TU WIEN Informatics

Design und Evaluierung innovativer Window Management Interaktionstechniken zur Steuerung mittels Eye-Tracking

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medieninformatik und Visual Computing

eingereicht von

Markus Podskubka, BSc
Matrikelnummer 00828647

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Thomas Grechenig
Mitwirkung: Christoph Wimmer

Wien, 14. Oktober 2021

Unterschrift Verfasser

Unterschrift Betreuung

TU WIEN Informatics

Design und Evaluierung innovativer Window Management Interaktionstechniken zur Steuerung mittels Eye-Tracking

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Media Informatics and Visual Computing

by

Markus Podskubka, BSc
Registration Number 00828647

to the Faculty of Informatics

at the TU Wien

Advisor: Thomas Grechenig
Assistance: Christoph Wimmer

Vienna, 14th October, 2021

Signature Author

Signature Advisor

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Design und Evaluierung innovativer Window Management Interaktionstechniken zur Steuerung mittels Eye-Tracking

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medieninformatik und Visual Computing

eingereicht von

Markus Podskubka, BSc
Matrikelnummer 00828647

ausgeführt am 14. Oktober 2021
Institut für Information Systems Engineering
Forschungsbereich Business Informatics
Forschungsgruppe Industrielle Software
der Fakultät für Informatik der Technischen Universität Wien

Betreuung: Thomas Grechenig

Wien, 14. Oktober 2021

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Markus Podskubka, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 14. Oktober 2021

Markus Podskubka

Danksagung

An dieser Stelle geht ein besonderer Dank an meine Lebensgefährtin Bettina und meine Tochter Hannah, die sehr viel Geduld und Liebe aufbrachten und mir dadurch Kraft und Motivation schenkten.

Des Weiteren bedanke ich mich bei meinen Eltern Anton und Annemarie, die mich stets unterstützt und mir mein Studium ermöglicht haben. Auch meinen Geschwistern Cornelia und Christian danke ich für ihre moralische Unterstützung.

Ebenso möchte ich mich bei meinem Betreuer Thomas Grechenig für die Betreuung meiner Diplomarbeit bedanken. Weiters danke ich Christoph Wimmer, der mir während der gesamten Arbeit über stets mit seinem Rat und seiner Unterstützung zu Seite stand.

Auch allen freiwilligen Testpersonen, die sich die Zeit genommen haben an der Evaluierung teilzunehmen, gilt mein persönlicher Dank.

Kurzfassung

In jedem Betriebssystem mit grafischer Oberfläche sind heutzutage Window Management Systeme (WMS) nicht mehr wegzudenken und werden als selbstverständlich wahrgenommen. Diese Konzepte wurden über viele Jahre hinweg für die Maus- und Tastatursteuerung verbessert und dadurch immer effizienter und flexibler. Ein Eingabegerät, welches in den letzten Jahren in den kommerziellen Bereich Einzug findet, ist der Eye-Tracker, mit welchem der von einem/einer BenutzerIn betrachtete Bereich am Bildschirm registriert und weiterverarbeitet werden kann.
Eye-Tracker als Eingabegeräte zur Steuerung eines Desktopcomputers sind ein vielversprechendes Forschungsfeld, weil es verglichen mit denen für die Maus und Tastatur noch sehr wenige Interaktionstechniken dafür gibt.
In dieser Diplomarbeit wurden fünf innovative Interaktionskonzepte entworfen, prototypisch implementiert und im Rahmen einer Benutzerstudie evaluiert. Der Prototyp beinhaltet die folgenden Konzepte: Anwendung wechseln und fokussieren, vergrößern und verkleinern von Inhalten, navigieren auf einer digitalen Landkarte, zwei Anwendungen nebeneinander darstellen, verschieben einer geöffneten Anwendung vom primären auf einen sekundären externen Bildschirm. Für die Evaluierung dieser Konzepte wurde für alle Aufgaben die Fehlerrate (Effektivität) und die Ausführungszeit (Effizienz) zwischen der Eye-Tracking Steuerung und der etablierten Maus-/Tastatursteuerung verglichen. Es nahmen zehn Personen an der Evaluierung teil und jeder/jede TeilnehmerIn führte jede Aufgabe für beide Eingabearten durch (Within-Subject Design).
Das Ergebnis der statistischen Auswertung des Splitscreen Konzepts zeigt, dass sowohl bei der Fehlerrate als auch bei der Ausführungszeit kein signifikanter Unterschied zwischen der Eye-Tracking Steuerung und der Steuerung mittels Maus-/Tastatur existiert, somit wurde ein gleichwertiges Konzept gefunden. Alle anderen evaluierten Konzepte sind der Maus-/Tastatursteuerung in der Gesamtbewertung unterlegen, haben aber trotzdem gute Vorraussetzungen für zukünftige Forschungsarbeiten. Im direkten Vergleich liegt das System Usability Scale (SUS) Ergebnis (Zufriedenheit) der Maus- und Tastatursteuerung vorne, dennoch erreichte man für die evaluierten Eye-Tracking Konzepte ein SUS Ergebnis von 70,5 (Maus-/Tastatur: 84). Gesamt betrachtet zeigen die Ergebnisse der Evaluierung, dass Eye-Tracking Interaktionskonzepte durchaus Potential haben bestehende Konzepte der Maus-/Tastatursteuerung zu ergänzen oder sogar zu ersetzen.

Keywords: *Eye Tracking, WMS, Interaktionskonzepte, Human Computer Interaction (HCI), Assistive Technologie, Usability Evaluierung*

Abstract

Nowadays, no operating system with a graphical user interface can be imagined without WMS and is taken for granted. The concepts for mouse and keyboard control were improved over many years and have therefore become more and more efficient and flexible. An input device that has found its way into the commercial sector in recent years is the eye tracker, with which the area on the screen viewed by a user can be registered and further processed.
Eye trackers as input devices for controlling a desktop computer are a very promising field of research, as there are very few interaction techniques for them compared to those for the mouse and keyboard.
In this diploma thesis, five interaction concepts were designed, implemented as prototypes and evaluated in a user study. The prototype includes the following concepts: changing applications and focusing, zoom in and zoom out on content, navigating on a digital map, displaying two applications side by side, and moving an open application from the primary to a secondary external screen. To evaluate these concepts, the error rate (effectiveness) and the execution time (efficiency) of the tasks between the eye-tracking control and the established mouse/keyboard control were compared. Ten people took part in the evaluation and each participant performed each task for both input types (Within-Subject Design).
The result of the statistical evaluation of the splitscreen concept shows no significant differences in either the error rate or the execution time between the eye-tracking control and the mouse/keyboard control, thus an equivalent concept was found. All other evaluated concepts are inferior to mouse/keyboard control in the overall evaluation, but still have good prerequisites for future research work. In a direct comparison the SUS score (satisfaction) of the mouse and keyboard control is ahead, but a SUS score of 70,5 (mouse/keyboard: 84) was achieved for the evaluated eye-tracking concepts. Overall the results of the evaluation show that eye-tracking interaction concepts certainly have potential to supplement or even replace existing concepts of mouse/keyboard control.

Keywords: *Eye Tracking, WMS, Interaktionskonzepte, HCI, Assistive Technologie, Usability Evaluierung*

Inhaltsverzeichnis

KAPITEL 1

Einleitung

Den Einstieg in dieses Kapitel bildet die Beschreibung der Problemstellung dieser Arbeit. Darauf folgt das Subkapitel der Motivation und der Zielsetzung. Letzteres beschreibt zum einen die Erarbeitung des Resultats und zum anderen die aufgestellten Hypothesen dieser Diplomarbeit. Danach wird in einem weiteren Unterkapitel der inhaltliche Aufbau der Arbeit beschrieben.

1.1 Problemstellung

Während der Benutzung der heute weit verbreiteten Betriebssysteme wie Linux, macOS oder Windows werden meist mehrere Programme und somit auch deren grafische Oberfläche parallel betrieben. Um einen Überblick all dieser gleichzeitig geöffneten Programme zu behalten, verfügen Desktop- und Mobil-Betriebssysteme, über WMS. Mit Hilfe von WMS, dies erwähnen auch Robertson u. a. [RvDR+00], besteht die Möglichkeit verschiedene User Interface (UI) Elemente wie zum Beispiel Programmfenster, Menüs oder auch Dialoge zu steuern und mit ihnen zu interagieren. Sie dienen dazu die Benutzeroberflächen zu organisieren und strukturieren und unterstützen dadurch den/die BenutzerIn beim Arbeiten mit mehreren Programmen und somit die Produktivität zu steigern. In vielen Desktop-Betriebssystemen haben sich Standard-Controls wie das Schließen, Minimieren, Maximieren, Vergrößern, und Verkleinern eines Programmfensters etabliert. Diese Operationen sind hauptsächlich für die Bedienung mit der Maus und Tastatur konzipiert, woraus man ableiten kann, dass die Bedienung über diese Eingabegeräte am besten funktioniert.

Das Ziel dieser Arbeit ist es, geeignete Interaktionskonzepte, sowohl in „Single-“ als auch „Multi-Screen-Umgebungen“, basierend auf der Augensteuerung eines Desktop Computers als UI Prototyp zu entwickeln und zu untersuchen ob diese von den Benutzern/Benutzerinnen positiv angenommen und akzeptiert werden. Mit Hilfe von „Eye-Tracking“ Hardware

wird festgestellt, welchen Bereich der grafischen Oberfläche der/die BenutzerIn gerade fokussiert beziehungsweise welchem er Aufmerksamkeit schenkt. Dadurch ist es möglich softwaretechnisch entsprechend zu reagieren und somit innovative Konzepte in Bezug auf die einfache und benutzerfreundliche Bedienung des Systems zu konzipieren. Die Entwicklung des UI Prototypen ist insofern eine Herausforderung, da die heutigen WMS Controls hauptsächlich für die Steuerung über Maus und Tastatur als Eingabegeräte entwickelt wurden. Diese sind somit nicht direkt auf die Augensteuerung übertragbar, weshalb dafür neue Konzepte entwickelt beziehungsweise bestehende adaptiert werden müssen. Diese Konzepte sollen uns beim Arbeiten mit dem Computer unabhängig vom Einsatzgebiet unterstützen. Bezüglich der Zielgruppe soll es keine spezifischen Einschränkungen geben. Primär konzentrieren sich die Konzepte auf den Einsatz für Desktop-Umgebungen. Dazu soll der Prototyp nicht direkt in einem Betriebssystem integriert werden, sondern in einer Sandbox [Ste11]. Hierfür wird ein eigenständiges Programm entwickelt, welches die Funktionalität eines Window Management Systems simuliert.

1.2 Motivation

Eye-Tracking ermöglicht jenen Bereich am Bildschirm zu erfassen, welcher gerade vom/von Benutzer/der Benutzerin mit dessen Augen fokussiert wird. Diese Informationen werden als Daten für unterschiedlichste Zwecke verwendet. Zum Beispiel zum Aufbau einer Heatmap um die am häufigsten betrachteten Bereiche zu erfassen, oder die Daten werden verwendet um bestimmte Aktionen auszulösen wenn ein bestimmter Bereich betrachtet wird.
Bisher waren Eye-Tracker für die breite Masse zu hochpreisig, was einer der Hauptgründe war warum noch sehr wenig kommerzieller Einsatz im Desktopbereich Einzug gefunden hat. Doch in den letzten fünf Jahren wurde Eye-Tracking für die EndverbraucherInnen billiger und dadurch populärer. Der Eye-Tracker Hersteller Tobii [tob20] hat diesen Markt durch die leistbaren Eye-Tracker (Tobii Eye-Tracker 5 [tob21c], 229 Euro) maßgeblich vorangetrieben. Dies hat die Spieleindustrie [tob20] erkannt und durch die Integration von Eye-Tracking Steuerungskonzepten einen erheblichen Beitrag dazu geleistet.
Auch Microsoft bietet in ihrem Betriebssystem Windows 10 bereits Eye-Tracking als alternative Steuerung des Computers an. Die angebotenen Steuerkonzepte sind jedoch sehr auf die bekannten Eingabegeräte wie Maus und Tastatur ausgelegt. Für die Maus und Tastatur gibt es bereits diverse Interaktionskonzepte. Für die Maus wäre das zum Beispiel der Bildlauf hinauf und hinunter, ein Klick auf die linke und rechte Maustaste, oder ein Klick auf das Mausrad. Microsoft bietet nun unter anderem an, genau diese Funktionen über Eye-Tracking zu steuern. Es sind also keine Alternativen sondern nur ersetzende Konzepte. Mit den in dieser Arbeit entwickelten Interaktionskonzepten soll das Interesse durch das Aufzeigen der Einsatzmöglichkeiten von Eye-Tracking geweckt werden und einen Anreiz für zukünftige Forschungsarbeiten geschaffen werden.

1.3 Zielsetzung

Das Ergebnis dieser Arbeit soll eine Aussage darüber sein, ob geeignete Interaktionskonzepte basierend auf Eye-Tracking gefunden werden konnten. Dies bedeutet, dass der/die BenutzerIn die Konzepte versteht und akzeptiert. Ebenso sollen Kenntnisse im „Multi-Screen“ Bereich gewonnen werden, welche Auskunft darüber geben sollen, ob auch dort der Einsatz von „Eye Trackern“ möglich ist. Durch die Evaluierung der entwickelten Konzepte soll außerdem die Umgangsweise der BenutzerInnen mit den neuen Konzepten Aufschluss zu einer positiven oder negativen Beeinflussung der Bedienung eines Computers geben. Ein Zeitvergleich soll zeigen wie lange die BenutzerInnen im Vergleich zur Bedienung mit der Maus benötigen um bestimmte Funktionen auszulösen. Des Weiteren soll die Fehlerrate bei der Durchführung der Aufgaben betrachtet werden. Nachdem die Interaktionskonzepte dieser Arbeit evaluiert wurden kann festgestellt werden, welche Konzepte alltagstauglich sind und welche nicht. Dabei sollen die Konzepte für keine spezifische Zielgruppe entwickelt werden sondern die Allgemeinheit ansprechen. Ein Interview nach den Evaluierungen soll qualitative Ergebnisse im Hinblick auf die entwickelten Interaktionskonzepte liefern. Die Testpersonen werden außerdem nach ihrer allgemeinen Meinung bezüglich der Eye-Tracking Technologie befragt.
Die programmatisch entwickelten Prototypen können als Vorlage für weitere Studien dienen und unterstützen zukünftige Entwicklungen im Bereich der augenbasierten Steuerung von Computern.
Falls diese Studie ihr Ziel der Gestaltung geeigneter Konzepte nicht erreicht bedeutet dies nicht, dass der generelle Ansatz nicht funktioniert. Es kann auch andere Gründe für das Nichterreichen des Ziels geben zum Beispiel:

- unverständliche UI Konzepte für den/die BenutzerIn
- Ablehnung der Konzepte durch den/die BenutzerIn
- sowie technische Einschränkungen

In diesem Fall soll diese Arbeit als Vorarbeit dienen und die bereits erworbenen Erkenntnisse für zukünftige Forschungsarbeiten zur Verfügung stellen.
Des Weiteren werden alle entwickelten Konzepte ausschließlich für den Einsatz auf Desktopcomputern (PCs, Laptops) gestaltet. Diverse Ausprägungen der Interaktionsideen in zukünftigen Arbeiten sind jedoch ohne weiteres denkbar.

1.3.1 Methodik

Um das beschriebene Ziel zu erfüllen, setzt sich die vorliegende Diplomarbeit aus einem Theorie- und Praxisteil zusammen.

- **Theorie Schwerpunkte**
  Um das zuvor beschriebene Resultat zu erreichen, wurde in dieser Diplomarbeit

eine empirische Evaluationsforschung durchgeführt. Zu Beginn dieser Diplomarbeit wurde eine umfassende Recherche bezüglich bestehender WMS, sowie Systeme und Forschungsarbeiten zu Eye-Tracking als Eingabeart durchgeführt. Dadurch entstand ein Überblick der bereits vorhandenen Konzepte und Einsatzgebiete. Nach der intensiven Auseinandersetzung mit den Ergebnissen der vorangegangenen Recherche, existierte ein klares Bild über den aktuellen Stand der Forschung. Die Analyse des „State of the Art“ brachte des Weiteren eine Aufstellung der Vor- und Nachteile der bereits eingesetzten Technologien hervor.

- **Praxis Schwerpunkte**
  Zuerst sollen diverse UI Interaktionsideen gefunden und daraus eine Auswahl getroffen werden. Anstatt die Konzepte in ein bestehendes Window Management eines Betriebssystems zu integrieren und dadurch eine Einschränkung der möglichen Interaktionen zu riskieren, sollen die Interaktionskonzepte in Form eines Prototypen entwickelt werden. Die UI Konzepte werden als Szenarien, nach Nielsen [Nie94] abgebildet, wobei Nielsen hier die Reduzierung der Funktionalität des Prototyps als horizontalen Prototyp und die Umsetzung einer minimalen Anzahl an Konzepten als vertikalen Prototyp bezeichnet. Ein Szenario ist nach Nielsen [Nie94] eine Kombination dieser Prototypen. Mit Hilfe dieser Szenarien kann anschließend eine Evaluierung der Steuerungskonzepte bezogen auf die Benutzbarkeit und Sinnhaftigkeit von Eye-Tracking Systemen zur Unterstützung des Window Management im Alltag durchgeführt werden.

1.3.2 Hypothesen

Am Ende dieser Arbeit sollen die in diesem Subkapitel aufgeführten Hypothesen entweder widerlegt oder bestätigt werden.

- *H1: Übungsrunden führen zu einem Lerneffekt in Form schnellerer Task-Completion Time*
- *H2: Es gibt signifikante Unterschiede zwischen der Steuerung über Eye-Tracking und der Maus-/Tastatursteuerung*
  - *H2.1: Es gibt einen signifikanten Unterschied in der Fehlerrate im Vergleich zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung*
  - *H2.2: Es gibt einen signifikanten Unterschied in der Durchführungszeit im Vergleich zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung*
- *H3: Augensteuerung erhöht die Usability (SUS Ergebnis)*

1.4 Aufbau der Arbeit

Das Kapitel 2 behandelt sowohl Grundlagen als auch Definitionen und bietet einen Überblick der Entwicklung von Eye-Tracking sowie verschiedene Techniken dazu. Ebenfalls

wird auf die für diese Diplomarbeit wichtigsten Entwicklungen der Vergangenheit zum Thema Window Management eingegangen. Des Weiteren wird in diesem Kapitel der aktuelle Stand der Technik betrachtet. Am Ende der Grundlagen werden noch allgemeine Informationen der in dieser Diplomarbeit eingesetzten Technologien und Programmiersprachen näher beschrieben.

Anschließend werden im Kapitel 3 einige verwandte Arbeiten beschrieben und kurz erklärt. Das Kapitel wurde in weitere Unterkapitel unterteilt, da die Arbeiten jeweils unterschiedliche Ansätze haben den Eye-Tracker als Eingabegerät zur Steuerung eines Computers einzusetzen.

Alle aufgestellten Hypothesen sowie der relevanten Variablen werden in Kapitel 4 erläutert. Darüber hinaus werden die erarbeiteten Interaktionskonzepte für Single- und Multi-Screen vorgestellt. Die Konzepte werden dabei in alternative und neue Operationen im Vergleich zu bestehenden unterteilt.
Auch die technische Umsetzung und die vorangegangenen Überlegungen, sind in Kapitel 4 dargestellt. Weiters werden in diesem Kapitel die Komponenten des Prototypen als auch Details über die C# Windows Presentation Foundation (WPF) Anwendung näher betrachtet.

Im Kapitel 5 wird auf die Studienplanung der in dieser Arbeit durchgeführten Evaluierung eingegangen. Dabei wird zu Beginn dieses Kapitels die genaue Zielsetzung beschrieben. Danach folgt die Erläuterung des gewählten Studiendesigns welche die ausgearbeiteten Aufgaben für die Durchführung der Evaluierung beinhaltet. Das Unterkapitel des Setups beschäftigt sich mit den Vorbereitungen und der Infrastruktur für die Evaluierung. Der genaue Vorgang für die Ausübung der Evaluierung wird im Unterkapitel 5.4 charakterisiert.

Die Ergebnisse der Evaluierung werden in Kapitel 6 präsentiert. Im ersten Teil werden die Antworten des demografischen Pre-Testfragebogens ausgewertet und beschrieben. Danach folgt ein Subkapitel in jenem die Ergebnisse der Evaluierung des Prototypen analysiert werden. Dazu wird die Effektivität mit Hilfe der Fehlerrate, die Effizienz anhand der Durchführungszeiten sowie Zufriedenheit durch die Auswertung des SUS [Bro96] Fragebogens bewertet. Am Ende dieses Kapitels werden qualitative Ergebnisse durch die Auswertung der geführten Interviews analysiert.

Den Abschluss dieser Arbeit bildet das Kapitel 7. In diesem Kapitel werden die Ergebnisse dieser Arbeit diskutiert und interpretiert.

KAPITEL 2

Grundlagen

In diesem Kapitel wird zu Beginn die geschichtliche Entwicklung und die unterschiedlichen Techniken des Eye-Trackings erläutert. Danach folgt ein Unterkapitel in welchem auf die ersten Entwicklungen im Window Management Bereich während der siebziger und achtziger Jahre eingegangen wird. Des Weiteren wird auf den aktuellen Stand der Technik, bezogen auf Interaktionskonzepte im Window Management Bereich, näher eingegangen. Ein weiteres Unterkapitel beschäftigt sich mit der in dieser Diplomarbeit eingesetzten Hard- und Softwarekomponenten.

2.1 Eye-Tracking Grundlagen

Der Begriff „Eye-Tracking“ bezeichnet im wesentlichen das Aufzeichnen von Augenbewegungen. Dabei kann man laut Duchowski [Duc17] zwischen zwei Bezugspunkten für die Messung bei Eye-Tracking Systemen unterscheiden:

- Augenposition relativ zum Kopf
  Wie in [Duc17] beschrieben zählen hierzu Techniken wie die Sklerallinse mit Suchspule, oder auch die Elektro-Okulografie. Im Unterkapitel 2.1.2 wird auf diese Messtechniken näher eingegangen.

- Augenposition relativ zum Raum („Point Of Regard“ - POR)
  Duchowski [Duc17] beschreibt, dass man damit bei Mensch-Computer Interaktionen die Blickrichtung des/der Benutzers/Benutzerin auf einen bestimmten Bereich des Bildschirms meint. Weiters erwähnt er, dass bei den meisten videobasierten Systemen, kombiniert mit Hornhautreflexion, diese Messung zum Einsatz kommt.

2.1.1 Geschichte

Die Geschichte des Eye-Trackings wurde bereits ausgiebig von Jacob und Karn [JK03] analysiert. In diesem Unterkapitel wurden alle, für diese Arbeit relevanten Informationen, kurz zusammengefasst.

Schon gegen Ende des neunzehnten Jahrhunderts gab es die ersten Eye Tracker. Diese waren, wie Holmqvist [HNA+11] beschreibt, jedoch sehr komplex herzustellen und nicht sehr angenehm zu benutzen.

Edmund Huey, so schreiben Holmqvist [HNA+11], benutzte 1898 zum ersten Mal ein Beißbrett um den Kopf zu fixieren. Lupu und Ungureanu [LU13] berichten, dass Delabarre 1898 eine Art Kontaktlinse mit einem Loch in der Mitte („Paris Ring“) direkt auf die Hornhaut legte. Dieser Ring wurde, wie in [YS75] beschrieben, mechanisch mit Aufzeichnungsstiften verbunden.

Weiters erwähnt Jacob und Karn [JK03], dass sich Dodge und Cline drei Jahre später (1901) die Hornhautreflexion zu nutze machten und somit die erste, für Testpersonen zumutbare, Eye-Tracking Technik schufen. Dabei durften die Probanden ebenfalls den Kopf nicht bewegen. Es konnten horizontale Bewegungen mit Hilfe einer fotografischen Platte aufgezeichnet werden [JK03].

Nach den Angaben von Jacob und Karn [JK03] haben Judd, McAllister und Steel im Jahr 1905 sowohl horizontale als auch vertikale Augenbewegungen aufgezeichnet. Anders als zuvor Dodge und Cline, verwendeten sie nicht die Augenhornhautreflexion, sondern verwendeten ein weißes Material um das Auge zu markieren und dessen Bewegung so nachzuverfolgen [JK03]. Für die Aufzeichnungen setzten sie laut Jacob und Karn [JK03] die Technik der Fotografie von bewegten Bildern ein.

Ab diesem Zeitpunkt wurde Eye-Tracking für die Forschung immer interessanter und es wurde, wie Jacob und Karn [JK03] beschreiben, unter anderem die Hornhautreflexion mit der Technologie von bewegten Bildern kombiniert. Miles Tinker analysierte zum Beispiel mit der Aufzeichnung von Augenbewegungen das Leseverhalten bezogen auf verschiedene Merkmale wie Schriftgröße, Schriftart, Layout und viele mehr.

Gegen Ende der vierziger Jahre (1947), wie in [JK03] festgehalten, führten Paul Fitts und seine Kollegen eine Art „Usability Studie“ durch. Dabei zeichneten sie die Augenbewegungen von Piloten in einem Cockpit auf, um festzustellen welchen Bereichen am meisten Aufmerksamkeit geschenkt wird.

Ungefähr ein Jahr später entwickelten, Hartridge und Thompson den ersten Eye Tracker, welcher fix am Kopf befestigt ist. Dies bedeutete, dass Kopfbewegungen keinen Einfluss mehr hatten [JK03].

Einer der ersten Erfolge im Verfolgen von Augenbewegungen kann in das Jahr 1963 datiert werden. In diesem Jahr konnte Robinson [Rob63] mit Hilfe einer „Sklerallinse“ die Bewegung der Augen in horizontaler und vertikaler Richtung beziehungsweise Rollbewegungen feststellen. Kurz danach, nur drei Jahre später (1966) machten Fuchs und

Robinson [FR66] Versuche mit einem Affen. Sie implantierten ihm dabei die Spule direkt, wodurch längere Versuche und Aufzeichnungen ermöglicht wurden.

Lupu und Ungureanu [LU13] beschreiben, dass in den achtziger Jahren das Aufkommen der Personal Computer die Forschung im Eye-Tracking Bereich vorantrieben. Jacob und Karn [JK03] berichten weiters, dass in dieser Zeit die Forscher Eye-Tracking immer mehr für die Untersuchung der Mensch-Computer Interaktion nutzten.

Auch heute noch wird Eye-Tracking unter anderem als Werkzeug für „Usability Engineering“ eingesetzt.

2.1.2 Techniken

In diesem Unterkapitel werden Techniken zur Erfassung der Augenbewegungen beschrieben und durch Illustrationen dargestellt.

Sklerallinse mit Suchspule

Die Sklerallinse mit Suchspule, welche in Abbildung 2.1 zu sehen ist, ist eine Kontaktlinse welche am Rand der Linse zwei separate Drahtspulen eingearbeitet hat [Rob63]. Bei dieser Technik wird, wie Robinson [Rob63] beschreibt, ein Magnetfeld erzeugt, welches von der Linse, und somit von den Augenbewegungen der Testperson, beeinflusst wird. Dadurch können Bewegungen der Augen gemessen werden.

Damit diese Variante funktioniert muss der Kopf fixiert werden um die Messung der Augenbewegungen nicht zu beeinflussen. Diese Fixierung geschieht meist mit einer Kinnstütze, wobei meist zusätzlich ein Beißbrett verwendet wird.

Abbildung 2.1: 3D Sklerallinse mit Spule, für horizontale, vertikale und torsionale Augenbewegungsmessungen (Quelle: Chronos Vision [chr20]).

Okulografie

- *EOG - Elektro-Okulografie*
  Bei dieser Messtechnik werden, wie Young und Sheena [YS75] in ihrer Arbeit beschreiben, Hautelektroden rund um das Auge platziert und die elektrischen

Strömungsunterschiede gemessen (Abb. 2.2). Kaufman, Bandopadhay und Shaviv [KBS93] entwickelten 1993 in ihrer Arbeit eine Apparatur welche auf der Elektro-Okulografie aufbaut. Sie entwickelten zusätzlich eine Benutzeroberfläche, welche mit dieser Hardware über die Augen gesteuert werden kann.

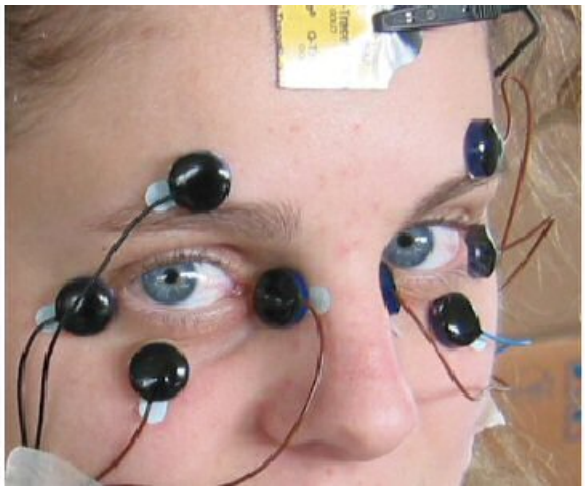


Abbildung 2.2: Elektro Okulografie (Quelle: Metrovision [met20]).

- *POG - Foto-Okulografie*
  Die Technik der Foto-Okulografie beschreibt die Analyse der Augenbewegungen durch das Fotografieren der Augen. Anschließend werden die Fotos manuell oder mit einem Bildverarbeitungsprogramm analysiert. Unter diesem Begriff werden alle Eye-Tracking Techniken zusammengefasst, welche mit „unterscheidbaren Merkmalen des Auges“ [Duc17] arbeiten.

- *VOG - Video-Okulografie*
  Bei der Video-Okulografie wird mit einer oder mehreren Kameras das Auge gefilmt. Wie Wassill und Kaufmann [WK00] beschreiben, wird die Verarbeitung der Einzelbilder von einem Bildverarbeitungsalgorithmus übernommen.

Videobasierte Pupillen- und Hornhautreflexion

Um festzustellen welchen Bereich der/die BenutzerIn am Bildschirm fokussiert, werden meist Eye Tracker, welche das POR Prinzip (siehe 2.1) verfolgen, eingesetzt. Wie auch Duchowski [Duc17] erwähnt, sind diese Geräte videobasiert und arbeiten dabei mit Kameras und einer Lichtquelle, welche meist Infrarot-Licht ist. In Abbildung 2.3 sieht man eine solche Hornhautreflexion bei welcher die Kamera beziehungsweise die Lichtquelle aus verschiedenen Blickwinkeln auf das Auge trifft.
Die neuesten Eye Tracker dieser Art, wie zum Beispiel der „Tobii Eye Tracker 4C“ [tob21b] (Abb. 2.10), haben zusätzlich noch einen eigenen Chip integriert, welcher laut

Tobii die Signale direkt verarbeiten kann und somit die Zentrale Recheneinheit („Central Processing Unit“ - CPU) des Computers selbst entlastet.

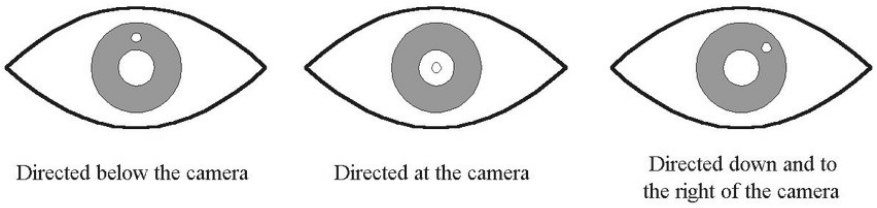


Abbildung 2.3: Hornhautreflexion POR (Quelle: Ball [Bal13]).

2.2 Window Management

Die ersten Entwicklungen von grafischen Oberflächen für Computer reichen mittlerweile schon fast sechzig Jahre zurück. So schreibt auch Tomitsch [Tom03] in seiner Arbeit, dass Ivan Sutherland bereits 1963 ein Programm namens „Sketchpad“ zur zweidimensionalen Interaktion entwickelte. Diese Anwendung hatte maßgeblichen Einfluss auf alle weiteren Entwicklungen grafischer Benutzeroberflächen, so Tomitsch [Tom03].
Ein weiterer Meilenstein wurde durch die Entwicklung eines heute nicht mehr wegzudenkenden Interaktionswerkzeugs gelegt. So konstruierte Douglas Engelbart im Jahr 1963, wie Tomitsch [Tom03] berichtet, das uns heute als Computermaus bekannte Eingabegerät. Wie im Artikel von Barnes [Bar07] beschrieben, entwickelte das Xerox PARC (Palo Alto Research Center) in den siebziger Jahren, im Zuge der Entwicklung der Programmiersprache Smalltalk, eine revolutionäre grafische Benutzeroberfläche. Wie man in Abbildung 2.4 sieht, ermöglichte diese Benutzeroberfläche mehrere geöffnete Fenster beziehungsweise Programme überlappend darzustellen.

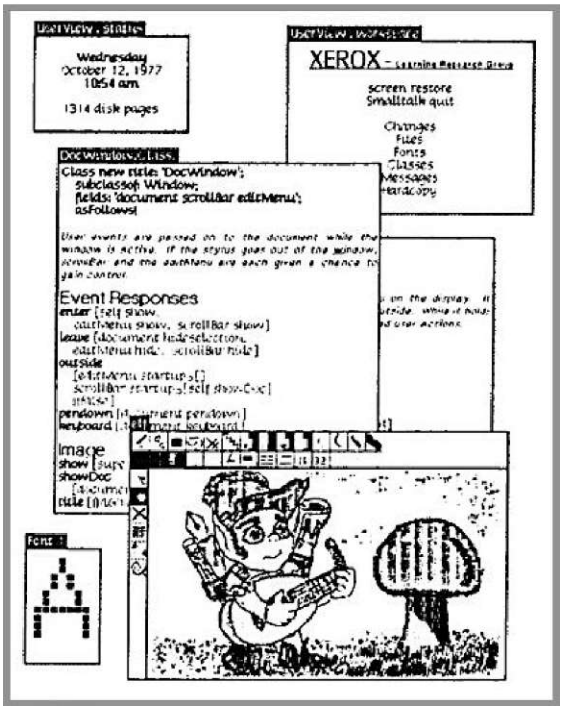


Abbildung 2.4: Smalltalk Benutzeroberfläche mit überlappenden Programmfenster (Quelle: Ingalls [Ing78]).

Wie auch Barnes [Bar07] in ihrem Paper beschreibt, prägten Anfang der achtziger Jahre diese Konzepte von PARC auch die Entwicklung grafischer Oberflächen der ersten Apple Computer (Abb. 2.5).

Tomitsch [Tom03] erwähnt, das auch Microsoft im Jahr 1985 ebenfalls ein grafisches Betriebssystem namens Windows 1.0 veröffentlichte. Die Programmfenster konnte Windows 1.0 zu diesem Zeitpunkt jedoch noch nicht überlappend darstellen. Im Jahr 1987 zog Microsoft mit Xerox und Apple jedoch gleich und veröffentlichte das Betriebssystem Windows 2.0 (Abb. 2.6), bei jenem ebenfalls die Darstellung überlappender Programmfenster ermöglicht wurde [Tom03].

Unter dem Open Source Betriebssystem GNULinux wurde ein etwas anderer Ansatz gewählt. Wie Delozier [Del09] anführt, ist GNULinux ein Betriebssystem ohne Window Manager und somit ohne grafische Oberfläche. Um eine grafische Benutzeroberfläche (Graphical User Interface (GUI)) zu integrieren, wurde, so Tomitsch [Tom03], in den achtziger Jahren das „X Window System“ entwickelt. Alle Grundfunktionalitäten, welche für eine GUI notwendig sind, werden von diesem System bereitgestellt. Somit bauen die mittlerweile unzähligen GNU/Linux Distributionen (z.B. Manjaro, Mint, Ubuntu, Debian, Fedora, etc.) auf diesem System auf, um Oberflächen zu gestalten und zu steuern.

Abbildung 2.5: Apple's Lisa Computer, Veröffentlichung 1983 (Quelle: Encyclopædia Britannica [app20b]).

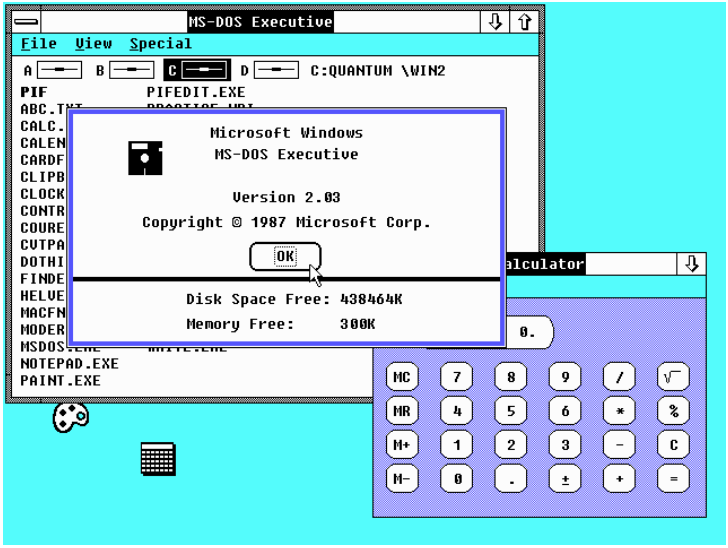


Abbildung 2.6: Microsoft Windows 2.x (Quelle: Department of Computer Science, University of Maryland [ms-20]).

Bis heute charakterisiert die Technologie überlappender Fenster die Window Manager der bekanntesten Desktop Betriebssysteme Microsoft Windows, Apple's macOS sowie diverse Linux Distributionen.
Durch die ständig wachsende Anzahl an verfügbaren beziehungsweise installierten Programmen ergab sich schlussendlich die Problemstellung diese vielen Programmfenster zu organisieren. Bis heute verwendete Techniken sind zum Beispiel die unter Microsoft

Windows bekannte Taskbar am unteren Bildschirmrand (Abbildung 2.7). Diese Taskbar ermöglicht es unter anderem alle als Fenster geöffneten Programme mit einem Klick zu erreichen. Des Weiteren erhält man dadurch einen besseren Überblick der geöffneten Programmfenster.

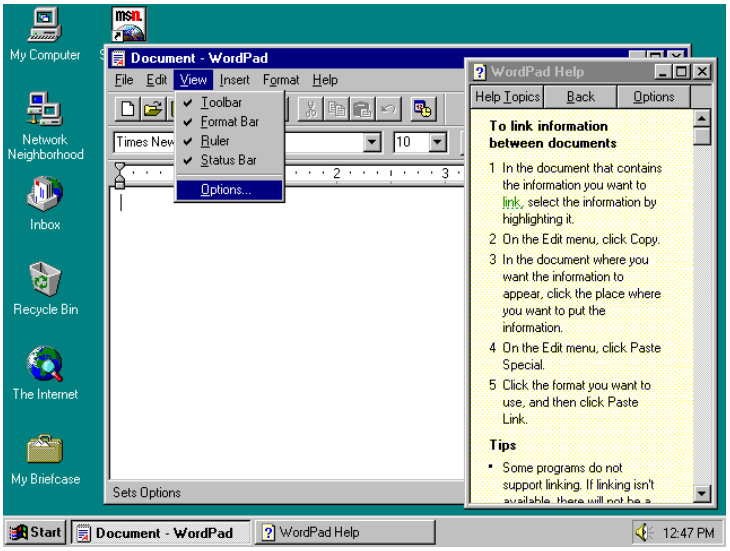


Abbildung 2.7: Microsoft Windows 95 Taskbar (Quelle: Department of Computer Science, University of Maryland [ms-20]).

Im Jahr 2003 veröffentlichte Apple mit ihrer Betriebssystemversion „Mac OS X Panther“ die „Exposé“ (in neueren macOS Versionen „Mission Control“) Funktion [Tom03], welche es ermöglicht alle geöffneten Fenster auf einen Blick zu sehen und so eine Übersicht zu erhalten (Abbildung 2.8).

Die Problematik einen Desktop sowie offene Programmfenster zu organisieren ist bis heute noch immer ein Thema beim Design moderner Benutzeroberflächen von Betriebssystemen. Die in den achziger Jahren vorgestellten Grundkonzepte bezüglich Window Management werden bis heute in teilweise leicht abgewandelter Form (z.B. modernisierte GUI) angewandt. In dieser Diplomarbeit werden deshalb mit Hilfe eines Eye Trackers neue Konzepte der Interaktion und Organisation von Programmfenstern in Form eines Prototypen entwickelt.

Abbildung 2.8: Apple Mac OS X Panther, Exposé/Mission Control Feature (Quelle: Macworld [app20a]).

2.3 Aktueller Stand der Technik

Da im Zuge dieser Diplomarbeit Interaktionskonzepte für den Einsatz von Augensteuerung entwickelt werden, wird in diesem Kapitel der aktuelle Stand der Technik betrachtet. Der Fokus liegt hier auf der Mensch-Computer Interaktion über Eye-Tracking. Es werden sowohl wissenschaftliche Anwendungen als auch industrielle Konzepte beschrieben. Dabei werden die bestehenden Anwendungsfälle in dieser Arbeit in assistive, interaktive und analytische Einsatzgebiete klassifiziert.

2.3.1 Eye Tracker

Die Interaktionskonzepte dieser Arbeit beziehen sich auf die regelmäßige Verwendung im Alltag oder Arbeitsalltag und sprechen somit eine gemeinsame Zielgruppe an. Deshalb listet dieses Unterkapitel Eye Tracker auf, bei denen sich die Anschaffungskosten in Grenzen halten und somit für die breite Masse leistbar bleiben. Auch der hohe Marktanteil [sta20] von Windows Betriebssystemversionen spricht für die Auswahl der aufgeführten Geräte, welche mit Windows 7 und höher kompatibel sind.

Tobii EyeX Tracker

Einer der ersten kommerziellen Eye Tracker für den Massenmarkt von Tobii war der EyeX Tracker. Es handelt sich hierbei um einen peripheren Tracker, der über USB 3.0 mit dem Personal Computer (PC) verbunden und am unteren Rand des Computerbildschirms positioniert wird. Die Bildabtastrate beträgt 70 Hz, und der Interaktionsabstand zum Bildschirm liegt im Bereich zwischen 50 und 90 Zentimeter. Gibaldi, Vanegas und Bex [GVB17] schreiben in ihrem Artikel, dass dieser Eye-Tracker mehr für den kommerziellen Bereich geeignet ist. Für Forschungszwecke bei denen kleinste Augenbewegungen nach-

verfolgt werden müssen ist dieser Eye-Tracker nicht geeignet [GVB17].

Abbildung 2.9: Tobii EyeX Tracker (Quelle: Tobii [tob21d]).

Tobii Eye Tracker 4C

Der Tobii Eye Tracker 4C (Abb. 2.10) ist ein Eye Tracker welcher sowohl in der Unterhaltungsindustrie für Computerspiele als auch für analytische Zwecke zum Einsatz kommt. Ebenso wie sein Vorgänger EyeX, handelt es sich hierbei um einen peripheren Tracker, welcher über USB 2.0 mit dem Computer verbunden ist. Bei 90 Hz liegt die Bildabtastrate etwas höher als bei seinem Vorgänger. Mit Hilfe eines direkt in den Eye Tracker integrierten Schaltkreises, genannt EyeChip, ist es möglich Rechenoperationen vom Computer auf den Chip auszulagern. Dies reduziert den Datenverkehr zwischen Eye Tracker und Computer auf ein Minimum. Derzeit ist es seitens Tobii [tob21a] nicht möglich die Daten mehrerer 4C Eye Tracker, welche alle an einem Computer angeschlossen sind, parallel zu verarbeiten. Das Verfolgen von Kopfbewegungen ist mit diesem Tracker bereits möglich, wobei diese Berechnungen nicht auf dem EyeChip ausgeführt werden und somit die beanspruchte Rechenleistung der CPU erhöhen.

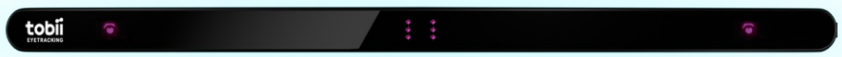


Abbildung 2.10: Tobii Eye Tracker 4C (Quelle: Tobii [tob21d]).

Tobii Eye Tracker 5

Auch einer der neuesten Eye Tracker von Tobii, der Eye Tracker 5 2.11 eignet sich sowohl für die Spiele- und Unterhaltungsindustrie als auch für Forschungs- und Analysezwecke. Die Bildabtastrate von 133 Hz liegt bei diesem Tobii Eye-Tracker noch höher als bei seinem Vorgänger 4C (siehe 2.3.1). Er insgesamt 285 Millimeter breit und somit um 50 Millimeter kleiner als der Eye Tracker 4C.

Tobii [tob21d] beschreibt, dass dieser Eye-Tracker durch einen neuen Algorithmus ein stabileres Tracking der Kopfbewegungen ermöglicht. Tobii [tob21d] verfolgt die Kopfbewegungen indem sie Gesichtspunkte (Abb. 2.12) platzieren und erwähnen dabei, dass es zu

Abbildung 2.11: Tobii Eye Tracker 5 (Quelle: Tobii [tob21c]).

Problemen beim Tracking des Kopfes kommen kann, wenn die Person einen großen Bart hat. Dadurch ist es möglich, dass die Gesichtszüge nicht mehr richtig erkannt werden.

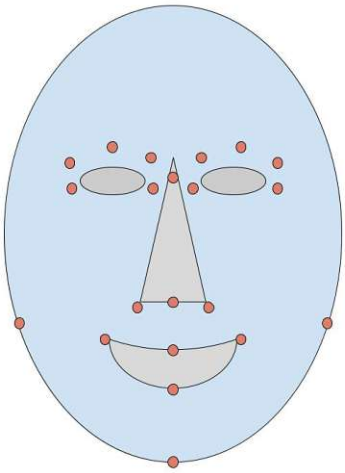


Abbildung 2.12: Gesichtspunkte (siehe Video von Tobii [tob21e]).

Gazepoint GP3 Eye Tracker

Auch das Unternehmen Gazepoint [gaz20] bietet Eye-Tracking Hardware an, unter anderem den GP3 Eye Tracker 2.13 mit einer Bildabtastrate von 60 Hz.

2.3.2 Assistive Konzepte

Hardwaresteuerung

Ein Einsatzgebiet des Eye-Trackers als assistive Technologie findet man im medizinischen Bereich. Dabei sind vor allem Menschen mit körperlichen Einschränkungen wie Lähmungen, defizite in der Feinmotorik oder Sprachproblemen ein großer Teil der Zielgruppe. Cio u. a. [CRMA19] haben in ihrer Arbeit unter anderem einen Eye-Tracker in Kombination mit Kameras benutzt um einen Roboterarm zu steuern.

Abbildung 2.13: Gazepoint GP3 Eye Tracker (Quelle: Tobii).

Dahmani [DCK+20] haben Eye-Tracking in ihrer Arbeit eingesetzt um motorisch eingeschränkte Personen wieder zu mobilisieren. Dazu wurde es ermöglicht einen Rollstuhl nur über die Augenbewegungen zu steuern. Als Eye-Tracker verwendeten sie Infrarotkameras die auf den Rahmen einer Brille befestigt wurden. Diese Kameras registrierten die Bewegungen der Augen und je nach Bewegungsrichtung dieser, bewegte sich der Rollstuhl in die gewünschte Richtung [DCK+20].

Ein weiteres Eye-Tracking Konzept zur Steuerung von Hardware wurde von Bissoli u. a. [BLJS+19] entwickelt. In ihrer Arbeit geht es um die Steuerung verschiedener im Haushalt befindlichen Geräte. Die angesprochene Zielgruppe im Artikel von Bissoli u. a. [BLJS+19] ist auch hier wieder jene der körperlich eingeschränkten Personen. Es wurde eine grafische Oberfläche entwickelt mit deren Hilfe man vernetzte Geräte über Eye-Tracking steuern kann. Die vernetzten Geräte welche gesteuert wurden sind zum Beispiel Lampen, Ventilatoren, Fernseher und Radios.

Softwaresteuerung

Auch Microsoft bietet in ihrem Betriebssystem Windows die Möglichkeit den Computer über die Augenbewegungen zu steuern. In Abbildung 2.14 sieht man eine Leiste, genannt „Launchpad“, mit großen Symbolen. Diese Schaltflächen dienen dazu die unterschiedlichen Interaktionen anzusteuern. Da diese Interaktionen alle nur eine bereits bestehende Funktionalität ersetzen wurde das Launchpad in dieser Arbeit als ein assistives Konzept klassifiziert.

Mit einem Blick auf eines der Symbole aktiviert man nach kurzer Verweilzeit die betrachtete Aktion. Über einige Schaltflächen hat man direkten Zugriff auf bestimmte Bereiche, wie zum Beispiel auf das Startmenü oder aktive Anwendungen. Zusätzlich gibt es Schaltflächen mit welchen man zu Benutzeroberflächen gelangt, welche die Steuerung der bekannten Eingabegeräte, Maus und Tastatur, erlaubt. So sind unter anderem alle Steuerungsmöglichkeiten (Links-, Rechts- und Doppelklick, Laufrad, Mauszeiger steuern)

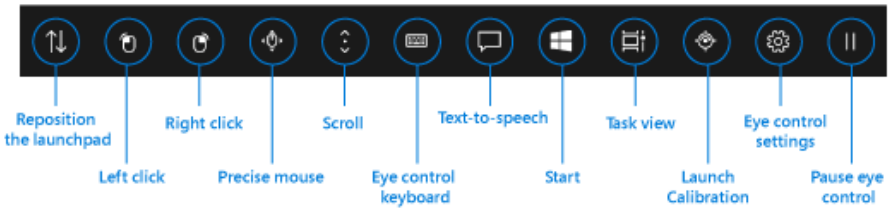


Abbildung 2.14: Windows 10 Eye Control Launchpad (Quelle: Microsoft [mic20]).

einer Computermaus sowie einer Bildschirmtastatur für die Augensteuerung bereitgestellt. Microsoft setzt auf Bedienungskonzepte, welche bereits in Windows 10 existieren (Start Knopf/Windows Symbol, Text-zu-Sprache, Programmübersicht). Die am meisten von diesen Konzepten angesprochenen Personen sind Menschen mit eingeschränkter Motorik. Auch die Sprachsteuerung, die ebenfalls aufgerufen werden kann, wird großteils von einer kleineren Zielgruppe, wie zum Beispiel Menschen mit Behinderung, genutzt.

2.3.3 Interaktive Konzepte

Computerspiele

Ein weiterer Sektor in dem Eye-Tracking bereits als interaktives Konzept Fuß gefasst hat ist die Computerspieleindustrie. Wie Antunes und Santana [AS18] beschreiben, werden die Augenbewegungsdaten sowohl für das Ermitteln des vom Spielcharakter betrachteten Bereichs, als auch für das Erzeugen von Inhalten („Rendering“) je nach Blickrichtung verwendet.
Bei Videospielen spricht man auch oft von visueller Aufmerksamkeit, womit jener Bereich gemeint ist, dem der Spieler gerade seine Aufmerksamkeit schenkt und von ihm fokussiert wird. Die folgenden Informationen über diverse Spiele wurden von Tobii's Videospiel Webseite [tob20] entnommen. In einigen Spielen wird mit Hilfe von Eye Trackern dynamisch die Lichtquelle auf den fokusierten Punkt ausgerichtet (Assassin's Creed® Odyssey). In „Ego-Shooter“ Spielen, kommen noch weitere Konzepte wie zum Beispiel das Anvisieren von Zielen über die Blickrichtung, oder das „in Deckung gehen“ bei Gefahr zum Einsatz (Tom Clancy's The Division® 2). Auch die Möglichkeit ein Objekt zu markieren findet in diversen Videospielarten seine Verwendung. Das Steuern der Kameras über die Augen beziehungsweise über die Kopfbewegungen ist ebenso ein oft verwendetes Konzept. Durch das Wegblicken vom Bildschirm wird in manchen Spielen das Pausieren des Spiels ausgelöst (Assassin's Creed® Rogue).

Mauszeiger

Folgende zwei Arbeiten zählen unter anderem zu den Anwendungen der Mauszeigersteuerung und wurden im Kapitel 3 bereits näher beschrieben. Sidorakis, Koulieris und Mania [SKM15] entwickelten in ihrer Arbeit 3D Benutzeroberflächen, welche über

die Augensteuerung bedient werden. Ein Konzept dabei war, den Mauszeiger mit den Augenbewegungen zu steuern.

Kim, Suh und Lee [KSL17] machten sich Eye-Tracking in Kombination mit einem Handgesten-Controller zu nutze. Dabei wird der Mauszeiger mit dem Eye-Tracker und für die präzise Positionierung mittels Handgesten gesteuert.

Virtual Reality

In Verbindung mit Head Mounted Displays und Virtual Reality gibt es einen weiteren Bereich in dem Eye-Tracking genutzt wird. Marwecki u. a. beschreiben in [MWO+19] mehrere Konzepte in ihrem „Mise-Unseen software system". Das Auge des/der BenutzerIn hat zu jedem Zeitpunkt immer nur den fokussierten Bereich scharf gestellt. Die Umgebung rund um den fokussierten Bereich werden im Gesichtsfeld unbewusst wahrgenommen. Bei einem Computerspiel werden so zum Beispiel bestimmte Ereignisse, die ohnehin in der Peripherie stattfinden, in reduzierter Qualität dargestellt. Dies sind zum Beispiel bewegte Objekte wie eine Explosion. Dafür benötigt der Computer viel Rechenleistung weshalb die Grafik länger braucht sich aufzubauen. Diese Ereignisse werden dann in geringerer Qualität dargestellt, bis der/die BenutzerIn den Bereich des Ereignisses wieder fokussiert. Statische Bereiche werden beispielsweise nur weich gezeichnet.

2.3.4 Analytische Anwendungsgebiete

Usability Engineering

Sehr häufig wird Eye-Tracking eingesetzt um die Usability einer Anwendung zu analysieren. Quimbita, Pupiales und Guerrero [QPG20] haben so zum Beispiel die Facebook Seite einer Universität evaluiert um herauszufinden wie hoch die Usability dieser Seite ist. Durch den Einsatz des Eye-Trackers konnte eine Heatmap erzeugt werden, welche die intensiv und weniger intensiv betrachteten Bereiche anzeigt [QPG20].

Um Usability Probleme einer e-learning Plattform zu entdecken, untersuchten Maslov und Nikou [MN20] unter anderem mit Hilfe von Eye-Tracking. Auch hier wurde eine Heatmap erstellt um Probleme und Komplexität bei der Steuerung der Plattform zu erkennen [MN20].

Marketing

In der Wissenschaft wird Eye-Tracking of genutzt, um herauszufinden welche Bereiche am Bildschirm am häufigsten betrachtet werden. Wedel und Pieters [WP17] beschreibt unter anderem den Einsatz von Eye-Tracking für Marketing Zwecke. Dabei wird zum Beispiel analysiert welche Bereiche/Teile einen/eine KonsumentIn bei Werbungen oder Produkten besonders anziehen.

Medizin

Auch andere Forschungsgebiete in der Wissenschaft nutzen Eye-Tracking, wie zum Beispiel das Gebiet der Entwicklungspsychologie. Boxhoorn u. a. [BBS+20] konnten unter Zuhilfenahme der Eye-Tracking Technologie in ihrer Arbeit für bestimmte Situationen einen Unterschied der Pupillenerweiterung zwischen Menschen mit einer Aufmerksamkeits-Defizit-Hyperaktivitäts-Störung (ADHS) und Personen mit einer Autismus-Spektrum-Störung feststellen.

Weitere

Die Internetseite von Tobii Pro [tob21a] beschreibt auch noch einige andere Anwendungsfälle in der Forschung, wie zum Beispiel Verpackungstests, Werbewirkungsforschung, sowie Unterstützung bei der Analyse und Beseitigung unnötiger Arbeitsgriffe.

2.4 Die Programmiersprache C#

Die Wahl der Programmiersprache, welche für die Erstellung des Prototypen verwendet wird, fiel auf Grund der vorhandenen Programmierschnittstelle zum Tobii Eye Tracker 4C auf die Programmiersprache C#. Aus diesem Grund wird diese Sprache und dessen Technologien zur Erstellung von Benutzeroberflächen folgend kurz vorgestellt.

2.4.1 .NET

Bei .NET handelt es sich um eine von Microsoft 2002 ins Leben gerufene Plattform, die von diversen Programmiersprachen (.NET Sprachen) genutzt wird, um in eine Zwischensprache (Microsoft Intermediate Language (MSIL)) übersetzt zu werden [Kot19]. Wie auch Kotz [Kot19] in seinem Buch erwähnt, hat dieses Vorgehen eine Ähnlichkeit zu Java, wo ebenfalls der Code in einen Zwischencode kompiliert wird.

2.4.2 Allgemeines zu C#

Eine dieser auf .NET basierten Sprachen ist C# (gesprochen: „see sharp“), bei dessen Konzeption laut Kotz [Kot19] vor allem darauf geachtet wurde, die besten Konzepte aus bekannten Sprachen zu vereinen. Mit der objektorientierten Programmiersprache C# ist es unter anderem möglich einfache Konsolenprogramme zu erstellen, oder Client-Server-, Desktop-, Mobil- und Webanwendungen zu entwickeln [Gri19]. Des Weiteren handelt es sich hierbei um eine typsichere Sprache. Griffiths [Gri19] erklärt in seinem Buch, dass C# syntaktisch sehr ähnlich zu den Sprachen C und C++ ist. Ohne Zweifel lassen sich auch Sprachfeatures von anderen objektorientierten Programmiersprachen wie Java (siehe Java Syntax Listing 2.1 vs. C# Syntax Listing 2.2) und JavaScript erkennen.

Listing 2.1: Java Syntax

```java
public class SyntaxCompare
{
    public static void main(String []args)
    {
        String prefix = "";

        if( args.length > 0 ) {
          prefix = args[0];
        }

        for (int i = 0; i < 5; i++) {
          System.out.println(prefix + i);
        }
    }
}
```

Listing 2.2: C# Syntax

```csharp
using System;
public class SyntaxCompare
{
  public static void Main(String[] args)
  {
    var prefix = "";
    if (args.Length > 0)
    {
      prefix = args[0];
    }
    for (var i = 0; i < 5; i++)
    {
      Console.WriteLine(prefix + i.ToString());
    }
  }
}
```

2.4.3 Das Framework WPF

Mit dem WPF Framework, ist es möglich interaktive Anwendungen zu entwickeln, so Kotz [Kot19]. Weiters erklärt Kotz [Kot19], dass mit der Hilfe der eXtensible Application Markup Language (eXtensible Application Markup Language (XAML)), siehe 2.4.4, die Logik beziehungsweise der Code für die grafische Oberfläche sauber von der Geschäftslogik getrennt verwaltet wird. In Abbildung 2.15 sieht man den eXtensible Markup Language (XML) Code und die daraus resultierende grafische Oberfläche.

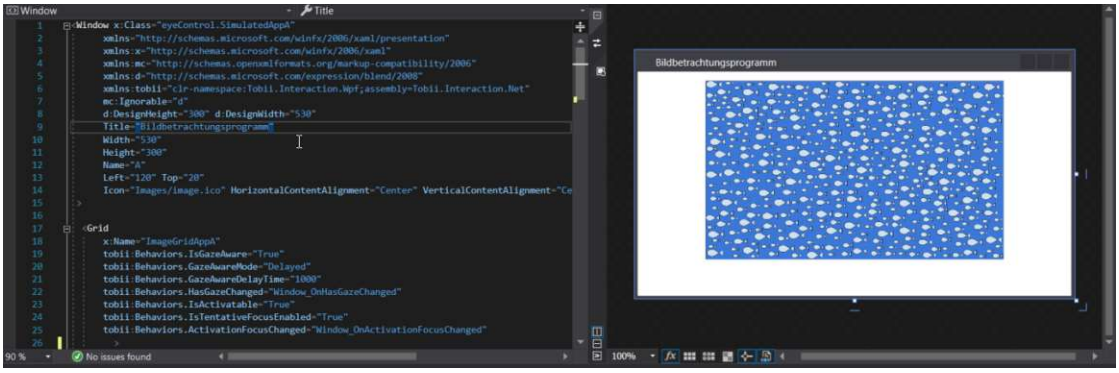


Abbildung 2.15: Microsoft Visual Studio 2019, WPF).

2.4.4 Die Beschreibungssprache XAML

Die grafische Benutzeroberfläche für eine WPF Anwendung wird in der Beschreibungssprache XAML entwickelt. XAML basiert auf XML und verwendet deshalb ebenfalls Tags um eine Oberfläche zu beschreiben beziehungsweise zu definieren. Ein Tag enthält Anweisungen in Spitzen Klammern, wobei innerhalb dieser Klammern zu Beginn der Tag Name steht und danach optional Attribute. Diese Attribute beschreiben die Eigenschaften des Tags. Um eine grafische Struktur aufzubauen können diese Tags ineinander verschachtelt werden. Im Listing 2.3 ist ein kurzer Codeauschnitt aus einer XAML Datei dieser Arbeit zu sehen.

Listing 2.3: Codebeispiel zu XAML

```
<!-- Grid Tag -->
<Grid
   <!-- Attribute/Einstellungen -->
   tobii:Behaviors.IsGazeAware="True"
   tobii:Behaviors.GazeAwareMode="Delayed"
   tobii:Behaviors.GazeAwareDelayTime="1000"
   tobii:Behaviors.HasGazeChanged="Window_OnHasGazeChanged"
   tobii:Behaviors.IsActivatable="True"
   tobii:Behaviors.IsTentativeFocusEnabled="True"
>
   <!-- MediaElement Tag (verschachtelt innerhalb des Grid Tags)-->
   <MediaElement
      Source="C:\Users\Test\Videos\test_video.mpg"
      Name="mePlayer"
      Stretch="Fill"
      LoadedBehavior="Manual" UnloadedBehavior="Stop"
   />
   <!-- MediaElement Tag Ende -->
```

```
<!-- Grid Tag Ende -->
</Grid>
```

2.5 Tobii Core Software Development Kit (SDK)

Für die Entwicklung einer Eye-Tracking Anwendung besteht die Notwendigkeit die Daten des Eye Trackers auszulesen beziehungsweise auf bestimmte Ereignisse entsprechend reagieren zu können. Dieser Abschnitt beschreibt die Grundlagen der Programmbibliothek (Core SDK) zum Tobii Eye Tracker für die Programmiersprache C#. Wie vom Unternehmen Tobii [tob19] beschrieben, kann man in Microsofts Visual Studio über den NuGet Paket Manager Bibliotheken von Drittherstellern herunterladen und installieren. Mit Hilfe dieses Software Development Kits (SDKs) können sowohl neue als auch bestehende C# Programme um Eye-Tracking Funktionalität erweitert werden. Tobii stellt des Weiteren eine komplette Referenz der Programmierschnittstelle (Application Programming Interface (API)) zur Verfügung. Diese API ermöglicht es zum Beispiel auf diverse Ereignisse des Eye Trackers zu reagieren. Im Listing 2.3 wurde ein Attribut mit dem Namen „tobii:Behaviors.HasGazeChanged“ gesetzt welches die Funktion „Window_OnHasGazeChanged“ aufruft, sobald das Grid fokussiert wurde. Das Listing 2.4 zeigt, wie nun durch die Verwendung der API darauf entsprechend reagiert werden kann. Dabei wird auf das „OnHasGazeChanged“ Ereignis (siehe Listing 2.3, Zeile 7) gehorcht und sobald der Blick des/der Benutzers/Benutzerin auf das Grid fällt (siehe Listing 2.4, Zeile 5), wird ein beliebiger Code ausgeführt. In diesem Fall ist das das Aktivieren (Fokus setzen) des Grids, sowie das Abspielen eines Videos.

Listing 2.4: Codebeispiel C# Logik zu XAML

```
private void Window_OnHasGazeChanged(
  object sender,
  HasGazeChangedRoutedEventArgs e)
{
  if (e.HasGaze)
  {
    this.Activate();
    this.playVideo();
  }
}
```

KAPITEL 3

Verwandte Arbeiten

In diesem Kapitel werden verwandte Arbeiten im Bereich der Steuerung eines Computers mittels Eye-Tracking beschrieben. Die Arbeiten wurde dazu in bestimmte Unterkapitel unterteilt. Das erste Unterkapitel 3.1 betrifft Anwendungen die unter anderem einen Eye-Tracker zur Maussteuerung einsetzen. Weitere Anwendungen die durch eine Kombination aus Eye-Tracker und Tastatur gesteuert werden, werden im Unterkapitel 3.2 vorgestellt.

3.1 Eye-Tracking und Maus

Kim, Suh und Lee [KSL17] haben in ihrer Arbeit einen Eye-Tracker mit einem Handgesten-Controller kombiniert. Das Paper beschreibt eine alternative Maussteuerung mit Hilfe eines Eye-Trackers. Das Problem dabei ist aber, dass der Eye-Tracker für das exakte platzieren eines Mauszeigers zu ungenau ist. Aus diesem Grund kombinierten Kim, Suh und Lee [KSL17] für die Platzierung des Mauszeigers den Eye-Tracker mit einem Handgesten-Controller. Um zwischen der Eye-Tracking Steuerung und der Gestensteuerung umzuschalten wurde folgendes Prinzip angewendet: Ist der Abstand zwischen Zeige- und Mittelfinger größer als ein vordefinierter Schwellwert steuert der Eye-Tracker den Mauszeiger, ist der Abstand kleiner reagiert der Mauszeiger auf die Gestensteuerung. Ein Klick wird ausgelöst indem der Daumen neben den Zeigefinger gelegt wird. [KSL17]

3.2 Eye-Tracking und Tastatur

Bereits 2007 erforschten Kumar, Paepcke und Winograd [KPW07] die Eye-Tracking Steuerung eines Computers. Sie entwickelten eine Steuerungsmöglichkeit, bei der Eye-Tracking in Kombination mit einer Tastatur eingesetzt wird. In ihrer Arbeit beschreiben Kumar, Paepcke und Winograd [KPW07], dass der/die BenutzerIn auf das gewünschte Objekt blickt und eine Taste drückt, danach blickt er/sie auf das Ziel und drückt nochmals eine bestimmte Taste. Dadurch wird eine Art „Drag and Drop“ ausgeführt, wobei die

fokussierte Stelle durch quadratisch angeordnete orange Punkte vergrößert dargestellt wird.

3.2.1 Eye-Tracking mit virtueller Tastatur

Eine Kombination aus virtueller Tastatur und einem Eye-Tracker erforschten Kumar u. a. [KHMS20]. Dazu zeigten sie am Bildschirm eine virtuelle Tastatur an, mit Hilfe dieser BenutzerInnen Wörter beziehungsweise Sätze schreiben können indem sie mit ihren Blicken auf der Tastatur ein Wort schreiben. Die Augen bewegen sich also über die Tastatur um ein Wort zu bilden ohne das man bei allen Buchstaben verweilt. [KHMS20]

In der Arbeit von Ma u. a. [MYW+18] wurde ein Brain Computer Interface (BCI) (misst und analysiert die Aktivitätsströme des Gehirns welche in Steuersignale umgewandelt werden) in Kombination mit einem Eye-Tracker eingesetzt um, wie auch schon im zuvor genannten Paper, über eine virtuelle Tastatur Wörter zu schreiben. Durch das Zusammenführen beider Datenquellen (BCI und Eye-Tracker Daten) wurde eine höhere Geschwindigkeit beim Schreiben auf der virtuellen Tastatur erreicht als bei der Durchführung mit nur einer Datenquelle (nur BCI, oder nur Eye-Tracking). [MYW+18]

3.3 Grafische Benutzeroberflächen

Menges u. a. [MKSS16] entwickelten ein Framework zur Erstellung grafischer Eye-Tracking Benutzeroberflächen. Dabei wurden laut Menges u. a. [MKSS16] alle zur Verfügung stehenden grafischen Komponenten speziell für den Einsatz von Eye-Tracking konzipiert. Zum Beispiel werden Buttons automatisch aktiviert wenn man diese betrachtet und kleiner wenn die Aktion vorbei ist [MKSS16]. Man kann unter anderem aus Elementen wie Buttons, Bilder und Texten wählen, wobei die meisten davon in ihrer Erscheinung und Größe anpassungsfähig sind.

In einer weiteren Arbeit von Menges u. a. [MKMS17] wurde ein Webbrowser implementiert, welcher speziell auf die Steuerung mittels Eye-Tracking ausgelegt ist. Es wurden unter anderem die wichtigsten Funktionalitäten wie zum Beispiel Texteingabe, Scrolling (Abb. 3.1), Hyperlink Navigation und das Browser Tab Management umgesetzt [MKMS17].

Von Sidorakis, Koulieris und Mania [SKM15] wurde eine Benutzeroberfläche für Head Mounted Displays in Verbindung mit Eye-Tracking entwickelt. Dabei können BenutzerInnen im virtuellen dreidimensionalen Raum unterschiedliche Anwendungen bedienen. Wie Sidorakis, Koulieris und Mania [SKM15] beschreiben, wurde eine Anwendung entwickelt die zur Auswahl anderer Anwendungen dient, wie eine Art dreidimensionaler Computer Desktop. Mit Hilfe dieser Anwendung konnten folgende Programme gestartet werden: ein Puzzle Spiel, ein Flugzeug Spiel, ein Programm zur Erfassung von E-Mails, ein Musikspieler und eine Fotogalerie. Durch das betrachten der grafischen Elemente werden bestimmte Aktionen ausgelöst. In Abbildung 3.2 sieht man als Beispiel die Anwendung der Fotogalerie.

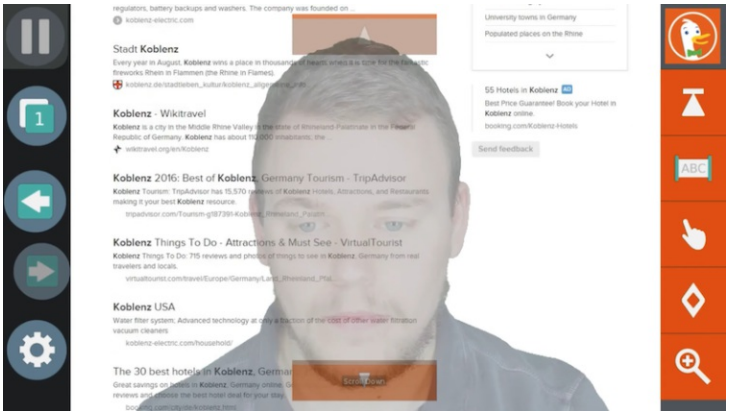


Abbildung 3.1: GazeTheWeb, Scrolling (Quelle: Video Screenshot[Lan21], Menges u. a. [MKMS17])

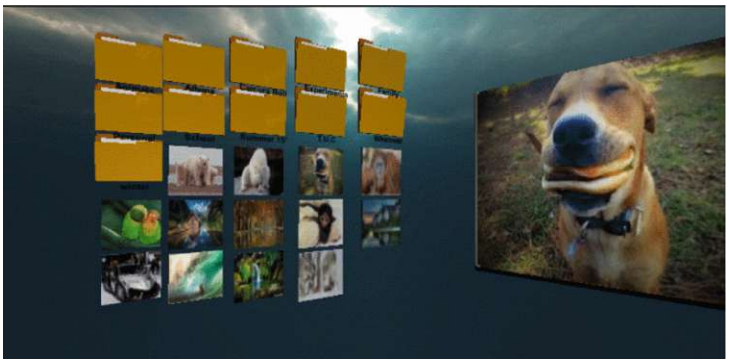


Abbildung 3.2: 3D Fotogalerie (Quelle: [SKM15])

Ein Open Source Projekt namens OptiKey [Jul21] besteht aus dem Konzept existierende Interaktionen wie zum Beispiel einen linken oder rechten Mausklick mit Hilfe des Eye-Trackers auszulösen. Dazu muss immer zuerst eine Aktion wie zum Beispiel ein Doppelklick, ausgewählt werden um danach diesen auf ein gewünschtes Element auszuführen. Die Screenshots wurden aus dem Präsentationsvideo von OptiKey [Jul21] gezogen. Diese demonstrieren einen linken Mausklick. Dabei wird nach einer kurzen Verweildauer zuerst die Aktionsgruppe (Abb. 3.3b) gewählt, danach muss die Aktion ausgewählt (Abb. 3.3b) werden, anschließend wird wieder mit kurzer Verweildauer ausgewählt auf welches Element (Abb. 3.3c) die Aktion ausgeführt werden soll und am Schluss wird der Bereich mit diesem Element vergrößert (Abb. 3.3d) und der Klick wird nach einer Verweildauer ausgeführt.

(a) Aktionsgruppe auswählen

(b) Aktion auswählen

(c) Element auswählen

(d) Aktion ausführen

Abbildung 3.3: OptiKey Steuerung (Quelle: [Jul21])

KAPITEL 4

Design und Implementierung

Dieses Kapitel werden die erarbeiteten Interaktionskonzepte beschrieben. Diese wurden in alternative und neue Operationen unterteilt. Um diese Konzepte zu realisieren und anschließend zu evaluieren wurde ein Software-Prototyp entwickelt auf dessen technische Umsetzung näher eingegangen wird. Gegen Ende dieses Kapitels wird das konzipierte Software- und Hardwaredesign des Prototypen beschrieben.

4.1 Alternative Operationen zu bestehenden Konzepten

In diesem Unterkapitel werden Eye-Tracking Alternativen zu bestehenden Konzepten der Maus-/Tastatursteuerung vorgestellt. Des Weiteren wird für jedes Eye-Tracking Konzept die technische Umsetzung beschrieben.

4.1.1 Applikationswechsler

In den meisten Fällen hat man beim Arbeiten mit einem Computer mehrere Anwendungen geöffnet zwischen denen man hin und her wechseln möchte. Zu diesem Anwendungsfall gibt es in den populären Betriebssystemen macOS, Windows und in diversen Linux Distributionen bereits sehr gute Konzepte. Meist gelangt man hier über eine Tastenkombination (Alt+Tab) auf der Tastatur zu einer Übersicht (Beispiel Abb. 4.1) der geöffneten Programme. Durch gedrückt halten der Alt-Taste bleibt die Übersicht offen. Danach drückt man so oft die Tabulator-Taste bis man bei der gewünschten Anwendung angekommen ist und lässt dann beide Tasten los. Springt man zu weit, kann man zusätzlich die Shift-Taste halten um eine Applikation zurückzuspringen. Der Nachteil bei diesem verbreiteten Konzept ist, dass es bei einer großen Anzahl laufender Programme zu unnötig langem Durchschreiten der Programmliste kommen kann. Durch Zuhilfenahme der Maus ergibt sich ein direktes und somit schnelleres Anwählen der gewünschten Applikation.

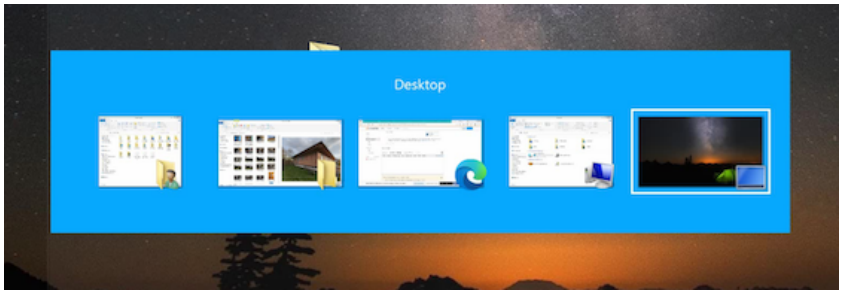
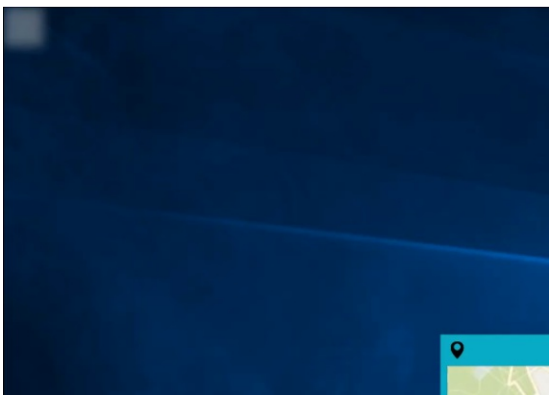
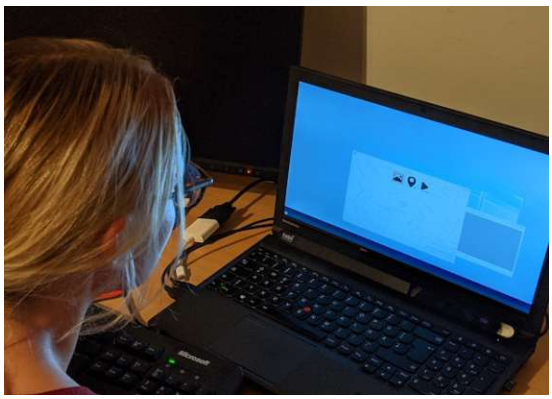


Abbildung 4.1: Anwendung wechseln in Microsoft Windows 10 (Quelle: Microsoft Windows 10).

Dieses Konzept wurde aufgegriffen um es mit Hilfe von Eye-Tracking nutzen zu können. Dabei wurde versucht den Vorteil der raschen Direktauswahl zu nutzen. Im Falle der Bedienung mit dem Eye-Tracker wird hierzu eine Ecke betrachtet. Diese sensitive Ecke wird in diesem Prototypen als „Hot-Corner“ bezeichnet. Wenn man diese Ecke betrachtet, erscheint eine kleine weiß schimmernde Kachel (Abb. 4.2a). Nach einer Verweilzeit von 500 Millisekunden öffnet sich anschließend der Applikationswechsler. Der Applikationswechsler ist technisch gesehen ein Programmfenster ohne Rahmen und Schaltflächen. Es ist transparent und liegt eine Ebene über allen anderen Anwendungen. Die geöffneten Anwendungen werden zentral mit deren Icon und Titel nebeneinander angezeigt. Um zu einer Anwendung zu wechseln muss man das jeweilige Icon betrachten. Beim fokussieren eines Icons wird als Feedback ein roter Rahmen um das Icon gezeichnet 4.2b). Um nicht unabsichtlich nach kurzem Blick auf ein Anwendungsicon zu diesem zu wechseln, wird der Wechsel zu der Anwendung des betrachteten Icons erst nach eineinhalb Sekunden ausgelöst.



(a) Hot Corner



(b) Anwendung im Applikationswechsler wählen

Abbildung 4.2: Applikationswechsler Illustration

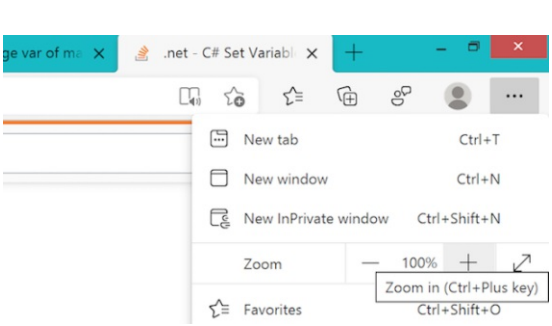
Wie schon im vorangegangenen Kapitel beschrieben, gibt es sehr häufig Situationen in

denen viele Anwendungen geöffnet sind und man schnell den Überblick verliert. Möchte man dann zu einer bestimmten Anwendung wechseln macht man dies meist über die Maus oder die Tastatur.

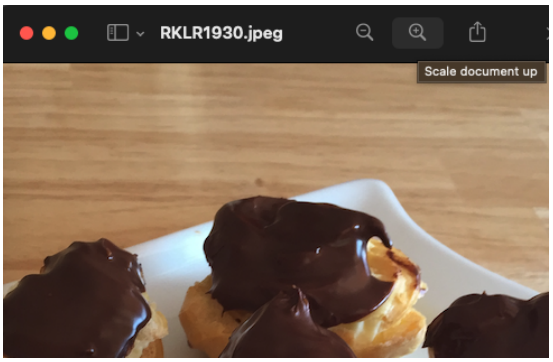
4.1.2 Bildbetrachtungsprogramm

Bei manchen Tätigkeiten während dem Arbeiten mit einem Computer möchte man Bereiche vergrößern. Oft handelt es sich hierbei um die Vergrößerung von Bildern, weil man die Details darauf erkennen möchte. Andere wiederum vergrößern Texte in einem Dokument um diese besser lesen zu können, sei es auf Grund einer Sehschwäche oder einfach nur um die Arbeit zu erleichtern. Der Zoom wird in sehr vielen Programmen über einen Mausklick auf eine Schaltfläche, über eine Eingabekombination auf der Tastatur (Strg-Taste halten und '+' drücken), oder über eine Kombination aus Tastatur und Mausrad durchgeführt. Bei Letzterem wird zum Beispiel die Strg-Taste gedrückt gehalten, während man die Scrollfunktion der Maus betätigt.

Zur Veranschaulichung sind in 4.3 einige Vergrößerungsschaltflächen aus verschiedenen Programmen abgebildet.



(a) Quelle: Microsoft Edge Browser



(b) Quelle: Apple macOS Big Sur Preview App

Abbildung 4.3: Vergrößerung bei diversen Programmen

Für Eye-Tracking wurde versucht diese Funktionalität intuitiver zu gestalten, somit wurden folgende beschriebene Konzepte implementiert.

Wenn der/die BenutzerIn einen gewissen minimalen Abstand zum Bildschirm erreicht hat, wird das entsprechende Objekt (Bild, Landkarte, etc.) vergrößert, sodass der/die BenutzerIn den Bereich besser sehen kann. Es kommt oft vor, dass diese Schaltfläche in unterschiedlichen Programmen jeweils an einer anderen Stelle in der Benutzeroberfläche positioniert ist. Der/Die BenutzerIn muss sich somit an die unterschiedlichen Positionierungen gewöhnen. Im Gegensatz zur Steuerung über eine Vergrößerungsschaltfläche hat die Eye-Tracking Variante den Vorteil, dass sie ohne Schaltflächen auskommt. Somit wird

gewährleistet, dass das Vergrößern und Verkleinern über viele Programme hinweg, mit dem gleichen Interaktionskonzept funktioniert.

Für die Eye-Tracking Interaktionstechnik wird der Abstand des Kopfes zum Eye-Tracker genutzt (Abb. 4.4) um so die Zoomstufen zu steuern.

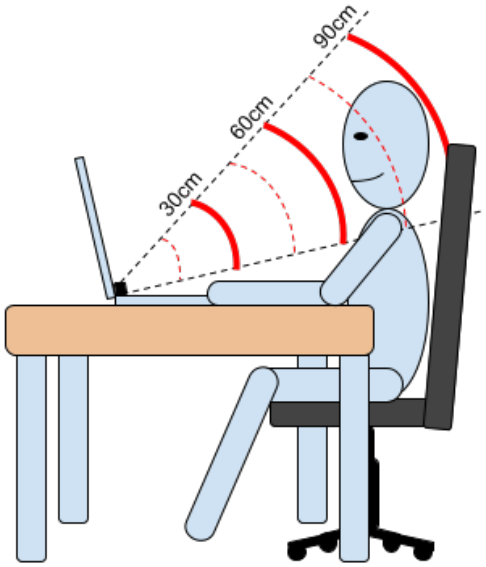


Abbildung 4.4: Abstand zum Eye-Tracker (Bildschirm)

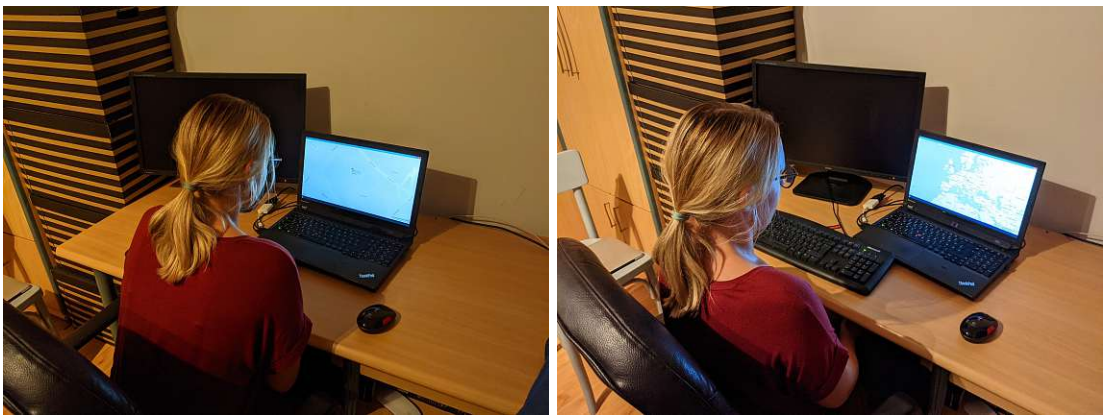
In einem Abstand von ungefähr einem Meter wurden zehn Zoomstufen implementiert. Wenn die Kopfposition des/der BenutzerIn unter den minimalen Abstand von zirka 30 Zentimeter kommt, kann der Eye-Tracker die Person nicht mehr erfassen. Selbiges gilt für den maximalen Abstand von ungefähr 95 Zentimeter ab dem es, nach beobachtetem Verhalten, für den Eye-Tracker schwierig wird den/die BenutzerIn zu erfassen.

4.1.3 Landkarte

Eine weitere Testanwendung welche das Zoomkonzept verwendet ist eine digitale Landkarte. In diesem Fall wurde die Entwicklerversion der ©Microsoft Bing Map [mic21] eingesetzt. Ebenso wie beim Bildbetrachtungsprogramm wird der Abstand zum Bildschirm verringert oder erhöht (Abb. 4.4) und somit die Landkarte vergrößert und verkleinert indem man sich vor- (Abb. 4.5a) oder zurücklehnt (Abb. 4.5b). Für die Landkarte wurden alle angebotenen Zoomstufen der Microsoft Bing Map übernommen, insgesamt sind das 19 Stufen.

Zur besseren Orientierung wurde eine Markierung mit einem „Pin“ auf die Karte gesetzt, welche den Startpunkt einer jeden Aufgabe kennzeichnet. Um nun auch auf der Karte

navigieren zu können wurde über die gesamte Landkarte ein für den/die BenutzerIn unsichtbares Gitternetz (Abb. 4.6) gelegt. Mit Hilfe dieses Gitternetzes wurden die Ränder sowie die Ecken der Karte zu sensitiven Bereichen deklariert. Der große Bereich in der Mitte bleibt insensitiv. Man kann sich dort auf der Karte umsehen ohne dass eine Aktion ausgelöst wird. Die sensitiven Bereiche reagieren nach einer Verweilzeit von 500 Millisekunden. Blickt man zum Beispiel in den Westen der Karte (linker Rand) in den sensitiven Bereich, so wandert man auch auf der Karte in Richtung Westen. Genau gleich verhält sich die Karte in alle anderen Himmelsrichtungen. Auch die Ecken der Landkarte bewirken eine Verschiebung. Betrachtet man die linke obere Ecke so wandert man auf der Karte Richtung Nordwesten. Analog verhalten sich alle anderen Ecken beim Fokussieren dieser (rechts oben - Nordosten, rechts unten - Südosten, links unten - Südwesten).



(a) Landkarte vergrößern (vorlehnen). (b) Landkarte verkleinern (zurücklehnen).

Abbildung 4.5: Landkarte vergrößern/verkleinern

4.1.4 Vollbild

Auch bei diesem Konzept handelt es sich um eine Alternative zur herkömmlichen Variante um ein Programm in den Vollbildmodus zu versetzen. Im Gegensatz zum Konzept der Vergrößerung ist hier jedoch die Position der Vollbildmodus-Schaltfläche in jedem Betriebssystem standardisiert. Die Schaltfläche ist, sofern das Programm für einen Vollbildmodus geeignet ist, pro Betriebssystem immer an der gleichen Stelle im Fensterrahmen zu finden (Abb. 4.7).

Analog zu den zuvor beschriebenen Anwendungsszenarien der Landkarte und dem Bildbetrachtungsprogramm, wird auch hier die Distanzmessung eingesetzt um bestimmte Aktionen zu triggern.
Sobald sich der/die BenutzerIn zurücklehnt beziehungsweise einen gewissen maximalen Abstand zum Bildschirm erreicht, wechselt das gerade fokussierte Fenster in den Vollbildmodus.
Dies hat den Vorteil, dass keine separate Aktion notwendig ist um in den Vollbildmodus

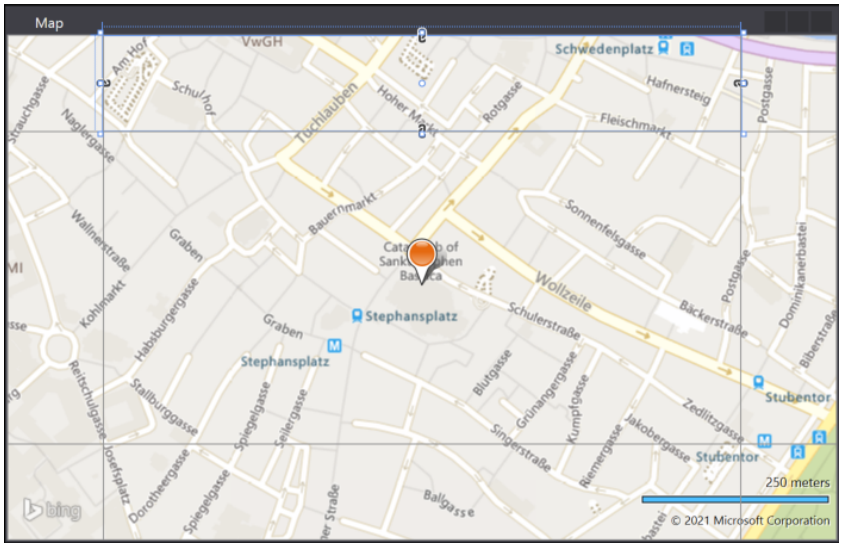
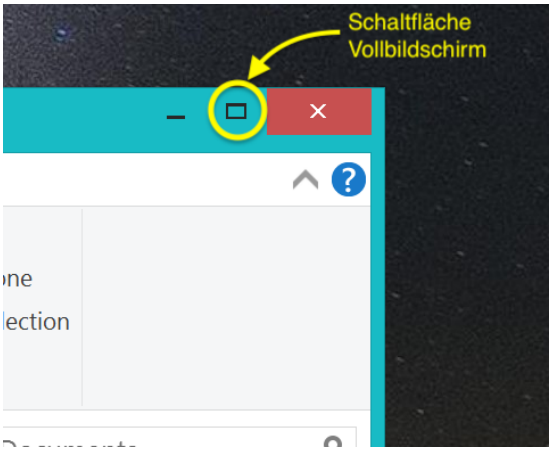
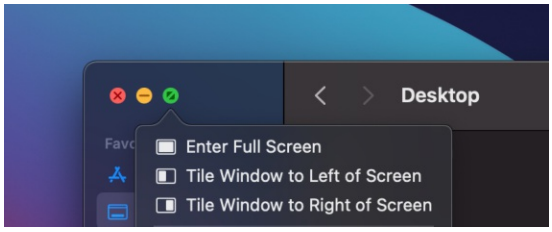


Abbildung 4.6: Gitternetz Überlagerung der Landkarte



(a) Quelle: Microsoft Windows 10



(b) Quelle: Apple macOS Big Sur

Abbildung 4.7: Vollbild

zu gelangen. So könnte diese Funktionalität bei gewissen Programmarten wie zum Beispiel bei einem Videoplayer, bei Bildbetrachtungsprogramme und bei Programmen zum Betrachten von Dokumenten hilfreich sein.
Eine Applikation befindet sich im „Normalzustand“ betreffend der Fenstergröße. Durch die Vergrößerung der Distanz des Kopfes zum Monitor wird ab einer Entfernung von 70 Zentimeter das Programmfenster der zuletzt fokussierten Anwendung maximiert. Dazu wird bei der Überschreitung dieses Schwellwerts der Status des Programmfensters auf „Maximized“ gesetzt. Sobald die Distanz wieder verringert wird erkennt der Prototyp dieses Ereignis und setzt den Fensterstatus wieder auf „Normal“.

4.1.5 Wiedergabe/Pause von Videos

Das Abspielen und Pausieren von Videos erfolgt in den meisten Programmen über die bekannten Schaltflächen „Abspielen“ und „Pause“ (Abb. 4.8). Aus diversen Gründen kann es jedoch vorkommen, dass ein/eine BetrachterIn sich vom Bildschirm abwendet und dem Video somit keine Aufmerksamkeit mehr schenkt. Das Video läuft somit weiter und man verpasst dadurch einen Teil des Videos.

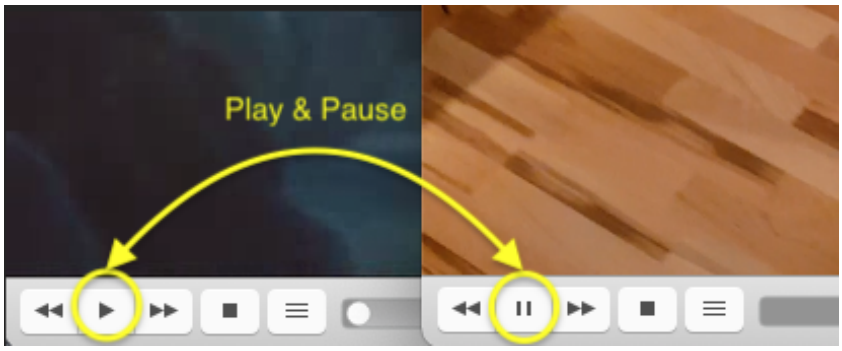


Abbildung 4.8: Wiedergabe/Pause Video Player (Quelle: VLC Video Player).

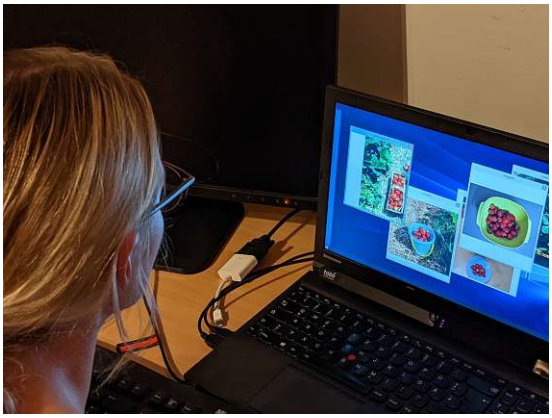
Um diesem Problem entgegen zu wirken wurde für Eye-Tracking ein Konzept entworfen, bei dem das Video unter anderem erst zum Abspielen beginnt wenn man das Fenster des Videoplayers ansieht. Fokussiert man dann wieder eine andere Anwendung pausiert das Video die Wiedergabe. Wechselt der Fokus des/der Benutzers/Benutzerin wieder auf das Video, setzt die Wiedergabe fort. Zusätzlich kann der Eye-Tracker über die Präsenz des/der Benutzers/Benutzerin gesteuert werden. Dabei erkennt der Eye-Tracker ob eine Person auf den Bildschirm blickt oder nicht. Das bedeutet wenn der/die BenutzerIn nicht mehr auf den Bildschirm blickt pausiert es. Die Wiedergabe startet erst wieder wenn der Eye-Tracker die Augen der Person wieder erfassen kann.

4.1.6 Splitscreen

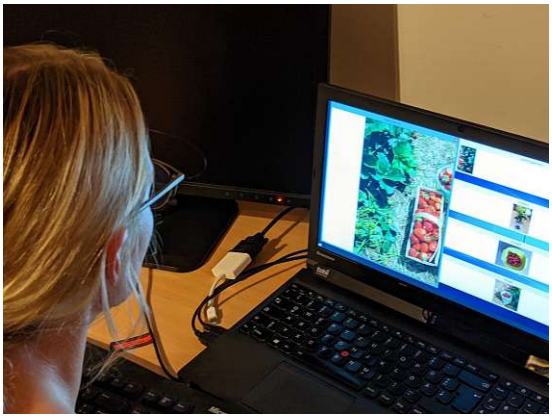
Um einen Splitscreen in Microsofts Windows 10 zu erreichen, kann man zum Beispiel ein Programmfenster an den linken Rand ziehen und ein weiteres an den rechten Rand. Beim berühren des linken oder rechten Randes vergrößert sich die Anwendung auf die jeweilige Bildschirmhälfte. Unter Apples macOS kann man über die Vergrößerungsschaltfläche (siehe 4.7b) einen Splitscreen auslösen. Man wählt dazu aus, auf welche Bildschirmhälfte sich das aktuelle Programmfenster verschieben soll. Wählt man zum Beispiel die linke Hälfte aus werden auf der rechten Hälfte alle anderen Anwendungen zur Auswahl angezeigt. Hat man dann eine Anwendung ausgewählt wird diese auf der rechten Bildschirmhälfte angezeigt.

Bei dieser Anwendung wird der rechte und linke Bildschirmrand als besonderer Bereich genutzt um bestimmte Aktionen auszulösen. In dieser Arbeit und für dieses Konzept

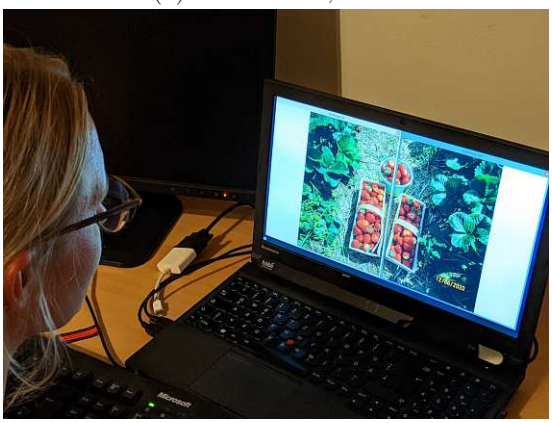
werden diese sensitiven Bereiche als „Hot-Borders“ bezeichnet. Wie auch schon vom Applikationswechsler 4.1.1 bekannt, verfärben sich auch hier die Ränder sobald man sie betrachtet leicht weiß und sind ein wenig transparent. Man erkennt somit dass dieser Bereich reagiert. Wenn ein Fenster fokussiert wurde und danach der Blick auf den linken Rand wandert, verfärbt sich der Rand wie beschrieben (Abb. 4.9a). Auch hier muss man für eine Sekunde mit den Augen verweilen bis eine Aktion ausgelöst wird. Dann wird die zuvor fokussierte Anwendung automatisch vergrößert und belegt die linke Bildschirmhälfte. Die Auswahl der Applikation für die rechte Hälfte, funktioniert ähnlich wie beim Splitscreen von Apple. Dabei teilen sich alle restlichen Anwendungen selbstständig den noch zur Verfügung stehenden Platz auf der rechten Bildschirmhälfte (Abb. 4.9b). Nach einem Blick auf die gewünschte Anwendung für die rechte Bildschirmhälfte, vergrößert sich auch dieses Programmfenster. Das Endresultat (Abb. 4.9c) ist eine Aufteilung zweier Anwendungen auf jeweils eine Hälfte des Bildschirms, dieses Konzept wird in dieser Arbeit als „Splitscreen“ bezeichnet.



(a) Hot-Border, links



(b) Splitscreen, rechte Seite wählen

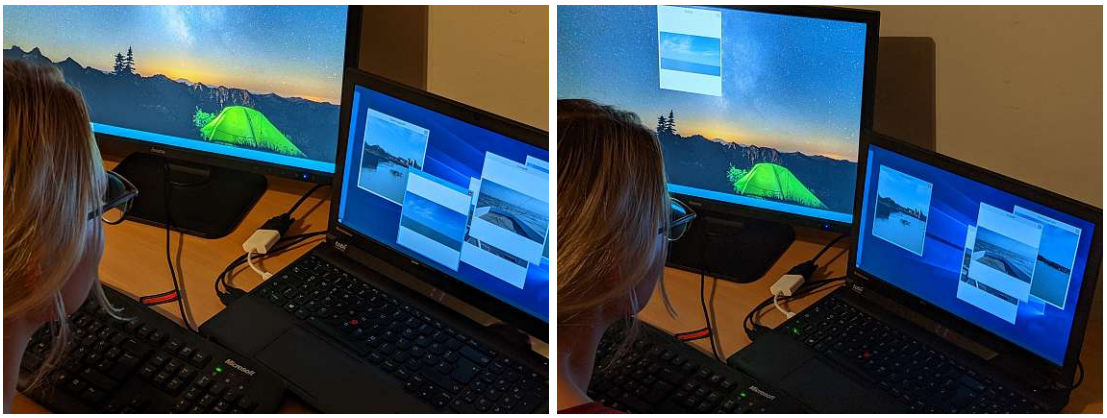


(c) Splitscreen, Erfolgreich

Abbildung 4.9: Splitscreen Illustration

4.1.7 Fenster verschieben

Für „Multi-Screen“ wurde ein Konzept entwickelt, welches sich wiederum die markanten Bereiche am Bildschirm sowie einer bestimmten Verweilzeit (engl. „Dwell-Time“) zu nutze macht.
Wenn man mit mehreren Bildschirmen arbeitet kann es häufiger vorkommen, dass man Anwendungen von einem Monitor auf den anderen verschieben möchte. Diese Operation kann fast in jedem Betriebssystem mit der Maus durchgeführt werden. In manchen Betriebssystemen, wie zum Beispiel in Microsoft Windows, kann die Verschiebung auch über eine Tastenkombination erreicht werden.
Auch für diese Operation wurde folgend beschriebene Alternative über Eye-Tracking geschaffen. Da es technisch nicht möglich war zwei Eye-Tracker gleichzeitig auf einem Computer zu betreiben, sind auch die potentiellen Interaktionstechniken begrenzt. Dieses Interaktionskonzept basiert ebenfalls, wie bereits vom Splitscreen 4.1.6 beschrieben, auf den „Hot-Borders“. Für dieses Konzept wird jedoch nur ein Hot-Border benötigt, mit dessen Hilfe man das Verschieben einer Anwendung auf den externen Monitor auslösen kann.
Hat man keinen externen Bildschirm angeschlossen, werden beide Bildschirmränder für das Splitscreen Konzept genutzt. Sobald man einen zusätzlichen Bildschirm an den Computer anschließt erkennt das der Prototyp und der linke oder rechte (konfigurierbar) sensitive Rand ist bereit um ein Programmfenster auf den externen Bildschirm zu verschieben. Das Auslösen dieser Aktion erfolgt ident zur zuvor beschriebenen Splitscreen Aktion. Nachdem eine Anwendung fokussiert wurde und man für eine Sekunde den Rand betrachtet hat (Abb. 4.10a), verschiebt sich die zuletzt betrachtete Applikation auf den externen Bildschirm (Abb. 4.10b).



(a) Hot-Border, links (b) Externer Monitor, verschoben

Abbildung 4.10: Externer Monitor, Anwendung verschieben

4.2 Neue Operationen

Neben den bereits beschriebenen alternativen Interaktionskonzepten wurden auch zwei neue Konzepte umgesetzt. Beide Konzepte arbeiten mit dem Ansatz, dass Inhalte einer Anwendung unkenntlich gemacht werden.

4.2.1 Sicherheit bei Abwesenheit

Bei diesem Konzept geht es um das Thema Datenschutz. Es gibt Situationen in denen man sensible Daten am Computer einsieht oder bearbeitet. Damit diese nicht für alle einsehbar sind, sperrt man den Computer bevor man seinen Platz verlässt. Das neue Konzept stellt alle Programmfenster verschwommen („blurred“) dar sobald sich der/die BenutzerIn nicht mehr im Erfassungsbereich des Eye-Trackers befindet. Dies bringt in öffentlichen Bereichen bezüglich Datenschutz bei ungesperrten Computern (kein Login erforderlich), wo sensible Inhalte nicht einsehbar sein sollten, durchaus mehr Sicherheit.

4.2.2 Fenster hervorheben

Sobald ein Fenster durch den/die BenutzerIn fokussiert wird, werden alle anderen Fenster im Hintergrund verschwommen („blurred“), sodass das fokussierte Programm hervorgehoben erscheint.

Durch die verschwommenen Anzeigen anderer Programme wird der/die BenutzerIn nicht von der tatsächlichen Arbeit abgelenkt. Diese Funktionalität soll sowohl zur Förderung der Konzentration als auch zur Steigerung der Produktivität beitragen.

4.3 Iterativer Designprozess

Bevor die Evaluierung der ausgewählten Ideen durch die Testpersonen begonnen wird, sollen Pilottests sicherstellen, dass keine unerwarteten Störungen, oder Ereignisse auftreten und das System wie gewünscht reagiert. Dabei soll eine Testperson den iterativen Designprozess der Konzepte begleiten um Feedback über die umgesetzten Ideen zu erhalten und sie so zu verbessern. Verläuft ein Pilottest negativ (System reagiert falsch), so muss der Prototyp solange überarbeitet werden, bis er das gewünschte Verhalten aufweist. Diese Iteration soll nicht über mehr als drei Runden gehen um den Rahmen und die Durchlaufzeit der gesamten Studie nicht zu überschreiten.
Danach kann die nächste Phase der Evaluierung, Usability Tests mit Hilfe von Testpersonen, gestartet werden.

4.4 Grundlegende Überlegungen zur Implementierung

Eines der Ziele bei der Entwicklung der bereits genannten Interaktionskonzepte ist es, dem/der BenutzerIn über die Augensteuerung eine intuitive Interaktion mit der grafischen Oberfläche anzubieten. Deshalb ist die Entscheidung der Wahl des Eye Trackers, welcher in

dieser Arbeit verwendet wird, eine der ersten die gefällt wurde. Die unter 2.3 aufgeführten Eye-Tracker standen hierbei zur Auswahl. Auf Grund des Preis-Leistungsverhältnisses und somit für viele BenutzerInnen attraktiv, fiel die Wahl auf den Eye Tracker 4C von Tobii. Die Auswahl der passenden Programmiersprache ist durch die Wahl der Eye-Tracker Hardware auf C++ und C# minimiert worden. Durch die Verwendung von C# und WPF wurde die Umsetzung einer simulierten Microsoft Windows Umgebung ermöglicht, welche den meisten Benutzern/Benutzerinnen bei der Evaluierung eine gewisse Vertrautheit gibt. Der Eye-Tracker wurde in Verbindung mit einem Lenovo T540p Notebook mit 16 Gigabyte Arbeitsspeicher und dem Betriebssystem Windows 8.1 betrieben. Weitere periphere Geräte, wie Maus oder Tastatur, sollen während der Eye-Tracking Interaktion nicht zum Einsatz kommen.

4.5 Komponenten

Die Interaktionskonzepte werden wie schon erwähnt in einer WPF Applikation umgesetzt. Dabei verwendet der Prototyp die von Tobii bereitgestellte API [tob19] um auf Eingaben beziehungsweise durch den/die BenutzerIn ausgelöste Ereignisse zu reagieren. Nach der Verarbeitung eines Ereignisses verändert sich dementsprechend die Benutzeroberfläche. Die Darstellung 4.11 zeigt den Aufbau der Komponenten des Prototyps. Alle Eye-Tracking Ereignisse auf welche reagiert werden soll werden in den jeweiligen XML Dateien definiert in denen auch der Aufbau der UI Elemente beschrieben ist. Um auf die von der API verfügbaren Ereignisse reagieren zu können definiert man eine Ereignismethode in der zur XML Datei zugehörigen C# Klasse.

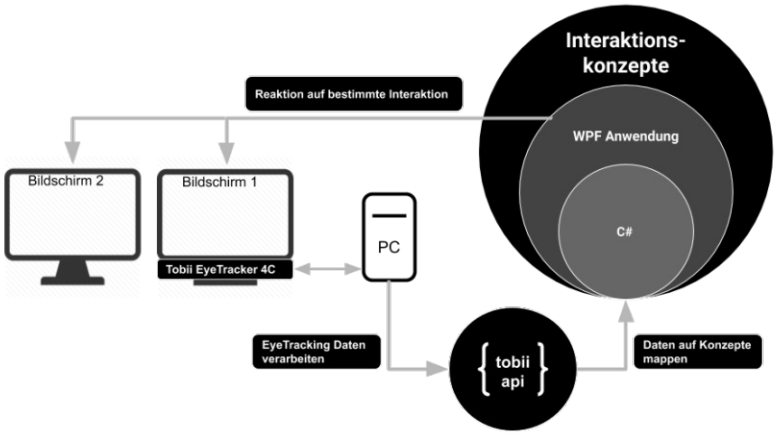


Abbildung 4.11: Komponenten des Prototyps.

4.6 WPF Anwendung

Um für die BenutzerIn eine möglichst realitätsnahe Umgebung zu schaffen, versuchte man sowohl den Desktop als auch Programmfenster im Software-Prototypen nachzubilden. Dabei lehnte man sich stark an Microsofts Betriebssystem „Windows“ an. Dabei wurde als Wiedererkennungsmerkmal die Taskbar inklusive dessen Startbutton mit Windows Logo nachgeahmt.
Zur Simulation von offenen Programmen werden drei verschiedene Fenster (Abbildung 4.12) angezeigt. Jede Anwendung simuliert eine andere Art von Programm.

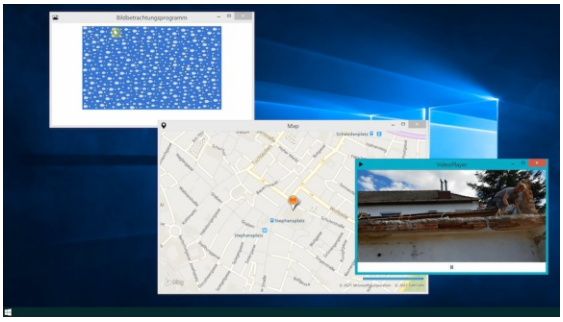


Abbildung 4.12: Basisanwendungen des Prototyps

4.6.1 Gaze-Trace

Der Gaze-Trace ist eine von Tobiis Standardinstallation mitgelieferte Möglichkeit die aktuell fokussierte Stelle am Bildschirm visuell darzustellen. Aktiviert man den „Gaze-Trace“ für den Eye-Tracker, so wird die betrachtete Stelle durch eine Art Seifenblase (Abb. 4.13) dargestellt. Der Durchmesser dieses Bereichs ist wie man sieht relativ groß, was bedeutet dass präzise Selektionen von nah aneinanderliegenden Schaltflächen mit dem in dieser Arbeit verwendeten Eye-Tracker (siehe 2.3.1) kaum bis gar nicht möglich sind.

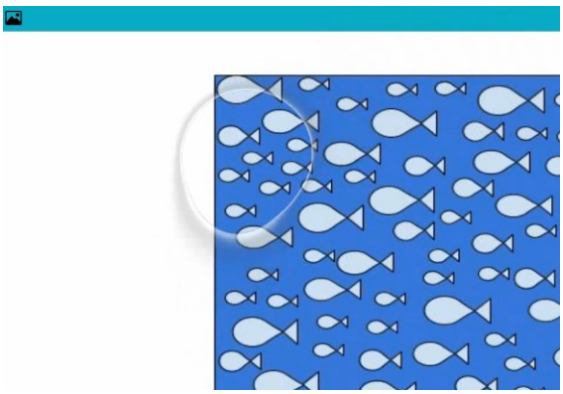


Abbildung 4.13: Gaze Trace

KAPITEL 5

Methodik der Evaluierung

Am Beginn dieses Kapitels wird die Zielsetzung dieser Diplomarbeit noch einmal verdeutlicht. Danach folgt ein Unterkapitel welches das Studiendesign sowie dessen Hypothesen und den Aufbau der Evaluierung erläutert. Den Abschluss dieses Kapitels bilden die zwei Unterkapitel Setup und Ablauf. Das Unterkapitel Setup beschreibt kurz die für die Evaluierung benötigten Ressourcen an Hard- und Software. Der Ablauf der Evaluierung wird am Ende der Studie genau beschrieben.

5.1 Zielsetzung

Ziel dieser Evaluierung ist es die entwickelten Interaktionstechniken bezüglich der Hypothesen 5.2.1 zu überprüfen. Um diese Aussagen treffen zu können werden die folgenden Metriken verwendet: Fehlerrate, Durchführungszeit (Task-Completion-Time), SUS [Bro96] Ergebnis 6.2.6. Das SUS Ergebnis wird als Gesamtvergleich zwischen Eye-Tracker und Maussteuerung über alle Aufgaben ausgewertet. Die anderen Metriken werden für jede einzelne Interaktionstechnik sowohl für die Steuerung mittels Eye-Tracking als auch für die Maussteuerung erhoben und miteinander verglichen. Eine Auswertung dieser Daten erfolgt für jede Aufgabe separat. Anhand dieser Daten werden statistische Auswertungen erzeugt und anschließend bewertet. Ebenso wie die quantitativen Werte fließen auch qualitative Ergebnisse in Form einer Auswertung der Interviewantworten in das Endresultat der Evaluierungsergebnisse ein. Nachdem alle Daten ausgewertet wurden werden diese interpretiert und ein Fazit gezogen. Dabei werden die Stärken und Schwächen der unterschiedlichen Eingabemethoden und Interaktionstechniken näher betrachtet und beschrieben.

5.2 Studiendesign

In diesem Abschnitt wird beschrieben wie die Studie und somit die Usability Evaluierung des Prototyps aufgebaut ist. Zu Beginn werden die Hypothesen und deren abhängige und unabhängige Variable beschrieben. Anschließend werden die Eigenschaften dieser Variablen näher erläutert. Auf die Auswahl der TeilnehmerInnen wird im Unterkapitel 5.2.4 eingegangen. Eine Beschreibung der definierten Übungen und Aufgaben erfolgt direkt nach dem Subkapitel 5.2.5. Den Abschluss dieses Kapitels bilden die zwei Unterkapitel Setup und Ablauf. Zum Einen beschreibt das Setup den technischen und logistischen Aufbau der Evaluierung und zum Anderen bietet das Subkapitel 5.4 eine genaue Beschreibung des Ablaufs dieser Evaluierung.

5.2.1 Hypothesen

- **H1: Übungsrunden führen zu einem Lerneffekt in Form schnellerer Task-Completion Time**

  Diese Hypothese überprüft die Annahme, dass nach einer kurzen Einschulung und darauf folgenden Übungsrunden ein Lerneffekt auftritt und dies zu einer reduzierten Aufgabendurchführungszeit führt. Durch diesen Effekt beim Lösen einer Aufgabe steigt sowohl die Zufriedenheit als auch die Akzeptanz der BenutzerInnen gegenüber dem Konzept und des Eye-Trackings.

  Unabhängige Variable: Wiederholungen
  Abhängige Variable: Task-Completion Time

- **H2: Es gibt signifikante Unterschiede zwischen der Steuerung über Eye-Tracking und der Maus-/Tastatursteuerung**

  Bei der zweiten Hypothese soll die Annahme überprüft werden, ob die Steuerung eines Computers über Eye-Tracking nicht maßgebend schlechter ist, als wenn man bestimmte Operationen über die gewohnte Maussteuerung durchführt. Dieser Unterschied soll über zwei Metriken gemessen werden. Deshalb wurde diese Hypothese in zwei Subhypothesen unterteilt:

  H2.1: Es gibt einen signifikanten Unterschied in der Fehlerrate im Vergleich zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung

  Diese Subhypothese überprüft die Annahme das die Steuerung eines Computers mit geeigneten Eye-Tracking Interaktionskonzepten keine höhere Fehlerrate im Vergleich zur Maussteuerung aufweist.

  Unabhängige Variable: Art der Steuerung

  1. Ausprägung: Maus

2. Ausprägung: Eye-Tracking

Abhängige Variablen: Fehlerrate

H2.2: Es gibt einen signifikanten Unterschied in der Durchführungszeit im Vergleich zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung

Mit dieser Unterhypothese soll überprüft werden, ob die für den Eye-Tracker entwickelten Window Management Interaktionstechniken einen zeitlichen Vorteil im Vergleich zur Maussteuerung bringen.

Unabhängige Variable: Art der Steuerung

1. Ausprägung: Maus
2. Ausprägung: Eye-Tracking

Abhängige Variablen: Task-Completion Time

- **H3: Augensteuerung erhöht die Usability (SUS Ergebnis)**

  Durch die Auswertung des SUS Fragebogens soll bei dieser Hypothese die Annahme überprüft werden, dass die für die Augensteuerung entwickelten Konzepte zu einer erhöhten Usability führen. Die Steigerung der Usability bezieht sich hier auf das Arbeiten mit mehreren geöffneten Anwendungen. Diese Hypothese ist für diese Diplomarbeit essentiell. Ist hier ein deutlich positiver Trend zu erkennen kann man daraus schließen, dass die entwickelten Konzepte den Einsatz von Eye-Trackern im Desktop Bereich durchaus fördern würden. Dadurch könnte dann in weiterer Folge, in Hinblick auf die mit den entwickelten Konzepten verbundenen Aufgaben, eine Effektivitäts- beziehungsweise Produktivitätssteigerung erzielt werden.

  Unabhängige Variable: Interaktionskonzepte
  Abhängige Variable: SUS

5.2.2 Variablen

Variable: Lerneffekt
Übung der Steuerung und Konzepte, BenutzerIn wird schneller
Rolle im Studiendesign: unabhängige kontrollierte Variable
Art der Merkmalsausprägung: stetig, mehrere Iterationen pro Interaktionskonzept (alle in 4 angeführten)
Skalenniveau: Ordinalskala

Variable: Task-Completion Time
Arbeitsgeschwindigkeit der Testperson mit dem System
Rolle im Studiendesign: abhängige Variable

Art der Merkmalsausprägung: stetig, Zeit messen (alle in 4 angeführten)
Skalenniveau: Verhältnisskala
Maß: Sekunden pro Durchführung

Variable: Fehlerrate
Anzahl der aufgetretenen Fehler durch unerwünschtes Verhalten
Rolle im Studiendesign: abhängige Variable
Art der Merkmalsausprägung: stetig, manuell Zählen (alle in 4 angeführten)
Skalenniveau: Verhältnisskala
Maß: Analyse der Bildschirmaufzeichnung

Variable: Interaktionskonzepte
Entwickelte Interaktionskonzepte für die Augensteuerung
Rolle im Studiendesign: unabhängige, kontrollierte Variable
Art der Merkmalsausprägung: polytom, Testbedingungen (alle in 4 angeführten)
Skalenniveau: Nominalskala

Variable: SUS
Gesamtes SUS Ergebnis
Rolle im Studiendesign: abhängige Variable
Art der Merkmalsausprägung: diskret, von 0 (schlecht) bis 100 (gut)
Skalenniveau: Verhältnisskala
Maß: Post-Test SUS-Fragebogen auf Deutsch

5.2.3 Within-Subject und Between-Subject Design

Wie MacKenzie [Mac] im Kapitel 5 seines Buches erläutert, ist die Anzahl der TeilnehmerInnen beim Within-Subject Design geringer und jeder/jede TeilnehmerIn führt jede Aufgabe in allen Ausprägungen durch. Auf diese Arbeit bezogen, müsste somit jede Person eine Aufgabe sowohl mit dem Eye-Tracker als auch mittels Maus-/Tastatursteuerung ausführen. Das Within-Subject Design hat zwar den Vorteil dass weniger TeilnehmerInnen organisiert werden müssen, jedoch hat man mehr Aufwand pro Aufgabe für den TeilnehmerIn, da mehr Tests (Durchläufe) notwendig sind.

Beim Between-Subject Design führt ein/eine TeilnehmerIn immer nur eine Ausprägung einer Aufgabe durch [Mac]. Für eine Studie bei der zum Beispiel ein Unterschied in der Lesegeschwindigkeit zwischen BrillenträgerInnen und Nicht-BrillenträgerInnen festgestellt werden soll, benötigt man eben diese zwei Benutzergruppen.

MacKenzie [Mac] schreibt, dass für das feststellen eines Lerneffekts durch das Wiederholen einer Aufgabe das Within-Subject Design gewählt werden muss. Die Vorteile des Within-Subject Designs gegenüber dem Between-Subject Design beschreibt MacKenzie wie folgt:

1. Weniger Teilnehmer müssen rekrutiert werden

2. Persönlichkeit, mentale-/physische Verfassung des Teilnehmers haben für alle Ausprägungen der Aufgabe den gleichen Einfluss

3. Kein ausbalancieren (TeilnehmerInnen jeder Gruppe haben gleiche Erfahrung mit Computern) von Gruppen notwendig, da es nur eine Gruppe gibt

5.2.4 TeilnehmerInnen

Weil es erwünscht ist einen Lerneffekt bei der Durchführung der Aufgaben zu erzielen, wurde das „Within-Subject Design“ [Mac] für die Evaluierung gewählt. Da diese Diplomarbeit während der Corona Pandemie entstanden ist und der persönliche Kontakt auf Grund einer Infektionsgefahr eingeschränkt werden musste, wurde die Teilnehmeranzahl in dieser Studie auf insgesamt zehn Personen beschränkt.

Im Zuge der Evaluierung gibt es keine demografischen Einschränkungen. Ein Kriterium auf das bei der Teilnehmerauswahl geachtet wird ist die Erfahrung im Umgang mit einem Computer und dem Betriebssystem Microsoft Windows 8 oder höher. Die TeilnehmerInnen sollten außerdem mindestens zwei Mal pro Woche mit einem Computer arbeiten.

Die Usability Evaluierung soll unter anderem zeigen, ob die Erfahrung der TeilnehmerInnen einen direkten Einfluss hat. Dabei wird der Umgang und die Effizienz beim Arbeiten mit den entworfenen Eye-Tracking Interaktionskonzepten beobachtet.

5.2.5 Aufbau der Übungen & Aufgaben Evaluierung

Um die entwickelten Interaktionskonzepte zu evaluieren, werden in diesem Abschnitt alle Aufgaben beschrieben. Die ausgearbeiteten Interaktionskonzepte sind allgemein gestaltet, so dass sie in diverse Anwendungen integriert werden können. In diesem Abschnitt werden die Aufgaben/Szenarien, wie in 1.3.1 erwähnt, basierend auf den in 4 beschriebenen Konzepten definiert. Diese Beschreibungen dienen unter anderem dazu, den/die BenutzerIn bei der Aufgabendurchführung zu unterstützen.

Nach der Durchführung der Evaluierung soll ein statistischer Vergleich zwischen Eye-Tracking und Maussteuerung gezogen werden. Wenn es technisch möglich ist, soll jede Aufgabe sowohl per Eye-Tracking als auch per Maus durchgeführt werden. Durch die Aufzeichnung des Bildschirminhalts wird die gesamte Evaluierung eines/einer jeden einzelnen Teilnehmers/Teilnehmerin festgehalten. Diese Bildschirmaufnahmen dienen in weiterer Folge der Analyse und Auswertung der Evaluierung.

Es ist wichtig für alle folgenden Aufgaben immer die gleichen Ausgangsbedingungen zu schaffen, daher ist folgendes als **Ausgangslage** festzuhalten:

Alle bestehenden Programmfenster werden geschlossen. Die Testperson öffnet danach durch einen Klick auf eine Schaltfläche alle für die Aufgabe benötigten Programmfenster. Bei dem in dieser Diplomarbeit umgesetzten Prototypen wurde die Schaltfläche für das Öffnen der Anwendungen gleichzeitig für das Wiederherstellen der Ausgangssituation

verwendet.

Alle Übungungsrunden und Aufgaben sind wie folgt aufgebaut:

- Aufgabenbezeichnung
  Die Kurzbeschreibung der Aufgabe.
- Vorbedingungen
  Alle Bedingungen die erfüllt sein müssen um die Aufgabe erfolgreich durchführen zu können.
- Aufgabe
  Eine genaue Beschreibung der Aufgabe zur Unterweisung des/der BenutzerIn.
- Wiederholungen (optional)
  Die Anzahl der Wiederholungen wie oft eine Aufgabe durchgeführt werden muss.
- Erwartetes Ergebnis
  Der Zustand welcher nach erfolgreicher Durchführung der Aufgabe erreicht werden soll.

Vor der Durchführung der Aufgaben hat jeder/jede TeilnehmerIn ohne genaue Vorgaben die Möglichkeit die Eye-Tracking Konzepte zu erforschen. Dafür hat jede Person Zeit sich ein wenig an die Augensteuerung zu gewöhnen und einer anfänglichen Skepsis und Berührungsängsten entgegenzuwirken. Diese Entdeckungsphase wird vor Beginn der eigentlichen Durchführung der Aufgaben stattfinden und umfasst mehrere Übungsrunden von diversen Eye-Tracking Konzepten.

5.2.6 Offene Übungsrunden

In diesem Unterkapitel werden alle Übungen genau spezifiziert. Die Übungsrunden sollen offen geführt werden. Dies bedeutet, dass der Studienleiter der Testperson in dieser gesamten Phase für Fragen, Unterstützung oder zur Erklärung von Konzepten zur Verfügung steht.

Übungsrunde 1

Bei dieser Übungsrunde handelt es sich um die Anwendung eines Konzeptes, welches die Privatsphäre adressiert. Der am Bildschirm dargestellte Inhalt soll für andere nicht erkennbar sein wenn niemand vor dem Computer sitzt. Eine Fremde Person hätte sonst die Möglichkeit aus der Distanz sensible Daten unbemerkt einzusehen.

Beschreibung eines möglichen Szenarios
Eine Person sitzt im Zug auf dem Weg zu einem wichtigen Meeting, wo diverse Unterlagen präsentiert werden müssen. Bei diesen Unterlagen handelt es sich um sensible

Firmendaten. Während der Zugfahrt werden diese Unterlagen noch überarbeitet. Die Person verlässt kurz den Platz ohne den Computerbildschirm zu sperren. Auf Grund eines integrierten Eye-Trackers wird nun der Inhalt verwischt dargestellt.

Aufgabenstellung für den/die BenutzerIn

Aufgabenbezeichnung:	Alle Bildschirminhalte verbergen
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt.
Aufgabe:	Es sind einige Anwendungen auf Ihrem Computer geöffnet. Sie müssen Ihren Platz verlassen. Stellen Sie nun sicher, dass die Inhalte auf Ihrem Bildschirm nicht einsehbar sind, ohne dass Sie den Bildschirm sperren, ausschalten oder den Laptop zuklappen. **Tipp:** *Solange der Eye-Tracker Sie erfassen kann sind die Inhalte nicht gesperrt.*
Erwartetes Ergebnis:	Die Inhalte am Bildschirm sind nicht mehr klar erkennbar.

Tabelle 5.1: Beschreibung Übung 1

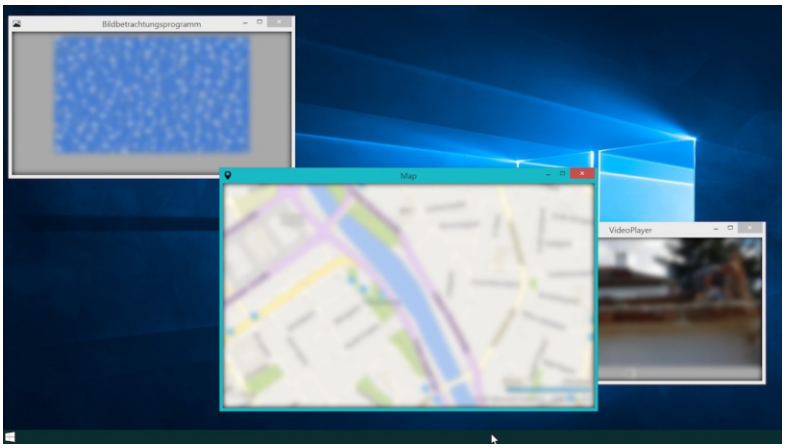


Abbildung 5.1: Übung 1: Alle Bildschirminhalte verbergen

Übungsrunde 2

Um sich besser auf eine Applikation konzentrieren zu können wurde ein Interaktionskonzept entwickelt, bei welchem alle nicht aktiven Applikationen visuell aus dem Fokus genommen werden. Diese Übungsrunde verdeutlicht dieses Konzept.

Beschreibung eines möglichen Szenarios
Ein/eine BenutzerIn hat mehrere voneinander unabhängige Anwendungen geöffnet. Er/Sie möchte konzentriert arbeiten, doch oft neigt man dazu von Programmen im Hintergrund abgelenkt zu werden. Durch das Konzept, dass nur fokussierte Programme klar erkennbar und alle anderen verwischt sind, kann sich der/die BenutzerIn auf das aktuell fokussierte Programm konzentrieren. Blickt die Person auf ein anderes Anwendungsfenster wird dies hervorgehoben und alle anderen verwischen.

Aufgabenstellung für den/die BenutzerIn

Aufgabenbezeichnung:	Anwendung wechseln
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt.
Aufgabe:	Vor Ihnen sind mehrere Anwendungen geöffnet. Wechseln Sie zwischen den Anwendungen mehrmals hin und her. **Führen Sie diese Aufgabe sowohl via Eye-Tracking als auch mit der Maus durch.**
Erwartetes Ergebnis:	Das fokussierte Fenster rückt in den Vordergrund und die anderen Fenster rücken visuell in den Hintergrund.

Tabelle 5.2: Beschreibung Übung 2

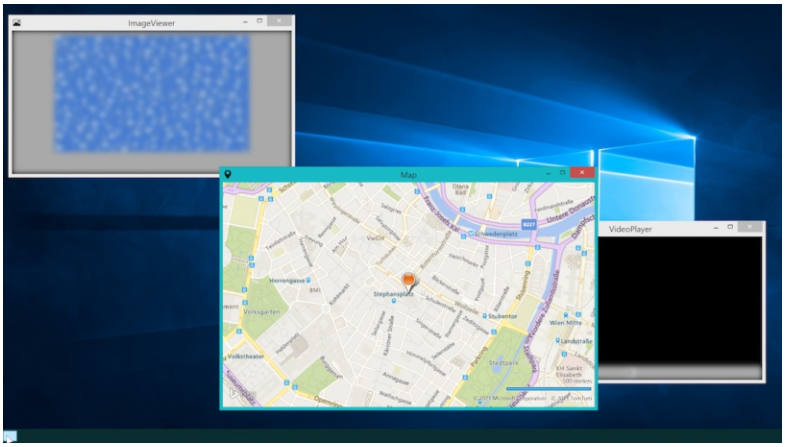


Abbildung 5.2: Übung 2: Anwendung wechseln

Übungsrunde 3

Das in dieser Übungsrunde vorgestellte Interaktionskonzept betrifft eine Tätigkeit welche heutzutage sehr häufig am Computer ausgeführt wird. Es geht hier um die Wiedergabe und das Pausieren eines Videos ohne die Maus oder Tastatur zu verwenden.

Beschreibung eines möglichen Szenarios

Dieses Konzept wird anhand zweier Szenarien beschrieben. Beim ersten Szenario handelt es sich um das Pausieren des Videos bei Präsenzverlust. Dabei pausiert das Video sobald sich der/die BenutzerIn vom Bildschirm abwendet und die Augen vom Eye-Tracker nicht mehr erfasst werden können. Nachdem der/die BenutzerIn wieder auf den Bildschirm oder auf das Video blickt setzt die Wiedergabe fort.
Das zweite Szenario beschreibt das Verhalten beim Fokussieren einer anderen Anwendung. Das Video pausiert sofort wenn der/die BenutzerIn auf eine andere Applikation blickt.

Aufgabenstellung für den/die BenutzerIn

Aufgabenbezeichnung:	Wiedergabe/Pause eines Videos
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt.
Aufgabe:	Sie möchten sich gerne ein Video ansehen und starten deshalb dessen Wiedergabe. Finden Sie hier nun selbst heraus, wie das Video via Eye-Tracking startet und pausiert. **Führen Sie diese Aufgabe zuerst via Eye-Tracking und danach mit der Maus durch. Nachdem Sie die Aufgabe mit der Eye-Tracking Steuerung erfolgreich abgeschlossen haben informieren Sie bitte Ihren Testleiter.**
Erwartetes Ergebnis:	Das Video startet/setzt fort oder pausiert bei entsprechender Steuerung.

Tabelle 5.3: Beschreibung Übung 3

Übungsrunde 4

Regelmäßig besteht der Bedarf eine Anwendung zu maximieren und wieder zu minimieren. Zu diesem Zweck, wurde eine Interaktionstechnik entwickelt welche genau das nur mit Hilfe des Eye-Trackers als alternative Eingabemethode ermöglicht.

Beschreibung möglicher Szenarien
Folgend werden einige Alltagssituationen beschrieben bei denen ein Maximieren der Anwendung Sinn macht.
Ein/Eine BenutzerIn möchte seinen Freunden Videos am Computer zeigen. Damit sie alle gut erkennen können vergrößert er/sie die Anwendung indem er/sie sich zurücklehnt.
Eine weitere Anwendung wäre eine Bildschirmpräsentation welche automatisch startet wenn man einen gewissen Abstand zum Bildschirm einnimmt.

Aufgabenstellung für den/die BenutzerIn

Aufgabenbezeichnung:	Anwendungsfenster maximieren und Ursprungsgröße wiederherstellen
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt.
Aufgabe:	Sie möchten sich auf eine Anwendung konzentrieren, da Sie mit dieser nun länger arbeiten werden. Aus diesem Grund maximieren Sie bitte das Bildbetrachtungsprogramm. Versuchen Sie den maximierten Modus wieder zu verlassen. **Führen Sie diese Aufgabe sowohl via Eye-Tracking als auch mit der Maus durch.**
Erwartetes Ergebnis:	Das Fenster ist abwechselnd maximiert, sodass es den gesamten Bildschirm einnimmt und dann wieder verkleinert auf die Größe vor der Maximierung.

Tabelle 5.4: Beschreibung Übung 4

5.2.7 Aufgaben

Nach der offenen Runde in welcher man sich mit dem Prinzip der Eye-Trackingkonzepte vertraut gemacht hat, folgt nun die Beschreibung der Aufgaben. Im Gegensatz zu den Übungsrunden ist nun der/die BenutzerIn auf sich allein gestellt.

Aufgabe 1

In dieser Aufgabe soll die Möglichkeit eine Anwendung zu wechseln demonstriert werden. Dabei wird das Konzept eines Applikationswechslers (siehe 4.1.1) für die Verwendung via Eye-Tracker umgesetzt.

Beschreibung eines möglichen Szenarios

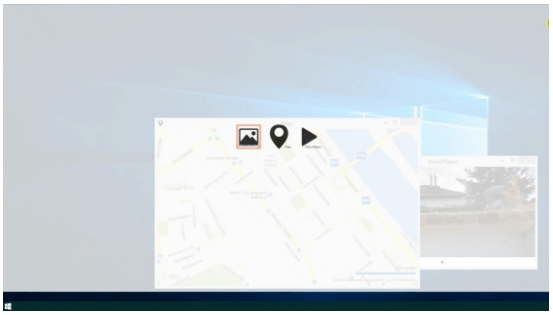
1. Die Programmfenster wurden bereits mit der Maus so verschoben, dass sich manche vollständig überlappen. Das bedeutet, dass man diese verdeckten Fenster mittels fokussieren nicht mehr erreichen kann.

2. Um dennoch das Fenster in den Vordergrund zu bringen und zu fokussieren, blickt man in die linke obere Ecke („Hot-Corner“, Abb. 5.3). Dadurch wird ein Programmwechsler geöffnet.

3. Nun blickt man auf das Symbol des zu öffnenden Programms und verweilt einen Augenblick darauf. Nach kurzer Zeit rückt das gewählte Programm in den Vordergrund

und erhält den Fokus.

(a) Aufgabe 1, linke obere Ecke betrachten



(b) Applikationswechsler, Icon betrachten um zu verdeckter Anwendung zu wechseln

Abbildung 5.3: Applikationswechsler Illustration

Aufgabenstellung für den/die BenutzerIn

Aufgabenbezeichnung:	Zu verdeckter Anwendung wechseln (via Applikationswechsler)
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt. Des Weiteren wird das Bildbetrachtungsprogramm von einem anderen Programm verdeckt.
Aufgabe:	Vor Ihnen sind mehrere Anwendungen geöffnet. Wechseln Sie zum Bildbetrachtungsprogramm. **Führen Sie diese Aufgabe zuerst via Eye-Tracking und danach mit der Maus durch. Nachdem Sie beide Steuerungsvarianten erfolgreich abgeschlossen haben informieren Sie bitte Ihren Testleiter.**
Wiederholungen:	3 Mal mit dem Eye-Tracker, 3 Mal mit der Maus
Erwartetes Ergebnis:	Das fokussierte Fenster rückt in den Vordergrund und die anderen Fenster rücken visuell in den Hintergrund.

Tabelle 5.5: Beschreibung Aufgabe 1

Aufgabe 2

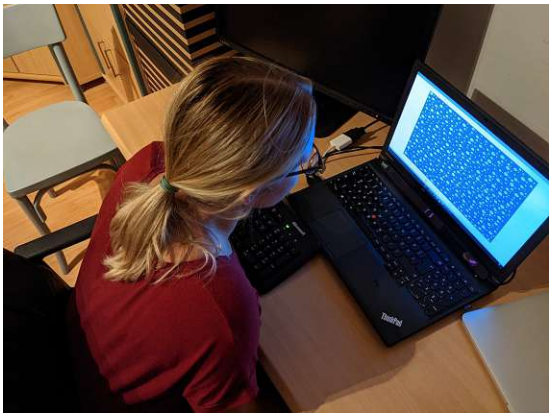
Um die Vergrößerungsfunktion leichter und einheitlicher zugänglich zu machen, wurde das Zoom-Konzept (siehe 4.1.2) entwickelt. Dieses Konzept arbeitet mit der Distanz des Kopfes zum Bildschirm um eine Vergrößerung (Abb. 5.4a) oder Verkleinerung (Abb. 5.4b) auszulösen.

Beschreibung eines möglichen Szenarios
Es kommt oft vor, dass man ein Bild vergrößern möchte, damit man Details besser erkennen kann. Zum Beispiel ist es bei der Nachbearbeitung von Bildern oft der Fall, dass man nur kleine Bereiche eines Bildes retuschieren muss. Danach muss man wieder herauszoomen um das Gesamtergebnis betrachten zu können. Dieser Vorgang verlangt somit dass sehr oft der Zoomfaktor geändert werden muss.

Aufgabenstellung für den/die BenutzerIn

Aufgabenbezeichnung:	Suchbild
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt.
Aufgabe:	Bei dieser Aufgabe handelt es sich um Suchbilder. Bei jeder dieser Aufgaben sollen Sie jenen Fisch finden, welcher in die entgegengesetzte Richtung zu allen anderen Fischen schwimmt. **Ihr Testleiter nennt Ihnen drei Aufgaben welche Sie via Eye-Tracking durchführen und drei zur Durchführung mit der Maus.**
Erwartetes Ergebnis:	Das Bild ist vergrößert dargestellt, sodass mehr Details erkennbar sind und der Fisch wurde gefunden.

Tabelle 5.6: Beschreibung Aufgabe 2



(a) Bild vergrößern.

(b) Bild verkleinern.

Abbildung 5.4: Suchbild Illustration

Aufgabe 3

Für diese Aufgabe wird die Abstandsmessung wieder zum Vergrößern und Verkleinern (siehe 4.1.2) eingesetzt. Die bereits erwähnten „Hot Borders“ sind in dieser Anwendung auch Bestandteil. Es wurde durch die Eye-Tracking Unterstützung versucht die Navigation auf einer digitalen Landkarte intuitiver und natürlicher zu gestalten (siehe 4.1.3).

Aufgabenstellung für den/die BenutzerIn

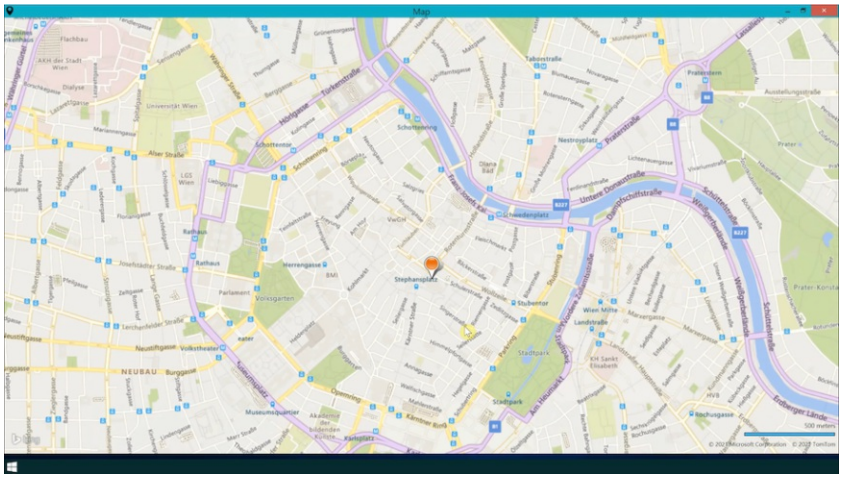


Abbildung 5.5: Aufgabe 3

Aufgabenbezeichnung:	Vergrößerung/Verkleinerung der Landkarte
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt.
Aufgabe:	Sie haben eine digitale Landkarte vor sich. Aktuell befinden Sie sich am Stephansplatz. Versuchen Sie nun in der Umgebung folgendes zu finden, wobei die Himmelsrichtung immer ausgehend vom Zentrum Wiens (Pin) ausgeht: A. Spanische Hofreitschule in Wien (süd-westlich) B. Wiener Prater (östlich) C. Zentralfriedhof im Bezirk Simmering (süd-östlich) D. Folgen Sie der Donau flussaufwärts bis zum Aubad bei Tulln E. Kopenhagen in Dänemark F. Wiener Rathaus (westlich) G. Augarten (nördlich) H. Tiergarten Schönbrunn im Bezirk Hietzing (süd-westlich) I. Folgen Sie der Donau flussabwärts bis zum Nationalpark Donau-Auen J. Lissabon in Portugal **Ihr Testleiter nennt Ihnen fünf Aufgaben welche Sie via Eye-Tracking durchführen und fünf zur Durchführung mit der Maus.**
Erwartetes Ergebnis:	Der/Die BenutzerIn findet die angegebenen Orte.

Tabelle 5.7: Beschreibung Aufgabe 3

Aufgabe 4

Auch in Aufgabe vier werden die Hot-Borders gut genutzt. Sie werden hier eingesetzt um zwei Applikationen in maximaler Größe nebeneinander anzeigen zu lassen.

Beschreibung eines möglichen Szenarios

Die ausführende Tätigkeit verlangt es, dauerhaft mit zwei Andwendungen zu arbeiten. Dabei wäre es von Vorteil, wenn beide Anwendungen direkt nebeneinander Platz finden. Um dies zu erreichen wechselt man in den „Splitscreen“. Man betrachtet dazu das Fenster welches man auf der rechten Seite platzieren will und blickt auf den rechten Rand. Das Fenster nimmt sich nun automatisch die verfügbare Größe des halben Bildschirms und man kann danach durch fokussieren der zweiten Anwendung, diese auf die linke Seite platzieren.

Aufgabenstellung für den/die BenutzerIn

Aufgabenbezeichnung:	Splitscreen
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt.
Aufgabe:	Fügen Sie die zwei angezeigten Bilder, bestehend aus einem linken und einem rechten Teil, zusammen um das Gesamtbild betrachten zu können. Dabei gibt es die folgenden Bildmotive zu lösen: A. Caipirinha B. Erdbeeren C. Kaffee D. Keller E. Schnee F. Seebühne **Ihr Testleiter nennt Ihnen drei Aufgaben welche Sie via Eye-Tracking durchführen und drei zur Durchführung mit der Maus.**
Erwartetes Ergebnis:	Es kann jeweils das Gesamtbild betrachtet werden.

Tabelle 5.8: Beschreibung Aufgabe 4

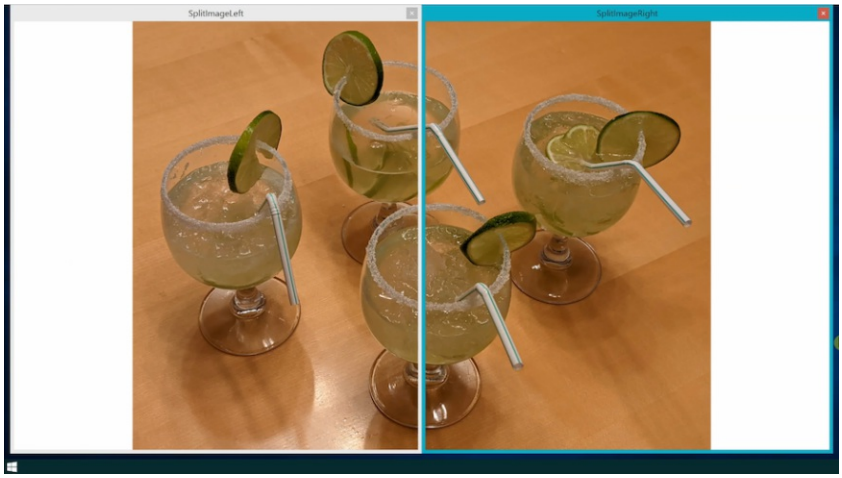


Abbildung 5.6: Splitscreen, erwartetes Ergebnis: Zwei Anwendungen nebeneinander geöffnet

Aufgabe 5

In einigen Berufsgruppen kommt es auf Grund der ausführenden Tätigkeiten am Computer zum Einsatz mehrerer Bildschirme. Zum Beispiel benötigen Software-Entwickler öfters zwei Monitore um auf einem das Fenster mit dem Programmcode zu sehen und auf dem anderen das Ergebnis und somit das fertige Programm. Ein weiteres Beispiel sind auch die Architekten, welche viel Platz zum Zeichnen ihrer Pläne brauchen. Somit kann es in all diesen Bereich dazu kommen, dass die diversen Anwendungsfenster zwischen den Monitoren hin und her verschoben werden müssen.

Beschreibung eines möglichen Szenarios

1. Zuerst fokussiert man das Fenster des Programms zur Betrachtung eines Bildes.
2. Danach wechselt der Blick wieder auf eine andere Anwendung.
3. Um jedes Programm auf einem separaten Bildschirm darzustellen, will man nun eines der Programmfenster auf den anderen Bildschirm verschieben.
4. Dazu blickt man zuerst auf das zu verschiebende Fenster, sodass es den Fokus erhält.
5. Danach wird, mit einer kurzen Verweildauer, der linke Bildschirmrand betrachtet.
6. Das Fenster ist nun auf den sekundären Monitor verschoben worden.

Aufgabenstellung für den/die BenutzerIn

Aufgabenbezeichnung:	Externer Monitor
Vorbedingungen:	Die Applikation wurde in ihren Ausgangszustand versetzt. Dem/Der BenutzerIn steht ein zusätzlicher Bildschirm für die Ausführung der Aufgabe zur Verfügung.
Aufgabe:	Sie haben zwei Bildschirme zur Verfügung. Verschieben Sie alle geöffneten Fenster von Bildschirm 1 (Laptop) auf Bildschirm 2 (zusätzlicher externer Bildschirm). Verwenden Sie dazu wieder die Anwendungen aus der Splitscreen Aufgabe 4.
Wiederholungen:	3 Mal mit dem Eye-Tracker, 3 Mal mit der Maus
Erwartetes Ergebnis:	Alle Fenster sind nun auf dem Bildschirm 2 sichtbar.

Tabelle 5.9: Beschreibung Aufgabe 5

5.3 Setup

Sowohl das Ausfüllen der Fragebögen als auch die Evaluierung des Prototypen findet in einem Heimbüro statt. Für die Durchführung der Aufgaben steht der Testperson ein Laptop zur Verfügung. Auf diesem Laptop ist Microsoft Windows 10 als Betriebssystem installiert. Die zu evaluierenden Konzepte sind als Prototyp bereits vorinstalliert und werden von jeder Testperson im selben Startzustand vorgefunden. Des Weiteren wird ein Monitor an den Laptop angeschlossen um die Multiscreen Konzepte zu evaluieren. Dieser wird direkt neben dem Laptop platziert. Der Eye-Tracker wird am unteren Bildschirmrand des Laptops mit Hilfe einer magnetischen Leiste montiert. Eine Maus und eine Tastatur dienen der Testperson bei bestimmten Aktionen als zusätzliche Eingabegeräte.

Abbildung 5.7: Setup

Abbildung 5.8 veranschaulicht das komplette Setup. In der Skizze sieht man den/die TeilnehmerIn an einem Schreibtisch sitzen. Der/Die TestleiterIn sitzt in einem bestimmten Winkel hinter der Testperson, sodass ein Einblick auf beide Bildschirme gewährleistet ist.

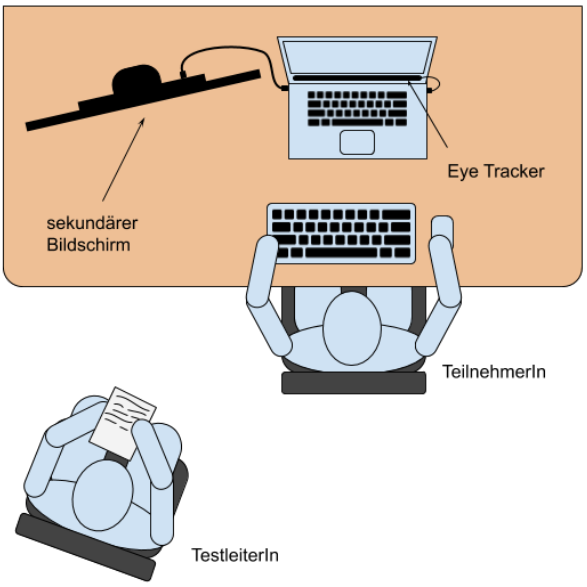


Abbildung 5.8: Beschreibung des Setups

5.4 Ablauf

In diesem Subkapitel wird der Ablauf der in dieser Arbeit durchgeführten Usability Evaluierung näher beschrieben. Bevor die eigentliche Evaluierung startet, wird dem/der BenutzerIn sowohl der Ablauf der Studie erklärt, als auch worum es sich bei Eye-Tracking handelt und wie es prinzipiell funktioniert. Danach stimmen die Benutzern/Benutzerinnen auf Grund der Datenschutzgrundverordnung (DSGVO) der anonymen Verwendung der Daten zu Forschungszwecken zu. Nach dieser kurzen Einführung beginnt der Hauptteil der Evaluierung, welche sich in folgende Abschnitte gliedert:

1. Pre-Test Fragebogen
2. Evaluierung des Prototypen
3. Interview
4. System Usability Scale (SUS) - Post-Test Fragebogen

Sowohl auf die Art der Fragebögen als auch auf den Aufbau der Fragebögen wird im Unterpunkt 5.4.1 gesondert eingegangen. Danach folgt eine genaue Ablaufbeschreibung

der Evaluierung des Prototypen. Dieser zweite Abschnitt (Evaluierung des Prototypen) wird in dieser Diplomarbeit wiederum in zwei Phasen aufgeteilt, die explorative Phase 5.4.2 und die Phase der Evaluierung mit zielgerichteten Aufgaben 5.4.2.

5.4.1 Fragebögen

Pre-Test Fragebogen

Jeder/Jede TeilnehmerIn wird zu Beginn der Studie mit Hilfe eines Pre-Test Fragebogens (8) zu seiner/ihrer Person befragt. Der Fragebogen wird gemeinsam mit der Testperson ausgefüllt und dabei werden unter anderem folgende demografischen Daten erhoben:

- Alter
- Geschlecht
- Beruf
- BrillenträgerIn
  Mit dieser Information kann nach der quantitativen und qualitativen Auswertung eine Aussage darüber gemacht werden, ob das tragen einer Brille Einfluss auf die Task-Completion Time oder Fehlerrate hat.

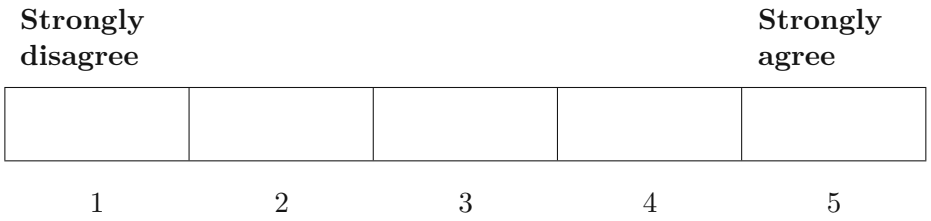
Neben diesen demografischen Daten werden auch Erfahrungen im Umgang mit einem Computer erfasst. Diese Fragen beziehen sich auf die hier aufgelisteten Bereiche:

- Häufigkeit
- Tätigkeit
- Betriebssystem

Diese Informationen können nach der Evaluierung hilfreich sein und dienen dazu bestimmte Verhaltensweisen der Testpersonen zu interpretieren.

System Usability Scale (SUS) - Post-Test Fragebogen

Nach der Evaluierung des Prototypen werden die TeilnehmerInnen gebeten einen weiteren Fragebogen auszufüllen. Als „Post-Test Fragebogen“ soll hier der standardisierte SUS Fragebogen, wie Brookee [Bro96] beschreibt, dienen. Dieser besteht, wie Brooke [Bro96] erläutert, aus fünf positiv und fünf negativ formulierten Aussagen bezüglich der Usability des evaluierten Systems. Dabei sollen die TeilnehmerInnen ihre Bewertungen nach ihrer ersten Tendenz beantworten, ohne lange darüber nachzudenken [Bro96].
Die Bewertungsskala für diesen SUS Fragebogen stellt Brooke [Bro96] folgendermaßen dar:



SUS Bewertungsskala, Quelle: Brooke [Bro96]

Der dazugehörige Fragebogen ist im Anhang dieser Diplomarbeit beigefügt. Für die Berechnung der SUS Bewertung gelten laut Brooke [Bro96] eigene Regeln. Brooke [Bro96] schreibt, dass zuerst alle Bewertungen aufsummiert werden müssen. Für die Bewertung muss nullbasiert gerechnet werden. Die Skala geht visuell von Eins bis Fünf, jedoch wird die Bewertung intern nullbasiert summiert (null bis vier). Das Ergebnis für eine positiv formulierte Aussage erhält man, indem man von der Skalenbewertung einer Aussage Eins subtrahiert. Für alle negativ formulierten Aussagen gilt es von Fünf die Bewertungszahl abzuziehen. Anschließend addiert man diese Ergebnisse und multipliziert das Resultat mit 2,5 um ein Gesamtergebnis zu erhalten (siehe [Bro96]).

5.4.2 Evaluierung des Prototypen

Kalibrierung

Bevor ein/eine BenutzerIn mit der Evaluierung der Interaktionskonzepte beginnen kann, muss der Eye-Tracker zuerst für ihn/sie kalibriert werden. Dazu wird die von Tobii bereitgestellte Kalibrierungssoftware benutzt. Durch die Kalibrierung kommt der/die TeilnehmerIn gleich zu Beginn mit dem Eye-Tracker, als ungewohntes Eingabegerät, in Berührung.
Wie bereits erwähnt wird dieser Abschnitt nochmal in zwei Phasen geteilt, welche in diesem Unterkapitel erläutert werden. Vor der Durchführung der Aufgaben und somit der zweiten Phase erhält der/die AnwenderIn eine kurze Demonstration und Beschreibung der Konzepte.

Explorative Evaluierung

In dieser Phase der Evaluierung wird der Testperson die Möglichkeit gegeben, den Prototypen selbst zu entdecken. Anfängliche Berührungsängste etwas falsch zu machen sollen der Testperson dabei genommen werden. Jeder soll selbstständig und explorativ die im Kapitel 4 genannten Interaktionskonzepte erforschen. Dies kann Spaß machen und den/die BenutzerIn motivieren, aber auch bei zu langer erfolgloser Entdeckungsphase zu Frustration führen. Deshalb wird diese Phase zeitlich auf maximal eine halbe Stunde begrenzt.
Wird ein Konzept durch kurze Entdeckungsphasen aufgenommen und verstanden, so

kann dies auf ein intuitives Konzept zurückgeführt werden. Wenn sich jedoch herausstellt, dass es den TeilnehmerInnen schwer fällt die Konzepte zu begreifen, oder sogar nur zu entdecken, dann deutet dies eventuell auf zu komplexe Interaktionskonzepte hin.

Demonstration der Konzepte

Wie schon zu Beginn dieses Unterkapitels unter 5.4.2 beschrieben, werden die Konzepte vor dem Hauptteil der Evaluierungsphase vorgestellt. Dabei wird dem/der BenutzerIn jedes Interaktionskonzept durch eine Demonstration vorgeführt.

Evaluierung mit zielgerichteten Aufgaben

Nachdem sich der/die TeilnehmerIn ein wenig mit der Anwendung vertraut gemacht hat, startet nach einer kurzen Pause von zehn Minuten die nächste Evaluierungsphase. In dieser Phase werden der Testperson die unter 5.2.7 vorgestellten Aufgaben kurz beschrieben und eventuell auftretende Fragen beantwortet.

Danach erfolgt die eigentliche Evaluierung der Eye-Tracking Interaktionskonzepte. Während der/die BenutzerIn die Aufgaben, beschrieben im Unterkapitel 5.2.7, durcharbeitet wird sie vom Testleiter beobachtet. Es werden Notizen bezüglich jeder einzelnen Aufgabe gemacht. Auch ein etwaiger Lerneffekt bei mehrfacher Ausführung einer Aufgabe soll dokumentiert werden.

Interview

Um auch qualitative Daten zu erhalten wird jede/r Teilnehmerin/Teilnehmer nach der Evaluierung einer Befragung unterzogen. Dabei werden folgende Fragen gestellt:

- Wie ist es Ihnen mit dem Eye-Tracker im Vergleich zur Maus ergangen?
- Welche Aufgaben, abgesehen von der Durchführungszeit, haben sich für Sie besser mit dem Eye-Tracker und welche besser mit der Maus angefühlt?
- Wie sind Ihre Eindrücke bezüglich der Eye-Tracking Technologie?

Die Antworten auf diese Fragen sollen Aufschluss darüber geben, ob und welche der entwickelten Interaktionstechniken positiv aufgenommen wurden. Auch die allgemeine Meinung der TeilnehmerInnen über die Eye-Tracking Technologie wird durch dieses Interview erhoben. Unabhängig von den gemessenen Metriken wie der Durchführungszeit und der Fehlerrate erfährt man, welche Konzepte sich gut und eventuell sogar besser als die Steuerung mittels Maus/Tastatur angefühlt haben.

KAPITEL 6

Ergebnisse der Evaluierung

Der erste Teil dieses Kapitels präsentiert die Auswertung der ausgefüllten demografischen Pre-Test Fragebögen. Im Anschluss folgt ein Subkapitel 6.2, welches für jede einzelne Aufgabe die Effektivität und die Effizienz mit Hilfe statistischer Auswertungen präsentiert. Am Ende dieses Subkapitels werden ergänzend die SUS Ergebnisse der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung als Metrik der Zufriedenheit verglichen. Qualitative Ergebnisse dieser Evaluierung bilden durch die Auswertung des Interviews den Abschluss dieses Kapitels.

Zu Beginn der Evaluierung wurde jedem/er TeilnehmerIn erläutert zu welchem Zweck diese Studie durchgeführt wird. Weiters wurde mit ausdrücklichem Hinweis erwähnt, dass nicht die Person sondern das System evaluiert wird. Es wurde darauf hingewiesen, dass im Zuge der Evaluierung der Bildschirminhalt inklusive Ton aufgezeichnet wird. Jeder/Jede TeilnehmerIn wurde darüber informiert, dass alle Daten anonymisiert und keine persönlichen Daten nach außen getragen werden.

6.1 Auswertung des Pre-Test Fragebogens

Wie schon im Kapitel 5.4.1 erläutert behandelt der Pre-Test Fragebogen die demografischen Daten der TeilnehmerInnen sowie zusätzliche Fragen zur Erfahrung im Umgang mit einem Computer. Alle erhobenen Daten der teilnehmenden Personen sind in den Tabellen 6.1 und 6.2 aufgelistet.
Insgesamt nahmen an der Studie zehn Personen teil, wobei diese aus fünf männlichen und fünf weiblichen TeilnehmerInnen bestanden. Ohne Ausnahme verwenden alle Personen mindestens fünf Mal pro Woche einen Computer für die unterschiedlichsten Bereiche. Dies ist auch beruflich bedingt, denn unter den TeilnehmerInnen finden sich die Berufe MaschinenbautechnikerIn, SoftwareentwicklerIn, BauleiterIn, ControllerIn, Marketing AssistentIn, SekretärIn, PsychologiestudentIn und SchülerIn wieder.

Person	Altersgruppe	Geschlecht	BrillenträgerIn	Beruf
1	30 - 39	m	nein	Programmierer
2	20 - 29	m	**ja**	BauleiterIn
3	20 - 29	w	**ja**	Marketing AssistentIn
4	20 - 29	w	nein	PsychologiestudentIn
5	<= 19	w	**ja**	SchülerIn
6	20 - 29	w	**ja**	ControllerIn
7	30 - 39	m	nein	ProgrammiererIn
8	50 - 59	w	**ja**	SekretärIn
9	30 - 39	m	nein	ProgrammiererIn
10	30 - 39	m	**ja**	MaschinenbauerIn

Tabelle 6.1: Demografische Daten der Teilnehmer

Wie man in der Tabelle 6.1 sieht, tragen sechs von zehn Testpersonen beim Bedienen eines Computers eine Brille. In der folgenden Abbildung 6.1 wird die Aufteilung der Altersgruppen dargestellt.

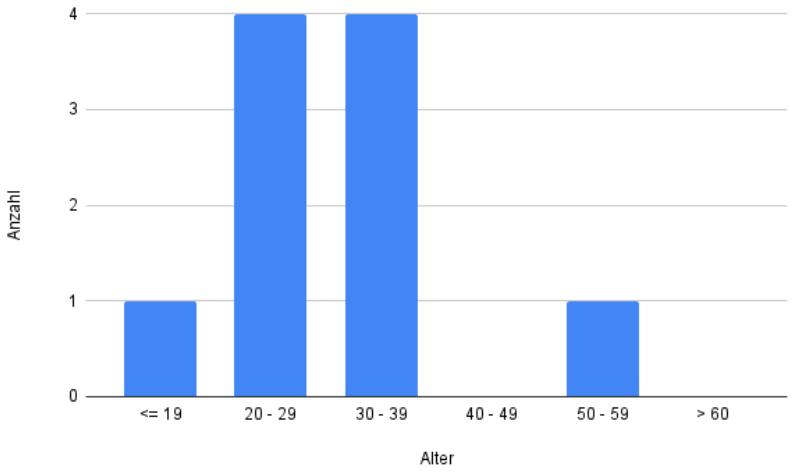


Abbildung 6.1: Alter der TeilnehmerInnen

In der Tabelle 6.2 werden unter anderem die von den TeilnehmerInnen genutzten Betriebssysteme aufgelistet. Acht von zehn Personen haben bisher die meiste Erfahrung mit dem Betriebssystem Microsoft Windows gesammelt, die anderen zwei TeilnehmerInnen sind sicherer im Umgang mit Apple's macOS. Da der Prototyp als WPF Anwendung umgesetzt wurde sind die meisten Steuerflächen den Windows BenutzerInnen vertraut.

Person	PC Nutzung p. Woche	Erfahrung (1-5)	Häufigste Tätigkeit	Betriebssystem
1	> 5 mal	5	Programmieren	Microsoft Windows
2	> 5 mal	3	Arbeiten, Excel	Microsoft Windows
3	> 5 mal	4	Social Media	Microsoft Windows
4	> 5 mal	5	Internet surfen	Microsoft Windows
5	> 5 mal	4	Social Media, Word	Apple macOS
6	> 5 mal	4	Excel	Microsoft Windows
7	> 5 mal	4	Programmieren	Apple macOS
8	> 5 mal	4	Textverarbeitung	Microsoft Windows
9	> 5 mal	5	Programmieren	Microsoft Windows
10	> 5 mal	4	Arbeiten, Spielen	Microsoft Windows

Tabelle 6.2: Erfahrung der Teilnehmer im Umgang mit einem Computer (Erfahrung: 1 = sehr gering, 5 = sehr hoch, siehe 8)

6.2 Auswertung der Evaluierung des Prototypen

Auf Grund des in dieser Studie gewählten „Within-Subject Designs“ wird jede Aufgabe von jedem/jeder TeilnehmerIn durchgeführt. Bezogen auf die Interaktion mit einem Computer mittels Augensteuerung handelt es sich hier für die meisten TeilnehmerInnen um eine völlig neue Erfahrung. Deshalb wurde entschieden mehrere Wiederholungen einer Aufgabe durchführen zu lassen, wobei ein dadurch entstehender Lerneffekt erwünscht war. Die erste Durchführung einer Aufgabe wurde somit immer als explorativer Durchgang bewertet. Nachdem in der ersten Iteration das Konzept entdeckt und verstanden wurde, waren die weiteren Wiederholungen für den Großteil der TeilnehmerInnen einfach zu lösen.

Die Unterkapitel gliedern sich hier in die einzelnen Aufgaben. Dabei wird für jede Aufgabe ein Vergleich zur Steuerung mittels Maus/Tastatur gezogen. Die in den folgenden Subkapitel präsentierten statistischen Auswertungen liefern Aussagen über folgende Metriken:

- Effektivität anhand der Fehlerrate
  - Wie schon beschrieben gab es pro Aufgabe mehrere Wiederholungen, wobei jede Iteration erfolgreich abgeschlossen wurde und es somit keinen Bedarf gibt eine Erfolgsrate auszuwerten.

    Da jede Wiederholung einer Aufgabe positiv, also erfolgreich, beendet wurde, ist eine Bewertung mit 1 im Erfolgsfall und 0 im Fehlerfall, sowie -1 bei einem Abbruch in dieser Evaluierung nicht möglich. Um dennoch eine Aussage über die Effektivität der entwickelten Interaktionstechniken treffen zu können, wurde das Maß der Fehlerrate herangezogen. Gezählt wurde dabei jedes für den/die

BenutzerIn unerwünschte Verhalten welches sowohl während der Evaluierung als auch mit Hilfe der Bildschirmaufzeichnungen mitgezählt wurde. Diese Fehlerrate wurde über jede einzelne Iteration einer Aufgabe erhoben.

- Effizienz mit Hilfe der Durchführungszeiten
  - In dieser Arbeit führten alle TeilnehmerInnen jede Aufgabe zuerst mit dem Eye-Tracker und danach mit der Maus und/oder Tastatur durch. Je nach Aufgabe gab es eine unterschiedliche Anzahl an Wiederholungen. Sehr oft passierte es, dass ein/eine TeilnehmerIn für den ersten Durchgang einer Aufgabe mit dem Eye-Tracker, aber auch mit Maus und Tastatur, deutlich länger brauchte als für die darauf folgenden Wiederholungen. Dies ist darauf zurückzuführen, dass das jeweilige Interaktionskonzept erst entdeckt und exploriert werden musste um die Aufgabe zu lösen. Die Durchführungszeit des ersten Durchgangs pro Aufgabe wurde deshalb als Ausreißer deklariert.
- Zufriedenheit durch die Auswertung des SUS Fragebogens
  - Nach der Beendigung der Evaluierung des Prototyps füllte jeder/jede TeilnehmerIn zwei SUS Fragebogen aus. Einmal für die Ausführung der Aufgaben mittels Eye-Tracker und dessen Interaktionskonzepte, sowie ein zweites Mal bezogen auf die Maus-/Tastatursteuerung. Die Resultate des standardisierten SUS Fragebogens wurden in diesem Kapitel miteinander verglichen.

6.2.1 Aufgabe 1 - Applikationswechsler

Effektivität - Fehlerrate

In dieser Aufgabe wurde die Fehlerrate ermittelt indem bestimmte Anwendungszustände gezählt wurden. Diese Anwendungszustände waren zum Beispiel Situationen in die sich der/die BenutzerIn gebracht hatte obwohl diese nichts mit der Aufgabe zu tun hatten. Bei dieser Aufgabe sollte die Person mittels Augensteuerung zu einer verdeckten Anwendung wechseln. Wurde eine Anwendung nun zum Beispiel in den Vollbildmodus versetzt oder der Splitscreen aus Aufgabe vier aktiviert, so wurde diese Situation als Fehler gezählt. Im Gegensatz zur Eye-Tracking Steuerung gab es bei der Durchführung mittels Maus und Tastatur keine einzige Situation welche als Wert für die Fehlerrate herangezogen werden konnte.

Das Ergebnis des t-Tests (Abb. 6.2) sagt aus, dass die Differenz der Fehlerrate zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung nicht signifikant war ($p > 0{,}05$). Die Hypothese H2.1 muss somit verworfen werden, es gilt die Nullhypothese.

Bei dieser Aufgabe wurden, wie man im Plot sieht (Abb. 6.4) fast keine Fehler gemacht. Der Mittelwert ist hier im Verhältnis 0,250 (Eye-Tracker) zu 0,000 (Maus/Tastatur).

Paired Samples T-Test

			Statistic	p
EERR	MERR	Wilcoxon W	3.00 [a]	0.371

[a] 18 pair(s) of values were tied

Abbildung 6.2: Fehlerrate T-Test (Applikationswechsler)

Descriptives

	N	Mean	Median	SD	SE
EERR	20	0.250	0.00	0.910	0.204
MERR	20	0.000	0.00	0.000	0.000

Abbildung 6.3: Fehlerrate, Mittelwert und Median (Applikationswechsler)

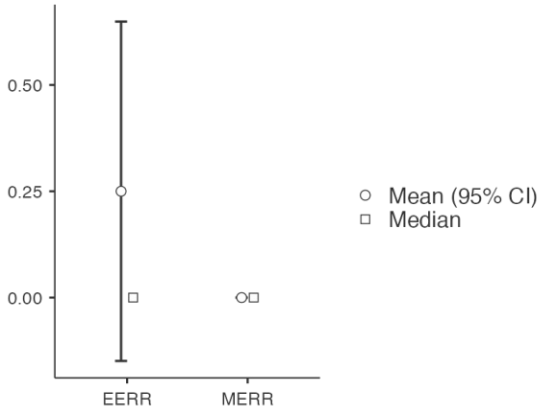


Abbildung 6.4: Fehlerrate, Mittelwert und Median Plot (Applikationswechsler)

Effizienz - Task-Completion Time

Um die Durchführungszeiten bei dieser Aufgabe zu protokollieren, wurde ein Start (Beginn der Zeitnehmung) und ein Ziel (Ende der Zeitnehmung) festgelegt. Nachdem die Anwendung auf die gewechselt werden soll hinter eine andere Applikation geschoben wurde, startete die Zeitnehmung sobald der/die BenutzerIn versuchte wieder zu dieser verdeckten Anwendung zu gelangen. Wenn die Testperson die Applikation wieder zum Vorschein gebracht hatte, wurde die Zeitnehmung gestoppt.

Der t-Test (Abb. 6.5) hat ergeben, dass die Differenz der Ausführungszeit mit dem Eye-Tracker und der Maus signifikant war ($p < 0{,}001$). Dies bedeutet das die Hypothese H2.2 für dieses Konzept als wahr angenommen werden kann.

Der Mittelwert (Abb. 6.6) der Durchführungszeit ist im Vergleich zur Maus-/Tastatursteuerung

Paired Samples T-Test

			Statistic	p
ET	MT	Wilcoxon W	153 [a]	<.001

[a] 3 pair(s) of values were tied

Abbildung 6.5: Task-Completion Time T-Test (Applikationswechsler)

(MT = 2,45) doppelt so hoch als bei der Steuerung mittels Eye-Tracking (ET = 5,10).

Descriptives

	N	Mean	Median	SD	SE
ET	20	5.10	5.00	2.47	0.552
MT	20	2.45	2.00	1.36	0.303

Abbildung 6.6: Task-Completion Time, Mittelwert und Median (Applikationswechsler)

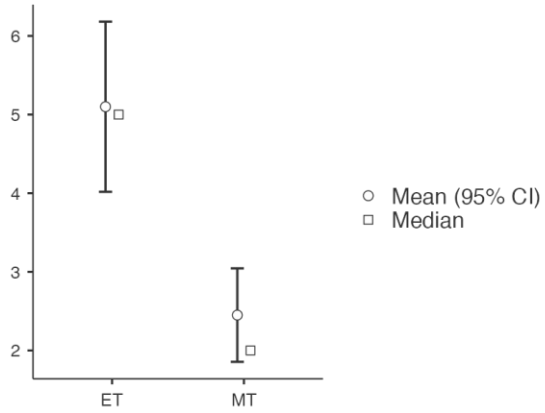


Abbildung 6.7: Task-Completion Time, Mittelwert und Median Plot (Applikationswechsler)

Die Anwendung des Applikationswechsler zeigt durch eine Varianzanalyse mit Messwiederholung bei der Eye-Tracking Steuerung einen signifikanten Unterschied ($p < 0{,}001$; Abb. 6.8). Mit einen Tukey-korrigierten Post Hoc Test (Abb. 6.9) erkennt man zwischen den Wiederholungen R1 und R2 ($p = 0{,}015$), sowie zwischen R1 und R3 ($p = 0{,}014$) einen Lerneffekt. Zwischen der zweiten und dritten (R2 und R3) Wiederholung ist die Differenz der Ausführungszeit nicht mehr signifikant ($p = 0{,}094$). Auf Grund des Lerneffekts kann die Hypothese H1 für dieses Konzept angenommen werden.

Within Subjects Effects

	Sum of Squares	df	Mean Square	F	p
RM Factor	58045	2	29023	12.9	<.001
Residual	40446	18	2247		

Note. Type 3 Sums of Squares

Abbildung 6.8: ANOVA mit Messwiederholung (Applikationswechsler)

Post Hoc Comparisons - RM Factor

Comparison							
RM Factor		RM Factor	Mean Difference	SE	df	t	p_{tukey}
R1	-	R2	92.50	25.906	9.00	3.57	0.015
	-	R3	94.10	26.012	9.00	3.62	0.014
R2	-	R3	1.60	0.670	9.00	2.39	0.094

Abbildung 6.9: Lerneffekt Post Hoc Test (Applikationswechsler)

6.2.2 Aufgabe 2 - Suchbild (Vergrößerung/Verkleinerung)

Effektivität - Fehlerrate

Bei der zweiten Aufgabe handelt es sich um das Vergrößern beziehungsweise Verkleinern eines Bildes. Beim unabsichtlichen Vergrößern oder Verkleinern des Bildes und der darauffolgenden Korrektur wurde ein Fehler protokolliert. Dieser Fehler wurde sowohl bei der Maussteuerung, wenn auch sehr gering, als auch bei der Steuerung mittels Eye-Tracking festgestellt. Hier passierten jedoch bei der Augensteuerung deutlich mehr Fehler, welche sich als Zoomsprünge, also mit direkt hintereinander folgenden Verkleinerungen und Vergrößerungen bemerkbar machten.

Beim Vergrößern/Verkleinern des Suchbildes gibt es bezüglich der Fehlerrate der Augensteuerung im Vergleich zur Maussteuerung, wie in der Abbildung 6.10 zu sehen ist, eine statistische Signifikanz ($p < 0{,}05$). In diesem Fall ist die Hypothese H2.1 als wahr anzunehmen.

Ein Diagramm zeigt uns in Abbildung 6.11, dass der Mittelwert der Fehlerrate bezüglich der Maus-/Tastatursteuerung ($N = 20$, $MERR = 0{,}150$) gegen Null geht.

Paired Samples T-Test

			Statistic	p
EERR	MERR	Wilcoxon W	55.0 [a]	0.005

[a] 10 pair(s) of values were tied

Abbildung 6.10: Fehlerrate T-Test (Suchbild)

Descriptives

	N	Mean	Median	SD	SE
EERR	20	1.400	1.00	1.569	0.3509
MERR	20	0.150	0.00	0.366	0.0819

Abbildung 6.11: Fehlerrate, Mittelwert und Median (Suchbild)

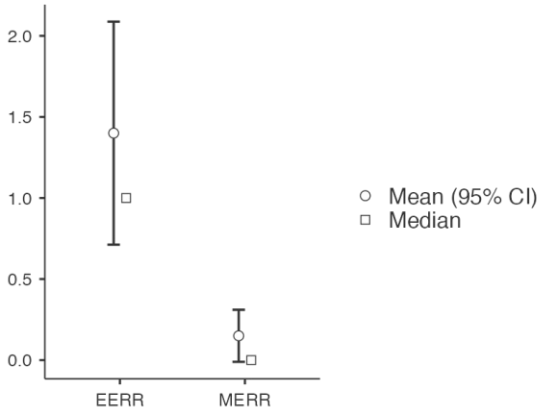


Abbildung 6.12: Fehlerrate, Mittelwert und Median Plot (Suchbild)

Effizienz - Task-Completion Time

Bei dieser Aufgabe wurde unmittelbar nach dem Erscheinen des Suchbilds die Zeitmessung gestartet. Erst nachdem der Fisch, welcher in die entgegengesetzte Richtung schwimmt, gefunden wurde, war die Aufgabe gelöst und die Zeitnehmung wurde gestoppt.

Der t-Test (Abb. 6.13) für die Task-Completion Time des Suchbildes hat beim Vergleich des Eye-Trackings und der Steuerung mittels Maus keine statistische Signifikanz ergeben ($p > 0{,}05$). Deshalb muss die Hypothese H2.2 für diese Aufgabe abgelehnt werden. Ein Bild mittels dieser Eye-Tracking Interaktionstechnik zu vergrößern oder verkleinern war somit in der Ausführungszeit nicht schneller als mit der Maus.

Der Vergleich des Mittelwertes (ET = 45,0; MT = 37,9) für Eye-Tracking und Maus,

Paired Samples T-Test

			Statistic	p
ET	MT	Wilcoxon W	128	0.411

Abbildung 6.13: Task-Completion Time T-Test (Suchbild)

zeigt in Abbildung 6.14 keinen großen Unterschied in der Durchführungszeit.

Descriptives

	N	Mean	Median	SD	SE
ET	20	45.0	42.0	31.0	6.92
MT	20	37.9	31.5	25.3	5.67

Abbildung 6.14: Suchbild Task-Completion Time, Mittelwert und Median

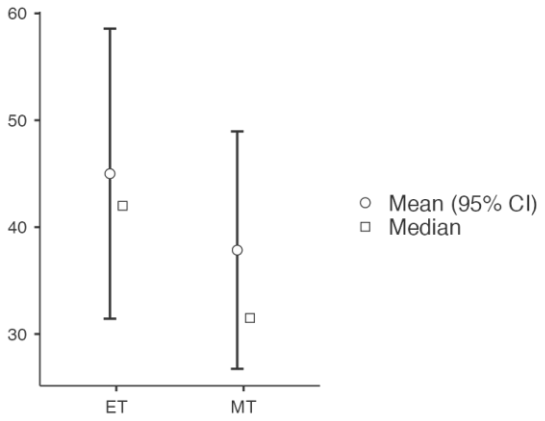


Abbildung 6.15: Suchbild Task-Completion Time, Mittelwert und Median Plot

Die Varianzanalyse mit Messwiederholung (Abb. 6.16) der Durchführungszeit, ergab sowohl für die Vergrößerung/Verkleinerung von Bildern mit Eye-Tracking als auch mit der Maus keine statistische Signifikanz ($p > 0{,}05$). Somit wird die Hypothese H1 für diese Interaktionstechnik abgelehnt, es gilt die Nullhypothese.

Des Weiteren erkennt man an den p-Werten ($p > 0{,}05$) des Tukey-korrigierten Post Hoc Tests (Abb. 6.17), dass kein Lerneffekt zwischen den Wiederholungen erkannt wurde.

Within Subjects Effects

	Sum of Squares	df	Mean Square	F	p
RM Factor	1009	2	505	0.505	0.612
Residual	17987	18	999		

Note. Type 3 Sums of Squares

Abbildung 6.16: ANOVA mit Messwiederholung (Applikationswechsler)

Post Hoc Comparisons - RM Factor

Comparison							
RM Factor		RM Factor	Mean Difference	SE	df	t	p_{tukey}
R1	-	R2	9.50	15.7	9.00	0.606	0.820
	-	R3	13.90	11.6	9.00	1.199	0.483
R2	-	R3	4.40	14.8	9.00	0.297	0.953

Abbildung 6.17: Lerneffekt Post Hoc Test (Suchbild)

6.2.3 Aufgabe 3 - Landkarte

Diese Aufgabe verlangt, dass der/die BenutzerIn ein Gebäude, Orte, Plätze und Städte auf einer Landkarte findet und anschließend in der Anwendung zentriert.

Effektivität - Fehlerrate

Als Fehler wird hierbei folgendes protokolliert:

- die Navigation in eine falsche Richtung
- die unerwünschte Vergrößerung oder Verkleinerung

Beide Fehler wurden zum Einen bei der Maussteuerung und zum Anderen beim Eye-Tracking Interaktionskonzept registriert. Wie schon erwähnt machen auch hier die ungewollten Zoomsprünge Probleme beim Navigieren. Für die Landkarte wurde die Bing Map von Microsoft eingesetzt und dabei wurden alle verfügbaren Zoomstufen auf die Distanz des Kopfes zum Eye-Tracker gelegt.

Anhand des t-Tests (Abb. 6.18) wurde erkannt, dass die Differenz der Fehlerrate des Navigierens mit dem Eye-Tracker und der Maus signifikant ($p < 0{,}001$) war. Die Hypothese H2.1 wird somit als wahr angenommen.

Paired Samples T-Test

			Statistic	p
EERR	MERR	Wilcoxon W	479 [a]	<.001

[a] 8 pair(s) of values were tied

Abbildung 6.18: Fehlerrate T-Test (Landkarte)

Der Mittelwert der Fehlerrate für Eye-Tracking (EERR = 3,850;Abb. 6.19) ist im Vergleich zur Maussteuerung (MERR = 0,900) mehr als viermal so hoch.

Descriptives

	N	Mean	Median	SD	SE
EERR	40	3.850	2.00	5.22	0.825
MERR	40	0.900	1.00	1.03	0.163

Abbildung 6.19: Fehlerrate, Mittelwert und Median (Landkarte)

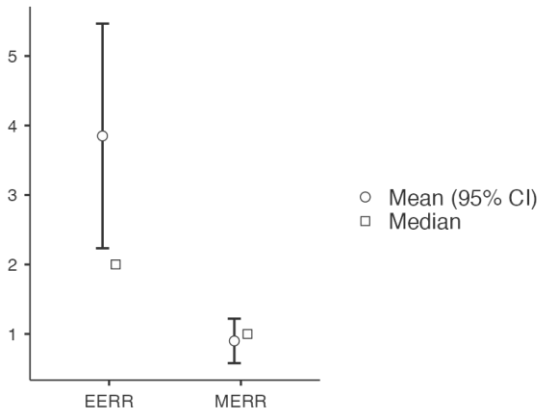


Abbildung 6.20: Fehlerrate, Mittelwert und Median Plot (Landkarte)

Effizienz - Task-Completion Time

Bevor die Zeitmessung gestartet wurde, wurde die Karte immer auf den gleichen Standort zurückgesetzt. Eine Aufgabe gilt als erledigt, sobald der/die BenutzerIn das Gebäude, den Ort, den Platz oder die Stadt auf der Karte gefunden und auf dem Bildschirm zentriert hat. Genau zu diesem Zeitpunkt wurde auch die Zeitnehmung wieder gestoppt. Die Zeitmessung wurde bei etwaigen Fehlern des/der Benutzers/Benutzerin nicht angehalten.

Der t-Test (Abb. 6.21) ergab dass die Differenz der Durchführungszeiten der Eye-Tracking

und Maus-/Tastatursteuerung signifikant war ($p < 0{,}001$). Somit wird die Hypothese H2.2 als wahr angenommen.

Paired Samples T-Test

			Statistic	p
ET	MT	Wilcoxon W	614 [a]	<.001

[a] 2 pair(s) of values were tied

Abbildung 6.21: Task-Completion Time T-Test (Landkarte)

Der Mittelwert (Abb. 6.22) für die Durchführung mit dem Eye-Tracker (ET = 60,1) zeigt bei der Zeitmessung einen großen Sprung im Vergleich zum Mittelwert der Maussteuerung (MT = 25,7).

Descriptives

	N	Mean	Median	SD	SE
ET	40	60.1	47.5	48.5	7.67
MT	40	25.7	12.0	31.7	5.02

Abbildung 6.22: Landkarte Task-Completion Time, Mittelwert und Median

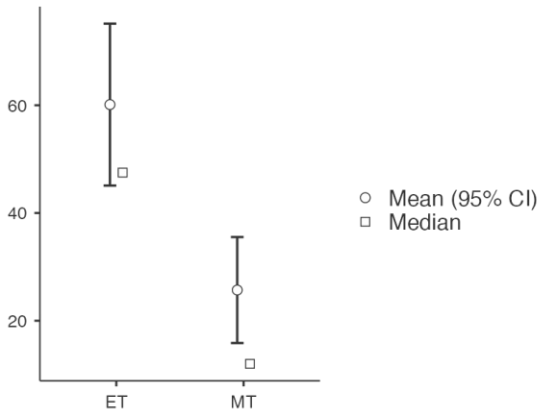


Abbildung 6.23: Landkarte Task-Completion Time, Mittelwert und Median Plot

Eine Varianzanalyse mit Messwiederholung (Abb. 6.24) hat ergeben, dass die Differenz der Ausführungszeiten zwischen den Wiederholungen für den Eye-Tracker nicht signifikant war ($p > 0{,}05$).

Within Subjects Effects

	Sum of Squares	df	Mean Square	F	p
RM Factor	50390	4	12598	2.37	0.071
Residual	191666	36	5324		

Note. Type 3 Sums of Squares

Abbildung 6.24: ANOVA mit Messwiederholung (Landkarte)

Des Weiteren konnte keine Signifikanz ($p > 0{,}05$) für einen Lerneffekt zwischen den Wiederholungen durch einen Tukey-korrigierten Post Hoc Test (Abb. 6.25) festgestellt werden. Es wird somit die Nullhypothese angenommen und die Hypothese H1 verworfen.

Post Hoc Comparisons - RM Factor

Comparison							
RM Factor		RM Factor	Mean Difference	SE	df	t	p_{tukey}
R1	-	R2	84.20	47.8	9.00	1.7622	0.447
	-	R3	67.80	45.1	9.00	1.5035	0.585
	-	R4	85.40	40.7	9.00	2.0969	0.299
	-	R5	45.70	45.1	9.00	1.0128	0.843
R2	-	R3	-16.40	26.3	9.00	-0.6226	0.968
	-	R4	1.20	20.0	9.00	0.0601	1.000
	-	R5	-38.50	24.7	9.00	-1.5608	0.553
R3	-	R4	17.60	14.9	9.00	1.1808	0.762
	-	R5	-22.10	21.2	9.00	-1.0401	0.831
R4	-	R5	-39.70	16.2	9.00	-2.4538	0.185

Abbildung 6.25: Lerneffekt Post Hoc Test (Landkarte)

6.2.4 Aufgabe 4 - Splitscreen

Bei dieser Aufgabe war das Ziel ein Bild welches aus zwei Teilen bestanden hat wieder zu einem ganzen Bild zusammenzufügen. Jeder Bildteil war in einem eigenen Programmfenster. Die Aufgabe war gelöst, sobald man den linken Teil auf die linke Hälfte und den rechten Teil auf die rechte Hälfte des Bildschirms positioniert hat.

Effektivität - Fehlerrate

Für die Steuerung mittels Eye-Tracking wurde ein Fehler protokolliert sobald

- ein Bild falsch positioniert
- eine falsche Aktion ausgelöst

wurde. Die gleichen Fehler wurden bei der Maus-/Tastatursteuerung protokolliert.

Betrachtet man das Ergebnis der Fehlerrate mit Hilfe eines t-Tests (Abb. 6.26) für die Splitscreen Aufgabe, erkennt man dass die Differenz zwischen der Eye-Tracking und Maussteuerung nicht statistisch signifikant ist. Die Hypothese H2.1 wurde damit für diese Aufgabe nicht bestätigt.

Paired Samples T-Test

			Statistic	p
EERR	MERR	Wilcoxon W	24.0 [a]	0.430

[a] 12 pair(s) of values were tied

Abbildung 6.26: Fehlerrate T-Test Splitscreen

Der Median der Fehlerrate ist in dieser Aufgabe für beide Steuerungstechniken 0,00. Dies bedeutet, dass bei 20 Messungen (N = 20) mindestens zehn Wiederholungen bei Eye-Tracking und Maus-/Tastatursteuerung fehlerfrei waren (Abb. 6.27). Des Weiteren sind auch die Werte, wie man in der Grafik sieht, überlappend, was auf eine annähernd gleiche Fehleranzahl hindeutet.

Descriptives

	N	Mean	Median	SD	SE
EERR	20	0.650	0.00	1.226	0.274
MERR	20	0.350	0.00	0.813	0.182

Abbildung 6.27: Splitscreen Fehlerrate, Mittelwert und Median

Effizienz - Task-Completion Time

Wie auch bei den anderen Anwendungen erfolgt die Zeitmessung immer nachdem der Ausgangszustand hergestellt wurde. Die Stoppuhr wird angehalten sobald die richtigen Fenster auf ihren Positionen sind.

In dieser Aufgabe konnte keine Signifikanz für die Differenz der Durchführungszeiten bezüglich der Eye-Tracking Interaktion und der Maus-/Tastatursteuerung über den t-Test ($p > 0.05$) festgestellt werden (6.29). Aus diesem Grund wird die Nullhypothese angenommen und die Alternativhypothese H2.2 verworfen.

Im Plot (Abb. 6.31) sieht man sehr gut die deutliche Überlappung der Durchführungszeiten der Eye-Tracking Steuerung und der Steuerung über die Maus. Daraus erkennt man,

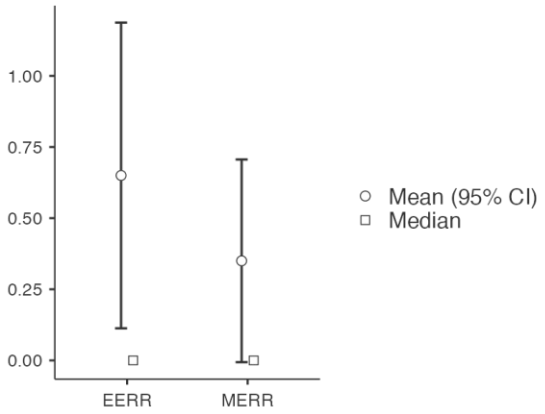


Abbildung 6.28: Splitscreen Fehlerrate, Mittelwert und Median Plot

Paired Samples T-Test

			Statistic	p
ET	MT	Wilcoxon W	122	0.538

Abbildung 6.29: Task-Completion Time T-Test (Splitscreen)

dass die Durchführungszeiten annähernd gleich sind.

Descriptives

	N	Mean	Median	SD	SE
ET	20	20.1	12.0	24.8	5.54
MT	20	14.3	10.0	11.7	2.62

Abbildung 6.30: Splitscreen Task-Completion Time, Mittelwert und Median

Keine statistische Signifikanz ($p > 0{,}05$) zeigte die Varianzanalyse mit Messwiederholung (Abb. 6.32) für das Splitscreen Konzept. Für die Eye-Tracking Splitscreen Interaktionstechnik gilt somit die Nullhypothese, die Hypothese H1 wird abgelehnt.

Anhand des Post-Hoc Tests (Abb. 6.33) erkennt man von Runde zu Runde auch keine Lerneffekte für die Splitscreen Aufgabe.

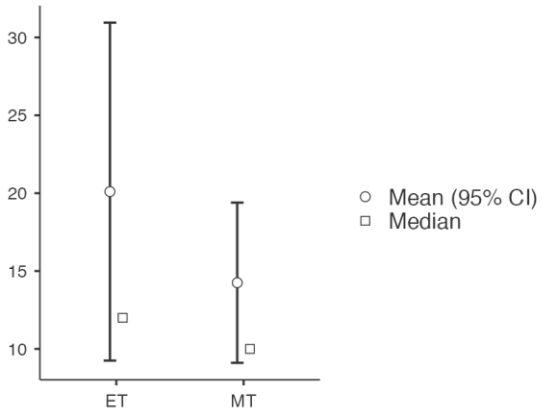


Abbildung 6.31: Splitscreen Task-Completion Time, Mittelwert und Median Plot

Within Subjects Effects

	Sum of Squares	df	Mean Square	F	p
RM Factor	11926	2	5963	2.65	0.098
Residual	40580	18	2254		

Note. Type 3 Sums of Squares

Abbildung 6.32: ANOVA mit Messwiederholung (Splitscreen)

Post Hoc Comparisons - RM Factor

Comparison							
RM Factor		RM Factor	Mean Difference	SE	df	t	p_{tukey}
R1	-	R2	46.40	22.9	9.00	2.030	0.160
	-	R3	36.40	26.2	9.00	1.387	0.387
R2	-	R3	-10.00	11.9	9.00	-0.839	0.690

Abbildung 6.33: Lerneffekt Post Hoc Test (Splitscreen)

6.2.5 Aufgabe 5 - Externer Monitor

Wie bereits erwähnt mussten die Testpersonen hier Anwendungen vom primären Monitor auf einen externen Monitor verschieben.

Effektivität - Fehlerrate

Für die Steuerung mittels Eye-Tracking und für die Maus-/Tastatursteuerung wurde ein Fehler protokolliert sobald eine falsche Aktion ausgelöst wurde. Bei der Maus- und Tastatursteuerung wurde für folgende Situationen ebenfalls ein Fehler vermerkt:

- der Klick mit der Maus auf das Fenster daneben ging
- das Anwendungsfenster zu früh wieder losgelassen wurde (Anwendung noch nicht vollständig am externen Bildschirm sichtbar)

Mit Hilfe eines t-Tests (Abb. 6.34) wurde festgestellt, dass die Differenz zwischen Maus-/Tastatursteuerung und Eye-Tracking Steuerung nicht signifikant war ($p > 0{,}05$). Aus diesem Grund muss die Hypothese H2.1 abgelehnt werden, es gilt die Nullhypothese.

Paired Samples T-Test

			Statistic	p
EERR	MERR	Wilcoxon W	4.00 [a]	0.073

[a] 13 pair(s) of values were tied

Abbildung 6.34: Fehlerrate T-Test Externer Monitor

Beim Plot der Fehlerrate des externen Monitors (Abb. 6.36) sieht man eine Überlappung. Des Weiteren sieht man, dass über die Eye-Tracking Steuerung annähernd gleich wenig Fehler gemacht wurden wie mit der Maus.

Descriptives

	N	Mean	Median	SD	SE
EERR	20	0.100	0.00	0.308	0.0688
MERR	20	0.350	0.00	0.587	0.1313

Abbildung 6.35: Externer Monitor Fehlerrate, Mittelwert und Median

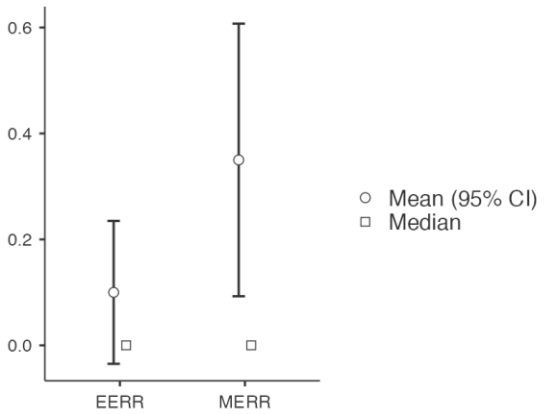


Abbildung 6.36: Externer Monitor Fehlerrate, Mittelwert und Median Plot

Effizienz - Task-Completion Time

Bei der Messung der Durchführungszeit eines Durchgangs, wurde die Zeitnehmung nach dem vollständigen Erscheinen aller Anwendungen gestartet. Sobald alle Programmfenster auf den externen Bildschirm verschoben wurden und alle Anwendungen zur Gänze sichtbar waren, wurde die Zeitnehmung wieder gestoppt.

Der t-Test (Abb. 6.37) der Task-Completion Time des externen Monitors sagt aus, dass der Unterschied der Eye-Tracking Steuerung und der Steuerung mittels Maus signifikant ist ($p < 0{,}001$). Die Hypothese H2.2 wird somit für dieses Interaktionskonzept angenommen.

Paired Samples T-Test

			Statistic	p
ET	MT	Wilcoxon W	190 [a]	<.001

[a] 1 pair(s) of values were tied

Abbildung 6.37: Task-Completion Time T-Test (Externer Monitor)

Wenn man den Mittelwert bezüglich der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung gegenüberstellt, so sieht man in Abbildung 6.38 deutlich, dass die Werte der Durchführungszeit für das Eye-Tracking Konzept mehr als doppelt so hoch sind.

Descriptives

	N	Mean	Median	SD	SE
ET	20	18.10	16.00	6.03	1.349
MT	20	9.70	9.50	2.96	0.661

Abbildung 6.38: Externer Monitor Task-Completion Time, Mittelwert und Median

In Abbildung 6.40 erkennt man eine statistische Signifikanz ($p < 0{,}001$) bei den Messwiederholungen. Betrachtet man diese Daten zusätzlich mit einem Tukey-korrigierten Post Hoc Test (Abb. 6.41), so sieht man, dass sowohl zwischen den Durchführungsrunden R1 und R2 ($p = 0{,}041$) als auch zwischen R1 und R3 ($p = 0{,}014$) und zwischen R2 und R3 ($p = 0{,}021$) eine Signifikanz, bezogen auf den Lerneffekt, aufgetreten ist. Aus diesem Grund wird die Hypothese H1 für diese Interaktionstechnik angenommen.

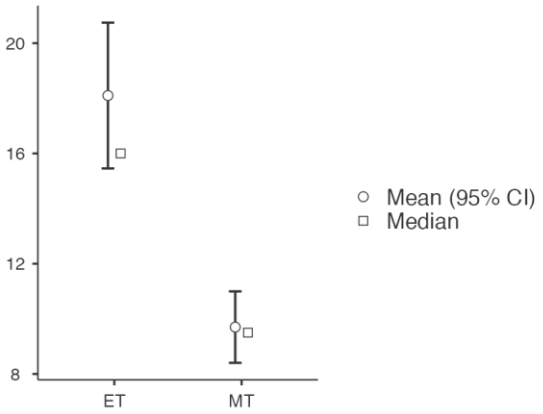


Abbildung 6.39: Externer Monitor Task-Completion Time, Mittelwert und Median Plot

Within Subjects Effects

	Sum of Squares	df	Mean Square	F	p
RM Factor	4874	2	2437	10.9	<.001
Residual	4031	18	224		

Note. Type 3 Sums of Squares

Abbildung 6.40: ANOVA mit Messwiederholung (Externer Monitor)

Post Hoc Comparisons - RM Factor

Comparison							
RM Factor		RM Factor	Mean Difference	SE	df	t	p_{tukey}
R1	-	R2	23.40	8.02	9.00	2.92	0.041
	-	R3	29.60	8.16	9.00	3.63	0.014
R2	-	R3	6.20	1.84	9.00	3.36	0.021

Abbildung 6.41: Lerneffekt Post Hoc Test (Externer Monitor)

6.2.6 Vergleich der Zufriedenheit - Punktezahl des SUS Fragebogens

Um eine Gesamtbewertung über die Zufriedenheit der Testpersonen (p1 - p10) zu erhalten, wurde wie erwähnt, ein SUS Fragebogen von allen TeilnehmerInnen ausgefüllt. Da wir in dieser Arbeit einen Vergleich zwischen der Eye-Tracking Steuerung und der Steuerung mittels Maus/Tastatur durchführen, wurde der SUS Fragebogen auch für beide Steuerungsvarianten ausgefüllt und ausgewertet.
Betrachtet man nun die Auswertung (Abb. 6.42) des SUS Fragebogens für die Steuerung mittels Maus/Tastatur, so existiert nur ein Wert unter 68 von Person p10. Um das errechnete SUS Ergebnis besser vergleichen zu können, beschreibt Sauro [Sau11] in seinem

Buch die Interpretation des SUS Ergebnisses als Prozentrang. Sauro [Sau11] hat dazu das SUS Ergebnis von 500 verschiedenen Evaluierungen herangezogen und davon das durchschnittliche SUS Ergebnis errechnet. Er kam dabei auf einen Mittelwert von 68 (entspricht laut Sauro [Sau11] einer höher wahrgenommenen Usability gegenüber 50% aller bereits getesteten Produkte). Diesen Wert legte er als Schulnotenwert C fest, wobei A die beste Note ist und F die schlechteste (Bewertung von A-F).

Für die Maus-/Tastatursteuerung ergibt die Auswertung des SUS Fragebogens gesamt über alle TeilnehmerInnen gemittelt ein Ergebnis von 84 (zirka 95% = Note: „A“ nach Sauro [Sau11]).

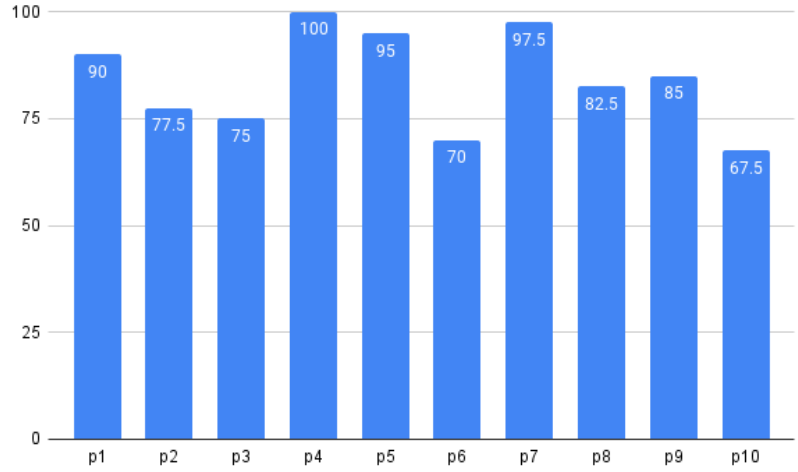


Abbildung 6.42: SUS Ergebnis Maus-/Tastatursteuerung

Die gleiche Auswertung wurde für die Eye-Tracking Steuerung durchgeführt. In der Abbildung 6.43 sieht man im Vergleich zu Maus-/Tastatursteuerung insgesamt vier Personen, bei denen der SUS Wert unter den Mittelwert von 68 fällt. Der Höchstwert (Person p9) liegt bei 85.
Der Gesamtwert des SUS Ergebnisses berechnet sich auf 70,5 (zirka 56% = Note: „B minus“ nach Sauro [Sau11]).

Auf Grund dieses Ergebnisses muss die Hypothese H3 abgelehnt werden, es gilt die Nullhypothese.

6.3 Auswertung des Interviews

Um auch qualitative Daten zu erheben, wurde zusätzlich zum SUS Fragebogen ein Interview mit den TeilnehmerInnen geführt. Nach der Durchführung der Aufgaben

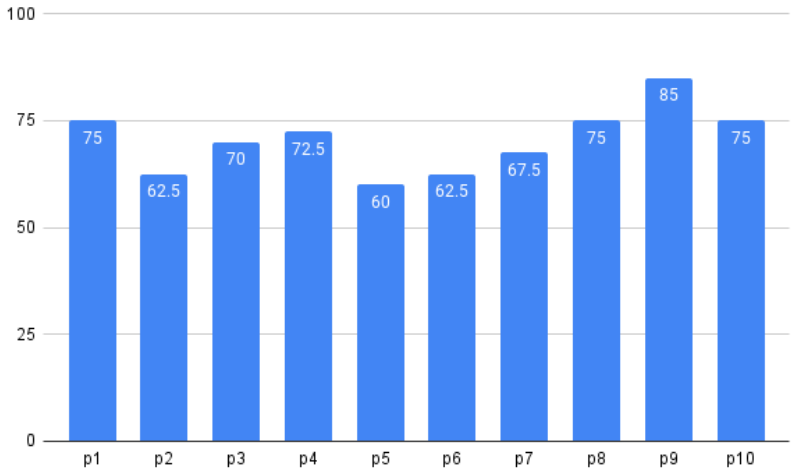


Abbildung 6.43: SUS Ergebnis Eye-Tracking Steuerung

wurden den Benutzern/Benutzerinnen drei Fragen zum Thema Eye-Tracking und den einzelnen Aufgaben/Konzepten gestellt. Die Antworten zu diesen Fragen wurden in den folgenden Subkapiteln analysiert und ausgewertet.

6.3.1 Vergleich Eye-Tracker vs. Maus

Die BenutzerInnen beantworteten folgende Frage:

Wie ist es Ihnen mit dem Eye-Tracker im Vergleich zur Maus ergangen?

Alle InterviewpartnerInnen wurden gebeten diese Frage für jede der durchgeführten Aufgaben separat zu beantworten. Aus diesem Grund wurde die Auswertung der Antworten folgend auch danach strukturiert.

Eindrücke zum Applikationswechsler

Bei dieser Aufgabe wie auch bei anderen kam sehr oft die Antwort, dass die Bedienung sehr einfach war sobald die Interaktionstechnik entdeckt und verstanden wurde. Sieben von zehn Personen bevorzugen hier jedoch die Taskleiste welche sie mit der Maus ansteuern um zu einer anderen Anwendung zu wechseln. Ein/Eine InterviewpartnerIn erwähnte, dass seiner/ihrer Meinung nach dieses Konzepte nur bei wenigen geöffneten Anwendungen Sinn macht, da man ansonsten eventuell die gewünschte Anwendung nicht mehr so leicht findet. Es wurde außerdem ein Konzept vorgeschlagen bei welchem man die Anwendungen in der Taskleiste auch mittels Eye-Tracker fokussieren kann um so zur gewünschten Anwendung zu wechseln.

Eindrücke zum Suchbild (Vergrößern/Verkleinern)

Im Gegensatz zur ersten Aufgabe, wirkte dieses Konzept intuitiver und die Funktionsweise war somit fast allen TeilnehmerInnen von Anfang an klar. Die Möglichkeit den Eye-Tracker beziehungsweise das Auslösen der Aktionen zu pausieren beziehungsweise zu sperren, war für die Hälfte der BenutzerInnen hier eine wesentlich fehlende Funktion. Dieser Wunsch wurde mit der Begründung geäußert, dass man ohne dieser Funktion starr sitzen muss weil man bei einer Veränderung der Distanz des Kopfes zum Bildschirm auch den Zoomlevel des Bildes verändert. Bezüglich der Geschwindigkeit des Vergrößerns und Verkleinerns sprachen sich die meisten TeilnehmerInnen für die Eye-Tracking Steuerung aus und empfanden es viel schneller als mit der Maus. Dass der Zoomlevel bei gleichbleibender Distanz zum Bildschirm sofort übernommen wird, wurde ebenfalls als sehr positiv in Bezug auf die Geschwindigkeit gesehen.

Eindrücke zur digitalen Landkarte

Eines der Hauptprobleme dieser Aufgabe beziehungsweise des Konzepts, welches fast alle BenutzerInnen beim Beantworten dieser Frage erwähnten, ist wiederum die Sensibilität des Zoomverhaltens. Das Vergrößern und Verkleinern reagiert laut Aussagen der TeilnehmerInnen viel zu sensitiv und unkontrolliert, was bedeutet dass sehr viele ungewollte „Zoom-Sprünge“ passieren, ähnlich wie bei der Suchbild Aufgabe. Auch hier wurde auf Grund dessen wieder der Wunsch nach der bereits erwähnten Sperrfunktion für den Eye-Tracker geäußert. Des Weiteren bemerkte eine Person, dass zu langsame Bewegungen anfangs keine Veränderung des Zoomlevels bewirkten und dann plötzlich ein großer Sprung erfolgte. Als zusätzliche Hilfe hätte sich eine der Testpersonen einen grafischen Hinweis bei sensitiven Bereichen gut vorstellen können, wobei alle Bereiche der Karte gemeint sind, welche eine Aktion auslösen sobald sie betrachtet werden. Knapp die Hälfte der TeilnehmerInnen denkt jedoch, dass es nur eine Frage des Trainings ist um das Navigieren mittels Eye-Tracker gut zu beherrschen. Gesamt betrachtet funktioniert die Steuerung der Karte für die meisten BenutzerInnen mit der Maus besser und auch schneller.

Eindrücke zum Splitscreen

Bis auf zwei BenutzerInnen sind bei dieser Aufgabe alle von der Interaktionstechnik überzeugt. Für einen/eine dieser zwei TeilnehmerInnen war es schwierig überhaupt herauszufinden wie diese Aufgabe zu bewerkstelligen ist. Der/Die andere BenutzerIn kommt gut mit der Maus zurecht und möchte sich nicht auf dieses Eye-Tracking Konzept umgewöhnen. Sowohl die Geschwindigkeit, um zwei Anwendungen am Bildschirm nebeneinander anzuzeigen, als auch die einfache Bedienung des Konzepts wurde vielfach gelobt.

Eindrücke zum Verschieben einer Anwendung auf den externen Bildschirm

Alle TeilnehmerInnen fanden das Interaktionskonzept des Eye-Trackers bei dieser Aufgabe sehr gelungen und passend. Der grafische Hinweis der Hot-Borders hat hier sehr geholfen.

Auf Grund der Gewohnheit ziehen zwei BenutzerInnen trotzdem die Maus dem Eye-Tracker vor. Ungefähr ein Drittel der Testpersonen meinte, dass die Ausführung dieser Aufgabe mit dem Eye-Tracker schneller war, da man mit der Maus das Anwendungsfenster über den ganzen Bildschirm ziehen musste. Einen klaren Geschwindigkeitsvorteil sieht einer der TeilnehmerInnen für BenutzerInnen, welche mit der Maus eher ungeübt sind.

6.3.2 Bevorzugte Steuerung

Es wurde folgende Frage gestellt:

> *Welche Aufgaben, abgesehen von der Durchführungszeit, haben sich für Sie besser mit dem Eye-Tracker und welche besser mit der Maus angefühlt?*

Beim Applikationswechsler, der ersten Aufgabe, ist die Wahl der bevorzugten Steuerung etwas deutlicher für die Maus ausgefallen. Sieben Personen können hier besser mit der Maus umgehen, zwei Personen sprechen sich für die Steuerung mittels Eye-Tracking aus und eine Person findet beide Varianten gleichwertig (Abb. 6.44).

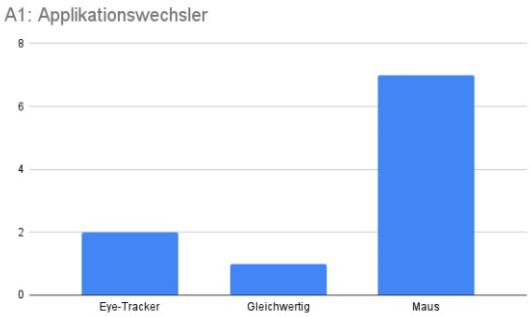


Abbildung 6.44: Maus als bevorzugte Steuerung für Aufgabe 1

Betrachtet man die Antworten für die Aufgabe zwei (Suchbild) und drei (Landkarte) so ist die Maus, wie in Abbildung 6.45 zu sehen, noch eindeutiger als Favorit erkennbar. Fast alle haben sich bei der Landkarte für die Maus als bevorzugte Steuerung entschieden. Nur eine Person empfand die Steuerung hier gleichwertig.

Bezogen auf die Auswertung des Interviews haben sich zwei Eye-Tracking Interaktionskonzepte durchgesetzt, der Splitscreen und das Verschieben eines Fensters auf einen anderen Monitor. Wie man in Abbildung 6.46 sieht, ziehen sechs Personen das Splitscreen Konzept des Eye-Trackers der Maus-/Tastatursteuerung vor, drei Personen empfinden es als gleichwertig und nur eine Person verwendet lieber die Maus.

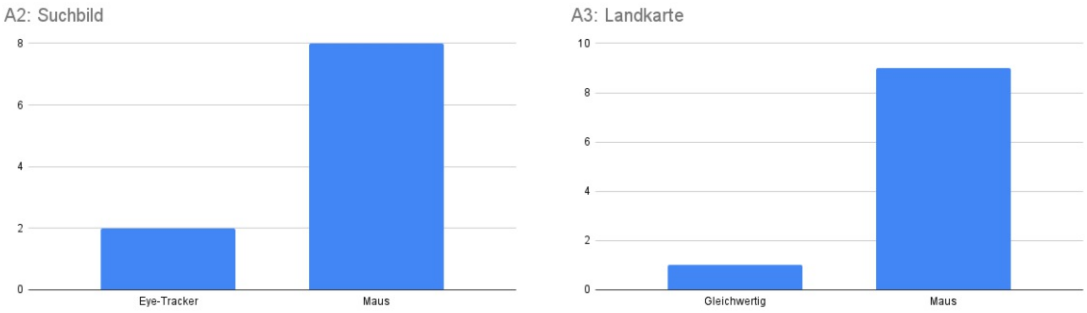


Abbildung 6.45: Interview, Maus als bevorzugte Steuerung

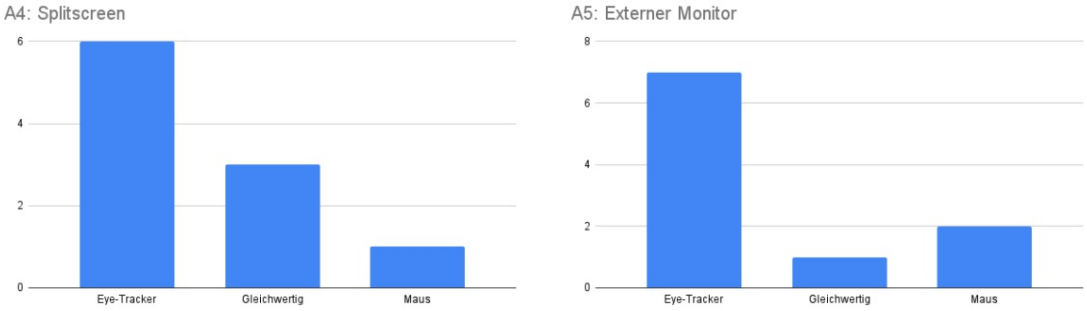


Abbildung 6.46: Interview, Eye-Tracking als bevorzugte Steuerung

6.3.3 Eindrücke bezüglich der Eye-Tracking Technologie

Die genaue Fragenstellung lautete:

Wie sind Ihre Eindrücke bezüglich der Eye-Tracking Technologie?

Bei den Antworten zu dieser Frage wurde versucht alle Antworten in bestimmte Schlagwörter beziehungsweise Gruppen einzugliedern.

Alltag

Eine Antwort welche ohne Ausnahme alle BenutzerInnen gegeben haben war, dass sie sich den Einsatz eines Eye-Trackers im Alltag durchaus vorstellen könnten und diese Technologie Zukunft hat.

Skepsis

Von einer Person fiel jedoch der Satz, dass diese Technologie noch weit entfernt sei um einen Computer nur per Eye-Tracking zu steuern. Für „Power-User“ die zum Beispiel

viel mit der Tastatur erledigen, wird Eye-Tracking nicht so häufig zum Einsatz kommen, erwähnte ein/eine BenutzerIn.

Assistiv

Einige Personen sind der Meinung, dass Eye-Tracking im Bereich der assistiven Technologien mehr Einzug finden sollte. Unter anderem für Menschen die ihre Hände aus diversen Gründen nicht benutzen können. Dazu gab es ergänzend noch die Anwort dass Eye-Tracking sehr hilfreich sein könnte, wenn gerade beide Hände benutzt werden und man ein Fenster fokussieren oder eine andere Aktion auslösen will.

Kreativ

Sehr viele BenutzerInnen merkten an, dass die Möglichkeit den Eye-Tracker schnell ein- und auszuschalten beziehungsweise ihn zu sperren damit keine Aktionen mehr ausgelöst werden, geboten werden sollte. Auch das Erarbeiten passender Konzepte wurde erwähnt und dass diesem mehr Aufmerksamkeit und Forschungszeit gewidmet werden sollte. Ein Drittel der BenutzerInnen ist der Meinung, dass Eye-Tracking im Computerspielebereich höchst interessant sein muss.

Urteile

Zwei BenutzerInnen fanden das Konzept des Fokussierens sowie das Abspielen und Stoppen eines Videos aus den Übungsaufgaben gut. Das Verschieben von Anwendungen von einem auf den anderen Bildschirm kann sich eine Person sehr gut im Alltag vorstellen. Eine weitere Aussage mehrer AnwenderInnen war, dass man die Konzepte kennen muss und es dann auch Spaß macht bestimmte Aktionen über Eye-Tracking zu erledigen.

KAPITEL 7

Diskussion

Das Thema dieser Arbeit ist die Entwicklung und Evaluierung innovativer Window Management Interaktionskonzepte zur Steuerung des Computers über einen Eye-Tracker. Diese Interaktionstechniken wurden in Form eines Prototypen entwickelt. Das Ziel war es am Ende dieser Arbeit folgende wissenschaftliche Fragestellungen beantworten zu können:

- Führen Übungsrunden zu einem Lerneffekt in Form schnellerer Task-Completion Time?
- Gibt es signifikante Unterschiede zwischen der Steuerung über Eye-Tracking und der Maus-/Tastatursteuerung?
  - Gibt es einen signifikanten Unterschied in der Fehlerrate im Vergleich zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung?
  - Gibt es einen signifikanten Unterschied in der Durchführungszeit im Vergleich zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung?
- Erhöht Augensteuerung die Usability?

Um diese Fragen beantworten zu können wurde der entwickelte Prototyp anschließend durch zehn Personen evaluiert. Jede Aufgabe wurde sowohl mit dem Eye-Tracker als auch mit der Maus-/Tastatur ausgeführt. Dabei wurde die Fehlerrate und die Task-Completion Time für jede einzelne Aufgabe erhoben und ein SUS Fragebogen für beide Eingabemethoden ausgewertet. Folgend werden die Ergebnisse der Fragebögen beziehungsweise die Resultate der evaluierten Interaktionskonzepte diskutiert.

Beim Pre-Test Fragebogen wurden demografische Daten der TeilnehmerInnen erhoben. In der Auswertung dieses Fragebogens 6.1 wurde erwähnt, dass sechs von zehn Personen eine

Brille beim Arbeiten mit dem Computer tragen. Bei der Kalibrierung des Eye-Trackers kam es nur bei einem/einer BrillenträgerInnen anfangs zu Problemen und es musste daher sechs Mal die Kalibrierung durchgeführt werden. Bei allen anderen Testpersonen wurden durchschnittlich drei Kalibrierungen ausgeführt. Obwohl die Kalibrierung bei allen BrillenträgerInnen schlussendlich erfolgreich war, mussten zwei von sechs ihre Brille (beide entspiegelt) während der Evaluierung abnehmen und ohne Brille fortfahren, da der Eye-Tracker Schwierigkeiten hatte deren Augen hinter der Brille zu erfassen. Beide BenutzerInnen hatten bezüglich ihrer Sehkraft kein Problem ohne Brille fortzufahren.

Es wurde kein Vorteil für Testpersonen aus technischen Berufen im Vergleich zu den anderen Berufsgruppen festgestellt. Es konnte jedoch während der Evaluierung beobachtet werden, dass Personen aus nicht technischen Berufen weniger voreingenommen waren und deshalb manche Konzepte schneller entdeckten als die TechnikerInnen.

Im Vergleich zu den Arbeiten von Menges u. a. [MKSS16] [MKMS17] wurden bei denen in dieser Arbeit entwickelten Interaktionskonzepten darauf geachtet, so wenig wie möglich zusätzliche grafische Elemente zur Steuerung zu verwenden. Dies hat den Vorteil, das sich die BenutzerInnen nicht mit einer neue Oberfläche auseinander setzen müssen und dass sich das gewohnte Betriebssystem in der Erscheinung nicht verändert. Eine Gemeinsamkeit zwischen der in dieser Diplomarbeit entwickelten Interaktionskonzepte und allen verwandten Arbeiten unter 3.3 ist, dass alle Eye-Tracking Konzepte ohne Zuhilfenahme der Maus/Tastatur funktionieren.

Qualitative Ergebnisse aus der Beobachtung

Anwendung wechseln
Bei dieser Interaktionstechnik geht es darum mittels Augensteuerung zwischen Anwendungen zu wechseln. Für das einfache Wechseln zwischen den Applikationen gab es nur eine Übungsaufgabe 5.2.6 für die Testpersonen. Durch einfaches fokussieren kann die Anwendung gewechselt werden. Die Beobachtungen dazu haben gezeigt, dass dieses Grundkonzept von allen BenutzerInnen sofort verstanden wurde.

Vergrößerung/Verkleinerung - Suchbild
Dieses Konzept arbeitet mit Hilfe der Distanzerkennung des Eye-Trackers. Dabei wird erkannt in welcher Entfernung sich der Kopf des/der Benutzers/Benutzerin vom Bildschirm befindet. Bei diesem Konzept erkannte man während der Evaluierung zum ersten Mal, dass eine sensible Reaktion auf Distanzänderung keinen Vorteil für den/die AnwenderIn bringt. Im Gegenteil, je sensibler ein Eye-Tracking Konzept auf Distanzänderungen reagiert desto ruhiger muss der/die BenutzerIn in einer bestimmten Entfernung verharren um keine Änderungen auszulösen.

Navigation - Digitale Landkarte
Die Ergebnisse der Evaluierung zeigen, dass es bei einer Anwendung wo es öfter notwendig ist die Distanz zum Bildschirm zu verändern, wenig Sinn macht das Interaktionskonzept des Vergrößerns und Verkleinerns einzusetzen. In dieser Arbeit ist die digitale Landkarte

eine dieser Anwendungen, bei jener man zum Beispiel Details erkennen will jedoch ohne gleich die komplette Karte Vergrößern zu müssen. Die Evaluierung dieser Aufgabe hat gezeigt, dass diese Zoomsprünge bei vielen BenutzerInnen zu einem Orientierungsverlust und Frustration führen.

Programmfenster aufteilen - Splitscreen
Beim Splitscreen teilen sich zwei Anwendungen den gesamten zur Verfügung stehenden Platz am Bildschirm. Eine Anwendung wird auf der linken Hälfte und die andere auf der rechten Hälfte des Bildschirms angezeigt. Wie im Kapitel 4.1.6 beschrieben, kann diese Aufgabe mit Hilfe der Hot-Borders gelöst werden. Sowohl bei den Beobachtungen als auch bei der anschließenden Auswertung hat sich das Splitscreen Konzept als eines der am besten funktionierenden Konzepte dieser Arbeit gezeigt.

Programmfenster verschieben - Externer Bildschirm
Obwohl es technisch nicht möglich war zwei Eye-Tracker auf einem Computer zu verwenden, wurde trotzdem ein Interaktionskonzept für einen zusätzlichen Monitor entwickelt. Dieses Konzept bietet dem/der BenutzerIn die Möglichkeit ein Programmfenster mittels Eye-Tracking Steuerung vom primären auf den externen Monitor zu verschieben. Auch dieses Konzept hat sich bezüglich der Akzeptanz durch den/die BenutzerIn und der Usability gemeinsam mit der Splitscreen Interaktionstechnik positiv hervorgehoben.

Beobachtungen während der Evaluierung dieses Konzeptes und die Antworten aus dem Interview haben gezeigt, dass Personen welche mit der Maus ungeübter sind die Eye-Tracking Steuerung als angenehmer empfanden, obwohl sie mit der Maus schneller waren.

Führen Übungsrunden zu einem Lerneffekt in Form schnellerer Task-Completion Time?

Anwendung wechseln
Die im Kapitel 6 ausgewerteten Ergebnisse zeigen für diese Anwendung sowohl zwischen dem ersten und zweiten Durchgang als auch zwischen dem ersten und dritten Durchgang einen deutlichen Lerneffekt.

Vergrößerung/Verkleinerung - Suchbild
Für das Vergrößern/Verkleinern beim Suchbild wurde zwar kein Lerneffekt festgestellt, jedoch liegen die durchschnittlichen Ausführungszeiten mittels Eye-Tracking (ET = 45,0) im Vergleich zur Maus-/Tastatursteuerung (MT = 37,9) nicht weit auseinander.

Navigation - Digitale Landkarte
Beim navigieren in der digitalen Landkarte wurde zwar auf Grund der Zeitmessungen pro Iteration kein Lerneffekt beobachtet, dennoch legte sich diese Unsicherheit ein wenig nachdem die BenutzerInnen mit dem Konzept vertrauter wurden. Sie wurden nicht schneller, aber dafür sicherer wodurch weniger Fehler gemacht wurden.

Programmfenster aufteilen - Splitscreen
Für den Splitscreen konnte kein Lerneffekt festgestellt werden.

Programmfenster verschieben - Externer Bildschirm
Aus den Ergebnissen der Evaluierung 6 geht hervor, dass dieses Interaktionskonzept zwischen allen Iterationen einen eindeutigen Lerneffekt aufweist.

Gibt es einen signifikanten Unterschied in der Fehlerrate im Vergleich zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung?

Anwendung wechseln
Fehler gab es im direkten Vergleich beider Steuerungsvarianten kaum. Das Ergebnis dieses Konzeptes ist, dass der Unterschied bezüglich der Fehlerrate nicht signifikant ist.

Vergrößerung/Verkleinerung - Suchbild
Die Sprünge beim Vergrößern und Verkleinern haben beim Suchbild Probleme bereitet und dadurch die Fehlerrate bei der Eye-Tracking Steuerung hoch getrieben (Mittelwert Eye-Tracker EERR = 1,400; Maus MERR = 0,150).

Navigation - Digitale Landkarte
Es traten Schwierigkeiten beim Navigieren auf, dabei bemerkten einige BenutzerInnen unabsichtliche Bewegungen auf der Karte, weil sie sich umsehen wollten und dabei länger in die sensitiven Bereiche geblickt haben. Das wirkte für viele der Testpersonen sehr unkontrolliert, weil Aktionen ausgelöst wurden ohne dass sie diese ausführen wollten. Durch diese vielen ungewollt ausgelösten Aktionen fehlte den Benutzern/Benutzerinnen während der Evaluierung eine Funktion mit der man Eye-Tracking beziehungsweise dessen Ereignisse deaktivieren kann. Wenn man die Fehlerrate beim Navigieren auf der Landkarte mittels Eye-Tracking (N = 40, Mittelwert EERR = 3,850) im Vergleich zur Maus-/Tastatursteuerung (N = 40, Mittelwert MERR 0,900) betrachtet, hätte eine solche Funktion diese Differenz eventuell reduzieren können.

Programmfenster aufteilen - Splitscreen
So wie auch bei anderen Interaktionskonzepten waren die BenutzerInnen in der ersten Iteration damit beschäftigt die Funktionsweise zu erforschen. Die Auswertung der Ergebnisse aus der Evaluierung zeigt, dass die Differenz zwischen den Fehlerraten der Eye-Tracking und Maus-/Tastatursteuerung nicht statistisch signifikant ist.

Programmfenster verschieben - Externer Bildschirm
Vergleicht man die Fehlerrate bei diesem Konzept zwischen Eye-Tracking und Maus, erkennt man das der Unterschied nicht signifikant ist. Beide Interaktionstechniken besitzen eine niedrige Fehlerrate.

Gibt es einen signifikanten Unterschied in der Durchführungszeit im Vergleich zwischen der Eye-Tracking Steuerung und der Maus-/Tastatursteuerung?

Anwendung wechseln
Beim Wechsel zu einer verdeckten Anwendung mit dem Eye-Tracking Konzept des Anwendungswechslers dauerte es beim Ersten Durchgang schon durchschnittlich 98,4 Sekunden (N = 10). Insgesamt waren es für Eye-Tracking und die Maus-/Tastatursteuerung jeweils drei Durchgänge. Bei der Maus-/Tastatursteuerung waren es für den ersten Durchgang durchschnittlich 8,3 Sekunden (N = 10). Der erste Durchgang beinhaltete das Konzept zu entdecken, daher wurde dieser für die statistische Auswertung gestrichen. Die BenutzerInnen waren bei der Durchführung dieser Aufgabe mit der Maus/Tastatur im Vergleich zum Eye-Tracker mehr als doppelt so schnell. Im Vergleich der Ausführungszeiten wurde ein signifikanter Unterschied festgestellt.

Vergrößerung/Verkleinerung - Suchbild
Wie man beim Prototypen des Bildbetrachtungsprogrammes gesehen hat, hat dieses Konzept auch Vorteile. Hat man die richtige Distanz des Kopfes zum Bildschirm gefunden, wird automatisch beim nächsten Bild der gewünschte Zoomfaktor übernommen. Dieser Vorteil spiegelt sich auch in der Auswertung der Task-Completion Time (Abb. 6.14) wieder, es gibt keinen signifikanten Unterschied zur Steuerung mittels Maus/Tastatur.

Navigation - Digitale Landkarte
Die Auswertung der Durchführungszeiten für die Navigation auf der digitalen Landkarte hat ergeben, dass es einen signifikanten Unterschied zwischen der Eye-Tracking Steuerung und der Steuerung mittels Maus/Tastatur gibt. Das Navigieren über den Eye-Tracker funktionierte langsamer als über die Maus/Tastatur.

Programmfenster aufteilen - Splitscreen
Wie die Fehlerrate zeigt auch der Vergleich zwischen den Durchführungszeiten keinen signifikanten Unterschied für das Splitscreen Konzept.

Programmfenster verschieben - Externer Bildschirm
Betrachtet man die Durchführungszeiten der Iterationen kann man eine statistische Signifikanz feststellen. Jedoch ist die Maussteuerung (Mittelwert MT = 9,70) bei diesem Konzept, trotz des starken Lerneffekts, eindeutig schneller als die Eye-Tracking Steuerung (Mittelwert ET = 18,10).

Erhöht Augensteuerung die Usability?

Nach der Evaluierung des Prototypen für die Eye-Tracking Steuerung und der Steurung mittels Maus, wurde von jedem/jeder BenutzerIn für jede Eingabeart der SUS Fragebogen ausgefüllt. Beim Vergleich der Prozentränge (Sauro [Sau11]) des SUS Ergebnisses der Eye-Tracking Steuerung mit 56% (SUS Score: 70,5) und der Maus-/Tastatursteuerung mit 95%

(SUS Score: 84) sieht man, dass die Maus-/Tastatursteuerung in der Gesamtbewertung vorne liegt. Dennoch bedeutet es laut Sauro [Sau11], dass eine höher wahrgenommene Usability bezüglich der entwickelten Eye-Tracking Interaktionskonzepte gegenüber 56% aller bereits getesteten Produkte besteht.

Fazit und mögliche Verbesserungen

Andwendung wechseln
Allgemein kann man sagen und das geht auch aus den Interviews hervor, dass dieses Interaktionskonzept die bestehenden Maus-/Tastaturkonzepte nicht ersetzen aber ergänzen kann. Die BenutzerInnen können sich diese Interaktionstechnik gut als Alternative Steuerung vorstellen.

Um zu einer Anwendung zu wechseln welche aktuell nicht sichtbar ist, wurde aktuell der Applikationswechsler aufgerufen. Es wäre nun eine Möglichkeit auch den oberen Rand zu nutzen um bestimmte Aktionen auszulösen, wie zum Beispiel das Minimieren eines Fensters. Des Weiteren könnte man eine zusätzliche Möglichkeit schaffen eine Applikation zu wechseln, indem man die Anwendungen auch in der Taskleiste fokussieren kann.

Vergrößerung/Verkleinerung - Suchbild & Navigation - Digitale Landkarte
Betrachtet man die Ergebnisse des Suchbildes und der Landkarte, so kann das Fazit gezogen werden, dass es einen signifikanten Unterschied zwischen der Eye-Tracking Steuerung und der Steuerung mittels Maus/Tastatur gibt. Für weitere Studien ist zu sagen, dass bei Konzepten basierend auf Distanzänderung zum Bildschirm darauf geachtet werden sollte, nicht zu sensibel auf die Änderung der Distanz zu reagieren. Für die Applikation des Suchbildes und der Landkarte würde dies bedeuten weniger Zoomstufen zu haben. Dadurch würde es möglicherweise zu weniger ungewollten Sprüngen im Zoomverhalten kommen. Auch zu bedenken ist dass grafische Hinweise, wie bei den Hot-Borders (siehe 4.1.6), stark helfen können ein Konzept schneller zu entdecken und zu verstehen.

Programmfenster aufteilen - Splitscreen
Gesamt gesehen kann man für dieses Konzept sagen, dass zum Erreichen eines Splitscreens die Steuerung mittels Eye-Tracking annähernd gleich gut funktioniert wie über die Maus-/Tastatursteuerung. Die Hälfte der Testpersonen kannte das Konzept von Microsoft's Windows 10 für den Splitscreen nicht, sie teilten die Applikationen auf indem sie die Programmfenster über den Rahmen größer zogen und dann manuell positionierten. Sechs von zehn Personen würden die Eye-Tracker Interaktionstechniken der Maus-/Tastatursteuerung vorziehen.

Aus den Ergebnissen dieses Konzeptes kann man ableiten, je performanter die Interaktion ist desto mehr Akzeptanz wird bei den BenutzerInnen erreicht.

Programmfenster verschieben - Externer Bildschirm
Obwohl es technisch nicht möglich war zwei Eye-Tracker auf einem Computer zu verwenden, wurde trotzdem ein Interaktionskonzept für einen zusätzlichen Monitor entwickelt.

Dieses Konzept bietet dem/der BenutzerIn die Möglichkeit ein Programmfenster mittels Eye-Tracking Steuerung vom primären auf den externen Monitor zu verschieben.

Eine Benutzerin hatte noch nie zuvor mit einem externen Monitor gearbeitet und machte eine interessante Aussage über die Intuitivität dieses Konzeptes. Sie empfand die Eye-Tracking Interaktion logischer als die Anwendungen mittels Maus auf den anderen Bildschirm zu ziehen. Die Person hätte es gar nicht erwartet dass man die Anwendungsfenster mit der Maus auf den externen Monitor verschieben kann.

Um dieses Konzept zu verbessern könnte man eventuell eine Möglichkeit schaffen mehrere Anwendungen zu markieren um diese dann gemeinsam auf den externen Bildschirm zu verschieben.

KAPITEL 8

Zusammenfassung und Ausblick

Diese Arbeit beschäftigt sich mit der Frage, ob durch den Einsatz eines Eye-Trackers und den dazu passenden Window Management Interaktionstechniken eine Verbesserung der Usability im Vergleich zur Maus-/Tastatursteuerung erreicht werden kann. Um geeignete Interaktionskonzepte entwickeln zu können, wurde zu Beginn dieser Arbeit eine umfassende Literaturrecherche durchgeführt. Um einen Überblick zu erhalten befasste sich diese Arbeit zu Beginn mit verwandten Arbeiten und beschäftigte sich danach mit den Grundlagen zum Thema Eye-Tracking.

Um eine mögliche Verbesserung der Usability messen und auswerten zu können, musste zuerst ein Prototyp entwickelt werden. Dazu wurden vor der Entwicklung die Komponenten des Prototypen zusammengestellt. Als Eye-Tracker wurde der Tobii Eye-Tracker 4C gewählt. Da dieser Eye-Tracker bereits ein gut entwickeltes SDK zur Verfügung stellte, fiel die Wahl auf die Programmiersprache C#.

Nachdem die Hard- und Softwarkomponenten feststanden, wurden im Zuge eines iterativen Designprozesses die Interaktionskonzepte entwickelt. Um die Ergebnisse einer Evaluierung des Prototypen zwischen der Steuerung mittels Eye-Tracker und der Maus-/Tastatursteuerung vergleichen zu können, musste bei der Entwicklung darauf geachtet werden, dass es für alle Konzepte auch eine Steuerungsalternative für die Maus/Tastatur gibt.

Bei der Entwicklung der Interaktionskonzepte wurde darauf geachtet, dass alle Eye-Tracking Interaktionen ohne Zuhilfenahme der Maus oder Tastatur ausgeführt werden können. Auch auf dauerhaft sichtbare grafische Elemente zum Auslösen einer Aktion wurde verzichtet. Für die Evaluierung des Prototypen wurden Übungen und Aufgaben definiert, alle Interaktionskonzepte finden sich darin wieder. Es wurde entschieden die Evaluierung mit zehn Testpersonen durchzuführen. Die Interaktionstechniken wurden für den täglichen Einsatz auf einem Desktopcomputer konzipiert. Aus diesem Grund gab es bei der Auswahl der Testpersonen keine bestimmten Kriterien die erfüllt werden mussten.

Zur Erhebung demografischer Daten und relevanter Hintergrundinformationen der Testpersonen wurde ein Pre-Test Fragebogen erstellt. Im ersten Teil des Fragebogens füllten die BenutzerInnen Angaben zu ihrer Person aus. Bei den personenbezogenen Daten handelt es sich um demografische Daten wie zum Beispiel das Alter, das Geschlecht, BrillenträgerIn (ja/nein) und den ausgeübten Beruf. Die Information ob eine Person eine Brille trägt oder nicht, war insofern interessant, da es eventuell zu Problemen bei der Kalibrierung des Eye-Trackers führen konnte. Der zweite Teil des Fragebogens befasste sich mit der Erfahrung im Umgang mit einem Computer. Neben der Häufigkeit der Benutzung eines Computers pro Woche, der Computererfahrung und der häufigsten Tätigkeit am Computer, wurde auch nach dem meist genutzten Betriebssystem gefragt.

Für die Evaluierung wurde das „Within-Subject Design“ [Mac] gewählt. Somit führten alle TeilnehmerInnen jede Übung und Aufgabe mit beiden Eingabemethoden (Eye-Tracker und Maus-/Tastatursteuerung) durch. Die Verteilung der Geschlechter war ausgeglichen, es nahmen fünf Frauen und fünf Männer an der Evaluierung teil. Auch die Altersgruppen waren relativ gleich verteilt, es gab eine Person unter 19, vier Personen waren zwischen 20 und 29, vier zwischen 30 und 39 und eine Person war zwischen 50 und 59 Jahre alt. Es wurden weder auf Grund des Geschlechts noch wegen des Alters Vor- oder Nachteile erkannt. Bezüglich der Kalibrierung des Eye-Trackers gab es einen Unterschied für einen/eine BrillenträgerInnen und den Personen ohne Brille. Diese eine Person hatte mit der Brille Probleme bei der Kalibrierung des Eye-Trackers. Erst nach dem sechsten Mal war der Eye-Tracker ausreichend kalibriert. Für alle anderen Personen waren es höchstens drei Kalibrierungsdurchgänge. Während der Evaluierung mussten jedoch zwei Personen ohne Brille fortsetzen, dies war aber laut eigenen Angaben der BenutzerInnen kein Problem. Alle TeilnehmerInnen gaben an, täglich einen Computer zu verwenden. Der Prototyp wurde unter Microsoft Windows 8.1 ausgeführt und auch die Oberfläche entsprach diesem. Obwohl zwei TeilnehmerInnen Apple macOS AnwenderInnen waren, konnten keine Vor- oder Nachteile bezüglich der Bedienung beobachtet werden. Vergleicht man die Ausführungszeiten der Aufgaben der zwei macOS BenutzerInnen mit allen anderen, erkennt man für keine dieser beiden Gruppen einen maßgeblichen Vorteil. Somit kann diese Eigenschaft unserer Testpersonen als relativ neutral betrachtet werden und hat deshalb keinen Einfluss auf die Durchführungszeit einer Aufgabe.

Für die Konzepte des Suchbildes und der Landkarte wurde kein Lerneffekt festgestellt, somit ist die Hypothese H1 für dieses Konzept abzulehnen. Die Eye-Tracking Steuerung weist beim Vergrößern/Verkleinern des Suchbildes mehr Fehler auf als mittels Maussteuerung, somit ist die Hypothese H2.1 wahr, denn es gibt einen signifikanten Unterschied. Für die Durchführungszeit trifft dies nicht zu, was bedeutet H2.2 wird abgelehnt, das Zoomen ist somit nicht signifikant langsamer gegenüber der Maussteuerung.

Die Ergebnisse für die Interaktionstechnik des Applikationswechslers und für das Konzept des Verschiebens eines Programmfensters auf einen externen Bildschirm sind gleich. Es weisen beide Konzepte Lerneffekte (H1 wahr) und Signifikanz bezüglich der Differenz zwischen der Task-Completion Time der beiden Eingabearten auf (H2.2 wahr). Für beide Konzepte kann die Hypothese H2.1 abgelehnt werden, da kein signifikanter Unterschied

zur Fehlerrate der Maussteuerung erkannt wurde.

Für das Interaktionskonzept des Splitscreens wurde kein Lerneffekt festgestellt (H1 abgelehnt) und es gab auch keine signifikanten Unterschiede bezüglich der Fehlerrate (H2.1 abgelehnt) und der Task-Completion Time (H2.2 abgelehnt). Das bedeutet im Umkehrschluss, dass dieses Interaktionskonzept eine hohe Usability aufweist und im Vergleich zur Maus-/Tastatursteuerung gut funktioniert. Es wurde somit ein gut funktionierendes Konzept für die Eye-Tracking Steuerung gefunden. Alle anderen Interaktionskonzepte haben dennoch Potential sich weiter zu entwickeln. So könnten zum Beispiel für das Vergrößern und Verkleinern von Bildern weniger Zoomstufen verwendet werden.

Aus Sicht des SUS Ergebnisses konnte die Usability durch die entwickelten Eye-Tracking Interaktionskonzepte gesamt gesehen nicht erhöht werden (H3 abgelehnt). Betrachtet man die Konzepte einzeln, erkennt man durch die Ergebnisse der Evaluierung 6 Erfolge anhand der Effektivität (Fehlerrate), Effizienz (Durchführungszeiten) und des Lerneffekts.

In dieser Arbeit wurden mehrere potentielle Interaktionskonzepte für die Steuerung eines Desktopcomputers mittels Eye-Tracking entwickelt und untersucht. Es wurde dadurch die Interaktionstechnik des Splitscreens als ein gut geeignetes Konzept gefunden, welches Potential besitzt irgendwann in die unterschiedlichsten Betriebssysteme integriert zu werden. Weitere Forschungsarbeiten sollen dadurch inspiriert werden ähnliche oder völlig neue Konzepte zu erforschen und die Mensch-Computer-Interaktionen aus einem anderen „Blickwinkel“ zu sehen.

Abbildungsverzeichnis

Listings

Tabellenverzeichnis

Akronyme

API Application Programming Interface. 24, 39

BCI Brain Computer Interface. 26

GUI Graphical User Interface. 12

HCI Human Computer Interaction. xi, xiii

MSIL Microsoft Intermediate Language. 21

PC Personal Computer. 15, 65

SDK Software Development Kit. 24, 97

SUS System Usability Scale. xi, xiii, 4, 5, 41, 43, 44, 60, 61, 63, 66, 81, 82, 89, 93, 94, 99, 120

UI User Interface. 1–4, 39

WMS Window Management Systeme. xi, xiii, 1, 2

WPF Windows Presentation Foundation. 5, 22, 23, 39, 64

XAML eXtensible Application Markup Language. 22, 23

XML eXtensible Markup Language. 22, 23, 39

Literaturverzeichnis

[app20a] Apple macos Exposé/Mission Control, 2020.

[app20b] Apple's Lisa computer, 2020.

[AS18] João Antunes and Pedro Santana. A study on the use of eye tracking to adapt gameplay and procedural content generation in first-person shooter games. *Multimodal Technologies and Interaction*, 2(2):23, 2018.

[Bal13] Linden Ball. *Eye-tracking and reasoning: What your eyes tell about your inferences*. 11 2013.

[Bar07] Susan B. Barnes. Alan Kay: Transforming the Computer into a Communication Medium. *IEEE Annals of the History of Computing*, 29(2):18–30, 2007.

[BBS+20] Sara Boxhoorn, Nico Bast, Hans Supèr, Leonie Polzer, Hannah Cholemkery, and Christine M Freitag. Pupil dilation during visuospatial orienting differentiates between autism spectrum disorder and attention-deficit/hyperactivity disorder. *Journal of Child Psychology and Psychiatry*, 61(5):614–624, 2020.

[BLJS+19] Alexandre Bissoli, Daniel Lavino-Junior, Mariana Sime, Lucas Encarnação, and Teodiano Bastos-Filho. A human–machine interface based on eye tracking for controlling and monitoring a smart home using the internet of things. *Sensors*, 19(4):859, 2019.

[Bro96] John Brooke. Sus: a "quick and dirty'usability. *Usability evaluation in industry*, page 189, 1996.

[chr20] Chronos Vision - Scleral Search Coils, 2020.

[CRMA19] Yann-Seing Law-Kam Cio, Maxime Raison, Cédric Leblond Ménard, and Sofiane Achiche. Proof of concept of an assistive robotic arm control using artificial stereovision and eye-tracking. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 27(12):2344–2352, 2019.

[DCK+20] Mahmoud Dahmani, Muhammad EH Chowdhury, Amith Khandakar, Tawsifur Rahman, Khaled Al-Jayyousi, Abdalla Hefny, and Serkan Kiranyaz. An intelligent and low-cost eye-tracking system for motorized wheelchair control. *Sensors*, 20(14):3936, 2020.

[Del09] Eric P Delozier. The GNU Linux desktop: an open access primer for libraries. *OCLC systems & services*, 25(1):35–42, 2009.

[Duc17] Andrew T. Duchowski. *Eye Tracking Methodology: Theory and Practice*. Springer International Publishing, 2017.

[FR66] Albert F. Fuchs and David A. Robinson. A method for measuring horizontal and vertical eye movement chronically in the monkey. *Journal of applied physiology*, 21(3):1068–1070, 1966.

[gaz20] Gp3 Eye Tracker, 2020.

[Gri19] Ian Griffiths. *Programming C# 8.0: Build Cloud, Web, and Desktop Applications*. O'Reilly Media, 2019.

[GVB17] Agostino Gibaldi, Mauricio Vanegas, and Peter J. Bex. Evaluation of the Tobii Eyex Eye tracking controller and Matlab toolkit for research. *Behavior Research Methods*, 49:923–946, 2017.

[HNA+11] Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Jarodzka Halszka, and Joost van de Weijer. *Eye Tracking: A comprehensive guide to methods and measures*. OUP Oxford, 2011.

[Ing78] Daniel H. H. Ingalls. The Smalltalk-76 programming system design and implementation. In *POPL '78*, 1978.

[JK03] Robert JK Jacob and Keith S. Karn. Commentary on Section 4. Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. *The mind's eye*, pages 573–605, 2003.

[Jul21] JuliusSweetland. Github Open Source Project OptiKey, 2021.

[KBS93] Arie E. Kaufman, Amit Bandopadhay, and Bernard D. Shaviv. An eye tracking computer user interface. In *Proceedings of 1993 IEEE Research Properties in Virtual Reality Symposium*, pages 120–121. IEEE, 1993.

[KHMS20] 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*, pages 1–12, 2020.

[Kot19] Jürgen Kotz. *Visual C# 2019 – Grundlagen, Profiwissen und Rezepte*. Carl Hanser Verlag GmbH & Company KG, 2019.

[KPW07] Manu Kumar, Andreas Paepcke, and Terry Winograd. Eyepoint: practical pointing and selection using gaze and keyboard. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 421–430, 2007.

[KSL17] Hansol Kim, Kun Ha Suh, and Eui Chul Lee. Multi-modal user interface combining eye tracking and hand gesture recognition. *Journal on Multimodal User Interfaces*, 11(3):241–250, 2017.

[Lan21] Universität Koblenz Landau. Gazetheweb: A Framework to Enable Gaze based Web Access, 2021.

[LU13] Robert Gabriel Lupu and Florina Ungureanu. A survey of eye tracking methods and applications. *Buletinul Institutului Politehnic din Iasi, Automatic Control and Computer Science Section*, 3:72–86, 2013.

[Mac] I. Scott MacKenzie. Within-subjects vs. Between-subjects Designs: Which to Use?

[met20] Metrovision - Electro oculography, 2020.

[mic20] Microsoft - Eye Control, 2020.

[mic21] Microsoft - Bing Map, 2021.

[MKMS17] Raphael Menges, Chandan Kumar, Daniel Müller, and Korok Sengupta. Gazetheweb: A gaze-controlled web browser. In *Proceedings of the 14th International Web for All Conference*, pages 1–2, 2017.

[MKSS16] 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*, pages 1–6, 2016.

[MN20] Ilia Maslov and Shahrokh Nikou. Usability and UX of Learning Management Systems: An Eye-Tracking Approach. In *2020 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 1–9. IEEE, 2020.

[ms-20] Microsoft Windows OS, 2020.

[MWO+19] Sebastian Marwecki, Andrew D Wilson, Eyal Ofek, Mar Gonzalez Franco, and Christian Holz. Mise-Unseen: Using Eye Tracking to Hide Virtual Reality Scene Changes in Plain Sight. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, pages 777–789, 2019.

[MYW+18] Xinyao Ma, Zhaolin Yao, Yijun Wang, Weihua Pei, and Hongda Chen. *Combining brain-computer interface and eye tracking for high-speed text entry in virtual reality.* 2018.

[Nie94] Jakob Nielsen. *Usability Engineering.* Interactive Technologies. Elsevier Science, 1994.

[QPG20] Alan Quimbita, Andrés Pupiales, and Graciela Guerrero. Proposal to improve the usability of social networks using eye tracking: A study to optimize internal communication in the university context. In *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE, 2020.

[Rob63] David A. Robinson. A method of measuring eye movemnent using a scieral search coil in a magnetic field. *IEEE Transactions on bio-medical electronics*, 10(4):137–145, 1963.

[RvDR+00] George Robertson, Maarten van Dantzich, Daniel Robbins, Mary Czerwinski, Ken Hinckley, Kirsten Risden, David Thiel, and Vadim Gorokhovsky. The Task Gallery: A 3d Window Manager. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, pages 494–501, New York, NY, USA, 2000. ACM.

[Sau11] Jeff Sauro. *A Practical Guide to the System Usability Scale: Background, Benchmarks & Best Practices.* Measuring Usability LLC, 2011.

[SKM15] Nikolaos Sidorakis, George A. Koulieris, and Katerina Mania. Binocular eye-tracking for the control of a 3d immersive multimedia user interface. In *2015 IEEE 1st Workshop on Everyday Virtual Reality (WEVR)*, pages 15–18, March 2015.

[sta20] Marktanteile Betriebssysteme, 2020.

[Ste11] Mark Stephens. *Sandbox*, pages 1075–1078. Springer US, Boston, MA, 2011.

[tob19] Tobii Core SDK, 2019.

[tob20] Tobii Games, 2020.

[tob21a] Tobii Eye Tracker, 2021.

[tob21b] Tobii Eye Tracker 4C, 2021.

[tob21c] Tobii Eye Tracker 5, 2021.

[tob21d] Tobii Eye Tracker Specifications, 2021.

[tob21e] Tobii Head Tracking Specifications, 2021.

[Tom03] M. Tomitsch. Trends and evolution of window interfaces. 2003.

[UEC] SAP User Experience Community. Fragebogen zur systemgebrauchstauglichkeit.

[WK00] Klaus-Heiko Wassill and Herbert Kaufmann. Binokulare dreidimensionale Videookulographie. *Der Ophthalmologe*, 97(9):629–632, 2000.

[WP17] Michel Wedel and Rik Pieters. A review of eye-tracking research in marketing. *Review of marketing research*, pages 123–147, 2017.

[YS75] Laurence R. Young and David Sheena. Eye-movement measurement techniques. *American Psychologist*, 30(3):315, 1975.

Anhang

Pre-Test Fragebogen

1. **Demografische Daten**

   - In welcher Altersgruppe befinden Sie sich?
     - ☐ ≤ 19
     - ☐ 20 - 29
     - ☐ 30 - 39
     - ☐ 40 - 49
     - ☐ 50 - 59
     - ☐ > 60

   - Ihr Geschlecht?
     - ☐ weiblich
     - ☐ männlich
     - ☐ Anderes:

   - Tragen Sie beim Arbeiten mit dem Computer eine Brille?
     - ☐ ja
     - ☐ nein

   - Welchen Beruf üben Sie aus?

2. **Erfahrung im Umgang mit einem Computer**

- Wie oft pro Woche benutzen Sie Ihren Computer?
  - ☐ 2 mal
  - ☐ 3 mal
  - ☐ 4 mal
  - ☐ 5 mal und öfter

- Wie bewerten Sie Ihre Computererfahrung?
  - ☐ sehr gering
  - ☐ gering
  - ☐ mittel
  - ☐ hoch
  - ☐ sehr hoch

- Was ist Ihre häufigste Tätigkeit wenn Sie Ihren Computer benutzen?

- Mit welchem Betriebssystem hatten Sie bisher am meisten zu tun?
  - ☐ Microsoft Windows
  - ☐ Linux Distribution
  - ☐ Apple macOS

System Usability Scale (SUS) - Post-Test Fragebogen

	Strongly disagree				**Strongly agree**
1. I think that I would like to use this system frequently					
	1	2	3	4	5
2. I found the system unnecessarily complex					
	1	2	3	4	5
3. I thought the system was easy to use					
	1	2	3	4	5
4. I think that I would need the support of a technical person to be able to use this system					
	1	2	3	4	5
5. I found the various functions in this system were well integrated					
	1	2	3	4	5
6. I thought there was too much inconsistency in this system					
	1	2	3	4	5
7. I would imagine that most people would learn to use this system very quickly					
	1	2	3	4	5
8. I found the system very cumbersome to use					
	1	2	3	4	5
9. I felt very confident using the system					
	1	2	3	4	5
10. I needed to learn a lot of things before I could get going with this system					
	1	2	3	4	5

Quelle: SUS-Fragebogen nach Brooke [Bro96]

Deutsche Übersetzung des SUS:

Fragebogen zur System-Gebrauchstauglichkeit

1. Ich denke, dass ich das System gerne häufig benutzen würde.
2. Ich fand das System unnötig komplex.
3. Ich fand das System einfach zu benutzen.
4. Ich glaube, ich würde die Hilfe einer technisch versierten Person benötigen, um das System benutzen zu können.
5. Ich fand, die verschiedenen Funktionen in diesem System waren gut integriert.
6. Ich denke, das System enthielt zu viele Inkonsistenzen.
7. Ich kann mir vorstellen, dass die meisten Menschen den Umgang mit diesem System sehr schnell lernen.
8. Ich fand das System sehr umständlich zu nutzen.
9. Ich fühlte mich bei der Benutzung des Systems sehr sicher.
10. Ich musste eine Menge lernen, bevor ich anfangen konnte das System zu verwenden.

Quelle: SUS-Fragebogen nach SAP User Experience Community [UEC]