Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Ralf Wengerodt

z-Tree-Integration in Eye-Tracking im Wirtschafts- und Sozialwissenschaften-Forschungslabor der Universität Hamburg

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Ralf Wengerodt

z-Tree-Integration in Eye-Tracking im Wirtschafts- und Sozialwissenschaften-Forschungslabor der Universität Hamburg

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Stefan Sarstedt
Zweitgutachter: Prof. Dr. Olaf Zukunft

Abgegeben am 23.09.2019

Ralf Wengerodt

Thema der Arbeit

z-Tree-Integration in Eye-Tracking im Wirtschafts- und Sozialwissenschaften-Forschungslabor der Universität Hamburg

Stichworte

Eye-Tracking, z-Tree, ökonomische Experimente, Fernbedienung

Kurzzusammenfassung

Ziel dieser Arbeit ist es, ein Eye-Tracking-Verfahren für das Wirtschafts- und Sozialwissenschaften-Forschungslabor der Universität Hamburg zu etablieren und die Steuerung über eine Schnittstelle der vorhandenen z-Tree Software zu realisieren.

Ralf Wengerodt

Title of the paper

z-Tree-Integration in Eye-Tracking on the Economic and Social Sciences Research Laboratory of the University of Hamburg

Keywords

Eye tracking, z-Tree, economic experiments, remote control

Abstract

The target of this work is to establish an eye-tracking procedure for the Economics and Social Sciences Research Laboratory of the University of Hamburg and to integrate control via an interface of the existing z-Tree software.

Inhaltsverzeichnis

1 Einleitung

Hanno Beck stellte fest, dass mit Hilfe von traditionellen ökonomischen Experimenten Daten erzeugt werden können, die über die Verhaltensweisen von Probanden Rückschlüsse zulassen (Beck 2014). Unter Ausschluss von Einflüssen werden menschliche Entscheidungen so miteinander vergleichbar. Solche Experimente werden meist in einem Labor in einer künstlichen Umgebung durchgeführt, um vergleichbare Bedingungen zu schaffen. So können vorher aufgestellte Hypothesen möglichst objektiv überprüft werden. In einfachen Laborexperimenten liegen am Ende des Experiments einzig die anonymen Antworten der Probanden vor. Diese unterscheiden sich im Idealfall so zwischen den Treatments (Variation der Versuchsbedingungen), dass die zuvor aufgestellte Null-Hypothese falsifiziert werden kann. Es kann sinnvoll sein, sich nicht nur auf Antwortdaten (Antworten der Probanden) zu stützen, sondern zusätzliche Daten zu erheben. Dies kann über ein zusätzliches Messverfahren ermöglicht werden. Besonders eignet sich hierfür Eye-Tracking, denn die dadurch erhobenen Daten können Aufschluss über die Aufmerksamkeit eines Probanden liefern und darüber, ob zwischen den Treatments eingeführte Änderungen Einfluss auf das Verhalten der Probanden haben.

1.1 Zielsetzung

Ziel dieser Bachelorarbeit ist es, ein Eye-Tracking-Verfahren für das Wirtschafts- und Sozialwissenschaften-Forschungslabor der Universität Hamburg zu etablieren und die Steuerung über eine Schnittstelle der vorhandenen z-Tree Software (Zurich Toolbox for Readymade Economic Experiments) zu realisieren. Dabei wurden sowohl Konzepte zur Integration des Eye-Trackings in diese Software erstellt, bewertet und ausgewählt, als auch eine Fernbedienung zur Ausrichtung und Kalibrierung der Eye-Tracker programmiert.

1.2 Gliederung der Arbeit

Die Arbeit gliedert sich in zwei Teilbereiche. Der erste Teil der Arbeit beschäftigt sich mit der Integration des Eye-Tracking-Verfahrens in das WiSo-Forschungslabor (Kapitel 3 bis 6). Der zweite Teil der Arbeit befasst sich mit der Analyse, dem Entwurf und der Realisierung einer Fernbedienung für die Ausrichtung und Kalibrierung der Eye-Tracker (Kapitel 7 bis 10).
Insgesamt ist die Arbeit in elf Kapitel unterteilt.
Im ersten Kapitel wird ein kurzer Überblick über das Themengebiet gegeben und die Zielsetzung der Arbeit formuliert.
Kapitel 2 befasst sich mit den Grundlagen des Eye-Tracking und der verwendeten Software. Hier werden Begriffe und Werkzeuge erläutert, die für das weitere Verständnis der Arbeit von Bedeutung sind.
In Kapitel 3 wird analysiert, welche Anforderungen an das zu erstellende System gestellt werden. Es werden auch die Startbedingungen im Forschungslabor beschrieben und der gewünschte Endzustand definiert. Dabei werden auch die Vorgaben und Richtlinien für das Forschungslabor erläutert, die die Universität Hamburg aufstellt.
Kapitel 4 umfasst den Entwurf, in dem verschiedene Eye-Tracking-Konzepte vorgestellt und hinsichtlich Realisierbarkeit und Nutzen bewertet werden.
In Kapitel 5 wird die Umsetzung des ausgewählten Konzepts behandelt.
Das letzte Kapitel des ersten Teilbereichs behandelt die Validierung des realisierten Konzepts im Hinblick auf die zu Beginn der Arbeit definierte Zielsetzung.
Kapitel 7 befasst sich anschließend mit der Analyse der Fernbedienung und ist der Beginn des zweiten Teilbereichs der Arbeit. Es behandelt einen neu definierten Ist-Zustand und die sich aus dem vorangegangenen Teilbereich neu identifizierten Anforderungen.
Kapitel 8 beschreibt den Entwurf der Fernbedienung. Die Realisierung dieser Fernbedienung wird in
Kapitel 9 behandelt. Des Weiteren werden Besonderheiten berücksichtigt, welche bei der Erstellung der Fernbedienung berücksichtigt werden mussten.
In Kapitel 10 wird die erstellte Fernbedienung validiert – besonders in Bezug auf die in Kapitel 7 beschriebenen Anforderungen.
Im letzten Kapitel der Arbeit (Kapitel 11) wird ein Fazit gezogen und der Ausblick auf mögliche Erweiterungen und Verbesserungen gegeben.

2 Grundlagen

2.1 Eye-Tracking

2.1.1 Definition Eye-Tracking

Die Informationsverarbeitung eines Menschen ist noch nicht hinreichend erforscht und ähnelt so einer „Black box“, da man sie nicht direkt beobachten und messen kann.

Einen möglichen Zugang zu diesen kognitiven Prozessen bietet das Eye-Tracking-Verfahren. Durch die stetige Weiterentwicklung des Eye-Trackings ist es möglich geworden, verlässliche Forschungsdaten zu gewinnen. Durch diesen Fortschritt liegt nun eine mögliche Anwendung von Eye-Tracking im Bereich ökonomischer Experimente.

Beim Eye-Tracking handelt es sich um ein computergestütztes Verfahren, mit dem mittels verschiedener Sensorsysteme die Augenbewegungen eines Probanden erfasst werden. Die dabei entstandenen Daten – zum Beispiel über den Verlauf der Augenbewegung oder der Fixation – ermöglichen Rückschlüsse über die kognitiven Prozesse des Probanden und können mittels Software aufbereitet und ausgewertet werden.

2.1.2 Verschiedene Eye-Tracking-Verfahren

Nach Andrew Duchowski werden verschiedene Aufzeichnungsverfahren unterschieden, die Eye-Tracking-Daten erzeugen (Duchowski 2017): (1) die Elektrookulografie (EOG), (2) Kontaktlinsen mit magnetisch induzierter Spule, (3) Fotookulografie (Photo-OculoGraphy, POG) oder Videookulografie (Video-OculoGraphy, VOG) und (4) video-basierte Messungen von Pupillen- und Hornhautreflektionen.

(1) Mit Hilfe von Elektroden (Abbildung 1.1), die um das Auge herum angebracht werden, wird der Spannungsunterschied zwischen Netzhaut und Hornhaut abgeleitet. Durch Augenbewegungen ändern sich die Spannungen an den Elektroden. Diese Spannungsunterschiede geben Rückschlüsse auf die Augenbewegungen.

(2) Es wird eine spezielle Kontaktlinse (Abbildung 2.2) verwendet, die mechanisch mit der Hornhaut verbunden ist. Durch Bewegungen des Auges bewegt sich die Kontaktlinse mit. Die Bewegung der Kontaktlinse kann über verschiedene Methoden erfasst werden und somit Rückschlüsse auf die Augenbewegungen geben. Zur Erfassung können zum Beispiel Spulen oder reflektierende Markierungen in den Kontaktlinsen genutzt werden.
(3) Das Auge wird mit einer Kamera (Abbildung 2.3) aufgenommen. Diese Aufnahmen werden anschließend Frame für Frame ausgewertet. Bei der Auswertung werden verschiedene Methoden verwendet, um die Augenbewegungen zu bestimmen.
(4) Das Auge wird aus verschiedenen Positionen gefilmt. Hierfür wird das Auge meist mit Infrarotlicht (Abbildung 2.4) belichtet, um bessere Kontraste und eine Reflexion auf der Hornhaut zu erzeugen. Diese Aufnahmen werden sofort durch einen Computer ausgewertet und die Augenbewegungen gespeichert. Hierfür werden zwei oder mehrere Bezugspunkte gemessen. Meist werden hierfür das Zentrum der Pupille und die Reflexion, die durch das Infrarotlicht auf der Hornhaut entsteht, verwendet. Durch Augenbewegungen ändert sich die Relation zwischen diesen beiden gemessenen Punkten. Diese werden über die Kamera gemessen und die Augenbewegungen digital erfasst und berechnet.

Viele Eye-Tracking-Methoden können sowohl mobil (head-mounted Eye-Tracker) als auch stationär eingesetzt werden. Dieser Unterschied wirkt sich auch auf die erhobenen Daten aus. Die mobilen Geräte erfassen nicht nur die Augenbewegungen, sondern auch was der Proband sieht. Dies wird mittels einer Kamera möglich. Die stationären Geräte werden meist in Verbindung mit Monitoren verwendet und messen meist nur die Augenbewegungen und den Monitorinhalt. Somit messen mobile Geräte alles, was der Proband wahrnimmt und die stationären Geräte den Monitor-Output.

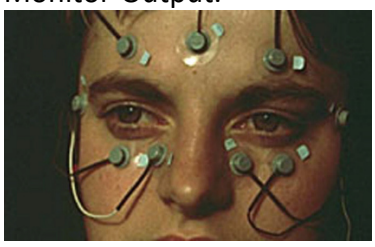

Abbildung 2.1: EOG-Proband mit angebrachten Elektroden. Foto: Duchowski (2017)

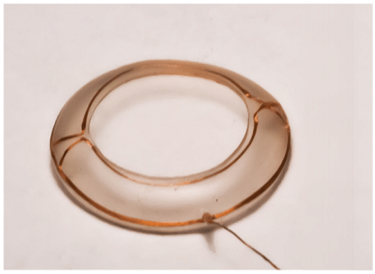

Abbildung 2.2: Kontaktlinsen mit magnetisch induzierter Spule. Foto: verändert nach Whitmire et al. (2016)

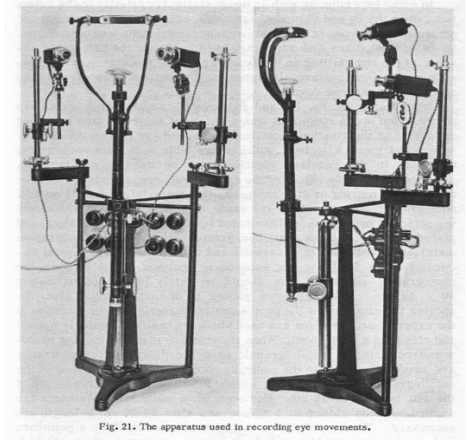

Abbildung 2.3: POG-/ VOG-Aufnahmegerät. Foto: Yarbus (1967), CC BY-SA 3.0

Abbildung 2.4: Video-basierte Messungen von Pupillen- und Hornhautreflektionen-Monitor mit Sensoreinheit am unteren Monitorrahmen

2.1.3 Anforderungen im WiSo-Forschungslabor und Auswahl des geeigneten Verfahrens

Die Stimuli werden im Wirtschafts- und Sozialwissenschaften Forschungslabor (WiSo-FL) (Abbildung 2.5) auf den Monitoren der Befragungsplätze dargeboten. Für die Messung und Datenerhebung ist daher nur die Anzeigefläche des Monitors als Aufzeichnungsbereich relevant. Somit wird kein mobiles System (head-mounted Eye

Tracker) benötigt und es kann auf ein stationäres System zurückgegriffen werden, welches die Auswertung erleichtert. Die wichtigste Anforderung im Labor ist auch die schwierigste: Die Probanden sollen so wenig wie möglich durch das Eye-Tracking abgelenkt werden. Darüber hinaus soll das Eye-Tracking simultan an 30 Befragungsplätzen durchgeführt werden. Somit werden solche Systeme ausgeschlossen, die eine lange Vorbereitungszeit oder einen hohen Betreuungsaufwand der einzelnen Probanden erfordern. Folgende Systeme können daher ausgeschlossen werden: die Elektrookulografie (EOG) und Kontaklinsen mit magnetisch induzierter Spule. Die Anforderungen werden am besten durch das System erfüllt, welches video-basierte Messungen von Pupillen- und Hornhautreflektionen vornimmt, weshalb dieses Verfahren für das WiSo-FL ausgewählt wurde. Durch die damit verbundene geringere Messgenauigkeit und -geschwindigkeit sind keine Lesestudien möglich. Bei den meisten Experimenten des WiSo-FL steht jedoch die Bestimmung von sogenannten „Areas of Interest“ (AoI) im Vordergrund, welche bei einer gewissen Größe mit diesem Verfahren zuverlässig gemessen werden können. Um eine Vergleichbarkeit der Messergebnisse zu gewährleisten, muss die Abtastrate des Eye-Trackers konstant sein. Daher wurde auf ein System mit statischer Frequenz zurückgegriffen. Eine dynamische Abtastrate ist für die Messung ungeeignet, da die damit erhobenen Daten nur schwer vergleichbar sind.

Abbildung 2.5: Wirtschafts- und Sozialwissenschaften-Forschungslabor der Universität Hamburg

2.2 z-Tree (Zurich Toolbox for Readymade Economic Experiments)

2.2.1 Aufbau der Software

z-Tree wurde von Urs Fischbacher an der Eidgenössischen Technischen Hochschule Zürich entwickelt (Fischbacher 1999). Durch die stetige Verbesserung und Erweiterung des Programms (Fischbacher 2007) hat sich z-Tree als Standardsoftware für sozial-ökonomische Experimente etabliert. Die Besonderheit dieses Programms gegenüber üblichen Umfrage-Programmen ist, dass die Probanden auf einen gemeinsamen Variablenpool zugreifen. Dadurch werden Interaktionen zwischen den Probanden ermöglicht, die für die meisten sozial-ökonomischen Experimente essentiell sind (Fischbacher et al. 2015). z-Tree besteht aus zwei Teilen: z-Tree und z-Leaf. z-Leaf ist der Client-Teil und läuft auf den PCs der Befragungsplätze. z-Leaf stellt somit die Schnittstelle zum Probanden dar und wird zur Darstellung der Stimuli und für die Eingabe der Reaktionen der Probanden verwendet. z-Leaf verbindet sich über die IP-Adresse oder den Computernamen mit z-Tree. Die Verbindung erfolgt über einen Parameter in z-Leaf oder einen Netzwerkordner, der von z-Leaf und z-Tree geteilt wird. z-Tree ist der Server-Teil und wird auf dem Experimentator-PC ausgeführt. z-Tree als Server ermöglicht die Steuerung und Überwachung von z-Leaf als Client. Darüber hinaus können mit z-Tree „Treatments“ (Experimente) und „Questionnaires“ (z.B. soziodemographische Fragebögen) erstellt werden. Die Erstellung dieser Programme wird durch einen GUI-Builder im Server-Teil unterstützt. z-Tree-Programme werden in einer Domain Specific Language erstellt, welche die Logik des Experiments beinhaltet. Die grafische Ausgabe wird über den GUI-Builder erstellt. Ein z-Tree-Programm kann in einzelne „Stages“ unterteilt werden. In diesen Stages können Code oder Grafikelemente erstellt werden.

2.2.2 Besonderheit von z-Tree: geteilter Variablenpool

Die Besonderheit der Software z-Tree liegt in der Verwendung eines gemeinsamen Variablenpools für die Probanden. Dieser ermöglicht erst die Erstellung von sozial-ökonomischen Experimenten mit Interaktion zwischen den Probanden. Ein Beispiel hierfür ist das Ultimatumspiel (Gürth 1982) mit dem die Besonderheit von z-Tree näher erläutert werden kann. In einem Ultimatumspiel gibt es zwei Spieler (A und B). Spieler A verfügt zu Beginn des Experiments über eine gewisse Ausstattung von X in ECU (experimental currency unit). Spieler A hat die Aufgabe Spieler B einen Anteil Y in ECU von seiner Ausstattung X in ECU anzubieten. Wenn Spieler B das Angebot von Spieler A annimmt, erhält Spieler A die Auszahlung von X minus Y in ECU und Spieler

B die Auszahlung von Y in ECU. Wenn Spieler B das Angebot ablehnt, erhalten beide Spieler eine Auszahlung von 0 in ECU. In diesem Experiment sind zwei Interaktionen zwischen den Spielern A und B vorgesehen. Der geteilte Variablenpool von z-Tree ermöglicht es dem Spieler B, das Angebot von Spieler A angezeigt zu bekommen und dem Spieler A, die Antwort von Spieler B auf sein Angebot zu erfahren. Dieses einfache Beispiel illustriert den großen Unterschied zwischen z-Tree und herkömmlicher Befragungssoftware, die keine Interaktion zwischen den Teilnehmern zulässt.

2.2.3 Stages

Ein z-Tree-Programm besteht meist aus mehreren Stages. Diese Stages können so definiert werden, dass sie verschiedene Funktionen erfüllen. Hierbei sind die häufigsten Aufgaben der Stage das Ausführen von Programmcode oder die Definition der Darbietung der Stimuli (Output) und der Aufnahme der Reaktionen der Probanden (Input). In z-Tree ist es möglich, ein Experiment über mehrere Runden durchzuführen, sodass auch eine Stage mehrfach durchlaufen werden kann. Vor Beginn der Anzeige („Screen“) einer Stage werden die zugehörigen Anzeigebedingungen („Condition“) geprüft. Sind diese erfüllt, wird die Stage ausgeführt. Es ist möglich, eine Stage auszuführen, diese dem Probanden aber nicht anzuzeigen. Wenn die Stage angezeigt wird, ist es möglich, einen Output für den Probanden zu generieren. Dies können Texte, Tabellen, Bilder, Sound- oder auch Video-Files sein. Durch die Verwendung von Variablen kann der Output auch dynamisch erzeugt werden. Als Eingabeelemente können auf dem Screen verschiedene Boxen erzeugt werden, wie z.B. Texteingabefelder, Eingabefelder für bedingten (min/max) Ganzzahlen oder reelle Zahlen, Schieberegler, Radioboxen und einfache Checkboxfelder. Der Input der Probanden wird in Variablen gespeichert und kann auch dynamisch verarbeitet werden. Die Besonderheit von z-Tree ist die Speicherung des Inputs aller Probanden in dem bereits erwähnten geteilten Variablenpool, mit dem dieser Input auch als Output für andere Probanden verwendet werden kann.

2.2.4 Synchroner und asynchroner Programmablauf in z-Tree

Ein z-Tree-Programm besteht aus mehreren selbst erstellten Stages, die dem Probanden angezeigt werden können. Der Programm- bzw. Experimentverlauf ist ausschlaggebend für das Eye-Tracking. Ein synchroner Programmablauf wartet beim Start einer Stage bis alle Probanden die vorherige Stage beendet haben. So starten alle Probanden die neue Stage zur selben Zeit und befinden sich immer in derselben Stage. Bei einem asynchronen Programmablauf bekommt jeder Proband sofort die

nächste Stage angezeigt, sobald dieser die aktuelle beendet hat. So ist es möglich, dass sich Probanden in unterschiedlichen Stages befinden. Im Extremfall ist in einem asynchronen Programm ein Proband noch am Anfang des Experiments, während ein anderer dieses bereits beendet hat. Ein asynchroner Programmablauf erschwert die Steuerung und Auswertung des Eye-Trackings, da es keine gemeinsamen Start- und Endzeiten der Stages gibt, sondern für jeden Probanden individuelle Zeitpunkte vorliegen. Eine Mischform aus asynchronem und synchronem Ablauf ist in z-Tree möglich, da dies für jede Stage separat definiert werden kann. Sollte ein Programm nur synchron ablaufen, existieren jedoch noch immer unterschiedliche Zeitpunkte beim Beenden einer Stage. Wenn ein Proband seine Stage beendet, wird ein Wartescreen angezeigt, bis alle anderen Probanden die jeweils aktuelle Stage abgeschlossen haben.

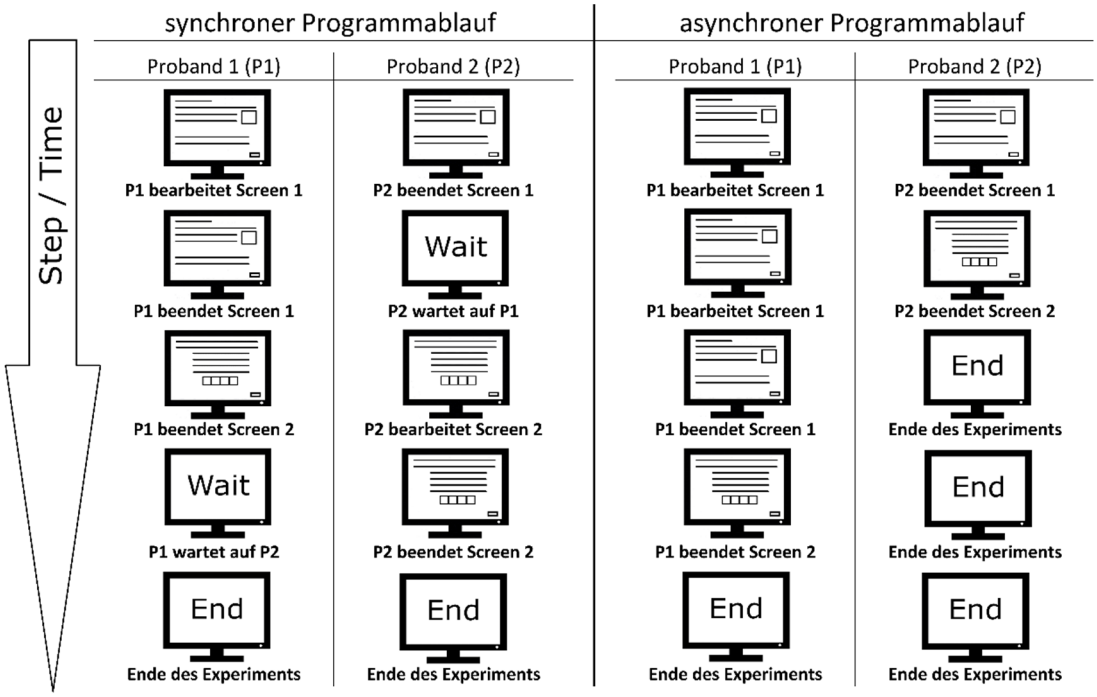

Abbildung 2.6: Synchroner / asynchroner Programmablauf

3 Analyse

3.1 Ist- und Soll-Zustand

3.1.1 Ist-Zustand

Im Wirtschafts- und Sozialwissenschaften Forschungslabor der Universität Hamburg werden an 30 computergestützten Befragungsplätzen sozial-ökonomische Experimente durchgeführt. Die technische Besonderheit dieser Experimente ist, dass der Proband durch technische Abläufe nicht beeinflusst werden darf. Die Abläufe müssen in allen Experimenten identisch sein, sodass die Ergebnisse vergleichbar bleiben. Die Software z-Tree hat sich seit Jahren bei ökonomischen Experimenten durchgesetzt und wird aus diesem Grund im WiSo-FL genutzt. Die Software z-Tree steuert über einen Experimentator-PC alle 30 Befragungsplätze des WiSo-FL, sobald diese über die Software z-Leaf mit dem Experimentator-PC verbunden sind. Somit wird das Labor zentral von einem Master-Arbeitsplatz gesteuert. Die geplanten Befragungen werden direkt an diesem in z-Tree erstellt und ausgeführt. Während des Experimentes ist es möglich, die Fortschritte der einzelnen Probanden in z-Tree zu überwachen. Die gesammelten Daten werden durch z-Tree in einer Excel-Datei am Experimentator-PC zusammengefasst.

3.1.2 Soll-Zustand

Ziel dieser Bachelorarbeit ist es, die sozial-ökonomischen Experimente mit einem zusätzlichen Messverfahren – nämlich dem Eye-Tracking – zu erweitern. Durch dieses soll es möglich werden, weitere Daten zu erfassen und valide Rückschlüsse aus dem Verhalten der Probanden zu ziehen. Die Einstellungen und Messungen des Eye-Trackers sollen die Probanden so wenig wie möglich stören. Eine zentrale Steuerung und Überwachung der Eye-Tracker soll wie bei z-Tree am Experimentator-PC möglich sein. Die erhobenen Daten sollen sinnvoll in die z-Tree-Daten integrierbar sein. Diese Anforderungen sollten bei der Ausschreibung der zu beschaffenden Hard- und Software berücksichtigt werden.

3.2 Vorgaben der Universität Hamburg für das WiSo-FL

Das WiSo-FL befindet sich räumlich in der Universität Hamburg und ist somit auch in das Rechennetz der Universität Hamburg eingebunden. Aus diesem Grund müssen die Computer des WiSo-FL die Richtlinien der WiSo-IT bzw. des universitätsweiten Rechenzentrums befolgen. Alle PCs befinden sich in der WiSo-Windows-Domain, die über Gruppenrichtlinien verwaltet werden. In diesem Netzwerk werden lediglich Domain-Benutzer zugelassen. Für die Durchführung der Experimente im WiSo-FL existiert ein stark eingeschränktes Benutzerprofil, welches nur Zugriffsrechte auf den Desktop besitzt. Dadurch ist es möglich, dass Personen anonym an den PCs der Befragungsplätze an den Experimenten teilnehmen können. Bei der Erweiterung des WiSo-FL um das Eye-Tracking-Verfahren müssen die oben genannten Richtlinien eingehalten werden.

3.3 Funktionale Anforderungen

Hier werden alle funktionalen Anforderungen, die an das System gestellt werden, noch einmal aufgeführt:

- Valide Eye-Tracking-Messergebnisse, die für wissenschaftliche Studien genutzt werden können
- Geringe Ablenkung der Probanden durch das Eye-Tracking, sodass ökonomische Experimente nicht verfälscht werden
- Integration des Eye-Tracking mit z-Tree
- Steuerung des Eye-Tracking-Experiments, die nach einer nur kurzen Einweisung durch den Experimentator selbst durchgeführt werden kann
- Einhaltung der technischen Anforderungen des Regionalen Rechenzentrum der Universität Hamburg
- Unterstützung bei der Vorbereitung eines Experiments
- Zentralisierte Durchführung eines Experiments über den Experimentator-PC
- Unterstützung bei der Nachbearbeitung eines Experiments
- Unterstützung bei der Auswertung der Eye-Tracking-Daten sowohl bei synchronen als auch asynchronen Programmabläufen

3.4 Nicht-funktionale Anforderungen

Zuverlässigkeit: Die Software muss stabil und fehlerfrei funktionieren. Abstürze eines einzelnen Clients dürfen nicht das Gesamtsystem beeinflussen.

Effizienz: Die Software darf sich nicht gegenseitig negativ beeinflussen. Das rechenintensive Eye-Tracking oder z-Tree dürfen nicht negativ durch fehlende

Systemressourcen beeinträchtig werden. Dies gilt für die lokalen Computer-, Netzwerk- und Server-Ressourcen.

Sicherheit: Die Software muss die Sicherheitsstandards der Universität Hamburg erfüllen.

Akzeptanz: Benutzer müssen die Software akzeptieren, die für sie entwickelt wurde. Das bedeutet, die Software muss verständlich, nützlich und kompatibel mit anderen Systemen sein, die die Benutzer verwenden (Sommerville 2012). Dies gilt für die Akzeptanz der Experimentatoren wie auch für die der Probanden.

Ethische Anforderungen und Datenschutz: Die Software, Hardware und der Ablauf des Eye-Trackings darf nicht diskriminierend auf Probanden wirken. Die körperliche Unversehrtheit der Probanden muss gewährleistet sein. Die Anonymität der Probanden muss jederzeit gewahrt bleiben.

Wartung: Software sollte so konzeptioniert werden, dass eine Weiterentwickelung möglich ist, um auf sich verändernde Kundenbedürfnisse angepasst zu werden. Dieses Merkmal ist entscheidend, denn Softwareanpassungen sind eine unvermeidliche Anforderung an eine sich verändernde Geschäftsumgebung (Sommerville 2012).

Übertragbarkeit: Das Graphical User Interface (GUI) und die Funktionen müssen mit geringem Aufwand anpassbar sein für Strukturänderungen im WiSo-FL oder den Einsatz in anderen Forschungslaboren.

3.5 Ablauf eines Eye-Tracking-Experiments

Der geplante Ablauf eines Eye-Tracking-Experiments gliedert sich in folgende Abschnitte:

- Vorbereitung der Befragungsplätze und des Experimentatorplatzes
- Anschließen der Eye-Tracker an die Befragungsplatz-PCs
- Laden des z-Tree-Codes und der Eye-Tracking-Einstellungen (z-Tree-Client) am Experimentator-PC
- Start von z-Leaf und der Eye-Tracking Software an den Befragungsplatz-PCs
- Einlass der Probanden
- Erläuterung des Experiments und des Eye-Trackings
- Kontrolle der Einverständniserklärung zum Eye-Tracking

- Ausführen des z-Tree-Codes
- Durchlauf der Ausrichtung und Kalibrierung
- Durchlauf des Experiments
- Auszahlung und Verabschiedung der Probanden
- Sicherung von z-Tree- und Eye-Tracking-Daten
- Export der Eye-Tracking-Daten

4 Entwurf

4.1 Konzepte zur Integration des Eye-Tracking-Verfahrens in z-Tree

Für die Implementierung des Eye-Tracking-Verfahrens in das WiSo-FL wurden im Rahmen dieser Bachelorarbeit verschiedene Konzepte zur Integration dieses Messverfahrens in die bereits vorhandene Datenerhebung mit z-Tree erstellt und bewertet. Die Auswahl erfolgte anhand der Realisierbarkeit, der in Kapitel 3 vorgestellten Anforderungen und eines möglichst effizienten Betriebs.

4.1.1 Eye-Tracking-Integration in z-Tree

In diesem Konzept übernimmt z-Tree die Steuerung des Eye-Trackers, womit eine enge Verknüpfung mit z-Tree möglich ist. Die durch den Eye-Tracker gesammelten Daten werden an z-Tree gesendet und dort verarbeitet. Die von z-Tree empfangenen Eye-Tracking-Daten werden als Input gespeichert und in der Ergebnisdatei von z-Tree ausgegeben. Es können direkte Befehle aus z-Tree an die Eye-Tracker gesendet und empfangen werden. Die gesendeten Befehle an den Eye-Tracker sind durch die Integration in das z-Tree-Programm mit dem Experiment synchronisiert. Somit ist ein synchroner und asynchroner z-Tree-Programmablauf möglich.
Da der Quellcode von z-Tree nicht als Open Source zur Verfügung steht, würde eine Integration die Zustimmung der z-Tree-Entwickler benötigen. Da die 30 Befragungsplätze im Forschungslabor gleichzeitig betrieben werden sollen, kann es außerdem zu Engpässen bei der Netzwerkkommunikation kommen. Da Eye-Tracking eine Echtzeit-Anwendung ist, kann dies zu Problemen oder Datenverlust im Experiment führen.

4.1.2 Paralleler Betrieb von Eye-Tracking und z-Tree

Der parallele Ansatz ist die einfachste Form der Integration beider Programme. Da keine Kommunikation zwischen Eye-Tracker und z-Tree stattfindet, muss keine Software angepasst werden. Vor dem Experiment kann die Ausrichtung und Kalibrierung des Eye-Trackers in der bereits vorhandenen Herstellersoftware erfolgen. Anschließend wird die Aufnahme gestartet und auf das z-Tree-Programm gewechselt. Da die meisten Ergebnisse von z-Tree nicht zeitgebunden, sondern den

entsprechenden Stages zugeordnet sind, müssen die z-Tree-Ergebnisse manuell mit den Daten der Eye-Tracking-Aufnahme verknüpft werden. Problematisch wirkt sich die fehlende zentrale Steuerung aus, da die Ausrichtung und Kalibrierung für jeden Probanden einzeln durchgeführt werden muss. Durch diese zeitaufwendige individuelle Betreuung der Probanden kann sich der Ablauf des Experiments stark verzögern und somit nicht immer gleich ablaufen. Dies kann sich negativ auf die Validität der Daten des Experiments auswirken. Zudem ist die Nachbearbeitung der Eye-Tracking-Daten ebenfalls zeitaufwendig, da die Wechsel der Stages in z-Tree nur visuell in den Eye-Tracking-Aufzeichnungen zur Verfügung stehen. Somit muss das Eye-Tracking-Video individuell für jeden Probanden geschnitten werden. Durch die Möglichkeiten, das z-Tree-Programm synchron oder asynchron ablaufen zu lassen und die unterschiedlichen Längen der Waiting-Screens entstehen bei jedem Probanden in jeder Stage unterschiedliche Videolängen, welche einzeln manuell für das Schneiden ermittelt werden müssen.

4.1.3 z-Tree-Integration in Eye-Tracking-Software

Die Integration von z-Tree in die Eye-Tracking-Software ist ein Ansatz, bei dem der Quellcode von z-Tree nicht geändert werden muss. Im Gegensatz zu dem Parallelbetrieb, wird durch die Integration in die Eye-Tracking-Software ein positiver Effekt auf die Auswertung erzielt. Ziel ist es, dass die in z-Tree erhobenen Daten automatisiert an die Eye-Tracking-Software übermittelt und von dieser verarbeitet werden. In z-Tree ist es möglich, während des Experiments externe Programme (über „External Program“) mittels z-Leaf an den PCs der Befragungsplätze auszuführen. Über diese Schnittstelle können Daten übermittelt und die Eye-Tracking-Software durch z-Tree gesteuert werden. Eine solche Eye-Tracking-Software muss mit verschiedenen Parametern gesteuert werden können. Darüber hinaus sollte ein Stagewechsel in z-Tree in den Eye-Tracking-Daten dokumentiert werden. Durch diese Informationen können die Eye-Tracking-Videos effizient für jeden Probanden automatisch anhand der Marker geschnitten werden.

4.1.4 Auswahl des Eye-Tracking-Konzepts

Eine Integration der Eye-Tracking-Daten in z-Tree hätte den höchsten Nutzen, da hiermit auch eine Steuerung von z-Tree mit dem Eye-Tracker möglich wäre. Da jedoch z-Tree eine Blackbox ist und die Computer-Rechenleistung und Netzwerk-Bandbreite bei 30 gleichzeitigen Aufnahmen recht hoch ist, ist dies aus technischer Sicht mit den Mitteln des WiSo-FL nur schwer realisierbar. Ein Parallelbetrieb von z-Tree und Eye-Tracking-Software wäre ohne großen Aufwand umsetzbar. Es gäbe jedoch – außer über das Eye-Tracking-Video – keine Verbindung zwischen z-Tree und dem Eye-

Tracker. Nach Berücksichtigung der genannten Vor- und Nachteile wurde das Konzept ausgewählt, bei welchem z-Tree in die Eye-Tracking-Software integriert werden soll. Der zusätzliche Nutzen gegenüber dem parallelen Betrieb wäre, dass das Eye-Tracking-Video mit Informationen aus z-Tree angereichert werden kann und eine zentrale Steuerung des Experimentes ermöglicht wird. Nach der Festlegung auf dieses Integrationskonzept konnte die entsprechende Ausschreibung für die Beschaffung der 30 Eye-Tracker erstellt werden.

5 Realisierung

5.1 Ausschreibung

Da die Beschaffung der Eye-Tracking-Hardware und Software ausgeschrieben wurde, war es möglich, die Ausschreibung um den Punkt einer parametergesteuerten Schnittstelle zu erweitern. Nach der Auswahl der Firma Tobii durch die Universität Hamburg wurde die Realisierung dieser Schnittstelle mit dem Auftragnehmer spezifiziert und diese als z-Tree-Client umgesetzt. Durch den z-Tree-Client wurde die Eye-Tracking-Software „Tobii Studio“ so erweitert, dass die beauftragten Anforderungen erfüllt waren.

5.2 Eye-Tracking im WiSo-FL

5.2.1 Hardware Tobii X60

Der Tobii X60 (Abbildung 5.1) besteht aus zwei Komponenten: der Eye-Tracking-Einheit und der externen Prozessor-Einheit. Die Eye-Tracking-Einheiten werden mit Hilfe einer magnetischen Aufnahmeschiene mittig an den unteren Monitorrahmen der Befragungsplatz-PCs befestigt. Die Eye-Tracking-Einheit verfügt über zwei Infrarot-Scheinwerfer, die in verschiedenen Modi die Augen der Probanden ausleuchten. Die zwei Kameras im Tobii X60 haben eine statische Abtastrate von 60Hz. Somit kann technisch eine gleichbleibende Aufnahmequalität garantiert werden. Die Eye-Tracking-Einheit wird per USB an die externe Prozessor-Einheit angeschlossen. Die externe Prozessor-Einheit berechnet die Bilddaten der Eye-Tracking-Einheit vor und übermittelt die so generierten Daten per LAN an den Befragungsplatz-PC. Durch diese Vorberechnung wird es ermöglicht, dass keine personenbezogenen Merkmale einzelner Probanden gespeichert werden können. Die so übermittelten Daten werden von der Software Tobii Studio verarbeitet und gespeichert.

Abbildung 5.1: Tobii X60 Eye-Tracking-Einheit (links) und externe Prozessor-Einheit (rechts)

5.2.2 Software Tobii Studio

Für die Verarbeitung und Speicherung der Daten wird die Software Tobii Studio verwendet. Diese erzeugt ein Eye-Tracking-Video vom Bildschirminhalt des Probanden und dessen Blickbewegungen. Durch eine Anpassung des Tobii Studio für das WiSo-FL ist es möglich, dieses über den z-Tree-Client zu steuern. Diese externe Steuerung ermöglicht es, dass alle 30 Probandenplätze simultan betrieben werden können. Diese Anpassung wurde in die Release Version 3.4.6 übernommen. Durch Änderungen in der Datei „Application.Settings.xml“ kann die Anpassung aktiviert werden. Nähere Angaben hierfür finden sich in der Release Note der Version 3.4.6 unter „1.1.1 Disabling the calibration“. Die Mindestanforderungen von Tobii Studio an den Befragungsplatz-PC sind wie folgt: ein Intel i7-Prozessor (Generation vier), 8 GB RAM und eine externe Grafikeinheit mit mindestens 2 GB Speicher.

5.2.3 Ausrichtung

Der Tobii X60 Eye-Tracker nutzt das Eye-Tracking-Verfahren „video-basierte Messungen von Pupillen- und Hornhautreflektionen“. Dieses Verfahren ermöglicht dem Eye-Tracker nur einen spezifischen Bereich abzudecken, in dem die Augenbewegungen eines Probanden erfolgreich aufgezeichnet werden können. Dieser Bereich ist eine Art virtuelle Box, die sich etwa in einer Entfernung von 70 cm zum Eye-Tracker befindet und etwa eine Größe von 50 cm mal 36 cm und eine Tiefe von etwa plus/minus 22.5 cm hat (Tobii Technology AB 2014). Bei der Ausrichtung des Eye-Trackers soll der Proband möglichst mittig in dieser virtuellen Box positioniert werden. Durch die mittige Ausrichtung ist die Toleranz gegenüber den Bewegungen des Probanden während des Experiments am größten. Die Ausrichtung wird durch ein visuelles Feedback am Befragungsplatz-PC erleichtert. Die optimale

Ausrichtung des Probanden kann durch Änderungen am Monitor oder der Sitzposition des Probanden erreicht werden. Dies kann der Proband selbstständig durchführen. Somit müssen nicht alle Befragungsplätze einzeln vom Experimentator ausgerichtet, sondern nur abschließend geprüft werden. Dabei ist darauf zu achten, dass eine angenehme Sitzposition und eine gute Erreichbarkeit von Computertastatur und -maus gegeben sind, da diese Position während des gesamten Experiments möglichst konstant bleiben sollte. Nach einer erfolgreichen Ausrichtung wird die Kalibrierung gestartet.

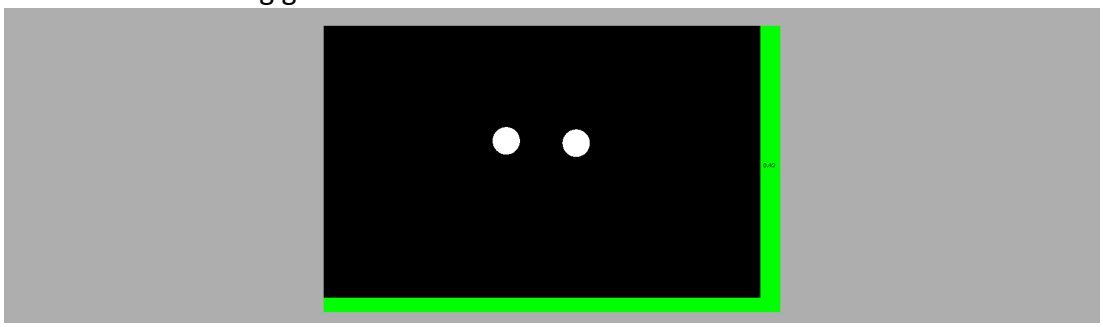

Abbildung 5.2: Erfolgreiche Ausrichtung

5.2.4 Kalibrierung

Das Tobii Studio Pro bietet eine 5-Punkt-Kalibrierung. Diese dient dazu, das ideale Auge, welches für die Berechnungen genutzt wird, an das Auge des Probanden anzupassen. Da kein Mensch ein ideales Auge besitzt, ist eine Kalibrierung zwingend erforderlich. Die 5-Punkt-Kalibrierung besteht aus fünf Schritten, bei der jeder Punkt einzeln angezeigt wird. Hierfür werden hintereinander die vier Punkte in der Nähe der Ränder des Bildschirms angezeigt und der letzte Punkt in der Mitte des Bildschirms. Bei der Messung muss der Proband möglichst genau in die Mitte des angezeigten Punktes schauen. Die so gewonnenen Ergebnisse werden mit der idealen erwarteten Position verglichen und grafisch aufgearbeitet. Als Ergebnis wird eine Grafik für jedes Auge erzeugt, welches die Abweichung vom idealen Auge an den Messpunkten anzeigt. Sind diese Abweichungen zu groß, kann eine weitere Kalibrierung versucht werden. Da die Kalibrierung nur die Abweichung angibt, ist sie kein Indikator über die Qualität der Aufnahme. Für die Messung der Qualität der Aufnahme wird ein sogenannter „Klicktest“ verwendet.

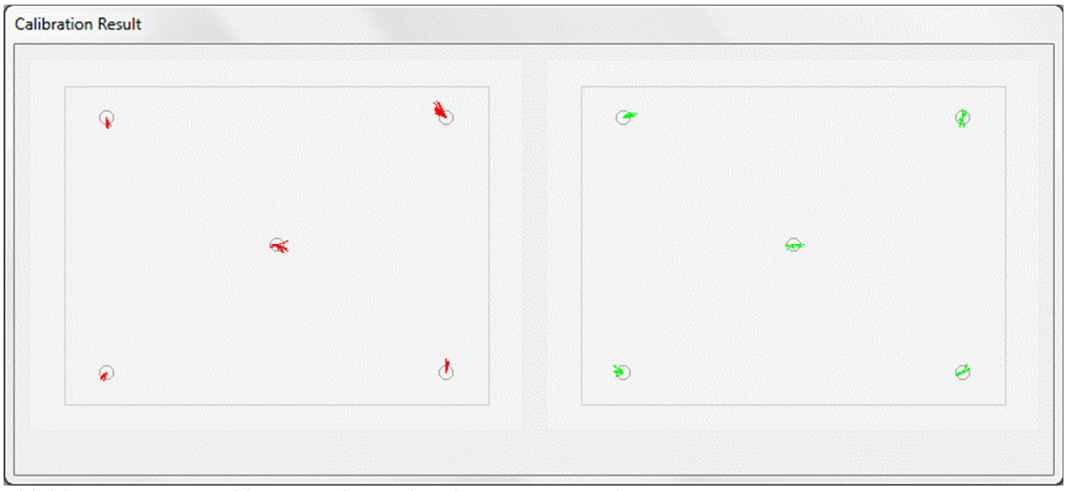

Abbildung 5.3: Gute Kalibrierung des Probanden mit Eye-Tracking

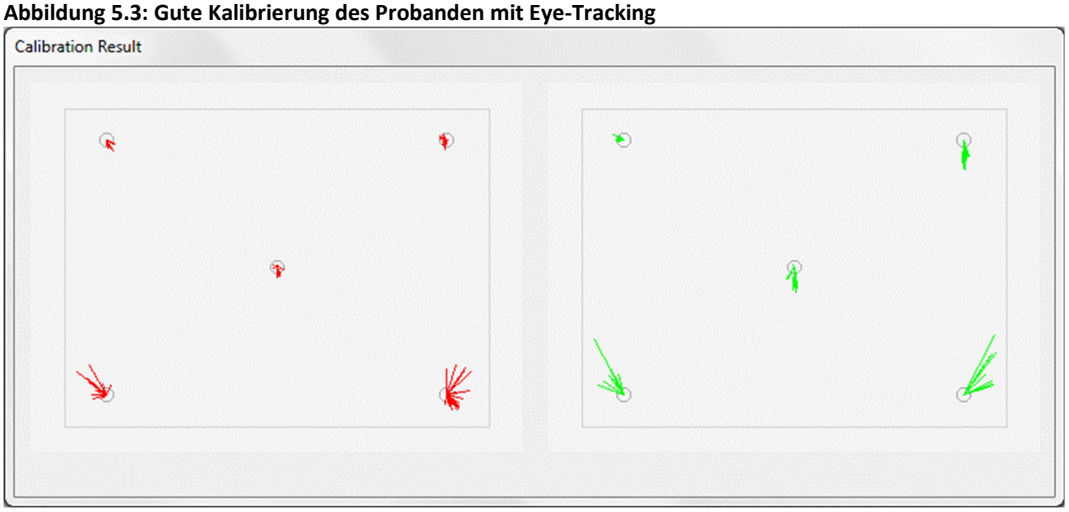

Abbildung 5.4: Mäßige Kalibrierung

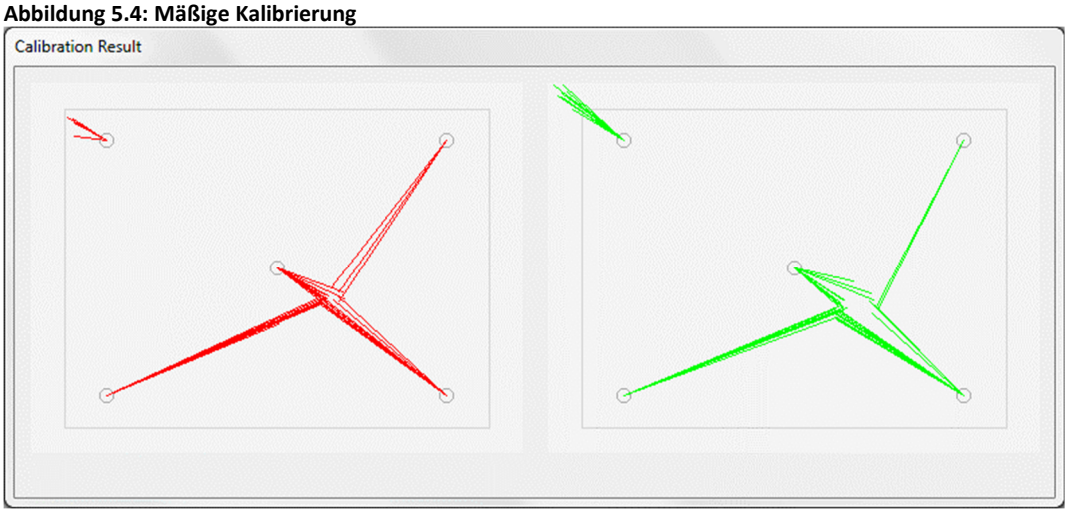

Abbildung 5.5: Schlechte Kalibrierung

5.2.5 Klicktest

Der Klicktest sieht analog wie die 5-Punkt-Kalibrierung aus. Der Proband wird aufgefordert möglichst genau mit dem Cursor in die Mitte des angezeigten Punkts zu klicken. Da im Gegensatz zur Kalibrierung nicht die Abweichung gemessen wird und die Kalibrierung schon Anwendung findet, können die so gemessenen Punkte als Referenz für die Qualität genutzt werden. Daraus folgt, dass umso genauer der angezeigte Punkt mit dem Messpunkt übereinstimmt, umso höher ist die Qualität der Aufnahme. Dies erfolgt natürlich unter der Annahme, dass der Proband auf den Punkt blickt, auf den er klickt. Bei längeren Experimenten sind mehrere Klicktests empfehlenswert, da sich die Qualität durch eine Änderung der Sitzposition des Probanden verändern kann. Da der Klicktest während des laufenden Experimentes durchgeführt wird, ist eine Umsetzung in der Experiment-Software (z-Tree) nötig.

5.2.6 z-Tree-Client

Der z-Tree-Client ist die parametergesteuerte Schnittstelle zwischen Tobii Studio und z-Tree und wurde von der Firma Tobii nach den Anforderungen des WiSo-FL zur Verfügung gestellt. Ziel ist die Steuerung der Eye-Tracking-Experimente allein mit Hilfe von z-Tree. Hierfür werden folgende Steuerungselemente benötigt: Zur Vorbereitung einer Eye-Tracking-Aufnahme muss der Proband ausgerichtet und der Eye-Tracker kalibriert werden. Für die Ausrichtung kann der z-Tree-Client das visuelle Feedbacksystem extern ein- beziehungsweise ausschalten. Zudem kann die Kalibrierung extern gestartet werden. Diese wird nach der Durchführung automatisch beendet und speichert das Ergebnis in einer Bilddatei. Diese Bilddatei wird nach den Vorgaben des WiSo-FL mit dem Namen des Befragungsplatz-PCs und dem aktuellen Zeitstempel gespeichert. Durch den Dateinamen lässt sich die Kabine identifizieren. Durch den Zeitstempel kann ermittelt werden, welche Kalibrierung im Experiment Anwendung gefunden hat. Wenn die Ausrichtung und Kalibrierung erfolgreich abgeschlossen sind, muss die Eye-Tracking-Aufnahme für das eigentliche Experiment extern gestartet und am Ende wieder gestoppt werden. Um die Integration von z-Tree-Daten in die Eye-Tracking-Software zu ermöglichen, erfolgt eine Übermittlung von Markern (wie beispielsweise bei einem Stagewechsel) in Echtzeit, die mit einem spezifischen Namen in Tobii Studio gespeichert werden. Darüber hinaus ermöglicht der z-Tree-Client eine Abfrage über den aktuellen Eye-Tracking-Status (d.h., ob die Augenbewegungen eines Probanden gerade erfolgreich aufgezeichnet werden können oder nicht). Diese Statusanzeige während eines Experiments erlaubt ein Echtzeit-Feedback für den Experimentator.

5.2.7 z-Tree-Client Test

Mit dem gelieferten z-Tree-Client (Version 161010) wurde ein vollständiger Funktionstest durchgeführt. Dieser Test wurde sowohl lokal an einem Befragungsplatz-PC, als auch ferngesteuert mit z-Tree über den Experimentator-PC durchgeführt. Bei diesem Test wurden keine Probleme festgestellt. Das detaillierte Ergebnis des Tests wurde im z-Tree-Client-Testprotokoll hinterlegt (siehe Anhang A.1 z-Tree-Client-Testprotokoll).

5.2.8 Kommunikation zwischen z-Tree und z-Tree-Client

z-Tree bietet die Möglichkeit, externe Programme auszuführen. Es ist möglich, diese Befehle an beliebiger Stelle im Code zu platzieren. Diese Programme können sowohl am Client (z-Leaf) oder auch am Server (z-Tree) ausgeführt werden. Durch diese Möglichkeit kann eine einseitige Kommunikation mit dem z-Tree-Client und damit mit Tobii Studio und dem Eye-Tracker hergestellt werden. Da mit dem z-Tree-Client allerdings keine Netzwerkkommunikation möglich ist, müssen die Befehle von z-Tree über z-Leaf an den z-Tree-Client lokal übermittelt werden. Letzterer übermittelt die Befehle weiter an Tobii Studio und den Eye-Tracker.

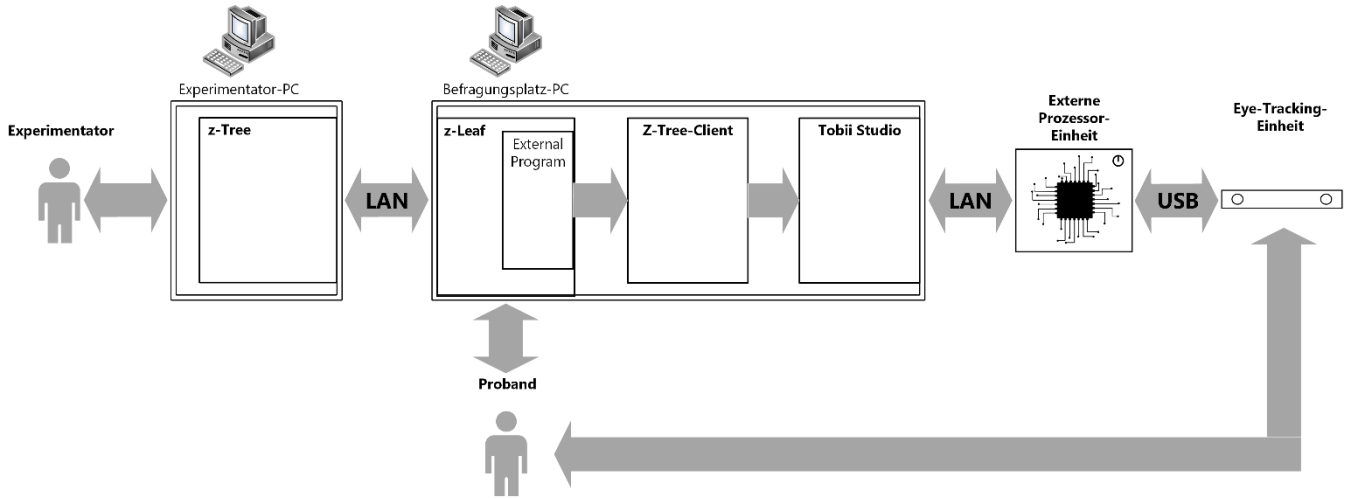

Abbildung 5.6: Kommunikation zwischen z-Tree und z-Tree-Client

5.3 Eye-Tracking-Daten

5.3.1 Videoaufnahme

Die Eye-Tracking-Daten werden in einer fortlaufenden Videoaufnahme gespeichert. Die Aufnahme beginnt mit dem Befehl „start“, der meist direkt an den Anfang des Experiments gesetzt wird. Die Videoaufnahme endet mit dem Befehl „finish“, welches nach den zu messenden Screens oder an das Ende des Experiments gesetzt wird. Durch die nötige Vorbereitung eines Eye-Tracking-Experiments, ist es nur einmal möglich, eine Aufnahme zu starten und zu stoppen. Da das „Tobii Studio start Klickskript“ (vgl. Kapitel 5.3.10) von Probanden beeinflusst werden könnte, kann dies nur zwischen zwei Experimenten erfolgen. Die Videodatei beinhaltet neben der Videoaufnahme auch alle Events wie Computermausklicks, gesetzte Marker und die Eye-Tracking-Daten.

5.3.2 Eye-Tracking Daten aufarbeiten

Durch das Starten und anschließende Stoppen eines Eye-Tracking-Experiments werden auf dem Probanden-Computer Eye-Tracking-Daten gespeichert. Diese Eye-Tracking-Daten müssen nach dem Sichern noch bearbeitet werden. Da die gespeicherten Daten offene Tobii Projekte sind, müssen diese noch exportiert werden, sodass sie miteinander verbunden (Merge in Tobii Studio) werden können. Dafür bietet das Tobii Studio keine Schnittstelle an, welche dies automatisiert ausführen kann. Somit ist nur das manuelle Einlesen der Daten, das Umbenennen der Aufnahme nach Kabinennummer und das anschließende Exportieren möglich. Da dieser Prozess sehr zeitintensiv ist, wurde im Rahmen dieser Bachelorarbeit ein Maus-Klick-Skript mit dem Programm „Sikuli“ erstellt. Sikuli ist eine Software, die im Massachusetts Institute of Technology entwickelt wurde. Sie ermöglicht es, in einer eigenen Skript-Sprache einen automatisierten Ablaufplan zu erstellen. Hierbei kann Sikuli im Gegensatz zu anderer Automatisierungs-Software auch auf den Bildschirminhalt reagieren. Sikuli übernimmt die Maussteuerung und kann anhand des Monitorbildes gesteuert werden. Durch die Programmierung eines Sikuli-Ablauf-Programms kann dieses den kompletten Export der Eye-Tracking-Daten übernehmen. Dieses Programm übernimmt somit den zeitaufwendigen Teil des Exports der Eye-Tracking-Daten.

5.3.3 Marker / Schneiden

Ist ein z-Tree-Experiment ein synchrones Experiment, sind die Startpunkte der einzelnen Screens bei jedem Probanden gleich. Die Endpunkte der einzelnen Screens sind aber unterschiedlich, da die Probanden den Screen unterschiedlich lang bearbeiten. Dies könnte zwar auch über einen Timeout der Screens vereinheitlicht werden, dies würde aber den Probanden beeinflussen und die Eye-Tracking-Ergebnisse verfälschen, da die meisten Screens ohne Zeitdruck beantwortet werden sollen. Der Timeout wäre auch nicht die Lösung für ein asynchrones Projekt, da diese auch einen variablen Startpunkt haben. Somit müssten die Start- und Endpunkte der Screens manuell in jedem Probandenvideo gesetzt werden, was sehr zeitintensiv wäre. Die Lösung für das Problem mit synchronem und asynchronem Programmablauf ist die Verwendung der Marker. Somit kann das Video automatisch nach den Screens geschnitten werden. Dies beschleunigt und vereinfacht die Auswertung nicht nur enorm, sondern erhöht auch die Genauigkeit, da das Video auf die Millisekunde genau geschnitten wird. Neben den Markern aus z-Tree werden auch alle Computermausklicks vom Probanden als Marker im Eye-Tracking-Video gespeichert.

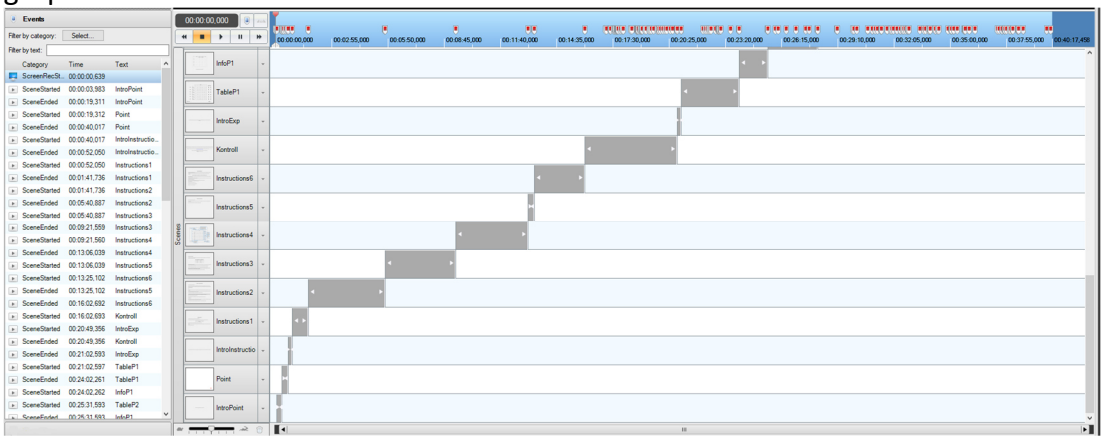

Abbildung 5.7: Gesetzte Marker durch z-Tree-Client (links) und automatisch geschnittene Screens anhand der Marker (rechts)

5.3.4 Eye-Tracking und z-Tree-Daten

Durch das automatische Schneiden und Exportieren kann schon kurze Zeit nach dem Experiment ein vollständiger Datensatz erzeugt werden. Dieser Datensatz beinhaltet die z-Tree-Daten, die in einer Excel-Tabelle gespeichert und nach Subjekt- bzw. Kabinennummer zeilenweise sortiert sind. Die Bilddateien der Kalibrierung sind nach Kabinennummer und Erstellungszeit benannt. Sie zeigen die Abweichung des Probanden vom „idealen Auge“. Die für das Eye-Tracking verwendete Kalibrierung ist die letzte, die erzeugt wurde. Alle anderen Kalibrierungen des Probanden kommen

nicht zur Anwendung. Alle Kalibrierungsergebnisse bleiben aber für die Auswertung erhalten, um festzustellen, ob es ein grundlegendes Problem bei der Kalibrierung mit dem Probanden gab. Die exportierten Eye-Tracking-Daten sind nach der entsprechenden Kabinennummer benannt und können mithilfe der Marker in einzelne Screens geschnitten werden. Diese so erzeugten Screens können direkt mit einer HeatMap, GazePlot und Cluster visuell ausgewertet werden. Die Erstellung von „Areas of Interest“ auf den einzelnen Screens ist auch möglich. So ist eine schnelle Beurteilung der einzelnen Bereiche möglich. Die Areas of Interest geben bei der Auswertung Auskunft über die Anzahl und Dauer der Augenbewegungen des Probanden in diesem Bereich. Neben der Auswertung der Eye-Tracking-Daten im Tobii Studio ist auch ein Export der Daten in eine andere Statistik-Software möglich. Folgende Formate können erstellt werden: „.tsv“ und „.xlsx“. Das Arbeiten mit Excel für die Eye-Tracking-Daten wird nicht empfohlen, da hier schnell das Verarbeitungslimit von Excel erreicht wird (Excel 2010 hat ein Limit von: 1.048.576 Zeilen mal 16.384 Spalten). Die exportierten Daten unterscheiden sich je nachdem welcher Filter in Tobii Studio verwendet wurde. Eine nachträgliche Änderung und Anpassung des Filters ist jederzeit in Tobii Studio möglich. Neben selbstdefinierten Filtern stehen zwei Hauptfilter zur Verfügung, die in den meisten Fällen verwendet werden können. Dies ist zum einen der „Tobii Fixation Filter“, der aus einer Punktwolke an Messungen eine Fixation in einem Bereich zusammenfasst und zum anderen der „RAW Data“-Filter, der jeden einzelnen Messpunkt ausgibt. Die Zuordnung der einzelnen Dateien nach Kabinennummer und Session-Zeit ermöglicht das Zusammenführen aller Daten, bei gleichzeitiger Anonymität des Probanden. Sofern auch ein Klickscreen verwendet wurde, besteht die Möglichkeit, die Qualität der Messung zu beurteilen. Dies kann über Statistik-Programme und die Verwendung des Computermausklick-Events automatisiert bestimmt werden.

Abbildung 5.8: Auswertung HeatMap

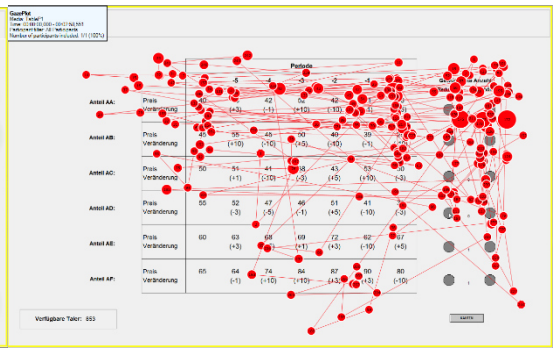

Abbildung 5.9: Auswertung GazePlot

Abbildung 5.10: Auswertung Cluster

Abbildung 5.11: Auswertung Areas of Interest

6 Validierung

6.1 Validierung des Konzepts

Das Konzept „z-Tree-Integration in Eye-Tracking-Software“ konnte erfolgreich realisiert werden. Die Kommunikation zwischen z-Tree und der Eye-Tracking-Software Tobii Studio konnte mit Hilfe des z-Tree-Clients umgesetzt werden. Somit können die Marker durch z-Tree gesetzt werden. Diese befinden sich in einem Toleranzbereich von unter drei Millisekunden, womit sie zum Schneiden des Eye-Tracking-Videos genutzt werden können. So wird es möglich, innerhalb von wenigen Minuten nach dem Experiment auswertbare Ergebnisse zu erzeugen. Dies ist besonders in Anbetracht der Menge an Daten, die durch die Verwendung von bis zu 30 Probandenplätzen pro Session entstehen, eine enorme Erleichterung für die Forschenden. Erste Gespräche mit Forschenden, die das neue System nutzen wollen, haben eine hohe Akzeptanz für das Konzept ergeben. Um bestehende und neue z-Tree-Experimente eye-tracking-fähig zu gestalten, sind nur geringe Änderungen nötig, die auch durch die Forschenden umgesetzt werden können. Der Unterschied zwischen einem ökonomischen Experiment ohne Eye-Tracking ist für den Probanden bei diesem Konzept auch nur während der Ausrichtung und Kalibrierung des Eye-Trackers zu spüren. Somit hält sich die Beeinträchtigung des Probanden durch das Eye-Tracking in angemessenen Grenzen.

6.2 Validierung der funktionalen Anforderungen

Die funktionalen Anforderungen konnten bis auf die Steuerung des Eye-Trackings erfolgreich umgesetzt werden.

6.2.1 Fehlende Flexibilität bei der Steuerung der Eye-Tracker über z-Tree

Die Steuerung der Ausrichtung der Probanden und Kalibrierung der Eye-Tracker über z-Tree konnte nicht ausreichend flexibel umgesetzt werden. Alle für das Eye-Tracking benötigten Funktionen (wie Ausrichtung, Kalibrierung und Start/Stopp der Aufnahme) können zwar in z-Tree gestartet werden, jedoch können Probleme bei der Ausrichtung oder Kalibrierung nicht kompensiert werden. Die Ausführung mehrerer Programme in z-Tree (d.h. einzelne Programme für Ausrichtung, Kalibrierung und das Experiment) bedingt eine konstante Teilnehmerzahl. Das bedeutet, dass bei Ausfall

des Eye-Trackings bei einzelnen Probanden der gesamte Durchlauf aller Probanden unterbrochen werden muss. Im Fall der Kalibrierung der Eye-Tracker muss der Experimentator entscheiden, wann eine Kalibrierung erfolgreich war. Das bedeutet, er muss Einfluss auf den Programmablauf einzelner Probanden nehmen. Dies ist in z-Tree nicht vorgesehen und eine Umsetzung über den z-Tree-Code hat sich als zu komplex (d.h. schlecht steuerbar) herausgestellt. Des Weiteren wurde die Möglichkeit geprüft, den Probanden die Entscheidung über die Güte der Kalibrierung zu überlassen. Diese wurde jedoch verworfen, da keine einheitliche Güte gewährleistet werden kann. Außerdem besteht so die Gefahr, die Resultate des eigentlichen Experiments zu verfälschen und somit die Validität der Messmethode in Frage zu stellen. Um die für die Ausrichtung und Kalibrierung der Eye-Tracker benötige Flexibilität zu erreichen, wurden diese von z-Tree getrennt und in eine Fernbedienung ausgelagert. Dadurch ist eine zentrale und simultane Steuerung durch den Experimentator möglich. Start und Stopp der Eye-Tracking-Aufnahme, sowie das Setzen der Marker bleiben weiterhin Bestandteil von z-Tree und ermöglichen so eine automatische und individuelle Steuerung dieser Bestandteile.

6.2.2 Basis-Steuerung der Eye-Tracker über z-Tree

Die fehlende Flexibilität in z-Tree für die Steuerung des Eye-Trackers ist für Basisfunktionen nicht problematisch. So kann weiterhin der Teil von z-Tree verwaltet werden, der nicht dynamisch ist. Dies beinhaltet das Starten und Stoppen der Eye-Tracking-Aufnahme, welche meist mit Beginn und Ende des ökonomischen Experiments identisch ist. Auch kann das Setzen der Marker als „screen Start“ und „screen Ende“ bei z-Tree verbleiben, da z-Tree auch den ScreenWechsel verwaltet. Der dynamische Teil des Eye-Trackings, der unter anderem das Ausrichten und Kalibrieren des Eye-Trackers umfasst, benötigt eine Lösung abseits von z-Tree.

7 Analyse der Fernbedienung

7.1 Veränderter Ist- und Soll-Zustand

7.1.1 Ist-Zustand

Der Ist-Zustand hat sich um die bisherigen Arbeiten, die in den vorherigen Kapiteln erläutert wurden, erweitert. Die Probandenplätze sind mit Tobii X60 Trackern ausgestattet worden. Diese werden über die spezielle Laborversion des Tobii Studio betrieben. Durch den z-Tree-Client gibt es eine Schnittstelle zu den Eye-Trackern, über die Befehle versendet werden können. Somit kann z-Tree die Basis-Befehle wie Start/Stopp der Aufnahme und die Marker „Screen Start“ und „Screen Stopp“ an die Eye-Tracking-Software senden. Diese Funktionalitäten wurden alle erfolgreich mit z-Tree geprüft.

7.1.2 Soll-Zustand

Der Soll-Zustand hat sich gegenüber der ersten Analyse nicht geändert. Da mit z-Tree eine gute Steuerung und Überwachung nicht möglich ist, wird eine andere Lösung benötigt. Diese Lösung soll durch die Fernbedienung ermöglicht werden.

7.2 Funktionale Anforderungen für die Fernbedienung

Für die Fernbedienung werden die Anforderungen noch einmal detailliert dargestellt.

- Individuelle Auswahl, an welche Probandenplätze Befehle gesendet werden
- Befehle können in ihrer Reihenfolge und Anzahl frei wählbar gesendet werden

Name	Beschreibung
Start der PCs	Zentrales Hochfahren der Befragungsplatz-PCs
Eye-Tracking-Projekt bereinigen	Löschen von alten Aufnahmen und Übertragung eines neuen Tobii Studio-Projekts
Aktivieren von Tobii Studio	Starten des Tobii Studios an den Befragungsplatz-PCs und Vorbereiten der Aufnahmefähigkeit
Starten von z-Leaf	z-Leaf auf den Befragungsplatz-PCs starten und in den Vordergrund bringen

Starten und Beenden des Ausrichtungsscreens	Individueller Start und Stopp des Ausrichtungsscreens des Eye-Trackings
Anzeige des Ausrichtungsscreens	Zentrales Anzeigen des Ausrichtungsscreens des Befragungsplatz-PCs am Experimentator-PC
Start der Eye-Tracking-Kalibrierung	Zentrales Anzeigen des Kalibrierungsscreens des Befragungsplatz-PCs am Experimentator-PC
Anzeige des Kalibrierungsergebnisses	Zentrales Anzeigen des Kalibrierungsscreens des Befragungsplatz-PCs am Experimentator-PC
Start der Eye-Tracking-Aufnahme	Starten der Eye-Tracking-Aufnahme auf den Befragungsplatz-PCs durch den Experimentator-PC
Stopp der Eye-Tracking-Aufnahme	Stoppen der Eye-Tracking-Aufnahme auf den Befragungsplatz-PCs durch den Experimentator-PC
Sichern der Eye-Tracking-Daten	Sicherung der Eye-Tracking-Daten von den Befragungsplatz-PCs an einem zentralen Speicherort
Befragungsplatz-PC-Feedback	Abfrage von Informationen über den Zustand des Eye-Trackings bei den Befragungsplatz-PCs
Herunterfahren der PCs	Zentrales Herunterfahren aller Befragungsplatz-PCs

7.3 Nicht-funktionale Anforderungen für die Fernbedienung

Die nicht-funktionalen Anforderungen für das Gesamtsystem gelten auch für das Teilsystem der Fernbedienung. Somit gelten dieselben Anforderungen wie in Abschnitt 3.4 des Entwurfs.

8 Entwurf der Fernbedienung

8.1 Name der Fernbedienung

Als Name für die Fernbedienung wurde „HAWKEYE" gewählt. Dieser setzt sich zusammen aus „HAW", an der diese Abschlussarbeit verfasst wurde und „EYE" für Eye-Tracking.
„HAWKEYE" (Falkenauge) ist außerdem auch der Name eines Helden aus dem Marvel-Universum. Dieser Held trifft durch seine außergewöhnlichen Bogenschusstalente immer sein Ziel.

8.2 Konzept

Durch die fehlende Flexibilität in z-Tree wird es nötig, eine externe, zentrale Steuerung einzusetzen. Diese wurde im Rahmen dieser Bachelorarbeit entwickelt und umgesetzt. Da die Vorbereitung eines Experiments, das Ausrichten der Position des Probanden und das Kalibrieren des Eye-Trackings durch z-Tree nur begrenzt möglich ist, sind dies die Hauptaufgaben der Fernbedienung (zentralen Steuerung). Da durch die Steuerung in z-Tree nur alle PCs angesprochen werden, kann der Code (und somit auch der Ablauf) nach dem Start nicht mehr geändert werden. Daher ist die Selektion der PCs und eine freie Auswahl der Befehle eine Anforderung an die Fernbedienung. Das Konzept der Fernbedienung sieht eine zweite Benutzeroberfläche neben z-Tree vor. Diese soll es ermöglichen, die Befehle dynamisch an die Probandenplätze zu senden und ein aktuelles Feedback über den Status des Eye-Trackings zu geben. Das Senden der Befehle über die Fernbedienung sollte unabhängig von z-Tree realisiert werden. Durch die Trennung der Fernbedienung von z-Tree ist die Steuerung unabhängig von dem Status von z-Tree jederzeit möglich.

8.3 Ablauf eines Eye-Tracking-Experiments mit der Fernbedienung

Der Experimentablauf bleibt fast identisch zum geplanten Ablauf in Kapitel 3.5.
Es wird nun aber unterschieden, welche Funktionen über die Fernbedienung erfolgen sollen.

- Vorbereitung der Befragungsplätze und des Experimentatorenplatzes
  - Fernbedienung: Hochfahren der Probanden-PCs
- Anschließen der Eye-Tracker an die Befragungsplatz-PCs
- Laden des z-Tree-Codes und der Eye-Tracking-Einstellungen (z-Tree-Client) am Experimentator-PC
- Fernbedienung: Start von z-Leaf und der Eye-Tracking-Software Tobii Studio an den Befragungsplatz-PCs
- Einlass der Probanden
- Erläuterung des Experiments und des Eye-Trackings
- Kontrolle der Einverständniserklärung zum Eye-Tracking
- Fernbedienung: Durchlauf der Ausrichtung und Kalibrierung
- Ausführen des z-Tree-Codes
- Durchlauf des Experiments
- Auszahlung und Verabschiedung der Probanden
- Fernbedienung: Sicherung von z-Tree- und Eye-Tracking-Daten

9 Realisierung der Fernbedienung

9.1 Mockup-Test

Um eine fehlende Flexibilität wie bei z-Tree auszuschließen, wurde die Fernbedienung im ersten Entwurf mit einem Papier-Mockup geprüft. Hierbei wurden alle Abläufe eines Eye-Tracking-Experiments im Labor mithilfe dem Mockup durchgeführt. Das Resultat war ein GUI-Design, welches die Abläufe eines Eye-Tracking-Experiments unterstützt. Durch diesen Test konnte auch ermittelt werden, ob die geplante Funktionalität der Fernbedienung ausreichend für die Steuerung eines Eye-Tracking-Experiments ist.

9.2 GUI der Fernbedienung

Die GUI der Fernbedienung wurde anhand des erstellten Mockups designt. Hierbei wurden einige Elemente für eine bessere Bedienbarkeit gegenüber des Mockups geändert.
Es wurde eine Ampel eingefügt, die den Verbindungsstatus farblich anzeigt, ob der Client mit dem Sever verbunden ist. Das Feedback des Eye-Trackers wurde mithilfe von Balken zeitlich dargestellt. Des Weiteren wurde eine Verlaufsanzeige ergänzt, die versendete Befehle anzeigt. Einige Befehle wurden mit einer Sicherungsabfrage versehen, um ein versehentliches Ausführen zu verhindern. Die GUI wurde mithilfe des GUI-Builders aus der NETbeans IDE erstellt.

Abbildung 9.1: HAWKEYE Gui im Startzustand

9.3 Netzwerkkommunikation

Die Anforderungen für die Netzwerkkommunikation im Forschungslabor sind nicht nur von der Anwendung geprägt, sondern auch von der Netzwerksicherheitsstruktur des Regionalen Rechenzentrums der Universität Hamburg. So werden zum Beispiel WebSockets im Netzwerk blockiert, die somit nicht zur Kommunikation genutzt werden können. Die Verwendung eines Message-Queuing wie RabbitMQ hat den Nachteil eines separaten Servers, der die Nachrichten verwalten muss. Dieser Server wäre eine zusätzliche Instanz, die im Netzwerk gewartet werden muss. Ein UDP Socket wäre die ressourcensparenste Lösung, hätte aber durch die lose Bindung zum Client Stabilitätsprobleme.

Durch die Verwendung eines TCP-Socket kann ein guter Kompromiss gefunden werden. Im Gegensatz zum UDP wird die Reihenfolge und Zustellung der Pakete garantiert. Die Kommunikation kann vollständig in den Fernbedienungscode integriert werden, womit die Wartung im Gegensatz zum Message-Queuing minimiert wird. Eine Freigabe des Ports in den lokalen Firewalls der Clients (Befragungsplatz-PCs) reicht, da die direkte Kommunikation durch das Rechenzentrum nicht blockiert wird. Durch die Verwendung eines TCP-Socket können die Befehle auch in Echtzeit gesendet und empfangen werden.

9.4 Aufbau der Fernbedienung

Aufgrund der Struktur des Forschungslabors mit 30 Clients/Befragungsplatz-PCs und einem Server/Experimentator-PC bietet sich ein Client-Server-Aufbau für die Fernbedienung an. Der Server beinhaltet die GUI, welche die Schnittstelle zum Experimentator ist. Diese dient als Eingabe und Feedback über die Clients. Über die TCP-Verbindung kann der Server die Befehle an die Clients schicken, die diese lokal abarbeiten.
Diese Befehle sind:

- Programme öffnen, wie Tobii Studio oder z-Tree
- Programme konfigurieren
- Befehle an den Eye-Tracker senden über die lokale z-Tree-Client Instanz
- Sichern von Ergebnissen

Der Server wird nicht durch Client-Prozesse blockiert und kann jederzeit Eingaben entgegennehmen. Der Client verfügt über einen zusätzlichen Watchdog-Prozess. Dieser Prozess ermöglicht es, am Server festzustellen, wenn ein Client keine Verbindung zum Server mehr hat. Um dieses sicherzustellen, sendet jeder Client in regelmäßigen Abständen einen „Herzschlag“ an den Server. Sollte in einem gewissen Zeitfenster kein Herzschlag von einem Client empfangen werden, wird die Verbindung zu dem entsprechenden Client getrennt und an der Server GUI angezeigt.

9.5 Spezielle Anpassung an das Labor

Aufgrund der Anonymität der Probanden muss auf den Befragungsplatz-PCs im WiSo-FL ein erhöhtes Sicherheitskonzept realisiert werden. Dies beinhaltet ein besonderes Windows-Benutzerprofil. Dieses Benutzerprofil, „Labuser“ zeichnet sich durch sehr eingeschränkte Rechte aus. So besitzt dieses Profil nur Standard-Windows-Rechte und eine Art Kiosk-Modus. Für die Eye-Tracking-Software Tobii Studio, die für eine Eye-Tracking-Aufnahme aktiv sein muss, ist aber ein Start mit Administratorenrechten notwendig. Dies wird über ein zweites Benutzerprofil erreicht, den „Labadmin“. Dieses Benutzerprofil darf aber auf Grund des Sicherheitskonzepts von Probanden nicht bedienbar sein. Das Programm „PsExec“ ermöglicht einen Fernzugriff und damit Befehle zielgerichtet auf einem entfernten Computer unter einem bestimmten Benutzer auszuführen. Diese Befehle werden nach einer Authentifizierung in der Windows Command-Line des jeweiligen Computers auf Ebene des Benutzers ausgeführt. So kann ein Befehl unter dem User „Labadmin“ ausgeführt werden und auf dem Bildschirm des eingeloggten „Labuser“ angezeigt werden. So wird es möglich, Programme mit Administratorenrechten unter dem „Labuser“ anzuzeigen. Das Senden eines Befehls dauert aber durch die Authentifizierung zwischen 30 und 200 Sekunden. Somit ist es mit diesem Programm

nicht möglich, ein Eye-Tracking-Experiment zu steuern. Es ist aber möglich, über das Programm PsExec den Client der Fernbedienung mit Administratorenrechten unter dem Profil des „Labadmin“ zu starten und im angemeldeten Profil des „Labuser“ anzuzeigen. Durch das Starten des Fernbedienungs-Client mit Administratorenrechten verfügt dieser auch automatisch über Administratorenrechte und kann nun die weitere Steuerung mit einer sehr viel geringeren Verzögerung durchführen. Dies ist möglich, da nicht, wie bei PsExec für jeden Befehl eine vollständige Authentifizierung über den Domain Controller nötig ist. Ein positiver Nebeneffekt ist, dass sich so auch der Client-Teil der Fernbedienung remote starten lässt.

9.6 Kommunikation im Detail

Die Fernbedienung verwendet drei Kommunikationsmethoden: eine direkte über PsExec, eine über den Fernbedienungs-Client und eine received-Methode.

9.6.1 Direkte Methode über PsExec

Die Fernbedienung startet einen neuen Java-Thread. Dieser ruft über den Java-ProcessBuilder das lokale Programm PsExec auf. PsExec ist ein Tool, welches sich in Microsoft Sysinternals PsTools befindet. Es ermöglicht auf Windows-Computern remote Programme auszuführen. Hierfür wird ein Adminkonto benötigt, welches die notwendigen Rechte besitzt. Über die Parameter „Ziel“, „Benutzername“, „Passwort“, „befehl“ kann die Fernbedienung die entsprechenden Befehle an den Befragungsplatz-PC senden. Der so gesendete Befehl wird beim Ziel unter dem angegeben Benutzer nach einer erfolgreichen Authentifizierung durch den Domain-Controller ausgeführt.

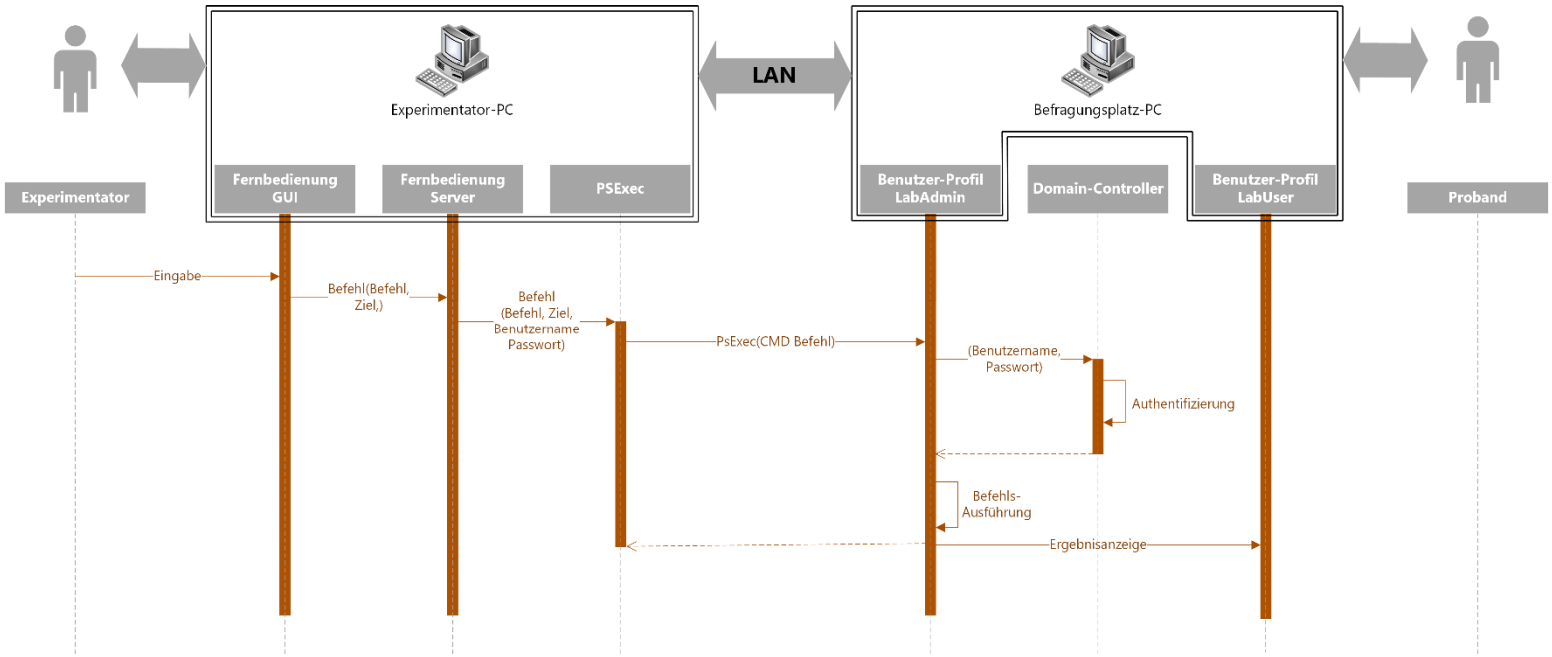

Abbildung 9.2: Direkte Methode über PsExec

9.6.2 Methode über den Fernbedienungs-Client

Als Voraussetzung für diese Art der Kommunikation muss am Ziel-Computer der Fernbedienungsclient mit Windows-Administratorenrechten gestartet sein. Der Start des Fernbedienungs-Clients ist manuell oder mit der direkten Methode der Fernbedienung möglich. Jeder gestartete Fernbedienungsclient verbindet sich über einen Java Socket mit der Fernbedienung. Für jeden verbunden Fernbedienungs-Client wird ein eigener Java Thread gestartet. Wird ein Befehl über die Fernbedienungs-GUI gestartet, wird dieser an den entsprechenden, für den Client zuständigen, Thread übergeben. Der Thread übermittelt den Befehl über den Socket an den Fernbedienungs-Client. Dieser führt den Befehl aus. Der Befehl wird unter dem Windows-Benutzer mit dessen Rechten ausgeführt, mit dem der Fernbedienungs-Client gestartet wurde. Durch die fehlende Prüfung des Domain Controlles können die Befehle in Echtzeit gestartet werden.

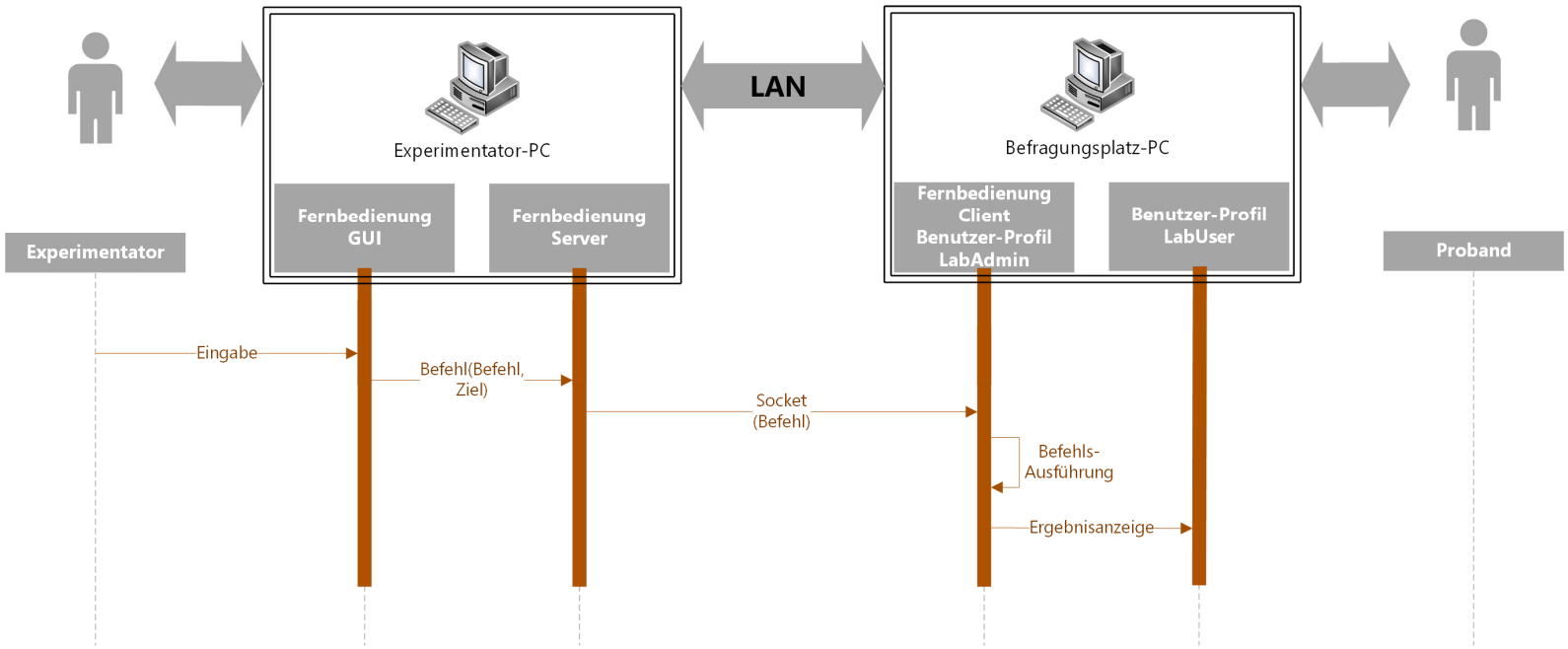

Abbildung 9.3: Methode über den Fernbedienungsclient

9.6.3 received Methode

Die Fernbedienung hat zu jedem gestarteten Fernbedienungsclient eine Java-Socket-Verbindung. Hierüber hat der Client die Möglichkeit, Nachrichten an die Fernbedienung zu senden. Diese Nachrichten werden von der Fernbedienung als Input verarbeitet.

9.7 Funktionen der Fernbedienung im Detail

PC starten:

- Dieser Befehl ermöglicht das Hochfahren der Befragungsplatz-PCs.
- Die Fernbedienung erzeugt ein WoL-Paket, welches über einen Broadcast an die MAC-Adresse des Befragungsplatz-PCs gesendet wird. Durch die BIOS-Einstellungen des Befragungsplatz-PCs reagiert dieser auch im ausgeschalteten Modus auf das WoL-Paket und schaltet sich ein.
- Als Schutz vor einer Überlastung des Stromnetzes im Labor prüft die Fernbedienung, wie viele Befragungsplatz-PCs gestartet werden sollen. Diese werden zeitversetzt in Vierergruppen gestartet.

PC herunterfahren:

- Dieser Befehl ermöglicht das Herunterfahren der Befragungsplatz-PCs. Für diesen Befehl muss eine Sicherheitsabfrage bestätigt werden, sodass die Befragungsplatz-PCs nicht versehentlich heruntergefahren werden.
- Die Fernbedienung sendet über die direkte Methode den Befehl „shutdown“ an die Befragungsplatz-PCs. Auf den Befragungsplatz-PCs wird lokal der Befehl „shutdown.exe /s /f /t 10“ ausgeführt. Dieser Windows-Befehl zwingt alle Anwendungen zum Schließen und fährt den Computer nach zehn Sekunden herunter.

Client starten:

- Auf den Befragungsplatz-PCs wird der Fernbedienungs-Client gestartet.
- Die Fernbedienung sendet über die direkte Methode den Befehl „start“. Dieser öffnet auf den Befragungsplatz-PCs eine „Windows Eingabeaufforderung“ über welche der Fernbedienungs-Client (HAWKEYEleaf.jar) gestartet wird. Die jar, die gestartet wird, befindet sich für eine bessere Wartbarkeit auf einem zentralen Netzlaufwerk „\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software \HAWKEYEleaf\“. Somit ist ein Austausch oder eine Versionsänderung des Fernbedienungs-Clients zentral möglich. Der gestartete Client verbindet sich mit der Fernbedienung am Experimentator-PC. Bei einer

erfolgreichen Verbindung wartet der Client auf Befehle vom Experimentator-PC und schickt regelmäßig einen Herzschlag an diesen.

Tobii-Daten löschen:

- Auf den Befragungsplatz-PCs werden die gespeicherten Eye-Tracking Daten gelöscht. Für diesen Befehl muss eine Sicherungsabfrage bestätigt sein, sodass die Daten nicht ungewollt gelöscht werden.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen Daten-Löschbefehl an den Befragungsplatz-PC. Der Fernbedienungs-Client löscht über die Java Class File den Projektordner ("C:\Users\Public\Documents\Tobii Studio Projects") und alle Daten, die dieser enthält.

Tobii-Projekt übertragen:

- Ein leeres Eye-Tracking-Projekt wird auf den Befragungsplatz-PC übertragen. Für diesen Befehl muss eine Sicherungsabfrage bestätigt sein, sodass die Daten nicht ungewollt überschrieben werden.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen Daten-Kopierbefehl an den Befragungsplatz-PC. Der Fernbedienungs-Client kopiert über die Java Class File den Projektordner von einem Netzwerkpfad (\\fs-s-clrfsd03.ad.uni-hamburg.de\\explab-ztree\users\Software\HAWKEYEleaf\) auf den Befragungsplatz-PC ("C:\Users\Public\Documents\Tobii Studio Projects") und alle Daten, die dieser enthält.

Tobii Studio starten:

- Am Befragungsplatz-PC wird das Tobii Studio gestartet. Nach dem Start des Tobii Studios wird ein Skript ausgeführt, welches alle nötigen Einstellungen tätigt, sodass eine Eye-Tracking-Aufnahme durchgeführt werden kann.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen „startTobiStudio“-Befehl an den Befragungsplatz-PC. Der Fernbedienungs-Client startet über den Java ProcessBuilder die Applikation Tobii Studio lokal auf dem Befragungsplatz-PC. Nach einem Timeout, der so bemessen ist, dass sich das Tobii Studio starten konnte, wird ein Klickskript durch die Javaklasse Robot ausgeführt. Dieses Klickskript führt folgende Aktionen im Tobii Studio aus:
    - Auswahl des Eye-Tracking-Projekts und dessen Start
    - Einstellung der Taste, welche das Eye-Tracking abbricht
    - Einstellung des Probanden
    - Bestätigung für den Aufnahmescreen

zLeaf starten:

- Dieser Befehl startet den z-Leaf am Befragungsplatz-PC. Nach dem Start von z-Leaf wird ein Skript ausgeführt, welches den z-Leaf in den Vordergrund des Probanden-Desktop bringt.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen „zLeafStarten"-Befehl an den Befragungsplatz-PC. Der Fernbedienungs-Client startet über den Java ProcessBuilder ein VBS-Skript, welches sich auf dem Netzwerkspeicher (\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\zLeafnet.vbs) befindet. Das VBS-Skript prüft, ob der z-Tree-Netzwerkpfad (\\Wiso-srv-db01\ztree) unter dem Netzwerklaufwerk („X") eingebunden ist. Wenn das Laufwerk nicht eingebunden ist, bindet es das Laufwerk ein. Nach einem Timeout wird mit dem Java ProcessBuilder über Laufwerk X des Befragungsplatz-PC zleaf.exe gestartet. Das Starten über ein Netzwerklaufwerk ist nötig, damit z-Tree die eingebundenen Mediendateien wie Bilder, Sounds oder den z-Tree-Client verwenden kann. Nach einem Timeout von 500ms wird über den Java ProcessBuilder das VBS-Skript „TopzLeaf" (\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\TopzLeaf.vbs) ausgeführt. Dieses Skript setzt den z-Leaf-Screen über alle anderen Screens in den Vordergrund.

Ausrichtung starten:

- Auf dem Befragungsplatz-PC wird der Ausrichtungs-Screen gestartet und in den Vordergrund des Befragungsplatz-Desktop gebracht.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen „Ausrichtung"-Befehl an den Befragungsplatz-PC. Der Fernbedienungs-Client startet über den Java ProcessBuilder den z-Tree-Client, der sich im Netzwerkpfad „\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\ZTClient.exe" befindet. Über den mitgeschickten Parameter „EyeWindow" wird der Ausrichtungs-Screen gestartet. Nach einem Timeout von 500ms wird über den Java ProcessBuilder das VBS-Skript „TopAusrichtung" (\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf \TopAusrichtung.vbs) ausgeführt. Dieses Skript setzt den Ausrichtungs-Screen über alle anderen Screens in den Vordergrund.

Ausrichtung anzeigen:

- Der Ausrichtungs-Screen des Befragungsplatz-PCs wird auf den Experimentator-PC als Bild übertragen und angezeigt.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen „AusrichtungBild"-Befehl an den Befragungsplatz-PC. Dieser führt über

Javaklasse Robot die Methode „createScreenCapture“ aus. Diese Methode erstellt auf dem Befragungsplatz-PC einen Screenshot des aktuellen Bildschirminhalts. Das so entstandene Bild wird an die zentrale Stelle der Fernbedienung gesendet und dem Experimentator angezeigt.

Ausrichtung beenden:

- Der Ausrichtungs-Screen wird am Befragungsplatz-PC beendet.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen „AusrichtungExit“-Befehl an den Befragungsplatz-PC. Der Fernbedienungs-Client startet über den Java ProcessBuilder einen Windows-taskkill-Befehl. Über den Parameter „/f /im \zTClient.exe\“ wird der zTClient.exe auf dem Befragungsplatz-PC beendet und somit auch der über z-Tree-Client geöffnete Ausrichtungs-Screen. Nach einem Timeout von 500ms wird über den Java ProcessBuilder das VBS-Skript „TopzLeaf“ (\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software \HAWKEYEleaf\TopzLeaf.vbs) ausgeführt. Dieses Skript setzt den z-Leaf-Screen über alle anderen Screens in den Vordergrund.

Kalibrierung starten:

- Der Kalibrierungsscreen wird am Befragungsplatz-PC gestartet. Nach der Durchführung wird das Ergebnis auf einem Netzwerkspeicherplatz gespeichert.
- Die Fernbedienung speichert die Aktivierungszeit für den jeweiligen Befragungsplatz-PC und sendet über den Fernbedienungs-Client einen „Kalibrierung“-Befehl an den Befragungsplatz-PC. Der Fernbedienungs-Client startet über den Java ProcessBuilder den z-Tree-Client (\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software \HAWKEYEleaf\ZTClient.exe) mit dem Parameter „Calibrate“. Nach einem Timeout von 500ms wird über den Java ProcessBuilder das VBS-Skript „TopKalibrierung“ (\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\TopKalibrierung.vbs) ausgeführt. Dieses Skript setzt den Kalibrierungsscreen über alle anderen Screens in den Vordergrund. Nachdem die Kalibrierungsprozedur abgeschlossen ist, beendet sich der z-Tree-Client automatisch und speichert das Kalibrierungsergebnis im Tobii Studio. Der so entstandene Abweichungsscreen wird zentral auf dem Netzlaufwerk (\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software \HAWKEYEleaf\calibrate\) als Bild unter dem Namen „Probanden-Computername“ und „Datum/Zeit“ abgelegt.

Kalibrierung anzeigen:

- Am Experimentator-PC wird von der Fernbedienung geprüft, ob es einen aktuellen Kalibrierungscreen des Befragungsplatz-PCs auf dem Netzwerkspeicher gibt. Wenn es einen gibt, wird dieser am Experimentator-PC angezeigt.
- Die Fernbedienung sucht nach einer Abweichungsscreen-Bilddatei auf dem Netzwerklaufwerk (\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\calibrate\). Mit Hilfe des Namens des Abweichungsscreens wird geprüft, ob es sich um den ausgewählten Befragungsplatz-PC handelt. Wenn die Erstellungszeit (die auch im Namen des Bildes gespeichert ist) nach der gespeicherten Zeit liegt, die für den Befehl „Start Kalibrierung“ gespeichert wurde, wird es dem Experimentator an der Fernbedienung angezeigt.

Tobii Aufnahme starten:

- Die Eyetracking-Aufnahme wird am Befragungsplatz-PC gestartet.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen „StartRec“ Befehl an den Befragungsplatz-PC. Der Fernbedienungs-Client startet über den Java ProcessBuilder den z-Tree-Client der sich im Netzwerk Pfad „\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\ZTClient.exe“ befindet. Über den mitgeschickten Parameter „Start“ wird im Tobii Studio die Eye-Tracking-Aufnahme gestartet.

Tobii Aufnahme beenden:

- Die Eye-Tracking-Aufnahme wird am Befragungsplatz-PC beendet.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen „EndRec“-Befehl an den Befragungsplatz-PC. Der Fernbedienungs-Client startet über den Java ProcessBuilder den z-Tree-Client, der sich im Netzwerk Pfad „\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\ZTClient.exe“ befindet. Über den mitgeschickten Parameter „Finish“ wird im Tobii Studio die Eye-Tracking-Aufnahme beendet.

Tobii Daten speichern:

- Die gesammelten Eye-Tracking Daten werden auf eine zentrale Stelle auf einen Netzwerkspeicherplatz kopiert.
- Die Fernbedienung sendet über den Fernbedienungs-Client einen „DataSichern“-Befehl an den Befragungsplatz-PC. Der Fernbedienungs-Client erzeugt auf dem Netzlaufwerk („\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\erg) einen Ordner mit dem Namen des Befragungsplatz-PCs. Die Eye-Tracking-Daten

des Befragungsplatz-PCs („C:\Users\Public\Documents\Tobii Studio Projects") werden mithilfe der Java Class File in den neu erzeugten Netzwerkordner kopiert.

Probanden Status:

- Der Befragungsplatz-PC überträgt in regelmäßigen Intervallen den Status des Eye-Trackers an den Experimentator-PC.
- Durch den Start des Fernbedienungs-Client am Befragungsplatz-PC hat sich ein Status Thread in einer Endlosschleife gestartet. Dieser Java-Thread fragt in einem Intervall alle fünf Sekunden über den Java ProcessBuilder den z-Tree-Client, der sich im Netzwerkpfad „\\fs-s-clrfsd03.ad.uni-hamburg.de\explab-ztree\users\Software\HAWKEYEleaf\ZTClient.exe“ befindet, mit dem mitgeschickten Parameter „Status“, wie der momentane Status des Eye-Trackers ist. Das Ergebnis dieser Abfrage wird in einem Intervall von fünf Sekunden an die Fernbedienung geschickt. Sollte der z-Tree-Client nicht unterhalb der fünf Sekunden einen Status liefern, wird die Abfrage abgebrochen und ein Status „unbekannt“ an die Fernbedienung gesendet.
  Dieser Status wird dem Experimenatator über die Fernbedinungs-GUI angezeigt.

10 Validierung der Fernbedienung

10.1 Validierung der funktionalen Anforderungen

Durch die Erstellung der Fernbedienung konnte der letzte offene Punkt der funktionalen Anforderungen erfüllt werden. Die Fernbedinung ermöglicht nun eine dynamische und zentrale Steuerung, womit es möglich wird, Eye-Tracking-Experimente simultan an 30 Probandenplätzen durchzuführen.

10.2 Nicht-funktionale Anforderungen

Die nicht-funktionalen Anforderungen können nicht immer direkt geprüft werden. Aus diesem Grund wurden Maßnahmen ergriffen, die einen positiven Einfluss auf die nicht-funktionalen Anforderungen haben. So wurden zum Beispiel für die ethischen Anforderungen und den Datenschutz Abläufe erzeugt, die nicht diskriminieren und die Anonymität der Probanden wahren. Auch wurde die Technik so eingerichtet, dass es keine Möglichkeit gibt, über das Eye-Tracking das Gesicht oder andere Merkmale, die Aufschluss über die Identität des Probanden geben könnten, zu erfassen. Für die Anforderung der Übertragbarkeit wurde die GUI modular und dynamisch erstellt. So ist es möglich, die Software schnell an weniger oder mehr Probandenplätze anzupassen. Es wurden auch bei der Programmierung der Fernbedienung verschiedene Werkzeuge wie die NetBeans IDE und Softwarearchitekturstile (Client-Server-Modell) zur Erfüllung der nicht-funktionalen-Anforderungen eingesetzt.

11 Fazit / Ausblick

11.1 Zusammenfassung

Im Rahmen dieser Bachelorarbeit sollte ein Eye-Tracking-System in das WiSo-Forschungslabor sinnvoll eingebunden werden. Durch die hauptsächliche Verwendung von z-Tree als Experimentalsoftware wurde eine Integration mit dieser Software nötig. Somit umfasste die Arbeit die Selektion der Eye-Tracking-Hardware, das Erstellen eines Konzepts der Integration von z-Tree und dessen Umsetzung und eine Steuerung, die es ermöglicht ein Eye-Tracking-Experiment simultan an 30 Probandenplätzen durchzuführen.

Für die Hardware wurde ein Tobii X60 gewählt, welcher videobasiert Messungen von Pupillen- und Hornhautreflexionen durchführt. Dies ermöglicht Eye-Tracking Messungen ohne größere Ablenkung des Probanden. Die Vorbereitungen für diese Eye-Tracking-Methode beschränken sich auf das Ausrichten des Probanden und Kalibrieren des Eye-Trackers, was es ermöglicht 30 Probanden simultan zu betreuen. Durch die Verwendung von Infrarotlicht und Kameras entstehen keine gesundheitlichen Risiken für den Probanden.

Für die Interaktion des Eye-Trackings und z-Tree wurde das Konzept der „z-Tree-Integration in Eye-Tracking-Software“ gewählt. Dieses Konzept sieht vor, dass z-Tree Informationen an die Eye-Tracking-Software schickt und so das Eye-Tracking-Video mit Informationen anreichert. Für diesen Zweck wurde die Schnittstelle z-Tree-Client entwickelt. Sie stellt eine Verbindung zu dem lokalen Eye-Tracking-Programm Tobii Studio dar. Durch diese Schnittstelle kann die Eye-Tracking-Aufnahme gestartet und gestoppt, die Ausrichtung und Kalibrierung gestartet sowie Marker gesetzt werden. Die Marker ermöglichen es die Eye-Tracking-Aufnahme automatisiert zu schneiden und stellen so eine enorme Arbeitsersparnis dar.

Im Verlauf der Arbeiten wurde festgestellt, dass die Steuerung des Eye-Tracking nicht ohne eine zusätzliche Software realisierbar ist. Aus diesem Grund wurde im zweiten Teil dieser Bachelorarbeit eine Steuerung mit dem Namen HAWKEYE realisiert. Durch die Fernbedienung können in Echtzeit an jeden einzelnen der 30 Probandenplätze Befehle gesendet werden. Dies ermöglicht es, auf Probleme bei der Ausrichtung des

Probenden und Kalibrierung des Eye-Trackers zu reagieren. Zusätzlich stellt die Fernbedienung eine Unterstützung bei der Vor- und Nachbereitung des Eye-Tracking-Experiments zur Verfügung.

11.2 Fazit

Im Zuge dieser Arbeit wurde eines der größten Eye-Tracking-Labore in Europa realisiert.
Es wurden bereits mehrere Eye-Tracking-Experimente mit insgesamt weit über 1000 Probanden durchgeführt. Hierbei führten Forscher aus verschiedenen Ländern die Experimente durch. Damit ist das Eye-Tracking-Labor über die Tore Hamburgs bekannt.
Die Anmeldezahlen der Probanden unterscheiden sich bei Eye-Tracking-Experimenten nicht deutlich von anderen Experimenten. Somit kann davon ausgegangen werden, dass das Eye-Tracking von den Probanden akzeptiert wird. Bis jetzt (Stand August 2019) gab es nur einen Probanden, der aufgrund von gesundheitlichen Bedenken ein Eye-Tracking-Experiment abgebrochen hat.

Es finden regelmäßige Präsentationen für Studenten und Forscher statt, um das Eye-Tracking-System vorzustellen. Schulungen für Forschende, die Eye-Tracking-Experimente durchführen wollen, erfolgen meist in einer kleinen Gruppe live am Eye-Tracking-System. Die ersten Sessions werden mit einer Betreuung des Forschungslabors durchgeführt, bis der Forscher die Routine hat, die Eye-Tracking Experimente ohne technische Betreuung durchzuführen.

Durch Verwendung der Fernbedienung zur Nacharbeit eines Experiments und das anschließende Durchlaufen des Klickskripts, stehen für die Forscher schon kurze Zeit nach dem Experiment auswertbare Datensätze zur Verfügung.

Durch die Trennung der Steuerung des Eye-Trackings von z-Tree, (mit Hilfe der Fernbedienung HAWKEYE) ist es auch möglich, Experimente ohne z-Tree durchzuführen. So können zum Beispiel auch Web-Experimente über Lime oder oTree durchgeführt werden.

Durch die enorme Zeitersparnis in der Vorbereitung von Eye-Tracking-Experimenten wurde eine Lite-Version der Eye-Tracking-Steuerung HAWKEYE mit dem Namen „MoinPCs“ für Standard-Experimente im WiSo-FL erstellt. Dies ermöglicht eine bedeutend schnellere Vorbereitung für Experimente durch das zentrale Hochfahren, Neustarten, Herunterfahren der PCs und der Verbindung von z-Leaf mit z-Tree. Als

zusätzliche Funktion für MoinPCs wurde das zentrale Öffnen von Browsern hinzugefügt.

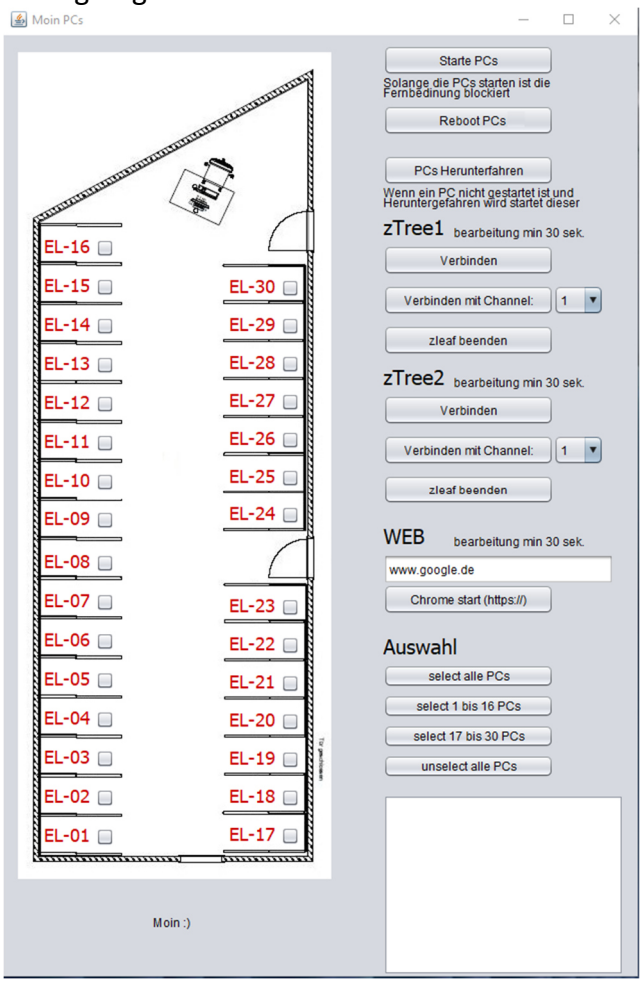

Abbildung 11.1: GUI von MoinPCs (Lite-Version von HAWKEYE)

11.3 Ausblick

Die Hardware besteht momentan aus 60-Hz-Trackern. Durch ein Hardwareupdate wäre es möglich, durch eine höhere Abtastrate zu erzielen. Somit können Eye-Tracking-Daten mit höherer Qualität erfasst und verarbeitet werden. Ab einer Abtastrate von über 500 Hz ist es möglich zuverlässig Sakkaden zu messen, somit wäre es unter Umständen möglich Lesestudien durchzuführen.

Die Marker in z-Tree-Experimenten bieten eine enorme Erleichterung bei der Auswertung der Eye-Tracking-Daten. Bisher gibt es bei Web-Projekten keine Möglichkeit diese zu setzen. Ein Prototyp für Webbrowser hat aber ergeben, dass das

Konzept der Marker auch für Webexperimente nutzbar ist. Dies müsste aber aufgrund des geschaffenen Zugriffs des Webbrowsers auf das Betriebssystem unter Sicherheitsaspekten geprüft werden. Wenn dieser Zugriff des Browsers auf das Betriebssystem ermöglicht ist, ist es möglich über Javaskript über den z-Tree-Client Marker zu setzen.

Die Weiterentwicklung der Fernbedienung ermöglicht weitere Verbesserungsmöglichkeiten für eine erhöhte Automatisierung. So wäre es möglich, mithilfe einer Künstliche Intelligenz die Kalibrierungsergebnisse zu validieren, um reliablere Ergebnisse zu erzeugen. Dies könnte mithilfe eines künstlichen neuronalen Netz erfolgen, welches die bisher erhobenen Kalibrierungsergebnisse als Trainingsmaterial nutzen könnte.

Literaturverzeichnis

[Beck 2014] Beck, H.: Behavioral Economics – Eine Einführung, Wiesbaden: Springer Gabler, 2014

[Duchowski 2017] Duchowski, A. T.: Eye Tracking Methodology – Theory and Practice, third edition, Cham: Springer International, Kapitel 5: S. 49-57, 2017

[Fischbacher 1999] Fischbacher, U.: z-Tree – Zurich Toolbox for readymade economic experiments: experimenter's manual. IEW Working paper 21, University of Zurich, 1999

[Fischbacher 2007] Fischbacher, U.: z-Tree: Zurich toolbox for ready-made economic experiments. Experimental Economics 10, Zurich: Springer Nature Switzerland, S. 171-178, 2007

[Fischbacher et al. 2015] Fischbacher, U.; Bendrick, K.; Schmid, S.: z-Tree 3.5 Tutorial and Reference Manual, Zurich: Department of Economics, 194 S., 2015

[Gürth 1982] Gürth, W.; Schmittberger, R.; Schwarze, B.: An experimental analysis of ultimatum bargaining. Journal of Economic Behavior & Organization 3(4): S. 367-388, 1982

[Sommerville 2012] Sommerville, I.: Software Engineering, München: Pearson Deutschland, 2012

[Tobii Technology AB 2014] Tobii Technology AB: Tobii X2-30 Eye Tracker User's manual, Version 1.0.3, 2014, https://www.tobiipro.com/learn-and-support/learn/steps-in-an-eye-tracking-study/setup/ setting-up-and-install-tobii-pro-x2-30-and-x2-60-eye-trackers/ - abgerufen am 10.08.2019

[Whitmire et al. 2016] Whitmire, E.; Trutoiu, L.; Cavin, R.; Perek, D.; Scally, B.; Phillips, J.; Patel, S.: EyeContact: scleral coil eye tracking for virtual reality. Proceedings of the 2016 ACM International Symposium on Wearable Computers 184-191, 2016. doi: 10.1145/2971763.2971771

[Yarbus 1967] Yarbus, A. L.: Eye Movements and Vision. New York: Plenum. 1967

Danksagung

Zuerst gebührt mein Dank Herr Prof. Dr. Sarstedt, für die Betreuung und Begutachtung meiner Bachelorarbeit.

Ich bedanke mich auch beim WiSo-Forschungslabor der Universität Hamburg für die Möglichkeit diese Bachelorarbeit zu erstellen.

Mein besonderer Dank gilt Anne Lerp und Dr. Hannes Lerp die mir mit viel Geduld und Hilfsbereitschaft mir zur Seite standen.

Mein Dank, gilt auch den Forschern Dr. Katrin Gödker und Dr. Moritz Lukas für die Verwendung ihrer Eye-Tracking-Daten zur Erstellung der Auswertungsbilder (Abbildung 5.7 bis 5.10) und des Markerbildes (Abbildung 5.6).

Für das Korrekturlesen möchte ich mich auch bei Nathalie Szymanski, David Lucius, Sven Boris Bornemann und Moritz Molle bedanken.

A. Anhang

A.1 z-Tree-Client-Testprotokoll

Der Test des ZTClient erfolgt anhand einen Test Protokoll manuell.
Da der ZTClient eine Schnittstelle zu dem Tobii Studio darstellt, wäre ein Automatisierter Test sehr komplex. Ein Weiterer Grund ist das es sich um einen spezial Software handelt, die im besten Fall nur einmal geprüft werden muss. Wenn alle Punkte des Test Protokoll erfolgreich erfüllt wurden ist die Übergabe und Entwicklung des ZTClient abgeschlossen.

ZTClient Test Protokoll		Efolgreich JA / NEIN
Status	Fehlerfreies Ausführen	JA / NEIN
	Test Rückgabewert 0 (Eye not Track)	JA / NEIN
	Test Rückgabewert 12 (Eye Track)	JA / NEIN
Connect	Fehlerfreies Ausführen	JA / NEIN
	Test Rückgabewert 0 (successful Connect)	JA / NEIN
	Test Rückgabewert 4 (unsuccessful Connect)	JA / NEIN
EyeWindow	Fehlerfreies Ausführen	JA / NEIN
	GUI Full Screen	JA / NEIN
	GUI Lyout	JA / NEIN
Calibrate	Fehlerfreies Ausführen	JA / NEIN
	Test Rückgabewert 0 (successful)	JA / NEIN
	Test Rückgabewert 1 (fail)	JA / NEIN
	GUI Full Screen	JA / NEIN
	GUI Lyout	JA / NEIN
	Übergabe an Tobii Studio	JA / NEIN
Start	Fehlerfreies Ausführen	JA / NEIN
	Start der Aufnahme in Tobii Studio	JA / NEIN
Finish	Fehlerfreies Ausführen	JA / NEIN
	Stop der Aufnahme in Tobii Studio	JA / NEIN
SceneStart <Parameter>	Fehlerfreies Ausführen	JA / NEIN
	Übergabe an Tobii Studio	JA / NEIN
	Übergabe des Parameter	JA / NEIN
SceneStop <Parameter>	Fehlerfreies Ausführen	JA / NEIN
	Übergabe an Tobii Studio	JA / NEIN
	Übergabe des Parameter	JA / NEIN

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, den _____ _____