

# **BACHELORARBEIT**

zur Erlangung des akademischen Grades

„Bachelor of Science in Engineering“ im Studiengang  
„Smart Homes and Assistive Technologies“

## **Steuerung eines Computers mittels Gesichtsmimik.**

Integration von OpenFace in AsTeRICS und  
Entwicklung von Modellen zur  
Gesichtsmimikerkennung.

Ausgeführt von: Juanita Berenice Muzquiz Gastelum

Personenkennzeichen: 1810768033

1. Begutachter: Martin Deinhofer

Wien, 28.01.2021

## **Eidesstattliche Erklärung**

„Ich, als Autor / als Autorin und Urheber / Urheberin der vorliegenden Arbeit, bestätige mit meiner Unterschrift die Kenntnisnahme der einschlägigen urheber- und hochschulrechtlichen Bestimmungen (vgl. Urheberrechtsgesetz idGf sowie Satzungsteil Studienrechtliche Bestimmungen / Prüfungsordnung der FH Technikum Wien idGf).

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig angefertigt und Gedankengut jeglicher Art aus fremden sowie selbst verfassten Quellen zur Gänze zitiert habe. Ich bin mir bei Nachweis fehlender Eigen- und Selbstständigkeit sowie dem Nachweis eines Vorsatzes zur Erschleichung einer positiven Beurteilung dieser Arbeit der Konsequenzen bewusst, die von der Studiengangsleitung ausgesprochen werden können (vgl. Satzungsteil Studienrechtliche Bestimmungen / Prüfungsordnung der FH Technikum Wien idGf).

Weiters bestätige ich, dass ich die vorliegende Arbeit bis dato nicht veröffentlicht und weder in gleicher noch in ähnlicher Form einer anderen Prüfungsbehörde vorgelegt habe. Ich versichere, dass die abgegebene Version jener im Uploadtool entspricht.“

---

Ort, Datum

---

Unterschrift

## Kurzfassung

Heutzutage lebt eine große Anzahl von Menschen mit Beeinträchtigung in der Europäische Union [1]. Diese sind oft von Armut betroffen und sozial ausgegrenzt [1]. Die Hauptursache für die schlechte Stellung dieser Menschen ist der Mangel an Lösungen, um die Barrieren zu überwinden, die sie daran hindern, an verschiedenen Aktivitäten des modernen Lebens teilzunehmen [2].

Das Ziel dieser Arbeit ist, neue Lösungen für Personen mit Beeinträchtigung, bei denen keine Handfunktion vorhanden ist, zu erstellen, damit diese Computer mittels Gesichtsmimik steuern können. Der Gesichtsausdruck wird mittels des Gesichtserkennungsprogramms OpenFace gescannt.

Zu diesem Zweck wurden drei AsTeRICS-Modelle erstellt: Ein Modell zur Maussteuerung durch Kopfbewegung und Action Units, ein Modell zur Maussteuerung durch Verfolgung der Position der Nase und Action Units und ein Modell zur Maussteuerung durch die Blickrichtung und Action Units. Letztgenanntes Modell ist nur in Verbindung mit einem Grid für unterstützte Kommunikation wie z.B. einem AsTeRICS-Grid zu verwenden.

In dieser Arbeit wurden Tests durchgeführt, welche zeigen, dass die Maussteuerung durch Kopfbewegung oder Verfolgung der Nasenposition zusammen mit weiterer Gesichtsmimik eine Lösung zur Steuerung von Computern darstellen.

Für die Erstellung der Modelle wurde OpenFace in AsTeRICS integriert, indem ein neues AsTeRICS-Plugin erstellt wurde.

**Schlagwörter:** Gesichtsmimik, Gesichtsmimikerkennung, Action Units, AsTeRICS-Modelle, OpenFace, Maussteuerung,

# **Abstract**

Content

**Keywords:** Keyword1, Keyword2, Keyword3, Keyword4, Keyword5

## **Danksagung**

# Inhaltsverzeichnis

1	Einleitung .....	7
1.1	Problemstellung .....	7
1.2	Zielsetzung .....	7
1.3	Motivation .....	8
1.4	Methodisches Vorgehen .....	8
1.5	Strukturierung der Arbeit.....	8
2	Methoden der Mimikerkennung.....	9
2.1	Facial Action Units .....	10
3	OpenFace .....	11
3.1	Gesichtsausdruckserkennung.....	13
3.2	Erkennung von Gesichtsmerkmalen.....	14
3.3	Schätzung der Kopfhaltung .....	16
3.4	Schätzung der Blickrichtung.....	17
3.5	OpenFaceOffline.....	20
4	AsTeRICS Framework .....	22
5	Integration von OpenFace in AsTeRICS .....	23
5.1	Erstellung des Client Socket-Plugins.....	24
6	Erstellung neuer AsTeRICS-Modelle für die Maussteuerung durch Gesichtsmimik	
6.1	28 Maussteuerung durch Kopfbewegung und AUs .....	30
6.2	Maussteuerung durch Verfolgung der Position der Nase .....	34
6.3	Maussteuerung durch Erkennung der Blickrichtung .....	37
7	Resultate.....	40
7.1	Bewertung der Resultate.....	43
8	Schluss .....	44
	Literaturverzeichnis .....	45
	Abbildungsverzeichnis.....	47
	Tabellenverzeichnis.....	49

# **1 Einleitung**

## **1.1 Problemstellung**

In der Europäischen Union leidet ein Sechstel der Bevölkerung an einer körperlichen Beeinträchtigung [2] das sind ungefähr 80 Millionen Menschen [3]. Eine der Hauptursachen für Beeinträchtigung ist das Alter. In Europa nimmt der Anteil älterer Menschen an der Bevölkerung stetig zu, daher wird erwartet, dass die Zahl der Menschen mit Behinderungen innerhalb weniger Jahre merkbar steigt [2]. Obwohl die Diskriminierung von Menschen mit Beeinträchtigung seit langem in der EU verboten ist, sind viele Menschen mit besonderen Bedürfnissen vom sozialen Leben, der Bildung und dem Arbeitsmarkt ausgeschlossen oder deren Teilnahme an diesen elementaren Aktivitäten ist begrenzt [1].

Eine Studie des European Disability Forum (EDF) hat wichtige Daten über die aktuelle Situation dieser Gruppe von Menschen ermittelt. Laut dieser Studie leben 28,7% aller Menschen mit Behinderungen in der EU in Armut und sind sozial ausgesgrenzt. Nur die Hälfte dieser Menschen in Jahr 2017 hatte einen Job, im Vergleich zu 75% bei Menschen ohne Behinderung [1]. Nur 63% der jungen Menschen zwischen 16 und 19 Jahren mit erheblichen Hindernissen für ihre Arbeitsfähigkeit nahmen an Bildung teil, verglichen mit 83% der Menschen ohne Behinderung [2]. Und die Liste der Nachteile, mit denen diese Menschen konfrontiert sind, geht noch weiter [2].

Die Hauptursache für die schlechte Stellung dieser Menschen ist der Mangel an Lösungen, um die Barriere zu überwinden, sodass sie an den unterschiedlichen Aktivitäten des modernen Lebens teilnehmen können. Es gibt noch nicht genügend Werkzeuge der Hilfestellung, um die Bedürfnisse dieser Menschen zu erfüllen [2]. Da der Zustand, die Umstände, die Fähigkeiten und die Bedürfnisse jeder Person unterschiedlich sind, sollte es auch unterschiedliche Lösungen geben, die für jede Person und jede Situation geeignet sind.

Daher es ist besonders wichtig, neue und unterschiedliche Lösungen mit Hilfe der Technologie zu entwickeln, die an die Bedürfnisse jedes Einzelnen angepasst sind, um Menschen mit Beeinträchtigung in das moderne Leben zu integrieren und ihre Lebensqualität zu verbessern.

## **1.2 Zielsetzung**

Ziel dieser Arbeit ist, eine neue Lösung für Personen mit Beeinträchtigungen, bei denen keine Handfunktion vorhanden ist, zu erstellen, damit diese Computer steuern können. Diese Lösung soll einfach zu bedienen und wirtschaftlich zugänglich sein.

Dafür wird ein AsTeRICS-Modell entwickelt, das den Gesichtsausdruck scannt, um eine Computermaus zu steuern, wenn bestimmte Gesichtsausdrücke ausgeführt werden.

## 1.3 Motivation

Heutzutage ist der Computer ein wichtiger Bestandteil unserer Gesellschaft und unseres Alltags. Einen Computer steuern zu können ist nicht nur Grundvoraussetzung für das Berufsleben, sondern auch für viele private Aufgaben und Beschäftigungen. Ein Computer kann Menschen mit Behinderung helfen, viele Kommunikations-, Lern- oder Bewegungsbarrieren zu überwinden.

Aus diesem Grund bietet dieses Projekt die Möglichkeit der Nutzung von Computer für Menschen mit Behinderung, bei denen keine Handfunktion vorhanden ist, an. Die Verwendung eines Computers mittels Gesichtsausdrucks eröffnet zahlreiche und vielfältige Möglichkeiten. Einige dieser Möglichkeiten sind: der Zugang zu Unterstützter Kommunikation (UK), die Steuerung eines Smart Home Systems, der Zugang zu Online-Banking, Teilnahme in Social Media, Teilnahme am Fernstudium und Nutzung zahlreicher Computerprogramme und Applikationen. Auf diese Weise können Menschen mit Beeinträchtigungen sich in das soziale Leben, das Bildungswesen und in den Arbeitsmarkt integrieren und es damit schaffen, ihre Lebensqualität zu verbessern.

## 1.4 Methodisches Vorgehen

Die Vorgehensweise für die Integration einer modifizierten Version von OpenFace [4] in AsTeRICS [5] und die Entwicklung von Modellen zur Gesichtsausdruckserkennung ist wie folgt:

In erster Linie wurde eine neue AsTeRICS Komponente erstellt (der ClientSocket-Plugin), welche die gestreamten Daten von OpenFace empfängt und für die anderen Komponenten zur Verarbeitung zur Verfügung stellt. Zweitens wurden neue AsTeRICS Modelle erstellt, welche die neue ClientSocket Komponente verwendet um die Computermaus mittels Gesichtsmimik zu steuern. Schließlich wurden einige Tests durchgeführt, um die Funktion der Maus durch die neuen Modelle zu bewerten.

## 1.5 Strukturierung der Arbeit

Das erste Kapitel enthält die Einleitung zu dieser Arbeit. Das zweite Kapitel gibt einen Überblick über die am meisten verwendeten Gesichtsmimikerkennungs-Methoden. Das dritte Kapitel beinhaltet eine Einführung in die Anwendung des Gesichtsmimikerkennungs-

Programms OpenFace, welches für dieses Projekt verwendet wird. Das vierte Kapitel ist eine Einführung in das AsTeRICS Framework. Im fünften Kapitel wird Integration von OpenFace erklärt. In Kapitel sechs werden verschiedene erstellte AsTeRICS Modelle erklärt. Die Resultate und ihre Bewertung werden im siebten Kapitel behandelt. Schließlich werden in Kapitel acht die Erkenntnisse und Schlussfolgerungen zu dieser Arbeit präsentiert.

## 2 Methoden der Mimikerkennung

Das Erkennen von Gesichtsausdrücken war für viele Disziplinen schon immer ein Thema von großem Interesse, denn der Gesichtsausdruck liefert viele Informationen über eine Person, zum Beispiel über ihr Geschlecht, ihre Herkunft, ihr Alter, ihren Geisteszustand und ihren Gesundheitszustand. Unser Gesichtsausdruck spielt auch eine Schlüsselrolle in der Kommunikation, da er die Grundlage der nonverbalen Kommunikation ist [6]. Zusätzlich hilft uns die Zusammensetzung unserer Gesichtsmuskeln, Menschen von anderen zu unterscheiden und Menschen zu identifizieren. Vor allem der letzte Punkt hat zur Entwicklung von Techniken beigetragen, mit denen Gesichtsausdrücke schnell und zuverlässig erkannt werden können [6].

Die erste systematische Arbeit über die spezifischen Wirkungen der Gesichtsmuskeln und ihre emotionalen Bedeutungen der Menschengesicht und Mimiksprache. Diese wurde im Jahr 1969 von Hjortsjo, Professor für Anatomie an der Universität Lund in Schweden, entwickelt. Sein Handbuch enthält eine Kodierung und Dekodierung von Gesichtsausdrücken. Dank seiner Methode ist es möglich, die Kontraktionen der Gesichtsmuskeln einzeln oder in Kombination zu bestimmen [7].

Die am weitesten verbreitete Methode zur Gesichtsausdruckserkennung ist das Facial Action Coding System (FACS). Dieses System wurde von Paul Ekman und Vincent W. Friesen in Jahr 1978 entwickelt. [6]

Das Facial Action Coding System (FACS) ist ein beschreibendes System, welches auf einer detaillierten Beschreibung von Veränderungen im Aussehen aufgrund von Gesichtskontraktionen basiert [6]. Die Techniken von Ekman und Friesen (FACS) können zwischen Bewegungen unterscheiden, welche relevant für das Spezifizieren von Emotionen sind und anderen Bewegungen. Das System ist auch in der Lage eine simulierte Emotion von einer echten zu unterscheiden, da bei unechten Emotionen die angeforderten Gesichtsbewegungen häufiger asymmetrischer sind als spontane emotionale Ausdrücke [8].

## 2.1 Facial Action Units

Facial Action Units (AUs) sind die Gesichtsaktionseinheiten, die am häufigsten bei der Analyse des menschlichen Gesichtsausdrucks verwendet werden. Die Action Units definieren bestimmte Gesichtskonfigurationen, die durch die Kontraktion einer oder mehrerer Muskelgruppen entstehen. Die Facial Action Units sind unabhängig von der Interpretation von Emotionen [6]. Einige dieser Action Units sind in Tabelle 1 und Abbildung 1 gezeigt.

Action Units Obergesicht	
AU1	Heben der Augenbrauen innen
AU2	Heben der Augenbrauen außen
AU4	Zusammenziehen der Augenbrauen
AU5	Heben des oberen Augenlides
AU6	Zusammenziehen des äußeren Teils des Ringmuskels um die Augen
AU7	Zusammenziehen des inneren Teils des Ringmuskels um die Augen
Action Units Untergesicht	
AU 9	Rümpfen der Nase
AU 10	Anheben der oberen Lippe
AU 12	Anheben der Mundwinkel
AU 14	Einziehen der Mundwinkel
AU 15	Herabziehen der Mundwinkel
AU 16	Nach unten Ziehen der Unterlippe
AU 17	Hinaufschieben des Kinns
AU 20	Gestreckte Lippen
AU 23	Spannen der Lippen, nach innen gerollt
AU 25	Öffnen der Lippen
AU 26	Öffnen des Mundes durch Entspannung der Unterkiefermuskulatur
AU 28	Lippen saugen
AU 45	Blinzeln (mit beiden Augen)

Tabelle 1. Facial Action Units (FACS) [6]

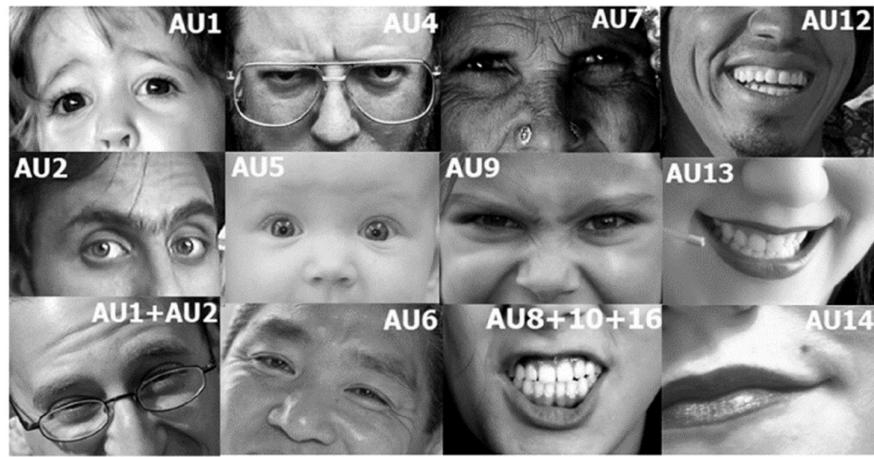


Abbildung 1. Facial Action Units. Quelle: <https://github.com/TadasBaltrusaitis/OpenFace/wiki/Action-Units> [9]

### 3 OpenFace

OpenFace wurde zur automatischen Analyse des Gesichtsverhaltens entwickelt und ist das erste Open-Source-Tool, das die Erkennung von Gesichtsmerkmalen, der Kopfhaltung, der FACS und der Blickrichtung integriert [10]. Dieses Toolkit wurde ursprünglich von Tadas Baltrusaitis in Zusammenarbeit mit dem CMU MultiComp Lab unter der Leitung von Prof. Louis-Philippe Morency entwickelt. Einige der ursprünglichen Algorithmen wurden an der Rainbow Group der Universität Cambridge erstellt. Die OpenFace-Bibliothek wird im CMU MultiComp Lab in Zusammenarbeit mit Tadas Baltrusaitis noch aktiv entwickelt [9] .

Das moderne OpenFace Framework, implementiert moderne Algorithmen zur Analyse des Gesichtsverhaltens via Maschine Learning. Das System läuft in Echtzeit, wobei alle Gesichtsverhaltensmodule zusammenarbeiten [10] (siehe Abbildung 2). Außerdem wird der Code in den Programmiersprachen C++, Matlab und Python zu Verfügung gestellt [10].

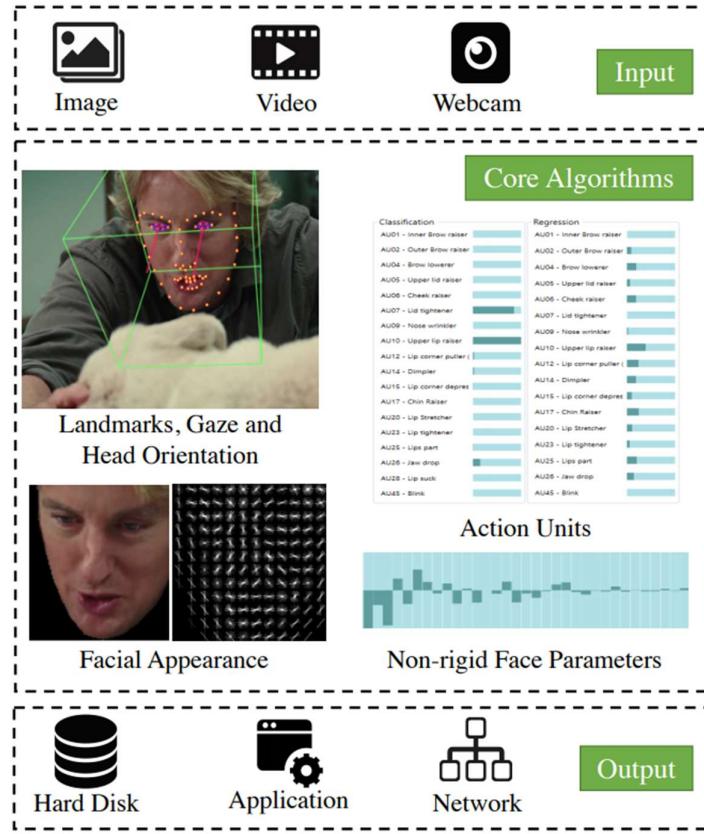


Abbildung 2. OpenFace Framework, Darstellung der Implementierung moderner Algorithmen zu Analyse des Gesichtsverhaltens. Quelle: [https://github.com/TadasBaltrusaitis/OpenFace/wiki/Action-Units \[9\]](https://github.com/TadasBaltrusaitis/OpenFace/wiki/Action-Units)

Das Ausgabeformat für die Tracking-Daten ist sowohl für einzelne Bilder und als auch für Bildsequenzen das comma-separated-values (CSV) Dateiformat. Tabelle 2. zeigt die ersten fünf Daten und ihre Bedeutung (in den jeweiligen Spalten) aus der von OpenFace erstellten CSV-Datei. Dies sind die Basisdaten für jede Datenanalyse. Die Spaltenüberschrift für die Tracking-Daten von *Head Pose*, *Landmarks*, *Eye Gaze*, *Action Units* werden einzeln in den nächsten Subkapiteln behandelt.

CSV-Datei von OpenFace					
Spalte	1	2	3	4	5
Header	frame	face_id	timestamp	confidence	succes
Inhalt	Nummer des Frames (bei Sequenzen)	Face ID (Für Sequenzen mit mehreren Gesichtern)	Der Timer des Videos, in Sek (bei Sequenzen).	Konfidenzniveau der Gesichtserkennung (in Prozentsatz)	Ist der Track erfolgreich (gibt wie gut ein Gesicht verfolgt wurde)

Tabelle 2. Die fünf ersten Spalten der CSV-Datei generiert von OpenFace

### 3.1 Gesichtsausdruckserkennung

Wie in Kapitel 2 beschrieben, werden Gesichtsausdrücke mit Facial Action Units (AUs) dargestellt, welche die Aktivierung von Gesichtsmuskeln objektiv beschreiben. OpenFace erkennt 18 AUs via Maschine Learning. Die AUs können auf zwei Arten beschrieben werden: Präsenz und Intensität. Die Präsenz ist gegeben, wenn eine AU erkannt wurde. Die Intensität wird auf einer Skala von 1 bis 5 angegeben, wobei 1 die minimale und 5 die maximale Intensität ist. OpenFace liefert beide Daten mit Ausnahme von AU 28, für welche nur die Präsenz vom System registriert wird (siehe Tabelle 3 und Abbildung 3)

Action Units Obergesicht	
AU1	Heben der Augenbrauen innen
AU2	Heben der Augenbrauen außen
AU4	Zusammenziehen der Augenbrauen
AU5	Heben des oberen Augenlides
AU6	Zusammenziehen des äußeren Teils des Ringmuskels um die Augen
AU7	Zusammenziehen des inneren Teils des Ringmuskels um die Augen
Action Units Untergesicht	
AU 9	Rümpfen der Nase
AU 10	Anheben der oberen Lippe
AU 12	Anheben der Mundwinkel
AU 14	Einziehen der Mundwinkel
AU 15	Herabziehen der Mundwinkel
AU 16	Nach unten Ziehen der Unterlippe
AU 17	Hinaufschieben des Kinns
AU 20	Gestreckte Lippen
AU 23	Spannen der Lippen, nach innen gerollt
AU 25	Öffnen der Lippen
AU 26	Öffnen des Mundes durch Entspannung der Unterkiefermuskulatur
AU 28*	Lippen saugen
AU 45	Blinzeln (mit beiden Augen)

Tabelle 3. Facial Action Units, die mit OpenFace erkannt werden können

\* OpenFace liefert nur Daten über den Präsenz von UA 28 und nicht über deren Intensität.

Lower Face Action Units					
AU9	AU10	AU11	AU12	AU13	AU14
Nose Wrinkler	Upper Lip Raiser	Nasolabial Deepener	Lip Corner Puller	Cheek Puffer	Dimpler
Lip Corner Depressor	Lower Lip Depressor	Chin Raiser	Lip Puckerer	Lip Stretcher	Lip Funneler
Lip Tightener	Lip Pressor	Lips Parts	Jaw Drop	Mouth Stretch	Lip Suck

Abbildung 3. Einige im Bild dargestellte Action Units, die mit OpenFace erkannt werden können.

Quelle: "Facial Expression Analysis" by Fernando De la Torre and Jeffrey F. Cohn [11]

## 3.2 Erkennung von Gesichtsmerkmalen

Das Modell zur Erkennung von Gesichtsmerkmalen von OpenFace wurde auf unterschiedliche Gesichtsorientierungen trainiert. Das ermöglicht die Erkennung und Verfolgung der Gesichter trotz Kopfbewegungen und Kopfdrehungen [12].

Abbildung 4 zeigt, dass OpenFace die Gesichtsmerkmale erkennen kann, unabhängig von der Ausrichtung des Gesichts.



Abbildung 4. Beispiel für die Erkennung von Gesichts-Landmarken in verschiedenen Gesichtsorientierungen mit OpenFace. Quelle: Constrained Local Neural Fields for robust facial landmark detection in the wild [12]

Der von OpenFace verwendete Algorithmus erstellt ein einfaches lineares Mapping vom Begrenzungsrahmen des Gesichts. Dieses lineare Mapping umfasst 68 Gesichtsmerkmale. Abbildung 5 zeigt die Lage der 68 Gesichtsmerkmale, die OpenFace erkennen kann [12].

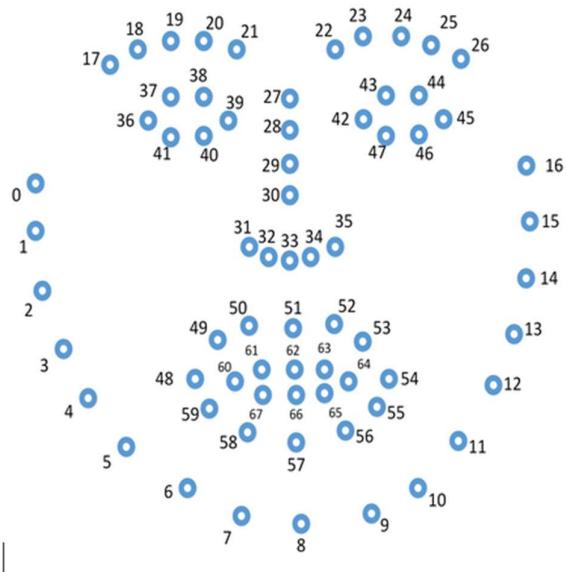


Abbildung 5. 68 Facial Landmarks. Quelle: Constrained Local Neural Fields for robust facial landmark detection in the wild [12]

Zur Verarbeitung der Daten gibt OpenFace die Daten von der Lage der Gesichtsmerkmale sowohl in 3D als auch in 2D aus. Die Lage der 2D Gesichtsmerkmale ist in Pixel gegeben und die Lage der 3D Gesichtsmerkmale ist in Millimeter gegeben. Damit die Lage der Gesichtsmerkmale in 3D genau wird, sollten zusätzlich Kopfhaltung und Blick bewerten werden. Tabelle 4 zeigt die Spaltenüberschriften, unter denen die Gesichtsmerkmale -Daten in der CSV-Datei zu finden sind.

Gesichtsmerkmale 2D		
Daten	Spaltenüberschrift	Einheit
X Koordinaten	x_0, x_1 ... x_67	Pixel
Y Koordinaten	y_0, y_1 ... y_67	Pixel

Gesichtsmerkmale 3D		
Daten	Spaltenüberschrift	Einheit
X Koordinaten	X_0, X_1 ... X_67	Millimeter
Y Koordinaten	Y_0, Y_1 ... Y_67	Millimeter
Z Koordinaten	Z_0, Z_1 ... Z_67	Millimeter

Tabelle 4. Gesichtsmerkmale in der CSV-Datei

### 3.3 Schätzung der Kopfhaltung

Das OpenFace Modell zur Schätzung der Kopfhaltung ist in der Lage sowohl Kopf-Translationsbewegungen als auch Kopf-Rotationsbewegungen zu extrahieren. Es ist auch möglich eine genaue Schätzung der Kopfhaltung zu erhalten trotz der Unterschiede der Kamerakalibrierung [10]. In Abbildung 6 kann man den 3D-Begrenzungsrahmen um den Kopf sehen, welcher die Schätzung der Kopfhaltung von OpenFace veranschaulicht.

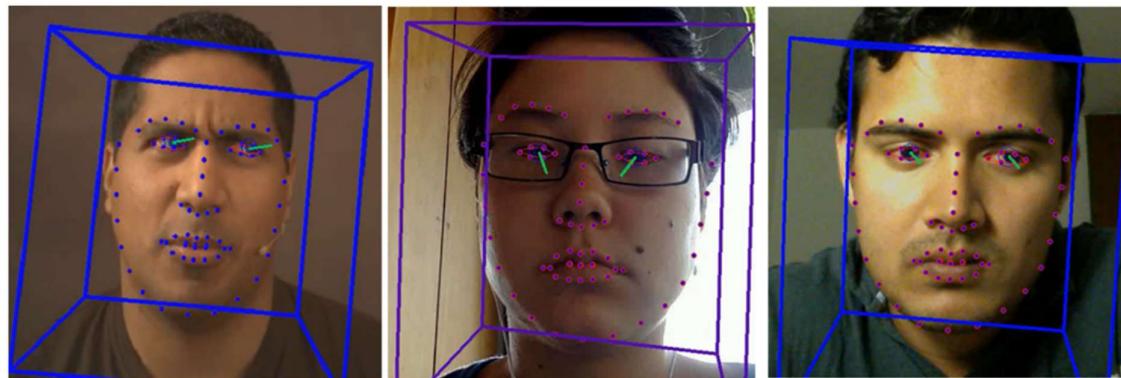


Abbildung 6. Schätzung der Kopfhaltung von OpenFace. Quelle: OpenFace 2.0: Facial Behavior Analysis Toolkit [10].

Die Daten der Kopfhaltung in OpenFace für die Position des Kopfes in Bezug auf die Kamera sind in Millimetern angegeben. Positives Z bedeutet die Entfernung des Kopfs von der Kamera. Die Drehung wird in Radiant angegeben um die X-, Y- und Z-Achse mit der Konvention  $R = Rx * Ry * Rz$ , linkshändiges positives Vorzeichen. Dies kann als Pitch ( $Rx$ ), Yaw ( $Ry$ ) und Roll ( $Rz$ ) angesehen werden [10]. Die Drehung erfolgt in Weltkoordinaten, wobei die Kamera der Ursprung ist [10]. Abbildung 5 zeigt die Kopfhaltung in Bezug auf die Kamera: Roll-Nick-Gier-Winkel, englisch roll-pitch-yaw angle, welche die drei Freiheitsgrade eines menschlichen Kopfes beschreiben [11]. Die Spaltenübersicht, wo die Daten der Kopfhaltung in der CSV-Datei zu finden sind, ist in Tabelle 5 gezeigt.

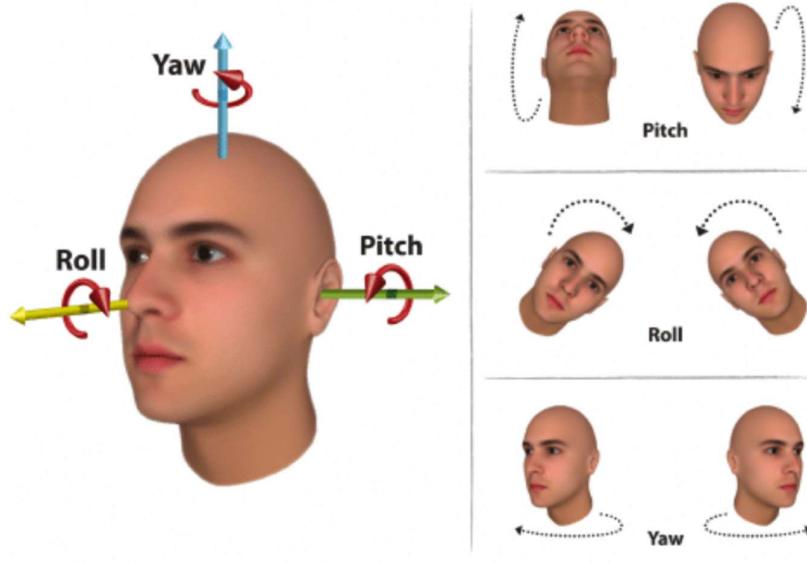


Abbildung 7. Ausrichtung des Kopfes in Bezug auf Nick-, Roll- und Gierbewegungen (Pitch, Roll, Yaw). Quelle: Reference Real-Time Head Pose Estimation for Mobile Devices [11]

Kopf-Position in Bezug der Kamera		
Daten	Spaltenüberschrift	Einheit
Lage in X	pose_Tx,	Millimeter
Lage in Y	pose_Ty,	Millimeter
Lage in Z	pose_Tz	Millimeter
Drehung des Kopfes		
Daten	Spaltenüberschrift	Einheit
Nick-Winkel (pitch)	pose_Rx	Radian
Gier-Winkel (yaw)	pose_Ry	Radian
Roll-Winkel (roll)	pose_Rz	Radian

Tabelle 5. Daten der Kopfposition in der CSV-Datei

### 3.4 Schätzung der Blickrichtung

Die Schätzung der Blickrichtung ist durch die Position der Augenlider, der Iris und der Pupille gegeben. Zur Berechnung des Blickvektors wird die Lage der Pupille individuell für jedes Auge ermittelt. Zur Schätzung der Blickrichtung wendete OpenFace eine große Anzahl von

fotorealistischen Bildern von Augen als Trainingsdaten an. Auf diese Weise kann OpenFace die Blinkrichtung eines jeden Mensch unabhängig von der Augenform einschätzen [13] (siehe Abbildung 8).

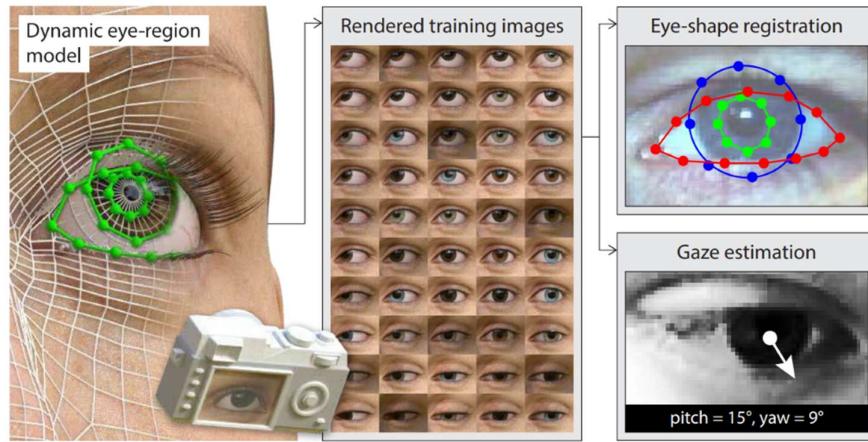


Abbildung 8. Schätzung der Blickrichtung von OpenFace mittels Maschine Learning.

Quelle: Rendering of Eyes for Eye-Shape Registration and Gaze Estimation [13]

OpenFace erkennt 55 Augenregionen (siehe Abbildung 10) sowohl in 2D als auch in 3D. Die Schätzung der Blickrichtung ist als Vektor und als Winkel angegeben. Die Position der Augenregionen in 2D sind in Pixel angegeben, in 3D in Millimetern [13].

Tabelle 6 zeigt die Spaltenüberschrift für die Schätzung der Blickrichtung und die Koordinaten der Augenregionen auf dem CSV-Datei.

Blickrichtungs-Vektor		
Daten	Spaltenüberschrift	Einheit
Linkes Auge – Blickrichtung in X	gaze_0_x	-
Linkes Auge – Blickrichtung in Y	gaze_0_y	-
Linkes Auge – Blickrichtung in Z	gaze_0_z	-
Rechtes Auge – Blickrichtung in X	gaze_1_x	-
Rechtes Auge – Blickrichtung in Y	gaze_1_y	-
Rechtes Auge – Blickrichtung in Z	gaze_1_z	-
Blickrichtungs-Winkel		
Daten	Spaltenüberschrift	Einheit
Blickrichtungs-Winkel in X	gaze_angle_x	Radian
Blickrichtungs-Winkel in Y	gaze_angle_y	Radian

Merkmale der Augenregion		
Daten	Spaltenüberschrift	Einheit
Augen- Merkmale 2D in X	eye_lmk_x_0 ... eye_lmk_x_55	Millimeter
Augen- Merkmale 2D in Y	eye_lmk_y_0 ... eye_lmk_y_55	Millimeter
Augen- Merkmale 3D in X	eye_lmk_X_0 ... eye_lmk_X_55	Millimeter
Augen- Merkmale 3D in Y	eye_lmk_Y_0 ... eye_lmk_Y_55	Millimeter
Augen- Merkmale 3D in Z	eye_lmk_Z_0 ... eye_lmk_Z_55	Millimeter

Tabelle 6. Auslesen von Blickrichtung und Augenmerkmalen

Die Daten Gaze\_0\_x, Gaze \_0\_y und Gaze \_0\_z definieren den Blickrichtungsvektor des Auges in Weltkoordinaten für Auge 0. Die Auge 0 ist das linke Auge im Bild. Die Daten Gaze\_1\_x, Gaze\_1\_y und Gaze\_1\_z bilden den Blickrichtungsvektor des rechten Auges [13].

Die Daten gaze\_angle\_x und gaze\_angle\_y geben die durchschnittliche Blickrichtung des Auges in Radian in Weltkoordinaten für beide Augen an. Wenn eine Person die Augen von links nach rechts bewegt, führt dies zu einer Änderung von gaze\_angle\_x (von positiv zu negativ). Wenn eine Person die Blickrichtung von oben nach unten ändert, führt dies zu einer Änderung von gaze\_angle\_y (von negativ zu positiv). Wenn der Winkel nahe bei 0 liegt bedeutet das, dass eine Person nach vorne schaut. Abbildung 9 zeigt die Darstellung der von OpenFace geschätzten Blickrichtung. [13]

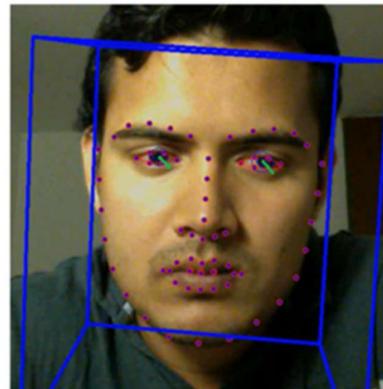


Abbildung 9. Darstellung der von OpenFace geschätzten Blickrichtung (Pfeile in grün). Quelle: [9]

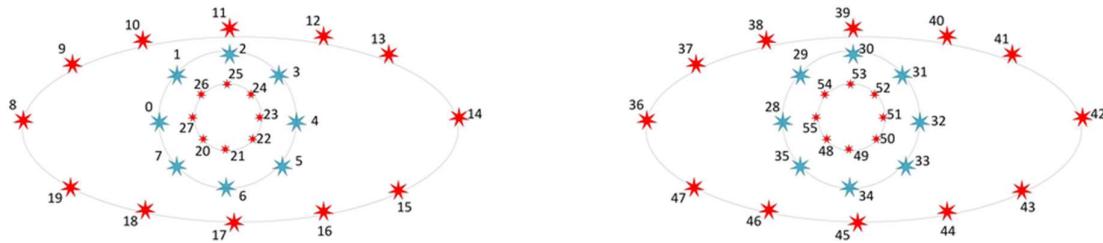


Abbildung 10. Position der Merkmale im Augenbereich. Quelle: [9]

### 3.5 OpenFaceOffline

Die OpenFaceOffline Anwendung ist das Graphical User Interface (GUI) für Windows von OpenFace. Die Applikation bietet die Möglichkeit, einzelne Bilder, Bildsequenzen, Videoaufnahmen oder Echtzeit-Videos zu analysieren (siehe Abbildung 11).

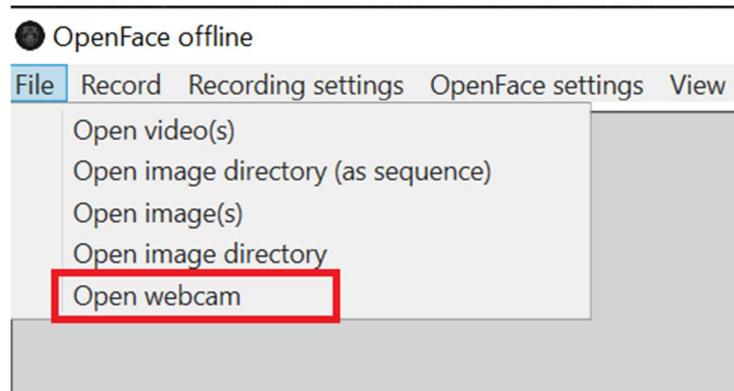


Abbildung 11. Optionen zur Analyse von Bildern oder Videoaufnahmen. Die Auswahl zur Analyse von Echtzeit-Videos ist in rot markiert

Mit OpenFaceOffline ist es möglich auszuwählen, welche Eigenschaften analysiert werden sollen:

- Kopfhaltung
- Blickrichtung
- Gesichtsmerkmale oder AUs einzeln
- mehrere Eigenschaften zugleich
- alle Eigenschaften zugleich

Die Anwendung bietet auch die Option die Auflösung der verwendeten Webcam auszuwählen. Die Analyse der Daten kann mit diesem Interface grafisch dargestellt werden (siehe Abbildung 12 und Abbildung 13).

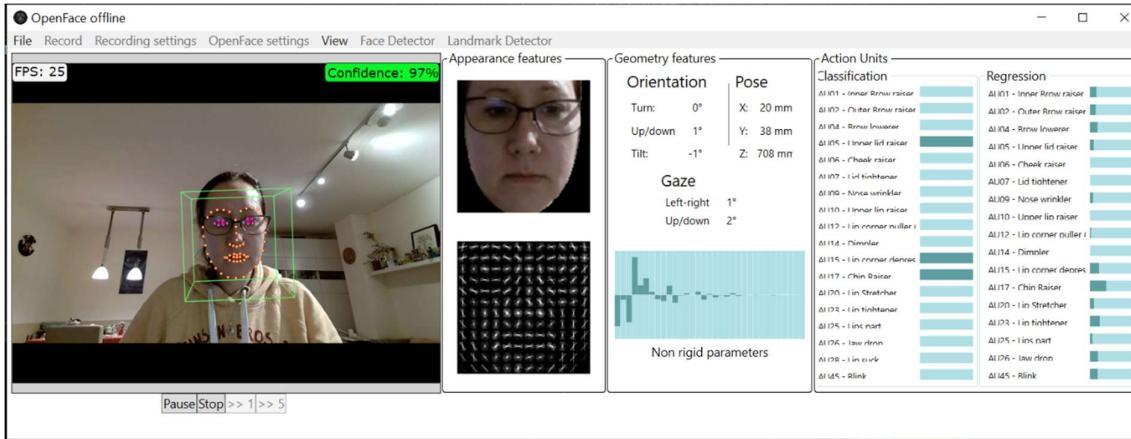


Abbildung 12. Graphische Darstellung der Videoanalyse von Kopfhaltung, Blickrichtung, Gesichtsmerkmalen und AUs

In diesem Projekt wird die von Martin Deinhofer modifizierte Version von OpenFace verwendet, welche die Daten via Socket streamt [4]. Diese Version bietet die Einstellmöglichkeit, ob die CSV-Datei gespeichert wird oder die Daten via Socket gesendet werden oder beides (siehe Abbildung 13).

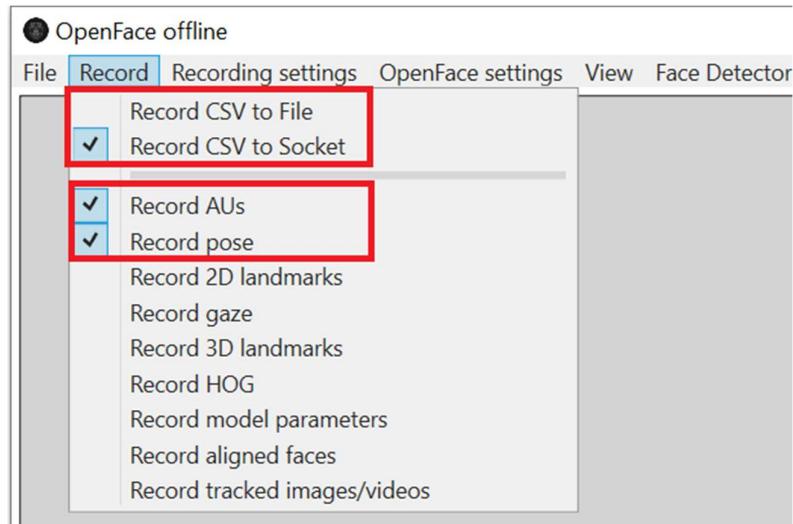


Abbildung 13. Option für das Speichern oder Streamen der Tracking-Daten und Auswahl der Eigenschaften zur Analyse

## 4 AsTeRICS Framework

Das AsTeRICS Framework ist Teil des AsTeRICS-Projekts. Dieses wurde teilweise von der Europäischen Kommission gegründet und verschiedene Institutionen wie die Fachhochschule Technikum-Wien haben an diesem Projekt teilgenommen. Die Abteilung für Embedded Systems der Fachhochschule Technikum-Wien war für das technische Management des Projekts verantwortlich. Ziel dieses Projektes ist es, verschiedene Assistive Technologien zu entwickeln, welche Menschen mit Beeinträchtigung helfen, sich in das soziale und Arbeits-Leben zu integrieren. [5]

Das AsTeRICS Framework ist ein Open Source, um unterstützende Lösungen zu geringen Kosten zu erstellen. Der Begriff AsTeRICS steht für Assistive Technology Rapid Integration and Construction Set. [5]

Die im AsTeRICS-System enthaltenen Tools sind: der grafischer Editor ACS (AsTeRICS Configuration Suite) für den Entwurf des Modells (siehe Abbildung 14), das Laufzeitsystem, ARE (AsTeRICS Runtime Environment), welches alle AsTeRICS Modelle ausführt, ein Erstellungsassistent für neue Komponenten (das AsTeRICS PluginCreationWizard-Tool), und ein Erstellungsassistent für die Erweiterung des AsTeRICS Grids. Das AsTeRICS-System beinhaltet auch Benutzer- und Entwicklerhandbücher, sowie eine Reihe von Demo-Vorlagen für mögliche Anwendungen. [5]

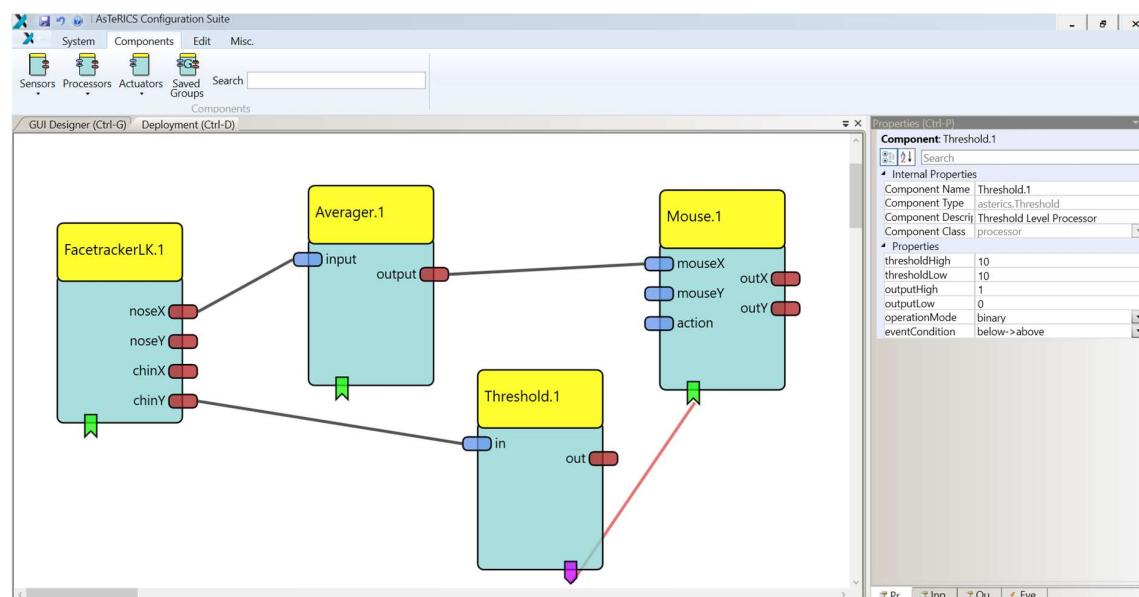


Abbildung 14. Beispiel für die Erstellung von Eines Modells mit der AsTeRICS Configuration Suite.

## 5 Integration von OpenFace in AsTeRICS

Die Integration von OpenFace in AsTeRICS erfolgt mittels Datenübertragung über Sockets, dafür sind zwei Prozesse notwendig:

1. Die Verwendung der modifizierten Version von OpenFace von Martin Deinhofer, welche die Tracking-Daten via Socket streamt [4].
2. Die Erstellung eines AsTeRICS ClientSocket-Plugins, welches sich mit dem Socket verbindet und die Daten von OpenFace empfängt (siehe Abbildung 15).

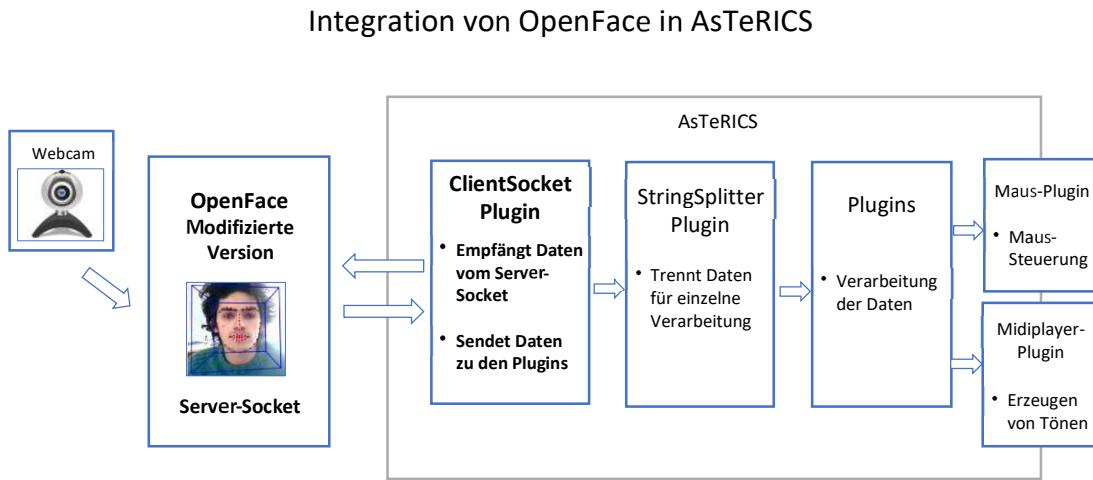


Abbildung 15. Integration von OpenFace in AsTeRICS

Für die Datenübertragung verwendet der Server-Socket von OpenFace das Kommunikationsprotokoll TCP (Transfer Control Protocol). Der Socket ist am lokalen Port 11000 angebunden. Um die Datenübertragung über den Socket zu starten, muss diese Option in der OpenFaceOffline Anwendung ausgewählt werden, indem man im Menü *Record*, die Option *Record CSV to Socket* auswählt (siehe Abbildung 16).

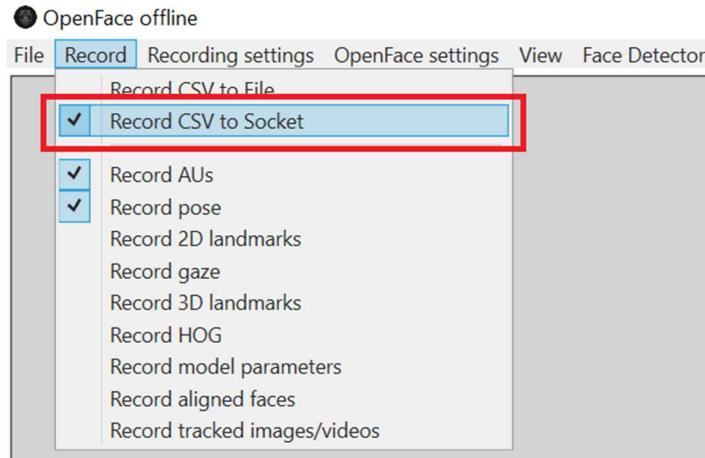


Abbildung 16. Option Send Data zum Server-Socket in OpenFace

Das ClientSocket-Plugin sendet die Daten zur StringSplitter Komponente. Mit dem StringSplitter werden Daten ausgewählt, welche spezifische Facial Action Units, Gesichtsmerkmale, Blickrichtungs- und Kopfhaltungsinformationen enthalten und zur Maus-Komponente gesendet, um die Computermaus zu steuern oder zur Midiplayer-Komponente, um Töne zu erzeugen.

## 5.1 Erstellung des Client Socket-Plugins

Das ClientSocket-Plugin wird mithilfe des AsTeRICS PluginCreationWizard-Tools erstellt. Die Funktion des Client-Sockets ist das Empfangen von Daten vom Server-Socket und diese an andere Komponenten zu senden. Das ClientSocket-Plugin ist so konfiguriert, dass es universell eingesetzt werden kann. Das bedeutet, dass das Plugin mit jedem anderen Server über TCP oder UDP kommunizieren kann. Dafür wurden die Portnummer, das Protokoll und die IP-Adresse in den Properties konfiguriert (diese können geändert werden). Da der Client und der Server auf dem gleichen Computer laufen, wird der Localhost (127.0.0.1) als Default IP-Adresse definiert. Die Default Portnummer ist 1111 und das Default Protocol ist TCP.

Die Kommunikation des ClientSocket-Plugins mit anderen Komponenten kann durch Eingangs- und Ausgangskanäle (Inputs und Outputs) oder durch Events (Trigger und Listener Events) erfolgen. Für dieses Projekt wurden die Events zwar deklariert aber nicht implementiert. Durch den Output werden die empfangenen Daten zu anderen Komponenten gesendet. Der Datentyp für den Input und den Output kann ein String oder binär sein. Outputs wurden in diesem Projekt jedoch nicht implementiert.

Tabelle 7 zeigt die Konfiguration zur Erstellung der ClientSocket-Komponente mit dem AsTeRICS PluginCreationWizard-Tool und in Abbildung 17 kann man das fertige ClientSocket-Plugin in der AsTeRICS Configuration Suite (ACS) sehen.

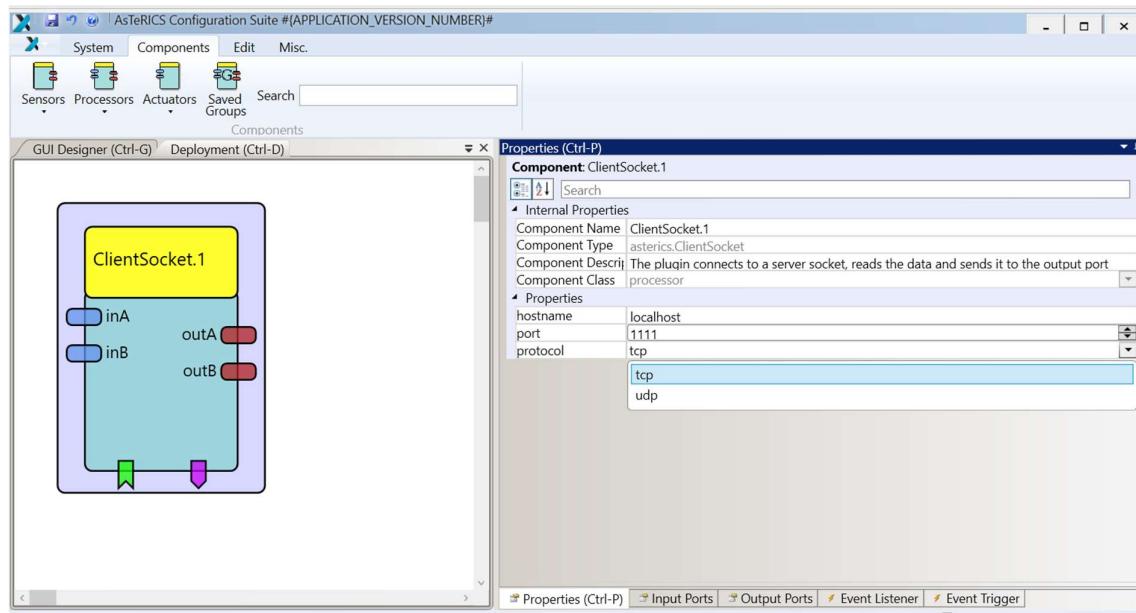


Abbildung 17. ClientSocket-Plugin

Client Socket-Plugin Konfiguration		
Setup		
<b>PluginName</b>	ClientSocket	
<b>Type</b>	Processor	
<b>Subcategory</b>	Communication	
<b>Path</b>	...\\AsTeRICS\\ARE\\components\\	
<b>Plugin Description</b>	The plugin connects to a server socket, reads the data and sends it to the output port	
Input Ports		
<b>Input Name</b>	<b>Data Type</b>	<b>Description</b>
inA	string	Data to send to the socket
inB	integer	Binary data to send to the socket (as Byte stream)
Output Ports		
<b>Output Name</b>	<b>Data Type</b>	<b>Description</b>
outA	string	Data read from the socket
outB	integer	Binary data read from the socket (as byte stream)

Properties			
Property Name	Daten Type	Default Data	Description
hostname	string	Localhost	IP address or name of server program
port	integer	1111	Port of server program
protocol	integer	Tcp (UDP als zweite Option)	Protocol for communication
Event Listener Ports*			
Event Name	Description		
connect	Connect to server socket		
disconnect	Disconnect from server socket		
reconnect	Disconnect/Connect to server socket		
Event Trigger Ports*			
Event Name	Description		
Connected	The socket was connected		
disconnected	The socket was disconnected		

Tabelle 7. Konfiguration des ClientSocket-Pugins mit der AsTeRICS PluginCreationWizard-Tool

\*Die Events wurden in Rahmen dieses Projekt nicht implementiert

Im JAVA-Code, welcher automatisch vom PluginCreationWizard-Tool generiert wurde, ist die Funktion `connect ()` in der Klasse `ClientSocketInstance.java` implementiert. Hier wird die Kommunikation mit dem Server, das Empfangen und Senden von Daten durchgeführt.

Der ClientSocket wird in einem neuen Thread mit dem Server verbunden. Das Empfangen von Daten erfolgt mit dem BufferedReader „`reader`“. Mit der Funktion `sendData ()` sendet der ServerClient die empfangenen Daten zum Output OutA (im code als „`opOutA`“ deklariert). Das dient zur Datenübertragung zu den anderen Komponenten. Die Verbindung mit dem Server endet automatisch, wenn keine Daten mehr vom Socket empfangen werden. Der ClientServer auch Daten zum Server senden mithilfe des BufferedWriter „`writer`“.(siehe Abbildung 18)

```

public void connect() {
    //Connect to server socket in a new thread
    Thread socketThread=new Thread(new Runnable() {
        public void run() {
            try {
                //connect
                clientSocket = new Socket(propHostname, propPort);
                //read line from socket and send it to output port outA which is represented by the variable opOutA.
                writer = new BufferedWriter(new OutputStreamWriter(clientSocket.getOutputStream()));
                reader = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
                writer.flush();
                do {
                    String line = reader.readLine();
                    opOutA.sendData(ConversionUtils.stringToBytes(line));
                } while (reader.ready());
                writer.close();
                reader.close();
                clientSocket.close();
            } catch (Exception e) {
                //TODO: handle exception
                System.out.println(e);
            }
        }
    });
    socketThread.start();
}

```

Abbildung 18. Die Funktion *Connect ()* startet die Verbindung und die Kommunikation mit dem Server

Die Funktion *connect ()* wird aufgerufen, wenn auf der ARE die Option *start* wird, um die Verbindung mit dem Server zu starten (siehe Abbildung 19).

```

@Override
public void start()
{
    connect();
    super.start();
}

```

Abbildung 19. Funktion, die die Kommunikation zwischen AsTeRICS und den Server in ARE startet.

## 6 Erstellung neuer AsTeRICS-Modelle für die Maussteuerung durch Gesichtsmimik

Für die Erstellung der Modelle wurde die Desktop Version von AsTeRICS Configuration Suit verwendet. Die Basis Komponenten für jedes Modell, welche OpenFace in AsTeRICS integriert, sind der ClientSocket und der StringSplitter, welche über den Output- bzw. Input kommunizieren (siehe Abbildung 20).

Das ClientSocket-Plugin muss mit dem Port 11000 verbunden werden, da dieser der Port zur Verbindung mit dem OpenFace- Server-Socket ist. Da der Client und der Server auf demselben Computer laufen, wird die Default IP-Adresse Localhost verwendet. Die Property protocol muss TCP sein (siehe Abbildung 21).

Die Property startIndex des stringSplitter-Plugins muss abhängig vom Index der gesuchten Datenelemente in der CSV-Datei eingestellt werden. Diese soll man auf der von OpenFace gespeicherte CSV-Datei auslesen. Die Daten in der CSV-Datei sind mit Komma getrennt, daher sollte die Property Separator „ , “ sein (siehe Abbildung 22).

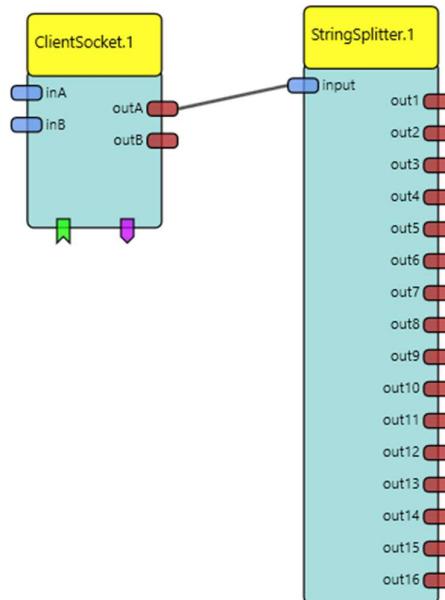


Abbildung 20. Verbindung zwischen dem ClientSocket-Plugin und dem StringSplitter-Plugin

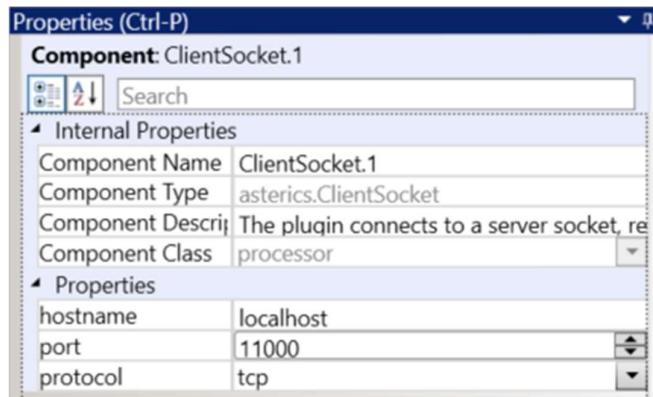


Abbildung 21. Einstellung des ClientSocket-Plugin

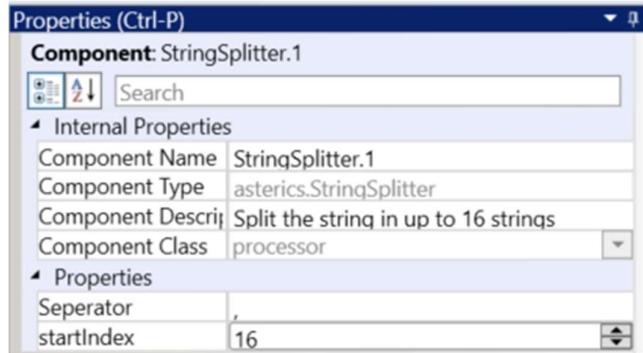


Abbildung 22. Einstellung des StringSplitter-Plugins

Es wurden drei AsTeRICS-Modelle für die Maussteuerung mittels Gesichtsmimik erstellt:

- Modell für die Maussteuerung durch Kopfbewegung und AUs
- Modell für die Maussteuerung durch die Verfolgung der Position der Nase und AUs
- Modell für die Maussteuerung durch die Blickrichtung

## 6.1 Maussteuerung durch Kopfbewegung und AUs

Mit diesem Modell wird die Maus mittels Kopfbewegungen gesteuert. Die Bewegung rechts-links der Maus erfolgt durch das Rollen (roll) des Kopfes. Die Bewegung oben-unten der Maus erfolgt durch das Nicken (pitch) des Kopfes. Das Klicken auf die linke Maustaste erfolgt durch das Spannen der Lippen (Mund nach innen rollen (AU23\_r)). Das Klicken auf die rechte Maustaste erfolgt durch Öffnen des Mundes (AU26\_r) und der Doppelklick erfolgt durch das Saugen der Lippen (AU28\_c) (siehe Abbildung 23).

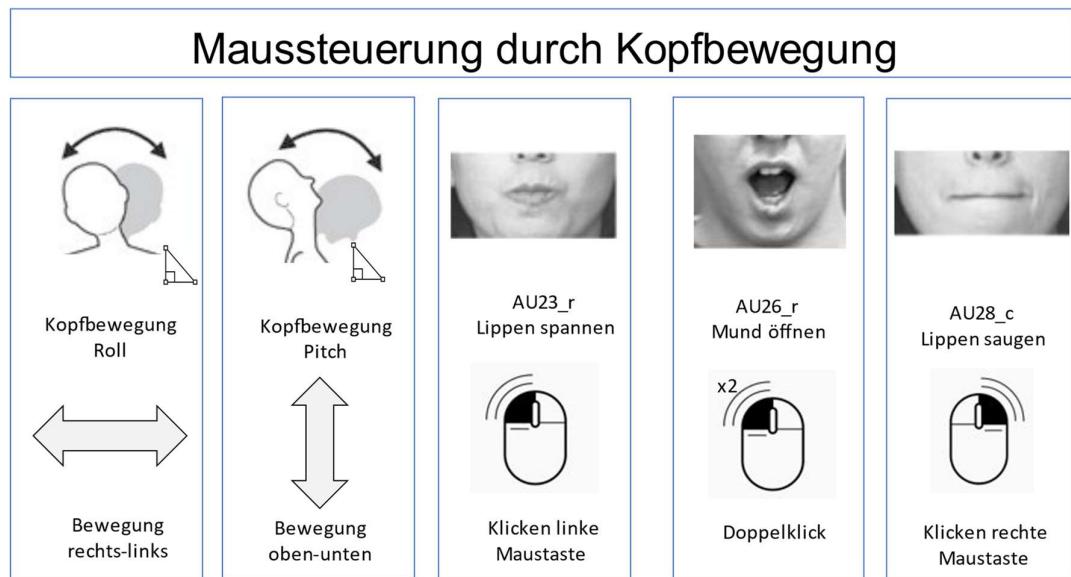


Abbildung 23. Funktionsweise der Maussteuerung durch Kopfbewegung und durch AUs

Die Daten über die Bewegung des Kopfs roll und pitch sind in Radiant gegeben. Um mit den Daten auf eine intuitivere und einfachere Weise zu arbeiten, wird der Winkelwert von Radiant in Grad umgewandelt (Abbildung 24). Eine Kopfbewegung von mehr als 10 Grad nach rechts löst eine Bewegung des Mauszeigers nach rechts aus. Bewegt sich der Kopf zwei Grad in die entgegengesetzte Richtung, bleibt der Mauszeiger stehen. Dieser Vorgang wird über das Threshold-Plugin, das CrosshairCursor-Plugin und das Mouse-Plugin realisiert (Abbildung 25). Das gleiche Verfahren gilt für die Bewegung der Maus nach links, oben oder unten (siehe Abbildung 26).

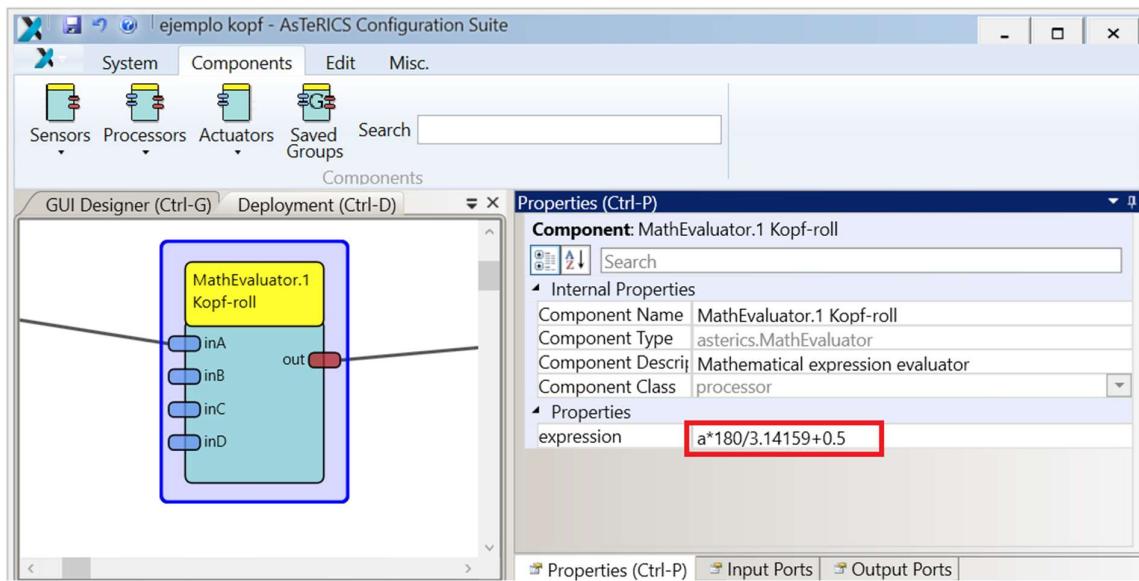


Abbildung 24. Umwandlung Radiant auf Grad mit dem MathEvaluator-Plugin

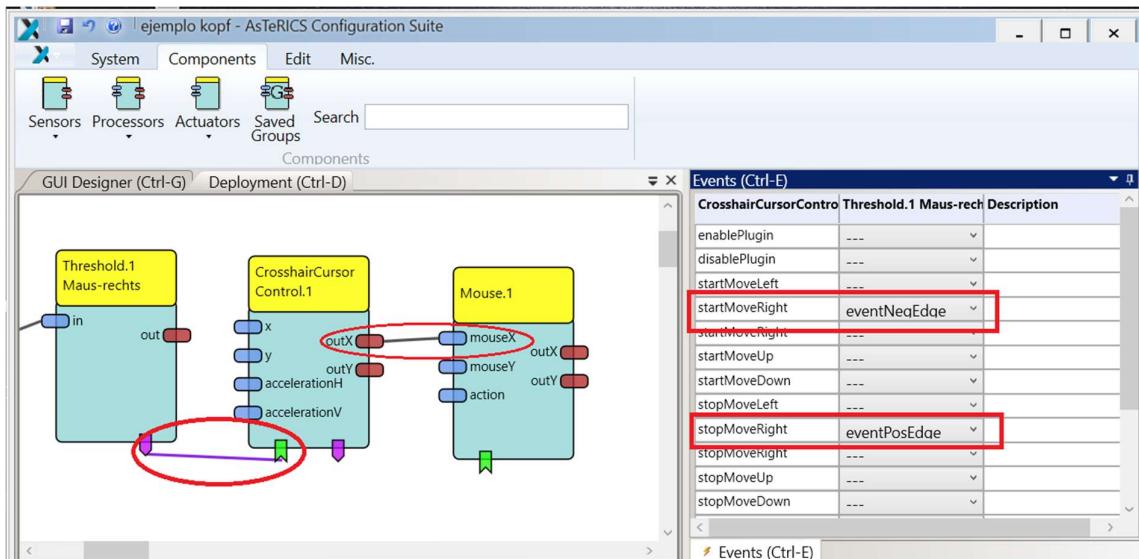


Abbildung 25. Verfahren für die Mausbewegung nach rechts.

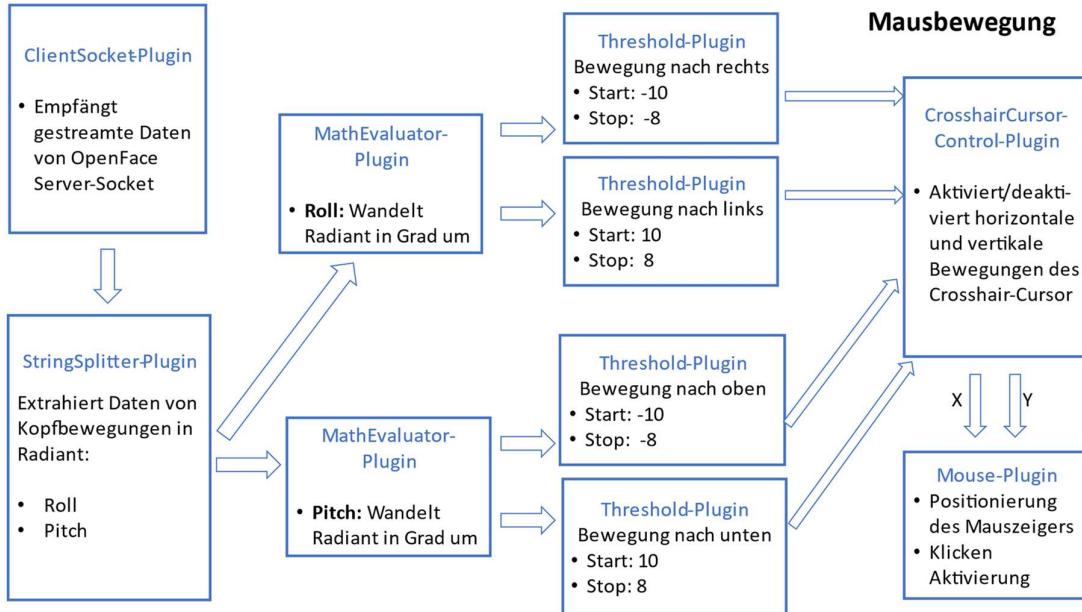


Abbildung 26. Schematische Darstellung des Modells für die Mausbewegung mittels Kopfbewegungen (roll und pitch)

Das Klicken auf die linke Maustaste wird aktiviert, wenn für Action Unit AU23\_r eine Intensität der Stufe 2 überschritten wird. Das Klicken auf die rechte Maustaste wird aktiviert, wenn für Action Unit AU26\_r eine Intensität der Stufe 3 überschritten wird. Durch das Threshold-Plugin wird detektiert, ob die festgelegte Intensität (Schaltschwelle) überschritten wurde. Das Verfahren für das Klicken auf die Maustasten ist in der Abbildung 28Abbildung 26 dargestellt.

Das Doppelklicken erfolgt mittels der Präsenz der Action Unit AU28\_c, wenn diese mindestens 600 Millisekunden andauert. Die Präsenz wird mit dem Threshold-Plugin bestimmt, wenn den Wert 1 überschritten ist. Zusätzlich zu dem Threshold-Plugin wird ein Timer-Plugin verwendet, welches die Dauern der AU28\_c misst, sobald diese vorhanden ist (siehe Abbildung 27).

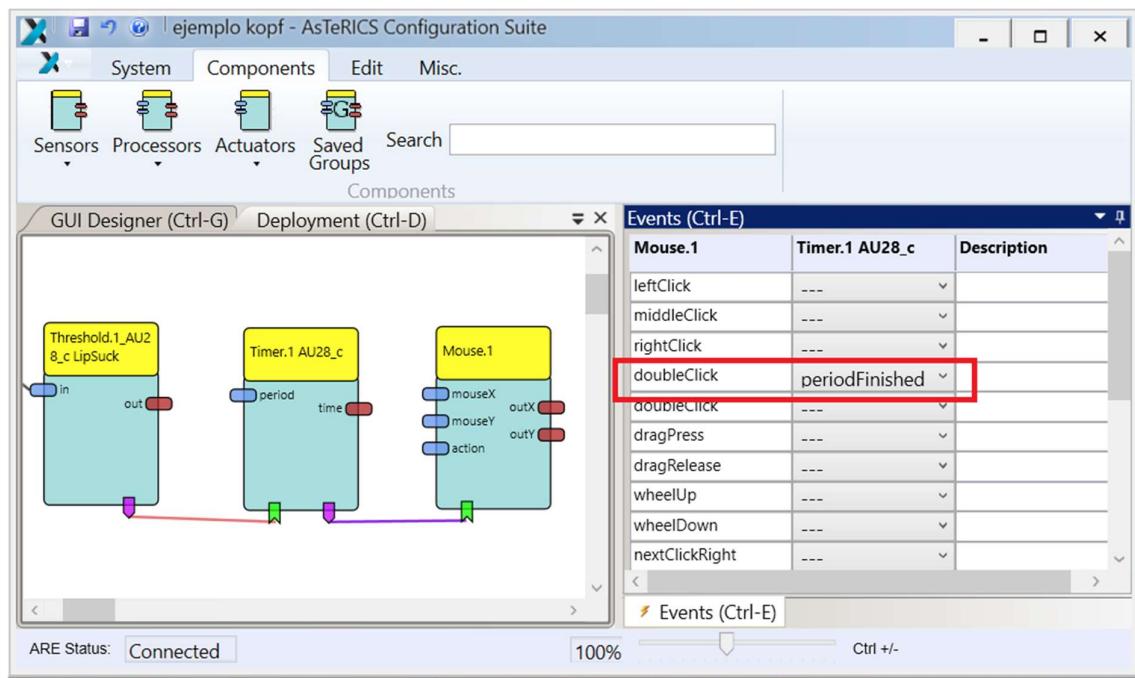


Abbildung 27. Auslösung des Doppelklicks durch Lippensaugen für 600 Millisekunden

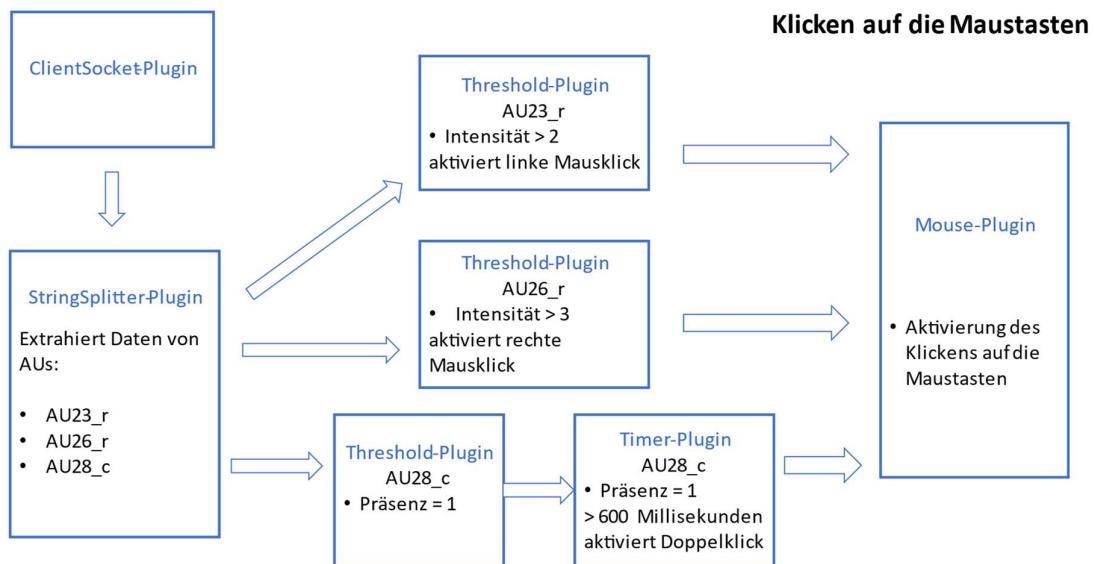


Abbildung 28. Schematische Darstellung des Modells für das Klicken auf die Maustasten

Die notwendigen Einstellungen in OpenFace für das Laufen dieses Modells sind:

- File -> Open webcam (Auflösung der Webcam auswählen)
- Record-> CSV to socket
- Record -> AUs
- Record -> pose

## 6.2 Maussteuerung durch Verfolgung der Position der Nase

Mit diesem Modell wird die Maus mittels der Verfolgung der Position der Nase gesteuert. Die vertikale und horizontale Bewegung der Maus erfolgt durch die Änderung der Position der Nase in Bezug auf den Bildschirm. Die Position der Nase ändert sich, wenn der Kopf sich nach rechts, links, oben oder unten bewegt. Das Klicken auf die Maustasten erfolgt durch die Erkennung einiger Action Units. Das Klicken auf die linke Maustaste erfolgt durch das Öffnen des Mundes (AU25\_c + AU26\_c). Beim Öffnen des Mundes für 2 Sekunden erfolgt ein Doppelklick. Das Klicken auf die rechte Maustaste erfolgt durch das Blinzeln der Augen für eine Sekunde (AU45\_c). Abbildung 29 zeigt die Funktionsweise des Modells.

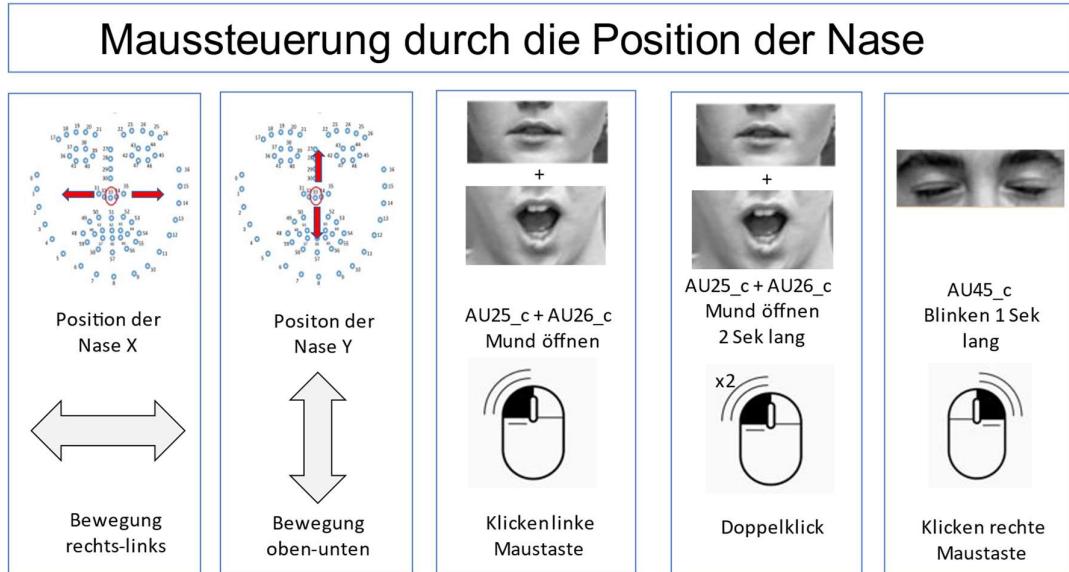


Abbildung 29. Maussteuerung durch Verfolgung der Position der Nase und durch AUs

Um die Position der Nase zu erkennen, müssen die Daten über die Gesichtsmerkmale in 2D extrahiert werden, welche in Pixel angegeben sind. Der Index für das Gesichtsmerkmal Nase in OpenFace ist 33. Mit dem StringSplitter-Plugin werden die Koordinaten X und Y in die Spalten `x_33` und `y_33` ausgegeben.

Das System bekommt die Position der Nase 32mal pro Sekunde (FPS der Kamera = 32). Das Modell berechnet den Durchschnitt der letzten 5 Werte. Danach wird die Differenz aus der aktuellen Durchschnittsposition ( $X_2$ ,  $Y_2$ ) und die Durchschnittsposition vor 10 Millisekunden ( $X_1$ ,  $Y_1$ ) berechnet. Diese Differenz, multipliziert mit dem Faktor 7, bestimmt den Abstand, die Richtung und die Geschwindigkeit der Bewegung des Mauszeigers. Um die Geschwindigkeit des Mauszeigers zu ändern, kann man den Faktor mit dem Slider-Plugin ändern. Ein weiterer Faktor -1 kehrt die Richtung der Mausbewegung um. Auf diese Weise wird ein Effekt der Richtungsumkehr korrigiert und somit bewegt sich die Maus korrekterweise nach rechts, wenn die Position der Nase nach rechts verschoben wird (siehe Abbildung 31).

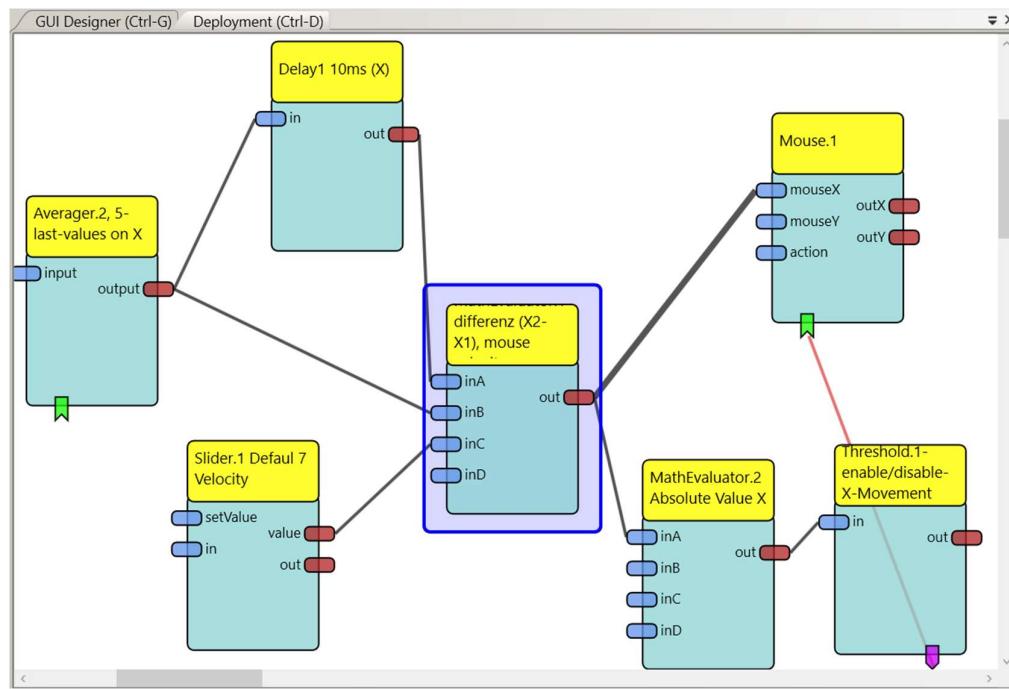


Abbildung 30. Konfiguration der Mausbewegung rechts-links durch Verfolgung der Nasenposition

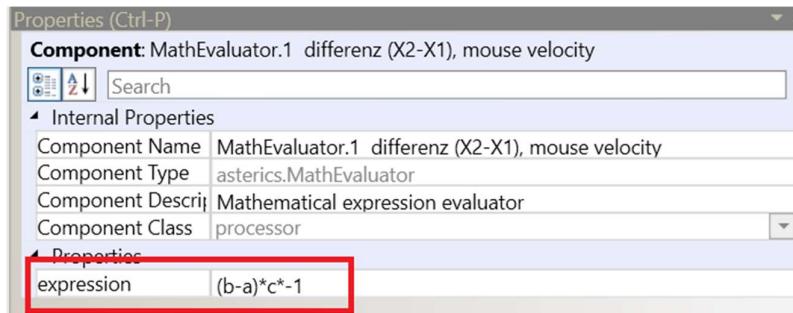


Abbildung 31. Berechnung der Mausbewegung mit Richtung und Geschwindigkeit

Um das Zittern der Maus zu vermeiden, wird zusätzlich zur Berechnung der durchschnittlichen Position ein Threshold-Plugin verwendet, welches die Mausbewegung aktiviert, wenn die Positions differenz mindestens 3 Pixel beträgt.

In Abbildung 32 ist die schematische Darstellung des Modells für die Mausbewegung rechts-links mittels Verfolgung der Position der Nase dargestellt. Das Verfahren für die Bewegung oben-unten funktioniert analog, nur dass in diesem Fall die Koordinate Y verwendet wird.

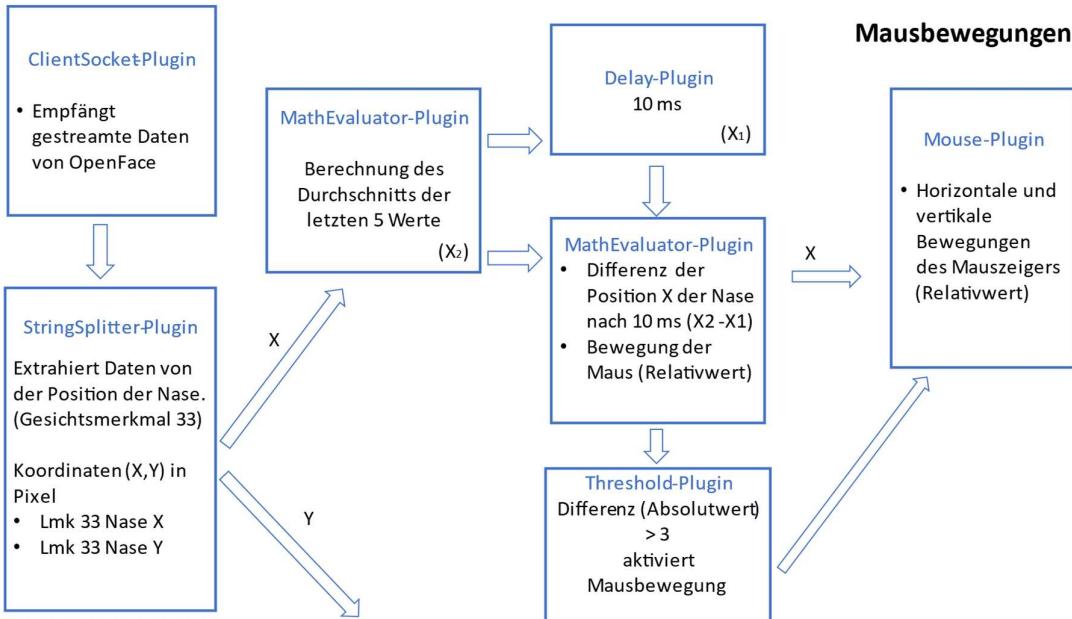


Abbildung 32. Schematische Darstellung des Modells für die Mausbewegung Rechts-links durch Verfolgung der Nasenposition

Das Verfahren zur Aktivierung von Klicks mit den Maustasten ist dasselbe wie im vorherigen Modell, nur dass für dieses Modell das alleinige Vorhandensein von AUs verwendet wird. Um

zu definieren, ob der Mund absichtlich geöffnet wurde, wird die Präsenz der AU25\_c und AU26\_c zusammen berechnet. Dies vermeidet das Klicken mit der Maus, wenn der Mund unabsichtlich leicht geöffnet wird.

Die notwendigen Einstellungen in OpenFace für das Laufen dieses Modells sind:

- File -> Open webcam (Auflösung der Webcam auswählen)
- Record-> CSV to socket
- Record -> AUs
- Record -> 2D landmarks

## 6.3 Maussteuerung durch Erkennung der Blickrichtung

Mit diesem Modell wird die Maus mittels Blickrichtung gesteuert. Mithilfe dieses Modells kann der Mauszeiger nur horizontal bewegt werden. Die Mausbewegung ist durch die Blickrichtung rechts-links gegeben. Das Klicken auf die linke Maustaste erfolgt durch das Heben der Augenbrauen (siehe Abbildung 33).

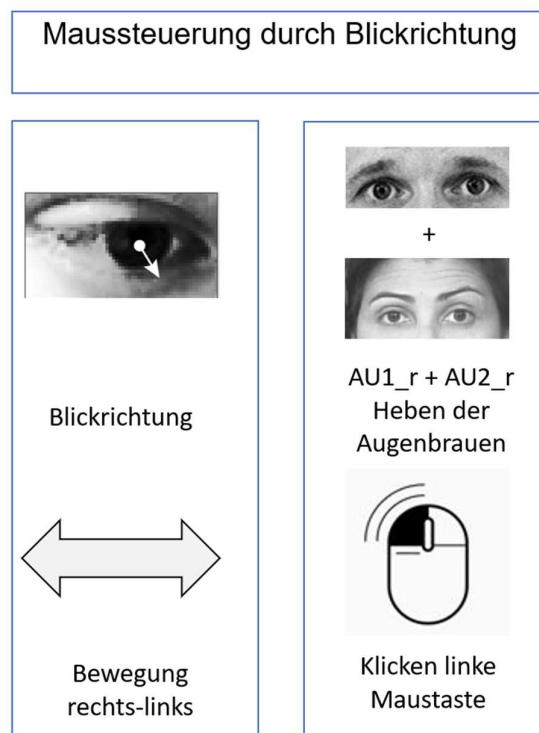


Abbildung 33. Funktionsweise der Maussteuerung mittels Blickrichtung

Die Blickrichtung wird von OpenFace in Radiant angegeben. Um intuitiver mit den Daten arbeiten zu können, wandelt das Modell die Einheit Radiant in Grad um. Danach wird der Winkel mit dem Skalierungsfaktor 1000 multipliziert. Diese Vorgehensweise erlaubt, jede minimale Änderung des Blickwinkels wahrzunehmen (siehe Abbildung 35). Ein Winkel größer als -0,001 Grad aktiviert die Bewegung nach rechts. Ein Winkel größer als 0,001 Grad aktiviert die Bewegung nach links. Bei Änderung der Blickrichtung um 0,0002 Grad in die entgegengesetzte Richtung wird die Bewegung angehalten (siehe Abbildung 34 und Abbildung 36).

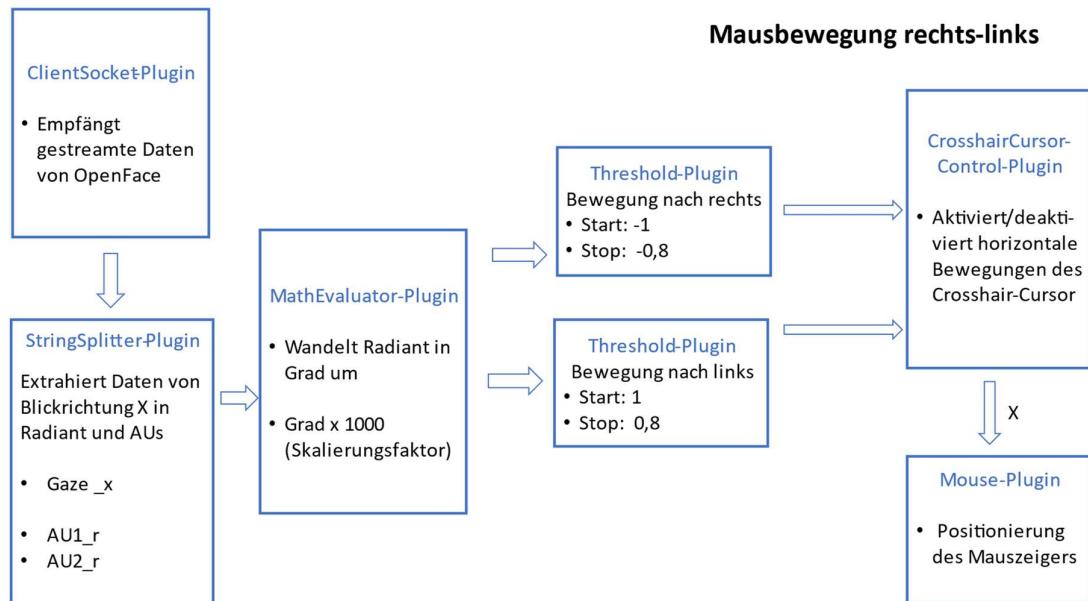


Abbildung 34. Schematische Darstellung des Modells für die horizontale Mausbewegung

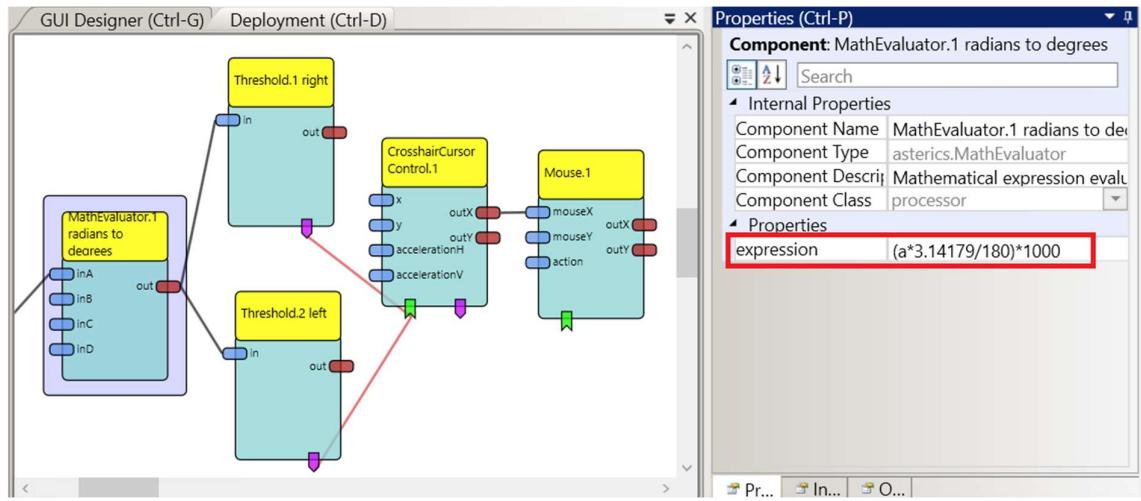


Abbildung 35. Umwandlung Radiant auf Grad und Skalierungsfaktor

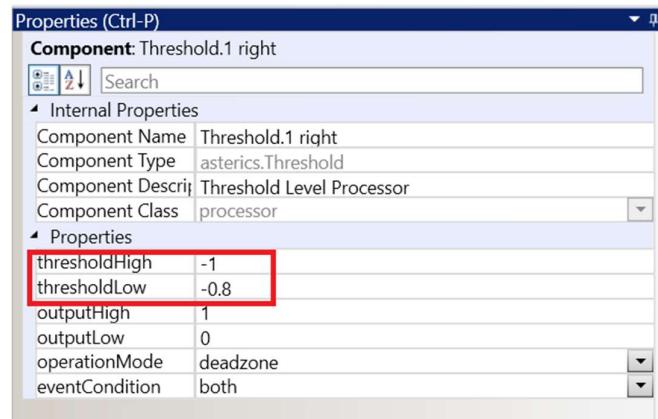


Abbildung 36. Aktivierung und Deaktivierung der Mausbewegung mit dem Threshold-Plugin

Dieses Modell ist für die Verwendung in Verbindung mit einem AsTeRICS-Grid vorgesehen und ist für Menschen konzipiert, die weder Hand- noch Kopfbewegungsfunktion besitzen. Das Projekt schlägt ein AsTeRICS-Grid für eine einfache Kommunikation mittels Maussteuerung durch Blickrichtung rechts-links vor. Abbildung 37 zeigt das vorgeschlagene AsTeRICS-Grid für dieses Modell.

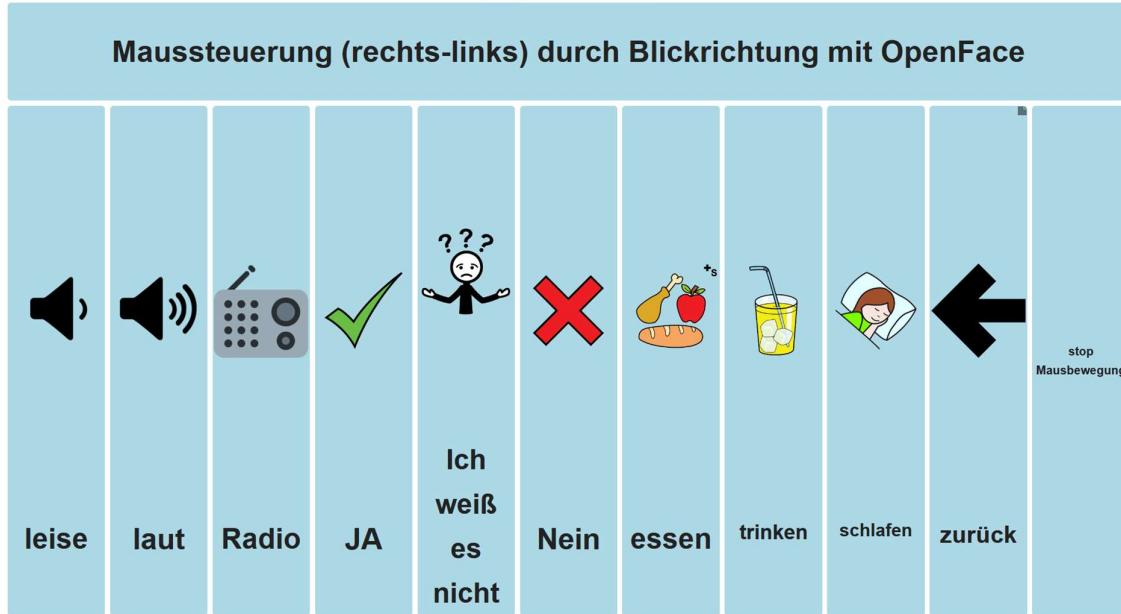


Abbildung 37. Vorgeschlagenes AsTeRICS-Grid für das Blickrichtungs-Modell

Die notwendigen Einstellungen in OpenFace für das Laufen dieses Modells sind:

- File -> Open webcam (Auflösung der Webcam auswählen)
- Record-> CSV to socket
- Record -> AUs
- Record -> 2D Landmarks
- Record -> gaze

## 7 Resultate

Es wurden Tests durchgeführt, um die Genauigkeit der durch Gesichtsmimik gesteuerten Maus mit der Genauigkeit einer Standardmaus, welche per Hand bedient wird, zu vergleichen. Um die Mauspräzision zu messen wurde AimBooster [14] verwendet. AimBooster ist ein Tool, mit dem die Genauigkeit, Geschwindigkeit oder Reaktionsgeschwindigkeit bei der Steuerung einer Maus gemessen werden können. Das Tool wurde konfiguriert, um die Mauspräzision beim Klicken auf Ringe zu messen, welche zu verschiedenen Zeiten und auf verschiedenen Positionen auf dem Bildschirm erscheinen und nach 10 Sekunden wieder verschwinden. Das Ziel des Tests ist, die meisten Ringe in einem Zeitraum von 60 Sekunden mit einem Mausklick zu treffen. Jeder Ring war 100 Pixel groß. Das Zielen auf die Ringe erfolgte beim Klicken auf die linke Maustaste. Die Genauigkeit wurde als Prozentsatz angegeben (siehe Abbildung 38).

Für den Test des Modells zur Maussteuerung mittels Blickrichtung wurde AimBooster dahingehend eingestellt, sodass die Mausbewegung auf einem Raster von nur 100 Pixel Höhe getestet werden konnte (siehe Abbildung 39 ).

Um die Ergebnisse zu evaluieren wurde der Durchschnitt der 5 besten Versuche jeder Person berechnet, da das Ergebnis der ersten Versuche niedriger war als das der letzten, bei denen die Probanden(innen) mehr Übung im Umgang mit der Maus erworben hatten. Schließlich wurde für jede Methode der Durchschnitt der Ergebnisse aller Probanden(innen) zusammen berechnet.

Der Versuch mit der rechten Hand ergab die besten Ergebnisse und erreichte eine Genauigkeit von 100 % gefolgt von dem Versucht mit der linken Hand mit 75% Genauigkeit. Mit der Maussteuerung mittels Kopfbewegung wurde eine Genauigkeit von 53% erreicht, mit der Position der Nase 52% und mit der Blickrichtung 43%. (siehe Tabelle 8. Ergebnis der Mauspräzision und Erfüllung von Aufgaben. Ein grüner Haken bedeutet, dass die Aufgabe erfüllt werden konnte)

Die Tests wurden von 4 Personen zwischen 11 und 45 Jahren ohne Behinderung und mit Erfahrung im Umgang mit einer Maus mit der rechten Hand durchgeführt. Jede Person hatte 25 Versuche mit jeder der Methoden.



Abbildung 38. Beispiel eines Präzisionstests der Modelle mit AimBooster. Für die Standard Maus, Kopfbewegung und Position der Nase.



Abbildung 39. Beispiel eines Präzisionstests des Modells zur Maussteuerung durch Blickrichtung.

Maussteuerung Methode	Genauigkeit	Internet-Surfen	Öffnen von Dokumenten	Unterstützte Kommunikation AsTeRICS-Grid
Rechte Hand	100%	✓	✓	✓
Linke Hand	75%	✓	✓	✓
Kopfbewegung	53%	✓	✓	✓
Position der Nase	52%	✓	✓	✓
Blickrichtung	43%	nicht zutreffend	nicht zutreffend	✓

Tabelle 8. Ergebnis der Mauspräzision und Erfüllung von Aufgaben. Ein grüner Haken bedeutet, dass die Aufgabe erfüllt werden konnte

Es wurde auch ein zweiter Test durchgeführt, bei dem die Probanden(innen) drei Aufgaben mit dem Maus erfüllen mussten. Die Aufgaben waren: ein Dokument zu öffnen, im Internet zu surfen und ein AsTeRICS-Grid zu benutzen und zwar mit Hilfe der Modelle zur Maussteuerung durch Kopfbewegung und Position der Nase und im Vergleich dazu mit einer Standardmaus mit der linken und der rechten Hand. Am Ende der Tests mussten die Probanden(innen) beantworten, mit welcher Methode sie die Aufgabe erfüllen konnten. Alle Probandeninnen antworteten, dass sie alle Aufgaben mit den verschiedenen Methoden erfüllen konnten. Sie erwähnten, dass sie mit der Mausteuerung durch Gesichtsmimik Schwierigkeiten hatten und mehr Zeit zur Erfüllung der Aufgaben gebraucht hatten, aber mit der mehr Übung wurden die Fähigkeiten verbessert. Das Modell zur Maussteuerung mittels Blickrichtung wurde nur mit dem im Projekt vorgeschlagenen AsTeRICS-Grid getestet. Die Probanden(innen) beantworteten, dass sie das Grid mit der Blickrichtungs-Methode benutzen konnten. Die erzielten Ergebnisse in beiden Tests sind in Tabelle 8 zusammengefasst.

Die Tests wurden auf einem Computer mit einer Bildschirmauflösung von 1080p, mit einer integrierten Webcam mit einer Auflösung von 1280 x 720 und eine Bildfrequenz von 30 fps (Bilder pro Sekunde) durchgeführt.

Alle drei Modelle wurden mit der integrierten Webcam und einer USB-Kamera mit verschiedenen Auflösungen getestet. Beide hatten eine Bildfrequenz von 30 fps. Die verwendeten Auflösungen waren:

- 640 x 480
- 960 x 540
- 1280 x 720
- 1920x1080

Das Verhalten der Maus war sehr ähnlich mit den verschiedenen Auflösungen.

## 7.1 Bewertung der Resultate

Um einen Computer zu bedienen, können die Funktionen der Maus mit den im Projekt entwickelten Modellen zur Maussteuerung durch Kopfbewegung und Position der Nase aktiviert werden. Die Funktion der Tastatur kann mittels der Modelle und der verschiedenen Tastaturen in AsTeRICS-Grids durchgeführt werden.

Die teilweise geringe Genauigkeit der Maussteuerungsmethoden, welche im Rahmen dieses Projekts entwickelt wurden, ist größtenteils auf die mangelnde Erfahrung der Probanden(innen) bei der Steuerung einer Maus zurückzuführen und im speziellen auf die fehlende Erfahrung in der Steuerung von Objekten mittels Kopf- und Augenbewegung oder Gesichtsmimik. Die Genauigkeit der Maussteuerung mittels Gesichtsmimik kann durch Übung verbessert werden [15].

Die Erstellung eines Modells zur simultanen Mauststeuerung in x- und y- Richtung mittels Blickrichtung ist nicht gelungen. Das Problem war, dass die Maussteuerung mithilfe von Blickrichtungswinkeln zu sensibel war. Aufgrund der physiologischen Eigenschaften der Augen ist es fast unmöglich den Blick mit einer gleichmäßigen Geschwindigkeit über den Bildschirm zu führen. Eine bewusste horizontale Bewegung der Augen bewirkt zugleich unbewusste Bewegungen in vertikaler Richtung und umgekehrt.

## 8 Schluss

Im Rahmen dieses Projekts wurden drei Modelle erstellt:

- Das Modell zur Maussteuerung durch Kopfbewegung und Action Units
- Das Modell zur Maussteuerung durch Verfolgung der Position der Nase und Action Units
- Das Modell zur Maussteuerung durch die Blickrichtung und Action Units

Letztgenanntes Modell ist nur in Verbindung mit einem Grid für unterstützte Kommunikation wie z.B. einem AsTeRICS-Grid zu verwenden. Dieses Modell ist ideal für Personen, bei denen die Handfunktion und die Kopfbewegung nicht vorhanden sind.

Die durchgeführten Tests zeigten, dass die Funktionalität einer Computer-Maus mithilfe der erstellten Modelle zur Maussteuerung durch Kopfbewegung oder durch Verfolgung der Position der Nase ersetzt werden kann. Darüber hinaus kann jedes der drei AsTeRICS-Modelle zur Computersteuerung durch Gesichtsmimik an die unterschiedlichen Bedürfnisse von Menschen mit Beeinträchtigung angepasst werden.

## Literaturverzeichnis

- [1] „Parlament fordert neue ehrgeizige EU-Strategie zugunsten von Menschen mit Behinderungen.“ *Europäisches Parlament*, 19 06 2020.
- [2] R. E. d. E. I. Discapacida, „Good practices in inclusive education and disability in Europe. Europa Social Fund,“ *Fondo Social Europeo*, September 2012.
- [3] „Employment and disability in the European Union.“ | *European Parliamentary Research Service*.
- [4] M. Deinhofer, „OpenFace-SocketStream-2.2.0deinhofer,“ 17 12 2020. [Online]. Available: <https://github.com/deinhofer/OpenFace/releases/tag/v2.2.0deinhofer>. [Zugriff am 20 Januar 2021].
- [5] AsTeRICS, „Create Customized Low-Cost Assistive Technologies for People with Disabilities.,“ 2020 July 5. [Online]. Available: <https://www.asterics.eu/>. [Zugriff am 20 01 2021].
- [6] J. F. Cohn und P. Ekman, Measuring Facial Action by Manual Coding, Facial EMG,, 2004.
- [7] H. Mohammed, M. Dzulkifli, H. B. Ahmad und M. Farhan, „A Crucial Investigation of Facial Skin Colour Research Trend and Direction,“ *International Journal of Multimedia and Ubiquitous Engineerin*, Januar 2015.
- [8] M. Christian, Mimik als Ausdruck von Gefühlen, Graz: Masterarbeit, 2017.
- [9] T. Baltrusaitis, „OpenFace,“ 2021. [Online]. Available: <https://github.com/TadasBaltrusaitis/OpenFace/wiki/Action-Units>. [Zugriff am 20 Januar 2021].
- [10] T. Baltrušaitis, A. Zadeh, Y. Chong Lim und L.-P. Morency, „OpenFace 2.0: Facial Behavior Analysis Toolkit,“ *IEEE, International Conference on Automatic Face and Gesture Recognition*, 2018.
- [11] E. Arcoverde, R. Duarte, R. Barreto, J. Magalhaes, C. Bastos, T. Ing Ren und G. Cavalcanti, „Enhanced real-time head pose estimation system for mobile device.,“ *Integrated Computer Aided Engineering*. 21. 281-293. 10.3233/ICA-140462., 2014.

- [12] T. Baltrušaitis, P. Robinson und L.-P. Morency, „Constrained Local Neural Fields for robust facial landmark detection in the wild,“ *IEEE Int. Conference on Computer Vision Workshops, 300 Faces in-the-Wild Challenge*, 2013.
- [13] W. Errolll, B. Tadas, Z. Xucong, S. Yusuke, R. Peter und B. Andreas, „Rendering of Eyes for Eye-Shape Registration and Gaze Estimation,“ in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.
- [14] AimBooster, „AimBooster,“ 18 Oktober 2013. [Online]. Available: <http://www.aimbooster.com/>. [Zugriff am 26 1 2021].
- [15] M. Peters und J. Ivanoff, „Performance Asymmetries in Computer Mouse Control of Right-Handers, and Left-Handers With Left- and Right-Handed Mouse,“ *Experience, Journal of Motor Behavior*, 31:1, 86-94, 1999.
- [16] J. Reilly, J. Ghent und J. McDonald, „Modelling, Classification and Synthesis of Facial Expressions,“ in *Affective Computing*, Ireland, 2008.
- [17] T. Baltrušaitis, P. Robinson und M. Marwa, „Cross-dataset learning and person-specific normalisation for automatic Action Unit detection,“ *Facial Expression Recognition and Analysis Challenge, IEEE International Conference on Automatic Face and Gesture Recognition*, 2015.

# Abbildungsverzeichnis

Abbildung 1. Facial Action Units. Quelle: <a href="https://github.com/TadasBaltrusaitis/OpenFace/wiki/Action-Units">https://github.com/TadasBaltrusaitis/OpenFace/wiki/Action-Units [9]</a> .....	11
Abbildung 2. OpenFace Framework, Darstellung der Implementierung moderne Algorithmen zu Analyse des Gesichtsverhaltens. Quelle: <a href="https://github.com/TadasBaltrusaitis/OpenFace/wiki/Action-Units">https://github.com/TadasBaltrusaitis/OpenFace/wiki/Action-Units [9]</a> .....	12
Abbildung 3. Einige im Bild dargestellte Action Units, die mit OpenFace erkannt werden können. Quelle: "Facial Expression Analysis" by Fernando De la Torre and Jeffrey F. Cohn [11] .....	14
Abbildung 4. Beispiel für die Erkennung von Gesichts-Landmarken in verschiedenen Gesichtsorientierungen mit OpenFace. Quelle: Constrained Local Neural Fields for robust facial landmark detection in the wild [12] .....	14
Abbildung 5. 68 Facial Landmarks. Quelle: Constrained Local Neural Fields for robust facial landmark detection in the wild [12] .....	15
Abbildung 6. Schätzung der Kopfhaltung von OpenFace. Quelle: OpenFace 2.0: Facial Behavior Analysis Toolkit [10].....	16
Abbildung 7. Ausrichtung des Kopfes in Bezug auf Nick-, Roll- und Gierbewegungen (Pitch, Roll, Yaw). Quelle: Reference Real-Time Head Pose Estimation for Mobile Devices [11] ....	17
Abbildung 8. Schätzung der Blickrichtung von OpenFace mittels Maschine Learning. Quelle: Rendering of Eyes for Eye-Shape Registration and Gaze Estimation [13].....	18
Abbildung 9. Darstellung der von OpenFace geschätzten Blickrichtung (Pfeile in grün). Quelle: [9].....	19
Abbildung 10. Position der Merkmale im Augenbereich. Quelle: [9].....	20
Abbildung 11. Optionen zur Analyse von Bildern oder Videoaufnahmen. Die Auswahl zur Analyse von Echtzeit-Videos ist in rot markiert .....	20
Abbildung 12. Graphische Darstellung der Videoanalyse von Kopfhaltung, Blickrichtung, Gesichtsmerkmalen und AUs .....	21
Abbildung 13. Option für das Speichern oder Streamen der Tracking-Daten und Auswahl der Eigenschaften zur Analyse .....	21
Abbildung 14. Beispiel für die Erstellung von Eines Modells mit der AsTeRICS Configuration Suite.....	22
Abbildung 15. Integration von OpenFace in AsTeRICS .....	23
Abbildung 16. Option Send Data zum Server-Socket in OpenFace .....	24
Abbildung 17. ClientSocket-Plugin .....	25
Abbildung 18. Die Funktion <i>Connect ()</i> startet die Verbindung und die Kommunikation mit dem Server.....	27
Abbildung 19. Funktion, die die Kommunikation zwischen AsTeRICS und den Server in ARE startet.....	27

Abbildung 20. Verbindung zwischen dem ClientSocket-Plugin und dem StringSplitter-Plugin .....	28
Abbildung 21. Einstellung des ClientSocket-Plugin.....	29
Abbildung 22. Einstellung des StringSplitter-Plugins .....	29
Abbildung 23. Funktionsweise der Maussteuerung durch Kopfbewegung und durch AUs....	30
Abbildung 24. Umwandlung Radiant auf Grad mit dem MathEvaluator-Plugin.....	31
Abbildung 25. Verfahren für die Mausbewegung nach rechts.....	31
Abbildung 26. Schematische Darstellung des Modells für die Mausbewegung mittels Kopfbewegungen (roll und pitch) .....	32
Abbildung 27. Auslösung des Doppelklicks durch Lippensaugen für 600 Millisekunden .....	33
Abbildung 28. Schematische Darstellung des Modells für das Klicken auf die Maustasten... 33	
Abbildung 29. Maussteuerung durch Verfolgung der Position der Nase und durch AUs.....	34
Abbildung 30. Konfiguration der Mausbewegung rechts-links durch Verfolgung der Nasenposition .....	35
Abbildung 31. Berechnung der Mausbewegung mit Richtung und Geschwindigkeit .....	36
Abbildung 32. Schematische Darstellung des Modells für die Mausbewegung Rechts-links durch Verfolgung der Nasenposition.....	36
Abbildung 33. Funktionsweise der Maussteuerung mittels Blickrichtung .....	37
Abbildung 34. Schematische Darstellung des Modells für die horizontale Mausbewegung... 38	
Abbildung 35. Umwandlung Radiant auf Grad und Skalierungsfaktor.....	39
Abbildung 36. Aktivierung und Deaktivierung der Mausbewegung mit dem Threshold-Plugin .....	39
Abbildung 37. Vorgeschlagenes AsTeRICS-Grid für das Blickrichtungs-Modell .....	40
Abbildung 38. Beispiel eines Präzisionstests der Modelle mit AimBooster. Für die Standard Maus, Kopfbewegung und Position der Nase.....	41
Abbildung 39. Beispiel eines Präzisionstests des Modells zur Maussteuerung durch Blickrichtung.....	42

## Tabellenverzeichnis

Tabelle 1. Facial Action Units (FACS) [6].....	10
Tabelle 2. Die fünf ersten Spalten der CSV-Datei generiert von OpenFace.....	12
Tabelle 3. Facial Action Units, die mit OpenFace erkannt werden können.....	13
Tabelle 4. Gesichtsmerkmale in der CSV-Datei.....	16
Tabelle 5. Daten der Kopfposition in der CSV-Datei .....	17
Tabelle 6. Auslesen von Blickrichtung und Augenmerkmalen.....	19
Tabelle 7. Konfiguration des ClientSocket-Pugins mit der AsTeRICS PluginCreationWizard-Tool .....	26
Tabelle 8. Ergebnis der Mauspräzision und Erfüllung von Aufgaben. Ein grüner Haken bedeutet, dass die Aufgabe erfüllt werden konnte .....	42