graphics and animation as feedback can assist users to discriminate between inspections and selections, to reduce error in interactive operations. The dwell time selection method can be improved with visual feedback about the dwelling progress, a preview of the effect of the selection in the area of the dwelling, and audio feedback. Moreover, dwell time can be adjusted to fit the user's expertise in the interface and may even be automatically deduced for distinct regions of an interface [36]. Usually, interfaces for gaze control consist of buttons that are triggered by a fixation that duration exceeds a preset dwell time. The buttons are sized to be accurately selected with eye tracking, contain text or a symbol, and often provide visual feedback about the dwelling progress and the selection. We call such an interface element *virtual button* in the remainder of this thesis.

Lately, eye gestures have become popular in research. Eye gestures are eye movements that follow a predefined pattern, similar to the pattern lock on phones. The benefit of eye gestures is that they are not relying on accurate gaze signals, because only relative distances of consecutive gaze samples are considered. Theoretically, eye gestures do not require a calibration of the eye-tracking system. However, eye gestures might be perceived as not natural when a user is asked to make eye movements without a corresponding stimulus. The eye is not used to move without an object to focus on. Thus, early attempts to use eye gestures for selection [154] or typing [45] without stimulus required a lot of training. More recent approaches made use of moving elements in interfaces as stimuli that a user could follow for selection [159]. Eye gestures bear a huge potential for all kinds of gaze-controlled interaction, as they allow gaze-based interaction with something as small as a wristwatch [61].

A lot of research has considered additional input devices for selection, leaving only the determination of the target to gaze input. Eye tracking was combined with a mouse [197, 53] or a keyboard [107]. Recently, also touch [152, 99] and voice [163] modalities were integrated with gaze.

6.1.2 Unimodal Interaction

Considering gaze as the sole input allows for hands-free interactions, even in noisy environments. Eye tracking as an input method may be used for public ticket machines, information monitors, clinical environments, and to improve the accessibility of interfaces in general. Interaction with gaze alone has to cope with the Midas touch problem and the limitations in the accuracy.

Eye Pointing

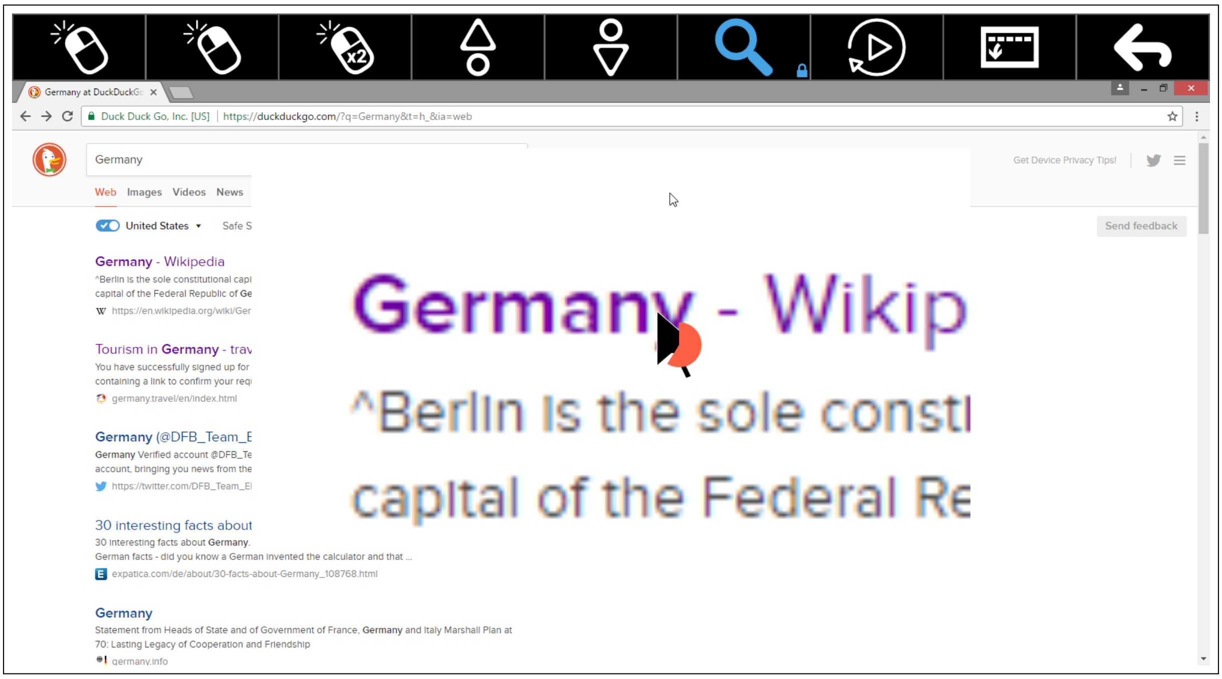
A simple but effective method for pointing with gaze is to employ a multi-step approach of dwell time and magnification. Lankford [116] proposed a four-step technique to emulate mouse events on a classical desktop system. The user first looked at the region which contained the target. After an initial dwell time, a window popped up near the center of the screen. In this window, the region around the fixation was displayed as a magnified image. The user then fixated on the coordinate in the window where she wished to have performed a mouse action. After another dwell time, a menu popped up, offering six different mouse actions as virtual buttons on the screen. Again, after a dwell time on the virtual button corresponding to the desired mouse action, the pointing process was completed and the corresponding mouse event was spawned. See Figure 6.1 for the similar implementation of the mouse emulation in the recent “OptiKey” software. Further works proposed different approaches for the magnification procedure, e. g., in the appearance of a fish-eye lens [8], yet the principle remained the same.

Besides magnification of the complete interface, there were also approaches to magnify or scale single elements of the interface. Špakov and Miniotas [176] proposed a system to select items in a standard-sized drop-down menu more accurately. The item on which the POR was sensed did vertically expand, while the text within the item stayed on the same screen position. When there was an inaccuracy in the estimation of the gaze signal, the actual POR of the user might have been on an item that was below or above the expanding item. The expansion of the item would have moved the desired item on the screen and a user’s gaze had followed that movement. This could be recognized as a relative shift in the gaze signals. Then the gaze signals were adapted accordingly, and the desired item could be selected.

Another approach was to separate pointing and selection not only timewise, but also spatially, which allowed for reducing the dwell times drastically. Lutteroth et al. [123] marked interaction elements, e. g., hyperlinks on a Web page, with unique colors. On the side of the screen, each color occupied a distinct area. A user could first look at the desired hyperlink and then look at the area on the side of the screen with the color corresponding to the marking color of the hyperlink in order to select the hyperlink.

Eye gestures may even eliminate the need for dwell time, why they are potentially more efficient to use. “GazeEverywhere” [159] by Schenk et al. was an approach using eye gestures to perform selections and improve the accuracy of the eye-tracking system. After the fixation at a POR on the screen, two moving dots appeared above and below the fixated coordinate. The movement of dots and relative gaze signals could be compared to decide whether the user follows one of the dots with their gaze. In case the user follows one of the two dots, a mouse click was spawned on the fixated coordinate. Furthermore, the orthogonal offset of the POR to the moving dot could be entered into an offset grid, which was used to improve the accuracy of the gaze signal. The moving direction of the dots did alternate between a horizontal and a vertical direction, such that the accuracy in both dimensions could be improved with the offset grid.

The characteristics of gaze as *“what you look at is what you get”* [89] had initiated the comparisons of gaze signals with other pointing mechanisms like a mouse, pen, or touch. The performance of gaze-based pointing is usually assessed with Fitts’ law [126], which evaluates the metrics of throughput and accuracy. Furthermore, the usability of eye tracking as an input method has been assessed as per device comfort ISO 9241-9 questionnaire [199], which is primarily used for traditional pointing devices like a mouse, pen, or joystick. In general, the performance of gaze input was found to be lower than conventional pointing mechanisms [125]. Therefore, various methods of eye pointing, e. g., multi-step magnification [16], fish-eye lens magnification [8], or smooth pursuit of visual targets [159] had been evaluated and compared employing time-required-for-pointing and achieved accuracy.



Germany - Wikipedia
^Berlin is the sole constitutional capi
capital of the Federal Republic of Ge
https://en.wikipedia.org/wiki/Ger

Tourism in Germany - trav
You have successfully signed up for
containing a link to confirm your req
germany.travel/en/index.html

Germany (@DFB_Team_E
Germany Verified account @DFB_Te
account, bringing you news from the
https://twitter.com/DFB_Team_E

30 interesting facts about
30 interesting facts about Germany.
German facts - did you know a German invented the calculator and that ...
expatica.com/de/about/30-facts-about-Germany_108768.html

Germany
Statement from Heads of State and of Government of France, Germany and Italy Marshall Plan at 70: Lasting Legacy of Cooperation and Friendship
germany.info

Germany - Wikip
^Berlin is the sole consti
capital of the Federal Re

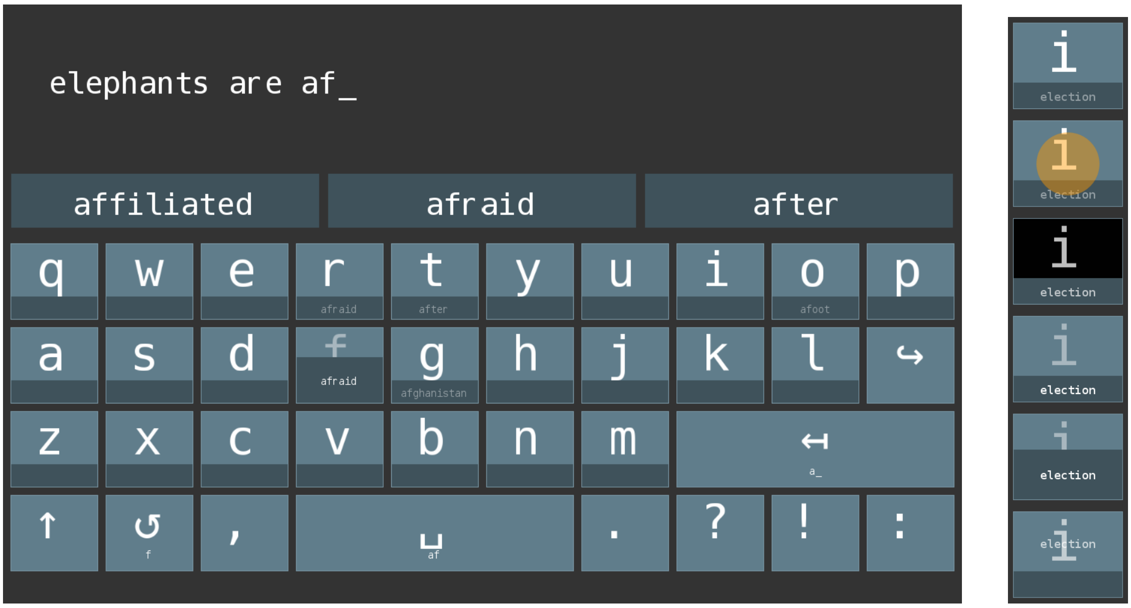
Figure 6.1: Pointing in the OptiKey software [179]. The pointing is implemented as a two-step dwell-time process. The first dwell time triggers a magnification, the second dwell time confirms the selection at the fixated coordinate.

Eye Typing

Lankford [116] introduced a gaze-controlled keyboard that displayed a virtual button for each key on the screen in 2000. The virtual buttons could be selected via dwell time. Additionally, a dictionary provided choices that matched with the characters typed by a user. The choices were displayed on further virtual buttons on the screen, which could be also selected using dwell time.

Over the last years, researchers put much effort to accelerate eye typing, e. g., with more integrated suggestions, to save on required dwell times. Diaz-Tula et al. [49] designed a keyboard that provided information in the foveal region, such that the eye movements could be reduced. They improved the throughput by augmenting keys with a prefix of the previous input and suffixes with suggestions, to save on inspection times. Additionally, they integrated word suggestions in a box on the right side of the keyboard. Recent works even suggested embedding word suggestions into the keys [165]. A pilot study revealed that 54.4% of all word suggestions were selected via gazing on the keys. One participant achieved even a maximum of 92.6% (Figure 6.2).

There were also methods to integrate eye gestures instead of dwell time into eye typing. Kurauchi et al. [110] designed a “EyeSwipe” keyboard that was inspired by swiping on touch-screen key-



(a) Suggestions were embedded to the keys and a multi-step dwell time was used for selections.

(b)

Figure 6.2: Eye-typing interface of "GazeTheKey" [165]: (a) suggestions were embedded to the keys and (b) a multi-step dwell time was used for selections of, firstly, a letter and, secondly, a suggestion.

boards. Eye gestures indicated the first and last characters of a word, while the in-between characters were entered by glancing over the keys in the keyboard. A gaze-controlled version of the well-known Dasher keyboard was also successfully evaluated [185].

The performance of the eye-typing methods has been evaluated with the metrics of words-per-minute, keystrokes, and error rate [167, 166].

6.1.3 Multimodal Interaction

Multimodal interaction usually employs gaze signals to pick one from the available targets, while the selection is confirmed with another modality. Such multimodal interaction must cope with the limitations of accuracy in eye tracking, but not with Midas touch.

Gaze signals were combined with a physical pointing device. Zhai et al. [197] developed a system that warped a cursor toward the POR and allowed for fine adjustments and selections with other input devices, e. g., relative mouse movements, and mouse clicks. They deduced that the pointing speed was like using the mouse only, however, the users subjectively tended to feel faster using their multimodal pointing technique. Further developments extended the approach with a touch-sensitive mouse [53].

Gaze signals were combined with a physical keyboard. Kumar et al. [107] proposed a look-press-look-release method. A user looked at the desired target. A keypress on the keyboard let the system magnify the area around the desired target. Then, the user could refine her selection in the magnified area and release the key, which confirmed the selection. This approach was very fast and reliable.

Gaze signals were combined with touch input. Pfeuffer and Gellersen [152] presented various combinations of gaze signals and touch input. They redirected touches to the looked-at target; thus, provided whole-screen-reachability, while only using a single hand for both holding the touch-screen device and touch-input selections. Further works investigated how to combine gaze and touch for PIN entry [99] or combined a keyboard using touch on a smartphone to start and to end the word entry, while characters could be selected via eye swiping [100].

Gaze signals were combined with voice input. Sengupta et al. [162] investigated different methods for selecting and correcting errors using voice and gaze. They attempted to overcome the unavailability of spatial information in voice-based interaction systems with gaze. The user's POR was used to select an erroneous word by (i) dwelling on a word, or (ii) by uttering a "select" voice command, or (iii) by saying an assigned number. The word could be corrected by either choosing from a suggestion or by spelling the word via voice input. They found the dwelling option to be the fastest method while requiring the least effort. The dwelling option also received the best subjective feedback from the participants of their study.

In this thesis, we focus on eye tracking as an input method for unimodal interaction. The unimodal interaction with eye tracking is primarily employed to make computers accessible to people with motor impairment, without relying on further input modalities.

6.2 Adapted Single-purpose Interfaces

In this section, we look at applications in which interfaces have been designed to be operated with gaze. We call such kind of interaction *direct* gaze control. These interfaces are mostly prototypes, which are limited in their functionality to a single purpose and are often not maintained after user studies. Nevertheless, the designs of these prototypes provide insights into the potential of interfaces that are assembled to work effectively with gaze as input.

6.2.1 Drawing with Eye Tracking

Drawing allows people to express their creativity and feelings. Especially, people with motor impairment, and, even more specifically, children with motor impairment, are considered as a target group for gaze-controlled drawing applications.

Hornof and Cavender [83] developed "EyeDraw," an application that enabled children with severe motor impairment to draw with their eyes. The authors proposed an interface with two states. One state

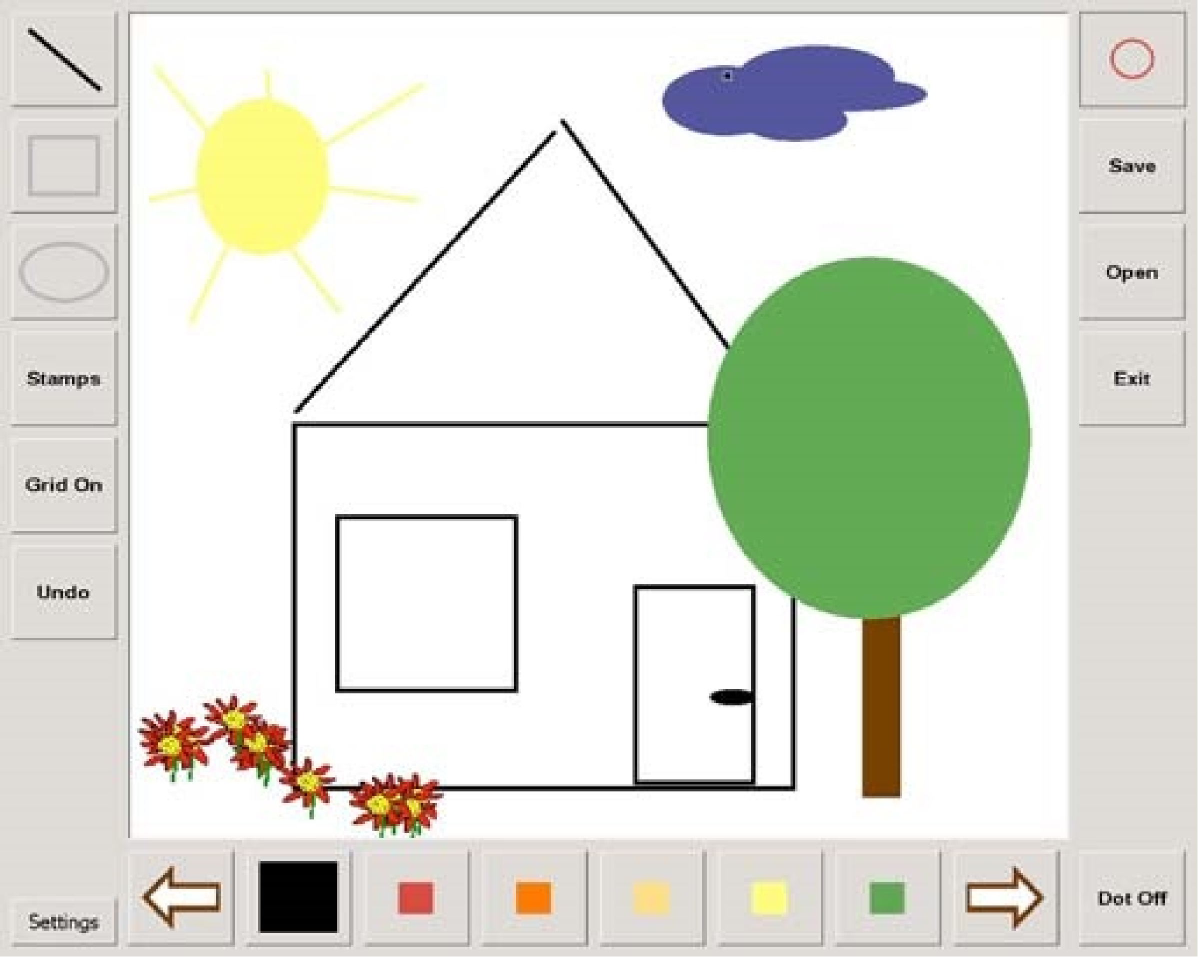


Figure 6.3: Interface of EyeDraw [83]

allowed for free looking and the other state allowed for drawing. A user defined the start and the end of a line through fixations; instead of drawing pixel-by-pixel like a brush using gaze. Transitions between the states could be performed via dwell time on the canvas. The application allowed for drawing lines, circles, rectangles, polygons, and stamps. The drawing tools were available as virtual buttons that surrounded the canvas, see Figure 6.3. A grid of dots could be displayed on the canvas to help a user fixating a POR. Furthermore, a dedicated mode for looking was implemented. Any dwell-time interaction on the canvas could be paused through the selection of a virtual button in the interface.

The idea of a drawing application with gaze signals was enhanced in multiple works. For example, Heikkilä [80] developed “EyeSketch.” The application allowed the drawing of shapes that could be moved, resized, and recolored after the initial placement of the shapes. The tools for the manipulation were controlled with eye gestures and blinking.

6.2.2 Writing with Eye Tracking

Inserting text for data processing, documentation, and communication is the traditional purpose of computers. Especially digital communication has been in the focus of gaze-controlled applications for people with motor impairment. There are various commercial grid-based applications on the market, see Figure 6.4. They offer virtual buttons in a grid-like interface. The virtual buttons represent common phrases, which allow the formulation of queries of everyday communication. There exist systems with a deep hierarchy of virtual buttons, customizable grid layouts, eye typing, and combinations with speech and touch input. See Figure 6.4 for an example of a commercial application with a grid-like interface.

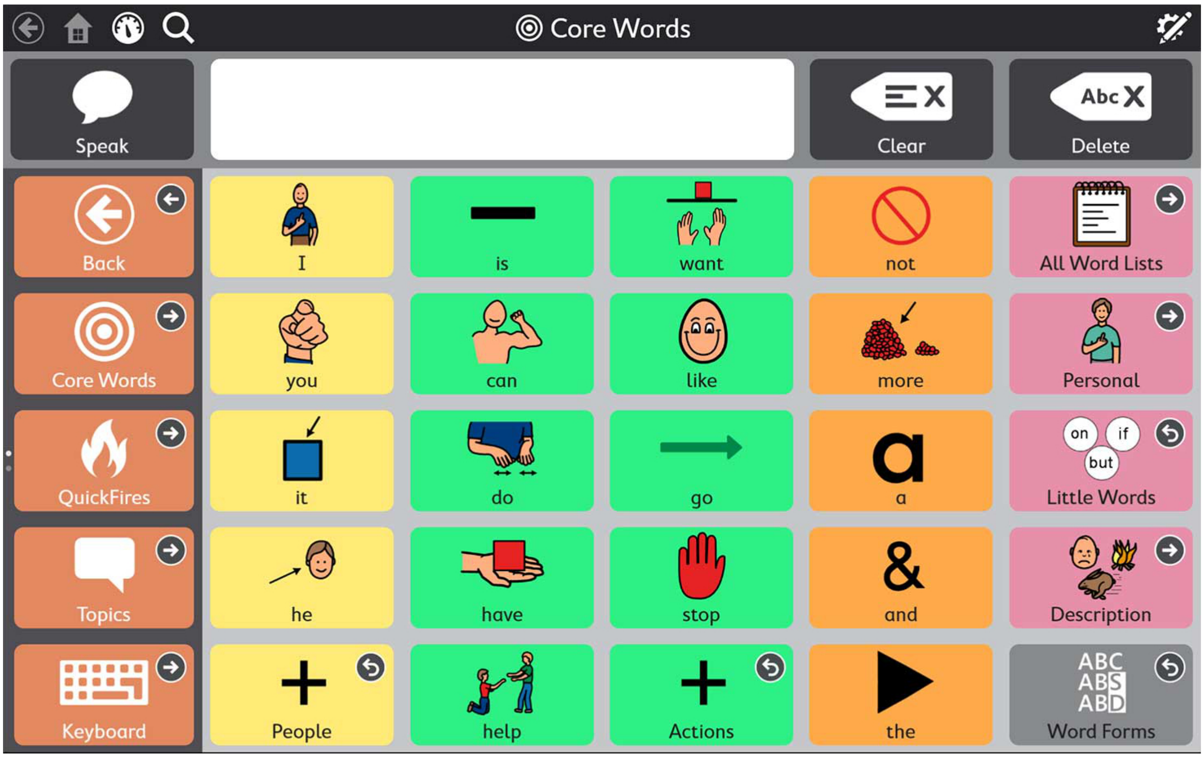


Figure 6.4: Interface of Tobii Snap + Core First for Windows[1]

Research in word processing using gaze signals has rather focused on multimodal interaction, as it bears huge potential for a broad audience. Beelders and Blignau [17] enhanced Microsoft Word with voice and gaze as input. They used voice commands for formatting, cursor movement, text selection with speech dictation; and gaze input for pointing, e. g., cursor placement in the document and eye typing in a virtual keyboard. The software was highly configurable, as different selection methods for eye pointing were available, e. g., dwell time, blinking, or a physical key-press. However, it was found that the voice- and

[1] `https://www.tobiidynavox.com/globalassets/pictures/software/snap/product-listing-images/scf-2.jpg`, accessed on 31st May 2020.

gaze-based interaction caused a significantly higher error rate than the traditional combination of mouse and keyboard.

Sindhwani et al. [171] developed “ReType,” a text editor for quick text editing with keyboard and gaze. The software was gaze-assisted and attempted to erase the need for mouse pointing in certain scenarios. It allowed common editing operations, while the hands could remain on the keyboard. The text editor was enhanced with a specific mode to retype. The mode to retype could be entered via a key or heuristics on the gaze signals. A user would start to type an edit string, which was matched with the entire text. Phrases with low string distance were highlighted as candidates to be replaced with the edit string. The user looked at the desired candidate and pressed a key to apply the edit.

6.2.3 Gaming with Eye Tracking

Games have the potential to introduce many people in a playful way to the interaction with eye tracking. Furthermore, the gaze signals of players can be used to gain a deeper understanding of human perception. However, it is often difficult to adapt games for gaze-controlled interaction due to their complexity and high frequency in input commands.

Games for Entertainment with Eye Tracking

Vickers et al. [188] evaluated locomotion tasks in games using eye tracking, in which an avatar could be moved over terrain by players. They divided the screen into rectangular zones that controlled the movement of the avatar upon fixation, e. g., moving the avatar forward or changing its walking direction. Their interface design allowed to steer an avatar in a conventional adventure computer game, like World of Warcraft. However, the interaction with eye tracking was slower and more error-prone than input by mouse and keyboard. In addition, games usually require a broader set of interactions, which was not covered by their approach.

The chicken shoot was a classic two-dimensional shooter game, in which the player had to hit flying chickens with a shotgun. Usually, aiming with the crosshair is performed through movements of the mouse, and shooting and reloading with the mouse buttons. Isokoski et al. [86] translated gaze signals into mouse and keyboard events and then injected these into the game. The crosshair was changed to follow the gaze. The shooting and reloading via mouse buttons were replaced with a switchable automatic machine gun. An off-screen target above the screen could be fixated to switch the gun on or off. After four to five trials, most participants outperformed the mouse and keyboard control condition in the final score. Especially, the faster positioning of the crosshair by gaze, compared to the manual positioning, improved their scores.

Isokoski et al. [86] also discussed “EyeChess,” developed by Špakov. Turn-based games and other games, which do not require a high frequency in input commands, can be easily adapted for interaction

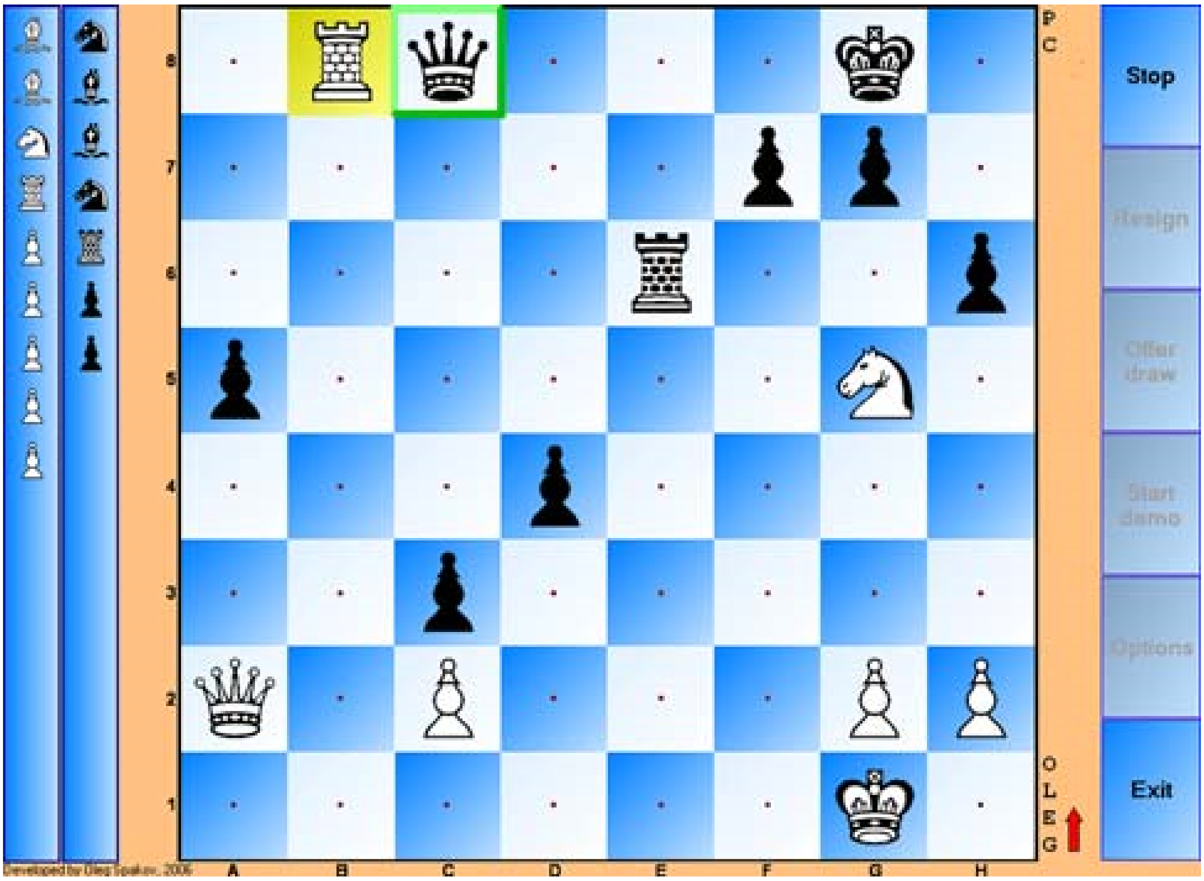
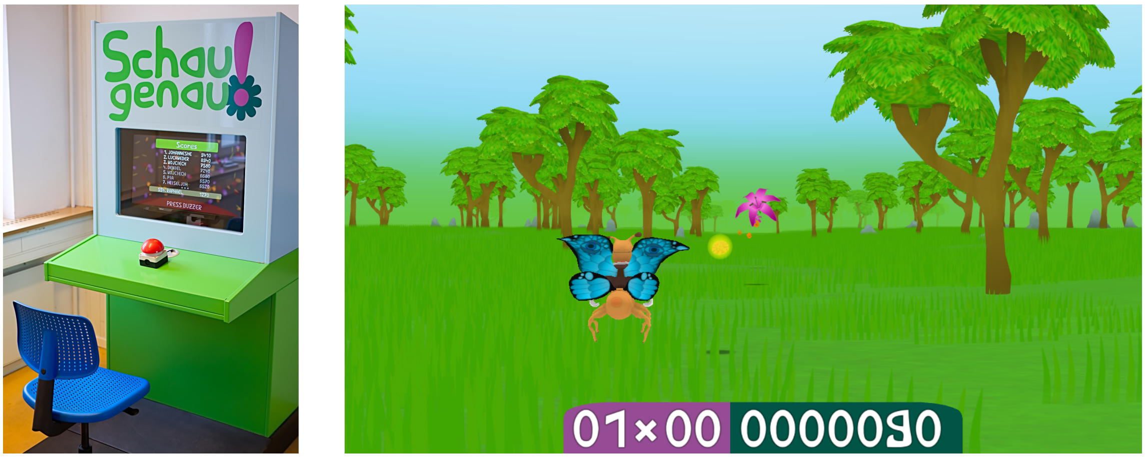


Figure 6.5: The interface of EyeChess [86]

via gaze input. However, the user experience can be significantly improved by adapting the interface itself for gaze-based interaction. The elements in the interface of EyeChess were made large enough to be selectable via dwell time without additional magnification. Furthermore, the elements provided feedback about selection and featured dots to focus on in their center. The changes in comparison to a mouse-operated chess application were minimal, yet they were evaluated as critical to allow a satisfactory control with eye tracking as the input method. See Figure 6.5 for a screenshot of EyeChess.

Games with a Purpose with Eye Tracking

In the context of research, games with a purpose are popular to offer knowledge to players and to retrieve feedback from them in a playful way. In combination with eye tracking, saliency information from gaze on photos can be gathered by letting users look at photos as part of the game. The saliency data may be useful to tag photos and improve image search in the future.



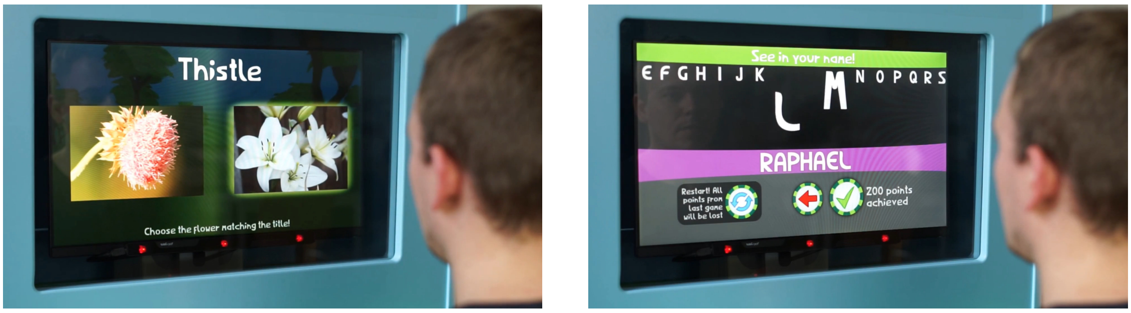
(a) Arcade-game like box **(b)** Screenshot from the gameplay

Figure 6.6: The Schau genau! [138] arcade-box was placed at a state horticulture show in Germany for half a year. Nearly 3000 complete user sessions were recorded.

Walber et al. [189] developed a two-dimensional gaze-controlled game, “eyeGrab,” for photo categorization. Players categorized several photos according to their relevance for a given tag. While the game was entertaining for players, the aim was to enrich the photo context information. Players were asked to look at photos falling down the screen. The task was to classify the photos as relevant or irrelevant to a given tag. A player could select a photo by fixation and could categorize the photo by dwelling on a corresponding virtual button. A player received points for each correctly categorized photo, negative points for each wrongly categorized photo, and no points for photos that fell off the screen without classification by the player. The speed of the photos that fell was increased over time, to increase the difficulty of the game.

We have developed a three-dimensional successor to the game, “Schau genau!” [158, 138].[2] In the game, the gaze signals are used to control an avatar in the appearance of a butterfly. See Figure 6.6 for the setup and a screenshot of the game. The task of a player is to collect flowers with the avatar to gain points. Flowers are spawned in the distance and the avatar flies constantly over a meadow toward the flowers. The interface of the game consists only of two panels on the bottom of the screen. The collected points are displayed in the green panel on the bottom right. The current multiplier is displayed in the purple panel on the bottom left. The game terminates when the avatar is caught by a spider web, which is spawned analogously to the flowers. During the game, both game speed and the density of the spider webs linearly increase, making the game more difficult. The research purposes of the game are threefold: we have integrated control styles with different levels of intelligence, we educate the players about flower species as the serious game part, and we collect gaze signals on photos of flowers.

[2] `https://github.com/raphaelmenges/schaugenau`


Thistle
Choose the flower matching the title!
See in your name!
E F G H I J K L M N O P Q R S
RAPHAEL
Restart! All points from last game will be lost
200 points achieved

(a) Picture mode of the game **(b)** Interface to enter a nickname

Figure 6.7: Schau genau! [138] game with eye-tracking device mounted below the monitor. One can see the three red illumination units of the eye-tracking device.

The controls of the avatar with gaze signals are of special interest in this game. The avatar moves on a two-dimensional plane, which is placed parallel to the screen. Three mappings of gaze-on-screen to the world position of the avatar on that plane, featuring different levels of intelligence, have been defined and were randomly assigned to different players. The first approach is the interpretation of the gaze on-screen as avatar position, like a mouse emulation. The second approach features a grid-based positioning, similar to the first approach, but with very coarse fixation filtering. The third approach supports the player by making use of the knowledge about the virtual world. The gaze is checked to lay upon a flower in the distance, and the avatar is moved toward the future collision position with that flower. The visual position of the flower in the distance and the future collision position are not the same, as the perspective projection moves objects in the distance closer to the center of the screen. When the gaze is upon the avatar, its opacity is reduced to enable a player to see what is behind the avatar.

The serious part of the game educates the players about flower species, who are rewarded with an increase of the multiplier for their knowledge. After a certain time interval, the game state switches from the normal game into the picture mode (Figure 6.7 (a)), in which one tag and two pictures of flowers are presented to the player. The player shall select the picture that depicts the flower which corresponds to the displayed tag. For each correct selection, the multiplier is increased. During this decision process, gaze signals on the two pictures are collected. These gaze signals may provide insightful details in the future, like which portion of the picture led the player to identify a flower's species.

In this immersive gaze-controlled game environment, several interaction elements were included with respect to the size, shape, and visual feedback. The photo in Figure 6.7 (b) shows the game screen for the player inserting a nickname for the high-score table. The alphabet is displayed on the top, where the player can scroll horizontally through the letters by fixating on a letter. The fixated letter moves toward the center of the screen and enlarges until a dwell time is reached and the letter is selected. If the player fixates on another letter in the meantime, the previous letter is scaled down again and not selected. At the bottom of the screen, the player can either confirm the inputted nickname or delete the last written letter by selecting the respective virtual buttons. All these interface and interaction components of Schau genau! were very well received by the participants, yielding a high usage and positive feedback.

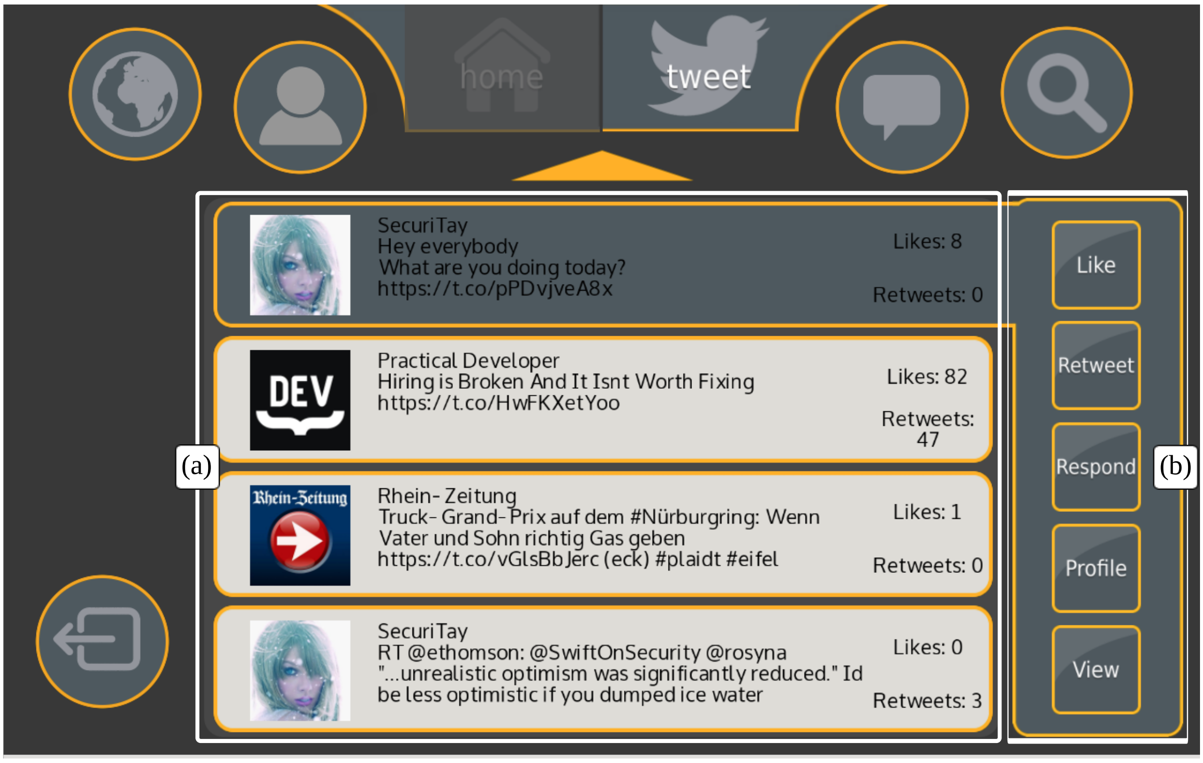
The game has been presented in an arcade box made of wood. The box includes a height-adjustable chair in front of a screen, which is placed behind glass. A Tobii EyeX eye-tracking device has been attached to the lower part of the screen's frame. For the sole purpose of starting and aborting the game, a single red buzzer has been placed on a tray between the chair and the monitor. Nearly 3,000 completed user sessions were recorded on a state horticulture show in Germany, which demonstrates the impact and acceptability of gaze-controlled interaction among lay users as implemented in our interface. The control style had a statistically significant, yet very small effect on high scores: games with avatar positioning like with mouse emulation led to the lowest, games with indirect avatar positioning that employ the knowledge about the flower location led to the highest scores. This is another hint about the importance to consider gaze signals not as a replacement for mouse pointing but to treat them specifically in each context.

6.2.4 Social Media with Eye Tracking

As of today, social media allows sharing multimedia content on a big scale with friends and strangers. One of the biggest social networks is Twitter,[3] a platform that allows users to share their opinion, to follow the posts of other users, and to post media on a personal wall. Users may respond to entries on another user's personal wall and forward entries from the same on their own. Social media platforms may act as an open window to the world for people with motor impairments — if they can use such services. Usually, Twitter is accessed via a Web interface or smartphone applications. We have developed a stand-alone Twitter application with a gaze-controlled interface [102].

The design and positioning of interaction elements are significant aspects of interfaces created for gaze control. Unintended activation of interaction confuses users and hurts the user experience. The most important part of Twitter is the personal wall. The personal wall contains posts from other users that the user follows. See Figure 6.8 for our implementation of a gaze-controlled personal wall. The posts are presented in the center of the screen, called *content area*. A user can select a post in the content area by just focusing on it, e. g., while reading the contents. When a post at the bottom or top is selected, the system scrolls the post toward the center of the view. This behavior allows intuitively to browse through the available posts. The currently selected post is colored in dark gray, which informs the user about the selection and connects the post to the available actions in the *action bar*. The action bar is placed on the right-hand side for the content area and provides contextual actions for the selected post. This spatial separation of content and actions allows a user to scroll through the content without triggering unintended actions. Each selected post can be forwarded, responded to, liked, and unliked, and one can visit the profile of the user who created the post. Actions that require textual input automatically present a virtual keyboard for eye typing, which works with virtual buttons.

[3] `https://twitter.com`



SecuriTay
Hey everybody
What are you doing today?
https://t.co/pPDvjveA8x
Likes: 8
Retweets: 0

Practical Developer
Hiring is Broken And It Isnt Worth Fixing
https://t.co/HwFKXetYoo
Likes: 82
Retweets: 47

Rhein-Zeitung
Truck-Grand-Prix auf dem #Nürburgring: Wenn Vater und Sohn richtig Gas geben
https://t.co/vGlsBbJerc (eck) #plaidt #eifel
Likes: 1
Retweets: 0

SecuriTay
RT @ethomson: @SwiftOnSecurity @rosyna "...unrealistic optimism was significantly reduced." Id be less optimistic if you dumped ice water
Likes: 0
Retweets: 3

Figure 6.8: Screenshot of the personal wall in the gaze-controlled interface of our Twitter application [102]. (a) marks the content area that displays recent posts. A post can be selected by dwelling on it, e. g., when a user reads it. (b) marks the action bar, which offers actions that can be performed on the selected post.

In addition to the personal wall from Figure 6.8, five other views can be accessed with the global navigation on the top of the interface. Most views of the interface share the concept of content area and action bar:

- Send a post
- Discover trends
- Visit own user profile
- Private messaging for direct user communication
- Search and view other profiles

To develop a gaze-controlled interface for an existing application, one needs low-level access to its functionalities. Twitter offers a public representational state transfer (REST) API[4] for developers to access a

[4] https://dev.twitter.com/rest/public

user's information and to submit posts and messages. For this, a user has to activate communication over this REST API,[5] which is performed in our application at the first successful login.[6] The API follows the OAuth[7] protocol. There are various libraries[8] available to access the REST API conveniently in different programming languages. Since our application is written in C++ language, we decided to delegate the twitcurl[9] library for all communication with the social network. The application is open-source and available on GitHub.[10]

We have performed a comparative evaluation with 13 participants at the university. The experiment was designed to compare objective and subjective user experience between using the mobile Web page with an emulation approach and our gaze-controlled interface. The participants were asked to perform specific tasks representing the common social media usage: to write a post and to publish it, to find a particular user and follow another user, to find and like a certain post about a specific topic, and to explore the application as one would do for social media browsing (5 – 10 minutes). We report that our gaze-controlled interface is regarded as more intuitive and easier interpretable by the participants than the emulation approach. Despite its novelty, the novel interface performs well on usability measures and required less mental demand from the participants.

The results of the evaluation imply that interfaces designed for gaze control require tight integration of gaze signals for a better user experience. However, it is not a scalable approach to design and develop an extra gaze-controlled interface for each application scenario from scratch. Most interfaces designed for gaze control share similar mechanisms, like virtual buttons, and virtual keyboards. Therefore, we argue that there is a need for a framework to design such interfaces.

6.3 Framework for Gaze-controlled Interfaces

We have developed a framework called "eyeGUI" [135][11] that makes it easy to compose gaze-controlled interfaces with elements that are suitable for gaze input.

Gaze-controlled Interfaces with eyeGUI To cater to the limitations of accuracy in eye-tracking and selection issues like the Midas touch problem [89], applications that use eye tracking as an input method primarily depend on the presentation, manipulation, visual cues, and feedback of elements in their interface. The eyeGUI framework provides a variety of interface elements, like virtual buttons, virtual keyboards, images, text displays, and text editors, that work effectively with gaze input and

[5] `https://apps.twitter.com`

[6] `https://dev.twitter.com/oauth/application-only`

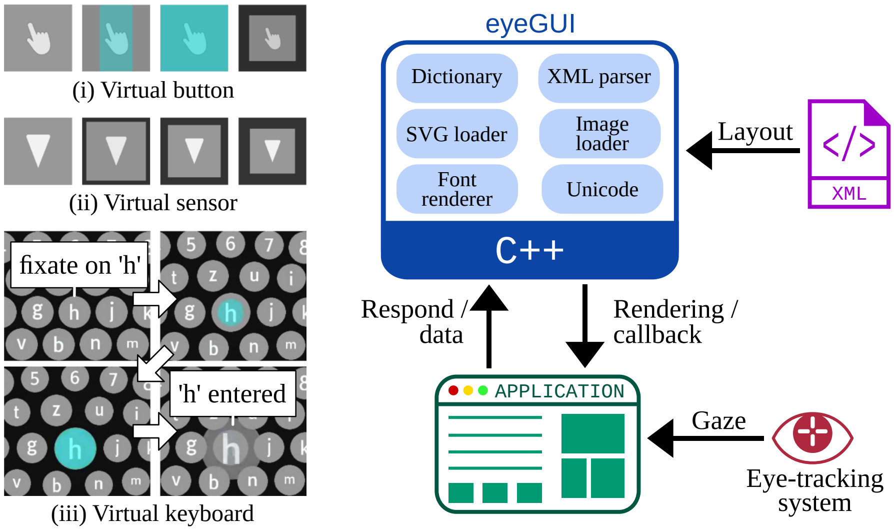
[7] `https://dev.twitter.com/oauth`

[8] `https://dev.twitter.com/overview/api/twitter-libraries`

[9] `https://github.com/swatkat/twitcurl`

[10] `https://github.com/MAMEM/GazeTheWeb/tree/master/Tweet`

[11] `https://github.com/raphaelmenges/eyeGUI`



(a) Interface elements of eyeGUI

(b) The software architecture of eyeGUI

Figure 6.9: The eyeGUI [135] framework for designing gaze-controlled interfaces.

allow composing interfaces for many kinds of applications. The interface elements can be grouped into layouts and further ordered with grids, stacks, and scrollable overflows. The elements can be customized in their size, appearance, or behavior, e. g., virtual buttons can be given an arbitrary symbol that scales automatically when the size of the interface changes, yet the ratio of height and width of the symbol stays the same.

All interface elements in eyeGUI are designed especially for gaze control in their size, appearance, and interaction, e. g., virtual buttons get activated when the gaze hits them for a set dwell time, and they shrink after triggering to provide a user with feedback about the activation [104]. A colored overlay increasing in size works as a visual representation of the remaining dwell time until the activation. The screenshots in Figure 6.9 (a) show the interaction with three different interaction elements from the eyeGUI framework. On the top (i), the states of a virtual button are depicted. After the user fixates on the virtual button, a cyan overlay fills the virtual button. When the dwell time is over, the cyan overlay completely covers the virtual button, vanishes, and the virtual button shows a press animation. In the second row (ii), the interaction with a virtual sensor is shown. A virtual sensor is an interaction element that instantly reacts to gaze and spawns signals to the application. The longer the gaze lays upon the element, the stronger the signals become. This instant interaction is useful, e. g., for smooth scrolling of contents. The four images at the bottom (iii) show different stages of a virtual keyboard with magnifying effect in the dwell-time-based character selection. Besides the interface elements, the framework also offers other eye-tracking-specific features, like displaying the gaze path during operation.

Architecture of eyeGUI The framework has been developed in C++ 11 and uses cross-platform compatible OpenGL for rendering. Each layout and its contained elements are defined in XML files. Furthermore, colors, sounds, dwell times, and animations can be adjusted in a style sheet. Each interface element can be assigned classes from a style sheet and classes in a style sheet can inherit properties from each other in a hierarchical manner. All events that are spawned from interaction elements are sent to registered listeners, which can handle incoming events on their behalf. The listeners can be created in the custom application environment to interact with external APIs. Eye-tracking devices would send gaze signals to the application, which implements a receiver for the gaze signals and filters them toward gaze data. The gaze data is then passed to the eyeGUI framework, which handles interaction and calls events in the interface (e. g., when a virtual button is selected), and the application would react to those interactions at calls to the registered listeners. For development purposes, the control paradigm can be switched to mouse input to emulate gaze signals. Figure 6.9 (b) depicts the architecture of the eyeGUI framework. The integration of eyeGUI is similar to popular OpenGL interface libraries like ImGui[12] and AntTweakBar.[13] A developer is free to choose how to create a window and in which way to initialize the OpenGL context. Before the render loop is entered, the GUI object for eyeGUI must be instantiated and an arbitrary number of layouts from XML files can be added. During the render loop, for every frame, the most recent gaze sample is used to update the interface, which then provides feedback on whether the input has been used by any layout. Based on that feedback, a developer can decide how to update any custom application content. This enables developers to overlay their rendering with eyeGUI and use the gaze signal not only to allow interaction with eyeGUI, but also with their custom creations. All functions are accessible through a single header file with C++ functions, and the memory allocation for displayed images and other media content is handled automatically.

The eyeGUI framework has been used by us in two gaze-controlled applications. First, we have used eyeGUI in the gaze-controlled client to access Twitter from the previous section in this chapter. Second, we have used eyeGUI in a gaze-controlled Web browser that is presented in the next chapter.

6.4 Web Browsing with Eye Tracking

One of the most common and most useful applications that today's computer users access is the World Wide Web. Access to the Web is limited for people with various kinds of disabilities, which might be caused by disease, accident, or due to aging [76]. Although standards are a noteworthy component of the overall effort towards universal accessibility, e. g., the W3C Web Accessibility Initiative,[14] the reality is that the majority of Web pages are not developed with accessibility in mind. The current state of accessibility support on the World Wide Web requires tools that can handle all Web pages, regardless of structure or adherence to accessibility standards.

[12] `https://github.com/ocornut/imgui`

[13] `http://anttweakbar.sourceforge.net`

[14] `http://www.w3.org/WAI`

Accessible Web Browsing

Various approaches and software were proposed to enhance Web accessibility. The concept of non-visual browsers was studied for the benefit of people with visual impairment. “CSurf” [128] was a non-visual browser, which extracted relevant information from a Web page and outputted it via an audio screen reader. Jensen and Øvad [90] proposed to embed a sign language dictionary in a browser to optimize Web accessibility for people with hearing impairment. “Firefixia” [157] was an accessibility Web browser toolbar that adapted the presentation of Web content to support people with dyslexia. The “Web Trek” [44] browser utilized multimedia, like an image-tile-based search engine, to provide Web access for people with cognitive disabilities.

One population of users for whom the Web is especially important are those with motor disabilities, because an accessible Web may enable them to do things that they might not otherwise be able to do: shopping, getting an education, or running a business. In that regard, voice-enabled Web browsers [3] or Web browser addons like “HandsFreeChrome”[15] might provide people with motor impairment an option to interact with the Web using voice input commands. However, voice input suffers from recognition and privacy issues. Moreover, most people with motor impairments, including cerebral palsy, amyotrophic lateral sclerosis (ALS), brain stem strokes, and certain spinal cord injuries, have also impaired speech, leaving fewer options for communication. In this regard, a single switch, albeit being a low bandwidth input device, is a widely used control device due to its simplicity and economy. There were several browser extensions and approaches [74, 131, 172] to support Web navigation through single switch input. These approaches either simulated the “tabbing” action on a keyboard and scanned through interaction elements at a time interval, or they were based on the extraction of all hyperlinks to create a separated list of the hyperlinks with selection through linear or incremental search. There were also Web browsers controlled by brain-computer interfaces for people who are completely paralyzed [19, 94]. However, the interaction was limited, extremely slow, and error-prone.

Gaze-controlled Web Browsing

Eye tracking has emerged as a non-intrusive way of interaction [89] and can provide an effective modality to enhance Web accessibility for people with motor impairment. For some users, gaze might be the most suitable channel for communication. For example, for people at the advanced stages of ALS, eye muscles are often among the last to deteriorate.[16] We identify two directions to integrate eye tracking and the Web environment. Either one can bring eye tracking into the Web environment, or vice versa.

Bringing Eye Tracking into the Web Environment One prominent approach is to make gaze available in the Web environment, enriching the interaction means and augment the experience of users.

[15] `https://www.handsfreechrome.com`

[16] `https://www.sciencedaily.com/releases/2017/01/170126093252.htm`, accessed on 31st May 2020.

The "Text 2.0" framework [21] by Biedert et al. offered a Web browser plug-in that allowed gaze-responsive Web pages, enhancing the traditional mouse-based interaction. The gaze-based interaction could be implemented by Web developers through a novel set of event handlers for Web page elements, e. g., `onFixation`, `onGazeOver`, and `onRead`. Wassermann et al. [192] built upon this concept and proposed a browser-independent framework to bring eye tracking into the Web environment. They described a native client to communicate with an eye-tracking device. The Web environment received generic gaze data from the native client via the Web socket standard and spawned corresponding events on elements of a Web page, utilizing the widespread jQuery library. The software allowed using various eye-tracking devices to capture gaze signals and the interpretation on a Web page within any Web browser that implemented the required Web standards. Wassermann et al. showed the practicality of their approach with the integration of gaze in an online eLearning platform. The inclusion of gaze signals allowed the eLearning platform to provide a user meaningful feedback on the relationships between different elements, to provide hints about information a user may have missed, and to integrate elaborated psychological studies. Although it is a pertinent guideline for Web developers to include gaze-based interactions in their application, the proposed mechanisms did not resolve the problem of browsing the current Web with gaze-controlled interactions. Hence, there is a need for introspection methodology to identify interaction elements automatically, so that the interaction can be adapted for gaze control.

Bringing the Web into the Eye-Tracking Environment To enhance Web accessibility with eye tracking, there have been a few approaches to allow gaze-based interaction for certain actions like hyperlink navigation and scrolling. Abe et al. [2] demonstrated a custom eye-tracking environment to control a limited number of Web browser functionalities. "WeyeB" [154] was a browser prototype that implemented hyperlink navigation via eye gestures and scrolling through virtual buttons. Lutteroth et al. [123] proposed a method for hyperlink navigation based on color-coding of hyperlinks and off-screen color panels, which could be looked at for selection of a respectively colored hyperlink.

All mentioned approaches did neither investigate the extraction of complex interaction elements in the Web environment like text inputs, select fields, or videos, nor their adaptation for effective gaze-based interaction. Moreover, their limited adaptation for gaze input worked exclusively for non-dynamic Web pages, as they only initially parsed a Web page for interaction elements. Hence, their approaches lack the design and functionality to work for the modern Web environment with changing contents on a Web page. Thus, we need to better understand the semantics of interfaces in the Web to design better interactions with eye tracking.

6.5 Interface Semantics

We define *interface semantics* as the position, size, and type of elements on a Web page in terms of *which kind of information they present* and *how a user can interact with them*.

Adaptation of Interfaces using Interface Semantics

Numerous approaches adapt interfaces to account for varying means of input by utilizing the knowledge about the interface semantics. Considering the ubiquity of mouse pointing in computer applications, researchers developed techniques to improve the pointing performance of users by considering the interface semantics. Many interfaces resized pointing targets to enhance the spatial accuracy of pointing. The “Bubble Cursor” [72, 65] employed interface semantics to dynamically resize the activation area and to let the cursor snap to the nearest target. Blanch et al. [22] decoupled visual space and motor space of an interface to improve pointing performance. They adapted the pointer speed to the underlying interface elements to make it easier to select plausible options, to make it more difficult to select less plausible options, and to scale the interface elements in visual space according to the information they convey.

Besides the improvement of pointing performance, there were approaches to adapt general interface characteristics for non-conventional input devices. Wang and Mankoff [190] provided support for arbitrary mappings between input methods. They described an abstract layer between input and applications and defined a mathematical model to map between different input methods according to their information throughput. The mappings were not inherently context-aware, as they did not integrate a retrieval of the underlying interface semantics. In contrast, the “XWeb project” [150] provided dynamic adaptation for a variety of applications and worked across several different input methods, including mouse and keyboard, pen, laser pointer, and voice. For this purpose, XWeb contained an interface specification language that allowed the adaptation of input methods. XWeb therefore only worked with interfaces written using the XWeb language. Analogously, Carter et al. [34] proposed a tool that automatically modified desktop applications whose interfaces were based on the Java Swing toolkit, to accommodate a variety of input methods, like pointers, keyboards, switches, or voice. Conceptually, they divided the user interaction into navigation actions – to select an interaction element among the available ones – and control actions – to select an option within the interaction element – and then adapted both independently. For example, in the case of keyboard-only input, they replaced a list with an interaction element that suggested available list items, based on the entered text. For switch input, they replaced a text field with an interaction element that allowed a user to select characters one at a time. They suggested a lookup table to augment or replace interaction elements according to the available input methods with interaction elements offering the more suitable interaction.

The discussed approaches expected that users can perceive the interface and inspect options with their eyes while using an input device (mouse, keyboard, touch, or switch) in parallel. The approaches assumed that users could scan the available elements and options and that users observed the effect of an input method while the interaction took place. However, this assumption is not valid for unimodal gaze-based interaction, because the eyes of a user are performing a double duty [125]. The eyes of a user are overloaded with both perception or inspection and selection of an option. Improving the user experience for gaze-based interaction requires consideration of specific characteristics and challenges of eye tracking. However, none of the above-mentioned approaches investigate how the knowledge of interface semantics could help overcome these challenges. In this thesis, we aim to reduce the interaction and visual overhead for gaze input and propose how the interface semantics of existing application interfaces could help to achieve this goal. The past approaches either took the knowledge about the interface semantics as granted or required an interface to be written in a specific language. However, we want to retrieve the interface semantics from the existing environment of the Web.

Methods for Interface Introspection

The interface semantics can be retrieved using various introspection methods. Introspection is a means to observe, monitor, and reflect. It plays a key role in many different fields such as psychology [98], sociology [33], and computer science [5]. In computer science, the term introspection relates to the tracking of objects and their properties such as type, location, and status within a known environment. In software engineering, introspection of programming components [124] allows for adapting modules to achieve more robust behavior, such as fault tolerance [174, 31]. In human-computer interaction, introspection of interface objects permits the adaptation of interaction elements to enhance the experience of users [177]. There are various approaches to introspect interface elements for a personalized and context-aware adaptation of interfaces, based on individual profiles and interests [4, 67]. In this thesis, we focus on how interface introspection can provide the interface semantics from Web pages and help to adapt interaction for the specific input method of eye tracking.

For desktop applications, there are several ways to obtain element-related information and to interact with interaction elements in interfaces from dedicated accessibility APIs [40, 177]. Apple has defined a universal access APIs for its Carbon and Cocoa toolkits.[17] Similarly, Microsoft provides the Active Accessibility[18] and System.Windows.Automation frameworks, and the X Window System offers an Assistive Technology Service Provider Interface (AT-SPI), which is supported by GTK+, Java/Swing, the Mozilla suite, StarOffice/ OpenOffice.org, Qt, and provides a toolkit-neutral way for accessibility services. Most of these APIs can query attributes like position, size, type, and state of interaction elements in an interface — and allow for interaction with the exposed elements. Furthermore, there have been generic approaches to retrieve interface semantics from the visual appearance of an interface by interpreting the pixel patterns on the screen [50, 51]. But the interpretation is error-prone and works only on a limited number of interface designs.

For the Web environment, retrieval, classification, and tracking of interaction elements is the basis that allows us to adapt the interaction with Web pages for eye tracking as an input method. Approaches of parsing the initial HTML structure of a Web page are not enough, as the adaptation needs to be revised as the Web page interface may change dynamically its contents, layout, and visibility of elements. Thus, it

[17] https://developer.apple.com/documentation/applicationservices/carbon_accessibility
[18] https://www.microsoft.com/en-us/accessibility

is necessary to let the Web browser extract the interaction-relevant elements of a Web page and observe changes in their properties. However, the introspection of elements in combination with the dynamics of Web pages was not yet reflected in Web page data extraction research, which is mostly focused on pure content extraction [35, 193]. Even recent works only described methods that allow for observation of preselected, dynamic elements [27, 114].

We suggest using the knowledge about the interface semantics, i. e., the interaction elements, to integrate interaction modes that are suitable for gaze-controlled interaction with the respective interaction element. Before we discuss and apply our approach of an efficient introspection method, we need to understand the shortcomings of the existing approach of emulating mouse and keyboard using gaze signals.

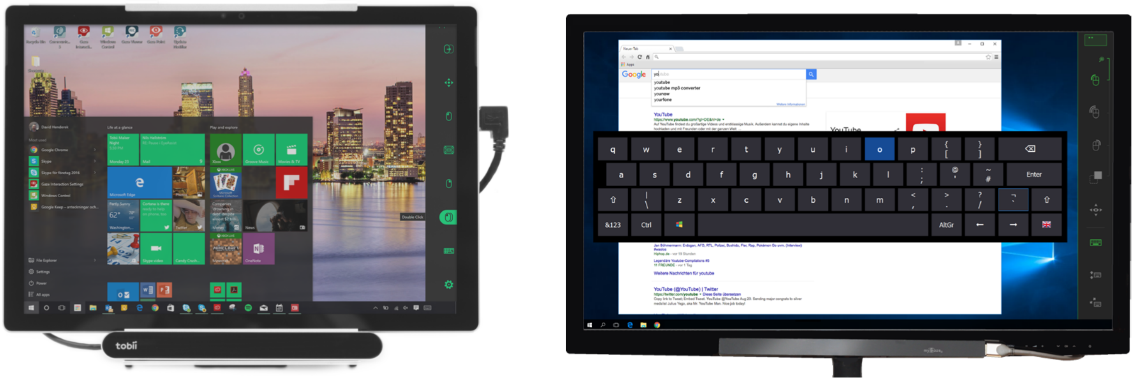
6.6 Emulation Approach

Eye tracking as an input method can break interaction barriers and improve the quality of life for people with a disability that is limiting their interaction through traditional input devices in the digital environment. Remote eye-tracking systems establish a non-intrusive communication channel between a user and a computer. Thus, digital accessibility for users with reduced mobility can be improved, and gaze control may also assist users in daily life situations where they cannot use their hands [130]. However, most interfaces are not designed to be controlled with eye-tracking devices, which provide gaze data with limited accuracy. Furthermore, gaze control requires unconventional selection techniques [108] to distinguish between perception and inspection behavior or a selection intention of a user.

Methods for better pointing and typing with gaze are an imperative aspect of research about interaction with eye tracking. Eye pointing and eye typing can be easily combined into emulation software that can emit mouse and keyboard events. Emulation software can provide people with motor impairment an opportunity to control a broad set of computer applications.

Concept of the Emulation Approach

A common approach to employ gaze as a communication channel for controlling application interfaces is to emulate mouse and keyboard devices through an additional *command-translation layer*. The command-translation layer is implemented as a graphical emulation panel, which contains virtual buttons. The emulation approach senses fixations on those virtual buttons that exceed a certain dwell time to emit mouse and keyboard events. The virtual buttons are sized such that they can be conveniently selected with gaze. Pointing within the traditional interface of the controlled application is based on magnification to compensate for the accuracy limitations of eye tracking. This allows for an *indirect* control of application interfaces through gaze via the command-translation layer — in contrast to direct gaze control. The approach has existed for several years, e. g., the “Eye-gaze Response Interface Computer Aid” (ERICA) software [116] was incepted in 1983 and included gaze-control mechanisms, featuring mouse and keyboard emulation at the operating system level. More recent work on gaze-based computer access uses the same approach with integrating mouse and keyboard functions [200]. Today's commercial eye-tracking systems like Tobii Dynavox Windows Control[19] or Visual Interaction myGaze Power[20] incorporate similar mechanisms. See Figure 6.10 for examples of commercial eye-tracking systems as sold by commercial companies. Recently, Microsoft has even integrated gaze-controlled mouse and keyboard emulation into the Windows 10 operating system.[21] Moreover, there exist open-source alternatives, like OptiKey [179], which work effectively with affordable eye-tracking devices available on the market.



(a) Tobii Windows Control [56]

(b) Visual Interaction myGaze Power [70]

Figure 6.10: Eye-tracking systems for computer access as sold by Tobii AB and Visual Interaction GmbH. The remote eye-tracking device is mounted below the screen and provides gaze signals to emulation software on the computer. The software of both companies shares the concept of indirect control over the common desktop and application interfaces via a command-translation layer. A user has presented an overlaying emulation panel, which is placed in both systems on the right side of the screen. By selecting virtual buttons in these emulation panels through a fixation, a user can choose how their gaze is translated to mouse or keyboard events.

Function of the Emulation Approach

We explain the function of the emulation approach by using the example of controlling a Web browser application, as visualized in Figure 6.11: A remote eye-tracking device captures the gaze of a user (1) and transfers the observations as gaze signals to the emulation software. Real-time filtering is applied to the gaze signals to compensate for issues with precision (2). An emulation panel is rendered to an extra window on the screen (3). The emulation panel is presented to the user (4) next to the window

[19] https://www.tobiidynavox.com/software/windows-software/windows-control

[20] http://www.mygaze.com/products/assistive-products/mygaze-power

[21] https://support.microsoft.com/en-us/help/4043921/windows-10-get-started-eye-control

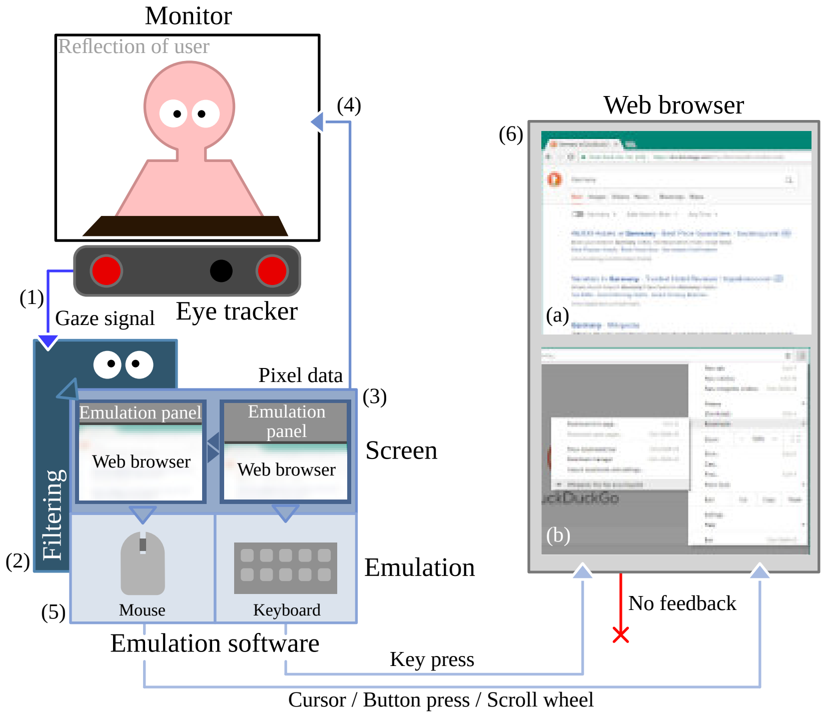


Figure 6.11: Emulation of mouse and keyboard to browse the Web with a desktop Web browser through indirect gaze control.

of the Web browser application interface. The virtual buttons in the emulation panel are grouped by the input device they can emulate, for instance, OptiKey [179] offers one emulation panel for mouse emulation and another emulation panel for keyboard emulation. Selections of virtual buttons in the emulation panels are translated to mouse events in the mouse emulation panel or keyboard events in the keyboard emulation panel (5). Because the screen space is limited, it is not feasible to put all available emulation functionalities associated with individual virtual buttons, which are sized to account for the limited accuracy of eye-tracking systems, into a single emulation panel or to display all emulation panels at once. Switching between the emulation panels is achieved through a separate virtual button. The Web browser receives the emitted events as if they were originating from a physical device and the application acts like controlled by the conventional input method (6) for both Web page (a) and Web browser (b) access through the original application interface.

Problems of the Emulation Approach

Figure 6.12 shows a Web search task accomplished with the emulation approach that requires several clicking, scrolling, and typing efforts from a user. We identify two central problems of input action overhead and inspection overhead with the user experience of the emulation approach.

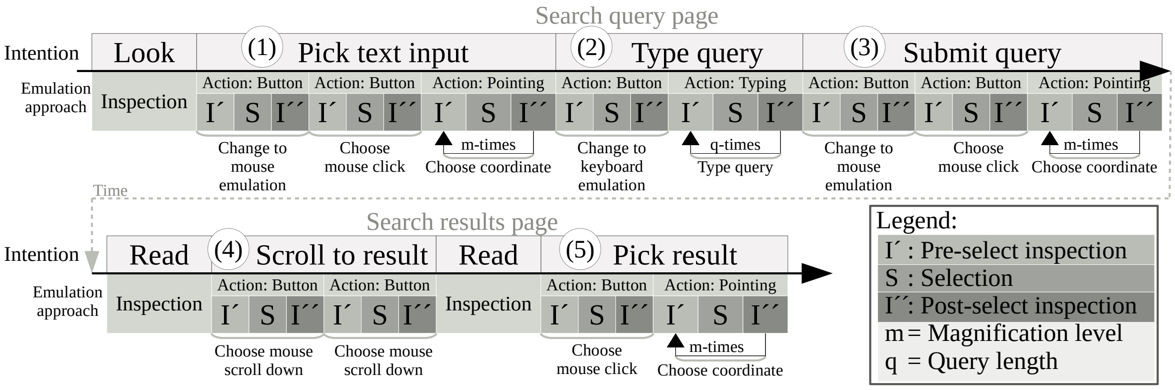


Figure 6.12: Input actions to place a query on a search engine Web site and to select a result with the emulation approach. We assume a gaze-based pointing method with multiple magnification steps and a dwell-time-based virtual keyboard. A user opens a search engine Web page, picks the text input to set the focus on (1), types the search query (2), and clicks on the submit button of the search engine (3). On the Web page with the search results a user scrolls down two times (4) and finally picks a result (5).

Input Action Overhead The emulation approach does not serve automatically an appropriate interaction mode. A user needs to cognitively map each interaction element in an application interface onto the most appropriate emulation panel. For example, a user first needs to click on the text input to type into (1), and then switch to the panel for emulation of the keyboard (2). This is a cognitively demanding task, especially for a person with motor impairment, who may have never used a mouse or a keyboard. In general, requiring a user to decide between two modes that emulate two different devices leads to high memory load [84] and negatively impacts the user experience. Furthermore, switching between the emulation panels introduces additional input actions, extending inspection and selection efforts.

Inspection Overhead Both the application interface and emulation panel are placed on separate screen areas and there is no feedback from the application to the emulation. For an input action, a user must shift her attention toward the window with the emulation panel to inspect which input event to perform. Once a user has selected a virtual button in the emulation panel to cause an event, she must switch the visual attention back onto the application interface to inspect the effect on the application, and back again to the emulation panel when further input actions are required. For example, in a Web browser application controlled through the emulation approach, each time a user initiates a scroll down or scroll up command, she needs to check the effect on the Web page within the Web browser viewport (4). The additional perceptual and cognitive load caused by shifting the focus and repeated scanning of the environment [129, 87] detriments the user experience.

Users do not use pointing and typing in isolation and the acceptance of technology primarily depends on how these atomic interactions let users interact with applications through their interfaces as a whole. The emulation approach acts as a command-translation layer between the eye-tracking environment and interfaces that expect events from traditional input devices. A user must first imagine which event from the physical input device the interface expects and then employ the command-translation layer for emulating the corresponding input device accordingly. Moreover, the usability of emulation approaches has only been assessed based on the accomplishment of atomic interactions or the subjective assessment of the eye-tracking input method against other input means.

6.7 Conclusion

Eye tracking has been explored as an input method for interaction in various application domains. However, most interfaces that have been designed for gaze control are research prototypes with limited functionality. Those prototypes are not useful for end-users, especially for the target group of people with motor impairment. They use eye tracking primarily with an emulation approach, to replicate the functionality of mouse and keyboard devices. But the emulation approach induces issues in user experience because of input action overhead and inspection overhead. It is important to explore how gaze-based interaction can instead follow the principles of user-centered design, providing a better user experience for end-users.

CHAPTER 7

Adaptation of a Web Page Interface for Gaze-based Interaction

In this chapter, we adapt the interface of Web pages and design a Web browser for better gaze-based interaction. First, we discuss how the knowledge about the interface semantics can be utilized to improve the user experience in Section 7.1. Second, we suggest to retrieve, classify, and track interface semantics, i. e., interaction elements, in an efficient way for dynamic Web pages through introspection in Section 7.2. Third, we demonstrate how we adapt the interface of Web pages and design a Web browser for eye tracking as the input method in Section 7.3. For this, we augment interaction elements on Web pages with *gaze-sensitive icons*. These gaze-sensitive icons can provide interaction modes upon the selection that are convenient to operate with gaze.

7.1 Propositions for Improved User Experience

In every application scenario, user interaction with an interface involves potentially multiple input actions to complete a task or attain the information needed. In gaze-based interaction, eyes perform input actions that inherently require multiple sequential inspections and selection efforts. The emulation approach overloads this effort on top because additional input actions and inspections are required from users to map the emulation context with the interface semantics. This invariably affects the primary measures of user experience, i. e., task completion time, usability, and workload. In this thesis, we argue for seamless integration of gaze input in the application ecosystem, i. e., a deeper integration of gaze signals and interface elements to reduce the number of input actions and the inspection overhead.

Reduction of the Input Action Overhead

The common design principle of *"minimal input actions"* [169] states that a function should be reachable with as few input actions as possible. However, the emulation approach imposes an overhead of input actions, for example by demanding a user to switch between emulation panels according to the required events. The switching does not only contribute to higher task completion time but also makes the interaction more tedious for a user, as a selection is usually performed via dwell time or eye gestures.

We argue that the input action overhead could be reduced by lessening the interpretation gap between the user intention and the interaction elements. In this regard, we propose to utilize knowledge about interface semantics to associate the functionality of an interaction element with suitable gaze-based interaction. For example, interaction with a hyperlink on a Web page could be associated with a left mouse click event, whereas the interaction with a text input would be associated with a virtual keyboard for eye typing. More sophisticated optimizations could be achieved, e. g., if text input is designed for password entry, a gaze-based text entry interface for secure password entry might be instantiated [105].

Reduction of the Inspection Overhead

For the required input actions, an inherent part of interaction by a user is to visually inspect the interface. A user needs to assess which interaction element to select, before a selection, and to inspect the effect of the selection on the application, afterward.

Reduction of the Pre-select Inspection The design guideline *"recognition than recall"* [145] argues that a user should not be expected to remember the functionality of an interface, but to recognize the functionality through an intuitive design. This issue is even more relevant in gaze-based interaction due to the dual roles of the eyes for inspection and selection. In dwell-time-based selection, a slow inspection might cause an unintentional selection and discomfort to a user. Thus, it is imperative to minimize the time needed for visual search within a gaze-controlled interface. Understanding the semantics of interaction elements, we may reduce the time needed for visual search during the inspection before a selection. For example, we may augment interaction elements with visual indicators, to provide a user with a clue about the kind of interaction that is to be expected with an element, and to allow in-place interaction with the element.

Reduction of the Post-select Inspection The design principle *"visibility of system status"* [169] advises making the status of a computer application easy to perceive by a user. But the emulation approach divides the screen space between the emulation panel and the application interface. After issuing an event in the emulation panel through a selection, a user must shift her attention to the application interface and inspect the effect of the event there. The knowledge about the interface semantics however allows for providing in situ feedback about selections and may thus help to curb the number of attention shifts while reducing cognitive load in controlling an interface with gaze input. For example, we can retrieve the scroll offset of a document and provide feedback about the scrolling in a relevant position on the screen, where a user may proceed or finish with scrolling.

For the realization of the propositions in the Web environment, we require the interface semantics of Web pages and access to the Web browser functionalities.

7.2 Retrieval of Interface Semantics in the Web

Introspection of a potential dynamic Web page needs to work efficiently at run-time on the entire DOM tree, to track changes in appearance and functionality of a Web page interface. The DOM standard features the Mutation Observer [71], which is a mechanism to track changes to the DOM tree and DOM node properties. The Mutation Observer is available in the Web engines of current browsers[1] and allows for efficient tracking of changes in dynamic Web pages — independently of the operating system and the Web engine. We inject a JavaScript snippet including a Mutation Observer instance into the `document.body` node of the DOM tree every time a Web page is loaded. The Mutation Observer receives records about the creation, update, or removal of DOM nodes and properties within the DOM tree. An example from our implementation is depicted in Figure 7.1, where a Mutation Observer retrieves, classifies, and tracks a newly attached text input and stores information like visibility, size, position, id, and textual content in a list dedicated to capturing such information about all interaction elements.

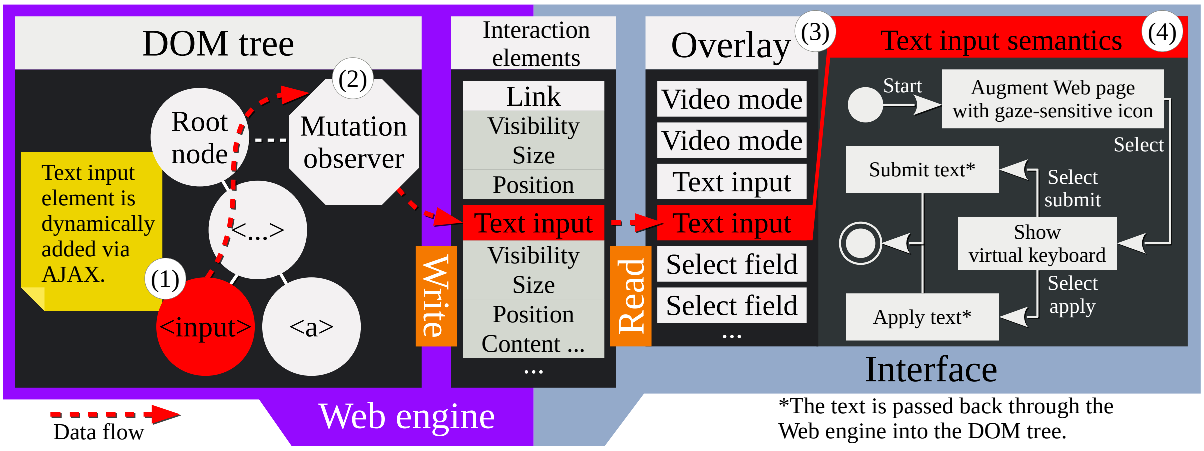


Figure 7.1: Data flow for interaction element classification and tracking. In the example, a text input is added dynamically to the DOM tree of the Web page via an AJAX execution. The node with tag <input> is appended to the DOM tree (1) below an existing structure. The Mutation Observer on the root node (2) is notified about the change and our approach classifies the node as interaction element of type text input (see Figure 7.2 for details about the classification process). The interface-defining code is called to perform a lookup of the found interaction elements and augments the Web page interface with an associated gaze-sensitive icon (3). When the user selects the gaze-sensitive icon associated with the text input, a virtual keyboard for eye typing to enter and submit the text is shown (4).

We perform a rule-based classification process, based on the tag, the type, and the role property assigned to a DOM node by the attributes of the HTML element. Figure 7.2 summarizes the classification process we have implemented. For example, an element `<input type="search"/>` from the HTML code is

[1] Check compatibility of different Web browsers with the Mutation Observer at the bottom: `https://developer.mozilla.org/en-US/docs/Web/API/MutationObserver`

mapped onto a DOM node that expects textual input from the user; thus, the element is classified as text input interaction element. In contrast, an element defined as `<input type="submit"/>` is treated as hyperlink interaction element. The classification of DOM nodes in interaction elements and their adaptation for gaze input is extensible for further combinations of properties and DOM nodes, like audio tags, radio buttons, or text fields for password entries.

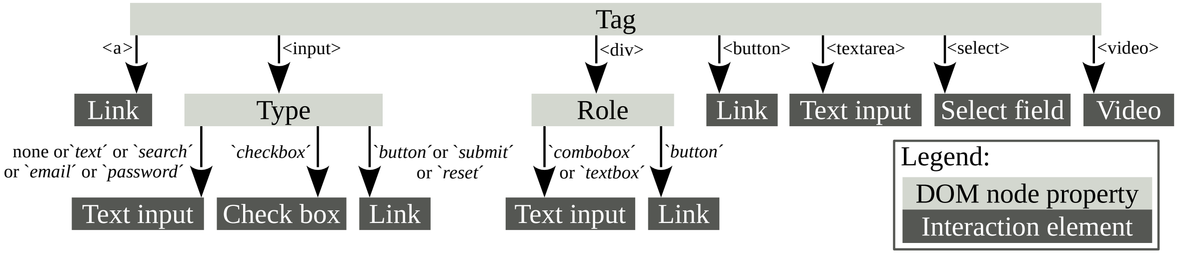


Figure 7.2: Classification of DOM nodes to interaction elements. This classification can be easily extended in the future to support further kind of interaction elements on a Web page.

The current DOM standard comprising the Mutation Observer comes with the limitation that the Mutation Observer cannot track changes of computed styles. Computed styles are style properties of a DOM node that are not defined by the DOM node itself but adopted from a property defined by parent DOM nodes. For example, a container element, which is not classified to be an interaction element and, hence is not tracked, changes its visibility property. The change in visibility may also affect the observed descendant nodes, e. g., an interaction element that is text input. The current DOM standard does not foresee that the Mutation Observer is notified about such a change of visibility. One might attempt to observe all DOM nodes of the DOM tree and estimate the impact of changes on descendant DOM nodes. However, observing the complete DOM tree is slow on complex Web pages, and replicating the behavior of the styling mechanism is a hard task. Therefore, we perform a frequent poll of the properties of interaction elements during run time and regularly check for changes. Polling can be limited to interaction elements deemed suitable for adaptation to gaze-based interaction because the creation and removal of DOM nodes are fully observable by the Mutation Observer itself.

Besides the Web page content from the `<body>` element, we also extract meta-information from the `<head>` element of a Web page document and retrieve status information through the utilized Web engine. The acquired data, e. g., the title and URL, the favicon, the area extent of a Web page, and the current scroll offset, is imperative to adapt not only the interaction within the space of a Web page but with a Web browser interface in general. We retrieve the browser control functionalities from the Web engine to load a Web page from a URL, to scroll a Web page in the viewport, and to adjust the zoom level. We make the general Web browser functionalities available through the gaze-controlled interface. Thus, we use the page meta-information for a better user experience in operating those general Web browsing functionalities.

The proposed methodology for introspection can be implemented on Web engines that follow the DOM standard, however, the gaze adaptation itself requires custom interface components, e. g., for the augmentation by gaze-sensitive icons and the integration of the eye-tracking environment into the Web browser, like a virtual keyboard for eye typing.

7.3 Adaptation of an Web Page Interface for Gaze Input

In this thesis, we use introspection to retrieve the interface semantics and to adapt interaction for eye tracking as the input method, which has not been explored before. We call such an interface a *gaze-adapted* interface. The Web standard offers various types of interaction elements on a Web page and the means of interaction varies for these elements. First, we consider complex interaction elements that offer multiple options of interaction, e. g., text inputs, select fields, or videos. Second, we consider simple interaction elements that offer a single option for interaction and are more frequently found in Web pages compared to complex interaction elements, e. g., hyperlinks or checkboxes. Third, we consider the interaction with the Web browser to scroll a Web page within the viewport, to change the URL, to mark and to choose a bookmark, or to manage tabs.

Adaptation of Complex Interaction Elements

Figure 7.3 shows an example of complex interaction elements (marked as (a), (b), and (c)) on a Web page.[2] We augment these complex elements with gaze-sensitive icons, to adapt the interaction based on their attribute properties such as size, position, and type. For a gaze-sensitive icon, the positioning is determined by the position and size of its associated interaction element on the Web page; the appearance and interaction mode is determined by the type of interaction element as classified according to Figure 7.2. These gaze-sensitive icons enhance the user experience as follows. First, the size of gaze-sensitive icons allows for an accurate selection via eye tracking. Second, each gaze-sensitive icon features a symbol to provide the user a visual cue about the type of the associated interaction element. Both, the spatial relation of the gaze-sensitive icons to the associated Web page contents and the symbol itself, potentially reduce the pre-select inspection time. Third, a selection of a gaze-sensitive icon summons the most appropriate interaction mode for the specific type of interaction element, as depicted in Figure 7.4. This reduces the overhead in input actions in comparison to the emulation approach because individual type recognition and switching between different emulation panels are not required. We describe the adapted complex interaction elements and their associated gaze-controlled interface modes in the following:

(a) Text input is a Web page interaction element that allows a user to insert text for a search query,

[2] A video entry on the MSN Web portal has been used as running examples in this section. The Web page could be reached under the following URL: `https://www.msn.com/en-us/news/video/roaring-crowds-greet-new-royal-baby/vi-AAweZlk`, accessed on 27th April 2018.

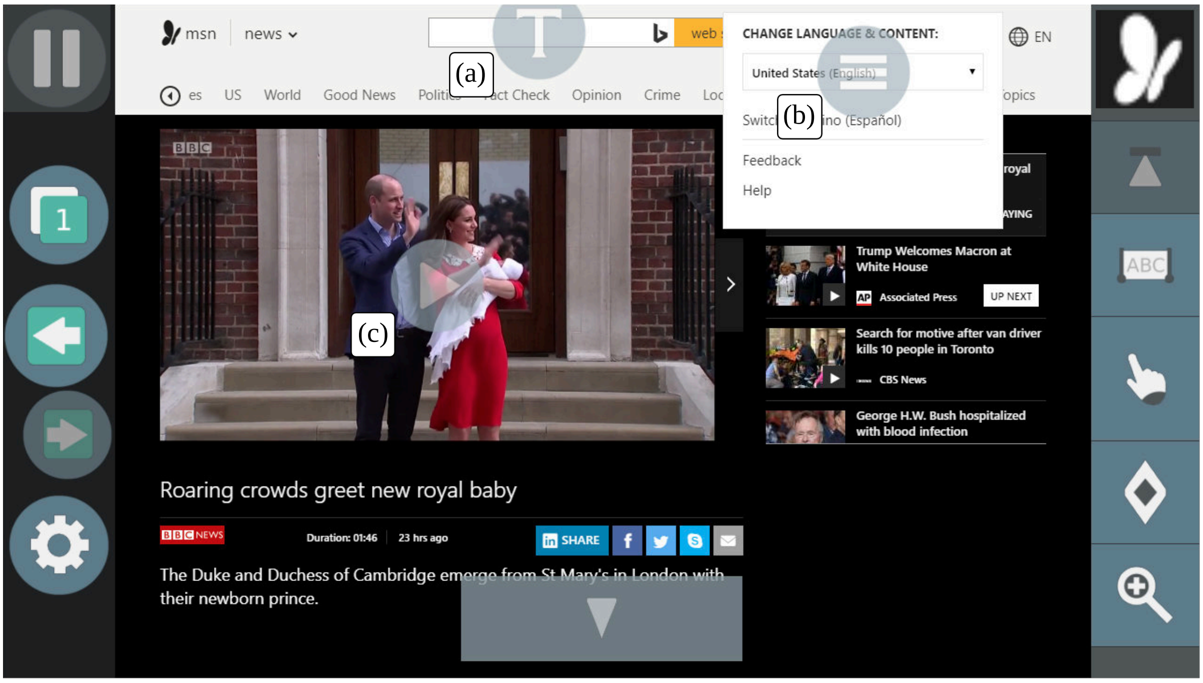
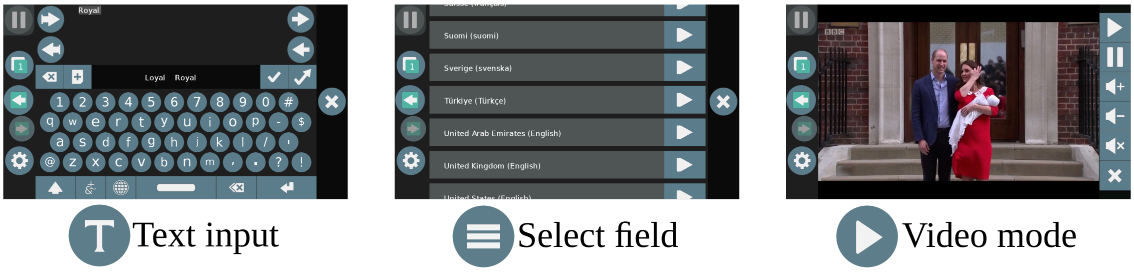

msn
news
web
CHANGE LANGUAGE & CONTENT:
United States (English)
Switc
ino (Español)
Feedback
Help
EN
es
US
World
Good News
Politi
act Check
Opinion
Crime
Loc
opics
(a)
(b)
(c)
royal
AYING
Trump Welcomes Macron at White House
Associated Press
UP NEXT
Search for motive after van driver kills 10 people in Toronto
CBS News
George H.W. Bush hospitalized with blood infection
Roaring crowds greet new royal baby
BBC NEWS
Duration: 01:46
23 hrs ago
SHARE
The Duke and Duchess of Cambridge emerge from St Mary's in London with their newborn prince.
1
ABC

Figure 7.3: Complex interaction elements on a Web page are augmented with gaze-sensitive icons.



(a) Virtual keyboard for text input

(b) List of options from a select field

(c) Mode with video controls

Figure 7.4: Gaze-sensitive icons on a Web page are associated with gaze-controlled interface modes for sophisticated gaze-based interaction.

information transactions like filling a form or sending a chat message. In conventional interaction, text insertion is performed via a physical keyboard. However, in gaze-based interaction, sophisticated eye-typing methods are required to translate gaze signals into keystrokes. Most of the eye-typing methods display each character of the alphabet as a virtual button. Therefore, each keystroke can be interpreted as an interaction option, which makes text input a complex interaction. We provide the user with an interface mode featuring a virtual keyboard for eye typing, including the option to directly submit a query without any additional selection on the Web page. Other semantics of a text input can be queried for further optimizations of the virtual keyboard, e. g., to automatically provide a discrete and secure way to insert passwords or to determine the expected language to support a user in the typing task.

(b) A select field is a Web page interaction element that allows a user to select among predefined options, like date of birth, filters of search results, or localization. The select field offers multiple options for a user to choose from, and, hence we categorize the select field as a complex interaction element. We present the available options in an interface mode with virtual buttons associated with each option, allowing for a convenient gaze-based interaction. If the number of options extends beyond the vertical screen space, the currently inspected option is vertically moved to the center of the screen and the interface mode automatically reveals further options in the direction of movement through scrolling.

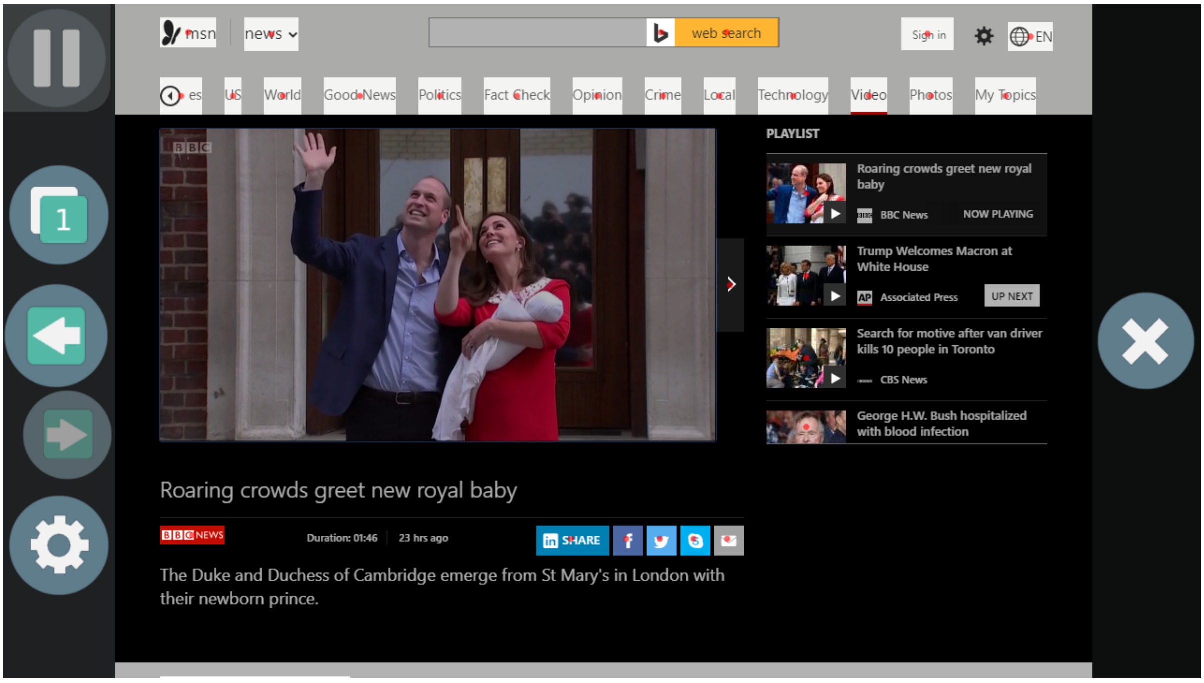
(c) A video is a Web page interaction element that embeds video content into a Web page. Videos offer multiple options for control, e. g., play, pause, or skip, which makes the video element a complex interaction element in our definition. Videos are especially cumbersome to interact with a gaze-based emulation approach, as the controls of a video on a Web page usually disappear when no mouse movement is detected over the video. However, mouse movement in the emulation approach is only inherently performed through positioning the mouse cursor to emulate a mouse click event. Instead, we suggest a gaze-controlled interface mode that shows the video instead of the Web page and features associated virtual buttons to control the video.

The interaction with the above-listed complex interaction elements has been gaze-adapted for the scope of our studies, however, adaptation is not limited to text inputs, select fields, and videos. Similarly, interface semantics about less common but standardized interaction elements like sliders or audio might be defined and their interaction adapted with dedicated gaze-controlled interface modes.

Adaptation of Simple Interaction Elements

In the Web browsing scenario, users face a high number of interaction elements with a single means of interaction, like hyperlinks or checkboxes. Contrary to the complex interaction elements, which might have to be handled in various ways depending on their exact type, these elements are generalizable to the same type of interaction, i. e., a click to navigate to a hyperlink or a click to toggle a check box. Hence, a dedicated gaze-sensitive icon is not required for each of these simple interaction elements. Furthermore, gaze-sensitive icons for gaze control need to occupy a minimum amount of screen space to account for accuracy limitations in eye tracking. A gaze-sensitive icon for each hyperlink or check box would lead to overlapping gaze-sensitive icons and therefore to ambiguous interpretation of gaze signals. We inherently associate all simple interaction elements with a click mode for gaze-based pointing. A user can activate the click mode by selecting the virtual button with the “finger-pointing” symbol in the right panel, see Figure 7.3. The click mode is implemented as a multi-step magnification of the Web page, to allow for an accurate selection among all candidate simple interaction elements [16].

Knowing the position of these elements provides the possibility of dynamic adjustments to deal with the limited accuracy of gaze estimation, i. e., a hyperlink can be selected in the near vicinity of a user's gaze coordinates.



Roaring crowds greet new royal baby

Trump Welcomes Macron at White House

Search for motive after van driver kills 10 people in Toronto

George H.W. Bush hospitalized with blood infection

Roaring crowds greet new royal baby

The Duke and Duchess of Cambridge emerge from St Mary's in London with their newborn prince.

Figure 7.5: Click mode for interaction with the simple interaction elements. The simple interaction elements on a Web page are highlighted and a red focus point is displayed over their center.

Moreover, minimalistic user interface design as of today, e. g., Google Material design,[3] often does not visually hint at simple interaction events like hyperlinks or Web page buttons through visual indicators like embossing or shadows. Users who use a mouse as a pointing device may observe the change from a "mouse pointer" to a "finger pointer" when hovering with the cursor over simple interaction elements at inspection. Our approach removes this prerequisite of hovering by highlighting the simple interaction elements in the click mode, see Figure 7.5. Furthermore, we overlay each simple interaction element with a *focus point* [106], rendered as a red dot in the center of a simple interaction element as displayed. The focus points support a user in the gaze-based selection process, as otherwise users tend to roam their gaze within a highlighted interaction element. Eye-tracking system then might be not able to detect a stable fixation, which is required for the selection. Both, highlighting and focus points, reduce the pre-select inspection overhead in comparison to the emulation approach. In addition, we provide local visual feedback about the issued selection by displaying a circular wave that collapses at the location of the selected simple interaction element. The visual feedback allows for fast post-select inspection of the selection without requiring a shift in attention.

[3] `https://material.io/design`

We argue that the described adaptation of complex and simple elements can enhance the user experience compared to an emulation approach. To showcase this, we revisit the sequence of input actions for a Web search scenario in the emulation approach (Figure 6.12 on page 112) with a corresponding improved sequence of input actions through the proposed adaptations in Figure 7.6.

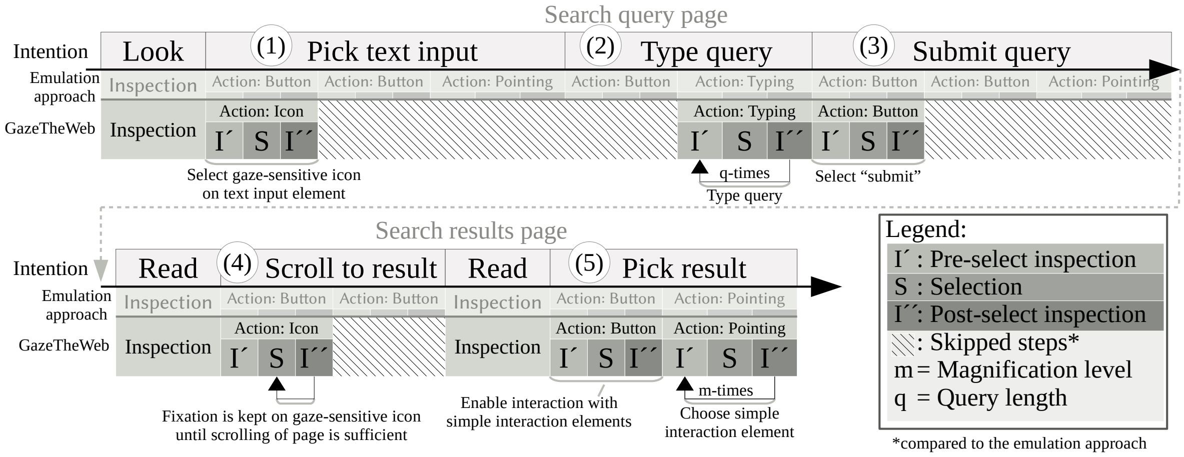
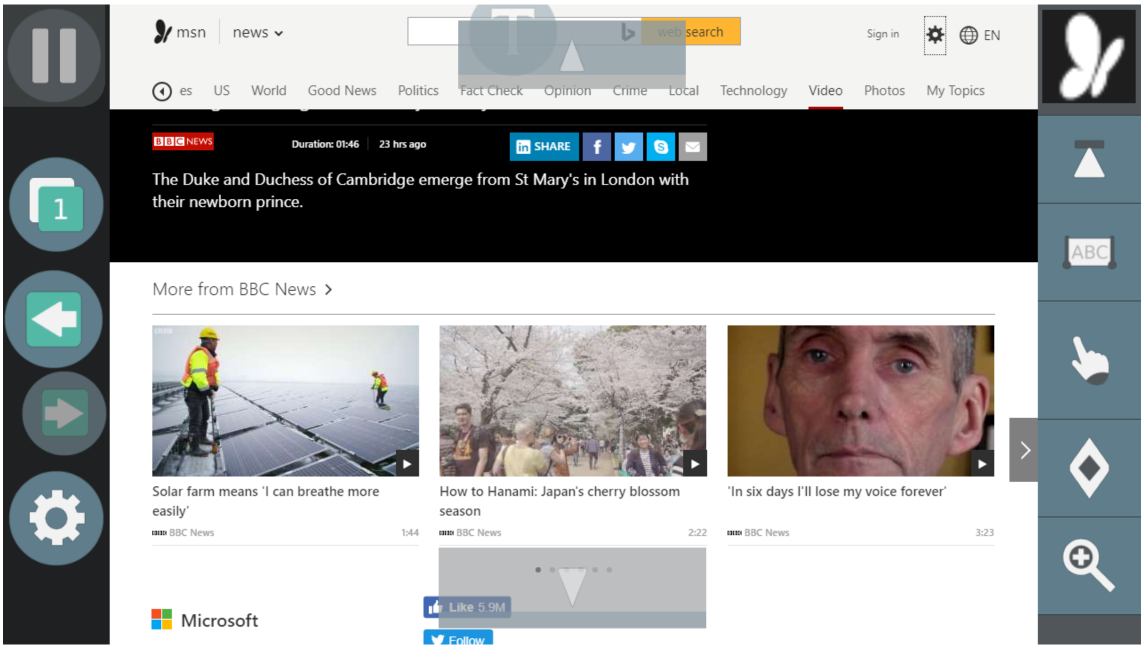


Figure 7.6: Input actions to place a query on a search engine Web site and selection of a result in the gaze-adapted Web page interface, analogously to Figure 6.12 on page 112. Instead of input actions to switch the emulation panels, a user can just select the gaze-sensitive icon associated with the text input (1). The virtual keyboard for eye typing is automatically presented after the selection of the gaze-sensitive icon and the user can type a query (2). The interface mode of the keyboard allows for the instant submission of the query, which saves another selection on the Web page (3). Scrolling down on the Web page is sped up, too, as the user can stay with her focus on the virtual sensor for scrolling (4). Additionally, a user's selection is facilitated by the highlight of the search results during the final pointing effort (5). This is not reflected in the figure, as the figure does only accounts for the reduction of input actions. In addition, reduction of inspection efforts can be expected at interaction, e. g., the gaze-sensitive icon associated with the text input is placed in situ on the Web page interface.

Design of the Web Browser Interface

We do not only consider the Web page proper, but we have also improve the user experience with a Web browser using gaze input in general. Kellar et al. [95] have examined user behavior in Web browsing and classify Web browsing functionality into two categories: *“within-task-session”* and *“new-task-session.”* They have defined within-task-session as the Web browser functionality used while a user works on a task, e. g., a Web search or shopping. In contrast, new-task-session functionality is primarily used by a user at the beginning of a task. This categorization has been incorporated into our main interface modes:

(a) The primary interface mode, as shown in Figure 7.7, contains the within-task-session functionalities.



The Duke and Duchess of Cambridge emerge from St Mary's in London with their newborn prince.

Solar farm means 'I can breathe more easily'

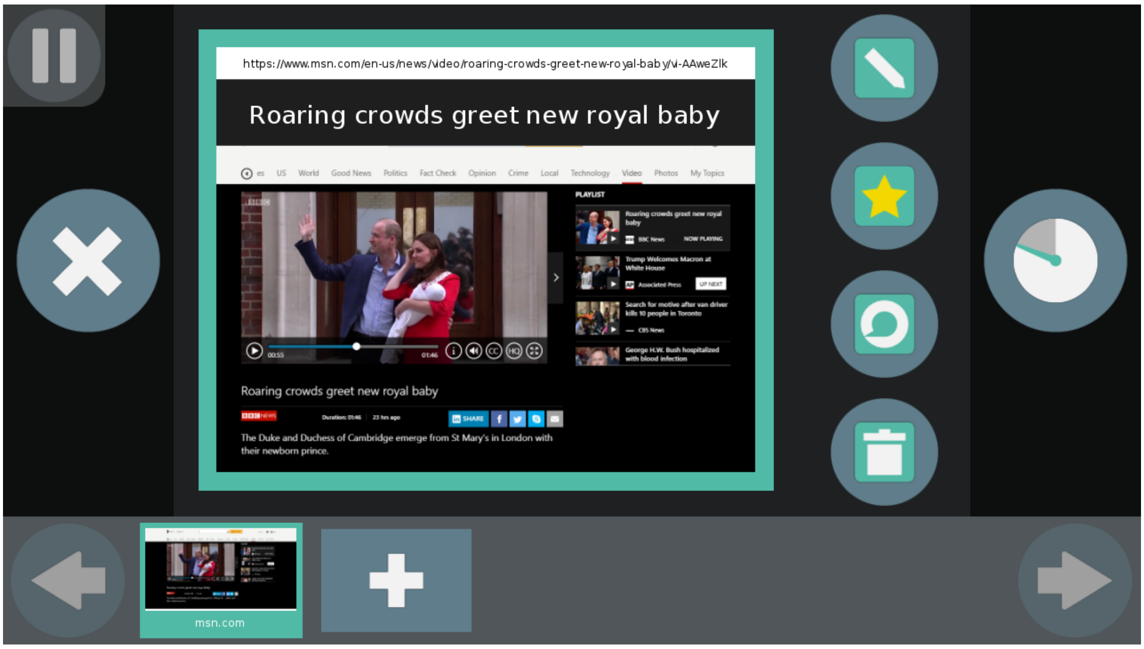
How to Hanami: Japan's cherry blossom season

'In six days I'll lose my voice forever'

Figure 7.7: Primary interface mode of GazeTheWeb with a viewport and within-task-session functionalities. The second virtual button from top on the left panel allows loading of secondary interface mode.

The interface presents the Web page with the augmented complex interaction elements inside a viewport in the center of the screen. There are two panels on both sides of the viewport to support interactions within a browsing session through the selection of virtual buttons, e. g., to activate click mode to select simple interaction elements, to go back to the previous page and forward to the next page, or to scroll directly to the top of the current page. Additional virtual sensors from the eyeGUI framework (see Section 6.3) are overlaid on the top and the bottom of the Web page viewport to support scrolling of the Web page within the viewport. Virtual sensors do not require a dwell time but react instantly to a fixation, which allows for smooth scrolling of the Web page in the viewport. The virtual sensors are only displayed if scrolling into the according to direction is available, e. g., when the Web page is scrolled to the top, the virtual sensor for scrolling upwards is hidden. The placement of the scrolling functionality within the Web page viewport reduces the pre-select inspection overhead for a user. Vertical fillings of the virtual sensors indicate the current scrolling status of the Web page. This in situ feedback about the scrolling status reduces post-select overhead, as a user can solely focus on the virtual sensors to scroll until the desired scroll offset is reached.

(b) The secondary interface mode, as shown in Figure 7.8, contains new-task-session functionalities. This includes functionalities to enter a URL, mark or choose bookmarks, and manage tabs. We enrich the secondary interface with information about the current Web page, like URL, title, and a preview of the viewport content. The presented information supports a user in selecting whether to load another



https://www.msn.com/en-us/news/video/roaring-crowds-greet-new-royal-baby/vi-AAweZlk

Roaring crowds greet new royal baby

Roaring crowds greet new royal baby — BBC News

Trump Welcomes Macron at White House — Associated Press

Search for motive after van driver kills 10 people in Toronto — CBS News

George H.W. Bush hospitalized with blood infection

Roaring crowds greet new royal baby

Duration: 01:46 | 23 hrs ago

The Duke and Duchess of Cambridge emerge from St Mary's in London with their newborn prince.

Figure 7.8: Secondary interface mode of GazeTheWeb with the new-task-session functionalities. Selection of the virtual button with the cross symbol loads the primary interface mode.

page or to change to another tab. The explicit exposure of functionalities through virtual buttons in the interface of the application reduces the count of required input actions and the inspection overhead, in comparison to the emulation approach.

We implemented the gaze-controlled Web browser, named “GazeTheWeb” [134],[4] using the Chromium Embedded Framework (CEF)[5] as Web engine and the eyeGUI framework [135] for the interface.

7.4 Conclusion

Following our propositions for improved user experience, we use the knowledge about the interface semantics in general and the interaction elements in specifics to adapt Web page interfaces and design a Web browser for gaze as input. Next, we need to evaluate whether the user experience is better in comparison to the emulation approach, whether the gaze-based interaction works for people with motor impairment, and whether the introspection and adaptation work for real-world Web usage. We think it is imperative to investigate how the gaze-based interaction impacts the user experience in terms of both, objective user performance in task execution and subjective usability and workload impressions.

[4] https://github.com/MAMEM/GazeTheWeb/tree/master/Browse

[5] https://bitbucket.org/chromiumembedded/cef

CHAPTER 8

Assessment of User Experience in Gaze-based Interaction

In this chapter, we propose and execute a methodology to assess the user experience in gaze-based interaction. There is a need to move beyond the concept of eye tracking being a useful input method, to be usable on a daily basis. This would lead to eye tracking being more acceptable as an interaction technology for end-users. We specifically evaluate the widespread feasibility of GazeTheWeb in adapting Web pages for gaze input. We assess its usability in handling dynamic Web pages with complex interaction elements and supporting everyday browsing tasks of end-users from the target user group, i. e., people with motor impairment. Hence, we come up with the following four hypotheses:

- **(H1) Speed:** Interaction with a gaze-adapted interface is faster than using the emulation of mouse and keyboard with eye tracking as the input method.
- **(H2) Usability:** Interaction with a gaze-adapted interface is more usable than using the emulation of mouse and keyboard with eye tracking as the input method.
- **(H3) Accessibility:** Gaze-adapted interfaces can be controlled by people with motor impairment with similar effectiveness as by people without motor impairment.
- **(H4) Scalability:** The method of retrieval, classification, tracking, and adaption of interaction elements on Web pages using a Mutation Observer works reliably for real-world Web pages.

First, our focus is to quantify how GazeTheWeb performs in comparison to an emulation approach in Section 8.1. We report a task-focused lab study on a simplistic search and Wikipedia browsing scenario, in which both GazeTheWeb and the emulation approach would be functional. Second, we aim to assess the feasibility of GazeTheWeb in the handling of dynamic Web pages with complex interaction elements and supporting everyday browsing tasks of users. For a realistic assessment, we primarily involved target users of gaze assistive technology to test the feasibility of GazeTheWeb, i. e., people with a motor impairment who would benefit from gaze-based interaction [101]. We conducted a controlled lab study to investigate if the users can accomplish the daily browsing activities such as communication (writing an E-mail, posting on social media), content creation (editing a photo), and entertainment (watching videos) using GazeTheWeb. We report the study in Section 8.2. Third, we have evaluated the overall feasibility of GazeTheWeb in a field study, installing GazeTheWeb at the home of participants for one month. We report the study in Section 8.3.

All three studies have been executed as part of the MAMEM research project.[1] The MAMEM project's goal has been to integrate people with motor disabilities back into society by enabling them to interact with computers using multimodal interaction channels.

8.1 Comparison of a Gaze-adapted Interface with Emulation

The goal of our first evaluation is to assess whether the implementation of the propositions in GazeTheWeb improves the user experience in comparison to the emulation approach. Therefore, we have conducted a lab study to evaluate GazeTheWeb against a state-of-the-art emulation approach, i. e., the comparative evaluation was a within-subjects design with the control method as the main independent variable including two levels: direct gaze control (GazeTheWeb) versus indirect gaze control (OptiKey with Google Chrome). We used OptiKey [179] as an instance of the emulation approach to control the popular Google Chrome browser.[2] OptiKey has received high praise in recent years as a tool to assist gaze-based computer access.[3,4] OptiKey has also been used at MIT[5] for supporting people with motor disabilities in computer access. You can find a detailed description of the gaze-based interaction with OptiKey in Appendix H.

For a reasonable comparison of OptiKey and GazeTheWeb, we used a simplistic information search and browsing scenario. This is because more complex interaction scenarios, like a video, are not accessible through the emulation approach. For example, on popular video platforms like YouTube, the videos embed their controls into the video element and only reveal them upon cursor movement. However, the emulation approach implements cursor placement rather than cursor movement. Hence, the controls of a video are not visible, and a user is not able to select them through the emulation approach. Therefore, we have chosen an interaction scenario where both approaches are functional for a user, such that the participants give us relevant feedback on the user experience.

Methodology

We quantify the task completion time, perceived workload, and usability. The main hypothesis is that GazeTheWeb allows for less time demand upon completing the search and browsing task, while users give higher ratings in usability and lower workload than for OptiKey. For this purpose, we conducted a lab study at our university with 20 participants (9 female and 11 male) in the age range from 23 to 31 years (age = 25.55 ± 2.16 years). All these participants were students at our university with no prior experience

[1] http://www.mamem.eu

[2] https://www.google.com/chrome

[3] http://www.businessinsider.com/an-eye-tracking-interface-helps-als-patients-use-computers-2015-9

[4] https://alsnewstoday.com/2016/03/08/article-for-als

[5] https://vimeo.com/148316508

in operating GazeTheWeb or OptiKey. Seven participants wore corrective lenses (3 female and 4 male). The participants were paid each an amount of ten euros for their effort after the trial. The eye-tracking device was attached to the bottom of a 24 in. monitor, which resolution had been set to 1,600 × 900 pixels. The used eye-tracking device was an independent variable between subjects, i. e., SMI REDn scientific eye-tracking device[6] (sampling frequency 60 Hz) was used for the study with ten participants. The other ten participants operated the software with a Tobii EyeX controller[7] (sampling frequency 30-70 Hz), which is an affordable eye-tracking device that is designed for consumer use. Artificial illumination and blocking of sunlight provided a similar lighting condition for all participants.

For both test levels (GazeTheWeb versus OptiKey and Google Chrome), dwell time has been used as a gaze-based selection method. The time threshold for the dwelling of both setups was set to one second, as appropriate for untrained users and chosen in related evaluations [154]. Depending on the setup, we analyze the required time to succeed in the tasks and the subjective analysis from the system usability scale (SUS) [26], NASA task load index (NASA-TLX) [73], design heuristics [164], and qualitative feedback. For the experiments, counterbalancing for the order of the setups was used to eliminate any bias of one setup over the other. In the rest of the section, the setups with GazeTheWeb and OptiKey and Google Chrome are referred to as *GTW* and *OK*, respectively.

Procedure

Before each execution, we gave the participants an explanation about the general nature of the tasks. After the oral explanation, we performed an initial eye-tracking system calibration and provided the participants with a training phase, which helped them to understand the functionality of the setups. Upon completion of the training, we granted them a short period to get accustomed to the setups on their own, so that they gathered certain confidence with a setup before the actual experimental session started. After a break of a few minutes, the experimental session started, including a second calibration of the eye-tracking system.

Tasks The tasks involved common user activities to search and browse the Web to find specific information. Each participant was instructed to enter the search string “Germany” on the DuckDuckGo search engine. They were asked to go to the English Wikipedia Web page about Germany from the search results. See Figure 8.1 for screenshots of both setups displaying the Web page with search results. Upon reaching the page about Germany, the participants had to find the section about the constituent states, and from there the first task was to select the state North-Rhine Westphalia (NRW). Upon reaching the page, they had to scroll to the cities of NRW and select the city Bonn from there. Once on the city page, they had to bookmark it and go back to the section of the constituent state of the NRW entry. The same

[6] SMI REDn scientific: `https://www.smivision.com/eye-tracking/products/remote-eye-tracking`

[7] Tobii EyeX controller: `https://tobiigaming.com`

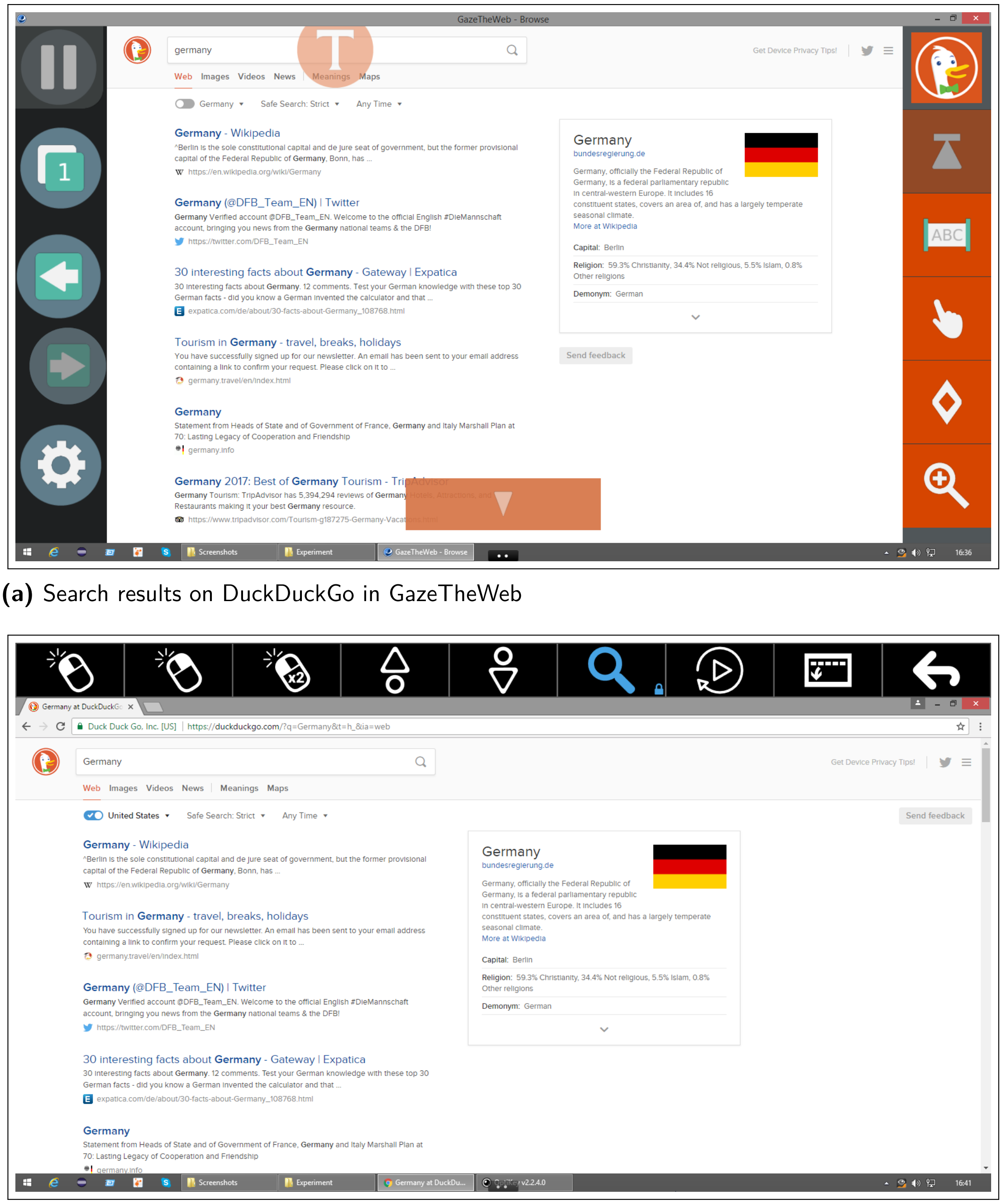
process was executed for the constituent state of Baden-Württemberg and the city Mannheim, respectively the state Lower Saxony and the city Oldenburg Land. Finally, the participants were instructed to access a city from the list of bookmarks.

Survey After the participants finished the tasks using one setup, they were handed a SUS and a NASA-TLX questionnaire. The SUS questionnaire is used to measure the overall usability of applications. The SUS contains ten questions, which are answered on a five-point Likert scale from *strongly disagree* to *strongly agree*. The NASA-TLX questionnaire contains six components: mental demand, physical demand, temporal demand, performance, effort, and frustration. For each component, the participant specified the most applicable scores on a scale from 1 (low) to 7 (high). In addition, a custom design heuristics evaluation for gaze-controlled interfaces has been used [164, 145], to analyze #1 How was the *visibility* of the main interaction elements? #2 How comfortable was the *size* of the interaction elements? #3 How *intuitive* was the reading and scrolling experience? #4 How *easy* was handling the link navigation in the browser? #5 How easy was it to recover from *errors* made? #6 How *close* do you feel is this browser to conventional browser environment? The questionnaire could be answered on a scale from 1, *strongly disagree*, to 10, *strongly agree*. In the end, participants were asked if they have any general feedback on what they liked, disliked, and comments for improvements.

Results

First, we tested the significance of the influence of the two eye-tracking systems. For this, we have performed unpaired significance tests on completion time, SUS, and NASA-TLX average raw value between the participant groups using a different eye-tracking system but the same software. Mann-Whitney U tests of completion time between the groups using different eye-tracking systems result in p = 0.43 with GTW and p = 0.85 with OK. Analogously, we performed Mann-Whitney U tests for the SUS scores that indicate p = 0.12 with GTW and p = 0.47 with OK. Mann-Whitney U tests for the average NASA-TLX raw scores indicate p = 0.23 with GTW and p = 0.91 with OK. Hence, the effect of the eye-tracking systems between participants was non-significant and we report the objective and subjective outcomes for the complete set of 20 participants from the evaluation in the following.

Objective Results All participants succeeded in the given tasks with both setups. The participants were on average over 135 seconds faster with GTW than with OK to complete the overall tasks. Furthermore, the time differences between both setups are normally distributed, according to a Shapiro-Wilk test with p = 0.05 as the threshold. This allows us to assess the significance of the differences by a paired t-test calculating the two-tailed p-value. We report a significant difference in the completion time for GTW (261.91 ± 49.25 seconds) and OK (397.43 ± 130.76 seconds), with t(19) = -5.23, p = 4.78E-5, and a high effect size of Cohen's d = 1.17 [38]. See Figure 8.2a for a box plot showing the consistent



germany

Web Images Videos News Meanings Maps

Germany ▾ Safe Search: Strict ▾ Any Time ▾

Get Device Privacy Tips!

Germany - Wikipedia
^Berlin is the sole constitutional capital and de jure seat of government, but the former provisional capital of the Federal Republic of **Germany**, Bonn, has ...
https://en.wikipedia.org/wiki/Germany

Germany (@DFB_Team_EN) | Twitter
Germany Verified account @DFB_Team_EN. Welcome to the official English #DieMannschaft account, bringing you news from the **Germany** national teams & the DFB!
https://twitter.com/DFB_Team_EN

30 interesting facts about **Germany** - Gateway | Expatica
30 interesting facts about **Germany**. 12 comments. Test your German knowledge with these top 30 German facts - did you know a German invented the calculator and that ...
expatica.com/de/about/30-facts-about-Germany_108768.html

Tourism in **Germany** - travel, breaks, holidays
You have successfully signed up for our newsletter. An email has been sent to your email address containing a link to confirm your request. Please click on it to ...
germany.travel/en/index.html

Germany
Statement from Heads of State and of Government of France, **Germany** and Italy Marshall Plan at 70: Lasting Legacy of Cooperation and Friendship
germany.info

Germany 2017: Best of **Germany** Tourism - TripAdvisor
Germany Tourism: TripAdvisor has 5,394,294 reviews of **Germany** Hotels, Attractions, and Restaurants making it your best **Germany** resource.
https://www.tripadvisor.com/Tourism-g187275-Germany-Vacations.html

Germany
bundesregierung.de
Germany, officially the Federal Republic of Germany, is a federal parliamentary republic in central-western Europe. It includes 16 constituent states, covers an area of, and has a largely temperate seasonal climate.
More at Wikipedia

Capital: Berlin
Religion: 59.3% Christianity, 34.4% Not religious, 5.5% Islam, 0.8% Other religions
Demonym: German

Send feedback

(a) Search results on DuckDuckGo in GazeTheWeb

Duck Duck Go, Inc. [US] | https://duckduckgo.com/?q=Germany&t=h_&ia=web

Germany

Web Images Videos News Meanings Maps

United States ▾ Safe Search: Strict ▾ Any Time ▾

Get Device Privacy Tips!

Send feedback

Germany - Wikipedia
^Berlin is the sole constitutional capital and de jure seat of government, but the former provisional capital of the Federal Republic of **Germany**, Bonn, has ...
https://en.wikipedia.org/wiki/Germany

Tourism in **Germany** - travel, breaks, holidays
You have successfully signed up for our newsletter. An email has been sent to your email address containing a link to confirm your request. Please click on it to ...
germany.travel/en/index.html

Germany (@DFB_Team_EN) | Twitter
Germany Verified account @DFB_Team_EN. Welcome to the official English #DieMannschaft account, bringing you news from the **Germany** national teams & the DFB!
https://twitter.com/DFB_Team_EN

30 interesting facts about **Germany** - Gateway | Expatica
30 interesting facts about **Germany**. 12 comments. Test your German knowledge with these top 30 German facts - did you know a German invented the calculator and that ...
expatica.com/de/about/30-facts-about-Germany_108768.html

Germany
Statement from Heads of State and of Government of France, **Germany** and Italy Marshall Plan at 70: Lasting Legacy of Cooperation and Friendship
germany.info

Germany
bundesregierung.de
Germany, officially the Federal Republic of Germany, is a federal parliamentary republic in central-western Europe. It includes 16 constituent states, covers an area of, and has a largely temperate seasonal climate.
More at Wikipedia

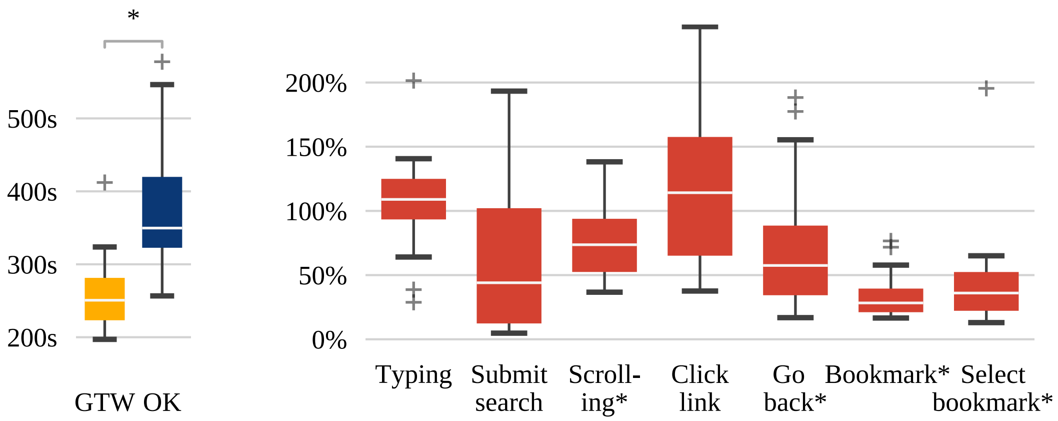
Capital: Berlin
Religion: 59.3% Christianity, 34.4% Not religious, 5.5% Islam, 0.8% Other religions
Demonym: German

(b) Search results on DuckDuckGo in OptiKey and Google Chrome

Figure 8.1: Screenshots of both setups from the first evaluation. The screenshots have been taken after the search query "Germany" had been submitted to the Duckduckgo search engine. Both setups display the Web page with the search results in the screenshots.

times the participant required to fulfill all tasks in GTW, whereas the times for OK showed much higher variety.

In a gaze-based interaction task on active stimuli, it is difficult to record exact timings of highly granular activities. In contrast to usual eye-tracking experiments (attention analysis on static stimuli, like images), there is a lot of information on the screen with a full set of Web browser controls and the Web page itself. Especially, it is not feasible to determine a hard threshold between inspection to *read* or inspection to *interact*, i. e., pre-select or post-select inspection.



(a) Completion time

(b) Times in GazeTheWeb in comparison to times in OptiKey as a baseline. 100% means a participant needed the same time in GazeTheWeb as in OptiKey, 50% means a participant needed only half the time with GazeTheWeb than with OptiKey for the same task.

Figure 8.2: Box plot of the times required by the participants for the execution of a browsing task. * marks a significant difference in the distribution of values between both setups in the reported dependent variable.

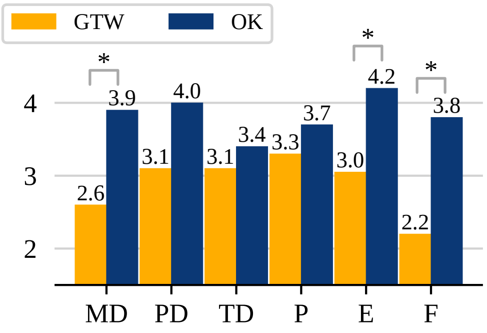
Thus, we report more details about the user performance by separating the task into atomic Web browsing activities, e. g., scrolling from the top of the page toward desired hyperlinks. We compare the timings within-subject in GTW with the timings in OK as a baseline and provide a box plot in Figure 8.2b. The timings for typing and hyperlink clicking appear to be similar, whereas submission of search, back navigation, and bookmark management have been achieved faster with GTW than using OK. We report as the result of two-tailed Wilcoxon signed-rank tests a significant improvement in times for GTW over OK for scrolling ($W = 20$, $Z = -3.17$, $p < 0.05$, $r = 0.71$), back navigation ($W = 41$, $Z = -2.39$, $p < 0.05$, $r = 0.53$), marking ($W = 0$, $Z = -3.92$, $p < 0.05$, $r = 0.88$) and selection ($W = 14$, $Z = -3.4$, $p < 0.05$, $r = 0.76$) of bookmarks.

Subjective Results The subjective evaluation of the experimental session uses the SUS for general usability assessment, the NASA-TLX to measure the workload, and custom heuristics criteria to evaluate the Web browsing experience.

The survey shows average SUS usability scores of 77.13 for GTW in contrast to 55.0 for OK, indicating an above-average acceptability rate of the gaze-adapted interface of GTW among the participants. According to a two-tailed Wilcoxon signed-rank test, the SUS score difference is significant (W = 28, Z = -2.8746, $p < 0.05$, r = 0.64).

The raw NASA-TLX scores are plotted in Figure 8.3a. The significant better ratings of GTW over OK in terms of NASA-TLX mental workload (MD) (W = 14.5, Z = -2.5842, $p < 0.05$, r = 0.67), level of effort (E) (W = 33.5, Z = -2.2646, $p < 0.05$, r = 0.53), and sense of stress and irritation (F) (W = 18.5, Z = -2.7456, $p < 0.05$, r = 0.67) using two-tailed Wilcoxon signed-rank tests. The physical demand (PD) received better ratings for GTW than OK, yet the difference cannot be shown to be significant using a two-tailed Wilcoxon signed-rank test. The feeling of success (P) in the accomplishment of the task was also slightly better for GTW in comparison to OK, since a lower value means a higher feeling of success in the NASA-TLX questionnaire design.

The results of the design heuristics questionnaire are shown in Figure 8.3b, where we can see that actions like the ease of recovering from errors (#5) in GTW receive better scores than in OK. The intuitive factor (#3) (W = 27, Z = -2.5477, $p < 0.05$, r = 0.6) and ease of hyperlink navigation (#4) (W = 30.5, Z = -2.5956, $p < 0.05$, r = 0.6) are rated significantly rated better for GTW, according to two-tailed Wilcoxon signed-rank tests.



(a) Raw NASA-TLX scores. Lower scores signify lower perceived workload. MD = Mental Demand, PD = Physical Demand, TD = Temporal Demand, P = Performance, E = Effort, F = Frustration.

Heuristics	GTW	OK
#1 Visibility	8.45	7.65
#2 Size	7.95	6.95
#3 Intuitivity*	8.0	6.05
#4 Link Navigation*	7.25	5.35
#5 Error Recovering	7.95	6.55
#6 Conventional	7.9	6.4

(b) Heuristics Scores. Higher scores are better.

Figure 8.3: Feedback by the participants of the lab study at our university. * marks a significant difference in the distribution of values between both setups using a two-tailed Wilcoxon signed-rank test.

Discussion

Objective results from the study indicate significantly lower overall task completion times for GazeTheWeb compared to the emulation approach. More specifically, we observe significant improvements in the activities like scrolling, back navigation, and bookmarking, which typically involve multiple selections and inspections. Typing of the search query and clicking of hyperlinks in GTW are similar to OK. This validates our experimental setup, as both setups employ similar mechanisms and selection methods. Both virtual keyboards, in GTW and OK, work with dwell-time-sensitive keys and are set to the same dwell time. Link clicking first needs the selection of the specific mode – in GTW by selection of the virtual button with the “finger-pointing” symbol on the right panel and in OK by a selection of the “left mouse click” virtual button – and then a multi-step magnification. More adapted Web browsing activities show the potential of gaze-adapted interfaces. The median of submitting the typed search query, scrolling, back navigation, and especially menu navigation like marking and selecting bookmarks are below the baseline of OK when using GTW, see Figure 8.2b. The results support our first hypothesis (H1), i. e., interaction with a gaze-adapted interface is faster than using the emulation of mouse and keyboard with eye tracking as the input method.

This strengthens our propositions to focus on more *usable* interaction experience and implies that the interface adaptation is an indispensable aspect of gaze-based interaction, irrespective of conventional issues like gaze signal quality. Hence, we argue that in addition to the technological advancements for more precise and accurate gaze signals, the research on interface user experience aspects needs to be evolved to make eye tracking more usable and acceptable interaction technology for end-users.

The subjective results from the study indicate a similar trend, i. e., a superiority of GazeTheWeb compared to the emulation approach, as the rating by the participants positions GTW far above the general median of setups. The average SUS score of 77.13 for GTW is close to the high order as per SUS guidelines.[8] A score equal to or greater than 80.3 indicates a positioning within the 10% of applications with very good usability and the tendency of users to recommend the software to a friend. The scores achieved by OptiKey are not satisfying, because the estimated score of 55.0 is close to the score of 51 and below, which would sort the usability of the application into the category among 15% of the worst evaluated applications. For the NASA-TLX scores, all major points of workload are rated better for GazeTheWeb. The participants felt especially less mental demand, less effort, and less frustration, but a higher sense of success. Our interpretation is that the additional command-translation layer of the emulation approach causes a higher overall workload, as the users had first to imagine how to achieve a task with mouse and keyboard and then to execute it via the gaze-controlled emulation tools. The heuristics questionnaire results validate that GazeTheWeb has a more suitable design for gaze-controlled Web interaction, as the feeling of controlling a conventional Web environment has been even higher for GazeTheWeb than for the combination of OptiKey and the well-known Google Chrome Web browser. All used metrics of SUS, NASA-TLX, and the custom heuristics questionnaire validate our second hypothesis (H2),

[8] http://www.measuringu.com/sus.php

showing that the gaze-based interaction is more usable with the gaze-adapted interface than with the emulation approach.

The explicit comments and feedback from participants are aligned with our hypotheses and the results. Most of the participants liked the intuitive interaction aspect of GTW, as comments by participants like *"It was easy to navigate and also very easy to get used to start working with it"* emphasize the good usability of GTW, whereas about OK a participant reported *"Too cumbersome and physically exhausting. Requires many clicks which are tiring for the eyes."* More specifically, participants explicitly reported about OK on their bad experience about required input actions, e. g., *"Multiple click for every action"* or *"Reduce left click mechanism using any other intuitive means. For example, let's say I am accessing the bookmarked URLs it should give easy access to the links by reducing the number of intermediate left clicks."* The improved aspect of inspection overhead in GTW was also evident, as the participants appreciated the visual indicators for optimizing pre-select inspection by comments such as *"Interaction elements are self-explainable as it was not at all in OptiKey"* and the in-situ feedback for optimizing post-select inspection in GTW *"I liked button feedback (if clicked)."* Regarding the high workload in the emulation approach, as indicated in NASA-TLX results, we have already discussed the issue that users might need to imagine how to achieve a task with mouse and keyboard and then to execute it via the gaze-controlled emulation tools. The explicit comments on OK usability substantiate the argument, e. g., *"Interaction elements should be easy to use, for a non-technical person it can be hard to use for example selecting button (like left click) every time after the [beginning of the] task"*, *"It seems too quick and [I] need to remember exact process I am following"* or *"Too complicated, too many interactions, too many clicks."*

Additionally, there were some general comments about head movement and fatigue as eye-tracking limitations that apply for both setups, e. g., comments such as *"The biggest problem is holding the head in the same position for a longer time period"* or *"Maybe [use] some eye tracking glasses, to make it possible to change the position of the head frequently."*

8.2 Lab Study about the Feasibility of a Gaze-adapted Interface

The lab study was conducted as part of the MAMEM project, first phase trial [147]. The complete trial for multimodal interaction was executed using various device configurations, i. e., an SMI REDn scientific eye-tracking device to estimate gaze, a BePlus LTM Bioelectric Signal Amplifier[9] to record electroencephalography (EEG) signals, and a Shimmer3 GSR+[10] to capture galvanic skin response (GSR), and heart rate (HR) signals. The trial consisted of four experimental sessions including (1) a training with gaze and EEG signals in a persuasive tutorial with gamification elements, (2) the recording of event-related potentials (ERP), (3) a sensorimotor rhythms (SMR) execution, and (4) dictated everyday-

[9] `http://www.ebneuro.biz/en/neurology/ebneuro/galileo-suite/be-plus-ltm`

[10] `http://www.shimmersensing.com/products/gsr-optical-pulse-development-kit`

Web-browsing-tasks using GazeTheWeb. In this thesis, we discuss the outcomes of the experimental session (4), as the dictated everyday-Web-browsing-tasks were performed solely with GazeTheWeb and eye tracking as the only input method.

Methodology

The trials were executed at three clinical sites, each responsible for one group of six participants with motor impairment. In total 18 participants with motor impairment took part in the lab study, which is a high number of participants in comparison to most other studies that evaluate eye tracking for interaction. Bates and Istance [16] let five participants test their zooming interfaces, Diaz-Tula and Morimoto [49] trained eight participants in the AugKey text entry system and Schenk et al. had twelve participants in their study about GazeEverywhere [159]. Moreover, the participants of those works did not have a motor disability — there are only a few studies about eye tracking for interaction that evaluate with people with motor impairment as participants, e.g., Ball et al. did a study with fifteen participants with ALS [12] using the ERICA software [116].

Six participants (1 female, 5 male) with Parkinson's disease (age = 64 ± 6.69 years) at the AUTH-School of Medicine, Greece; six participants (2 female, 4 male) with a neuro-muscular disease (age = 34.2 ± 6.18 years) at MDA Hellas, Greece; and another six participants (1 female, 5 male) with a spinal cord injury (age = 45 ± 16.4 years) at SHEBA-Academic Medical Center Hospital, Israel, participated in the trial. Additionally, 18 participants without motor impairment (5 female, 13 male; age = 45 ± 13 years) attended the experiments as the control group, with the same experimental setup as the target group. All 36 participants had no prior experience with eye tracking.

Before the trials, the clinical protocol was followed as per the ICH-GCP65 guidelines,[11] and ethical approval (Helsinki Approval) was obtained. The guidelines provide *"an international ethical and scientific quality standard for designing, conducting, recording and reporting trials that involve the participation of human subjects."*

Procedure

Before the experimental session of the dictated task was executed, all participants underwent an interactive tutorial with persuasion [147] as the first experimental session of the trial. The tutorial introduced the fundamental functionality of GazeTheWeb, necessary to fulfill the dictated tasks. The dwell time of the setup was set to one second, which is sufficient for untrained users and the same as for the lab study

[11] `http://www.ich.org/products/guidelines/efficacy/efficacy-single/article/good-clinical-practice.html`

[12] Centre for Research & Technology Hellas — Information Technologies Institute. Photographer: Tasos Papazoglou-Chalikias. 2017. MAMEM – Use the Computer using your Eyes & Mind. `https://www.youtube.com/watch?v=42yGmr3NE0k`

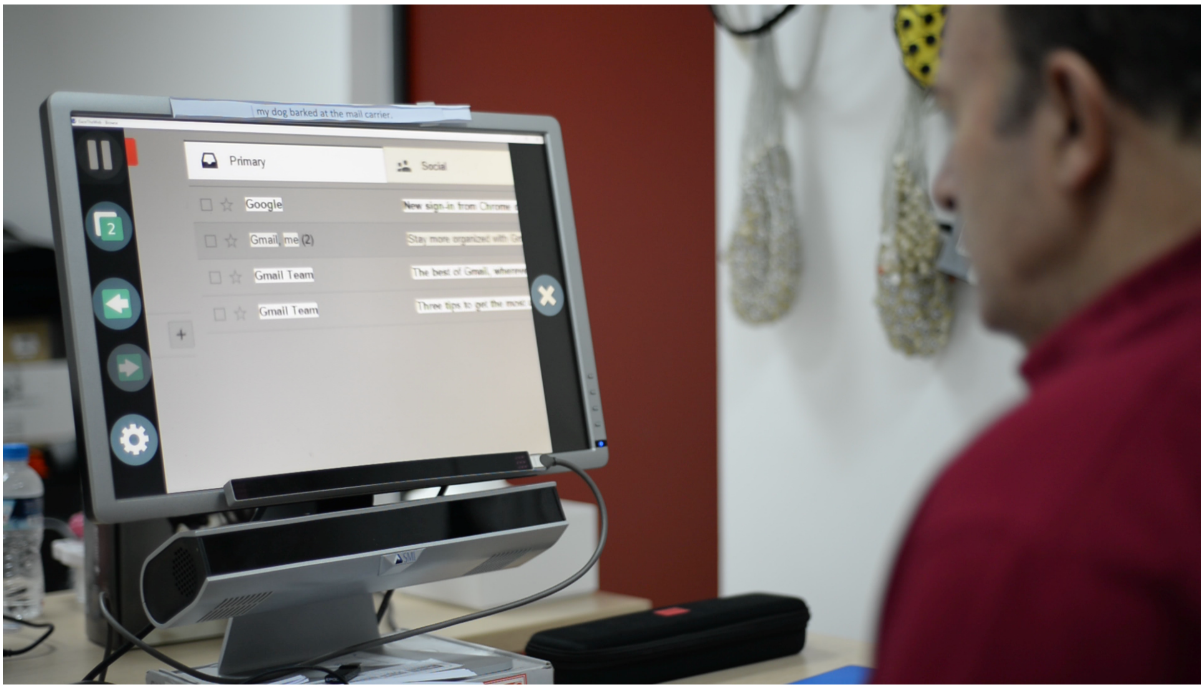


Figure 8.4: A photo from the MAMEM trials.[12]A participant with Parkinson's disease selects an E-mail.

at the university that we have described in the previous section. The participants were asked to perform dictated everyday tasks: To read and reply to an E-mail using *"Gmail.com"* (E-mail task), (see Figure 8.1) to edit a photo using *"Picresize.com"* (photo task), to post on social media using *"Twitter.com"* (social media task), and to watch a video using *"Youtube.com"* (video task). For more details about the single tasks within the experiment, please refer to Appendix I. Before the dictated tasks, the participants did a 15 minutes break from the trial. After the dictated tasks, the participants filled a SUS questionnaire. During the dictated tasks, the EEG, GSR, and HR signals were recorded for project-related purposes.

Results

Analogously to the previous section, we first present the results of the objective measures and then provide subjective feedback.

Objective Results GazeTheWeb successfully enabled the participants to complete the dictated tasks. The results are shown in Table 8.1, where the observation count indicates the number of participants that completed a task. There are some participants for whom a single or all tasks failed, which is indicated with the observation count. One participant with Parkinson's disease faced claustrophobia and could not proceed with the tasks; thus, was excluded from the experiment. For two participants with

Table 8.1: Task evaluation of target against control group, including E-mail task (E-mail), photo task (Photo), social media task (Social), and video task (Video). Times are provided in seconds.

	Measure	E-mail	Photo	Social	Video
Target group	Observation Count	15	15	15	15
	Time Mean	300.87	152.93	246.6	148.6
	Time SD	239.12	72.11	98.16	62.1
	Time Median	210	130	230	125
Control group	Observation Count	18	17	18	18
	Time Mean	255.17	144.06	253.22	157.44
	Time SD	147.39	76.34	99.44	68.56
	Time Median	202.5	148	220	135
Mann-Whitney U test	U-value	134.5	119	132.5	127.5
	p-value (two-tailed)	*1.0*	*0.76*	*0.94*	*0.8*

spinal cord injury, the eye-tracking device was not able to track their gaze. They were excluded from the trial as well. For one participant without motor impairment, there was an issue with the Internet connection during the photo editing task, why it could not be executed. Task completion times are denoted with time mean, standard deviation, and median measurements for the respective groups. The last row demonstrates the differences in task completion time between the participants of the target group and the participants of the control group. Because a Shapiro-Wilk test does indicate that the task completion times might be not normally distributed, we have decided to perform a Mann-Whitney U test, to compare the task completion times of the two unpaired groups of participants. The Mann-Whitney U tests of task completion times comparing target and control group reveal no significant difference between both groups, as all computed two-tailed p-values are by far greater than 5%.

Subjective Results The average SUS score of the target group is 72.17, while the average score of the control group lies at 77.36. These scores are above the average of 68 [26], indicating an above-average usability score of GazeTheWeb, analogous to the results of the lab study at the university from the previous section.

Discussion

The evaluation with the participants from the target groups demonstrates the feasibility regarding everyday tasks like writing an E-mail, editing a photo, participating in a social network, or watching a video. GazeTheWeb allowed the participants to successfully perform four real-world tasks on modern Web pages with eye tracking as the input method. The overall results support our third hypothesis (H3),

i. e., gaze-adapted interfaces can be controlled by people with motor impairment in similar effectiveness as by people without motor impairment.

Furthermore, the explicit comments and positive feedback from patients specify the acceptability of gaze-based interaction among target users. The participants appreciated the gaze-based interaction in GazeTheWeb, specifying comments such as *"I find it especially intriguing and fun that I can direct my actions with my eyes, rather than with my hands."* The participants also mention the persuasiveness of the technology *"I found this easy to learn, and the next thing that I want is to improve my speed at using the keyboard in this new way."* Some of the participants who were already comfortable with their existing means of interaction (e. g., switch input) liked the intuitiveness of the gaze-based interaction. However, at the same time comments like *"I am very curious to find out if it can exceed the comfort level that I have achieved with my current device"* indicate that the acceptability of eye tracking as assistive technology comprises challenges such as competing with existing means of hands-free interaction.

8.3 Field Study about the Feasibility of a Gaze-adapted Interface

There exist with over 1.6 billion a vast amount of Web sites,[13] and it is imperative to investigate the feasibility of GazeTheWeb in handling diverse Web pages that are frequently browsed in an everyday scenario. Hence, in this section, we report on the long-term usage of GazeTheWeb in a home environment. We present results from the second phase trials of the MAMEM project, in which GazeTheWeb had been offered to people with motor impairment for home use. The setup consisted of a laptop – with GazeTheWeb installed as a startup program – and an eye-tracking device deployed to the homes of the 30 participants for one month. The outcome of the field study does reveal whether the adaptations for gaze control are feasible in a real-world scenario of unrestricted Web browsing. Three participants had additionally performed a single-day multimodal experiment using an eye-tracking device, BCI, and GSR devices with a modified version of GazeTheWeb. However, we do not report the results from the multimodal experiment, as it is not in the scope of this thesis.

Methodology

Like the first phase of the MAMEM trials, each associated partner organization contacted participants of one target group. MDA Hellas, Greece, supported ten participants (4 female and 6 male, age = 31.5 ± 4.8 years) with neuro-muscular diseases. These participants are referred to as MDA 1 to MDA 10 in the following. AUTH-School of Medicine, Greece, supported ten participants (4 female and 6 male, age = 55.6 ± 7.3 years) with Parkinson's disease, referred to as AUTH 1 to AUTH 10 in the following. SHEBA-Academic Medical Center Hospital, Israel, supported ten participants (10 male, age = 38.1 ±

[13] `http://www.internetlivestats.com/total-number-of-websites`, accessed on 21st June 2019

10.8 years) with spinal cord injury, referred to as SHEBA 1 to SHEBA 10 in the following. The clinical protocol was followed as per the ICH-GCP65 guidelines and ethical approval was obtained.

A major aim of the study was to assess the natural Web browsing behavior of the participants. Hence, there were no guidelines, financial rewards, or requests to influence them for using the setup. Most of the participants already had some computer accessories and assistive solutions available at their homes. Therefore, we did not expect that the participants would completely switch their existing means of interaction and start using GazeTheWeb excessively. However, the persistent usage of GazeTheWeb even by some participants would be relevant indicators of its functionality and effectiveness in supporting daily browsing activities.

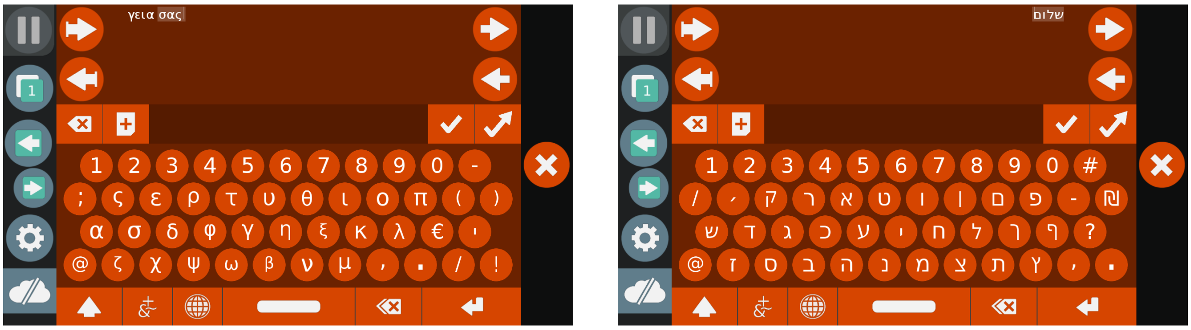
GazeTheWeb has been localized to match the language of the participants. The localization does not only include the screen texts of the interface, but also the interaction modes, i. e., the virtual keyboard for eye typing. The modified keyboard layouts for the Greek and Hebrew language to match the native language of the participants are depicted in Figure 8.5. The participants were able to choose between an English, a German, a Greek, and a Hebrew keyboard layout through a gaze-controlled drop-down menu (via the virtual button with the "globe" symbol on the lower panel of the interface mode for text input). The laptops dedicated for the field study were equipped with myGaze n eye-tracking devices by Visual Interaction GmbH, which perform dark pupil tracking at a frequency of 30 Hz. GazeTheWeb started automatically after the boot of the laptop and offered the participant to perform a calibration of the eye-tracking system. Furthermore, if the participant's eyes could not be detected for 30 seconds, the participant has been offered a recalibration upon return. The setup automatically logged the Web browsing activity to a Firebase[14] real-time database, e. g., the setup logged loaded Web pages and times, clicks, amount of inserted text, and general interface use.[15] The participants were allowed to disable all data recording by selecting a virtual button with a crossed cloud in the primary interface (see the virtual button beneath the "gear wheel" symbol at the bottom of the left panel in Figure 8.5).

Procedure

Before the field study, each of the participants had been visited to verify whether the eye-tracking device would work for them. Afterward, the setups had been placed for one month at the homes of the participants at an accessible place within their homes. At the time of deployment, a person from the medical supervisory team gave an introduction and initial guidance through the setup in the native language to the participant and caretaker. After the deployment, participants were free to use the setup as per their needs and preferences. The person from the medical supervisory team also provided telephone support in the native language, in case there were any issues during the usage. After one month of usage, the setup was collected back from the participants' homes, and the participants were asked to fill questionnaires including SUS to quantify their user experience.

[14] https://firebase.google.com

[15] Due to ethical reasons, we did not record the textual content itself but stored the string-edit distances.



(a) Greek layout with left to right direction of text.

(b) Hebrew layout with the right to left direction of the text.

Figure 8.5: GazeTheWeb has been localized to meet the constraints of the field study. The localization not only includes the screen texts but also the virtual keyboard layouts and text editing facilities.

Results

GazeTheWeb ran successfully for the entire one-month period, and we could observe the user activity through the log data on Firebase. More importantly, the participants could visit and interact with a variety of Web sites as per their needs and preferences, which signifies the usability of GazeTheWeb in supporting everyday browsing operations and handling dynamic Web pages. An average SUS score of 73.2 among all participants indicates the general acceptability and satisfaction delivered by GazeTheWeb. In the following, we report on the usage behavior of participants during one month.

Overview Throughout the field study, we have collected data during a setup run-time of 186.24 hours. The participants mostly used the system in the afternoon, as plotted in Figure 8.6. Because we could check for the presence of participants from the gaze signals, we may report about 118.93 hours of active participation in front of the setup. There has been regular use of the setup by some of the participants, as plotted in Figure 8.7. The participants have browsed 456 unique domains, on which they have visited 8,415 Web pages. See Table 8.2 for top-10 domains as browsed by the participants. In total, there have been 8,027 clicks on Web pages and 22,811 characters have been entered in text inputs. Furthermore, participants have browsed 498 URLs by typing the URL and added 189 bookmarks. They also made use of multi-tab browsing through 857 tab switches. We recorded 1,455 recalibrations of the eye-tracker system, i. e., not counting the initial calibrations.

Unsurprisingly, we found that the usage of GazeTheWeb follows a long-tailed distribution, with a few participants being very active and most participants using GazeTheWeb to a lesser extent. Less activity may come from a multitude of factors including (i) a low motivation of participants in general (including depressive moods of severely impaired subjects), (ii) a low interest in using a technical setup in general,

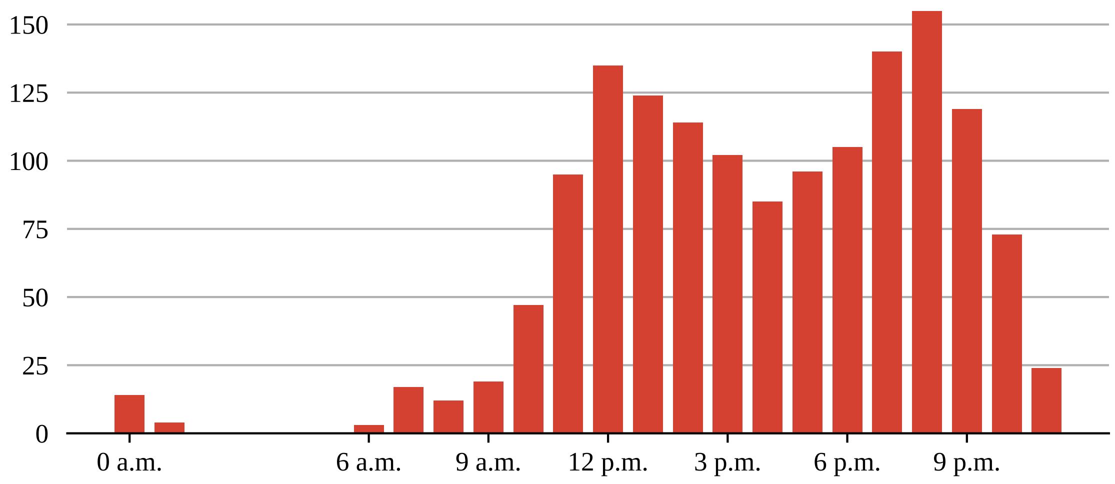


Figure 8.6: Starts of the setup by time of day.

(iii) a low interest in dealing with technical limitations of GazeTheWeb, or (iv) the inability to perform some activity with GazeTheWeb. With this field study, we do not target issues such as (i) or (ii), but we aim to understand the practical limitations of GazeTheWeb. Therefore, we assume that the more active participants are better representatives when it comes to judging the feasibility of GazeTheWeb.

Table 8.2: Top-10 visited Web sites by participants. "Visits" are caused through URL input, navigation, bookmarks, or history use. "Stays" are visits that exceed an active time of one minute, during which the participant had been registered by the eye-tracking system. "Pages" is the count of visited pages on the Web site. "Hours" is the time of browsing by a participant on the Web site. *"duckduckgo.com"* had been preset as a search engine.

Rank	Domain	Visits	Stays	Pages	Hours
1	facebook.com	369	229	1208	24.3
2	youtube.com	271	203	1396	26.5
3	duckduckgo.com	235	37	483	2.1
4	google.gr	123	24	270	1.6
5	mail.google.com	100	66	774	4.0
6	accounts.google.com	71	7	258	0.5
7	instagram.com	54	34	125	1.4
8	google.com	54	12	136	0.8
9	twitter.com	52	9	65	0.8
10	newsit.gr	50	46	257	3.7

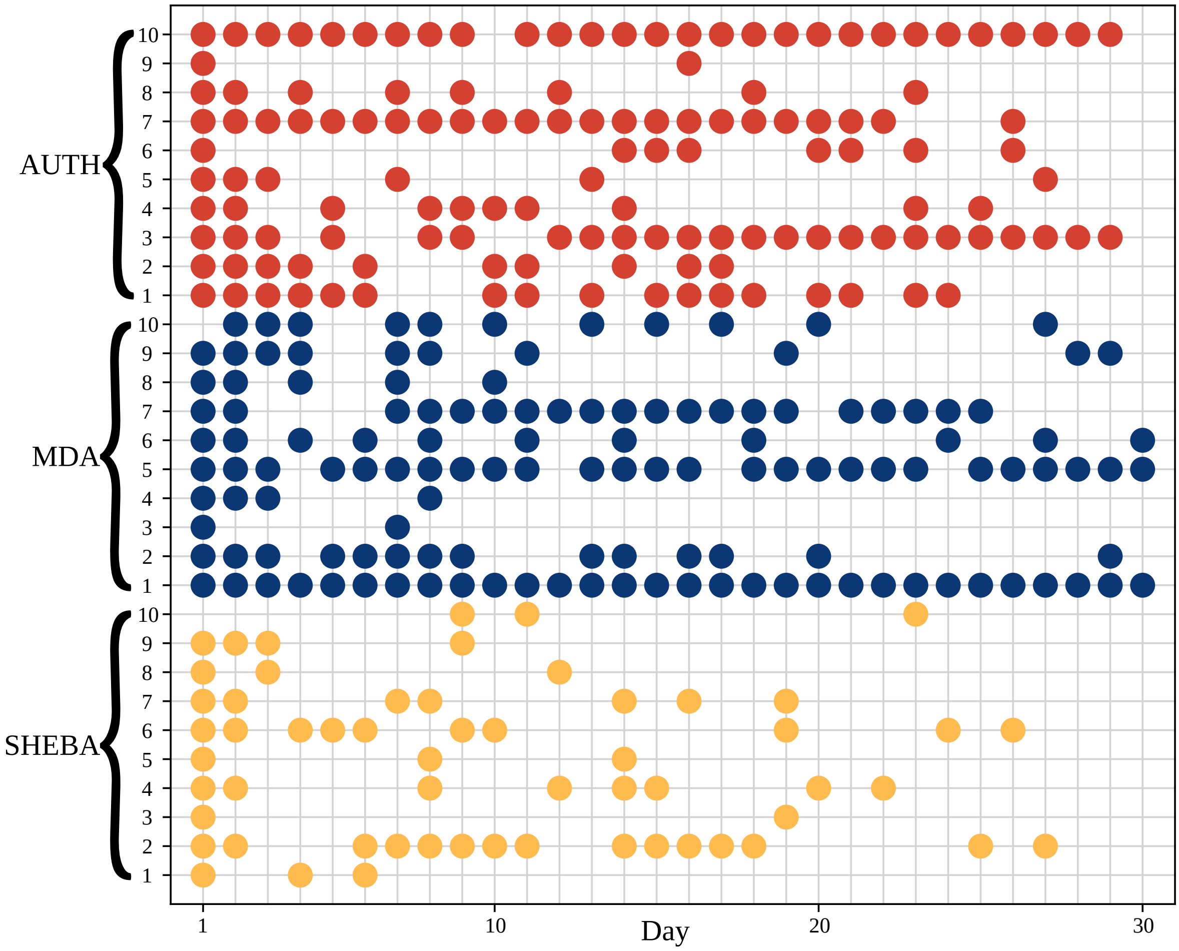
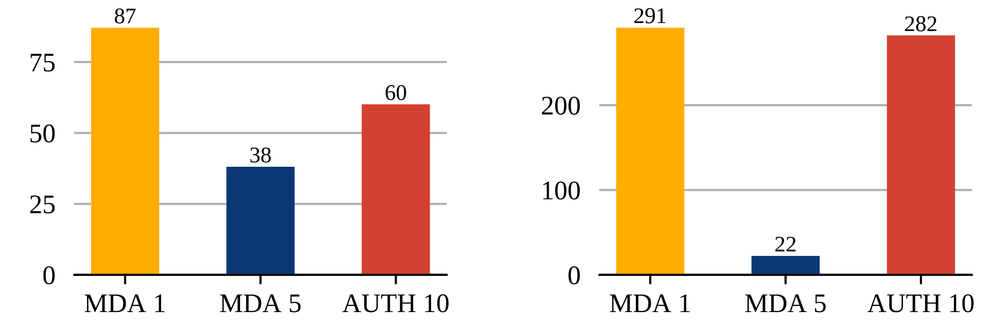


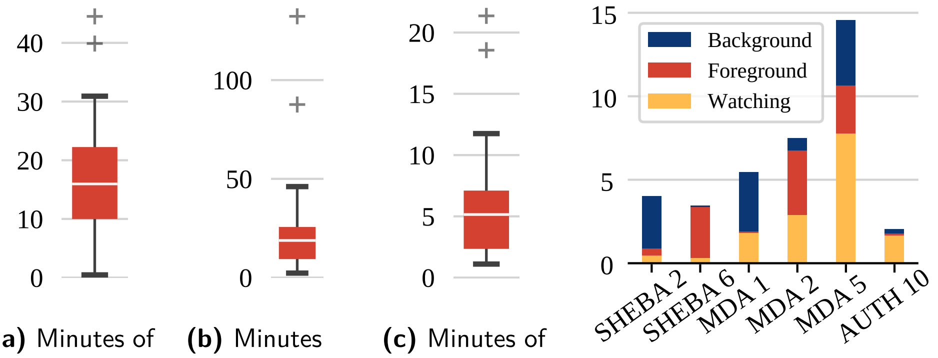
Figure 8.7: Daily use of GazeTheWeb in the field study. The vertical axis of the plot shows the participants. The horizontal axis displays the days since the setup of the setup at the homes of the participants. Each dot signifies the start of the setup by a participant on a specific day.

Most Active Participants The most active users have been MDA 1 with 23.27 hours in front of the setup, MDA 5 with 21.9 hours, and AUTH 10 with 27.75 hours. All three of them actively used bookmarks and two of them often switched between tabs, as signified in Figure 8.8. During the one month, the participants used many different Internet platforms with GazeTheWeb. Figure 8.9 plots the session times of each participant on their favorite platform daily. MDA 1 was very active on Facebook (Figure 8.9a), MDA 5 spent a lot of time on YouTube (Figure 8.9b), and AUTH 10 visited 29 times Google Mail, including 396 sub-pages, over 22 days (Figure 8.9c). In general, YouTube has been the second most visited domain and shows the highest number of spent hours. Some participants made use of the tabs in GazeTheWeb to browse the Web while a YouTube video was played in the background (Figure 8.9d).



(a) Usage of bookmarks by participants **(b)** Switching between tabs by participants

Figure 8.8: Usage of Web browsing functionalities in GazeTheWeb by most active participants.



(a) Minutes of MDA 1 on Facebook **(b)** Minutes of MDA 5 on YouTube **(c)** Minutes of AUTH 10 on Google Mail **(d)** Hours on YouTube

Figure 8.9: Prominent uses of GazeTheWeb in the field study. (a) shows a box plot of daily times that MDA 1 was actively on Facebook, day count = 30. (b) shows a box plot of daily times that MDA 5 was actively on YouTube, day count = 19. (c) shows a box plot of daily times that AUTH 10 was actively operating Google Mail, day count = 22. Values in (a) – (c) are provided in minutes. (d) shows the hours of different participants on YouTube video pages. Hereby, "Watching" is the time the eye-tracking system detected the participant in front of the screen; "Foreground" is the time the video has been visible but the participant was absent; "Background" is the time another tab was chosen visible.

Activities The diverse use of platforms across the participants can be also observed in interactions on the platforms. In Figure 8.10, the active minutes, clicks, and characters inputted have been plotted. We have clustered popular E-mail Web portals as E-mail activity, i. e., visits on *"mail.ru,"* *"live.com,"* *"outlook.live.com,"* *"mail.google.com,"* *"mail.yahoo.com,"* or *"mail.walla.co.il."* The use of the platforms is very heterogeneous among the participants. Additionally, we provide the five most visited domains,

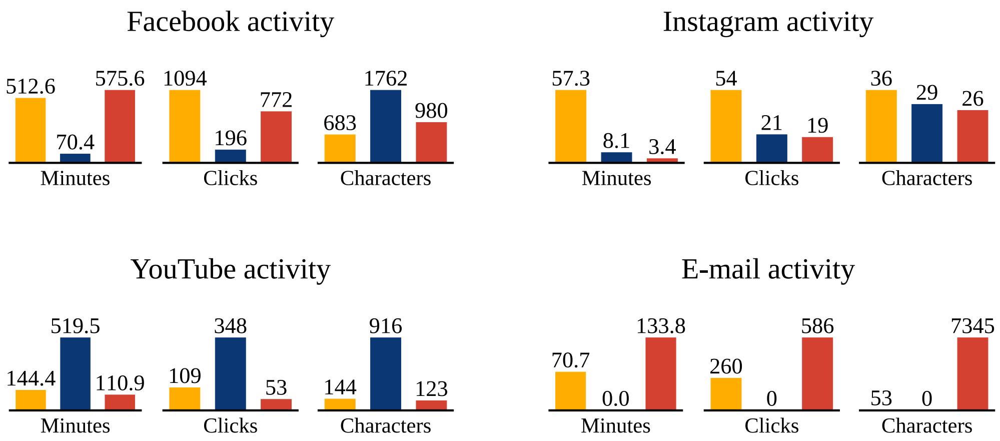


Figure 8.10: Web browsing activity of 🟧 MDA 1, 🟦 MDA 5, and 🟥 AUTH 10 on popular platforms.

Table 8.3: The five most visited non-social or communication domains by MDA 1.

Domain	Visits	Pages	Minutes	Clicks	Characters	Category
novasports.gr	18	88	81.4	85	0	Sports news
coursera.org	15	61	34.6	96	11	Education
sport24.gr	13	29	31.7	22	0	Sports news
nba.sport24.gr	12	16	8.7	6	0	Sports news
cosmote.gr	12	29	7.6	16	7	Mobile network operator

that we did not classify as social media or popular E-mail provider, of our most active participants, in Table 8.3, Table 8.4, and Table 8.5. The data shows that the participants – besides from social media and communication – used GazeTheWeb primarily to inform themselves about news and sports.

Table 8.4: Five most visited non-social or communication domains by MDA 5.

Domain	Visits	Pages	Minutes	Clicks	Characters	Category
newsit.gr	45	246	212.9	295	0	General news
google.gr	39	86	39.0	81	519	Search engine
zappit.gr	36	73	62.2	49	0	Gossip news
sport24.gr	28	31	22.7	9	0	Sports news
zougla.gr	18	47	51.6	28	0	Political news

Table 8.5: Five most visited non-social or communication domains by AUTH 10.

Domain	Visits	Pages	Minutes	Clicks	Characters	Category
accounts.google.com	51	159	10.1	7	21	Account management
webmail.auth.gr	39	310	71.1	435	1,302	E-mail (private provider)
slpress.gr	35	42	153.5	10	0	Political news
efsyn.gr	26	27	59.6	13	0	General news
lecturesbureau.gr	11	12	38.6	1	0	Arts

Discussion

The frequent usage of GazeTheWeb during the field study, including the visits of various Web sites, pages, and activities (e. g., clicks or text entry), indicates that users were able to interact with a variety of Web pages and to perform many different browsing operations. The longer stay duration and revisits of popular platforms like Facebook, Google Mail, and Youtube (incorporating complex interaction elements in a Facebook posting, an E-mail body, or video controls) imply that GazeTheWeb could effectively adapt the associated complex interaction elements for gaze-based interaction. The broad range of visited Web sites and the effective usage of popular social media portals supports the fourth hypothesis (H4), i. e., our method of retrieval, classification, tracking, and adaption of interaction elements on Web pages works reliably for real-world Web pages.

However, for the Web sites where the participants did not stay long or did not revisit, one could argue that interaction was difficult or even not functional. For example, the gaming domains specifically received less interest among participants, such as *"games.yo-yoo.co.il"* with two visits and a stay duration of 14 minutes, *"freegames.com"* with one visit for 1.8 minutes, and *"actiongame.com"* with three visits and a stay duration of 12.2 minutes. Ideally, the participants would stay longer in these domains, as the intent is to play the offered games. However, action games incorporate specialized controls for gameplay with a high frequency of mouse and keyboard input, which would be non-trivial to adapt for gaze-based interaction in the proposed methodology of GazeTheWeb. There has been only one stay at *"docs.google.com,"* with a duration of 6.2 minutes, and nobody has visited Google Maps. We incorporate these Web sites in our discussion about current limitations in the adaptation of interfaces for gaze input in the next section.

8.4 Limitations in Adaptation of Web Pages for Gaze Input

The evaluation of GazeTheWeb shows that gaze-based interaction is feasible with today's available eye-tracking devices and that gaze-adapted interfaces can offer a good user experience. We have identified the Web as an environment in which Web standards like HTML, CSS, and JavaScript make it possible to adapt the interaction with rich Web pages for eye tracking as the only input method. A vast majority of Web sites conform to the Web standards for interaction elements. Hence, the introspection and adaptation mechanism of GazeTheWeb works effectively with most Web pages that compose their content from those standard interaction elements. Figure 8.11 includes screenshots of GazeTheWeb displaying pages of Web sites from the Alexa Top 50 sites.[16] Here, (a) to (j) are examples of Web pages where interaction elements could be retrieved and adapted successfully through the general introspection and adaptation mechanism described in the prior chapter and implemented in GazeTheWeb.

Special Cases of Adaptations

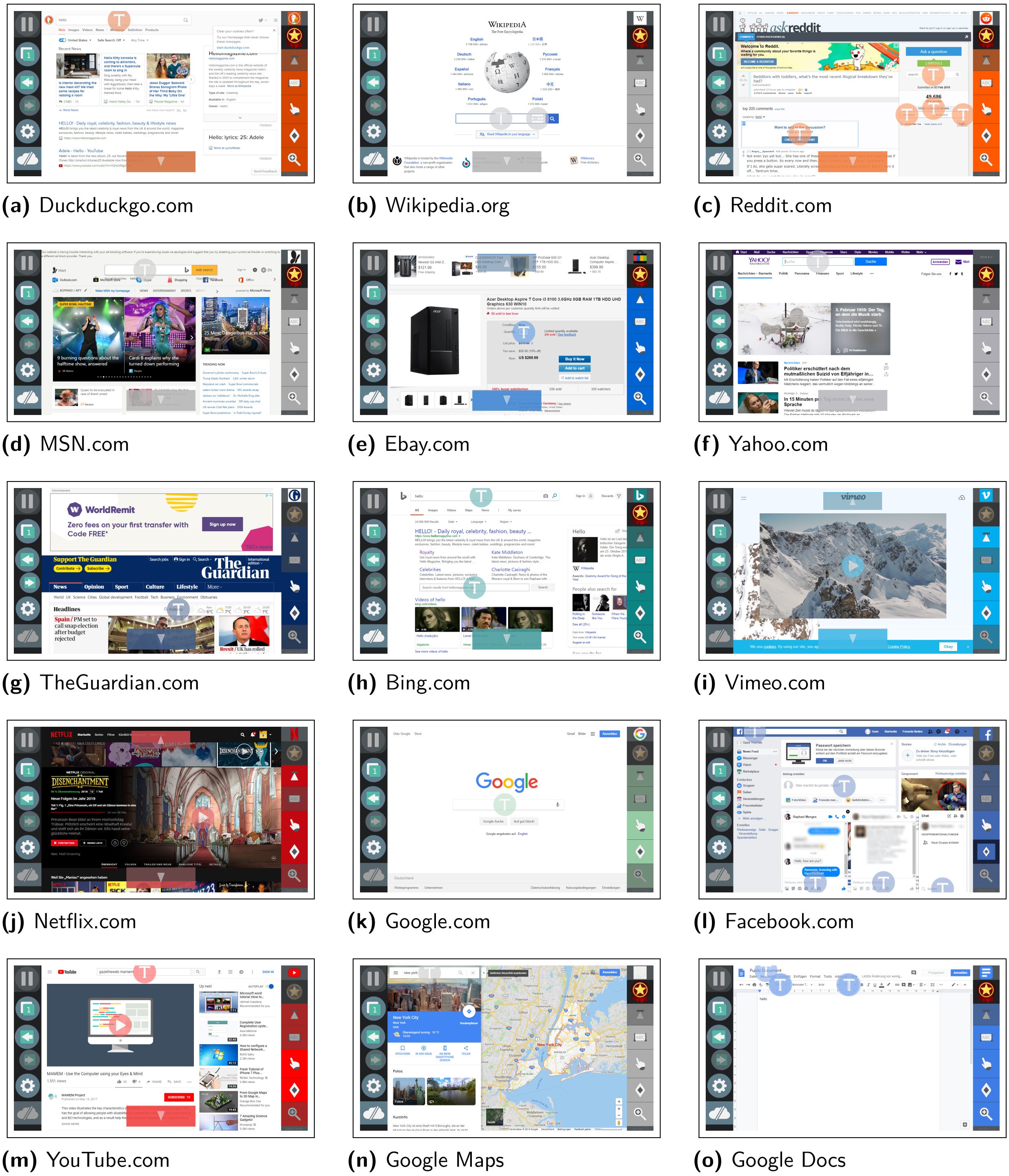
Web sites of some well-known Internet companies make use of compound elements with manually scripted behavior, which renders a ubiquitous adaptation difficult. Hence, a few popular Web sites required additional engineering to retrieve the relevant interaction elements. In Figure 8.11, (k) *"Google.com,"* (l) *"Facebook.com,"* and (m) *"YouTube.com"* are examples of Web sites for which we modified our adaptation approach as described in the following.

Google Search At the time of our assessment, there were at least two text inputs stacked onto each other for the search box on the Google start page. It appears that one text input is responsible to display suggestions in gray color and the other text input is used to display the actual user input in black color. The stacking is realized with the z-index attribute that defines the order in rendering. Thus, we have introduced a special case for the Google start page to recognize the text input that has to be filled with the phrase for the search query. One might argue for a general solution to adapt interaction elements by sorting according to their z-index. However, it is not easy to figure out which parts of an interaction element are in front when there are multiple elements with only partially overlapping areas.

Facebook Chat The Facebook chat prohibits the insertion of text into the text input of the chat window through JavaScript. The text appears for a short time after insertion and then it is automatically removed. This could be a strategy to avoid bots from spamming automatically on the chat. Hence, we implemented an alternative approach for text insertion in a Facebook chat window. Instead of inserting text through JavaScript, we take the text as received from the eye-typing interface mode and simulate keystrokes through the Web engine. We spawn keystrokes according to each character in the text, within a time window of a few milliseconds.

YouTube Video We faced two issues with video interaction. The first one applies to videos in general, the second one is specific to the YouTube video platform.

[16] `https://www.alexa.com/topsites`



(a) Duckduckgo.com

(b) Wikipedia.org

(c) Reddit.com

(d) MSN.com

(e) Ebay.com

(f) Yahoo.com

(g) TheGuardian.com

(h) Bing.com

(i) Vimeo.com

(j) Netflix.com

(k) Google.com

(l) Facebook.com

(m) YouTube.com

(n) Google Maps

(o) Google Docs

Figure 8.11: Various popular Web sites displayed in GazeTheWeb.

First, the full-screen video function does not work via a direct call from the JavaScript code. The `node.webkitRequestFullscreen()` mechanism employs the heuristic that the screen cannot be entirely covered without the consent of the user. However, the request for full-screen does work when activated from a button click event by the user. Therefore, we create an invisible button in the DOM tree upon entering the video interface mode, and we attach the function to make the video full-screen on the click event of that invisible button. A click by the user is simulated on the button, which makes the video full-screen. Afterward, we remove the invisible button from the DOM tree.

Second, when the described procedure is executed on a video on the YouTube platform, the video stays full-screen only for a short period in time and returns automatically to its page embedding. We were able to resolve this YouTube-specific issue by requesting full-screen not for the video itself, but the seventh-grade parent of the video node. We suspect that YouTube wants to avoid code injections that make the video full-screen while removing the advertisement overlays. The advertisement overlays are properly displayed with our approach, even though they are not embedded into the video file.

Challenges for Adaptation

There are more standards defined that might be considered to improve the adaptation even further, e. g., the `<nav>` tag,[17] which allows the identification of the menu bar of a Web page and which could be used for in-depth menu adaptations for gaze control. “Drag-n-Drop” has also been standardized;[18] thus, it is another candidate for adaptation in GazeTheWeb. Furthermore, there are numerous ARIA role[19] annotations available to include hints about the interface semantics to make the interface elements more accessible. We already use a small subset of AIRA roles in our classification of interaction elements, as described in Figure 7.2 on page 118. In the future, we would like to extend the support of additional AIRA roles like “Grid,” “Tree,” or “Menubar.”

Another challenge that has emerged in recent years is complex visibility configurations and CSS-based animations on Web pages. One example is the discussed text inputs on the Google start page, which are stacked over each other using the z-index. There is a plenitude of styling attributes that control the visibility of interaction elements on a local level, e. g., `visibility`, `opacity`, or `display`; and also attributes that control the visibility on a global level, such that the visibility is depending on the interplay of multiple elements, e. g., transform or z-index. The problem is less prominent than in gaze-based analysis (see Chapter 5), as we are only interested in interaction elements and we can rely on fallback mechanisms — like displaying different options to a user if we are not sure about the visibility configuration of the interaction elements. Nevertheless, we have initiated experiments using the `document.elementFromPoint` function of JavaScript, which returns the topmost element at a coordinate in the viewport. We can use that function to sample each interaction element and then decide about

[17] `https://developer.mozilla.org/en-US/docs/Web/HTML/Element/nav`

[18] `https://www.w3.org/TR/2010/WD-html5-20101019/dnd.html`

[19] `https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA/Roles`

its visibility, in addition to querying its styling attributes.

We observed that some popular Web applications that offer specialized controls using non-standardized interaction mechanisms do not work appropriately with GazeTheWeb. The sites (n) Google Maps and (o) Google Docs in Figure 8.11 showcase this scenario. A general adaptation for these compound elements is not feasible, as they base on specific scripts and non-standardized interaction mechanisms. Consequently, their accessibility is limited, i. e., the design hinders the adaptation of interaction for novel input devices. This issue is also evident in Web pages that attempt to prohibit bots to access certain areas of a page, e. g., *"CAPTCHAS."*[20] The latest versions of those mechanisms validate humans by their natural interaction through mouse and keyboard or touch events on Web sites. Neither gaze-based emulation nor our approach lets users pass those validations and users must choose alternative ways for validation. More semantic descriptions of interfaces would be desirable for successful adaptations for gaze control, or other input channels. However, not all common elements are standardized or semantically annotated, and we would have to infer the possibilities of interaction in different ways. For highly customized interaction contexts (e. g., map navigation, document editing, game controls), we imagine a future research approach would employ automatic interaction-template matching, i. e., to cater to different Web services with similar interaction behavior. For example, there are common patterns of interactions for map navigation (panning and zooming), regardless of the Web site that is offering the corresponding service like Google Maps,[21] Bing Maps,[22] or Open Street Maps.[23] An intelligent interaction-context recognition would detect the context of map navigation on these Web sites and offer a unified, gaze-adapted interface mode for map navigation. The interface mode would be the same for all map services, yet spawn interaction events as expected by the different services. This might be realized with an advanced interpretation of the JavaScript code and detection of the use of input events in the JavaScript code, which might be substituted with events caused by gaze input.

8.5 Conclusion

Eye gaze has been explored as a communication channel for interaction between human and computer systems for at least 30 years. However, most research and application have focused on how to compensate for the errors in gaze estimation and how to deal with the dual roles of the eyes to resolve the Midas touch problem. Since mouse as pointing device and keyboard as typing device has dominated the access to computers, considerable effort has been put in replicating the functionality of both interactions means with eye-tracking-based interaction. In the past, researchers and companies have taken gaze-control methods for these two interaction means and put them together into emulation approaches, to allow computer access for people with severe motor impairment.

[20] https://www.w3.org/TR/turingtest

[21] https://www.google.com/maps

[22] https://www.bing.com/maps

[23] https://www.openstreetmap.org

We discuss how the emulation approach introduces an overhead in human inspection efforts and input actions. We put up propositions based on common principles for a good user experience that aim to reduce the overhead of gaze-based interaction and visual search, taking the semantics of a controlled application's interface into account. To this end, we describe how to efficiently retrieve the interface semantics from dynamic Web pages. We provide details about our realization of the proposed principles in GazeTheWeb, a gaze-controlled Web browser that acts as a bridge between the Web and the eye-tracking environment. We utilize the proposed introspection method to retrieve the interface semantics of Web pages and to adapt the interaction elements, as well as to design the interaction with the Web browser for eye tracking as the input method. The approach of using interface semantics to adapt an interface improves user experience and has not been explored in the prior work of eye-tracking-based interaction.

We have evaluated the user experience of GazeTheWeb in comparison to the emulation approach. The results on task completion time, and more importantly the subjective measures like the outcomes of SUS and NASA-TLX surveys and heuristics questionnaire, signify that our proposed interface adaptations in GazeTheWeb improve the user experience in comparison to the current emulation approaches. This contrasts with eye-tracking research so far, which mostly assesses the speed and accuracy of isolated interactions like target acquisition with eye tracking compared to other modalities like mouse, keyboard, or touch. Furthermore, it emphasizes our contribution toward making gaze-based interaction not only useful but rather usable for daily use. We also validate the feasibility of GazeTheWeb in a lab and field study with the target users of eye tracking as assistive technology. The overall successful task completions and positive feedback from our lab studies, substantiated by long-term usage in a home environment, indicate that GazeTheWeb is effective in letting people with motor impairment execute everyday browsing tasks.

In this direction, the prototype of GazeTheWeb has already been used to integrate voice input and EEG for multimodal interaction in Web browsing environment [163, 93]. Moreover, the effect of persuasive technology with an artificial agent has been evaluated with GazeTheWeb [66]. We have also used GazeTheWeb to host a dedicated YouTube video player, which suggests further videos based on a user's attention [78]. Finally, other eye-tracking researchers foresee the integration of their algorithms in GazeTheWeb to improve gaze-based interactions [36].

Takeaway We argue that it is time for the eye-tracking technology to step out of the laboratory setting and be explored as input technology for daily use. Current research is mostly focused on micro-optimization of gaze-based interaction, e. g., finding the optimal dwell time, testing different selection methods, or adapting the size of interface elements. Often, the optimization of one aspect hinders the interaction in other terms. We conclude to shift the research perspective rather toward a macro-optimization of gaze-based interaction. The success of everyday use of eye tracking will depend on stable calibrations, intuitive interaction paradigms, and rewarding user experience. This will shift eye tracking as an input method beyond being an option for people with motor impairment into everybody's life. There is also a growing commercial interest toward eye tracking technology.[24,25,26] Thus, GazeTheWeb might be an important step toward the use of eye tracking in everyday applications by everyone.

The principles we outline in this work do not only allow for using gaze as a stand-alone input modality but also to combine gaze control with other input modalities providing for the intriguing potential to shape future computer-human interaction. There is research showing that interaction through eye tracking in combination with a trigger can be faster than a mouse alone [107]. In general, other modalities can remove the overload of inspection and selection from eye tracking and allow for an even more natural selection process than traditional input devices. Especially eye tracking in combination with voice input does offer huge potential, as both technologies compensate for each other's shortcomings, i. e., gaze provides the spatial context of attention, and voice plays an important role in confirming the intention of a user.

[24] `https://techcrunch.com/2017/06/26/apple-acquires-smi-eye-tracking-company`
[25] `https://techcrunch.com/2016/12/28/the-eye-tribe-oculus`
[26] `https://techcrunch.com/2016/10/24/google-buys-eyefluence-eye-tracking-startup`

CHAPTER 9

Outlook

The transformation of our society into the digital age is a steady process that impacts nearly all aspects of our life. We communicate via audio, text, or video chats through smartphones, we submit official documents like tax reports electronically with desktop computers, and we participate in online classes using tablet computers. The ways we interact with such digital media have still not converged, as the broad introduction of touch-screen phones about ten years ago and the huge improvement in voice assistants in recent years has shown. However, it has become clear that the most used digital environment is and will be the World Wide Web, with its open standards that allow it to be accessed on almost all modern computer devices. Web technology has already spread far beyond Web pages in a Web browser. Frameworks like Electron[1] and the Chromium Embedded Framework[2] allow deploying Web sites as desktop applications that look and behave like native applications. These frameworks are the foundation for popular applications like Microsoft Teams, Skype, Discord, Adobe Acrobat, Spotify, or Steam — which are all built using Web technologies.

The overall transformation of society toward the digital age is a challenge that all countries are undergoing. The European Union (EU) therefore has declared the process to transform our societies into the digital age as one core priority of today. Alongside, the EU has formulated high aims for this transformation so that all people can benefit from it [155]:

> *“Such a digital society should be fully inclusive, fair and accessible for all.”*

This signifies the importance of the goal of this thesis, to make digital services on the Web as usable as possible and accessible for all people.

Besides the potential positive effects of improving usability and accessibility on society, there is a growing commercial interest. The market for usability and user experience testing is driven by increasing revenues in e-commerce and mobile applications. The study “Application Testing Services” [132] from 2017 estimates the worldwide market for usability testing services with 26.59 billion USD in 2017 and predicts growth to 50.14 billion USD until 2022. Analogously, the market for support in digital accessibility also grows steadily. According to Tobii Dynavox, a brand of the market leader in eye-tracking technology Tobii AB that is specialized in accessible computing, at least 50 million people worldwide

[1] https://www.electronjs.org

[2] https://en.wikipedia.org/wiki/Chromium_Embedded_Framework

need assistive technology to communicate. This is also reflected in their annual sales of Tobii Dynavox systems, which have reached about 90 million USD in 2019. In comparison to the previous year, the value has increased currency-adjusted by nine percent [183]. But only 1 – 2 percent of the target population are using an assistive system that makes the digital environment accessible to them [181]. There is still a huge potential for growth in the accessibility market.

In the future, alternative means of input are not only useful for people with motor impairment but also healthy, aging adults. For older adults, it may take a long time or a huge effort to perform a pointing task with a mouse. Even touch screens may be difficult to operate for adults with arthritis or tremors [144]. Thus, people who need accessibility options indicated a desire to have multiple access options they can choose from (e. g., mouse, keyboard, joystick, touch screen, head tracking, eye tracking). We must ensure that a variety of access options are available to them quickly when they need them [12]. It is the best way of accessibility when people with an impairment can use the same interaction mechanisms as people without impairment. In recent years, the development of touch-screen devices with gesture control and voice-based assistants had already a positive impact on accessibility, as we got to know in conversations with participants of the MAMEM project. Eye tracking has the potential to add to the existing input means if it is available for more people. This would allow the analysis of usage patterns on much bigger datasets and induce novel interactions that can be also performed by people with motor impairment.

Our work encompasses methods to integrate eye tracking sustainably into the digital environment. The framework of visual stimuli discovery, to support the usability analysis, is using the rendering of a Web page. Thus, it will still be useful when standards of Web technology evolve and definitions change. Furthermore, our method to retrieve, classify, track, and adapt the control of interaction elements with eye tracking, which we use in GazeTheWeb, can be easily extended for interaction elements that might be introduced in the future. The propositions for a better user experience with eye tracking stay relevant in any case.

Eye tracking has been often declared to be a future technology for the mass, without fulfilling this promise. However, the latest developments in virtual and augmented reality headsets could finally make eye tracking available for a large number of users. The worldwide eye-tracking market was estimated with a volume of 456.3 million USD in 2017 and may grow to 1,818.1 million USD until 2024 [7]. This will positively impact the use of eye tracking to improve on usability and accessibility of the Web as the primary communication technology of the digital age.

APPENDIX A

Protocol for Recording of Dynamic Web Pages

In the following we provide the instructions as given to the participants in the dataset recording.

This dataset shall contain visual dynamics on Web pages. On each of the following Web sites, explore each visited page thoroughly. Hover over elements and menus, read descriptions, and navigate through carousels. You may also rest for some seconds and observe automatically changing slides. Try not to click on a hyperlink other than described in the tasks below.

Shopping

(1) `https://www.walmart.com` (click on the burger menu in the top-left corner and on carousel navigations)
   - (a) Click under “In The Spotlight” on “Toys,” in the bottom of the page
   - (b) Click on a toy of choice

(2) `https://www.amazon.com` (hover the departments and click carousel navigations, click on “quick look”)
   - (a) Click on “Deals in Electronics”
   - (b) Click on an item of choice.

(3) `https://store.steampowered.com` (click on the carousel navigations)
   - (a) Click on “Free Games”
   - (b) Click on a game of choice

News

(4) `https://www.reddit.com/r/pics/top/?t=month` (hover over user names in post replies)
   - (a) Click on the post “His adventure is just beginning”
   - (b) Go back to the overview and click on another post of choice

(5) `https://edition.cnn.com` (click on the burger menu in the top right corner)

   (a) Click under “entertainment” on “Screen” in the bottom of the page

   (b) Click on a review of choice

(6) `https://www.theguardian.com/international` (click on “More” in the menu bar)

   (a) Click on “Sport” in the top menu

   (b) Click on an article of choice

Health

(7) `https://www.nih.gov` (hover over the menu bar)

   (a) Click on “Health Information”

   (b) Click on an article of choice

(8) `https://www.webmd.com` (hover over the menu bar)

   (a) Click on “Eye Health” in the bottom of the page

   (b) Click on a “RELATED TO EYE HEALTH” topic of choice

(9) `https://www.mayoclinic.org` (click the burger menu on top)

   (a) Click on “Mayo Clinic’s campus in Arizona” in the bottom of the page

   (b) Click on a further hyperlink of choice

Cars

(10) `https://www.gm.com` (click on “OUR STORIES” in the menu bar, hover over the images)

   (a) Click on “Our Brands”

   (b) Click on a brand of choice

(11) `https://www.nissanusa.com` (click on the menu entries on top, click in the car carousel on the categories)

   (a) Click under “THE FUTURE OF NISSAN INTELLIGENT MOBILITY” on “Learn More”

   (b) Click a further hyperlink of choice

(12) `https://www.kia.com/us/en/home` (click in the car carousel on the categories)

   (a) Click under “Technology” on “Performance” in the bottom of the page

   (b) Click a further hyperlink of choice

APPENDIX B

Estimation of Scroll Offset

Across all Web sites, we have faced difficulties in aligning the scroll offset as recorded in the datacasts with the scroll offset that is visible in the video recordings. We could not find an affine transformation of the series of scroll offsets, e. g., adding a fixed delay to the timestamps, which would align the scroll offsets correctly. The non-synchronicity of reported scroll offsets and the visual scroll offset might be caused by the parallel execution of rendering, input processing, and script execution in modern Web engines. However, an incorrect measure of the scroll offset may trigger the visual change classifier falsely. Therefore, we have implemented a method to compute the scroll offset on a video recording using computer-vision features. In our method, we match features between consecutive frames and estimate a transformation that would explain the offset of the key points of the matching features between both frames. We interpret that transformation as the scroll offset.

Initially, we divide each frame in a 4×3 grid and compute in total 1,000 ORB [25] keypoints with descriptors on the Y channel (brightness information in the VP9 codec, see Appendix C for details). We exclude areas covered by fixed elements from the computations. Moreover, there might be elements on an interface, like buttons with a similar design, that occur multiple times. These elements can create highly similar descriptors, which would make a mapping between the descriptors of two frames ambiguous. Thus, we match the descriptors of a frame with each other and remove those pairs of descriptors that have a hamming distance of lower than five.

After having features for each frame, we match the features between two consecutive frames. We consider a feature pair with a hamming distance smaller or equal to ten as a match. We compute a homography with the RANSAC algorithm[1] on the matches, estimating the most probable transformation between the key points in the consecutive frames. We interpret the vertical translation component of that transformation as scrolling, which we use to adjust the scroll offset originally reported by JavaScript. Furthermore, we reset the scroll offset when a `document.hidden` event has been reported by JavaScript, as this event indicates that the Web browser is about to load a different Web page.

In the following, we show how scroll offsets differ between the datacast and our method. The x-axis represents the frame indices from the video recordings. The y-axis represents the scroll offset in pixels. The yellow line denotes the scroll offset as reported by the Qt WebEngine [39]. The red dotted line de-

[1] https://docs.opencv.org/master/d1/de0/tutorial_py_feature_homography.html

notes the scroll offset as reported by a JavaScript callback. The blue area (instead of a line for readability reasons) denotes the scroll offset as computed with our method. In our experience, the method works well most of the time and outperforms the methods relying solely on the Qt WebEngine or JavaScript callback. In case an incorrect estimation of the scroll offset is propagated to visual stimuli, we have counted them as incorrect in the computational evaluation in Section 5.5.

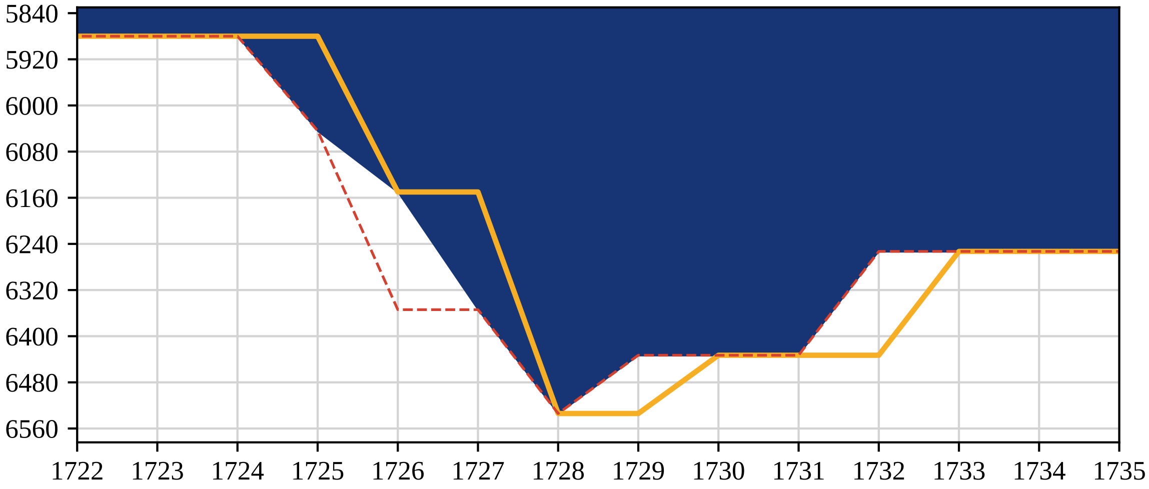


Figure B.1: A scroll sequence by the fourth participant on Amazon. General misalignment of scrolling as reported by the Qt WebEngine and JavaScript is visible.

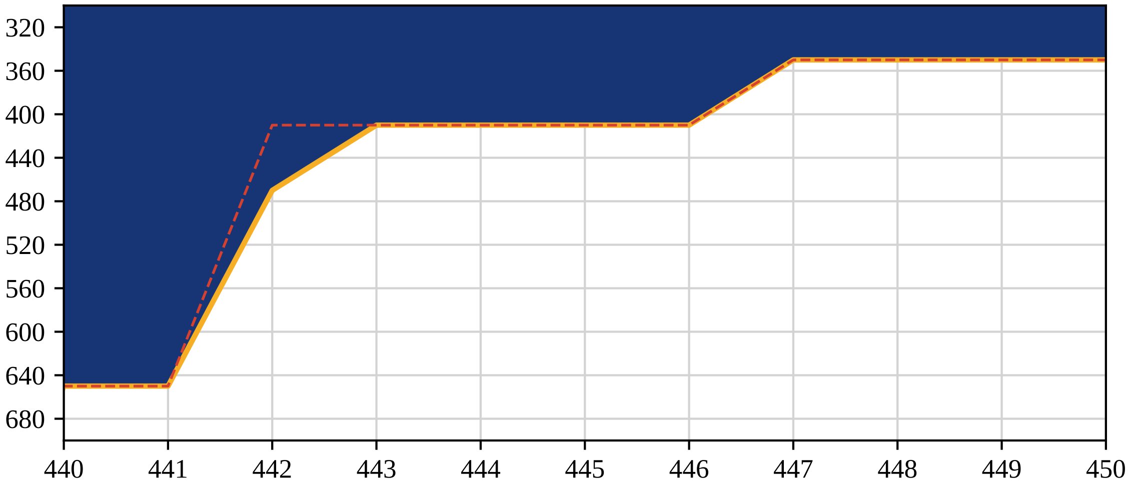


Figure B.2: A scroll sequence by the third participant on NIH. Here, our method agrees with the scrolling reported by the Qt WebEngine, but not with the scrolling reported by JavaScript.

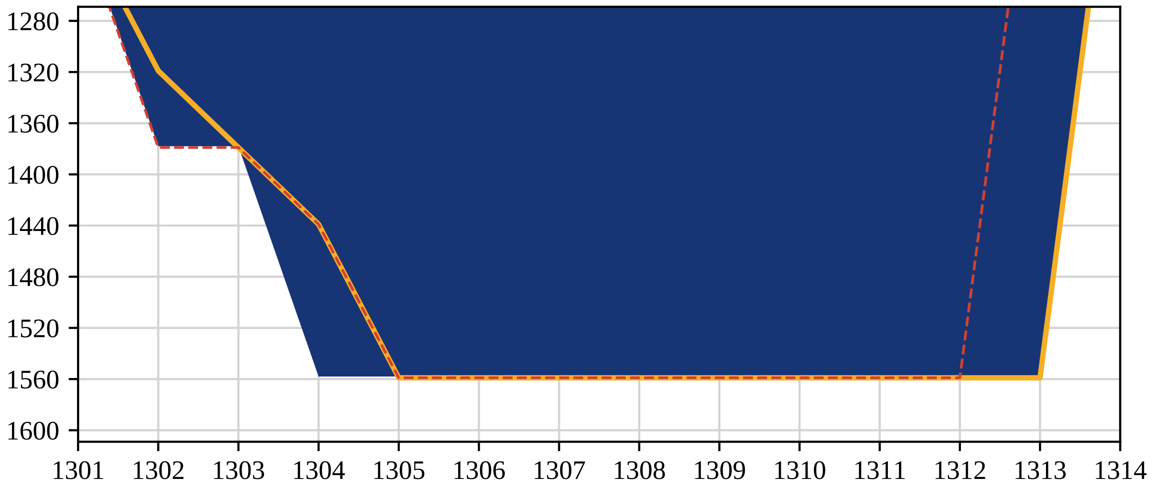


Figure B.3: A scroll sequence by the second participant on the Steam Web site. See frame 1,304 for a scroll offset that differs from the one observed in the video recordings, and frame 1,313 for JavaScript reporting about scrolling before rendering realized this offset.

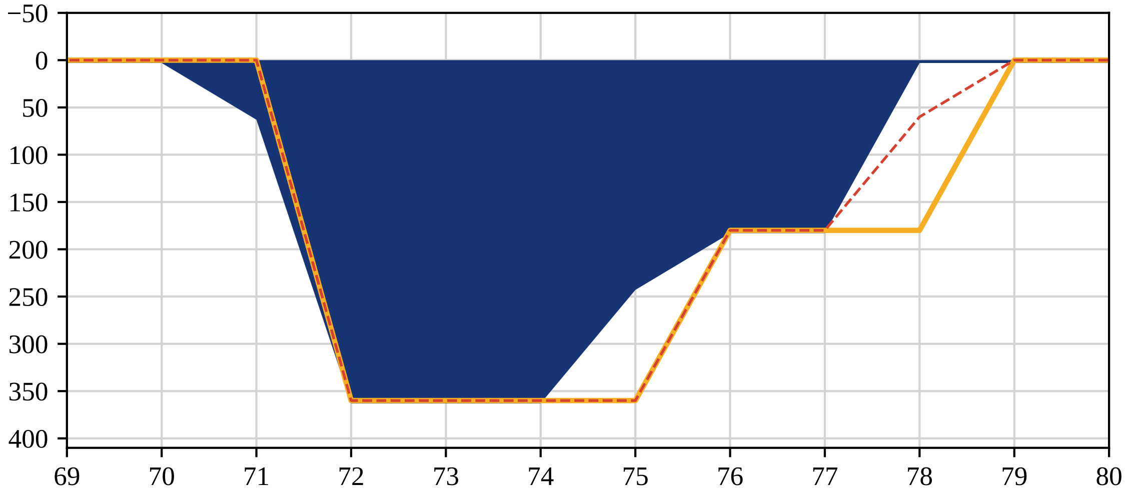


Figure B.4: A scroll sequence by the fourth participant on Amazon. General misalignment of scrolling reported by the Qt WebEngine and JavaScript is visible.

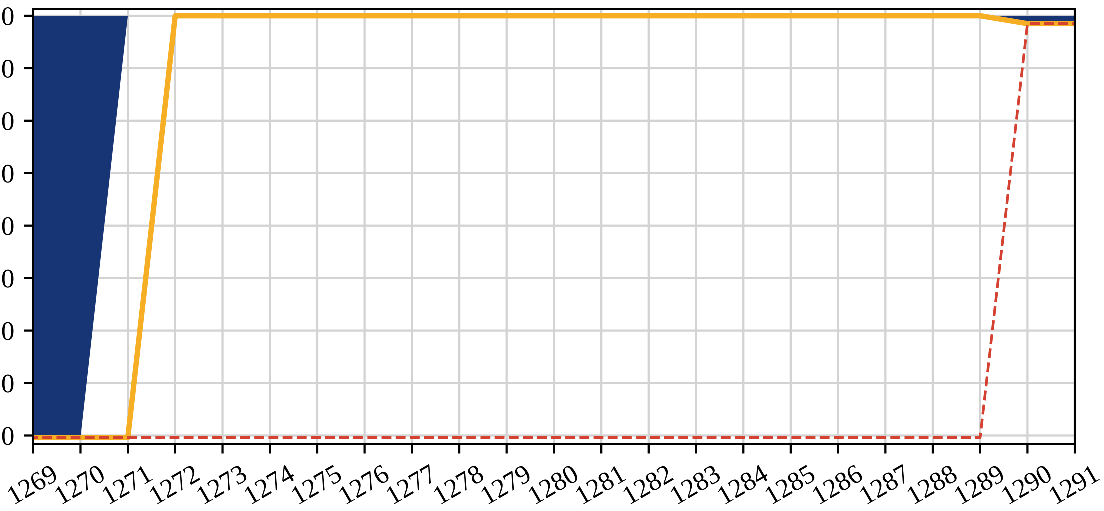


Figure B.5: A scroll sequence by the fourth participant on Amazon. On frame 1,271, a new Web page begins to load. The Qt WebEngine notices the change already one frame later. The JavaScript callback reports about the scrolling only after the Web page had been loaded and the user scrolled.

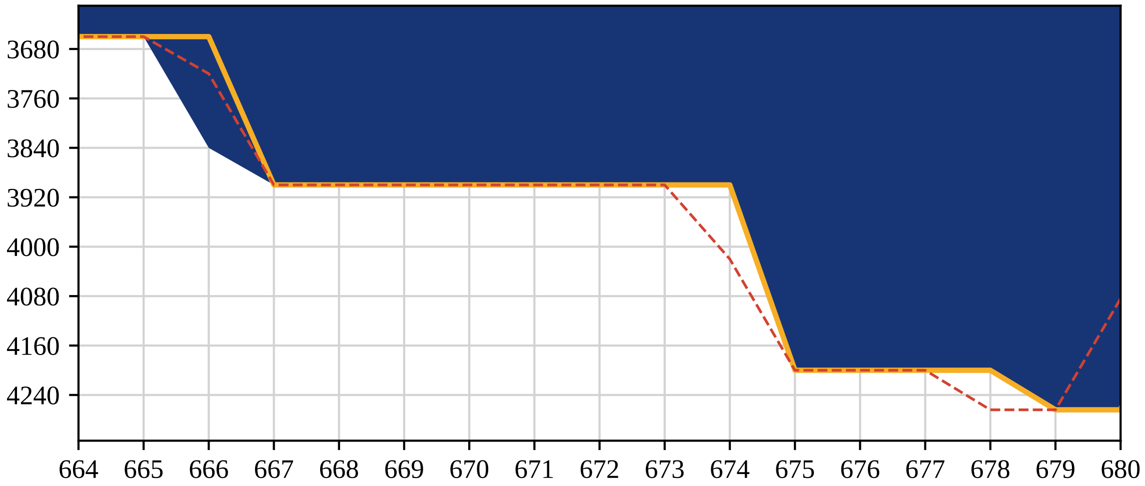


Figure B.6: A scroll sequence by the third participant on NIH. See frame 666 for wrong scroll values by both Qt WebEngine and JavaScript.

APPENDIX C

Features for Visual Change Classification

In this work, we propose to use computer-vision features to detect visual changes on video recordings of Web browsing sessions. We suggest 53 features as listed in Table 5.1 on page 64. In the following, we provide details about the features and variations of features, alongside the color spaces in which we calculate the features or variations, respectively. The features are designed to describe the difference between two image pairs, i. e., to detect a visual change from one frame in the video recording to the next frame in the video recording. The names of the features as listed in Table 5.1 and referred to in the work are given in parentheses.

Color Spaces An image from a video frame can be interpreted as a rectangular matrix of pixels. Each pixel holds information about its color. There are various color spaces to define the color. In our work, we employ the RGB, the HSL, and the YUV color spaces.

The RGB color space is the most common in computer vision. Each pixel holds brightness information about its red, green, and blue brightness. Almost all monitor panels nowadays consist of corresponding red, green, and blue subpixels.

The HSL color space is an alternative representation of the RGB color space and is more aligned with human perception of colors [91, 196]. It defines color as values of hue, saturation, and lightness.

The YUV color space is a color space that was used for analog TV broadcasts. The color is divided into a grayscale signal, the Y component, that can be straightforwardly interpreted by black-and-white TVs. The other two channels U and V contain the chroma signal. Those two channels can be interpreted by color TVs to enrich the picture with colors.

All video recordings in our dataset (Section 5.2) have been compressed using the VP9 codec. The VP9 codec[1] uses the YUV color space to store the color information of the video frames. It to stores the grayscale signal with full resolution in the Y channel, while downsampling the pixel resolution of a video frame for the chroma signal of the U and V channel. This is motivated by the fact that human perception is less sensitive to compression in grayscale – which defines the contrast of an image – than for compression of chroma [121].

[1] `https://www.webmproject.org/vp9/levels`

We take each video frame in the YUV color space and convert the colors to the RGB color space and the HSL color space. Moreover, we take the Y channel directly as a grayscale image. Certain features have been computed for multiple color spaces. We indicate the utilized color space in the description and the name of each feature. For example, the name `agg_h/s/l` describes the three variations of a feature that aggregates color values (`agg`). The three variations in `agg_h/s/l` are the aggregation in the hue (`h`), saturation (`s`), and lightness (`l`) channels.

Value-based Features Given an image pair, we can directly interpret the color values from the pixels. When there is a visual change between two images, some values in the pixel matrix change. We compare each pixel in one image with the pixel at the same coordinate in the other image and sum up the differences. We suggest two approaches for the summation of differences, resulting in two features that have eight variations each. First, we can sum up the value difference between corresponding pixels for various color spaces (`agg_bgr/b/g/r`, `agg_h/s/l`, and `agg_gray`). Second, we can count how many corresponding pixels have different values, regardless of the exact difference in the value. Again, this can be performed across various color spaces (`count_bgr/b/g/r`, `count_h/s/l`, and `count_gray`).

Histogram-based Features Histograms represent the distributions of values in the color channels of an image. Changes in the color distribution might indicate shifts in what is depicted on an interface, e. g., when a menu is expanded, or a viewport-filling gallery is shown. Given an image pair, we compute histograms with 16 uniform bins for each image, considering the RGB and HSL color space and the images as grayscale. Then, we compute the correlation between the corresponding histograms of both images (`corr_b/g/r`, `corr_h/s/l`, and `corr_gray`).

Edge-based Features Another popular method in image processing and recognition is to estimate discontinuities in images, for which human perception is especially sensitive. The edge change fraction [195] counts the pixels of edges that are either introduced or removed when transitioning from the preceding image to the succeeding. Given an image pair, the maximum count of those incoming edges and outgoing edges is considered as a feature (`change_fraction`). We extract edges from the images with a Canny filter [32]s on the grayscale image with a lower threshold of 64 and an upper threshold of 192. We set the dilation kernel to search for incoming and outgoing edges to five pixels diameter.

Signal-based Features Signal-based features are used in image processing to measure the quality differences between the two images. The features attempt to approximate the human perception of reconstruction quality in image compression. This is useful to estimate the effect of compression on image or video material and might be also useful to detect a visual change between two images. Given an image pair, we consider one image as the original image and the other image as the compressed image.

One signal-based feature we use is the peak signal-to-noise ratio (PSNR) on grayscale images. It considers the maximum intensity within the original image and the compressed image and relates the maximum intensity to the mean square error between the original image and the compressed image. We compute this feature for the grayscale version of the image pair (`psnr`).

Another signal-based feature we use is the structural similarity (SSIM) [191] index. The SSIM index compares pairwise windows from the original image and the compressed image. The comparison is performed with the combination of three characteristics of an image: luminance, contrast, and structure. The result is a decimal value in a range between -1 and 1. A value of 0 indicates no structural similarity. A value of 1 indicates perfect structural similarity. A negative value speaks for locally inverted image structures, yet is rather uncommon. The SSIM index can be calculated across an entire image, resulting in an SSIM map. We calculate the SSIM map using a sliding Gaussian window of size 11×11 pixels. The average of the SSIM map is reported as mean structural similarity (MSSIM). We calculate the MSSIM for the blue, green, and red channels of the RGB color space (`mssim_b/g/r`).

Optical-flow-based Features Objects in a video are recorded as a set of adjacent pixels. Optical flow assumes that between two contiguous video frames the set of pixels constituting an object either stays in one position or moves as a whole. Following this assumption, optical flow allows for recognizing moving objects. Given a pair of images, we compute a dense optical flow [63] on the grayscale version of the image pair. We report the mean and standard deviation of the angle (`angle_mean/std`) and the magnitude (`mag_max/mean/std`) of the detected movement.

SIFT-based Features The scale-invariant feature transform (SIFT) [122] algorithm is used to detect and describe local features in images. Given an image pair, we let OpenCV [25] choose 500 key points on the grayscale version of each image and compute a descriptor per keypoint. Then, we match the descriptors from both images using the Euclidean norm. We count the found matches and filter the matching descriptors with values of 0, 4, 16, 64, 256, and 512 as a threshold (`match/_0/4/16/64/256/512`). Matching keypoint pairs with a descriptor distance of zero are rare, whereas a set of matches at a descriptor distance of a value that is at least 500 contains almost all matches. We also check how many descriptor matches are also spatially close to each other on the images with a radius of less or equal to three pixels (`match_spatial`). The combination of descriptor matches and spatial matches – analogously to optical flow – can reveal moving objects in an interface. Furthermore, we report about minimum, maximum, mean, and standard deviation of the descriptor distances (`match_dist_min/max/mean/std`).

Text-based Features Interfaces often display substantial amounts of text. Frequently, dynamic interfaces present additional text on the fly, e. g., sub-categories in an online shop or pop-ups that provide details about a product. For example, when expanding the menu of a music shopping Web site, the words

“Phil Collins” may appear. Given an image pair, we detect text using the Tesseract 4.0[2] optical character recognition library. Tesseract employs an LSTM neural network architecture to detect text on images. We use the official training data for English as provided by the developers to train the neural network.[3] After the application of Tesseract on our data, we filter the detected text with a confidence value above 0.5 and erase non-ASCII characters.

We use a bag-of-words representation for comparing two sets of recognized words from the two images. It is to be noticed that one word can occur multiple times in a set. If we recognize two times “car”, the word “car” will be present two times in the set of recognized words. We exclude words of less than three characters. We match the words from the two images pairwise and report the number of terms that have no matching partner in the other image (`diff_words_count`. As another feature, we count the unique words in both images. For this, we take the two sets of words from the two images and erase duplicates in each set. Then, we erase words that occur in both images. Finally, we take the number of words left from both images (`unique_term_count`).

Another popular approach to compare texts is a character n-gram. Given a string, n-grams represent the string as the bag of all (possibly overlapping) substrings of length n. A string of length k has k-n+1 substrings of this kind. When considering n = 3 and the text “click here,” the n-grams would be “cli,” “lic,” and so forth. The tokens cover characters across words because space itself is considered as a character. Similar to Sikuli [194], a tool that uses image processing on the screen contents to recognize interface elements, we define n = 3 in our experiments. We consider the set of n-grams per image and find the matches. Then, we report features like the count of matches of n-grams between the two images (`n_grams_match_count`), the amount of n-grams (`n_grams_min_count/max_count`), the ratio of the count of matches against the minimum number of unique n-grams from both images (`n_grams_match_ratio`), the Jaccard similarity of the n-grams (`n_grams_jaccard`), and the vocabulary size of both images combined (`n_grams_vocabulary_size`).

[2] https://github.com/tesseract-ocr/tesseract

[3] https://github.com/tesseract-ocr/tessdata

APPENDIX D

Additional Visual Change Classifier

The table below shows the visual change classifiers when considering only value-based, edge-based, signal-based, and SIFT-based features. We employ four-fold cross-validation. One user session acts as training data, and three user sessions act as test data. We report F_1 scores and the shot-detection measures of coverage and overflow (excluding fixed elements). *R. Forest* is a random forest classifier. The best classifier result per Web site is printed in bold font. ↑ denotes that a higher value is better. ↓ denotes that a lower value is better.

Shopping Sites	Walmart		Amazon		Steam	
Classifier	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*
↑ Visual Same [F_1]	**94% ± 02%**	88% ± 02%	**95% ± 01%**	91% ± 01%	**92% ± 00%**	72% ± 03%
↑ Visual Change [F_1]	**88% ± 03%**	79% ± 02%	**85% ± 03%**	76% ± 01%	**81% ± 02%**	58% ± 01%
↑ Agg. Coverage	**0.96 ± 0.04**	0.81 ± 0.02	**0.96 ± 0.02**	0.87 ± 0.04	**0.91 ± 0.03**	0.82 ± 0.03
↓ Agg. Overflow	0.17 ± 0.07	**0.04 ± 0.03**	0.12 ± 0.04	**0.11 ± 0.03**	**0.08 ± 0.06**	0.11 ± 0.05

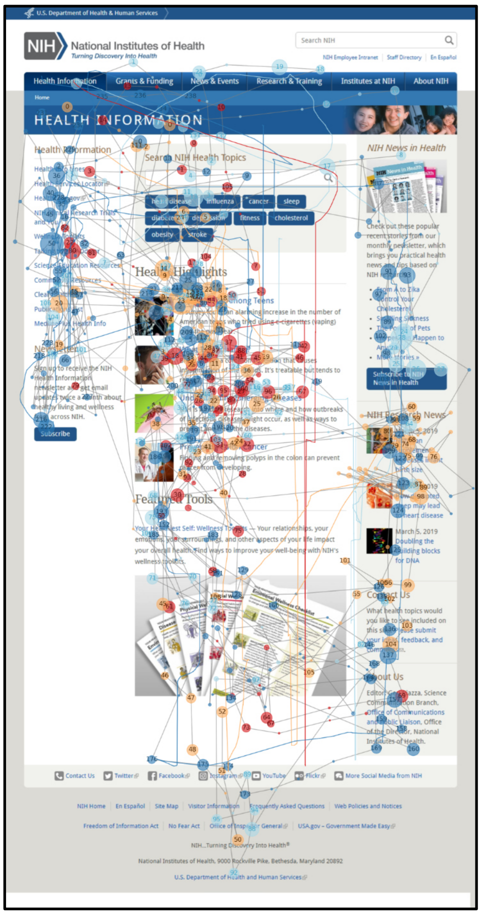
News Sites	Reddit		CNN		Guardian	
Classifier	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*
↑ Visual Same [F_1]	**96% ± 01%**	84% ± 03%	**93% ± 03%**	76% ± 07%	**97% ± 00%**	80% ± 02%
↑ Visual Change [F_1]	**74% ± 04%**	45% ± 05%	**60% ± 09%**	33% ± 01%	**79% ± 03%**	44% ± 03%
↑ Agg. Coverage	**0.96 ± 0.03**	0.75 ± 0.02	**0.83 ± 0.09**	0.53 ± 0.05	**0.87 ± 0.02**	0.51 ± 0.02
↓ Agg. Overflow	0.10 ± 0.05	**0.00 ± 0.00**	0.30 ± 0.20	**0.08 ± 0.13**	0.07 ± 0.05	**0.00 ± 0.00**

Health Sites	NIH		WebMD		MayoClinic	
Classifier	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*
↑ Visual Same [F_1]	**97% ± 01%**	87% ± 02%	**93% ± 01%**	86% ± 02%	**98% ± 00%**	94% ± 01%
↑ Visual Change [F_1]	**92% ± 03%**	72% ± 01%	**73% ± 02%**	63% ± 04%	**90% ± 01%**	79% ± 01%
↑ Agg. Coverage	**0.97 ± 0.02**	0.68 ± 0.09	**0.90 ± 0.03**	0.82 ± 0.04	**0.96 ± 0.02**	0.77 ± 0.02
↓ Agg. Overflow	0.11 ± 0.05	**0.00 ± 0.00**	0.13 ± 0.05	**0.07 ± 0.02**	0.06 ± 0.01	**0.05 ± 0.03**

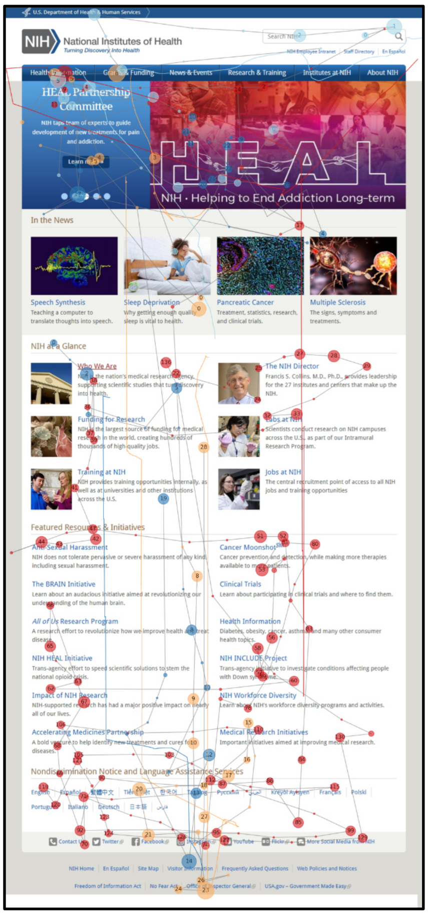
Car Sites	General Motors		Nissan		Kia	
Classifier	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*	*R. Forest*	*Baseline*
↑ Visual Same [F_1]	**97% ± 00%**	83% ± 03%	**97% ± 00%**	93% ± 01%	**97% ± 00%**	93% ± 01%
↑ Visual Change [F_1]	**82% ± 01%**	40% ± 03%	**90% ± 01%**	79% ± 02%	**84% ± 03%**	48% ± 01%
↑ Agg. Coverage	**0.91 ± 0.03**	0.69 ± 0.10	**0.96 ± 0.03**	0.79 ± 0.04	**0.90 ± 0.05**	0.80 ± 0.03
↓ Agg. Overflow	0.24 ± 0.07	**0.21 ± 0.19**	0.13 ± 0.06	**0.10 ± 0.10**	0.14 ± 0.10	**0.04 ± 0.07**

Discovered Visual Stimuli on NIH.gov

In the following, we show all thirty visual stimuli that we discovered in the video recordings of four users on NIH.gov, excluding fixed elements. Mouse traces and scanpaths by the four users have been added as overlays. Each user is represented by a unique color.

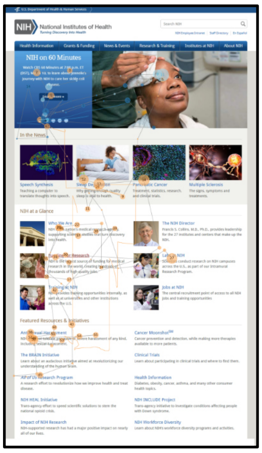


nih\stimuli\1_html\10.png



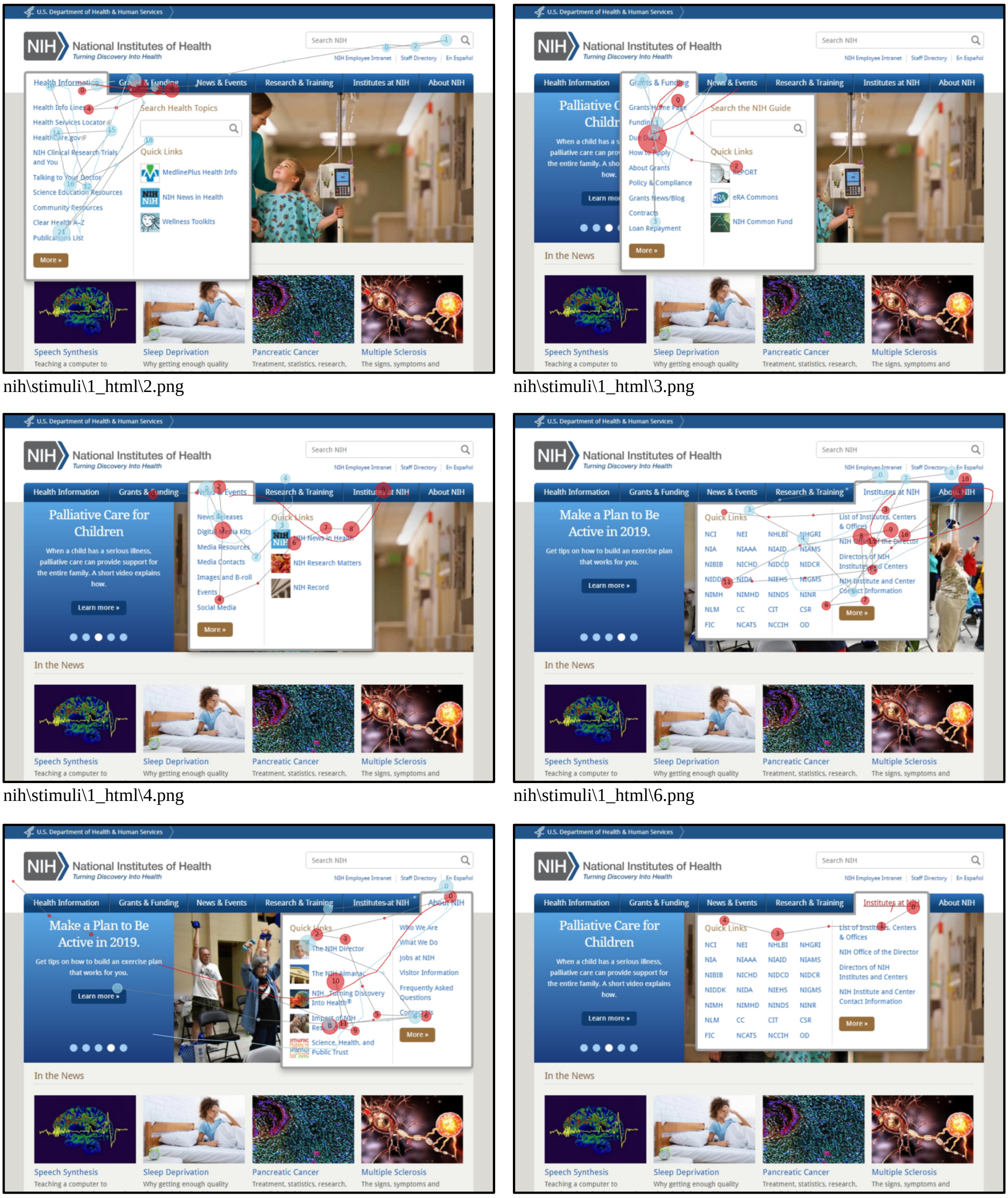
nih\stimuli\1_html\0.png

nih\stimuli\1_html\1.png



nih\stimuli\1_html\13.png

Figure E.1: Four visual stimuli discovered in the user sessions on NIH.



nih\stimuli\1_html\2.png

nih\stimuli\1_html\3.png

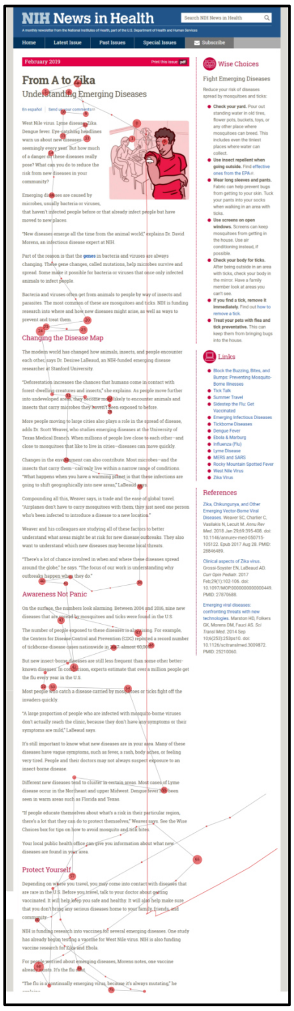
nih\stimuli\1_html\4.png

nih\stimuli\1_html\6.png

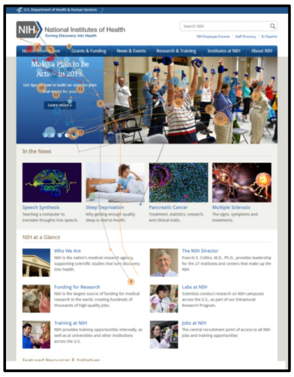
nih\stimuli\1_html\7.png

nih\stimuli\1_html\5.png

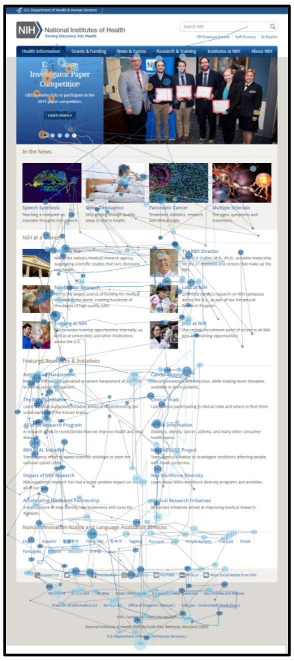
Figure E.2: Six visual stimuli discovered in the user sessions on NIH.



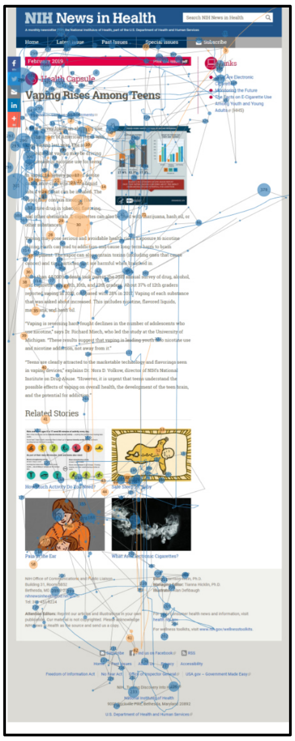
nih\stimuli\1_html\11.png


NIH National Institutes of Health
Making a Plan to Be Active in 2019
In the News
NIH at a Glance

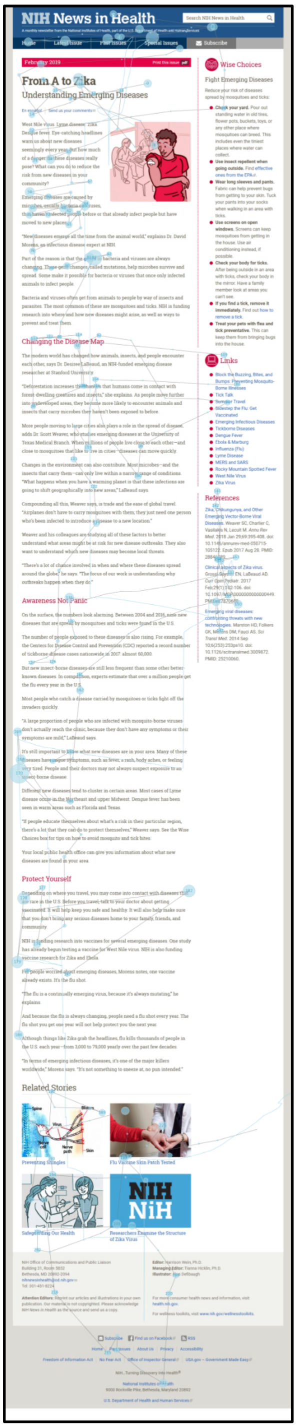
nih\stimuli\1_html\8.png


NIH National Institutes of Health
Investigator Paper Competition
In the News
NIH at a Glance
Featured Research & Initiatives

nih\stimuli\1_html\12.png

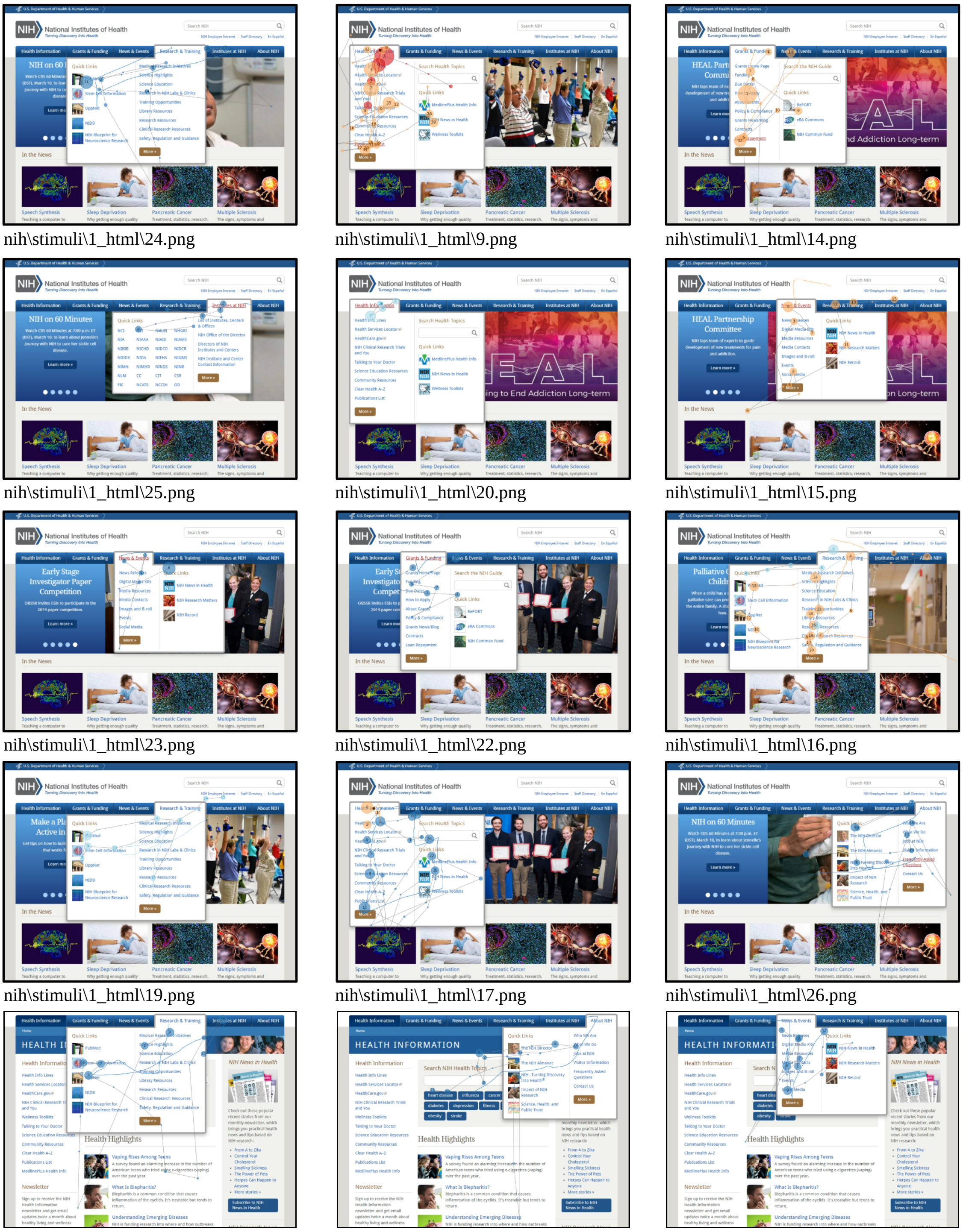


nih\stimuli\1_html\18.png



nih\stimuli\1_html\21.png

Figure E.3: Five visual stimuli discovered in the user sessions on NIH.



nih\stimuli\1_html\24.png

nih\stimuli\1_html\9.png

nih\stimuli\1_html\14.png

nih\stimuli\1_html\25.png

nih\stimuli\1_html\20.png

nih\stimuli\1_html\15.png

nih\stimuli\1_html\23.png

nih\stimuli\1_html\22.png

nih\stimuli\1_html\16.png

nih\stimuli\1_html\19.png

nih\stimuli\1_html\17.png

nih\stimuli\1_html\26.png

nih\stimuli\1_html\28.png

nih\stimuli\1_html\29.png

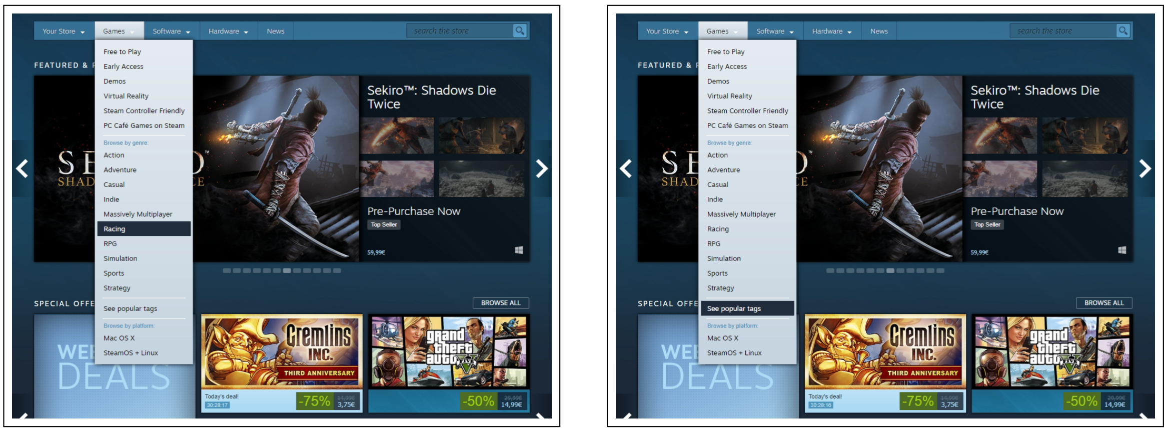
nih\stimuli\1_html\27.png

Figure E.4: Fifteen visual stimuli discovered in the user sessions on NIH.

APPENDIX F

Examples from the Computational Evaluation of the Visual Stimuli Discovery

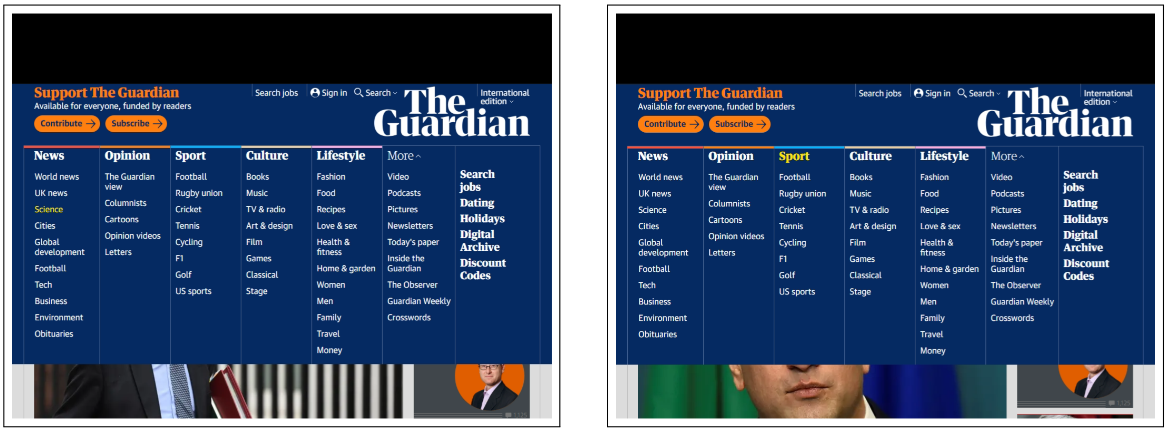
The computational evaluation in Section 5.5 shows that visual stimuli can reduce the amount of information that a usability expert has to examine substantially while reflecting the contents of the video recordings correctly. However, our implementation of visual stimuli discovery does not produce perfect results, i. e., sometimes the visual change classifier fails in detecting a visual change. Following, we show examples from the annotation process for the computational evaluation. On the left, we display the original frame from the video recording. On the right, we display the region from the visual stimulus that should represent the frame on the left. We explain for each example how we decided about the correctness in the representation of the frame in its respective visual stimulus.



(a) Frame from the video

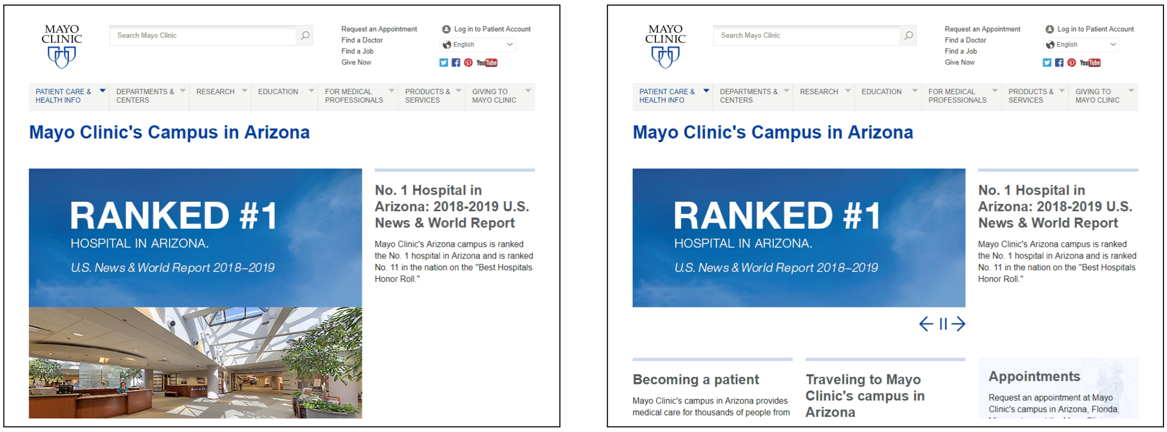
(b) Corresponding region in the visual stimulus

Figure F.1: On the left is frame 587 from the video recording of the second participant on Steam. On the right is the visual stimulus that represents that frame, cropped to the corresponding region. The frame has been marked as correctly represented by the visual stimulus. The interface is mostly the same on both images, but a different sub-menu entry has been highlighted. This falls under the category of *"Highlight of clickable content?"* in the decision process of visual change from Figure 5.5 on page 62, which signifies no visual change.



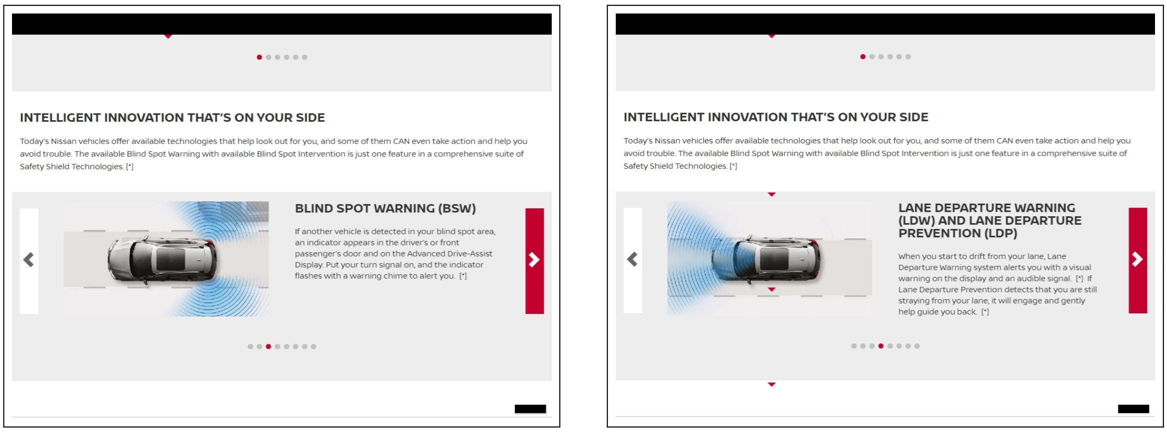
(a) Frame from the video **(b)** Corresponding region in the visual stimulus

Figure F.2: On the left is frame 174 from the video recording of the first participant on Guardian. On the right is the visual stimulus that represents that frame, cropped to the corresponding region. The frame has been marked as not correctly represented by the visual stimulus, because of the different photos behind the menu.


MAYO CLINIC
Search Mayo Clinic
Request an Appointment
Find a Doctor
Find a Job
Give Now
Log in to Patient Account
English
PATIENT CARE & HEALTH INFO
DEPARTMENTS & CENTERS
RESEARCH
EDUCATION
FOR MEDICAL PROFESSIONALS
PRODUCTS & SERVICES
GIVING TO MAYO CLINIC
Mayo Clinic's Campus in Arizona
RANKED #1
HOSPITAL IN ARIZONA.
U.S. News & World Report 2018–2019
No. 1 Hospital in Arizona: 2018-2019 U.S. News & World Report
Mayo Clinic's Arizona campus is ranked the No. 1 hospital in Arizona and is ranked No. 11 in the nation on the "Best Hospitals Honor Roll."
Becoming a patient
Mayo Clinic's campus in Arizona provides medical care for thousands of people from
Traveling to Mayo Clinic's campus in Arizona
Appointments
Request an appointment at Mayo Clinic's campus in Arizona, Florida,

(a) Frame from the video **(b)** Corresponding region in the visual stimulus

Figure F.3: On the left is frame 287 from the video recording of the first participant on MayoClinic. On the right is the visual stimulus that represents that frame, cropped to the corresponding region. The frame has been marked as not correctly represented by the visual stimulus, because of the missing carousel photo in the visual stimulus.



INTELLIGENT INNOVATION THAT'S ON YOUR SIDE

Today's Nissan vehicles offer available technologies that help look out for you, and some of them CAN even take action and help you avoid trouble. The available Blind Spot Warning with available Blind Spot Intervention is just one feature in a comprehensive suite of Safety Shield Technologies. [*]

BLIND SPOT WARNING (BSW)

If another vehicle is detected in your blind spot area, an indicator appears in the driver's or front passenger's door and on the Advanced Drive-Assist Display. Put your turn signal on, and the indicator flashes with a warning chime to alert you. [*]

INTELLIGENT INNOVATION THAT'S ON YOUR SIDE

Today's Nissan vehicles offer available technologies that help look out for you, and some of them CAN even take action and help you avoid trouble. The available Blind Spot Warning with available Blind Spot Intervention is just one feature in a comprehensive suite of Safety Shield Technologies. [*]

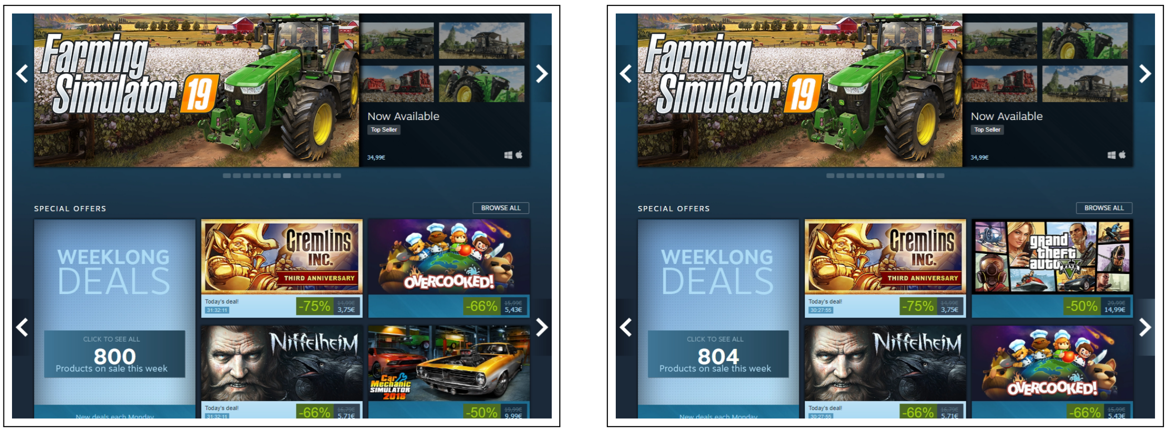
LANE DEPARTURE WARNING (LDW) AND LANE DEPARTURE PREVENTION (LDP)

When you start to drift from your lane, Lane Departure Warning system alerts you with a visual warning on the display and an audible signal. [*] If Lane Departure Prevention detects that you are still straying from your lane, it will engage and gently help guide you back. [*]

(a) Frame from the video

(b) Corresponding region in the visual stimulus

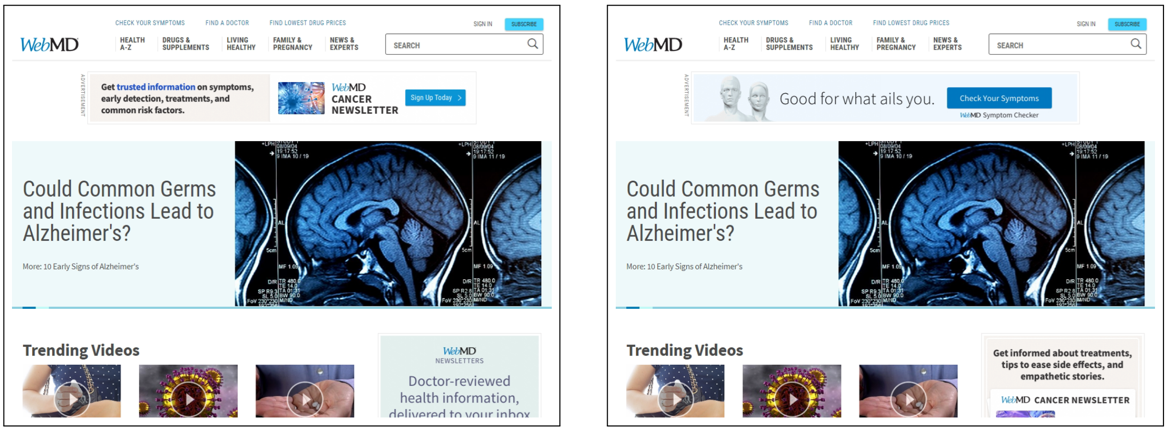
Figure F.4: On the left is frame 494 from the video recording of the first participant on Nissan. On the right is the visual stimulus that represents that frame, cropped to the corresponding region. The frame has been marked as not correctly represented by the visual stimulus, because of the different slides in the carousel.



(a) Frame from the video

(b) Corresponding region in the visual stimulus

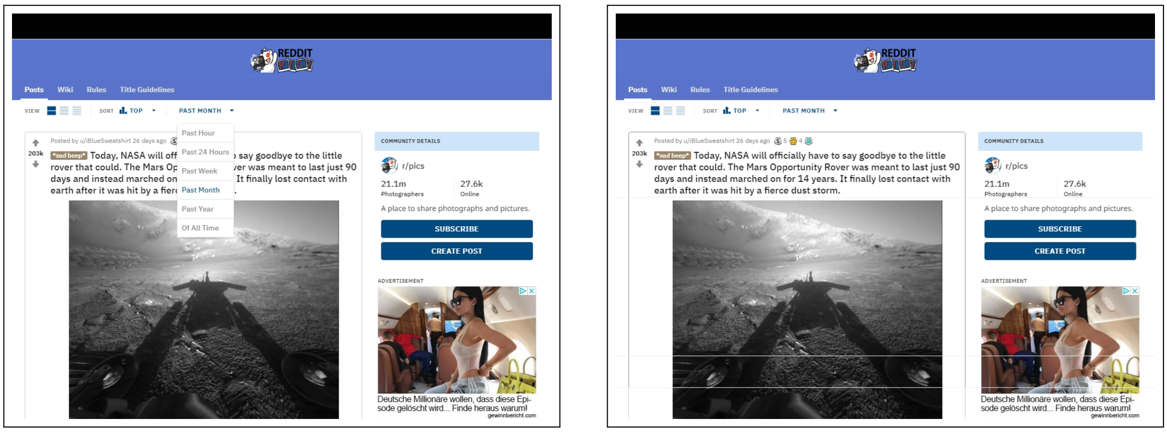
Figure F.5: On the left is frame 83 from the video recording of the first participant on Steam. On the right is the visual stimulus that represents that frame, cropped to the corresponding region. The frame has been marked as not correctly represented by the visual stimulus because the game tiles are different.


WebMD
CHECK YOUR SYMPTOMS
FIND A DOCTOR
FIND LOWEST DRUG PRICES
SIGN IN
SUBSCRIBE
HEALTH A-Z
DRUGS & SUPPLEMENTS
LIVING HEALTHY
FAMILY & PREGNANCY
NEWS & EXPERTS
SEARCH
Get trusted information on symptoms, early detection, treatments, and common risk factors.
WebMD CANCER NEWSLETTER
Sign Up Today
Good for what ails you.
Check Your Symptoms
WebMD Symptom Checker
Could Common Germs and Infections Lead to Alzheimer's?
More: 10 Early Signs of Alzheimer's
Trending Videos
WebMD NEWSLETTERS
Doctor-reviewed health information, delivered to your inbox
Get informed about treatments, tips to ease side effects, and empathetic stories.
WebMD CANCER NEWSLETTER

(a) Frame from the video

(b) Corresponding region in the visual stimulus

Figure F.6: On the left is frame 0 from the video recording of the first participant on WebMD. On the right is the visual stimulus that represents that frame, cropped to the corresponding region. The frame has been marked as not correctly represented by the visual stimulus, because of the different ad in the lower-right corner.


REDDIT
Posts
Wiki
Rules
Title Guidelines
VIEW
SORT
TOP
PAST MONTH
Past Hour
Past 24 Hours
Past Week
Past Month
Past Year
Of All Time
Posted by u/iBlueSweatshirt 26 days ago
203k
Today, NASA will officially have to say goodbye to the little rover that could. The Mars Opportunity Rover was meant to last just 90 days and instead marched on for 14 years. It finally lost contact with earth after it was hit by a fierce dust storm.
COMMUNITY DETAILS
r/pics
21.1m
Photographers
27.6k
Online
A place to share photographs and pictures.
SUBSCRIBE
CREATE POST
ADVERTISEMENT
Deutsche Millionäre wollen, dass diese Episode gelöscht wird... Finde heraus warum!
gewinnbericht.com

(a) Frame from the video

(b) Corresponding region in the visual stimulus

Figure F.7: On the left is frame 70 from the video recording of the first participant on Reddit. On the right is the visual stimulus that represents that frame, cropped to the corresponding region. The frame has been marked as not correctly represented by the visual stimulus because the menu is missing in the visual stimulus.

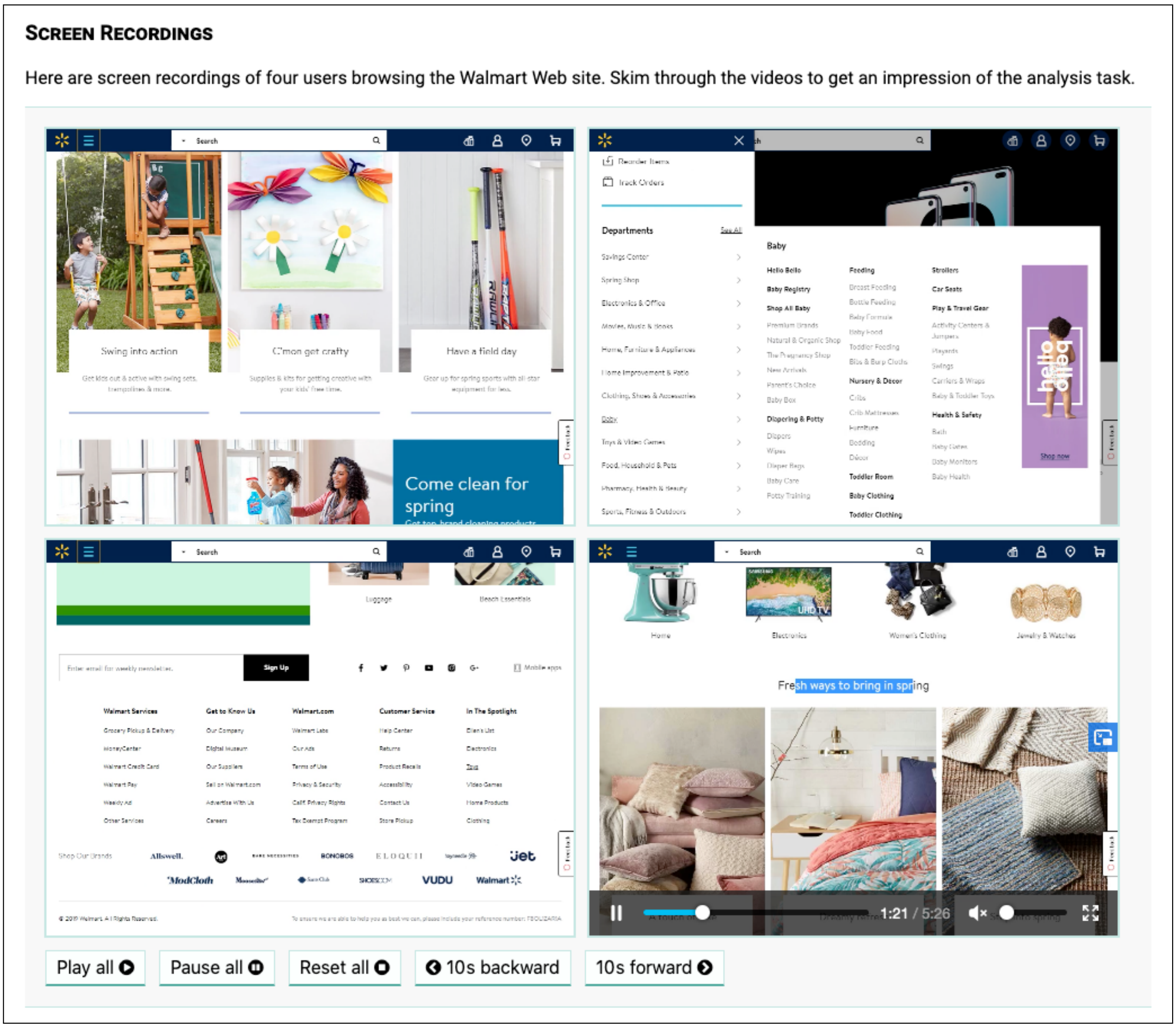
APPENDIX G

Custom Form Elements in the Expert Survey about Visual Stimuli Discovery

We have designed a survey to assess the visual stimuli discovery with usability experts.[1] The primary goal of the survey was to integrate video recordings of users browsing the dynamic Web pages and to display the corresponding visual stimuli interactively. In this way, the usability experts could get an insight into the individual analysis tasks and provide us with feedback about how they would perform those tasks and whether the visual stimuli discovery can improve their workflow.

Displaying the video recordings and the visual stimuli in the online survey have been challenging. We did not want to rely on an external hosting service for the video recordings. External hosting services may perform excessive compression of the videos, have issues in streaming performance, and introduce restricted controls over the video playback. Therefore, we have hosted the survey and all data on our Web server. This allowed us to add extended controls over the video recordings. A participant could either control an individual video or use the buttons below the videos to control all videos at once, i. e., to play, pause, reset, or skim the four videos. See Figure G.1 for a screenshot about how we presented the video recordings to the usability experts. Besides the video recordings, we have shown the corresponding visual stimuli to the participants. The height of the visual stimuli often exceeds the available screen space, why we could not use a standard picture gallery implementation. Therefore, we have implemented a gallery widget that allows a participant to choose from the generated visual stimuli, control the level of magnification, pan the visual stimulus via drag-and-drop, and toggle an overlay with mouse traces and eye gaze path on and off. See Figure G.2 for visual stimulus from the Walmart Web site without overlay, Figure G.3 for the same visual stimulus with overlay, and Figure G.4 for another visual stimulus in the same gallery widget.

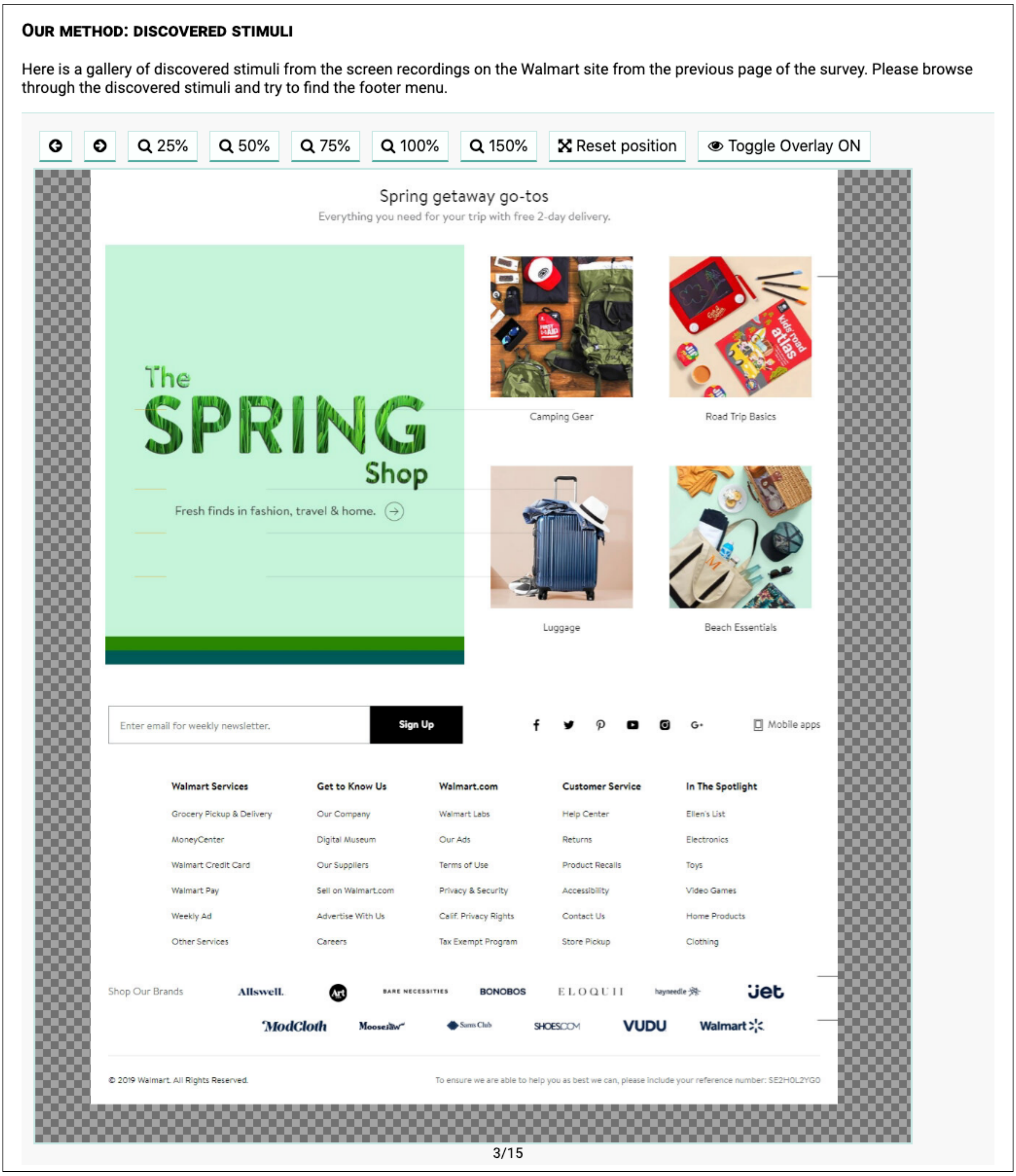
[1] `https://github.com/raphaelmenges/StupidSurvey.js`



Screen Recordings

Here are screen recordings of four users browsing the Walmart Web site. Skim through the videos to get an impression of the analysis task.

Figure G.1: We display the video recordings as the users have experienced the Web site on the screen. The screenshot shows the display of the four video recordings of the users on the Walmart Web site.



OUR METHOD: DISCOVERED STIMULI

Here is a gallery of discovered stimuli from the screen recordings on the Walmart site from the previous page of the survey. Please browse through the discovered stimuli and try to find the footer menu.

Spring getaway go-tos

Everything you need for your trip with free 2-day delivery.

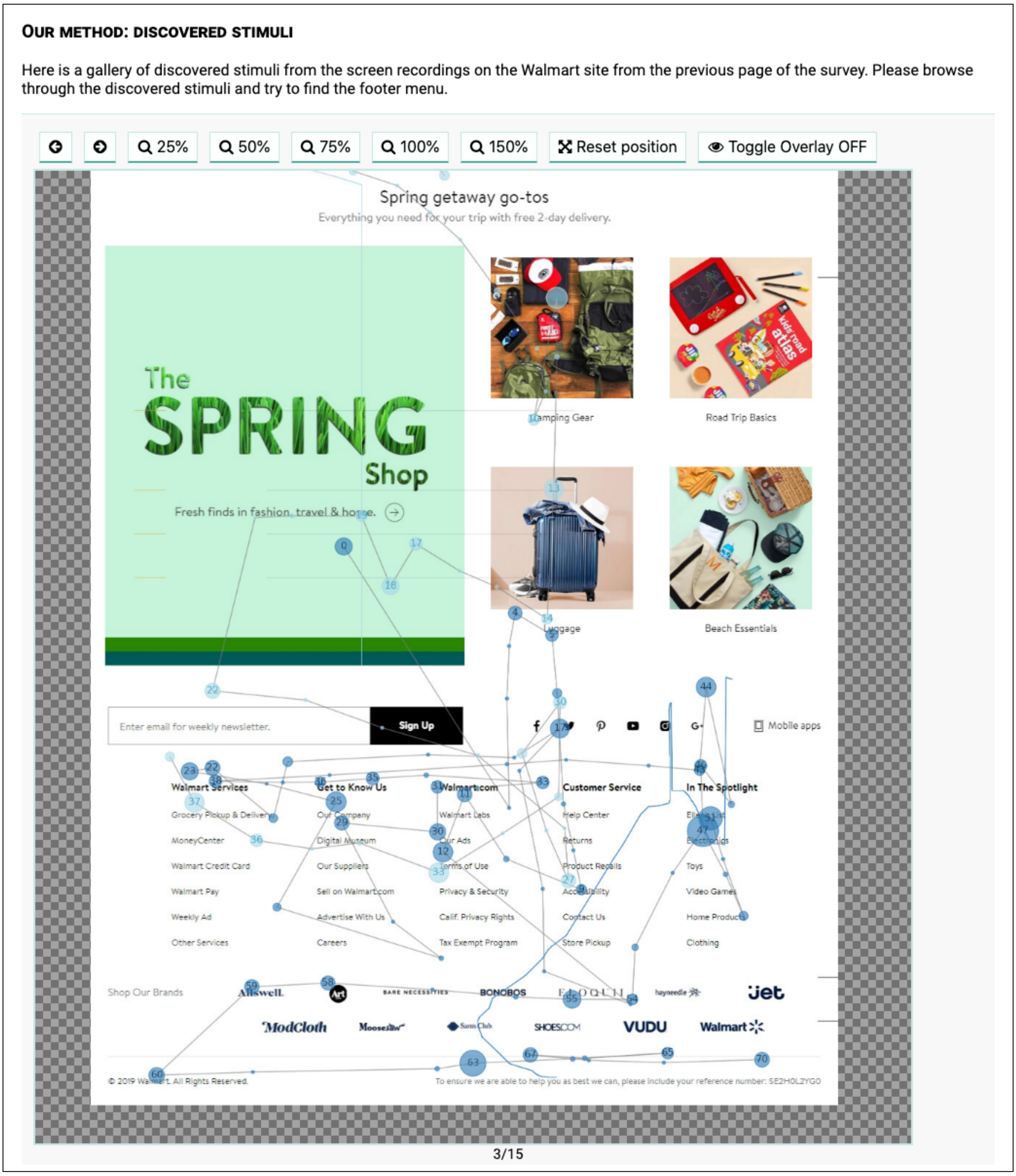
Walmart Services	Get to Know Us	Walmart.com	Customer Service	In The Spotlight
Grocery Pickup & Delivery	Our Company	Walmart Labs	Help Center	Ellen's List
MoneyCenter	Digital Museum	Our Ads	Returns	Electronics
Walmart Credit Card	Our Suppliers	Terms of Use	Product Recalls	Toys
Walmart Pay	Sell on Walmart.com	Privacy & Security	Accessibility	Video Games
Weekly Ad	Advertise With Us	Calif. Privacy Rights	Contact Us	Home Products
Other Services	Careers	Tax Exempt Program	Store Pickup	Clothing

© 2019 Walmart. All Rights Reserved.

To ensure we are able to help you as best we can, please include your reference number: SE2H0L2YG0

3/15

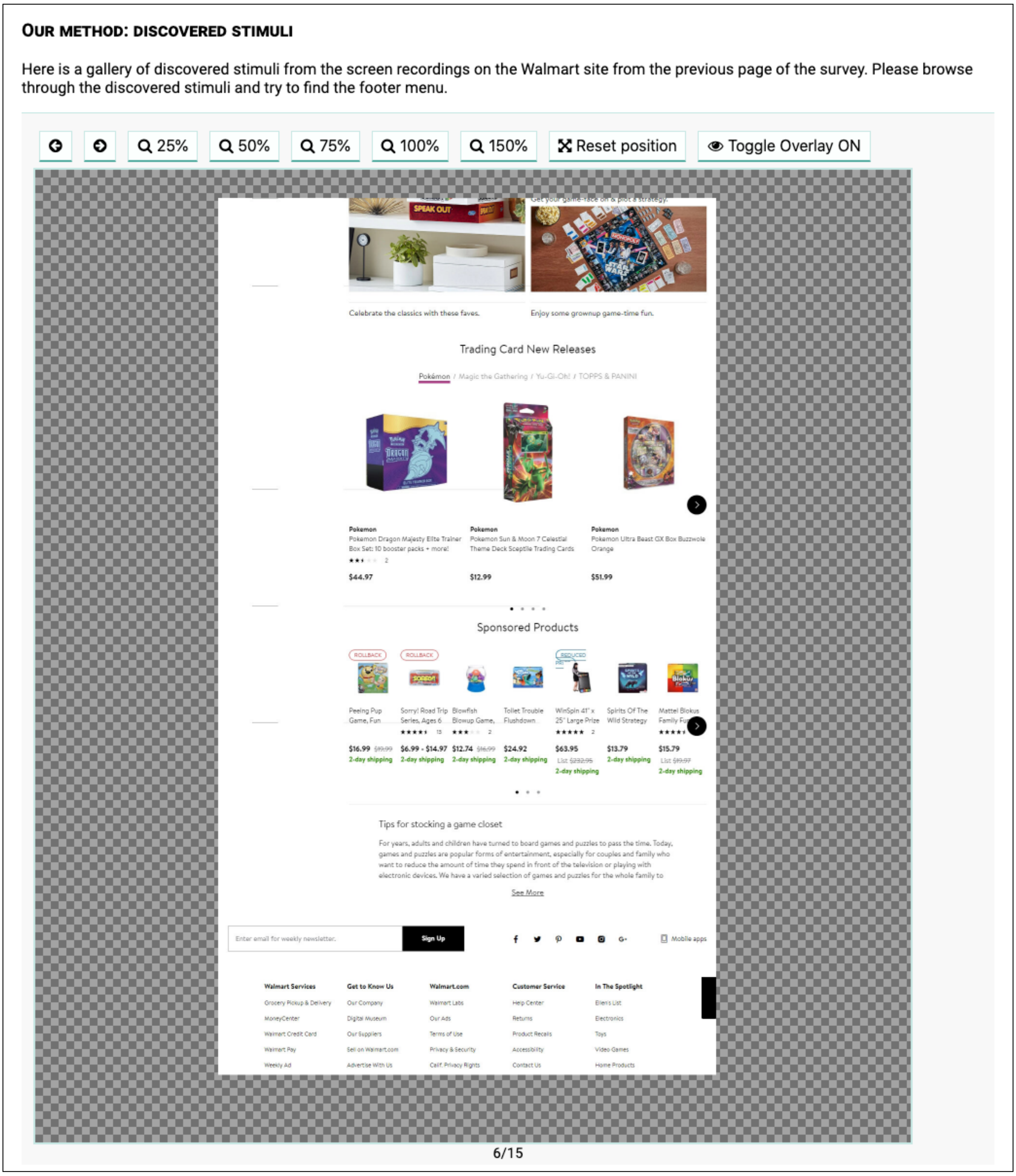
Figure G.2: We provide a gallery that displays the visual stimuli as generated by our method. The screenshot shows a visual stimulus from the Walmart Web site.



OUR METHOD: DISCOVERED STIMULI

Here is a gallery of discovered stimuli from the screen recordings on the Walmart site from the previous page of the survey. Please browse through the discovered stimuli and try to find the footer menu.

Figure G.3: A screenshot of the same visual stimulus in the gallery as in Figure G.2, but with gaze and mouse data overlaid.



OUR METHOD: DISCOVERED STIMULI

Here is a gallery of discovered stimuli from the screen recordings on the Walmart site from the previous page of the survey. Please browse through the discovered stimuli and try to find the footer menu.

25%
50%
75%
100%
150%
Reset position
Toggle Overlay ON

Get your game-face on & plot a strategy.

Celebrate the classics with these faves.

Enjoy some grownup game-time fun.

Trading Card New Releases

Pokémon / Magic the Gathering / Yu-Gi-Oh! / TOPPS & PANINI

Pokemon
Pokemon Dragon Majesty Elite Trainer Box Set: 10 booster packs + more!
★★★ 2
$44.97

Pokemon
Pokemon Sun & Moon 7 Celestial Theme Deck Sceptile Trading Cards
$12.99

Pokemon
Pokemon Ultra Beast GX Box Buzzwole Orange
$51.99

Sponsored Products

ROLLBACK
Peeing Pup Game, Fun
$16.99 ~~$19.99~~
2-day shipping

ROLLBACK
Sorry! Road Trip Series, Ages 6
★★★★★ 13
$6.99 - $14.97
2-day shipping

Blowfish Blowup Game,
★★★ 2
$12.74 ~~$16.99~~
2-day shipping

Toilet Trouble Flushdown
$24.92
2-day shipping

REDUCED PRI
WinSpin 41" x 25" Large Prize
★★★★★ 2
$63.95
List ~~$232.95~~
2-day shipping

Spirits Of The Wild Strategy
$13.79
2-day shipping

Mattel Blokus Family Fu
★★★★★
$15.79
List ~~$19.97~~
2-day shipping

Tips for stocking a game closet

For years, adults and children have turned to board games and puzzles to pass the time. Today, games and puzzles are popular forms of entertainment, especially for couples and family who want to reduce the amount of time they spend in front of the television or playing with electronic devices. We have a varied selection of games and puzzles for the whole family to

Walmart Services	Get to Know Us	Walmart.com	Customer Service	In The Spotlight
Grocery Pickup & Delivery	Our Company	Walmart Labs	Help Center	Ellen's List
MoneyCenter	Digital Museum	Our Ads	Returns	Electronics
Walmart Credit Card	Our Suppliers	Terms of Use	Product Recalls	Toys
Walmart Pay	Sell on Walmart.com	Privacy & Security	Accessibility	Video Games
Weekly Ad	Advertise With Us	Calif. Privacy Rights	Contact Us	Home Products

Figure G.4: A screenshot of another visual stimulus from the Walmart Web site in the gallery. A participant can fit a visual stimulus into the canvas of the gallery by zooming and panning the visual stimulus.

APPENDIX H

Interaction in the Web with OptiKey

The OptiKey interface consists of emulation panels with different modes of emulation, i. e., mouse and various keyboards. For the experimental setup, only the standard QWERTY keyboard (Figure H.1a) and mouse (Figure H.1b) modes were used. A user can switch between both modes with a virtual button in the emulation panel. In the following, we describe the interaction procedures required to fulfill the browsing tasks with OptiKey and Google Chrome, which were executed in the lab study at our university described in Section 8.1.

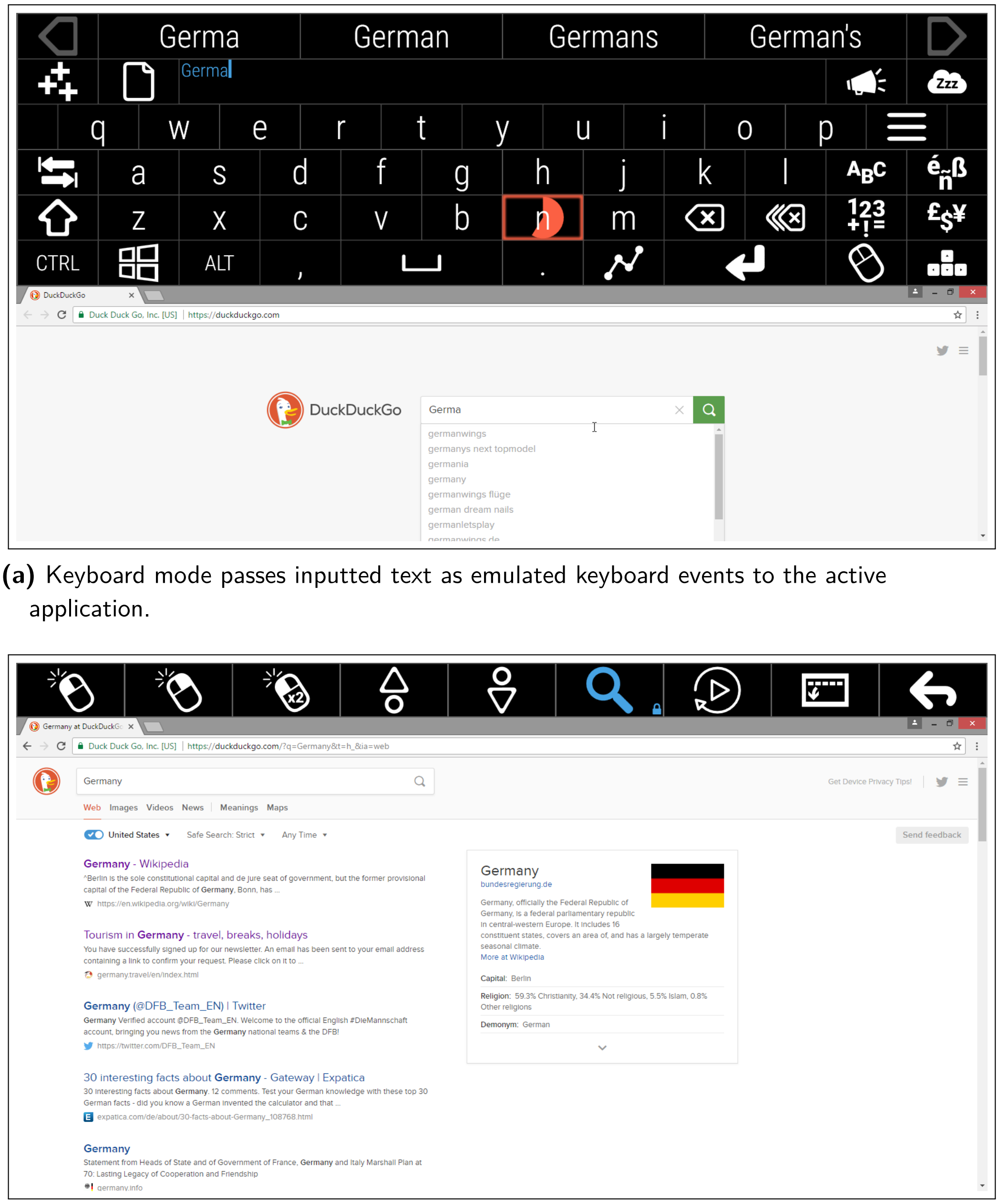
Navigation Navigation within a Web page is mostly performed via backward functionality or by hyperlinks. A user can operate these functions in Google Chrome using OptiKey via mouse emulation. In the mouse mode of the emulation panel, the “left mouse click” virtual button needs to be selected to initiate the pointing process. Then, a mouse pointer is displayed at the detected fixation upon the interface of the active application. After a certain duration of the fixation, the content beneath the gaze is magnified and presented in the center of the screen as a pop-up overlay. After another dwell time on the pop-up overlay, a click is performed at the fixation within the magnified content. The additional magnification step brings the pointing accuracy to a level that enables reliable selection of a hyperlink that is rendered in text in standard size. See Figure H.2 for the process of clicking.

Scrolling Scrolling on a desktop computer is usually performed with the scrolling wheel on a physical mouse. OptiKey adopts this paradigm and offers options to emulate the scrolling wheel in its mouse mode. To perform an upwards scroll event, the virtual button for scrolling up has to be selected. Respectively, the virtual button for scrolling down has to be selected to scroll down in the application interface. Then, the user has to fixate the coordinate in the interface of the active application where the scroll event should be spawned. After a dwell time at the fixated coordinate, the scroll emulation is performed. When a user needs to scroll on the same coordinate multiple times, another virtual button is available to repeat the last executed action.

Text Input In the keyboard mode, virtual buttons representing keys on a virtual keyboard are displayed in the interface. The QWERTY layout has been chosen for the experimental setup.

Bookmarking To add a Web page as a bookmark in Google Chrome, the user has to perform a click on a “star” symbol, which is placed in the interface of Google Chrome next to the address bar. We asked the participants to finish the task with a second click on the “Done” button in the bookmark pop-up.

For accessing the available bookmarks, a procedure of multiple mouse click events must be performed. In Google Chrome, the bookmarks are accessible through a sub-menu of the general menu. It becomes visible after clicking at the three dots, next to the star for adding the current page as a bookmark. In total, three clicks are necessary to access a saved bookmark; two clicks for the menu navigation and a third click for the selection.



(a) Keyboard mode passes inputted text as emulated keyboard events to the active application.

Germany at DuckDuckGo
Duck Duck Go, Inc. [US] | https://duckduckgo.com/?q=Germany&t=h_&ia=web
Germany
Web Images Videos News Meanings Maps
United States
Safe Search: Strict
Any Time
Get Device Privacy Tips!
Send feedback

Germany - Wikipedia
^Berlin is the sole constitutional capital and de jure seat of government, but the former provisional capital of the Federal Republic of **Germany**, Bonn, has ...
https://en.wikipedia.org/wiki/Germany

Tourism in **Germany** - travel, breaks, holidays
You have successfully signed up for our newsletter. An email has been sent to your email address containing a link to confirm your request. Please click on it to ...
germany.travel/en/index.html

Germany (@DFB_Team_EN) | Twitter
Germany Verified account @DFB_Team_EN. Welcome to the official English #DieMannschaft account, bringing you news from the **Germany** national teams & the DFB!
https://twitter.com/DFB_Team_EN

30 interesting facts about **Germany** - Gateway | Expatica
30 interesting facts about **Germany**. 12 comments. Test your German knowledge with these top 30 German facts - did you know a German invented the calculator and that ...
expatica.com/de/about/30-facts-about-Germany_108768.html

Germany
Statement from Heads of State and of Government of France, **Germany** and Italy Marshall Plan at 70: Lasting Legacy of Cooperation and Friendship
germany.info

Germany
bundesregierung.de
Germany, officially the Federal Republic of Germany, is a federal parliamentary republic in central-western Europe. It includes 16 constituent states, covers an area of, and has a largely temperate seasonal climate.
More at Wikipedia

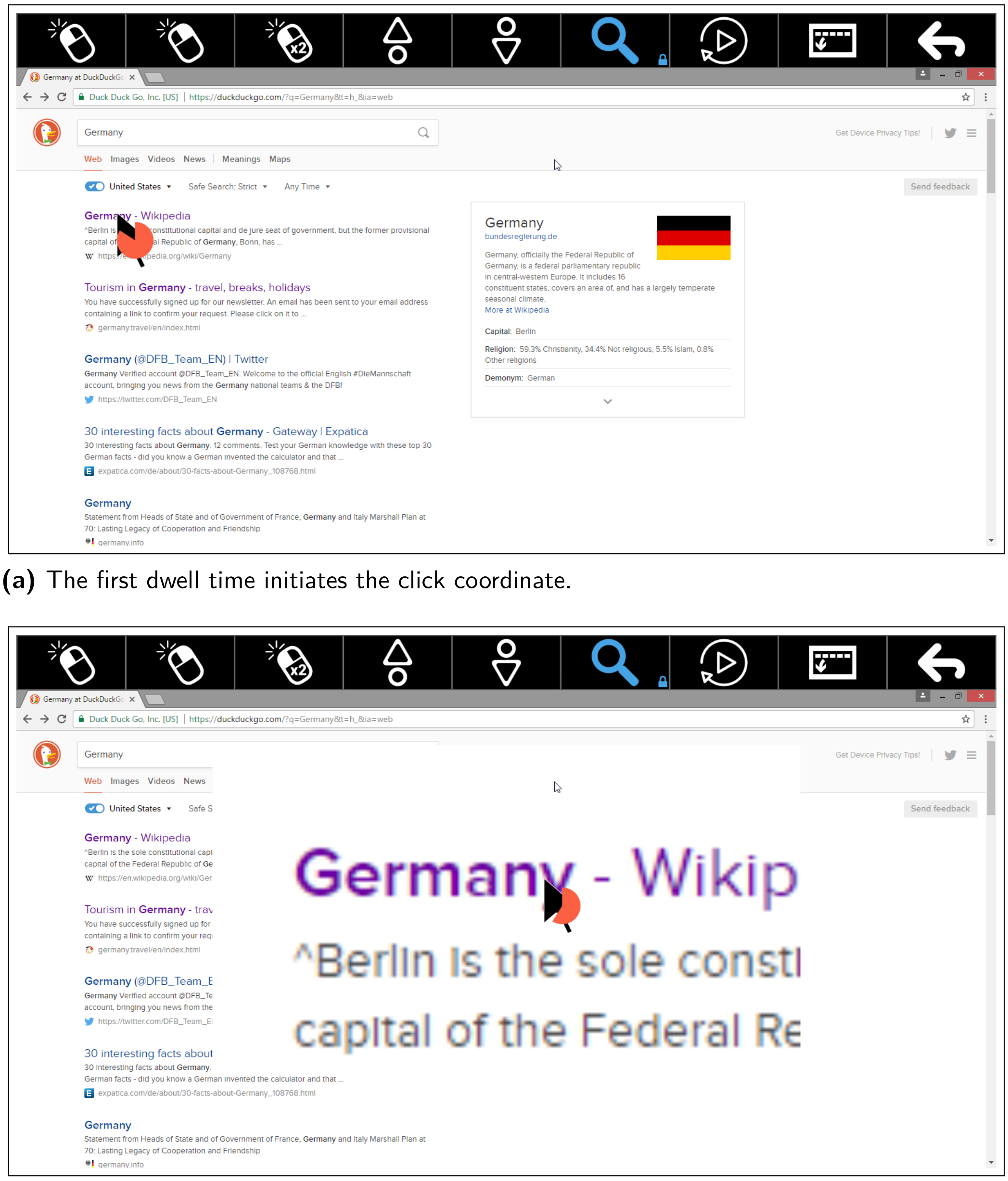
Capital: Berlin

Religion: 59.3% Christianity, 34.4% Not religious, 5.5% Islam, 0.8% Other religions

Demonym: German

(b) Mouse mode features various mouse events like left click (first virtual button from left) and scrolling (fourth and fifth virtual button from left).

Figure H.1: Two modes of OptiKey are used in the experiments. First, the keyboard mode (active after the startup of OptiKey) enables text input on a virtual keyboard. Second, the mouse mode features the emulation of common mouse events.



Germany - Wikipedia
^Berlin is the sole constitutional capital and de jure seat of government, but the former provisional capital of the Federal Republic of Germany, Bonn, has ...
https://en.wikipedia.org/wiki/Germany

Tourism in Germany - travel, breaks, holidays
You have successfully signed up for our newsletter. An email has been sent to your email address containing a link to confirm your request. Please click on it to ...
germany.travel/en/index.html

Germany (@DFB_Team_EN) | Twitter
Germany Verified account @DFB_Team_EN. Welcome to the official English #DieMannschaft account, bringing you news from the Germany national teams & the DFB!
https://twitter.com/DFB_Team_EN

30 interesting facts about Germany - Gateway | Expatica
30 interesting facts about Germany. 12 comments. Test your German knowledge with these top 30 German facts - did you know a German invented the calculator and that ...
expatica.com/de/about/30-facts-about-Germany_108768.html

Germany
Statement from Heads of State and of Government of France, Germany and Italy Marshall Plan at 70: Lasting Legacy of Cooperation and Friendship
germany.info

Germany
bundesregierung.de
Germany, officially the Federal Republic of Germany, is a federal parliamentary republic in central-western Europe. It includes 16 constituent states, covers an area of, and has a largely temperate seasonal climate.
More at Wikipedia

Capital: Berlin
Religion: 59.3% Christianity, 34.4% Not religious, 5.5% Islam, 0.8% Other religions
Demonym: German

(a) The first dwell time initiates the click coordinate.

Germany - Wikip
^Berlin is the sole consti
capital of the Federal Re

(b) Magnified screen as the second step in the mouse click emulation for more accurate pointing.

Figure H.2: After selection of the "left mouse click" virtual button on the left of the emulation panel, clicking is performed in two steps. First, a user fixates the region to click. After a dwell time, the region is magnified and displayed in the center of the screen. A second fixation with dwell time is necessary to choose the final coordinate of click. The progress of dwell time is visualized by the filling pie next to the big black arrow, which indicates the coordinate of the fixation.

APPENDIX

Tasks for the MAMEM First Phase Trials

The dictated tasks for the MAMEM first phase trials are denoted in the following. Required online accounts had been provided by the experimenter.

E-mail Task

1. User is asked to go to the tab overview
2. User is asked to add a new tab
3. User is asked to go to *“Gmail.com”* via manually typing of the URL
4. User is asked to sign into *“Gmail.com”*
5. User is asked to read a received E-Mail
6. User is asked to respond to an E-Mail by writing “hello world”
7. User is asked to send the E-Mail
8. User proceeds to task I

Photo Task

1. User is asked to go to the tab overview
2. User is asked to go to “Edit the URL”
3. User is asked to visit the bookmark overview
4. User is asked to go to the *“Picresize.com”* bookmark
5. User is asked to choose a sample picture of choice on the Web page
6. User is asked to rotate the picture 90 degrees counter-clockwise
7. User is asked to choose a special effect of choice
8. User is asked to choose “I’m Done, Resize My Picture!”
9. User proceeds to I

Social Media Task

1. User is asked to go to the tab overview
2. User is asked to add a new tab
3. User is asked to visit the bookmark overview
4. User is asked to go to the *“Twitter.com”* bookmark
5. User is asked to search for the *“MAMEM Project”*
6. User is asked to select the MAMEM project page
7. User is asked to follow the MAMEM project
8. User is asked to post a text message on MAMEM project’s page
9. User proceeds to I

Video Task

1. User is asked to go to the tab overview
2. User is asked to add a new tab
3. User is asked to visit the bookmark overview
4. User is asked to go to the *“YouTube.com”* bookmark
5. User is asked to search for *“gazetheweb mamem”*
6. User is asked to select the first video
7. User is asked to pause the video
8. User is asked to play again the video
9. User is asked to close the tab with the YouTube page
10. End of dictated tasks

Bibliography

[1] Tobii AB. *Sticky*. Jan. 2018. URL: `https://www.sticky.ai`.

[2] Kiyohiko Abe, Kosuke Owada, Shoichi Ohi, and Minoru Ohyama. “A system for Web browsing by eye-gaze input”. In: *Electronics and Communications in Japan* 91.5 (2008), pp. 11–18. URL: `http://dx.doi.org/10.1002/ecj.10110`.

[3] Sheetal K. Agarwal, Anupam Jain, Arun Kumar, and Nitendra Rajput. “The World Wide Telecom Web Browser”. In: *Proceedings of the First ACM Symposium on Computing for Development*. ACM DEV ’10. London, United Kingdom: ACM, 2010, 4:1–4:9. URL: `http://doi.acm.org/10.1145/1926180.1926185`.

[4] Deepak Ahya and Daniel Baudino. *Method to enhance user interface and target applications based on context awareness*. US Patent App. 10/853,947. May 2006.

[5] Pierre A. Akiki, Arosha K. Bandara, and Yijun Yu. “Engineering Adaptive Model-Driven User Interfaces”. In: *IEEE Transactions on Software Engineering* 42.12 (Dec. 2016), pp. 1118–1147.

[6] M. Elgin Akpınar and Yeliz Yesılada. “Vision Based Page Segmentation Algorithm: Extended and Perceived Success”. In: *Current Trends in Web Engineering*. Ed. by Quan Z. Sheng and Jesper Kjeldskov. Cham: Springer International Publishing, 2013, pp. 238–252.

[7] Allied Analytics LLP. *Eye Tracking Market by Type, Application, and Industry Vertical - Global Opportunity Analysis and Industry Forecast, 2018-2024*. 2018. URL: `https://www.researchandmarkets.com/reports/4580638/eye-tracking-market-by-type-application-and`.

[8] Michael Ashmore, Andrew T. Duchowski, and Garth Shoemaker. “Efficient Eye Pointing with a Fisheye Lens”. In: *Proceedings of Graphics Interface 2005*. GI ’05. Victoria, British Columbia: Canadian Human-Computer Communications Society, 2005, pp. 203–210. URL: `http://dl.acm.org/citation.cfm?id=1089508.1089542`.

[9] Rossen Atanassov and Arron Eicholz. *CSS Positioned Layout Module Level 3*. W3C Working Draft. http://www.w3.org/TR/2016/WD-css-position-3-20160517. W3C, May 2016.

[10] The Chromium Authors. *The Chromium Projects*. 2019. URL: `https://www.chromium.org`.

[11] The FFmpeg Authors. *FFmpeg*. 2019. URL: `https://ffmpeg.org`.

[12] Laura Ball, Amy Nordness, Susan Fager, Katie Kersch, Brianae Mohr, Gary Pattee, and David Beukelman. “Eye-Gaze Access to AAC Technology for People with Amyotrophic Lateral Sclerosis”. In: *Journal of Medical Speech-Language Pathology* 18 (Sept. 2010), pp. 11–23.

[13] Nikola Banovic, Tovi Grossman, Justin Matejka, and George Fitzmaurice. “Waken: Reverse Engineering Usage Information and Interface Structure from Software Videos”. In: *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*. UIST ’12. Cambridge, Massachusetts, USA: ACM, 2012, pp. 83–92. URL: `http://doi.acm.org/10.1145/2380116.2380129`.

[14] Ana Barreto. “Do users look at banner ads on Facebook?” In: *Journal of Research in Interactive Marketing* 7.2 (May 2013). URL: `http://dx.doi.org/10.1108/JRIM-Mar-2012-0013`.

[15] Michael Barz and Daniel Sonntag. “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: Adjunct*. UbiComp ’16. Heidelberg, Germany: Association for Computing Machinery, 2016, pp. 253–256. URL: `https://doi.org/10.1145/2968219.2971389`.

[16] Richard Bates and Howell Istance. “Zooming Interfaces!: Enhancing the Performance of Eye Controlled Pointing Devices”. In: *Proceedings of the Fifth International ACM Conference on Assistive Technologies*. Assets ’02. Edinburgh, Scotland: ACM, 2002, pp. 119–126. URL: `http://doi.acm.org/10.1145/638249.638272`.

[17] Tanya Beelders and Pieter Blignaut. “The Usability of Speech and Eye Gaze as a Multimodal Interface for a Word Processor”. In: *Speech Technologies*. Ed. by Ivo Ipsic. Rijeka: IntechOpen, 2011. Chap. 19. URL: `https://doi.org/10.5772/16604`.

[18] Wolfgang Beinhauer. “A Widget Library for Gaze-based Interaction Elements”. In: *Proceedings of the 2006 Symposium on Eye Tracking Research & Applications*. ETRA ’06. San Diego, California: ACM, 2006, pp. 53–53. URL: `http://doi.acm.org/10.1145/1117309.1117338`.

[19] Michael Bensch, Ahmed A. Karim, Jürgen Mellinger, Thilo Hinterberger, Michael Tangermann, Martin Bogdan, Wolfgang Rosenstiel, and Niels Birbaumer. “Nessi: an EEG-controlled web browser for severely paralyzed patients”. In: *Computational intelligence and neuroscience* 2007 (2007).

[20] David Beymer and Daniel M. Russell. “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. Portland, OR, USA: ACM, 2005, pp. 1913–1916. URL: `http://doi.acm.org/10.1145/1056808.1057055`.

[21] Ralf Biedert, Georg Buscher, Sven Schwarz, Manuel Möller, Andreas Dengel, and Thomas Lottermann. “The Text 2.0 Framework: Writing Web-based Gaze-controlled Realtime Applications Quickly and Easily”. In: *Proceedings of the 2010 Workshop on Eye Gaze in Intelligent Human Machine Interaction*. EGIHMI ’10. Hong Kong, China: ACM, 2010, pp. 114–117. URL: `http://doi.acm.org/10.1145/2002333.2002351`.

[22] Renaud Blanch, Yves Guiard, and Michel Beaudouin-Lafon. "Semantic Pointing: Improving Target Acquisition with Control-display Ratio Adaptation". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '04. Vienna, Austria: ACM, 2004, pp. 519–526. URL: `http://doi.acm.org/10.1145/985692.985758`.

[23] Tanja Blascheck, Kuno Kurzhals, Michael Raschke, Michael Burch, Daniel Weiskopf, and Thomes Ertl. "Visualization of Eye Tracking Data: A Taxonomy and Survey". In: *Computer Graphics Forum* 36.8 (2017), pp. 260–284. URL: `http://dx.doi.org/10.1111/cgf.13079`.

[24] Richard A. Bolt. "Gaze-orchestrated Dynamic Windows". In: *Proceedings of the 8th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '81. Dallas, Texas, USA: ACM, 1981, pp. 109–119. URL: `http://doi.acm.org/10.1145/800224.806796`.

[25] Gary Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[26] John Brooke. "SUS: A Retrospective". In: *Journal of Usability Studies* 8.2 (Feb. 2013), pp. 29–40. URL: `http://dl.acm.org/citation.cfm?id=2817912.2817913`.

[27] Brian Burg, Andrew J. Ko, and Michael D. Ernst. "Explaining Visual Changes in Web Interfaces". In: *Proceedings of the 28th Annual ACM Symposium on User Interface Software Technology*. UIST '15. Charlotte, NC, USA: ACM, 2015, pp. 259–268. URL: `http://doi.acm.org/10.1145/2807442.2807473`.

[28] Georg Buscher, Edward Cutrell, and Meredith Ringel Morris. "What Do You See when You'Re Surfing?: Using Eye Tracking to Predict Salient Regions of Web Pages". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '09. Boston, MA, USA: ACM, 2009, pp. 21–30. URL: `http://doi.acm.org/10.1145/1518701.1518705`.

[29] Zoya Bylinskii, Michelle A. Borkin, Nam W. Kim, Hanspeter Pfister, and Aude Oliva. "Eye Fixation Metrics for Large Scale Evaluation and Comparison of Information Visualizations". In: *Eye Tracking and Visualization: Foundations, Techniques, and Applications. ETVIS 2015*. Ed. by Michael Burch, Lewis Chuang, Brian Fisher, Albrecht Schmidt, and Daniel Weiskopf. Cham: Springer International Publishing, 2017, pp. 235–255. URL: `https://doi.org/10.1007/978-3-319-47024-5%5C_14`.

[30] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. "Extracting Content Structure for Web Pages Based on Visual Representation". In: *Proceedings of the 5th Asia-Pacific Web Conference on Web Technologies and Applications*. APWeb'03. Xian, China: Springer-Verlag, 2003, pp. 406–417.

[31] George Candea, Mauricio Delgado, Michael Chen, and Armando Fox. "Automatic Failure-Path Inference: A Generic Introspection Technique for Internet Applications". In: *Proceedings of the The Third IEEE Workshop on Internet Applications*. WIAPP '03. Washington, DC, USA: IEEE

Computer Society, 2003, pp. 132–141. URL: http://dl.acm.org/citation.cfm?id=832311.837386.

[32] John F. Canny. “A Computational Approach to Edge Detection”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 8.6 (June 1986), pp. 679–698. URL: https://doi.org/10.1109/TPAMI.1986.4767851.

[33] Ellis Carolyn. “Sociological Introspection and Emotional Experience”. In: *Symbolic Interaction* 14.1 (1991), pp. 23–50. URL: https://onlinelibrary.wiley.com/doi/abs/10.1525/si.1991.14.1.23.

[34] Scott Carter, Amy Hurst, Jennifer Mankoff, and Jack Li. “Dynamically Adapting GUIs to Diverse Input Devices”. In: *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility*. Assets ’06. Portland, Oregon, USA: ACM, 2006, pp. 63–70. URL: http://doi.acm.org/10.1145/1168987.1169000.

[35] Chia-Hui Chang, Mohammed Kayed, Moheb R. Girgis, and Khaled F. Shaalan. “A survey of web information extraction systems”. In: *IEEE transactions on knowledge and data engineering* 18.10 (2006), pp. 1411–1428.

[36] Zhaokang Chen and Bertram E. Shi. “Using Variable Dwell Time to Accelerate Gaze-Based Web Browsing with Two-Step Selection”. In: *International Journal of Human-Computer Interaction* 35.3 (2019), pp. 240–255. URL: https://doi.org/10.1080/10447318.2018.1452351.

[37] Shauvik R. Choudhary, Husayn Versee, and Alessandro Orso. “WEBDIFF: Automated identification of cross-browser issues in web applications”. In: *2010 IEEE International Conference on Software Maintenance*. Sept. 2010, pp. 1–10.

[38] Jacob Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, 1988.

[39] The Qt Company. *Cross-platform software development for embedded & desktop*. 2019. URL: https://www.qt.io.

[40] Albert M. Cook and Janice M. Polgar. *Assistive Technologies-E-Book: Principles and Practice*. Elsevier Health Sciences, 2014.

[41] CoolTool. *CoolTool*. Jan. 2018. URL: https://cooltool.com.

[42] Michael Cormier, Karyn Moffatt, Robin Cohen, and Richard Mann. “Purely Vision-based Segmentation of Web Pages for Assistive Technology”. In: *Comput. Vis. Image Underst.* 148.C (July 2016), pp. 46–66. URL: https://doi.org/10.1016/j.cviu.2016.02.007.

[43] Edward Cutrell and Zhiwei Guan. “What Are You Looking for?: An Eye-tracking Study of Information Usage in Web Search”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’07. San Jose, California, USA: ACM, 2007, pp. 407–416. URL: http://doi.acm.org/10.1145/1240624.1240690.

[44] Daniel K. Davies, Steven E. Stock, and Michael L. Wehmeyer. “Enhancing Independent Internet Access for Individuals with Mental Retardation through Use of a Specialized Web Browser: A Pilot Study”. In: *Education and Training in Mental Retardation and Developmental Disabilities* 36.1 (2001), pp. 107–113. URL: `http://www.jstor.org/stable/24481620`.

[45] Alexander De Luca, Martin Denzel, and Heinrich Hussmann. “Look into My Eyes!: Can You Guess My Password?” In: *Proceedings of the 5th Symposium on Usable Privacy and Security*. SOUPS ’09. Mountain View, California, USA: ACM, 2009, 7:1–7:12. URL: `http://doi.acm.org/10.1145/1572532.1572542`.

[46] Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. “Rico: A Mobile App Dataset for Building Data-Driven Design Applications”. In: *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. UIST ’17. Quebec City, QC, Canada: ACM, 2017, pp. 845–854. URL: `http://doi.acm.org/10.1145/3126594.3126651`.

[47] Biplab Deka, Zifeng Huang, Chad Franzen, Jeffrey Nichols, Yang Li, and Ranjitha Kumar. “ZIPT: Zero-Integration Performance Testing of Mobile App Designs”. In: *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. UIST ’17. Québec City, QC, Canada: Association for Computing Machinery, 2017, pp. 727–736. URL: `https://doi.org/10.1145/3126594.3126647`.

[48] Manfred Del Fabro and Laszlo Böszörmenyi. “State-of-the-art and future challenges in video scene detection: a survey”. In: *Multimedia Systems* 19.5 (Oct. 2013), pp. 427–454. URL: `https://doi.org/10.1007/s00530-013-0306-4`.

[49] Antonio Diaz-Tula and Carlos H. Morimoto. “AugKey: Increasing Foveal Throughput in Eye Typing with Augmented Keys”. In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI ’16. Santa Clara, California, USA: ACM, 2016, pp. 3533–3544. URL: `http://doi.acm.org/10.1145/2858036.2858517`.

[50] Morgan Dixon and James Fogarty. “Prefab: Implementing Advanced Behaviors Using Pixel-based Reverse Engineering of Interface Structure”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’10. Atlanta, Georgia, USA: ACM, 2010, pp. 1525–1534. URL: `http://doi.acm.org/10.1145/1753326.1753554`.

[51] Morgan Dixon, James Fogarty, and Jacob Wobbrock. “A General-purpose Target-aware Pointing Enhancement Using Pixel-level Analysis of Graphical Interfaces”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’12. Austin, Texas, USA: ACM, 2012, pp. 3167–3176. URL: `http://doi.acm.org/10.1145/2207676.2208734`.

[52] Heiko Drewes. “Eye Gaze Tracking for Human Computer Interaction”. In: (Mar. 2010). URL: `http://nbn-resolving.de/urn:nbn:de:bvb:19-115914`.

[53] Heiko Drewes and Albrecht Schmidt. "The MAGIC Touch: Combining MAGIC-Pointing with a Touch-Sensitive Mouse". In: *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part II*. INTERACT '09. Uppsala, Sweden: Springer-Verlag, 2009, pp. 415–428. URL: `https://doi.org/10.1007/978-3-642-03658-3%5C_46`.

[54] Andrew T. Duchowski. "A breadth-first survey of eye-tracking applications". In: *Behavior Research Methods, Instruments, & Computers* 34.4 (Nov. 2002), pp. 455–470. URL: `https://doi.org/10.3758/BF03195475`.

[55] Andrew T. Duchowski. *Eye Tracking Methodology: Theory and Practice*. Berlin, Heidelberg: Springer-Verlag, 2007.

[56] Tobii Dynavox. *Photograph of computer system with Tobii eye tracker and running Tobii Windows Control software*. 2017. URL: `http://www.tobiidynavox.de/wp-content/uploads/2016/06/TobiiDynavox%5C_EyeMobileMini%5C_front%5C_-1030x687.png`.

[57] Claudia Ehmke and Stephanie Wilson. "Identifying Web Usability Problems from Eye-tracking Data". In: *Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI...But Not As We Know It - Volume 1*. BCS-HCI '07. University of Lancaster, United Kingdom: British Computer Society, 2007, pp. 119–128. URL: `http://dl.acm.org/citation.cfm?id=1531294.1531311`.

[58] Sukru Eraslan, Yeliz Yesilada, and Simon Harper. "Crowdsourcing a Corpus of Eye Tracking Data on Web Pages: A Methodology". In: *Proceedings of Measuring Behavior 2018*. Manchester, UK, June 2018, p. 7.

[59] Sukru Eraslan, Yeliz Yesilada, and Simon Harper. "Eye Tracking Scanpath Analysis on Web Pages: How Many Users?" In: *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*. ETRA '16. Charleston, South Carolina: ACM, 2016, pp. 103–110. URL: `http://doi.acm.org/10.1145/2857491.2857519`.

[60] Sukru Eraslan, Yeliz Yesilada, and Simon Harper. "Eye tracking scanpath analysis techniques on web pages: A survey, evaluation and comparison". In: *Journal of Eye Movement Research* 9.1 (2015). URL: `http://dx.doi.org/10.16910/jemr.9.1.2`.

[61] Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. "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. Charlotte, NC, USA: ACM, 2015, pp. 457–466. URL: `http://doi.acm.org/10.1145/2807442.2807499`.

[62] EYEVIDO GmbH. *EYEVIDO Lab*. 2020. URL: `https://eyevido.de/en/cloud-eye-tracking/software-eyevido-lab`.

[63] Gunnar Farnebäck. "Two-frame Motion Estimation Based on Polynomial Expansion". In: *Proceedings of the 13th Scandinavian Conference on Image Analysis*. SCIA'03. Halmstad, Sweden: Springer-Verlag, 2003, pp. 363–370. URL: `http://dl.acm.org/citation.cfm?id=1763974.1764031`.

[64] Anna Maria Feit, Shane Williams, Arturo Toledo, Ann Paradiso, Harish Kulkarni, Shaun Kane, and Meredith R. Morris. "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*. CHI '17. Denver, Colorado, USA: ACM, 2017, pp. 1118–1130. URL: `http://doi.acm.org/10.1145/3025453.3025599`.

[65] Leah Findlater, Alex Jansen, Kristen Shinohara, Morgan Dixon, Peter Kamb, Joshua Rakita, and Jacob O. Wobbrock. "Enhanced Area Cursors: Reducing Fine Pointing Demands for People with Motor Impairments". In: *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*. UIST '10. New York, New York, USA: ACM, 2010, pp. 153–162. URL: `http://doi.acm.org/10.1145/1866029.1866055`.

[66] Sofia Fountoukidou, Jaap Ham, Uwe Matzat, and Cees Midden. "Using an Artificial Agent as a Behavior Model to Promote Assistive Technology Acceptance". In: *Persuasive Technology*. Ed. by Jaap Ham, Evangelos Karapanos, Plinio P. Morita, and Catherine M. Burns. Cham: Springer International Publishing, 2018, pp. 285–296. URL: `https://doi.org/10.1007/978-3-319-78978-1%5C_24`.

[67] Krzysztof Z. Gajos. *Automatically generating personalized user interfaces*. University of Washington, 2008.

[68] Eyezag GbR. *Eyezag*. Jan. 2018. URL: `https://eyezag.de`.

[69] James Gips and Peter Olivieri. "EagleEyes: An Eye Control System for Persons with Disabilities". In: *Proc. 11 th Int. Conf on Technology and Persons with Disabilities*. 1996, p. 13.

[70] Visual Interaction GmbH. *myGaze Power catalogue*. 2017. URL: `http://www.mygaze.com/fileadmin/download/mygaze%5C_power/myGaze%5C_Power%5C_catalogue.pdf`.

[71] Aryeh Gregor, Ms2ger, Alex Russell, Robin Berjon, and Anne van Kesteren. *W3C DOM4*. W3C Recommendation. http://www.w3.org/TR/2015/REC-dom-20151119/. W3C, Nov. 2015.

[72] Tovi Grossman and Ravin Balakrishnan. "The Bubble Cursor: Enhancing Target Acquisition by Dynamic Resizing of the Cursor's Activation Area". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '05. Portland, Oregon, USA: ACM, 2005, pp. 281–290. URL: `http://doi.acm.org/10.1145/1054972.1055012`.

[73] Human Performance Research Group. *Nasa Task Load Index (TLX): Paper and Pencil Package*. `http://humansystems.arc.nasa.gov/groups/tlx/downloads/TLX_pappen_manual.pdf`. Accessed: 2nd May 2016. 1988.

[74] Vicki L. Hanson, Jonathan P. Brezin, Susan Crayne, Simeon Keates, Rick Kjeldsen, John T. Richards, Calvin Swart, and Shari Trewin. “Improving Web Accessibility Through an Enhanced Open-source Browser”. In: *IBM Syst. J.* 44.3 (Aug. 2005), pp. 573–588. URL: `http://dx.doi.org/10.1147/sj.443.0573`.

[75] Katarzyna Harezlak, Pawel Kasprowski, and Mateusz Stasch. “Towards Accurate Eye Tracker Calibration - Methods and Procedures”. In: *Procedia Computer Science* 35 (2014). Knowledge-Based and Intelligent Information & Engineering Systems 18th Annual Conference, KES-2014 Gdynia, Poland, September 2014 Proceedings, pp. 1073–1081. URL: `http://www.sciencedirect.com/science/article/pii/S1877050914011594`.

[76] Simon Harper and Yeliz Yesilada. *Web accessibility: a foundation for research*. Springer Science & Business Media, 2008.

[77] Alexander G. Haupmann and Michael J. Witbrock. “Story Segmentation and Detection of Commercials in Broadcast News Video”. In: *Proceedings of the Advances in Digital Libraries Conference*. ADL ’98. Washington, DC, USA: IEEE Computer Society, 1998, pp. 168–. URL: `http://dl.acm.org/citation.cfm?id=582987.785930`.

[78] Ramin Hedeshy, Chandan Kumar, **Raphael Menges**, and Steffen Staab. “GIUPlayer: A Gaze Immersive YouTube Player Enabling Eye Control and Attention Analysis”. In: *ACM Symposium on Eye Tracking Research and Applications*. ETRA ’20 Adjunct. Stuttgart, Germany: Association for Computing Machinery, 2020. URL: `https://doi.org/10.1145/3379157.3391984`.

[80] Henna Heikkilä. “EyeSketch: A Drawing Application for Gaze Control”. In: *Proceedings of the 2013 Conference on Eye Tracking South Africa*. ETSA ’13. Cape Town, South Africa: ACM, 2013, pp. 71–74. URL: `http://doi.acm.org/10.1145/2509315.2509332`.

[81] Daniel Hienert, Dagmar Kern, Matthew Mitsui, Chirag Shah, and Nicholas J. Belkin. “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. Glasgow, Scotland UK: Association for Computing Machinery, 2019, pp. 73–81. URL: `https://doi.org/10.1145/3295750.3298921`.

[82] Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost Van de Weijer. *Eye tracking: A comprehensive guide to methods and measures*. OUP Oxford, 2011.

[83] Anthony J. Hornof and Anna Cavender. “EyeDraw: Enabling Children with Severe Motor Impairments to Draw with Their Eyes”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’05. Portland, Oregon, USA: ACM, 2005, pp. 161–170. URL: `http://doi.acm.org/10.1145/1054972.1054995`.

[84] Eric Horvitz. “Principles of Mixed-initiative User Interfaces”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’99. Pittsburgh, Pennsylvania, USA: ACM, 1999, pp. 159–166. URL: `http://doi.acm.org/10.1145/302979.303030`.

[85] ISO/TC 159/SC 4 Ergonomics of human-system interaction. *ISO 9241-210:2010*. Tech. rep. Switzerland: International Organization for Standardization, Mar. 2010.

[86] Poika Isokoski, Markus Joos, Oleg Spakov, and Benoît Martin. “Gaze Controlled Games”. In: *Univers. Access Inf. Soc.* 8.4 (Oct. 2009), pp. 323–337. URL: `http://dx.doi.org/10.1007/s10209-009-0146-3`.

[87] Howell Istance, Richard Bates, Aulikki Hyrskykari, and Stephen Vickers. “Snap Clutch, a Moded Approach to Solving the Midas Touch Problem”. In: *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*. ETRA ’08. Savannah, Georgia: ACM, 2008, pp. 221–228. URL: `http://doi.acm.org/10.1145/1344471.1344523`.

[88] Rob Jacob and Sophie Stellmach. “What You Look at is What You Get: Gaze-based User Interfaces”. In: *interactions* 23.5 (Aug. 2016), pp. 62–65. URL: `http://doi.acm.org/10.1145/2978577`.

[89] Robert J. K. Jacob. “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. Seattle, Washington, USA: ACM, 1990, pp. 11–18. URL: `http://doi.acm.org/10.1145/97243.97246`.

[90] Søren S. Jensen and Tina Øvad. “Optimizing web-accessibility for deaf people and the hearing impaired utilizing a sign language dictionary embedded in a browser”. In: *Cognition, Technology & Work* 18.4 (Nov. 2016), pp. 717–731. URL: `https://doi.org/10.1007/s10111-016-0385-z`.

[91] George H. Joblove and Donald Greenberg. “Color Spaces for Computer Graphics”. In: *SIGGRAPH Comput. Graph.* 12.3 (Aug. 1978), pp. 20–25. URL: `https://doi.org/10.1145/965139.807362`.

[92] Matt Jones, Gary Marsden, Norliza Mohd-Nasir, Kevin Boone, and George Buchanan. “Improving Web Interaction on Small Displays”. In: *Comput. Netw.* 31.11–16 (May 1999), pp. 1129–1137. URL: `https://doi.org/10.1016/S1389-1286(99)00013-4`.

[93] Fotis Kalaganis, Elisavet Chatzilari, Spiros Nikolopoulos, Yiannis Kompatsiaris, and Nikos Laskaris. “An error-aware gaze-based keyboard by means of a hybrid BCI system”. In: *Scientific Reports* 8.1 (2018). URL: `https://doi.org/10.1038/s41598-018-31425-2`.

[94] Ahmed A. Karim, Thilo Hinterberger, Jürgen Richter, Jürgen Mellinger, Nicola Neumann, Herta Flor, Andrea Kübler, and Niels Birbaumer. “Neural internet: Web surfing with brain potentials for the completely paralyzed”. In: *Neurorehabilitation and Neural Repair* 20.4 (2006), pp. 508–515.

[95] Melanie Kellar, Carolyn Watters, and Michael Shepherd. “The Impact of Task on the Usage of Web Browser Navigation Mechanisms”. In: *Proceedings of Graphics Interface 2006*. GI ’06. Quebec, Canada: Canadian Information Processing Society, 2006, pp. 235–242. URL: `http://dl.acm.org/citation.cfm?id=1143079.1143118`.

[96] Rodrigo M. Kishi, Tiago H. Trojahn, and Rudinei Goularte. “An Evaluation of Readily Usable Automatic Video Shot Segmentation Techniques”. In: *Proceedings of the 22nd Brazilian Symposium on Multimedia and the Web*. Webmedia ’16. Teresina, Piaui; State, Brazil: ACM, 2016, pp. 199–202. URL: `http://doi.acm.org/10.1145/2976796.2988174`.

[97] Hiromi Kobayashi and Shiro Kohshima. “Unique morphology of the human eye and its adaptive meaning: Comparative studies on external morphology of the primate eye”. In: *Journal of human evolution* 40 (June 2001), pp. 419–35.

[98] Kurt Koffka. “Introspection and the method of psychology”. In: *British Journal of Psychology* 15.2 (1924), pp. 149–161. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1111/j.2044-8295.1924.tb00170.x`.

[99] Chandan Kumar, Daniyal Akbari, **Raphael Menges**, Scott MacKenzie, and Steffen Staab. “TouchGazePath: Multimodal Interaction with Touch and Gaze Path for Secure Yet Efficient PIN Entry”. In: *2019 International Conference on Multimodal Interaction*. ICMI ’19. Suzhou, China: ACM, 2019, pp. 329–338. URL: `http://doi.acm.org/10.1145/3340555.3353734`.

[100] Chandan Kumar, Ramin Hedeshy, I. Scott MacKenzie, and Steffen Staab. “TAGSwipe: Touch Assisted Gaze Swipe for Text Entry”. In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI ’20. Honolulu, HI, USA: Association for Computing Machinery, 2020, pp. 1–12. URL: `https://doi.org/10.1145/3313831.3376317`.

[101] Chandan Kumar, **Raphael Menges**, Korok Sengupta, and Steffen Staab. “Eye tracking for Interaction: Evaluation Methods”. In: *Signal Processing to Drive Human-Computer Interaction: EEG and eye-controlled interfaces*. Ed. by Spiros Nikolopoulos, Chandan Kumar, and Ioannis Kompatsiaris. Healthcare Technologies, Institution of Engineering and Technology. Institution of Engineering and Technology, 2020. Chap. 6, pp. 117–144. URL: `https://digital-library.theiet.org/content/books/10.1049/pbce129e%5C_ch6`.

[102] Chandan Kumar, **Raphael Menges**, and Steffen Staab. “Assessing the Usability of Gaze-Adapted Interface against Conventional Eye-Based Input Emulation”. In: *2017 IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS)*. June 2017, pp. 793–798.

[103] Chandan Kumar, **Raphael Menges**, and Steffen Staab. “Eye-Controlled Interfaces for Multimedia Interaction”. In: *IEEE MultiMedia* 23.4 (Oct. 2016), pp. 6–13.

[104] Chandan Kumar, **Raphael Menges**, and Steffen Staab. "Eye-Controlled Interfaces for Multimedia Interaction". In: *IEEE MultiMedia* 23.4 (Oct. 2016), pp. 6–13. URL: `https://doi.org/10.1109/MMUL.2016.52`.

[105] Manu Kumar, Tal Garfinkel, Dan Boneh, and Terry Winograd. "Reducing Shoulder-surfing by Using Gaze-based Password Entry". In: *Proceedings of the 3rd Symposium on Usable Privacy and Security*. SOUPS '07. Pittsburgh, Pennsylvania, USA: ACM, 2007, pp. 13–19. URL: `http://doi.acm.org/10.1145/1280680.1280683`.

[106] Manu Kumar, Jeff Klingner, Rohan Puranik, Terry Winograd, and Andreas Paepcke. "Improving the Accuracy of Gaze Input for Interaction". In: *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*. ETRA '08. Savannah, Georgia: ACM, 2008, pp. 65–68. URL: `http://doi.acm.org/10.1145/1344471.1344488`.

[107] Manu Kumar, Andreas Paepcke, Terry Winograd, and Terry Winograd. "EyePoint: Practical Pointing and Selection Using Gaze and Keyboard". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '07. San Jose, California, USA: ACM, 2007, pp. 421–430. URL: `http://doi.acm.org/10.1145/1240624.1240692`.

[108] Manu Kumar and Terry Winograd. "GUIDe: Gaze-enhanced UI Design". In: *CHI '07 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '07. San Jose, CA, USA: ACM, 2007, pp. 1977–1982. URL: `http://doi.acm.org/10.1145/1240866.1240935`.

[109] Ranjitha Kumar, Arvind Satyanarayan, Cesar Torres, Maxine Lim, Salman Ahmad, Scott R. Klemmer, and Jerry O. Talton. "Webzeitgeist: Design Mining the Web". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 3083–3092. URL: `http://doi.acm.org/10.1145/2470654.2466420`.

[110] Andrew Kurauchi, Wenxin Feng, Ajjen Joshi, Carlos Morimoto, and Margrit Betke. "EyeSwipe: Dwell-free Text Entry Using Gaze Paths". In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. San Jose, California, USA: ACM, 2016, pp. 1952–1956. URL: `http://doi.acm.org/10.1145/2858036.2858335`.

[111] Kuno Kurzhals, Marcel Hlawatsch, Michael Burch, and Daniel Weiskopf. "Fixation-image Charts". In: *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*. ETRA '16. Charleston, South Carolina: ACM, 2016, pp. 11–18. URL: `http://doi.acm.org/10.1145/2857491.2857507`.

[112] Kuno Kurzhals, Marcel Hlawatsch, Florian Heimerl, Michael Burch, Thomas Ertl, and Daniel Weiskopf. "Gaze Stripes: Image-Based Visualization of Eye Tracking Data". In: *IEEE Transactions on Visualization and Computer Graphics* 22.1 (Jan. 2016). Ed. by IEEE, pp. 1005–1014. URL: `http://dx.doi.org/10.1109/TVCG.2015.2468091`.

[113] Kuno Kurzhals, Marcel Hlawatsch, Christof Seeger, and Daniel Weiskopf. “Visual Analytics for Mobile Eye Tracking”. In: *IEEE Transactions on Visualization and Computer Graphics* 23.1 (Jan. 2017), pp. 301–310. URL: `https://doi.org/10.1109/TVCG.2016.2598695`.

[114] Fabrizio Lamberti, Gianluca Paravati, Valentina Gatteschi, and Alberto Cannavo. “Supporting Web Analytics by Aggregating User Interaction Data From Heterogeneous Devices Using Viewport-DOM-Based Heat Maps”. In: *IEEE Transactions on Industrial Informatics* 13.4 (2017), pp. 1989–1999. URL: `https://doi.org/10.1109/TII.2017.2658663`.

[115] J. Richard Landis and Gary G. Koch. “The measurement of observer agreement for categorical data.” In: *Biometrics* 33 1 (1977), pp. 159–74.

[116] Chris Lankford. “Effective Eye-gaze Input into Windows”. In: *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications*. ETRA ’00. Palm Beach Gardens, Florida, USA: ACM, 2000, pp. 23–27. URL: `http://doi.acm.org/10.1145/355017.355021`.

[117] Pierre R. Lebreton, Toni Mäki, Evangelos Skodras, Isabelle Hupont, and Matthias Hirth. “Bridging the gap between eye tracking and crowdsourcing.” In: *Human Vision and Electronic Imaging*. 2015, 93940W. URL: `https://doi.org/10.1117/12.2076745`.

[118] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. “Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning”. In: *Journal of Machine Learning Research* 18.17 (2017), pp. 1–5. URL: `http://jmlr.org/papers/v18/16-365`.

[120] Thomas F. Liu, Mark Craft, Jason Situ, Ersin Yumer, Radomir Mech, and Ranjitha Kumar. “Learning Design Semantics for Mobile Apps”. In: *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. UIST ’18. Berlin, Germany: ACM, 2018, pp. 569–579. URL: `http://doi.acm.org/10.1145/3242587.3242650`.

[121] Margaret Livingstone. “Vision and art : the biology of seeing”. In: New York, N.Y: Abrams, Harry N., 2002, pp. 46–67.

[122] David G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision* 60.2 (Nov. 2004), pp. 91–110. URL: `https://doi.org/10.1023/B:VISI.0000029664.99615.94`.

[123] Christof Lutteroth, Moiz Penkar, and Gerald Weber. “Gaze vs. Mouse: A Fast and Accurate Gaze-Only Click Alternative”. In: *Proceedings of the 28th Annual ACM Symposium on User Interface Software; Technology*. UIST ’15. Charlotte, NC, USA: ACM, 2015, pp. 385–394. URL: `http://doi.acm.org/10.1145/2807442.2807461`.

[124] Yu-Seung Ma, Jeff Offutt, and Yong-Rae Kwon. “MuJava: A Mutation System for Java”. In: *Proceedings of the 28th International Conference on Software Engineering*. ICSE ’06. Shanghai, China: ACM, 2006, pp. 827–830. URL: `http://doi.acm.org/10.1145/1134285.1134425`.

[125] I. Scott MacKenzie. “Evaluating eye tracking systems for computer input”. In: *Gaze interaction and applications of eye tracking: Advances in assistive technologies*. IGI Global, 2012, pp. 205–225.

[126] I. Scott MacKenzie. “Fitts’ Law As a Research and Design Tool in Human-computer Interaction”. In: *Hum.-Comput. Interact.* 7.1 (Mar. 1992), pp. 91–139. URL: `http://dx.doi.org/10.1207/s15327051hci0701%5C_3`.

[127] Sonal Mahajan and William G. J. Halfond. “Detection and Localization of HTML Presentation Failures Using Computer Vision-Based Techniques”. In: *2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST)*. Apr. 2015, pp. 1–10.

[128] Jalal U. Mahmud, Yevgen Borodin, and I. V. Ramakrishnan. “Csurf: A Context-driven Non-visual Web-browser”. In: *Proceedings of the 16th International Conference on World Wide Web*. WWW ’07. Banff, Alberta, Canada: ACM, 2007, pp. 31–40. URL: `http://doi.acm.org/10.1145/1242572.1242578`.

[129] Päivi Majaranta. *Text entry by eye gaze*. University of Tampere, 2009.

[130] Päivi Majaranta, Hirotaka Aoki, Mick Donegan, Dan Witzner Hansen, and John Paulin Hansen. *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies*. 1st. Hershey, PA: IGI Global, 2011.

[131] Jennifer Mankoff, Anind Dey, Udit Batra, and Melody Moore. “Web Accessibility for Low Bandwidth Input”. In: *Proceedings of the Fifth International ACM Conference on Assistive Technologies*. Assets ’02. Edinburgh, Scotland: ACM, 2002, pp. 17–24. URL: `http://doi.acm.org/10.1145/638249.638255`.

[132] Markets and Markets Research Private Ltd. *Application Testing Services Market by Service Type (Professional, Managed), Testing Type (Functionality, Security, Automation), Delivery Model (Onshore, Offshore), Organization Size, Vertical, and Region - Global Forecast to 2022*. 2017. URL: `https://www.marketsandmarkets.com/Market-Reports/application-testing-services-market-235108129.html`.

[133] **Raphael Menges**, Sophia Kramer, Stefan Hill, Marius Nisslmueller, Chandan Kumar, and Steffen Staab. “A Visualization Tool for Eye Tracking Data Analysis in the Web”. In: *ACM Symposium on Eye Tracking Research and Applications*. ETRA ’20 Short Papers. Stuttgart, Germany: Association for Computing Machinery, 2020. URL: `https://doi.org/10.1145/3379156.3391831`.

[134] **Raphael Menges**, Chandan Kumar, Daniel Müller, and Korok Sengupta. “GazeTheWeb: A Gaze-Controlled Web Browser”. In: *Proceedings of the 14th Web for All Conference on The Future of Accessible Work*. W4A ’17. Perth, Western Australia, Australia: ACM, 2017, 25:1–25:2. URL: `http://doi.acm.org/10.1145/3058555.3058582`.

[135] **Raphael Menges**, Chandan Kumar, Korok Sengupta, and Steffen Staab. "eyeGUI: A Novel Framework for Eye-Controlled User Interfaces". In: *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*. NordiCHI '16. Gothenburg, Sweden: ACM, 2016, 121:1–121:6. URL: `http://doi.acm.org/10.1145/2971485.2996756`.

[136] **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Eye tracking for Interaction: Adapting Multimedia Interfaces". In: *Signal Processing to Drive Human-Computer Interaction: EEG and eye-controlled interfaces*. Ed. by Spiros Nikolopoulos, Chandan Kumar, and Ioannis Kompatsiaris. Healthcare Technologies, Institution of Engineering and Technology. Institution of Engineering and Technology, 2020. Chap. 5, pp. 83–116. URL: `https://digital-library.theiet.org/content/books/10.1049/pbce129e%5C_ch5`.

[137] **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Improving User Experience of Eye Tracking-Based Interaction: Introspecting and Adapting Interfaces". In: *ACM Trans. Comput.-Hum. Interact.* 26.6 (Nov. 2019). Accepted May 2019, 37:1–37:46. URL: `http://doi.acm.org/10.1145/3338844`.

[138] **Raphael Menges**, Chandan Kumar, Ulrich Wechselberger, Christoph Schaefer, Tina Walber, and Steffen Staab. "Schau genau! A Gaze-Controlled 3D Game for Entertainment and Education". In: *Journal of Eye Movement Research* (Wuppertal). Vol. 10. 6. 2017, p. 220.

[139] **Raphael Menges**, Hanadi Tamimi, Chandan Kumar, Tina Walber, Christoph Schaefer, and Steffen Staab. "Enhanced Representation of Web Pages for Usability Analysis with Eye Tracking". In: *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*. ETRA '18. Warsaw, Poland: ACM, 2018, 18:1–18:9. URL: `http://doi.acm.org/10.1145/3204493.3204535`.

[140] Vasileios Mezaris, Evlampios Apostolidis, and Alexandros Pournaras. *Video Shot and Scene Segmentation - MKLab*. URL: `https://mklab.iti.gr/results/video-shot-and-scene-segmentation`.

[141] Dalibor Mitrovic, Stefan Hartlieb, Matthias Zeppelzauer, and Christian Breiteneder. "Scene Segmentation in Artistic Archive Documentaries". In: *HCI in Work and Learning, Life and Leisure*. Vortrag: 6th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering, USAB 2010, Klagenfurt; 2010-11-04 – 2010-11-05. 6389: Springer-Verlag Berlin Heidelberg, 2010, pp. 400–410. URL: `http://publik.tuwien.ac.at/files/PubDat%5C_188968.pdf`.

[142] Dalibor Mitrović, Stefan Hartlieb, Matthias Zeppelzauer, and Maia Zaharieva. "Scene Segmentation in Artistic Archive Documentaries". In: *HCI in Work and Learning, Life and Leisure*. Ed. by Gerhard Leitner, Martin Hitz, and Andreas Holzinger. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 400–410.

[143] Sangwhan Moon, Terence Eden, Shwetank Dixit, Bruce Lawson, Xiaoqian Wu, Scott O'Hara, and Patricia Aas. *HTML 5.3*. W3C Working Draft. https://www.w3.org/TR/2018/WD-html53-20181018/. W3C, Oct. 2018.

[144] Atsuo Murata. "Eye-gaze input versus mouse: Cursor control as a function of age". In: *Int. J. Hum. Comput. Interaction* 21 (2006), pp. 1–14.

[145] Jakob Nielsen and Rolf Molich. "Heuristic Evaluation of User Interfaces". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '90. Seattle, Washington, USA: ACM, 1990, pp. 249–256. URL: `http://doi.acm.org/10.1145/97243.97281`.

[146] Jakob Nielsen and Kara Pernice. *Eyetracking Web Usability*. 1st. Thousand Oaks, CA, USA: New Riders Publishing, 2009.

[147] Spiros Nikolopoulos, Panagiotis C. Petrantonakis, Kostas Georgiadis, Fotis Kalaganis, Georgios Liaros, Ioulietta Lazarou, Katerina Adam, Anastasios Papazoglou-Chalikias, Elisavet Chatzilari, Vangelis P. Oikonomou, Chandan Kumar, **Raphael Menges**, Steffen Staab, Daniel Müller, Korok Sengupta, Sevasti Bostantjopoulou, Zoe Katsarou, Gabi Zeilig, Meir Plotnik, Amihai Gotlieb, Racheli Kizoni, Sofia Fountoukidou, Jaap Ham, Dimitrios Athanasiou, Agnes Mariakaki, Dario Comanducci, Edoardo Sabatini, Walter Nistico, Markus Plank, and Ioannis Kompatsiaris. "A multimodal dataset for authoring and editing multimedia content: The MAMEM project". In: *Data in Brief* 15 (2017), pp. 1048–1056. URL: `http://www.sciencedirect.com/science/article/pii/S2352340917305930`.

[148] RealEye sp. z o. o. *RealEye*. Jan. 2018. URL: `https://www.realeye.io`.

[149] Takehiko Ohno, Naoki Mukawa, and Atsushi Yoshikawa. "FreeGaze: A Gaze Tracking System for Everyday Gaze Interaction". In: *Proceedings of the 2002 Symposium on Eye Tracking Research & Applications*. ETRA '02. New Orleans, Louisiana: ACM, 2002, pp. 125–132. URL: `http://doi.acm.org/10.1145/507072.507098`.

[150] Dan R. Olsen Jr., Sean Jefferies, Travis Nielsen, William Moyes, and Paul Fredrickson. "Cross-modal Interaction Using XWeb". In: *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*. UIST '00. San Diego, California, USA: ACM, 2000, pp. 191–200. URL: `http://doi.acm.org/10.1145/354401.354764`.

[151] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and et al. "Scikit-Learn: Machine Learning in Python". In: *J. Mach. Learn. Res.* 12.null (Nov. 2011), pp. 2825–2830.

[152] Ken Pfeuffer and Hans Gellersen. "Gaze and Touch Interaction on Tablets". In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. UIST '16. Tokyo, Japan: ACM, 2016, pp. 301–311. URL: `http://doi.acm.org/10.1145/2984511.2984514`.

[153] Alex Poole and Linden J. Ball. “Eye tracking in human-computer interaction and usability research: Current status and future prospects”. In: *Prospects, Chapter in C. Ghaoui (Ed.): Encyclopedia of Human-Computer Interaction. Pennsylvania: Idea Group, Inc*. Jan. 2006, pp. 211–219.

[154] Marco Porta and Alessia Ravelli. “WeyeB, an Eye-controlled Web Browser for Hands-free Navigation”. In: *Proceedings of the 2nd Conference on Human System Interactions*. HSI’09. Catania, Italy: IEEE Press, 2009, pp. 207–212. URL: `http://dl.acm.org/citation.cfm?id=1689359.1689396`.

[155] Publications Office of the European Union. “Communication: Shaping Europe’s digital future”. In: (2020). URL: `https://ec.europa.eu/info/sites/info/files/communication-shaping-europes-digital-future-feb2020%5C_en%5C_4.pdf`.

[156] Dario D. Salvucci and Joseph H. Goldberg. “Identifying Fixations and Saccades in Eye-tracking Protocols”. In: *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications*. ETRA ’00. Palm Beach Gardens, Florida, USA: ACM, 2000, pp. 71–78. URL: `http://doi.acm.org/10.1145/355017.355028`.

[157] Vagner F. de Santana, Rosimeire de Oliveira, Leonelo D. A. Almeida, and Marcia Ito. “Firefixia: An Accessibility Web Browser Customization Toolbar for People with Dyslexia”. In: *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility*. W4A ’13. Rio de Janeiro, Brazil: ACM, 2013, 16:1–16:4. URL: `http://doi.acm.org/10.1145/2461121.2461137`.

[158] Christoph Schaefer, Matthias Kuich, **Raphael Menges**, Kevin Schmidt, and Tina Walber. “Schau genau! - an Eye Tracking Game With a Purpose”. In: *1st. Workshop on the Applications for Gaze in Games at CHI Play 2014* (2014).

[159] Simon Schenk, Marc Dreiser, Gerhard Rigoll, and Michael Dorr. “GazeEverywhere: Enabling Gaze-only User Interaction on an Unmodified Desktop PC in Everyday Scenarios”. In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI ’17. Denver, Colorado, USA: ACM, 2017, pp. 3034–3044. URL: `http://doi.acm.org/10.1145/3025453.3025455`.

[160] Michael Schiessl, Sabrina Duda, Andreas Thölke, and Rico Fischer. “Eye tracking and its application in usability and media research”. In: *MMI interaktiv* 6.6 (Jan. 2003).

[161] Julia Schwarz, Scott Hudson, Jennifer Mankoff, and Andrew D. Wilson. “A Framework for Robust and Flexible Handling of Inputs with Uncertainty”. In: *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*. UIST ’10. New York, New York, USA: ACM, 2010, pp. 47–56. URL: `http://doi.acm.org/10.1145/1866029.1866039`.

[162] Korok Sengupta, Sabin Bhattarai, Sayan Sarcar, I. Scott MacKenzie, and Steffen Staab. "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. Honolulu, HI, USA: Association for Computing Machinery, 2020, pp. 1–11. URL: `https://doi.org/10.1145/3313831.3376579`.

[163] Korok Sengupta, Min Ke, **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Hands-free Web Browsing: Enriching the User Experience with Gaze and Voice Modality". In: *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*. ETRA '18. Warsaw, Poland: ACM, 2018, 88:1–88:3. URL: `http://doi.acm.org/10.1145/3204493.3208338`.

[164] Korok Sengupta, Chandan Kumar, and Steffen Staab. "Usability Heuristics for Eye-controlled User Interfaces". In: COGAIN Symposium (Wuppertal, Germany). 19th European Conference on Eye Movements. 2017. URL: `http://cogain2017.cogain.org/camready/poster3-Sengupta.pdf`.

[165] Korok Sengupta, **Raphael Menges**, Chandan Kumar, and Steffen Staab. "GazeTheKey: Interactive Keys to Integrate Word Predictions for Gaze-based Text Entry". In: *Proceedings of the 22Nd International Conference on Intelligent User Interfaces Companion*. IUI '17 Companion. Limassol, Cyprus: ACM, 2017, pp. 121–124. URL: `http://doi.acm.org/10.1145/3030024.3038259`.

[166] Korok Sengupta, **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Impact of Variable Positioning of Text Prediction in Gaze-based Text Entry". In: *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. ETRA '19. Denver, Colorado: ACM, 2019, 74:1–74:9. URL: `http://doi.acm.org/10.1145/3317956.3318152`.

[167] Korok Sengupta, Jun Sun, **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Analyzing the Impact of Cognitive Load in Evaluating Gaze-Based Typing". In: *2017 IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS)*. Vol. Special Track on Multimodal Interfaces for Natural Human Computer Interaction: Theory and Applications. IEEE, June 2017, pp. 787–792.

[168] SensoMotoric Instruments. *beGaze*. accessed on 28th March 2020. Jan. 2020. URL: `https://gazeintelligence.com/smi-product-manual`.

[169] Ben Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. 3rd. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997.

[170] Jakub Simko and Jakub Vrba. "Screen Recording Segmentation to Scenes for Eye-Tracking Analysis". In: *Multimedia Tools Appl.* 78.2 (Jan. 2019), pp. 2401–2425. URL: `https://doi.org/10.1007/s11042-018-6369-7`.

[171] Shyamli Sindhwani, Christof Lutteroth, and Gerald Weber. “ReType: Quick Text Editing with Keyboard and Gaze”. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI ’19. Glasgow, Scotland Uk: ACM, 2019, 203:1–203:13. URL: `http://doi.acm.org/10.1145/3290605.3300433`.

[172] Laurianne Sitbon, Oscar Wong, and Margot Brereton. “Efficient Web Browsing with a Single-switch”. In: *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: The Future of Design*. OzCHI ’14. Sydney, New South Wales, Australia: ACM, 2014, pp. 515–518. URL: `http://doi.acm.org/10.1145/2686612.2686693`.

[173] Ray Smith. “An Overview of the Tesseract OCR Engine”. In: *Proceedings of the Ninth International Conference on Document Analysis and Recognition - Volume 02*. ICDAR ’07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 629–633. URL: `http://dl.acm.org/citation.cfm?id=1304596.1304846`.

[174] Jiguo Song and Gabriel Parmer. “Toward Predictable, Efficient, System-level Tolerance of Transient Faults”. In: *SIGBED Rev.* 10.4 (Dec. 2013), pp. 53–56. URL: `http://doi.acm.org/10.1145/2583687.2583700`.

[175] Sören Sonnenburg, Gunnar Rätsch, Sebastian Henschel, Christian Widmer, Jonas Behr, Alexander Zien, Fabio De Bona, Alexander Binder, Christian Gehl, and Vojtech Franc. “The SHOGUN Machine Learning Toolbox”. In: *Journal of Machine Learning Research* 11 (2010), pp. 1799–1802. URL: `https://dl.acm.org/citation.cfm?id=1859911`.

[176] Oleg Špakov and Darius Miniotas. “Gaze-based Selection of Standard-size Menu Items”. In: *Proceedings of the 7th International Conference on Multimodal Interfaces*. ICMI ’05. Torento, Italy: ACM, 2005, pp. 124–128. URL: `http://doi.acm.org/10.1145/1088463.1088486`.

[177] Wolfgang Stuerzlinger, Olivier Chapuis, Dusty Phillips, and Nicolas Roussel. “User Interface Façades: Towards Fully Adaptable User Interfaces”. In: *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*. UIST ’06. Montreux, Switzerland: ACM, 2006, pp. 309–318. URL: `http://doi.acm.org/10.1145/1166253.1166301`.

[178] Evgeny Suslikov. *FireShot*. Jan. 2018. URL: `https://getfireshot.com`.

[179] Julius Sweetland. *Optikey: Type, Click, Speak*. `https://github.com/OptiKey/OptiKey`. 2016.

[180] The WebM Project. *The WebM Project*. 2019. URL: `https://www.webmproject.org/code`.

[181] Tobii AB. *Roadshow Presentation*. Sept. 2019. URL: `https://www.tobii.com/siteassets/tobii-group/investor-relations/roadshow-sep-17-18-2019.pdf/?v=1`.

[182] Tobii AB. *Tobii Pro Lab User’s Manual*. Version 1.138.1. Apr. 2020.

[183] Tobii AB. *Tobii Year-End Report 2019*. 2020. URL: `https://www.tobii.com/group/investors/financial-reports/2020/2/tobii-year-end-report-2019/`.

[184] Ba Tu Truong, Svetha Venkatesh, and Chitra Dorai. "Scene extraction in motion pictures". In: *IEEE Transactions on Circuits and Systems for Video Technology* 13.1 (Jan. 2003), pp. 5–15.

[185] Outi Tuisku, Päivi Majaranta, Poika Isokoski, and Kari-Jouko Räihä. "Now Dasher! Dash Away!: Longitudinal Study of Fast Text Entry by Eye Gaze". In: *Proceedings of the 2008 Symposium on Eye Tracking Research & Applications*. ETRA '08. Savannah, Georgia: ACM, 2008, pp. 19–26. URL: `http://doi.acm.org/10.1145/1344471.1344476`.

[186] Jos Nicolaas Van Der Geest and Maarten Frens. "Recording eye movements with video-oculography and scleral search coils: a direct comparison of two methods". In: *Journal of Neuroscience Methods* 114.2 (2002), pp. 185–195. URL: `http://www.sciencedirect.com/science/article/pii/S0165027001005271`.

[187] Jeroen Vendrig and Marcel Worring. "Systematic Evaluation of Logical Story Unit Segmentation". In: *Trans. Multi.* 4.4 (Dec. 2002), pp. 492–499. URL: `http://dx.doi.org/10.1109/TMM.2002.802021`.

[188] Stephen Vickers, Howell Istance, and Aulikki Hyrskykari. "Performing Locomotion Tasks in Immersive Computer Games with an Adapted Eye-Tracking Interface". In: *ACM Trans. Access. Comput.* 5.1 (Sept. 2013), 2:1–2:33. URL: `http://doi.acm.org/10.1145/2514856`.

[189] Tina Walber, Chantal Neuhaus, and Ansgar Scherp. "EyeGrab: A Gaze-based Game with a Purpose to Enrich Image Context Information". In: *Proceedings of the 2nd European Workshop on Human-Computer Interaction and Information Retrieval, Nijmegen, The Netherlands, August 25, 2012*. 2012, pp. 63–66. URL: `http://ceur-ws.org/Vol-909/poster8.pdf`.

[190] Jingtao Wang and Jennifer Mankoff. "Theoretical and Architectural Support for Input Device Adaptation". In: *SIGCAPH Comput. Phys. Handicap.* 73-74 (June 2002), pp. 85–92. URL: `http://doi.acm.org/10.1145/960201.957220`.

[191] Zhou Wang, Alan Bovik, Hamid Sheikh, and Eero Simoncelli. "Image Quality Assessment: From Error Visibility to Structural Similarity". In: *Trans. Img. Proc.* 13.4 (Apr. 2004), pp. 600–612. URL: `http://dx.doi.org/10.1109/TIP.2003.819861`.

[192] Benjamin Wassermann, Adrian Hardt, and Gottfried Zimmermann. "Generic Gaze Interaction Events for Web Browsers Using the Eye Tracker as Input Device". In: *Proceedings of the 2012 Workshop on Emerging Web Technologies at Conference on World Wide Web*. Apr. 2012.

[193] Tim Weninger, Rodrigo Palacios, Valter Crescenzi, Thomas Gottron, and Paolo Merialdo. "Web Content Extraction: A MetaAnalysis of Its Past and Thoughts on Its Future". In: *SIGKDD Explor. Newsl.* 17.2 (Feb. 2016), pp. 17–23. URL: `http://doi.acm.org/10.1145/2897350.2897353`.

[194] Tom Yeh, Tsung-Hsiang Chang, and Robert C. Miller. “Sikuli: Using GUI Screenshots for Search and Automation”. In: *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology*. UIST ’09. Victoria, BC, Canada: ACM, 2009, pp. 183–192. URL: `http://doi.acm.org/10.1145/1622176.1622213`.

[195] Ramin Zabih, Justin Miller, and Kevin Mai. “A Feature-based Algorithm for Detecting and Classifying Scene Breaks”. In: *Proceedings of the Third ACM International Conference on Multimedia*. MULTIMEDIA ’95. San Francisco, California, USA: ACM, 1995, pp. 189–200. URL: `http://doi.acm.org/10.1145/217279.215266`.

[196] Herbert Zettl. “Essentials of Applied Media Aesthetics”. In: *Media Computing: Computational Media Aesthetics*. Ed. by Chitra Dorai and Svetha Venkatesh. Boston, MA: Springer US, 2002, pp. 11–38. URL: `https://doi.org/10.1007/978-1-4615-1119-9%5C_2`.

[197] Shumin Zhai, Carlos Morimoto, and Steven Ihde. “Manual and Gaze Input Cascaded (MAGIC) Pointing”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’99. Pittsburgh, Pennsylvania, USA: ACM, 1999, pp. 246–253. URL: `http://doi.acm.org/10.1145/302979.303053`.

[198] Xiaoyi Zhang, Anne Spencer Ross, and James Fogarty. “Robust Annotation of Mobile Application Interfaces in Methods for Accessibility Repair and Enhancement”. In: *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. UIST ’18. Berlin, Germany: ACM, 2018, pp. 609–621. URL: `http://doi.acm.org/10.1145/3242587.3242616`.

[199] Xuan Zhang and I. Scott MacKenzie. “Evaluating Eye Tracking with ISO 9241 - Part 9”. In: *Proceedings of the 12th International Conference on Human-computer Interaction: Intelligent Multimodal Interaction Environments*. HCI’07. Beijing, China: Springer-Verlag, 2007, pp. 779–788. URL: `http://dl.acm.org/citation.cfm?id=1769590.1769678`.

[200] Xuebai Zhang, Xiaolong Liu, Shyan-Ming Yuan, and Shu-Fan Lin. “Eye Tracking Based Control System for Natural Human-Computer Interaction”. In: *Computational Intelligence and Neuroscience: CIN* (2017).

List of Figures

List of Tables

Listings

Curriculum Vitae

Raphael Philipp Menges

Universität Stuttgart,
Universitätsstr. 32,
70569 Stuttgart,
Germany.

Tel.: +49 711 685-88112
E-mail: raphael.menges@ipvs.uni-stuttgart.de

Education

Year	Institution	Degree
2016	Universität Koblenz-Landau, Koblenz, Computational Visualistics Programme	M. Sc. in Computer Science
2014	Universität Koblenz-Landau, Koblenz, Computational Visualistics Programme	B. Sc. in Computer Science
2011	Wilhelm-Hofmann-Gymnasium, St.Goarshausen	Abitur

Positions

Years	Institution	Position
2021 – today	Universität Stuttgart, Stuttgart Institute for Parallel and Distributed Systems, Analytic Computing	Scientific Employee
2016 – 2021	Universität Koblenz-Landau, Koblenz Institute for Web Science and Technologies	Scientific Employee

Teaching

Year	Course
2017 – 2021	Tutorial: Machine Learning and Data Mining
2021	Proseminar: Eye Tracking
2020	Research Lab: Eye Tracking in Word Processing
2019	Research Lab: Eye Tracking Visualization Platform
2017	Research Lab: GazeTheWeb - Watch
2016	Research Lab: GazeTheWeb - Tweet

Supervised Theses

Degree	Author and Title	Status
B. Sc. in Computer Science	Daniel Vossen: Shot Detection in Screencasts of Web Browsing with Convolutional Neural Networks	Finished
B. Sc. in Computer Science	Christopher Dreide: Optical Text Recognition in the Web	Finished
B. Sc. in Computer Science	Christian Brozmann: Visualization of Transitions on Web Sites for Usability Studies with Eye Tracking	Finished
M. Sc. in Computer Science	Hanadi Tamimi: Intelligent Mapping of Eye-Tracking Gaze-Data on Fixed Web Page Elements	Finished

Publications

1. Ramin Hedeshy, Chandan Kumar, **Raphael Menges**, and Steffen Staab. "Hummer: Text Entry by Gaze and Hum". In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. CHI '21. Yokohama, Japan: Association for Computing Machinery, 2021. URL: `https://doi.org/10.1145/3411764.3445501`
2. **Raphael Menges**, Sophia Kramer, Stefan Hill, Marius Nisslmueller, Chandan Kumar, and Steffen Staab. "A Visualization Tool for Eye Tracking Data Analysis in the Web". In: *ACM Symposium on Eye Tracking Research and Applications*. ETRA '20 Short Papers. Stuttgart, Germany: Association for Computing Machinery, 2020. URL: `https://doi.org/10.1145/3379156.3391831`
3. Ramin Hedeshy, Chandan Kumar, **Raphael Menges**, and Steffen Staab. "GIUPlayer: A Gaze Immersive YouTube Player Enabling Eye Control and Attention Analysis". In: *ACM Symposium*

on Eye Tracking Research and Applications. ETRA '20 Adjunct. Stuttgart, Germany: Association for Computing Machinery, 2020. URL: `https://doi.org/10.1145/3379157.3391984`

4. Chandan Kumar, **Raphael Menges**, Korok Sengupta, and Steffen Staab. "Eye tracking for Interaction: Evaluation Methods". In: *Signal Processing to Drive Human-Computer Interaction: EEG and eye-controlled interfaces*. Ed. by Spiros Nikolopoulos, Chandan Kumar, and Ioannis Kompatsiaris. Healthcare Technologies, Institution of Engineering and Technology. Institution of Engineering and Technology, 2020. Chap. 6, pp. 117–144. URL: `https://digital-library.theiet.org/content/books/10.1049/pbce129e%5C_ch6`

5. **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Eye tracking for Interaction: Adapting Multimedia Interfaces". In: *Signal Processing to Drive Human-Computer Interaction: EEG and eye-controlled interfaces*. Ed. by Spiros Nikolopoulos, Chandan Kumar, and Ioannis Kompatsiaris. Healthcare Technologies, Institution of Engineering and Technology. Institution of Engineering and Technology, 2020. Chap. 5, pp. 83–116. URL: `https://digital-library.theiet.org/content/books/10.1049/pbce129e%5C_ch5`

6. **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Improving User Experience of Eye Tracking-Based Interaction: Introspecting and Adapting Interfaces". In: *ACM Trans. Comput.-Hum. Interact.* 26.6 (Nov. 2019). Accepted May 2019, 37:1–37:46. URL: `http://doi.acm.org/10.1145/3338844`

7. Chandan Kumar, Daniyal Akbari, **Raphael Menges**, Scott MacKenzie, and Steffen Staab. "TouchGazePath: Multimodal Interaction with Touch and Gaze Path for Secure Yet Efficient PIN Entry". In: *2019 International Conference on Multimodal Interaction*. ICMI '19. Suzhou, China: ACM, 2019, pp. 329–338. URL: `http://doi.acm.org/10.1145/3340555.3353734`

8. Korok Sengupta, **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Impact of Variable Positioning of Text Prediction in Gaze-based Text Entry". In: *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. ETRA '19. Denver, Colorado: ACM, 2019, 74:1–74:9. URL: `http://doi.acm.org/10.1145/3317956.3318152`

9. Nils Lichtenberg, **Raphael Menges**, Vladimir Ageev, Ajay Abisheck Paul George, Pascal Heimer, Diana Imhof, and Kai Lawonn. "Analyzing Residue Surface Proximity to Interpret Molecular Dynamics". In: *Computer Graphics Forum* (2018)

10. **Raphael Menges**, Hanadi Tamimi, Chandan Kumar, Tina Walber, Christoph Schaefer, and Steffen Staab. "Enhanced Representation of Web Pages for Usability Analysis with Eye Tracking". In: *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*. ETRA '18. Warsaw, Poland: ACM, 2018, 18:1–18:9. URL: `http://doi.acm.org/10.1145/3204493.3204535`

11. Korok Sengupta, Min Ke, **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Hands-free Web Browsing: Enriching the User Experience with Gaze and Voice Modality". In: *Proceedings of the*

2018 ACM Symposium on Eye Tracking Research & Applications. ETRA '18. Warsaw, Poland: ACM, 2018, 88:1–88:3. URL: `http://doi.acm.org/10.1145/3204493.3208338`

12. Spiros Nikolopoulos, Panagiotis C. Petrantonakis, Kostas Georgiadis, Fotis Kalaganis, Georgios Liaros, Ioulietta Lazarou, Katerina Adam, Anastasios Papazoglou-Chalikias, Elisavet Chatzilari, Vangelis P. Oikonomou, Chandan Kumar, **Raphael Menges**, Steffen Staab, Daniel Müller, Korok Sengupta, Sevasti Bostantjopoulou, Zoe Katsarou, Gabi Zeilig, Meir Plotnik, Amihai Gotlieb, Racheli Kizoni, Sofia Fountoukidou, Jaap Ham, Dimitrios Athanasiou, Agnes Mariakaki, Dario Comanducci, Edoardo Sabatini, Walter Nistico, Markus Plank, and Ioannis Kompatsiaris. "A multimodal dataset for authoring and editing multimedia content: The MAMEM project". In: *Data in Brief* 15 (2017), pp. 1048–1056. URL: `http://www.sciencedirect.com/science/article/pii/S2352340917305930`

13. Chandan Kumar, **Raphael Menges**, and Steffen Staab. "Assessing the Usability of Gaze-Adapted Interface against Conventional Eye-Based Input Emulation". In: *2017 IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS)*. June 2017, pp. 793–798

14. Korok Sengupta, Jun Sun, **Raphael Menges**, Chandan Kumar, and Steffen Staab. "Analyzing the Impact of Cognitive Load in Evaluating Gaze-Based Typing". In: *2017 IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS)*. vol. Special Track on Multimodal Interfaces for Natural Human Computer Interaction: Theory and Applications. IEEE, June 2017, pp. 787–792

15. **Raphael Menges**, Chandan Kumar, Daniel Müller, and Korok Sengupta. "GazeTheWeb: A Gaze-Controlled Web Browser". In: *Proceedings of the 14th Web for All Conference on The Future of Accessible Work*. W4A '17. Perth, Western Australia, Australia: ACM, 2017, 25:1–25:2. URL: `http://doi.acm.org/10.1145/3058555.3058582`

16. **Raphael Menges**, Chandan Kumar, Ulrich Wechselberger, Christoph Schaefer, Tina Walber, and Steffen Staab. "Schau genau! A Gaze-Controlled 3D Game for Entertainment and Education". In: *Journal of Eye Movement Research* (Wuppertal). Vol. 10. 6. 2017, p. 220

17. Korok Sengupta, **Raphael Menges**, Chandan Kumar, and Steffen Staab. "GazeTheKey: Interactive Keys to Integrate Word Predictions for Gaze-based Text Entry". In: *Proceedings of the 22Nd International Conference on Intelligent User Interfaces Companion*. IUI '17 Companion. Limassol, Cyprus: ACM, 2017, pp. 121–124. URL: `http://doi.acm.org/10.1145/3030024.3038259`

18. Chandan Kumar, **Raphael Menges**, and Steffen Staab. "Eye-Controlled Interfaces for Multimedia Interaction". In: *IEEE MultiMedia* 23.4 (Oct. 2016), pp. 6–13. URL: `https://doi.org/10.1109/MMUL.2016.52`

19. **Raphael Menges**, Chandan Kumar, Korok Sengupta, and Steffen Staab. "eyeGUI: A Novel Framework for Eye-Controlled User Interfaces". In: *Proceedings of the 9th Nordic Conference on*

Human-Computer Interaction. NordiCHI ’16. Gothenburg, Sweden: ACM, 2016, 121:1–121:6. URL: `http://doi.acm.org/10.1145/2971485.2996756`

20. Christoph Schaefer, Matthias Kuich, **Raphael Menges**, Kevin Schmidt, and Tina Walber. “Schau genau! - an Eye Tracking Game With a Purpose”. In: *1st. Workshop on the Applications for Gaze in Games at CHI Play 2014* (2014)