

Höhere technische Schule fur Informationstechnologie
Shkolla e mesme profesionale private pér teknologji informacioni
Österreichiche Schule Peter Mahringer
Shkolla Austriake Shkodër

Gesichtsregistrierung und Gesichtserkennung

Diplomarbeit Nr. 17.06

Klasse 5A, Schuljahr 2019/20



Ausgeführt von: Aron Terzeta
Rei Hoxha
Egli Hasmegaj
Jordi Zmiani

Projektbetreuer1: Matthias Maurer
Projektbetreuer2: Dominik Stocklasser
Projektbetreuer3: Andreas Kucher

Shkoder, 22. März 2020

Eidesstattliche Erklärung

Wir versichern, dass wir die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt haben. Wir haben uns keiner anderen als der im beigefügten Quellenverzeichnis angegebenen Hilfsmittel bedient. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht.

Aron Terzeta

Ort, Datum
Rei Hoxha

Unterschrift

Ort, Datum
Egli Hasmegaj

Unterschrift

Ort, Datum
Jordi Zmiani

Unterschrift

Ort, Datum

Unterschrift

Approbation Datum u. Unterschrift	PrüferIn	IT-Koordinator/Direktion
--------------------------------------	----------	--------------------------

Sämtliche in dieser Diplomarbeit verwendeten personenbezogenen Bezeichnungen sind geschlechtsneutral zu verstehen.

Kurzfassung

Vom Auftraggeber wird ein System gefordert, das Gesichter erkennt, um einen kontrollierten Zugang in der Schule zu ermöglichen und die Sicherheit der Schule dadurch zu erhöhen. Alle Gesichter von den aktuellen Schülern und Lehrer sollen erkannt werden. Es soll auch der Unterschied zwischen einer reellen Person und einem Foto berücksichtigt werden.

Wir haben uns für diese Idee entschieden, weil Sicherheit heute hoch interessant und relevant ist. Es geht hier darum, einen kontrollierten Zugang in Institutionen mittels Gesichtserkennung zu ermöglichen, da Gesichter eindeutig für jede Person sind. Die größten Herausforderungen und Voraussetzungen des Projekts im Planungsprozess. Eine andere Voraussetzung ist der Gebrauch von zwei Kameras, damit der Unterschied zwischen einer reellen Person und einem Foto erkannt wird.

Abstract

This paper represents the face detection and recognition system that enables the detection of a human face and is able to identify it. It is thought to improve the security system of an institution while controlling the access of certain locations, in particular rooms. Further it achieves the goal of differentiating between a real person and a photo.

Ky punim paraqet sistemin e zbulimit dhe njohjes së fytyrës që mundëson zbulimin e një fytyre njerëzore dhe është në gjendje ta identifikojë atë. Mendohet se përmirëson sistemin e sigurisë së një institucionit ndërsa kontrollon hyrjen në disa lokacione, dhoma. Më tej ajo arrin qëllimin e diferencimit midis një personi të vërtetë dhe një fotografie që identifikohet.

Dieses Dokument stellt das Gesichtserkennungs- und -erkennungssystem dar, mit dem ein menschliches Gesicht erkannt und identifiziert werden kann. Es wird geplant, dass es das Sicherheitssystem einer Institution verbessert und gleichzeitig den Zugang zu bestimmten Orten und Räumen kontrolliert. Ferner soll zwischen einer realen Person und einem Foto unterschieden werden.

Danksagung

Inhaltsverzeichnis

1	Allgemeines	1
2	Planung	3
2.1	Projektziele	3
2.1.1	Ziele	3
2.1.2	Nicht Ziele	4
2.1.3	Optionale Ziele	4
2.2	Projektplanung	4
2.3	Projektmanagementmethode	5
3	Gesichtsregistrierung	6
3.1	Umsetzung	6
3.1.1	Allgemein	6
3.1.2	Verwendete Technologien	6
3.1.3	Technische Lösung	7
3.1.4	Herausforderungen	13
3.1.5	Qualitätssicherung und Controlling	15
3.2	Ergebnisse	17
3.2.1	Implementierung	17
4	Datenbank und 2D vs. 3D Unterschied	18
4.1	Umsetzung	18
4.2	Technische Lösungen	21
4.2.1	Datenbank	21
4.2.2	2D vs. 3D Unterschied	24
4.3	Herausforderungen	27
4.3.1	Datenbank - MariaDB	27
4.3.2	Disparity Map	28
4.3.3	Python Scripting	31
4.3.4	OpenCV	32
4.4	Qualitätssicherung	32

4.5	Ergebnisse	33
4.5.1	Datenbank	33
4.5.2	2D vs. 3D Unterschied	34
4.6	Evaluierung und Resümee	34
4.6.1	Planung vs. Realisierung	34
4.6.2	Wertschöpfung und Lessons Learned	34
5	Bildverarbeitung	35
5.1	Allgemeines	35
5.1.1	Entwicklungsumgebung und Technologien	35
5.1.2	Frameworks und Bibliotheken	36
5.2	Technische Lösungen	37
5.2.1	Lösungsweg - Structed Software design	38
5.2.2	Gesichtserkennung	39
5.2.3	Normalisierung	41
5.2.4	Zuschneiden von Gesichter	45
5.2.5	Gesichtsschlüsselpunkte Extraktion	46
5.3	Herausforderungen, Probleme und deren Lösung	48
5.3.1	Lösungen	48
5.4	Qualitätssicherung, Controlling	49
5.5	Ergebnisse	50
5.6	Evaluierung und Resümee	50
5.6.1	Planung vs. Realisierung	50
5.6.2	Ergebnisse	50
6	Gesichtserkennung	52
6.1	Allgemeines	52
6.2	Technische Lösung	53
6.2.1	Hardware und Aufbau	53
6.2.2	Software	56
6.3	Herausforderungen, Probleme, und deren Lösung	71
6.4	Projektmanagement und Controlling	73
6.5	Ergebnisse	75
6.6	Was wurde gelernt und welche Verbesserungsmöglichkeiten gibt es	75
7	Ausblick	77
7.1	Wie geht es weiter ?	77
7.2	Was passiert mit dem Ergebnis?	77
7.3	Was passiert mit dem Prototyp?	77

7.4 Gibt es Folgeprojekte?	77
7.5 Gibt es Interessanten?	77
8 Handbuch	78

Kapitel 1

Allgemeines

Unten werden die Idee, das Thema und die Aufgabenstellung dieser Diplomarbeit verfasst.

In der heutigen Zeit ist die Technologie eines der wichtigsten Themen für die Welt. Mithilfe der Technologie hat die Menschheit wesentliche Fortschritte gemacht im Bereich der Lebensqualitätsverbesserung und -erleichterung. Mehr als jemals zuvor kommen intelligente Systeme zum Einsatz, die den Arbeitsaufwand und die Arbeitskomplexität reduzieren. Diese Systeme werden in den wichtigsten Bereiche des Lebens verwendet, wie zum Beispiel: Menschlichen Beziehungen, Business, Transport, Banken, Medizin, Bildung und Kommunikation. Selbstverständlich gibt es zahlreiche Versuche von verschiedenen Menschen, auf diesen Systemen zu zugreifen. Deshalb ist die Sicherheit in diesen Systemen eine der aktuellsten und wichtigsten Themen. Der unkontrollierte Zugriff auf solche wichtigen und delikaten Systeme würde zu fatalen Konsequenzen führen. Aus diesem Grund muss sehr genau und detailliert geplant werden, welche Sicherheit Strategien die klügsten bzw. die Besten sind.

Das Hauptthema dieses Projekts ist genau die Sicherheit der Systeme, und zwar die Ermöglichung des kontrollierten Zugriffs von Menschen in verschiedenen Gebäuden. Es wurde die Verantwortlichkeit übernommen, der Zugriff in der Österreichischen Schule "Peter Mahringer" überwiesen und kontrolliert zu machen. Zu diesem Zweck wird in diesem Projekt ein intelligentes System entwickelt, dass die Aufgabe hat, menschliche Gesichter (Gesichtern von Schülern, Lehrern, Eltern usw.) zu registrieren und zu erkennen. Es wird höchstwahrscheinlich in der Schultür integriert, weil dort müssen alle Personen unbedingt vorbeigehen. Dieses Projekt wird eine große Hilfe für die Schule sein, aufgrund der von Kameras und komplexen Algorithmen angebotenen Möglichkeiten. Beispielweise wird es überprüft, ob die Person, die reinkommt ein Student oder Lehrer ist, oder es wird notiert, wer die Schule wann verlässt oder besucht. Hat eine Person kein Zugriff, wird sie nicht herein gelassen.

Um dieses Projekt zu realisieren, wird das Projekt in drei große und wichtige Teile zerlegt. Das erste Teil heißt Bildverarbeitung und hat die Aufgabe, Gesichtsbildern von der Kamera zu normalisieren und zu verarbeiten, damit sie in geeigneten Format für den Gesichtsvergleichung sind. Der zweite Teil beschäftigt sich mit der Registrierung der Gesichter von Personen am Server. Der dritte Teil ist die Gesichtserkennung. Hier geht es um die Erkennung der Gesichter der Benutzer des

Systems.

Das Team besteht aus Aron Terzeta, Egli Hasmegaj, Rei Hoxha und Jordi Zmiani.
Aufgaben sind wie folgend geteilt.

- Aron beschäftigt sich hauptsächlich mit dem Gesichtsregistrierungsteil.
 - Egli kümmert sich um die wichtigsten Gesicht Daten zu extrahieren(Größe und Form der Augenhöhlen, Nase, Wangenknochen und Kiefer). Position/ Verhältnisse der Hauptmerkmale relativ zueinander herausheholen. Aufbereitung der Daten für Abgleich.
 - Rei macht den Systemaufbau und die Gesichtserkennung
 - Jordi kümmert sich um die Datenbank und 2D vs 3D Unterschied.
- Wir sind dafür hoch motiviert, dieses Projekt richtig umzusetzen.

Kapitel 2

Planung

Dieses Kapitel beschreibt im Detail wie die Diplomarbeit gestaltet und abgegrenzt ist. Die Abgrenzung der Arbeit ist entscheidend wegen der hohen Komplexität des Projektes. Sie erfolgt durch Ziele, nicht-Ziele und optionale Ziele. Das ist im Unterkapitel 2.1 genau verfasst. Weiter folgt die Planung im Kapitel 2.3. Es werden hier das Lösungskonzept und die Projektmanagement erklärt. Es wird nun spezifiziert welche Projektmanagementmethode eingesetzt wurde.

2.1 Projektziele

In diesem Abschnitt werden die Ziele, nicht Ziele und optionalen Ziele beschrieben.

2.1.1 Ziele

Ziele sind wesentlich für jedes Projekt. Deshalb wurden die Ziele dieses Projekts in drei Kategorien geteilt. In der ersten Kategorie gehören Ziele, die unbedingt erfüllt werden müssen. Anderfalls würde das Projekt scheitern.

1. Live vs. Foto unterscheiden. (3-dimensionale Erkennung an Gesicht machen. Tiefe messen damit zwischen einer Person und einem Foto differiert wird.)
2. Gesichts-Schlüsselpunkt-Extraktion, um ein Gesicht zu identifizieren.
3. Größe und Form der Augenhöhlen, Nase, Wangenknochen und Kiefer analysieren.
4. Position/Verhältnisse der Hauptmerkmale relativ zueinander herausholen.
5. Bilderdaten in Vektoren umwandeln mithilfe eines Algorithmus.
6. Abstimmung (Vergleichen mit den Gesichtspunkten aller Personen in der Datenbank, um zu sehen, ob die Person schon registriert wurde).
7. Bis zu 500 Personen in einer Datenbank speichern.
8. 10 Tests, jeder Test in einer anderen Raumkondition, um möglichst viele Betriebskonditionen zu testen.

9. Datenbankdesign(Struktur der Datenbank, Datenbank erstellen, Datenbank einrichten)
10. Error checking (Überprüfen, ob Fehler während des Betriebs auftreten)
11. Safe Mode (eine Batterie, Back-ups in einem lokalen Server)
12. Min. Arbeitsvorbereitung (Min. Gesichtsdetektionszeit)
13. Admin account erstellen

2.1.2 Nicht Ziele

Hier sind die Nicht-Ziele definiert, damit das Projekt begrenzt ist und damit Nichts gemacht wird, was nicht angefordert war.

1. Mehr als ein Gesicht gleichzeitig erkennen.
2. Maske, Brille oder Hüte während der Gesichtsdetektion tragen.
3. Gesicht in Bewegung erkennen.
4. Person ins Profil oder andere Position sein.
5. Thermische Kamera einsetzen.

2.1.3 Optionale Ziele

Hier gehören Ziele, die optional sind. Das heißt sie sind nicht zwingend und wurden eingesetzt nur nachdem alle wichtigen und primären Ziele erfüllt sind.

1. Öffnung der Haustüren oder jeder anderen Tür mit Gesichtserkennung.
2. LCD-Display Implementation.
3. Licht neben der Kamera (Night Vision implementieren damit die Erkennung/Registrierung auch dann funktioniert, wenn es dunkel ist.)

2.2 Projektplanung

Unsere Big Picture¹ ist unser erstes grobes Design, das die Lösungsskizze des Projekts beschreibt. Es gibt bestimmte Gründe, warum Big Picture und Structured Design verwendet wurden, um die Software zu beschreiben. Diese Methode ermöglicht eine sehr gute Darstellung und Beschreibung des Lösungswegs. Ist schnell und leicht zu machbar. Alles ist klar sichtbar und nicht kompliziert. Big Picture und Structured

¹vermittelt ein Überblick über das gesamte System.[41]

Design² folgt das Top-Down Prinzip, das heißt die Funktionen werden hierarchisch zerlegt (Jede Funktion wird in die folgenden Ebenen detaillierter beschreibt).

Structured Design und Big Picture haben keine Begrenzung. Dort können eindeutig alle Funktionen, Schnittstellen, Signalen und Daten beschrieben werden, sodass es von allen leicht zu verstehen ist. Sehen Sie Abb. 2.1

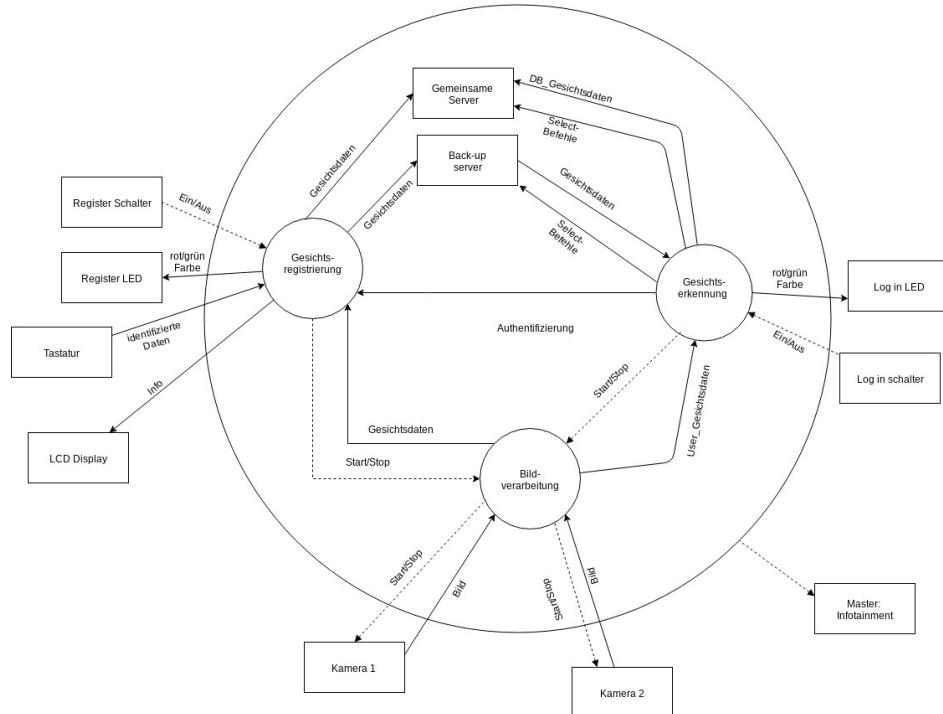


Abbildung 2.1: Big picture

2.3 Projektmanagementmethode

Als Projektplanmethode haben wir Scrum, eine agile Methode, gewählt, weil es die Möglichkeit bietet, komplexe Projekte mit einem kleinen Personenkreis zu verwalten. Scrum ist ideal für Software- bzw. Hardware-Entwicklungsteams, weil das Team während des Projekts verschiedene Änderungen an seinem Plan vornehmen muss. Aus diesem Grund ist es besser, tägliche Zielvorgaben zu haben und in einem kurzen Zeitraum von 1 bis 4 Wochen so genannte Sprints durchzuführen, bei denen das Ziel am Ende dieser Sprints ein Prototyp ist. Verschiedene Prototypen herzustellen und am Ende den richtigen auszuwählen, ist die beste Wahl für die Projektmanagementmethode zur Gesichtserkennung. Es gibt auch tägliche Pläne, in denen sich das Team zusammensetzt und entscheidet, was die Ziele für den Tag sind und was sie tun müssen.[26]

²ist eine grafische Veranschaulichung des gesamten Systems. Das bedeutet, es wird wie bei einer Zwiebelschale mit der äußersten Schicht begonnen. Jede weitere Schicht der Zwiebel geht weiter ins Detail.[41]

Kapitel 3

Gesichtsregistrierung

In diesem Kapitel wird genauer beschrieben, wie der Gesichtsregistrierungsteil funktioniert.

3.1 Umsetzung

Hier wird erklärt, wie der Gesichtsregistrierungsteil implementiert wird.

3.1.1 Allgemein

Sicherheit ist heutzutage hoch interessant und relevant in vielen technischen und nicht technischen Bereichen. Das System, das entwickelt wird, hat mit Gesichtern von Personen zu tun. Alle wissen, dass das Gesicht für jede Person anders ist. Jede Person wird mit ihrem Gesicht authentifiziert, weil es einzigartig ist. Das Gesicht hat Merkmale, die von verschiedenen Algorithmen bestimmt werden können, um diese für die Authentifizierung der Personen zu verwenden

Das System ist in zwei Teile geteilt. Es gibt den Registrierungsteil und den Erkennungsteil. Bei dem Registrierungsteil wird die komplette Registrierung der Schüler und Schülerinnen, der Lehrer und Lehrerinnen durchgeführt. Bei dem Erkennungsteil wird die Authentifizierung der Schüler und Schülerinnen gemacht.

3.1.2 Verwendete Technologien

Hier werden die Technologien beschrieben, die verwendet wurden. Das andere Paket namens "git" wird auch verwendet. Git ist ein Versionsverwaltungssystem, das verschiedene Versionen einer Software auf einem Server speichert. Auf dem Server gibt es dann verschiedene Versionen der Software und wir können zu den alten Versionen zurückkehren. Es wird meistens bei der Implementierung-Phase verwendet, um die Veränderungen des Source-Codes zu verwalten.

Ein anderes Paket, das wir verwenden, heißt cmake³ ist ein Paket, das gebraucht wird,

³ein erweiterbares Open-Source-System, das den Erstellungsprozess in einem Betriebssystem und auf compilerunabhängige Weise verwaltet.

wenn man mit OpenCV⁴ arbeitet. Es muss das System so konfiguriert sein, damit das OpenCV-Framework in einem C++ Programm verwendet werden kann. Deshalb brauchen wir spezielle cmake Befehle, die dies ermöglichen.

”CMake wird verwendet, um den Softwarekompilierungsprozess mithilfe einfacher plattform- und compilerunabhängiger Konfigurationsdateien zu steuern und native Makefiles und Arbeitsbereiche zu generieren, die in einer Compilerumgebung Ihrer Wahl verwendet werden können.”[22]

Das gleiche passiert auch, wenn z.B. Python statt C++ verwendet wird. Die anderen Pakete wie z.B. libgtk2.0-dev pkg-config, libavcodec-dev, libavformat-dev, libswscale-dev, sind benötigte Paketen, damit man das OpenCV-Framework verwenden kann.

Linux als Betriebssystem [37]

Linux ist das meist verwendete Betriebssystem der Welt. Es ist eine open-source Software. Linux ist flexibel, man kann die einzelnen Modulen wegnehmen, ohne dass das Betriebssystem abstürzt. Der Benutzer kann auch die Kernkomponenten wählen wie z.B. welcher System-Grafiken angezeigt werden, bzw. die ganzen Komponenten der Benutzeroberfläche können ausgewählt werden. Ich habe Linux aus verschiedene Gründe gewählt. Linux ist für eingebettete Systeme sehr geeignet. Es ist sicher gegen Schadprogrammen, Viren und Trojanern. Linux ist einfacher. Vorher war es ein kompliziertes System, seit den Bemühungen der Ubuntu-Fundationen und der Ubuntu-Distribution ist es sehr einfach verwendbar.

Python als Programmiersprache

”Python ist eine Programmiersprache, die 1991 veröffentlicht wurde. Python besitzt eine einfache Lesbarkeit und eine eindeutige Syntax. Python lässt sich leicht erlernen und unter UNIX, Linux, Windows und Mac OS verwenden.”[27] Warum Python gewählt wurde, hat verschiedene Gründe. Python hat weniger Schlüsselwörter, reduziert die Syntax auf das Wesentliche und optimiert die Sprache. Ein Programm, das in Python geschrieben ist, ist vom Betriebssystem unabhängig. Das bedeutet, es kann Plattform unabhängig interpretiert werden.

3.1.3 Technische Lösung

In diesem Unterabschnitt wird alles was mit Technik zu tun hat erklärt. Es wird eine detaillierte Beschreibung im Bezug auf die technischen Lösungen gemacht.

Das System besteht aus verschiedenen sogenannten Terminatoren. Ein Terminator befindet sich außerhalb des zu definierenden Systems. Es kann eine andere Person, System oder eine Organisation sein. ”Terminatoren können von unserem System Informationen, Nachrichten, Materialien oder Energie erhalten oder das System empfängt diese.”[40]

Ein Terminator ist der ”Register-Schalter”. Er initialisiert das ganze System. Schalter in der Technik ist nichts anderes, als ein Gerät zum Ein- und Ausschalten des Stroms

⁴eine Bibliothek von Programmierfunktionen, die hauptsächlich auf Computer Vision in Echtzeit abzielen.

oder zum Leiten des Stromflusses. Wenn der Schalter gedrückt wird, bekommt das System einen Input, transformiert ihn und gibt dann einen Output. Das System ist sehr einfach verwendbar.

Die Register-LED⁵ dient als eine Anzeige. Wenn mit dem System etwas nicht stimmt, dann wird mit rot geleuchtet. Wenn alles passt, dann wird mit grün geleuchtet. Eigentlich hat die normale LED nur eine Farbe, aber es wird eine spezielle LED verwendet, namens RGB LED⁶. Eine RGB LED hat drei Grundfarben, rot, grün und blau. Mit diesen drei Farben kann man alle Farben darstellen. Es könnte auch zwei LEDs geben, rot und grün, aber es ist effektiver, eine RGB LED zu verwenden.

Eine spezielle Eigenschaft des Systems ist die Verwendung einer Tastatur. Sie wird verwendet, weil die einzelnen Personen ihren Namen, bzw. ihr Email eingeben müssen. Die andere spezielle Eigenschaft ist die Verwendung eines LCD-Bildschirms⁷. Da werden z.B. Errors oder die Login-Daten angezeigt. Es ist für den Benutzer einfach zu erkennen, wenn es ein Problem mit dem System gibt. Eigentlich ist die Hauptfunktion des LCD-Bildschirms, den Input des Benutzers anzuzeigen. Warum es so geplant ist? Das Problem besteht darin, dass, wenn der Benutzer seine Email-Adresse schreibt, Fehler passieren können, weil er nicht sieht, was er schreibt. Um das zu vermeiden, wird der LCD-Bildschirm verwendet. Der Benutzer kann sehen, was er schreibt.

Um registrierte Personen mit ihren Gesichtsdaten zu speichern, braucht das System einen Server. Auf diesem Server läuft ein Datenbank Management System. Die Datenbank ist so konfiguriert, dass die Person mit ihren Informationen gespeichert werden können.

Das System hat auch einen Backup-Server. Die Daten werden parallel bei dem Hauptserver sowie bei dem Backup-Server gespeichert, damit die Daten noch verfügbar sind, wenn der Hauptserver ein Problem hat. Die Verwendung des Backup-Servers ist zustande gekommen, weil das System 24/7 arbeiten muss. Wenn der Hauptserver Wartungen oder Probleme hat, kann der Backup-Server die Arbeit übernehmen. Auf Abb. 3.1 wird gezeigt, wie der Gesichtsregistrierung-Teil arbeitet.

⁵Light-emitting Diode, fließt Strom durch, strahlt sie Licht aus

⁶rot, grün, blau LED

⁷Liquid Crystal Display, präsentiert die elektrischen Signale in Form von visuellen Bildern.

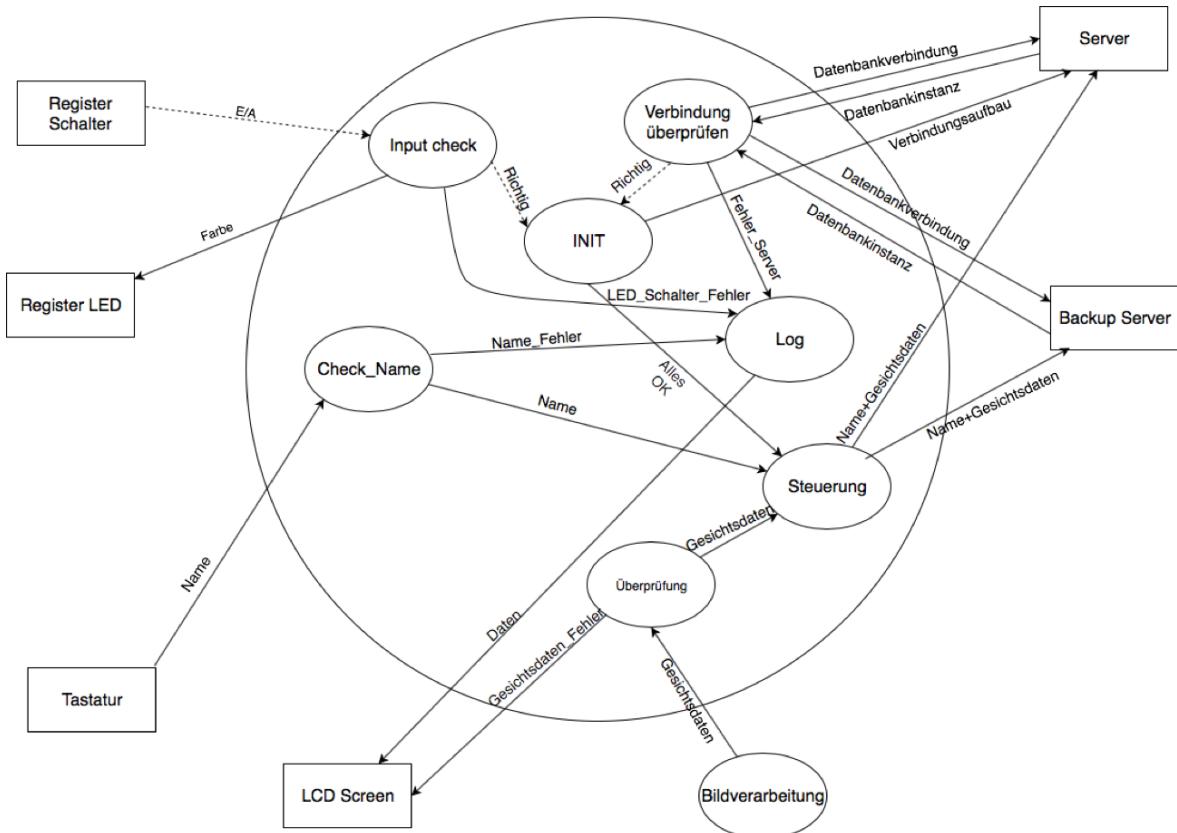


Abbildung 3.1: Structured Software Design bzw. erste Ebene

Das System hat verschiedene Terminatoren. Die Terminatoren sind im Kapitel 3.1.3 erklärt. Auf Abbildung 3.1 können 6 Terminatoren identifiziert werden:

- Register Schalter - Externer Trigger Schalter zum Starten des Registrierungsteils
- Register LED - Anzeiger
- Tastatur - Input Hardware für die Datenerfassung(Name,Email,Rolle)
- LCD Screen - Anzeiger
- Server bzw. Backup Server - eigener Server für die Speicherung der Daten

Auf der erste Ebene können 7 Hauptprozesse identifiziert werden:

- Input check - Kontrolliert in einer unendlichen Schleife, ob der Register Schalter gedrückt wird.
- Check_Name - Kontrolliert, ob der Name in einem richtigen Format (Nur Buchstaben) eingegeben wird.
- Verbindung überprüfen - Überprüft, ob die Server im Betrieb sind und, ob die Verbindung richtig aufgebaut ist
- Log - Speichert die Fehler
- Steuerung - Übernimmt die zentrale Steuerung der Anwendung
- INIT - Initialisiert das ganze System
- Überprüfung - Die Gesichtsdaten bzw. Gesichtspunkte werden kontrolliert und entweder bei der Steuerung weitergeschickt oder auf dem LCD Screen angezeigt, wenn einen Fehler auftritt.

Um Schalter und LEDs im System verwenden zu können, brauchen wir ein spezielles Paket namens "RPi.GPIO". Dieses Paket macht es möglich, den Raspberry PI mit Hardware (LED und Schalter) verbinden zu können. Dafür werden GPIOs verwendet. Der Schalter hat 3 Anschlüsse. Einer wird mit 5V verbunden, der andere mit Ground und der dritte ist für Daten. Dies wird dann mit einem GPIO-Port in Raspberry PI verbunden. Das gleiche gilt auch für die LED, damit sie vom Raspberry PI kontrolliert verwendet kann, wird sie mit einem GPIO-Port verbunden. Mithilfe dieser GPIO-Ports bekommt das System eine Information, wenn der Schalter gedrückt wird. Wert "1" bedeutet, dass der Schalter gedrückt wird und die verschiedenen Skripten aufgerufen werden.

Schritte:

1. Am Beginn des Skripts wird diese Zeile schreiben: "#!/usr/bin/python". Es gibt zwei Gründe, warum diese Zeile geschrieben wird. Der erste Grund ist, dass dieses Program mit einem Python-Interpreter ausgeführt wird, der zweite ist, der Inhalt der Datei wird von der Python-Binärdatei unter /usr/bin/python interpretiert

2. Alle Paketen importieren. Sehen Sie Listing. 3.1

```

1 import RPi.GPIO as GPIO
2 import time
3 import os
4 import subprocess
5 import sys
6

```

Listing 3.1: Packages zu importieren

- RPi.GPIO ist ein Paket, das verwendet wird, um Zugriff auf die sogenannten GPIO-Ports⁸ zu haben. Vorher habe ich erwähnt, wenn wir Zugriff auf die HW-Komponenten haben wollen, die mit Raspberry PI verbunden sind, brauchen wir die GPIO-Ports. Um diese GPIO-Ports in Python zu verwenden, brauchen wir das sogenannte Paket "RPi.GPIO". Es gibt verschiedene Pakete, die einen Zugriff zwischen GPIO-Ports und Python ermöglichen, wie z.B. rpi.gpio, GPIOZero usw. Es wird das rpi.gpio Paket verwendet, weil es leicht verständlich, programmierfreundlich und einfach zu verwenden ist. [42]
- time ist ein Paket in Python. Von diesem Paket wird nur die Funktion "sleep" verwendet. Diese Funktion pausiert das python-Programm. [31]
- os ist ein wichtiges Paket in unserem Skript. Es erlaubt uns, dass wir in einem Python-Skript andere Skripte aufrufen können. Es ist egal, in welcher Programmiersprache diese Skripten geschrieben sind. Es gibt auch verschiedene Methoden, wie man verschiedene Skripte in einem Python-Skript

⁸Eingehende und Ausgehende digitale Signale, am Eingangsport können verwendet werden, um digitalen Messwerte mitzuteilen, die von Sensoren empfangen werden

aufrufen kann. Man kann mit dem subprocess Paket, eine Main-Funktion in dem Skript starten und die verschiedenen Funktionen des anderen Skripts aufrufen.

- Das “subprocess” Paket dient zur Verbindung zwischen verschiedenen Prozessen, in meinem Fall, den Aufruf eines Skriptes.
- Das sys Paket wird verwendet, um Console Parameter zu bekommen. Das bedeutet, wenn das Skript aufgerufen wird, wie z.B. login.py dann kann man login.py einen Parameter mitgeben: login.py <parameter>

3. GPIO-Ports Datenrichtung einrichten. Datenrichtung für LED ist “out”, weil das LED als ein Output für unseres System dient. Datenrichtung für Schalter ist “in”, weil der Schalter als ein Input für unseres System dient. Auf Listing. 3.2 ist der Python-Code.

```

1  GPIO.setmode(GPIO.BCM)
2  GPIO.setup(23,GPIO.OUT)
3  GPIO.setup(18,GPIO.IN)
4  GPIO.setup(17,GPIO.IN)
5  GPIO.setup(27,GPIO.OUT)
6

```

Listing 3.2: GPIO-Ports Konfiguration

Es gibt verschiedene Betriebsarten für GPIO wie BCM und Board. Ich verwende BCM⁹, weil ich das Paket RPi.GPIO verwende. Mit diesem Paket darf nur die Betriebssart ”BCM ” verwendet werden. [32] Für die Registrierung der Schüler und Schülerinnen bzw. Lehrer und Lehrerinnen ist es nötig, dass der Administrator sich bei dem System einloggt. Die Überprüfung, ob der Administrator da ist oder nicht, wird mit einem Vergleich von zwei Bildern gemacht. Ein Bild von Administrator ist gespeichert, das andere wird gemacht. Dazu wird das Skript, das Bilder macht, aufgerufen, und dann vergleicht man es mit einem Skript mit dem anderen Bildern. Es gibt “matched” zurück, wenn die Gesichter bei den beiden Bildern übereinstimmen und ”not matched” wenn die Gesichter nicht übereinstimmen.

4. Dann kommt der Teil “Input check”. Hier verwende ich dann die Methode “input”. Die Methode befindet sich im Paket “RPi.GPIO” und gibt entweder true oder false zurück. Im Verzeichnis /sys/class/gpio/gpio<GPIO-PORT>” gibt es zwei Dateien, value and direction. Direction für den Port des Schalters ist IN und

⁹Broadcom Pin Number

für den Port des LEDs ist OUT. Mit der Methode “input” hole ich den Wert (value) der Schalter-Port. Wenn der Schalter gedrückt wird, wird der Wert “1” herausgekommen , 1 repräsentiert “true”. Das bedeutet, dass die Input-methode “true” zurückliefert und das Programm läuft weiter.

Nachdem der Schalter gedrückt wird, wird ein Skript aufgerufen. Dieses Skript dient zur Registrierung der Person in der Datenbank. Für die Registrierung der Schüler und Schülerinnen bzw. Lehrer und Lehrerinnen ist es nötig, dass der Administrator sich bei dem System einloggt. Ist der Administrator da, können die Personen mit der Registrierung beginnen. Diese Person wird nach ihrem Vornamen, Nachnamen, Email und Rolle gefragt. Mit der Rolle ist gemeint, die Funktion bzw. die Stelle dieser Person in der Schule, ob die Person ein Schüler, Lehrer oder ein Administrator ist. Die Rolle wird vom Administrator eingegeben. 1 steht für Administrator, 2 für Schüler und 3 für Lehrer und Lehrerinnen. Sie wird eigentlich nur für die Verwaltung des Systems verwendet. Wenn der Administrator wissen möchte, wie viele Lehrer oder Schüler schon registriert sind, oder ob es noch nicht registrierte Personen gibt, dann schaut er in der Datenbank nach. Die E-Mail speichern wir dann in einer Variable und diese Variable übergeben wir dann an ein anderes Skript. Dieses Skript erstellt mit der Kamera eine Verbindung und macht ein Bild. Der Name des Bildes ist gleich mit der Email der Person. Es ist so gewählt, weil es für das Einfügen der Daten in der Datenbank und bei der Speicherung des Paths des Bildes in der Datenbank mit dem gleichen Namen wie die E-Mail einfacher ist.

Es wird die E-Mail verwendet, weil es eine performantere Suche in der Datenbank ermöglicht. Die E-Mail ist einzigartig, nur einmal für jede Person. Von 1000 Datensätze wird nur der Datensatz ausgegeben, in dem die E-Mail mit der eingetippten E-Mail vom Benutzer übereinstimmt.

In der Datenbank gibt es nur eine Tabelle. In dieser Tabelle wird alles gespeichert: Name, E-Mail, Rolle, Path des Bildes und alle Gesichtspunkte. Für die Registrierung der Personen in der Datenbank werden zwei Schritte durchgeführt. Der erste Schritt ist das Einfügen der neuen Person mit ihrem Vornamen, Nachnamen, E-Mail und Rolle in der Datenbank. Der zweite Schritt ist das Update der Gesichtspunkte. Es wird ein Bild gemacht, von diesem Bild werden die Gesichtspunkte extrahiert. Danach werden diese Gesichtspunkte in den zuständigen Spalten der Datenbank gespeichert. Dieser Schritt wird durch eine Update-Anweisung gemacht. Nachdem die Update-Anweisung erfolgreich ist und die Person mit ihrem Vornamen, Nachnamen, E-Mail, Rolle, und ihren Gesichtspunkten gespeichert ist, wird die LED grün angesteuert. Wenn ein Fehler auftritt, wird die LED rot angesteuert. Es gibt einige Fehler, die bei der Registrierung auftreten können:

- (a) Auf dem Bild gibt es kein Gesicht oder mehr als ein Gesicht.
- (b) Das Gesicht ist schlecht positioniert und es ist nicht möglich, die Gesichtspunkte zu extrahieren.
- (c) Die Person ist schon registriert.

Wenn ein von den oben erwähnten Fehler auftritt, wird das auf dem Bildschirm angezeigt und die LED wird rot angesteuert.

3.1.4 Herausforderungen

Das Projekt hatte für mich viele Herausforderungen. Das lag daran, dass es ein ziemlich großes Projekt war, und ich neue Technologien verwendet habe, die ich vorher nie verwendet hatte. Ich habe z.B. keine Erfahrung mit OpenCV, Python und andere Dinge, die ich später erwähnen werde. Ich habe von diesen Herausforderungen und Problemen viel gelernt. Einerseits bin ich froh, andererseits bin ich frustriert, weil dies das Enddatum des Projekts verzögert hat. Die Herausforderungen waren:

- **opencv zu installieren.** Es hat 3 Woche gedauert, bis ich es installiert hatte.
- **Beginn des Projektes.** Der Beginn eines Projektes ist immer schwierig. Die Koordination im Team war sehr schwierig. Ich, als Projektleiter, musste allen sagen, wie sie arbeiten sollen, wo sie die Dateien finden können usw. Das war die größte Herausforderung.
- **Git repository, Einrichtung von git.** Manche Teammitglieder wussten sehr wenig von git und ich musste es ihnen erklären. Manchmal gab es merge conflicts, weil sie pull gemacht haben, ohne dass Sie die Änderungen committed haben. Ich sollte dies lösen, weil ich mehr Erfahrung mit git hatte.
- **Python als Programmiersprache.** Wir wollten vorher mit C++ arbeiten, aber es war sehr schwierig, OpenCV in Visual Studio zu installieren. Manche von uns wollten in Windows arbeiten und der einzige Weg war, mit Visual Studio zu arbeiten. Es hat nicht funktioniert, deshalb sind wir zu Python gewechselt. Wir haben Python gewählt, weil opencv in Python sehr einfach installierbar ist. Mit Python hatten wir keine große Erfahrung. Alles, was ich mit Python gemacht habe, ist eine Verbindung mit der Datenbank und Aufbau der SQL-Abfragen (Insert,Select,Update,Delete). Alle andere Wissen sollte ich selbst von Bücher, Internet, Tutorials lernen. Die große Herausforderung hier war, die richtigen Quellen zu finden.
- **Verwendung der Kamera und verbinden mit Python.** Ich wusste nicht, welche Funktionen man verwendet, um die Verbindung mit der Kamera zu erstellen.
- Bei der älteren Version von **Raspbian** heißt das Paket, das python mit dem Datenbank Management System(MySQL) verbindet, 'python-mysqldb' und jetzt heißt es 'python-mariadb'. Ich wusste das nicht und dies hat mir etwas Zeit gekostet.
- **Abhängigkeiten zwischen einzelnen Arbeitsteilen.** Die Aufgaben waren so geteilt, dass es Abhängigkeiten zwischen ihnen gab. Das hat dann zu einer Verspätung der Projektabgabe geführt, weil die Teammitglieder aufeinander warten mussten.
- Die großen Teile meiner **Planung** haben gepasst, nur wenige Kleinigkeiten musste ich ändern. Sie sind erst in der Implementierungsphase angezeigt.

- Problem mit dem Zugriff auf die Elementen eines **Numpy-Arrays**.
- **Verbindung der Skripte miteinander** und Verknüpfung der Variablen, die sich in verschiedenen Skripten befinden.
- **Aufteilen der Gesichtspunkte in zwei Arrays**. In einem Array nur die X-Werte und in dem anderen Array nur die Y-Werte.
- **Die Aufgabenteilung** zwischen den Teammitgliedern war unausgewogen.

Ich habe diese Lösungen für die Herausforderungen gefunden:

1. Ich habe viel Tutorials geschaut, Websites gelesen, wie opencv in Raspberry PI installiert werden kann. Ich habe viele verschiedene Methoden ausprobiert, aber keine guten Ergebnisse erzielt. Nach vielen Versuchen wurde es erfolgreich installiert. Dannach habe ich verschiedene Skripte in Python gemacht, um es zu testen. Manche der Skripte haben funktioniert, manche nicht. Eine kleine Herausforderung war für mich, dass ich die Skripte, die nicht richtig funktioniert haben verbessert. Anschließend habe ich herausgefunden, dass das Problem bei dem Kompilieren von opencv aufgetreten sind (cmake). Ich habe es noch einmal vom Beginn kompiliert. Das hat das Problem gelöst, alle Skripte funktionieren, es gibt keine Fehler mehr, die mit dem opencv Paket zu tun haben.
2. Ein Treffen mit meiner Gruppe vor dem Beginn des Projektes war notwendig. Ich habe Ihnen gesagt und erklärt, in welchen Verzeichnise sie arbeiten sollten, die Struktur der Dokumentation, welcher Kommunikationskanal verwenden wir, um Probleme, Herausforderungen usw. zu besprechen. Jede Person hatte ihre Vorschläge, um Aufgaben zu lösen, und dieses Treffen hat sehr lange gedauert, bis alle verstanden hatten, wie, wo, was, wann gemacht werden soll. Aber auch nach dem Treffen gab es zwischendurch Missverständisse bzw. Probleme mit der Kommunikation, z.B. wurde nicht im richtigen Verzeichnis gearbeitet usw.
3. Ein Git-Repository zu erstellen und einzurichten war einfach. Ich hab es online in github.com erstellt, einen Name eingegeben und dann die anderen Teammitglieder als Collaborators hinzugefügt. Um strukturierter zu arbeiten, habe ich verschiedene Branches angelegt. Es gab mit dem Befehl “push” und “pull” Probleme. Das habe ich gelöst, indem ich allen Teammitgliedern gesagt habe, dass, wenn sie in einem Github-Repository arbeiten möchten vor dem Beginn der Arbeit einen “pull” machen müssen, damit die Änderungen, die von anderen in dem Repository gemacht wurden, mit der Version am Computer synchronisiert wird. Sie wussten nie, was die anderen in diesem Repository gemacht haben. Sie machen “push”, ohne zu sagen, dass sie einen “push” gemacht haben. Das führt dann zu merge-Probleme usw.
4. Ich habe jedem Teammitglieder gesagt, er muss mindestens zwei Wochen mit dem Lernen von Python verbringen. Tutorials ansehen, Beispiele selbst probieren, die Quellen dafür selbst finden.

5. Für die Verbindung der Kamera mit OpenCV, gibt es einen Skript in der offiziellen Website-Dokumentation von OpenCV. Da habe ich alle Funktionen gesucht und gefunden, die ich brauchte, um die Kamera in Python verwenden zu können.
6. Um die Abhängigkeiten zu minimieren, hat jedes Teammitglied andere Aufgaben bekommen, als die, die in der Dokumentation stehen. Ich war gezwungen, diese Änderung zu machen, sonst hätte das Projekt viel länger gedauert.
7. Bei der Implementierung sind Kleinigkeiten herausgekommen, die bei der Planung nicht berücksichtigt waren. Die habe ich direkt in der Implementierung verbessert, ohne dass ich noch einmal die Planung machte. Aber ich habe diese Kleinigkeiten zur Kenntnis genommen, damit ich in der Zukunft keinen solchen Fehler(Kleinigkeiten) mehr in der Planungsphase machen werde.
8. Ich habe die Dokumentation von Numpy Array nicht gut gelesen. Ich bat meinem Betreuer um Hilfe, weil ein Problem mir viel Zeit genommen hat. Der Betreuer hat mir dann geholfen, die Lösung zu finden.
9. Alle Teammitglieder haben ihre Skripts erstellt. Es war meine Aufgabe, alle diese Skripts mit einem Prototyp-Skript zu verbinden. Es war keine wissenschaftliche Arbeit, man musste viel testen, während man den Prototyp erstellte. Nach vielen Tests, habe ich erfolgreich einen Prototyp erstellt.
10. Eigentlich sollte Egli die Gesichtspunkte in ein Array speichern. Sie konnte dieses Ziel nicht erfolgreich abschließen und ich sollte ihr helfen. Zuerst wollte ich alle Gesichtspunkte in ein 2D Array speichern. Nach vielen Versuchen kam ich zu dem Schluss, dass es schwierig war und viel Zeit kosten würde. Stattdessen habe ich die Gesichtspunkte in zwei Arrays gespeichert. In einem Array habe ich nur die x-Werte gespeichert und in dem anderen Array habe ich nur die y-Werte gespeichert. Das war eigentlich die einzige Möglichkeit, wie ich dieses Problem lösen konnte.
11. Am Beginn der Diplomarbeit hat das Team eine Aufgabenteilung gemacht. Mit der Zeit haben die Teammitglieder bemerkt, dass nicht jedes Teammitglied die gleiche Menge an Aufgaben hat. Dannach haben wir uns dazu entschlossen, eine neue Aufgabenteilung zu machen. Zuerst hatte ich auch den 2D vs 3D Unterschied zu realisieren. Mit der neuen Aufgabenteilung ist es ein Ziel von Jordi und nicht mehr ein Ziel von mir.

3.1.5 Qualitätssicherung und Controlling

Risiko ist meistens eine Einschätzung, was es einem Unternehmen kostet, wenn die Projektziele nicht erreicht werden. Ich, als Projektleiter, musste das machen. Eine Risikoanalyse zu erstellen ist sehr schwierig, weil es mit der Zukunft zu tun hat. Zuerst musste ich an die Zukunft denken, bei welchen Bauteilen können Fehler auftreten, welche Programme können ausfallen. Das bedeutet, einen Überblick über die Zukunft zu haben und einzuschätzen, was für Fehler und Risiken es geben kann. Dann schätzte ich die Wahrscheinlichkeit ihres Eintretens ab und am Ende die Maßnahmen. Dahinter

versteckt sich eine große Arbeit.[29] Auf Abb. 3.2 wird die Risikoanalyse dargestellt. Wahrscheinlichkeit, Kosten usw. sollen beachtet werden.

Risikoanalyse: Gesichtsregistrierung und Gesichtserkennung

Risikotyp	Nr.	Wahr-sch.	Aus-wirk.	Ampel	Manager	Beschreibung	Behandlung und Kontrolle	Termin / Nächster Schritt
Standardrisiken								
Ressourcen	1	1	8	8	Rei Hoxha	Ausfall von Ressourcen	Ressourcen im Voraus sichern (Reserven an Mitarbeitern, an HW, an Zeit)	Projekt gleich abschließen
Planung	2	4	8	32	Aron Terzeta	Schlechte Planung (verschiedene Eintrittsfälle nicht berücksichtigt)	Nocheinmal mit der Planung beginnen	Den Projektantrag ablehnen
Technik	3	3	3	9	Egli Hasmegaj	Nicht eindeutig Gesichtspunkte-Extraktion	Fertige Skripts aus dem Internet holen	Die Implementierung zu anderen Firmen zulassen
Staat	4	8	7	56	Aron Terzeta	Rechtliche Aspekte (Persönliche Informationen) + Datenschutz	Mit dem Staat vor der Implementierung sprechen	Geld zu Staat bezahlen, damit der Staat nicht in Schwierigkeiten uns bringt
Planung	5	2	4	8	Jordi Zmiani	Mehr Benutzer als geplant (Datenbankdesign)	ein skalierbares Design der Datenbank	nocheinmal Datenbank erstellen
Zeit	6	3	5	15	Alle	Spät mit Arbeit begonnen, z.B. die Installation von OpenCV sollte 4 Stunden aber ist 2 Tage	Schneller dann arbeiten oder vorher dem Kunden sagen, dass es ein bisschen spät das Produkt fertig ist.	Kunden sagen, dass entweder wartet bis zum Ende(eine große Verspätung) oder nimmt das nicht fertige Produkt zurück
Staat	7	4	0	0	Niemand	Wächter wird seinen Job verlieren, weil System viel Know-How braucht	-	-
Kommunikation	8	4	8	32	Alle	Wenn das Team keine gute Beziehung zwischen den Mitgliedern hat	Versuchen, einen gemeinsamen Weg und Sprache zu finden, wenn nicht, neue Teammitglieder	Team wechseln
Technik	9	3	3	9	Rei Hoxha	HW nicht genug, keine Know-How, wie man die speziellen Bauteile verwenden kann	Reserven, Bedienungsanleitungen lesen(auch in Internet suchen)	Experten fragen, Hilfe von Experten bekommen
Technik	10	5	5	25	Alle	Mangelnde Einfahrung	Tutorials sehen	Teile von Algorithmen und Skripts von Internet holen
Technik	11	3	3	9	Aron Terzeta	Nachträgliche Änderungswünsche des Systems	Vorher planen (Skalierbarkeit und Erweiterung)	Nocheinmal Planung
Technik	12	2	3	6	Aron Terzeta	Veränderung am kritischen Weg	Schnell den neuen kritischen Weg finden	Projekt ohne kritischen Weg(sehr gefährlich)
Technik	13	1	3	3	Aron Terzeta	fehlende Terminüberwachung	-	-
Zeit	14	2	2	4	Alle	Zeitprognose unterschätzen	Sagen, dass es eine Verspätung gibt. Der Kunde und der Chef muss es wissen	-
Zeit	15	3	4	12	Jordi Zmiani	Mangelnde Puffer in der Kalkulation	Mehr Stunde daran bis zum Ende arbeiten	Hilfe von außen bekommen
Ressourcen	16	1	1	1	Rei Hoxha	Mangelhafte Kontrolle der Projektkosten	Selbst dann den Mangel bezahlen	-
Ressourcen	17	2	4	8	Rei Hoxha	fehlende Ausrüstung	Direkt mit dem Projektleiter sprechen, und dann er entscheidet, ob es gekauft, ausgeliehen oder ... wird	-
Planung	18	0	1	0	Aron Terzeta	Geringe Personalkapazitäten	Entweder bleiben wir mit diesen Personalkapazität und das Produkt später fertig machen oder neue Personal einstellen, damit das Produkt in time fertig zu machen	-
Ressourcen	19	1	4	4	Alle	Ausfall einzelner Projektglieder	Ein anderer Projektglieder diese Arbeit machen	Einem neuen Team das Projekt einrichten

Abbildung 3.2: Risikoanalyse in Excel

Es gibt eine Priorität bei den Maßnahmen. Je höher die Zahl in der Spalte „Ampel“ ist, desto wichtiger die Maßnahmen und die Risiken sind. Die, in denen die rote Farbe ist, werden zuerst behandelt. Wenn es grün ist, bedeutet es, dass das Risiko eine sehr kleine Rolle auf das System spielt. Sie werden am letztens behandelt, wenn alle andere Maßnahmen für die roten und gelben Risiken getroffen werden. Spalte “Ampel”:

- 0 : das Risiko spielt keine Rolle oder eine sehr kleine Rolle. Die Maßnahmen für diese Risiken können getroffen werden.
- 1-3 : das Risiko hat eine Bedeutung. Die Maßnahmen für diese Risiken sollen getroffen werden.
- 3-100 : das Risiko spielt eine große Rolle auf das System. Die Maßnahmen für diese Risiken müssen getroffen werden.

3.2 Ergebnisse

Die Zeit für den Abschluss der Diplomarbeit ist vorbei. Jetzt sollen alle Ziele erfolgreich abgeschlossen werden. Meine Ziele sind eigentlich erfolgreich abgeschlossen. Eine Person kann sich mit ihrem Vornamen, Nachnamen, E-Mail und Rolle registrieren lassen. Die Gesichtspunkte werden von dem Gesicht dieser Person extrahiert und in der Datenbank zusammen mit Vornamen, Nachnamen, E-Mail und Rolle gespeichert. Das System hat zwei Administratoren. Die Administratoren loggen sich mit einem Passwort ein. Ohne das Einloggen des Administratores kann keine Person registriert werden. Am Beginn war geplant, dass das System ein LCD-Screen enthält. Jetzt geht es sich mit der Zeit nicht aus, dieses LCD-Screen zu programmieren und im System zu implementieren. Statt des LCD-Screens wird einen Bildschirm verwendet. Es wird nun alles auf dem Bildschirm angezeigt. Diese Veränderung hat eigentlich keine Bedeutung für das System.

3.2.1 Implementierung

Nachdem einer umfangreichen und guten Arbeit meinerseits, kann ich einen Prototyp mit einigen Funktionen vorweisen. Die Funktionen, die in diesem Prototyp integriert sind, gehören zu den Zielen, die ich realisieren sollte. Diese Funktionen sind:

- Administrator-Konto. Die Registrierung einer neuen Person geht nicht ohne ein Administrator-Konto. Der Administrator muss bei dem System eingeloggt sein, damit der Registrierungsteil funktioniert. Der Administrator loggt sich mit einem Passwort bei dem System ein.
- Eine Person wird mit ihrem Vornamen, Nachnamen, E-Mail und Rolle in der Datenbank gespeichert
- Ein Bild von einer Person wird gemacht
- Von diesem Bild werden die Gesichtspunkte extrahiert
- Die Gesichtspunkte werden in den Arrays gespeichert.
- Ich hole dann die Werte von diesen Arrays und speichere sie in der Datenbank.

Kapitel 4

Datenbank und 2D vs. 3D Unterschied

In diesem Kapitel wird die Umsetzung von der Datenbank sowie auch die Differenzierung zwischen 3D und 2D Gesichtern besprochen. Es wird zuerst eine allgemeine Beschreibung gemacht, danach werden allen Schritten von der Plan-Phase bis zu der Implementierungsphase besprochen. Alle Probleme, Herausforderungen und Ergebnisse dieses Teils des Projektes werden genau erklärt.

4.1 Umsetzung

In der gezeigt Tabelle 4.1 werden alle Technologien, die mit diesem Teilbereich des Projekts zu tun haben. Zuerst wird eine allgemeine Erklärung der Technologie gemacht, und danach werden mehr Details beschrieben, wie z.B die verwendeten Bibliotheken bzw. wieso diese gewählt wurden. Diese Beschreibung ist für die Datenbank und den 2D vs. 3D Unterschied.

Technologie	Beschreibung	Lizenz
MySQL	Datenbankverwaltungssystem	Kostenlos
MariaDB	Datenbankverwaltungssystem	Kostenlos
Python	Objekte-orientierte Programmiersprache	Kostenlos
OpenCV	Bildverarbeitung Programmbibliothek	Kostenlos

Tabelle 4.1: Technologien

MySQL

"MySQL ist eine CLient/Server Implementierung, die aus einem Server-Dämon mysqld und vielen Client Programmen, sowie Bibliotheken für PERL, PHP/3, PHP/4 sowie ASP besteht. SQL ist eine standardisierte Datenbanksprache, die das Speichern, Updaten und den Zugriff auf Informationen erleichtert. Beispielsweise kann man Produktinformationen eines Kunden auf einem WWW-Server speichern und abrufen. MySQL ist



Abbildung 4.1: MySQL Logo [2]

„äußerst schnell und flexibel genug, um sogar Bilder und Log-Dateien darin abzulegen. In der Praxis ist MySQL sehr viel schneller, als z.B. ORACLE oder INFORMIX.“[39]

MySQL hat ein breites Anwendungsspektrum und wird meistens in Verbindung mit PHP, Linux, Python usw. verwendet. Der Grund warum MySQL verwendet wird, ist die einfache Bedienung und Verwaltung dieser Datenbank. Da es sich bei diesem DBS um eines der am häufigsten verwendeten DBS handelt, gibt es eine Vielzahl von Tools, die zum Verwalten der Datenbank verwendet werden können.

MariaDB

„MariaDB ist aktuell die am schnellsten wachsende Open-Source-Datenbanklösung. Sie wird hauptsächlich von der MariaDB Corporation entwickelt und ist ein Fork von MySQL. Mittlerweile bietet das Datenbankverwaltungssystem mit seinen diversen kostenfreien Features vieles, was MySQL nicht oder nur kostenpflichtig zur Verfügung stellt (z.B. eine Speicher-Engine zur performanten Verarbeitung von riesigen Datenmengen; ein Datenbank-Proxy zur sicheren und hoch-verfügbaren Verwaltung skalierbarer Installationen u.v.m.). Im Gegensatz zu MySQL verfügt MariaDB jedoch nicht über einen eigenen Client wie die Workbench. Eine gute kostenfreie Alternative stellt HeidiSQL dar, jedoch verfügt diese über kein Dashboard, welches z.B. die Funktionsweise des Servers darstellt und damit Optimierungsentscheidungen erleichtert.“[28]



Abbildung 4.2: MariaDB Logo [3]

Es ist hauptsächlich für Linux ausgelegt. Es bietet eine einfache Verbindung mit dem Linux-Betriebssystem so wie auch mit Python, C ++ und vielen anderen Programmiersprachen. Da dies das Standard-Linux-DBS ist und unsere Software unter Linux mit Python-Skripten geschrieben wurde, ist MariaDB das DBS unserer Wahl.

Python

Python ist eine Programmiersprache, die dem User mit seiner einfachen Syntax hilft. Für die meisten Betriebssysteme steht Python frei zur Verfügung. Alles was mit Objektorientierung zu tun hat, wird in Python unterstützt.[36]



Abbildung 4.3: Python Logo [4]

Der Grund, warum die Software in Python geschrieben wurde, liegt an der einfachen und schnellen Möglichkeit, um Datenbankskripte und Gesichtserkennungsprogramme zu erstellen.

OpenCV

OpenCV (Open Source Computer Vision Library) ist eine Open-Source-Bibliothek für Machine Learning Programme und Computer Vision Software. Der Grund zur Entwicklung von OpenCV ist, eine Möglichkeit anzubieten, wo Objekte analysiert und erkannt bzw. identifiziert werden. Der Code von OpenCv ist leicht nutzbar und änderbar, weil es als Berkeley Software Distribution (BSD) Produkt lizenziert ist. Es ist für Windows, Linux, Android und Mac verfügbar.[20]



Abbildung 4.4: OpenCV Logo [5]

Obwohl es in C ++ geschrieben wurde, bietet es viele Schnittstellen zu anderen Programmiersprachen wie Python, die in diesem Projekt benutzt werden. OpenCV ist eine plattformübergreifende Bibliothek und da es viele Deep-Learning-Algorithmen und Frameworks enthält, ist unsere Wahl für die Objekterkennung.

4.2 Technische Lösungen

In diesem Teilbereich werden die Technischen Lösungen bzw. die Planung und Implementierung der verschiedenen Aufgaben der Datenbank und Gesicht-Spoofing Erkennung erklärt.

4.2.1 Datenbank

Wie bereits erwähnt, haben wir uns für MySQL und MariaDB als DBS entschieden, wobei meistens das zweite zum Einsatz kommt. Es ist wichtig zu beachten, dass die Datenbank nicht über die Befehlszeile oder die MySQL/MariaDB-Konsole angesprochen wird, sondern mithilfe von Python-Skripten. In diesem Fall ist jeder gegebene Befehl in den Python-Dateien gespeichert, und wenn etwas schief ging, konnten die Skripte einfach verbessert und Fehler behoben werden.

Um eine Datenbank zur Gesichtserkennung und -identifikation zu entwerfen, mussten einige Nachforschungen angestellt werden, bei denen es hauptsächlich darum ging Beispiele für vorhandene Datenbanken zu finden. Auf diese Weise konnten wir uns eine Vorstellung davon machen, wie Datenbanken aussehen. Obwohl die meisten Datenbanken nicht kostenlos waren und einige Unterlagen unterschrieben werden mussten, um die vollständigen Details zu erhalten, war dies mehr als ausreichend, um einen Ausgangspunkt zu haben. Einige der gefundenen Datenbanken sind:

- 3D Mask Attack Dataset[6]
- The AR Face Database, The Ohio State University[7]
- Caltech Faces[8]
- CAS-PEAL Face Database[9]
- The Color FERET Database, USA[10]

Auf den ersten Blick scheint es nicht sehr schwierig zu sein, eine Gesichtserkennungsdatenbank zu erstellen, aber da sich das Projekt entwickelte, mussten viele Änderungen vorgenommen werden, da sie an die Gesichtsdaten der Person weitergegeben wurden.

Der nächste Schritt bestand darin nachzuforschen, wie in Python die Schnittstelle mit MariaDB aussieht. Da von vielen Linux-Benutzern empfohlen wird, Python für die Verbindung zu MariaDB zu wählen, wurde der Programmierer aufgefordert, den Code mithilfe von Methoden und try and except-Anweisungen zu schreiben, um saubere Skripte zu erstellen. Auf diese Weise würde der Code sehr einfach zu analysieren und zu verbessern sein, damit die Fehlermeldungen so schnell wie möglich beseitigen zu können.

Beschreibung der Tabellen

Nach der Anfangsplanung wurden die folgenden zwei Tabellen (Bild 4.5) definiert:

- Person
- Log

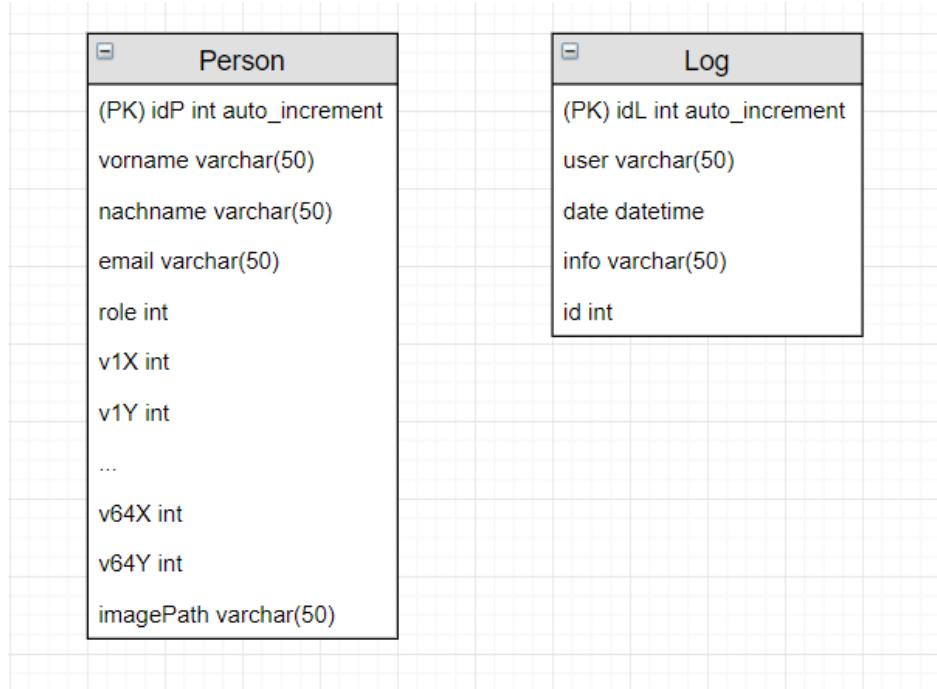


Abbildung 4.5: Datenbank - RDM

Tabelle Person

Die erste Tabelle, Person, enthält alle User mit ihren grundlegenden Informationen, wie Vorname, Nachname, Rolle, Email, Gesichtspunkten sowie auch der Pfad von dem Bild. Die Benutzer des Systems werden in zwei Gruppen, Admin und normaler User, geteilt. Damit es einen Unterschied zwischen diesen beiden Gruppen gibt, gibt es die Spalte Rolle. Diese Spalte bekommt als Wert die Zahl 1, wenn der Person ein Admin ist, und die Zahl 0 für einen normalen Users.

Die Spalte idP ist eine eindeutige Spalte, die es für die Identifizierung der Tupelzeile, deshalb wird sie als Primär Schlüssel definiert. Sie wird auch auto increment definiert, damit die Tupel automatisch nummeriert werden.

Die Spalte Nachname enthält den Nachname der Person, deshalb wird sie als Datentyp Varchar mit der Länge 50 gespeichert, weil Varchar zum Speichern von Strings geeignet ist.

Die Spalte Email enthält die Email der Person, deshalb wird als Datentyp Varchar mit der Länge 50 gespeichert, weil Varchar zum Speichern von Strings geeignet ist.

Die Spalte Rolle enthält die Rolle der Person, deshalb wird als Datentyp Char mit der Länge 1 gespeichert, weil Char zum Speichern von Strings geeignet ist. Um eine einfache Nummer zu speichern, ist hier auch den Datentyp Int verwendbar.

Jeder Punkt besteht aus einer X und Y Koordinate, und von jedem Bild gibt es einen Startpunkt, wo die anderen Punkten relativ zu diesem Punkt gespeichert werden. Für diese Punkte wird den Datentyp decimal werden, weil die Koordinaten Kommazahlen sind. Insgesamt werden von einem Bild 64 Gesichtspunkten gespeichert.

Die Spalte imagePath speichert den Pfad der Bilder und wird mit dem Datentyp Varchar mit einer Länge von 50 gespeichert, weil Varchar zum Speichern von Strings geeignet ist. Die Bilder werden mit ihrem Pfad und nicht als ganze Bilder gespeichert, weil wenn diese Bilder in der Datenbank gespeichert werden, würde die Datenbank mit Daten überfüllt werden. Zum speichern von ganzem Bilder wird Medium Blob verwendet, aber dieser Datentyp braucht zu viel Speicherplatz von unserer Datenbank. Der RaspberryPi ist nicht dafür geeignet große Bild-Datenbanken zu speichern und bietet wenig Speicherplatz, deshalb könnte es zu einem Overflow der Datenbank kommen.

Log Tabelle

Die zweite Tabelle ist eine Tabelle, in der alle Aktivitäten protokolliert werden. In dieser Tabelle sollten die Grundinformationen gespeichert werden, wie z.B wer hat was bzw. wann gemacht. Diese Frage muss die Tabelle beantworten.

Die Spalte idL ist eine eindeutige Spalte, die zur Identifizierung von Tupel dient, deshalb wird sie auch als Primär Schlüssel definiert. Sie wird auch auto increment definiert, damit die Tupel automatisch nummeriert werden.

In der Spalte User soll die Person gespeichert werden, die die Änderungen durchgeführt hat. Hier wird der Datenbankuser gespeichert, weil um etwas zu verändern muss sich die Person zuerst in der Datenbank einloggen. Die Spalte hat den Datentyp Varchar mit der Länge 50, weil die Funktion Current User, dort eingetragen wird.

Die Spalte Date, wie der Name sagt, speichert die Zeit, wann eine SQL Statement ausgeführt wird. Die Spalte hat den Datentyp datetime, und da wird die Funktion now(), die die aktuelle Zeit darstellt, gespeichert.

Die Spalte Info, informiert den User, was für eine Änderung gemacht wurde, z.B neue User wurde eingelegt. Diese Spalte bekommt den Datentyp Varchar, weil Varchar zum Speichern von Strings geeignet ist. Die Werten in dieser Spalte werden durch Triggers definiert, also wenn etwas in der Datenbank passiert, kümmern die Trigger sich um das Eintragen der Daten.

4.2.2 2D vs. 3D Unterschied

Ein großes Problem von Gesichtserkennungssystemen ist das Gesicht-Spoofing, bei dem ein Hacker bzw. Angreifer versucht das System zu umgehen. Einfache Systeme werden durch ein normales Gesichtsbild hereingelegt. Das Bild wird vor die Kamera gegeben, und die Kamera erkennt diese Person als ob sie wirklich vor die Kamera stehen würde. Es gibt ein paar Antispoofing Methoden, z.B wie:

- 3D Depth Perception
- Blinzeln
- Deep Learning
- Liveness Detection

Lösungsweg

In diesem Projekt wurde als Antispoofing Methode die Blinzeln Methode verwendet. Bei dieser Methode wird, wenn eine reale Person vor der Kamera steht, in einer Zeitspanne von Millisekunden überprüft, ob wirklich eine Person oder ein Bild vor der Kamera ist. Normalerweise blinzelt eine Person 15-20 Mal pro Minute, aber diese Methode erkennt nicht nur das Blinzeln der Person, sondern auch wenn sich die Augen bewegen. Deshalb dauert es nicht so lange zum Erkennen und Identifizieren. Auf dem Bild werden zwei Linien, eine horizontale und eine vertikale, gezeichnet. Wenn sich die Länge dieser Linien mehrmals ändert oder viel zu klein wird, dann wird das als ein „Blinzeln“ erkannt. Wenn ein Bild vor der Kamera ist, sind diese Länge immer gleichbleibend, und so erkennt das Programm den Unterschied zwischen einer reale Person und einem Bild.

Beispiele

Als erstes Beispiel wird eine Person vor der Kamera gezeigt (Bild 4.6), fund in einer sehr kleinen Zeitdauer wird die Nachricht Person detected zurückgeleitet. Diese Nachricht informiert das System, dass vor der Kamera eine reale Person ist (Bild 4.7).



Abbildung 4.6: Reale Person

```
root@kali:~/3D# ./blink1.py
Person detected
root@kali:~/3D#
```

Abbildung 4.7: Ausgabe: Person Detected

Als zweites Beispiel wird ein Bild von einem Kind genommen, und vor der Kamera gehalten (Bild 4.8). Dieses Mal ist das Ergebnis anders, das Programm liefert keine Nachricht zurück. Das Programm läuft weiter bis wirklich eine Person vor der Kamera steht, sonst wartet das System auf eine Rückmeldung (Bild 4.9).



Abbildung 4.8: Bild eines Kindes



Abbildung 4.9: Ausgabe: Nicht Detected

Zeitdauer

Ziel des Systems ist das Gesicht so schnell wie möglich zu erkennen. Deshalb wurden ein paar Testfällen durchgeführt (Bilder 4.10 4.11 4.12 4.13), in denen die Länge dieses Antispoofing Programms getestet wurde. Auf den Bildern sind 3 verschiedene Zeittäufen, aber die wichtigste, ist die reale Zeitdauer. Die reale Zeitdauer misst die Zeit, vom Start des Programms bis zum Ende. Im Durchschnitt, mit dem Einschalten der Kamera, braucht das Programm ca. 4.4 Sekunden um eine reale Person zu erkennen.

```
root@kali:~/3D# time python3 blink1.py
Person detected

real    0m4.490s
user    0m2.966s
sys     0m0.200s
```

Abbildung 4.10: Zeitdauer 1

```
root@kali:~/3D# time python3 blink1.py
Person detected

real    0m4.508s
user    0m3.003s
sys     0m0.190s
```

Abbildung 4.11: Zeitdauer 2

```
root@kali:~/3D# time python3 blink1.py
Person detected

real    0m4.408s
user    0m2.914s
sys     0m0.172s
```

Abbildung 4.12: Zeitdauer 3

```
root@kali:~/3D# time python3 blink1.py
Person detected

real    0m4.372s
user    0m2.885s
sys     0m0.189s
```

Abbildung 4.13: Zeitdauerl 4

4.3 Herausforderungen

Jedes Projekt hat seine Herausforderungen, und wenn etwas schief geht sollten Alternativen gefunden werden. Auch in diesem Projekt gab es solche Probleme, die durch Hilfe von Tutorials und Blogs gelöst wurden.

4.3.1 Datenbank - MariaDB

Die meisten Probleme sind bei der Datenbank aufgetreten, wegen der kleinen Unterschiede zwischen MySQL und MariaDB. Manchmal wurde mit dem Programmierregeln von MySQL gearbeitet, obwohl MariaDB in Verwendung war. Zwischen diese zwei Datenbankverwaltungssystem gibt es kleine Unterschiede, die zu Problemen führen. Die eingetretenen Fehler werden nun besprochen:

ERROR 1452: Cannot add or update a child row: a foreign key constraint fails. Das Problem hier war, dass keine Tabelle geändert werden konnte, wegen der Beziehungen zwischen Tabellen. In Prinzip sollte zuerst die Kind Tabelle gelöscht werden, und dann die Eltern Tabelle, weil sonst gibt es einen Konflikt, weil bei der Kind Tabelle ein Fremdschlüssel definiert ist, auf den die Tabelle verweist.

You have an error in your SQL syntax; check the manual that corresponds to your mariadb server version for the right syntax to use near .. At line 1

Dieser Fehler wurde ausgelöst, wenn ein Trigger Script ausgeführt wurde. Der Fehler ist beim Delimiter aufgetreten, weil wenn ein Trigger durch ein Script erstellt wird, braucht man keine Delimiter, sondern nur wenn es manuell durch die Command Line ausgeführt wird, musste der Delimiter festgelegt werden. Hier kann man die Unterschied zwischen MariaDB und MySQL sehen, in MySQL wird dieser Code ohne Probleme ausgeführt.

Mysql-Server is missing

Dieser Fehler zeigt, dass kein MySQL Server auf dem Betriebssystem installiert ist. Dieses Problem ist eingetreten, wegen Änderungen im Linux System. MySQL Server wurde nicht mehr unterstützt, es sollte mariadb-server durch den Befehl *sudo apt-get install mariadb-server* installiert werden.

ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2 "No such file or directory")

Dass war nur ein kleiner Fehler (Bild 4.14), weil wenn man in Kali Linux programmiert und gearbeitet hat, musste der MariaDB Server immer noch einmal durch den Befehl

service mariadb restart gestartet werden. Im Debian Betriebssystem, das auf RaspberryPi installiert ist, muss der Server nicht gestartet werden, weil er automatisch in Betrieb ist.

```
root@kali:~/Datenbank# mysql
ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/run/
mysqld/mysqld.sock' (2)
```

Abbildung 4.14: Mysql-Server is missing

4.3.2 Disparity Map

Am Begin war geplant, als Antispoofing-Methode die 3D Depth Perception zu implementieren. In dieser Methode werden zwei Kameras gebraucht, wo zwei Bilder gemacht werden, eines von jeder Kamera (Bild 4.15). Dann werden sie verglichen, um eine 3D Objekt zu erstellen. Zuerst wird eine Mischung von beiden Bildern zum Erstellen einer sogenannten Disparity Map gemacht (Bild 4.16). Eine Disparity Map ist ein schwarz-weiß Bild, das die Tiefe der einzelnen Objektpunkten darstellt.



Abbildung 4.15: Linke und Rechte Bild [11]

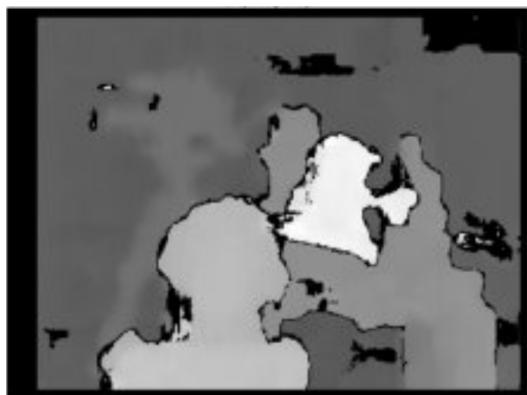


Abbildung 4.16: Beispiel Disparity Map [11]

Daher kann die Disparity Map in ein 3D File umgewandelt werden. Es gibt Programme, wie MatLab, das eine Matplotlib Bibliothek hat, um dieses Disparity Map in einem 3D Graph zu zeigen (Bild 4.17). Wenn man die 3D Koordinaten eines Punktes hat, kann man verschiedene Punkten vergleichen z.B Nasen-Punkte mit Augen-Punkte. So wird gezeigt, dass diese Punkten verschiedene Distanzen von der Kamera haben. Während ein normales Bild 2D ist, erkennt das Programm automatisch, dass vor der Kamera ein Bild und nicht eine reale Person ist.

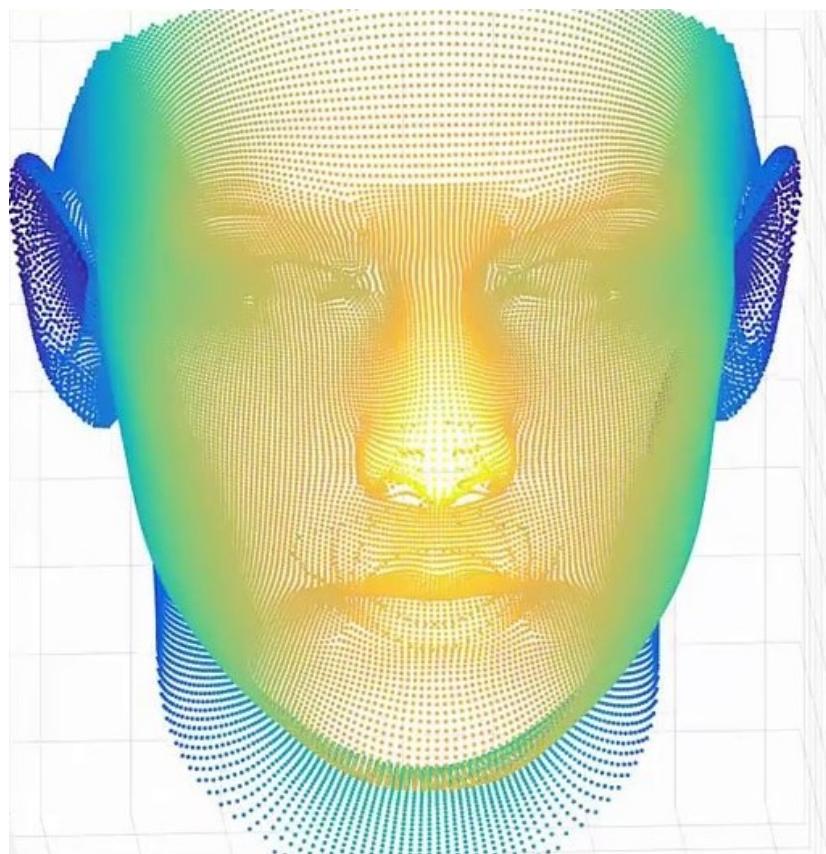


Abbildung 4.17: 3D Gesicht Matlab [12]

Die große Herausforderung war die Recherche und Erstellen von der Disparity Map. Das Tutorial bzw. die Dokumentation waren entweder sehr kurz erklärt und veraltet, oder sehr anspruchsvoll. Nach ein paar Tests vom Erstellen von Disparity Maps war die Erfolgsrate gering, eine von drei Disparity Maps wurde richtig erzeugt.

Aus dem ersten Beispiel wurde eine gute Disparity Map erzeugt (Bild 4.18), und mit dieser Disparity Map kann man bis zur 3D Umwandlung weiterarbeiten.



Abbildung 4.18: Jordi Linke Bild 1



Abbildung 4.19: Jordi Linke Bild 1



Abbildung 4.20: Gute Disparity Map

Aber aus den anderen Bildern wurde keine gute Disparity Map erzeugt (Bild 4.23). Mit dieser Disparity Map war die 3D Umwandlung nicht realisierbar.



Abbildung 4.21: Jordi Linke Bild 2



Abbildung 4.22: Jordi Linke Bild 2



Abbildung 4.23: Schlechte Disparity Map

4.3.3 Python Scripting

Python ist eine beliebte Programmiersprachen von vielen Benutzern, aber Python unterscheidet sich von anderen Programmiersprachen durch die Code Struktur. Die meisten Fehler wurden bei der Übergabe von den Parametern zwischen den verschiedenen Funktionen erzeugt.

TypeError: not all arguments converted during string formatting python

Der folgende Code nimmt als Parameter die Daten von dem neuen User, damit der User in der Datenbank registriert wird. Aber wenn man die Methode nur testen möchte, und als Parameter nur den Namen für eine einfache Registrierung übergibt, dann tritt dieser Fehler ein.

```
1 insertvalues1(curs, 'jordi', 'zmiani', 'jorzmi14@htl-shkoder.com', 1)
```

Bei der ersten Möglichkeit sollten zwei Parameter übergeben werden, denn nur bei einem String Parameter wurde dieser Fehler generiert. Beim der zweiten Möglichkeit musst dieser Parameter durch die Funktion format() gepräst werden.

TypeError: 'str' object is not callable (Python)

Str in Python ist eine spezielle Methode, und diese Methode sollte nicht überschrieben werden, das heißt, dass es keine Variable bzw. Funktion mit diesem Namen erstellt werden sollte. Aber trotzdem wurde dieser Fehler generiert, weil er auftritt, wenn ein Parameter auf einen falschen Datentyp gepräst wird.

4.3.4 OpenCV

OpenCV Bibliotheken können manchmal schwierig zu installieren sein, weil es so viele Möglichkeiten zum Einrichten von OpenCV gibt, die aber von dem Betriebssystem abhängen. In den meisten Fällen, wenn die Installation Fehler produziert, muss OpenCV noch einmal, von Beginn eingerichtet werden.

Unable to locate package libjasper-dev

Dieses Package ermöglicht das Benutzen von ein paar Funktion von OpenCV, das ist eine von vielen Bibliotheken, die eingebunden werden soll. In diesem Fall konnte diese Bibliothek nicht installiert werden, weil sich das Repository geändert hat, und deshalb mussten die Bibliotheken in einer älteren Version heruntergeladen werden. Mit den folgenden Befehlen wurden diese Bibliotheken installiert.

```
1 echo "deb http://us.archive.ubuntu.com/ubuntu/ yakkety universe" |  
    sudo tee -a /etc/apt/sources.list  
2 sudo apt update  
3 sudo apt install libjasper1 libjasper-dev
```

4.4 Qualitätssicherung

”Einer der wohl bekanntesten Berater, Lehrer und Autoren zum Thema Qualität ist der Amerikaner W. Edwards Deming. Deming entwickelte auf der Basis der allgemeinen Problemlösungsmethode den sogenannten Plan-Do-CheckAct-Zyklus (PDCA-Zyklus). Auf der Plan-Phase werden Probleme betrachtet und Lösungsmaßnahmen erarbeitet. Die Do-Phase ist die Phase der Umsetzung bzw. Ausführung der zuvor gefundenen Lösungen. In der Check-Phase wird bewertet, ob die Maßnahmen zum Erfolg geführt haben. Innerhalb der nachfolgenden Act-Phase findet eine Standardisierung der erfolgreichen Maßnahmen statt, die fortan als Basis für weitere Verbesserungen dient.”[1]

Auf dem Bild 4.24 wird der PDCA Zyklus dargestellt. Man sieht, dass diese Qualitätsmethode gut für die Verbesserung von Software und Programmen geeignet ist. Deshalb wurde auch diese Methode gewählt, weil durch Error Checking konnten die Fehler gefunden, analysiert und verbessert werden. Das passt gut zum Team, da es keine Erfahrungen mit bestimmenden Programmiersprachen bzw. Softwareprogrammen gab.

Bei den Herausforderungen (Kapitel 4.3) war klar, dass wenn Probleme erkannt wurden, diese durch Hilfe von Tutorials bzw. Net Blogs gelöst werden. Durch verschiedene Zyklen wird man, wenn etwas nicht funktioniert, das Problem schnell finden und später wird ein neuer Lösungsweg bestimmt. So werden die Risiken so früh wie möglich eliminiert.

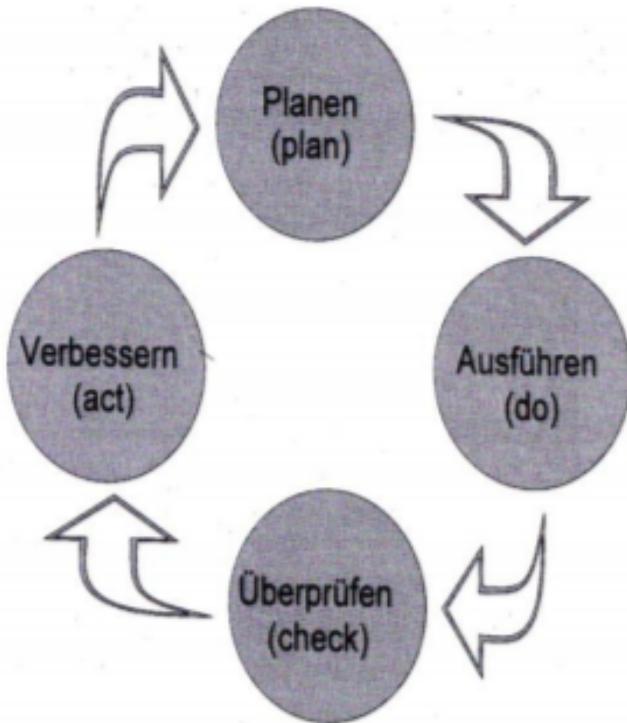


Abbildung 4.24: Plan-Do-Check-Act-Zyklus [1]

4.5 Ergebnisse

In diesem Kapitel geht es um die Evaluierung und Erklären der Ergebnisse bzw. was wurde erreicht.

4.5.1 Datenbank

Für den Bereich Datenbank wurde jede Woche programmiert, seit dem Beginn des Projekts wurden Recherchen über verschiedenen Arten von Gesichtserkennungs-Datenbanken gemacht, und später hat die Planung und das Design der Datenbank begonnen. Es gab viele Änderungen abhängig von den anderen Teilbereichen des Projektes. Die Datenbank musste immer geändert werden, und an die Gesichtsdaten angepasst werden. Die folgenden Punkten der Datenbank wurden realisiert:

- Datenbankdesign
- Erstellen von Tabellen
- Triggers & Prozeduren
- Zugriffsrechte
- Datenbank Backup

4.5.2 2D vs. 3D Unterschied

In diesem Bereich gab es viele Herausforderungen und Änderungen, deshalb ist der Lösungsweg mehrmals geändert oder aktualisiert worden. Zu Beginn war geplant, mit Hilfe von zwei Kameras die Tiefe des Bildes zu finden, und danach mit dieser Tiefe weiterzuarbeiten. Aber dass war schwierig umsetzbar, daher wurde ein anderer Lösungsweg gefunden. Nun wird eine Person mit Hilfe von Blinzeln identifiziert.

4.6 Evaluierung und Resümee

In diesem Kapitel geht es um die Reflexion, Evaluierung der Ergebnisse, sowie auch was von der Planung abgewichen ist.

4.6.1 Planung vs. Realisierung

In diesem Teil gab es viele Änderungen, deshalb ist wichtig eine gute Projektplanung zu haben. Obwohl das Team realistische Ziel festlegt, es kann immer sein, dass die Mitglieder zu Herausforderungen kommen, die ein anderen beeinflussen. Wichtig ist nicht aufzugeben, sondern andere Lösungswegen zu finden, deshalb ist Zeitmanagement von großer Bedeutung.

4.6.2 Wertschöpfung und Lessons Learned

Etwas besonders in dieser Projektarbeit ist, dass man nicht nur im technischen Aspekt lernt, sondern auch wie jemand mit anderen Leuten umgeht. Das Team lernt auch wieso Projektplanen ein sehr wichtiger Teil des Projektes ist, und auch das nicht jedes Ziel realisierbar ist. Deshalb sollten die Ziele so realistisch wie möglich.

Im technischen Bereich waren ein paar Bibliotheken nicht so schwierig zu lernen, man muss ein bisschen Geduld haben. Es gibt nicht immer gute Tutorials zum Erklären von Sachen, sondern es sollte auch die Dokumentation der Bibliotheken Zeile für Zeile gelesen werden.

Kapitel 5

Bildverarbeitung

Diese Diplomarbeit besteht aus vielen unterschiedlichen Modulen, die um bestimmte Ziele bzw. Aufgaben zu lösen gut aufgeteilt sind. Ein sehr wichtiger Teil dieses Projektes ist der Bildverarbeitungsteil. In dem folgenden Abschnitt der Ausarbeitung wird genau erklärt und beschrieben wie diese Aufgabe gelöst wurde und was dafür verwendet wurde.

5.1 Allgemeines

In diesem Allgemeinen Abschnitt werden die verwendeten Technologien und Bibliotheken beschrieben.

5.1.1 Entwicklungsumgebung und Technologien

Die gewählte Entwicklungsumgebung war grundsätzlich eine virtuelle Maschine die Linux auf einem Windows System geboten hat. Linux im Gegensatz zu Windows ermöglicht volle Kontrolle über Updates und Upgrades und dadurch könnten komplexe Aufgaben einfacher erledigt werden. Alles wird leicht und bequem durch die Konsole eingegeben. So wurden die Entwicklung und das Testen der genutzten Algorithmen und andere Technologien erleichtert. [33]

Zusätzlich ist Raspberry Pi als Backup System verwendet worden dies, funktioniert auch mit einem lauffähigen Linux Betriebssystem (Debian). Der Raspberry Pi ist klein, funktioniert aber wie ein normaler Rechner und ist kostengünstig. Für technische/elektronische Projekte wie das Betreffende, ist es perfekt geeignet. Für die Implementierung des Codes wurde hauptsächlich die Programmiersprache Python verwendet. Python ist eine höhere Programmiersprache. Dies bedeutet dass es näher an menschlichen Sprachen ist. Also ist ein in Python geschriebener Code sehr leicht von einem Mensch zu lesen, zu verwalten und zu warten.

Es bietet zahlreiche Module, mit seiner großen und robusten Standardbibliothek an, von denen man beruhigt, abhängig vom Bedarf, auswählen kann. Sehr leicht kann

man in der Dokumentation der Python-Standardbibliothek nachsehen um sich besser mit den Funktionalitäten auszukennen. [38]

”Git ist ein freies und Open Source verteiltes Versionskontrollsysteem, das entwickelt wurde, um alles von kleinen bis zu sehr großen Projekten mit Geschwindigkeit und Effizienz abzuwickeln.”[21]

Die Versionierung der Software Änderungen erfolgte durch Git. Es hat sich als nützlich herausgestellt, da die Arbeit dadurch zwischen die Projektmitgliedern sehr gut koordiniert und verwaltet wurde.

5.1.2 Frameworks und Bibliotheken

Das Schlüsselwort von unserem Projekt war das Framework bzw. die Bibliothek „OpenCV“. OpenCV ist eine open-source Bibliothek für Computer Vision. Ganz allgemein kann sie als eine Bibliothek für Bildverarbeitung betrachtet werden. Sie beschäftigt sich unter anderem mit der Manipulation und Analyse von Bildern, die Analyse von denen und um die Bestimmung von Muster bzw. Objekte, für verschiedenen Zwecken. Ganz berühmte Anwendungsgebieten sind Gesichtserkennung und Stereo Vision. Stereo Vision bedeutet die Extrahierung von Informationen aus einem Bild in eine 3-dimensionale Ebene. OpenCV wurde unter anderen Bibliotheken aufgrund seiner vielen guten Eigenschaften und seiner Flexibilität ausgewählt. Es umfasst hunderte von Computer-Vision-Algorithmen und besteht aus strukturierten Einheiten bzw. Modulen die als feststehende Bibliotheken implementiert sind. Es ist Cross-Platform, in C/C++ geschrieben und unterstützt auch Python. [30]

SciPy hat sich als nützlich, beim Lösen von gewisse mathematische Angaben und bei zusätzliche Installationen von Bibliotheken herausgestellt. SSciPy ist eine kostenlose, Open-Source-Python-Bibliothek für wissenschaftliches und technisches Rechnen.”[44] Dies sind einige der Kernpakete von SciPy die für das Projekt verwendet werden.

NumPy legt die Basis für das wissenschaftliche Rechnen mit Python. Es unterstützt große mehrdimensionale Arrays und Matrizen. Es enthält viele anspruchsvolle Mathematische Funktionen um die Arrays zu bearbeiten. NumPy kann also als leistungsfähiger mehrdimensionaler Behälter für generische Daten verwendet werden. Es können beliebige Datentypen definiert werden.

Dlib ist ein so genanntes Toolkit, das Algorithmen für Machine Learning und Tools zum Erstellen komplexer Software zur Lösung von Probleme der realen Welt beinhaltet. Es wird vielseitig eingesetzt, in Robotik, bei Mobilgeräte und unter anderem in Computer Vision.[35] Es wurde bei der Extrahierung der Gesichtsmerkmale gebraucht. Das Logo von Dlib ist auf das 5.1 ersichtlich.



Abbildung 5.1: dlib logo [35]

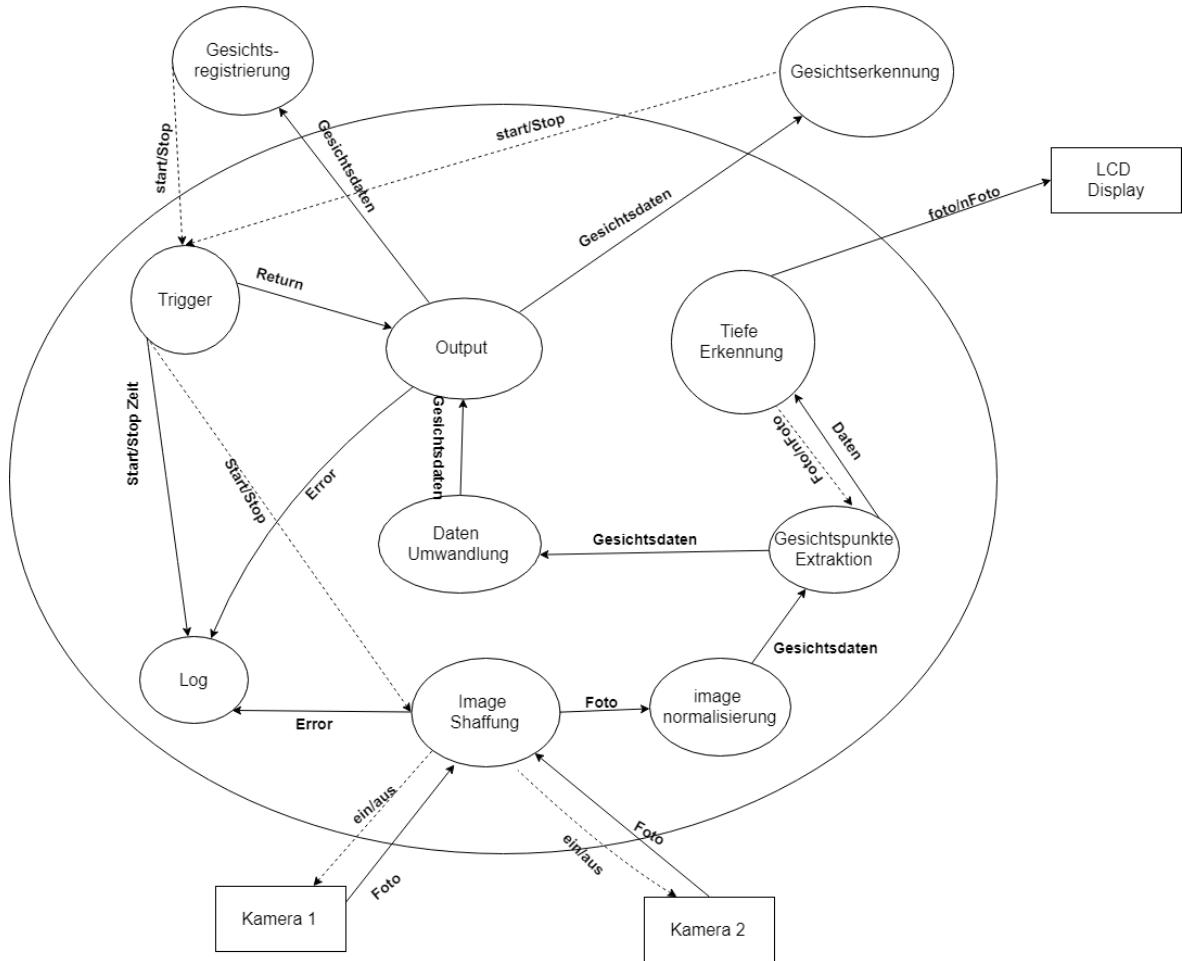


Abbildung 5.2: Bildverarbeitung Structured Design

5.2 Technische Lösungen

Bei der Gesichtsregistrierungs und Gesichtserkennungs Diplomarbeit war es eine schwierige und herausfordernde Aufgabe, sie sorgfältig zu planen, um die Effektivität bei der Arbeit zu steigern und das Endprodukt zufriedenstellend zu machen. Wichtig war es die Ziele richtig zu setzen und sie gut abzugrenzen, damit es zu keinen Lücken bei der Implementierung kommt. Aus diesem Grund musste die Arbeitsteilung gut geregelt werden, damit jedes Teammitglied sich auf bestimmte Modul der Arbeit konzentrieren konnte. Es wurde auch berücksichtigt, dass jedes Teammitglied das machte was ihm/ihr am besten gefällt und was ihm/ihr am leichtesten fällt. Die Planung der betreffenden Bereiche der Projektarbeit ist durch die Methode „Structured Design“ erfolgt. Structured Design ist eine Erweiterung von der „Big Picture“ Methode, die dazu dient, ein technisches System mit ihren Schnittstellen von außen grob zu beschreiben. Es ist in verschiedenen Ebenen unterteilt, von außen beginnend. Nun wird die erste Ebene der Structured Design von der Bildverarbeitungsteil dargestellt.

5.2.1 Lösungsweg - Structed Software design

Legende:

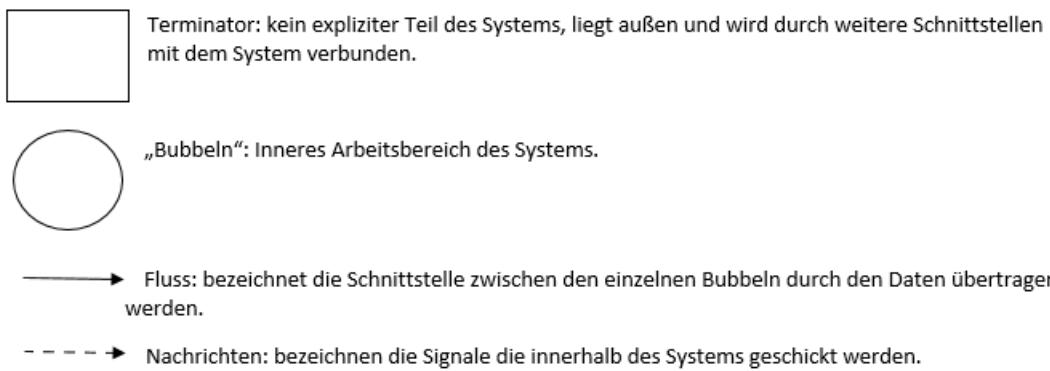


Abbildung 5.3: Erste Ebene Erklärung

Wie es in der Abb.5.2 ersichtlich ist, ist der Bildverarbeitungsteil in diese Einheiten unterteilt:

1. Bildaufnahme
2. Bild Normalisierung
3. Gesichtsschlüsselpunkten Extrahierung
4. Log
5. Output
6. Trigger

Die Schnittstellen nach außen sind das Gesichtserkennungsmodul und das Gesichtsregistrierungsmodul. Diese schicken ein Signal an den Trigger der dann das Bildaufnahmemodul initialisiert. Unabhängig von welchen dieser beiden Schnittstellen die Anfrage kommt, macht das Bildaufnahmemodul mit den zwei Kameras ein Foto.

Implementierungsnahe wurde nur eine Kamera verwendet, da die Stereo Vision Probleme gegeben hat. Nachdem das Foto gemacht wird folgt die Image Normalisierung bzw. das „Image Processing“. Wenn ein Bild zu klein ist, wird die Größe angepasst, wenn ein Gesicht verdreht ist wird es gerade rotiert. Mehr dazu wird im Normalisierungabschnitt erklärt.

Danach folgt die Erkennung von Gesichtern, das Ausschneiden von denen und die Extrahierung der Gesichtsschlüsselpunkte. Sie werden dann zum Abgleich oder Registrierung je nach Bedarf, geschickt.

Bei der Umsetzung wurden bestimmte Bereiche miteinander verknüpft, was zu Folgenden führte: was bei der Planung steht, stimmt nicht völlig mit der Methoden zur Umsetzung überein.

5.2.2 Gesichtserkennung

Das erste Ziel, dass bei der Gesichtserkennung erfüllt worden war ist das Finden von Gesichtern, also die Feststellung, wo sich die Gesichter im Bild befinden.

Dafür ist der vorgefertigter Klassifikator „*Haar Cascade Classifiers*“ benutzt worden. Gesichtserkennung durch „Haar“ Merkmale ist eine sehr effektive Methode die von Paul Viola und Michael Jones entwickelt wurde, indem sie Merkmale gruppiert haben und die Erkennung von diesen dadurch schneller erfolgt ist. Die Arbeit heißt „*Rapid Object Detection using a Boosted Cascade of Simple Features*“. [43]

Unten wird eine kurze technische Übersicht über diese Klassifikator gegeben.

Es geht hier um Maschinelles Lernen, wie diese Kaskadenfunktionen durch viele negative(Bilder ohne Gesichter) und positive(Bilder mit Gesichter) Bilder trainiert wurden um Objekte zu erkennen. In diesem Fall arbeiten wir selbstverständlich mit Gesicht Objekten. Dafür wurden Merkmale Haare benutzt. Auf Abbildung 5.4 ist dies dargestellt. Jedes Haar Merkmal ist nichts anders als ein Wert, den man durch das Subtrahieren der Summe der Pixel unter dem weißen Rechteck von der Summe der Pixel unter dem schwarzen Rechteck erhält.

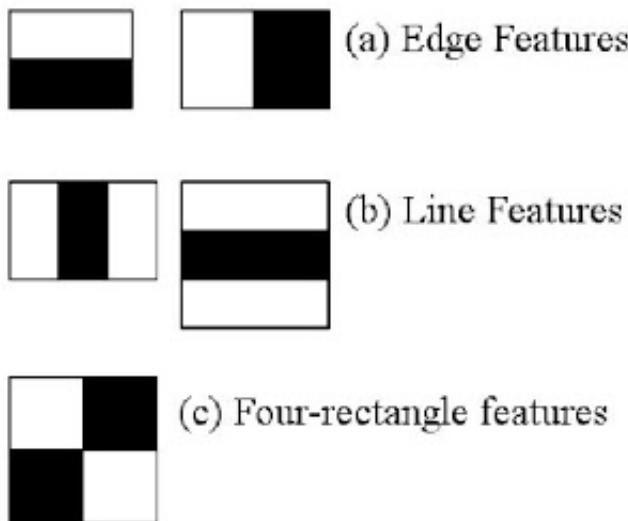


Abbildung 5.4: Haar Features[43]

Man muss aufpassen, da nicht alle Merkmale von Nutzen sein könnten. Man kann zum Beispiel deutlich auf Abbildung 5.5 sehen, wie die Nase viel heller als die Region bei den Augen ist. Die Selektion von den besten Merkmalen wird durch den Adaboost Algorithmus berechnet.

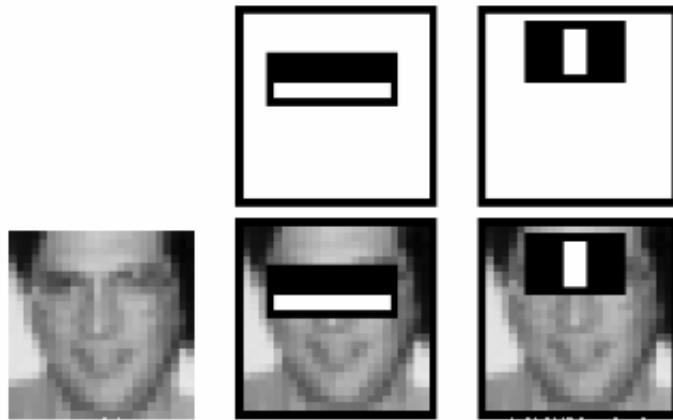


Abbildung 5.5: Haar

Mit dieser Absicht wenden wir alle Funktionen auf alle Trainingsbilder an. Für jedes Merkmal wird der beste Schwellenwert ermittelt, der die Gesichter in positive und negative klassifiziert.

Umsetzung in Code

In Listing 5.1 ist ein Code Abschnitt der Gesichtserkennung dargestellt.

```

1 face_cascade =
2 cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
3
4 image = cv2.imread('gesicht.jpg')
5
6 gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
7
8 faces = face_cascade.detectMultiScale(gray, 1.1, 4)
9
10 for(x, y, w, h) in faces:
11
12 cv2.rectangle(image, (x, y), (x+w, y+h), (255,0,0), 2)
13

```

Listing 5.1: Kern Code für Gesichtsdetektion

Es wurde die trainierte Klassifikator-XML-Datei (haarcascade_frontalface_default.xml) die sich im GitHub-Repository von OpenCV befindet, verwendet. In den Zeilen **1** und **2** wurde sie geladen und in die Variable `face_cascade` gespeichert. In den Zeile **4** wird das Bild von den Funktion `imread` geladen. Die Erkennung funktioniert nur bei Graustufenbildern. Daher wurde das Farbbild in Graustufen umgewandelt, wie in der Zeile **6** ersichtlich. Der Funktion `detectMultiScale` in der Zeile **8** erkennt die Gesichter im Bild. Diese Funktion ist sehr wichtig und braucht 3 Argumente – das Eingabebild, der Skalierungsfaktor (`scaleFactor`) und die Zahl `minNeighbours`. `scaleFactor` gibt die an, um wie viel das Bild vergrößert bzw. verkleinert wird.

minNeighbours gibt an, wie viele Nachbarrechtecke jedes Rechteck haben muss. Dieser Parameter bestimmt die Qualität der erkannten Gesichter: Ein höherer Wert führt zu weniger Erkennungen, jedoch zu einer höheren Qualität. Es ist bei uns nicht so entscheidend viele Gesichter zu erkennen, sondern die Qualität. Es wird der Wert 4 benutzt.

In Zeile **10** wird durch die Gesichter *faces* iteriert.

x und *y* entsprechen den Koordinaten vom Bild, *w*, *h* bezeichnen die Breite und Höhe *width*, *height*.

Schließlich wird in Zeile **12** mit den Funktion *cv2.rectangle* ein Rahmen auf dem Gesicht gezeichnet.

Die Funktion bekommt diese Parameter: *image* für die Bildeingabe, *x*, *y* beschreiben den Startpunkt, *w*, *h* den Endpunkt, die Farbe und die Dicke der Rahmen werden in den 2 letzten Parameter angegeben.

Dieser Code wurde so umgeändert, dass er in der Lage ist, durch mehrere *Predictors* bzw. Klassifikatoren, die Genauigkeit der Gesichtserkennung zu erhöhen.

Auf Abbildung 5.6 sieht man deutlich ein erkanntes Gesicht.

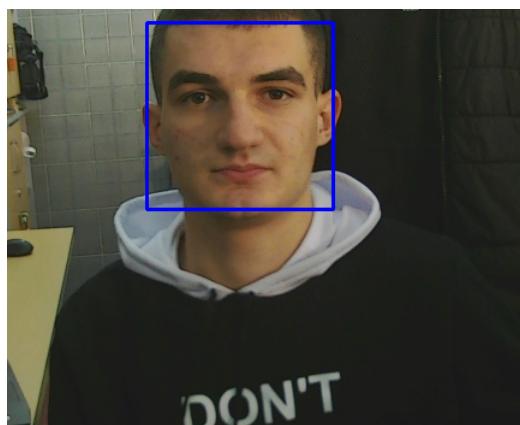


Abbildung 5.6: Output: Box auf Gesicht

5.2.3 Normalisierung

In dieser Kapitel wird ein sehr wichtiger Bereich, die sogenannte Normalisierung eines Bildes, genau erklärt.

Was ist Normalisierung?

Normalisierung in *Computer Vision* ist der Prozess der Bereitstellung von Daten, bei dem "falsche" Daten, die die Genauigkeit unseres Systems beschädigen können, korrigiert werden.

Normalisierung heißt in unserem Fall Ausrichtung. Die Ausrichtung der Gesichter, die im Bild "falsch" positioniert sind.

Also entspricht Normalisierung dem Prozess in dem man zuerst die geometrische Struktur von Gesichtern in digitalen Bildern identifiziert und dann versucht eine maßgebliche

Ausrichtung des Gesichts basierend auf Skalierung und Rotation zu erhalten.

Methoden für Normalisierung

Es gibt viele Methoden mit denen man Bilder normalisieren könnte. Einige von denen basieren auf pre-definierten 3D-Modellen und transformieren dadurch die Eingabebilder, sodass die Gesichtsschlüsselpunkte der Eingabebild mit denen von dem 3D Modell übereinstimmen.

Die in dieser Diplomarbeit verwendete Methode, eine eher einfachere, verlässt sich nur auf die Gesichtspunkte selbst, um eine normalisierte Darstellung des Gesichts durch Affine- und Ähnlichkeits-Transformation zu erhalten. Diese Methode wurde deswegen implementiert, weil sie sehr effizient ist.

Warum Normalisierung?

Der Grund warum in dieser Arbeit Normalisierung von Daten durchgeführt wird, ist dass viele Gesichtserkennungsalgorithmen, einschließlich unserem, von der Anwendung dieser Ausrichtung sehr profitieren. Es wird dadurch die Präzision der Gesichtserkennung erhöht.

Implementation

Die hier verwendete Methode für die Normalisierung, wie oben kurz erwähnt wurde, wendet Ähnlichkeitstransformation bei zwei Paaren entsprechender Punkte an. Die Punkten sind die von Dlib extrahierten Gesichtsmerkmale, siehe Kapitel 5.2.5.

OpenCV benötigt in diesem Fall 3 Punkte zur Berechnung der Ähnlichkeitsmatrix. Wir nehmen somit als dritten Punkt, den dritten Punkt eines gleichseitigen Dreiecks mit diesen beiden gegebenen Punkten an.

Was eine Ähnlichkeitstransformation ist, lässt sich mathematisch beschreiben. Eine Ähnlichkeitstransformation ist eine Transformation wie z.b.: Reflexion, Rotation oder Translation.

Wenn eine Figur durch eine Ähnlichkeitstransformation transformiert wird, wird ein Bild erstellt, das der ursprünglichen Figur ähnlich ist. Mit anderen Worten, zwei Figuren sind ähnlich, wenn eine Ähnlichkeitstransformation die erste Figur zur zweiten Figur transformiert.

Ein solches Prinzip ist auch in unsere Methode eingesetzt worden. Es wird eine zweites, dem ersten Bild ähnliches Bild, erstellt.

```
1
2 def similarityTransformMat(initialPoints, destinationPoints):
3     ...
```

```

4 tform = cv2.estimateAffinePartial2D(np.array([initialPoints]),
5     np.array([destinationPoints]))
6 return tform[0]
7

```

Listing 5.2: Implementation Normalisierung

Im Code Abschnitt 5.2 wird die Methode gezeigt, die als Parameter die Anfangs- und die Zielpunkte nimmt und aus den beiden jeweils einen dritten Punkt berechnet.

Sie werden dann in Arrays gespeichert und aus denen wird die Ähnlichkeitstransform von den in OpenCV eingebetteter Funktion *cv2.estimateAffinePartial2D* berechnet.

Ausgabe ist eine 2D affine Transformation, also eine 2x3 Matrix, oder eine leere Matrix, wenn die Transformation nicht geschätzt werden konnte.

”Die Funktion schätzt eine optimale affine 2D-Transformation mit 4 Freiheitsgraden, die auf Kombinationen aus Translation, Rotation und gleichmäßiger Skalierung beschränkt sind. Verwendet den ausgewählten Algorithmus für eine robuste Schätzung.” [30]

Die nächste Funktion macht die Ausrichtung des Gesicht im Bild. Es bekommt ein, von der Funktion *cv2.imread* eingelesenes Bild, die gewünschte Größe, und die Dlib Gesichtsmerkmale als Parameter.

```

1 def faceAlign(image, size, faceLandmarks):
2     (h, w) = size
3     initialPoints = []
4     destinationPoints = []
5
6     initialPoints = [faceLandmarks[36], faceLandmarks[45]]
7
8     destinationPoints = [(np.int(0.3*w), np.int(h/3)), (np.int(0.7*w),
9     np.int(h/3))]
10
11    similarityTransform = similarityTransformMat(initialPoints,
12        destinationPoints)
13
14    faceAligned = np.zeros((image.shape), dtype=image.dtype)
15
16    faceAligned = cv2.warpAffine(image, similarityTransform, (w, h))
17
18    return faceAligned
19

```

Listing 5.3: Gesichtsausrichtung

Im Code Abschnitt 5.3 in Zeile **6** wird die Position der linken Ecke des linken Auges und der rechten Ecke des rechten Auges von dem Eingabebild genommen.

In Zeile **8** wird die Position der linken Ecke des linken Auges und der rechten Ecke des rechten Auges im ausgerichtete Bild berechnet.

In Zeile **10** und **11** wird die Ähnlichkeitstransformation durch die vorher erstellten Funktion *similarityTransformMat* berechnet.

In Zeile **12** wird das ausgerichtete Gesicht in einem Tupel gespeichert.

Schließlich wird die Ähnlichkeitstransformation durch die Methode *cv2.warpAffine* angewendet.

Diese Methode bekommt das Tupel vom Bild, *similarityTransform*, und die Größen vom Bild als Parameter. Das Ergebnis wird dann von der Funktion zurückgegeben. (Zeile **16**)

Eine affine Transformation ist jede Transformation, die die Kollinearität (d. H. Alle auf einer Linie liegenden Punkte, die anfänglich nach der Transformation noch auf einer Linie liegen) und die Entferungsverhältnisse (z. B. der Mittelpunkt eines Liniensegments bleibt der Mittelpunkt nach der Transformation) bewahrt. Eine affine Transformation wird auch als Affinität bezeichnet.[24]

Zusätzlich wurde noch eine Glättungsfilter angewendet. Dies geschieht durch die Funktion *cv2.filter2d*, die ein Bild mit dem Kernel faltet.

Sie bekommt folgende Parameter: *output* - Zielbild.

kernel - Faltungskern (oder vielmehr ein Korrelationskern), eine einkanalige Gleitkommamatrix.

ddepth - gewünschte Tiefe des Zielbildes.

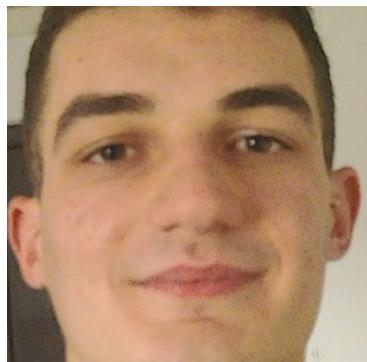
Der Kernel ist auf Codeabschnitt 5.4 genau sichtbar.

```

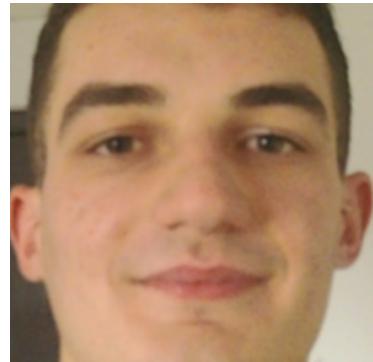
1 kernel = np.ones((5,5),np.float32)/25
2
3 output = cv2.filter2d(output,-1, kernel)
4

```

Listing 5.4: Glättungsfilter



(a) Nicht normalisiert



(b) Normalisiert

Abbildung 5.7: Unterschied: normalisiert, nicht normalisiert



Abbildung 5.8: Unterschied: normalisiert, nicht normalisiert

Auf Abbildungen 5.8 und 5.7 sieht man ganz klar die Unterschiede die die Normalisierung in Bildern verursacht.

5.2.4 Zuschneiden von Gesichter

Nachdem Gesichter gefunden werden, muss man daraus eigene Bilder machen, die das Extrahieren der Schlüsselpunkte erleichtern.

```

1 if nrFace > 0:
2     for face in faces:
3         for(x, y, w, h) in faces:
4             r = max(w, h) /2
5             centerx = x + w /2
6             centery = y + h /2
7             nx = int(centerx - r)
8             ny = int(centery - r)
9             nr = int(r * 2)
10            faceimg = image[ny:ny+nr, nx:nx+nr]
11            filenam = input("Give new filename for cropped photo: \n")
12            image2 = cv2.imwrite(filenam,faceimg)
13        elif nrFace <= 0:
14            print("no faces found")

```

Listing 5.5: Code Abschnitt: Gesicht Zuschneiden

In Listing 5.5 sieht man folgendes:

nrFace bestimmt die Anzahl der Gesichter, die von den Kaskadenklassifikator gefunden wurden. Nur wenn dieser Wert größer als 0 ist soll das Programm weiterlaufen. Es wird durch jedes Gesicht iteriert.

Zeilen **1-6** berechnen das Zentrum vom neuen Bild und in Zeile **7** wird ein neues Bild mit den berechneten Parametern angelegt.

Mit `imwrite` wird das Bild gespeichert (**Zeile 12**). Auf Abb.5.9 sieht man deutlich nur

das Gesicht.



Abbildung 5.9: Output: Zugeschnittenes Bild

5.2.5 Gesichtsschlüsselpunkte Extraktion

Nun kommt ist es zu einem sehr wichtigen Punkt meines Teils der Diplomarbeit, die Extraktion Gesichtsschlüsselpunkte.

Dazu wurden die Gesichtsmerkmale „*Facial Landmarks*“ von dlib verwendet.

Die Gesichtsmarkierungen werden verwendet, um Bereiche des Gesichts zu finden und darzustellen, wie zum Beispiel: die Augen, die Augenbrauen, die Nase, den Mund und den Kiefer.

Wie macht man das? Wie erkennt man Gesichtsmerkmale?

Das ist eine Teilmenge des Problems der Formvorhersage.

Bei einem vorgegebenen Eingabebild (und normalerweise einem ROI¹⁰, die das interessierende Objekt angibt) versucht ein Formvorhersager, wichtige Punkte entlang der Form zu lokalisieren.

Dieser Formvorhersager, der im Dlib integriert ist wurde von Kazemi and Sullivan in ihrem Paper: *One Millisecond Face Alignment with an Ensemble of Regression Trees* entwickelt.[34]

Dieser Methode werden viele Trainingsdaten zur Verfügung gestellt, damit sie eine Kombination von Regressionskurven trainiert, um die Positionen der *Facial Landmarks* direkt aus den Pixelintensitäten selbst zu berechnen.[34]

Dadurch werden unsere gewünschten 68 Gesichtsmarkierungen mit hohen Vorhersagbarkeit extrahiert und als x, y Koordinaten weitergegeben. Zusätzlich ist es auch so, dass diese Koordinaten nicht von den Dimensionen des Bildes abhängig sind. Die Indizes der 68 Koordinaten sind in der Abb.5.10 dargestellt:

¹⁰Region of interest, Region von Interesse

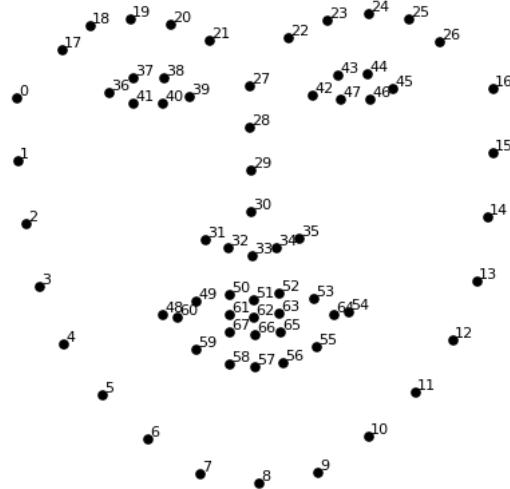


Abbildung 5.10: Gesichtsschlüsselpunkte [34]

Nachdem der Prediktor geladen wird, speichern wir die Gesichtsschlüsselpunkte in einem Formobjekt mit 68(x,y) Koordinaten.

Es wird hier nicht die *resize* Funktion verwendet, weil das Bild an Qualität verliert und es rechenintensiver ist, je größer das Eingabebild wird.

Jede Koordinate läuft in einer Schleife durch und entspricht dem spezifischen Gesichtsmerkmal im Bild. Auf Abbildung 5.11 sieht man die extrahierten Daten von einem Gesicht.

0. 0.425 0.1359375	17. 0.4458333333333336 0.11875	34. 0.5895833333333333 0.1984375	51. 0.5770833333333333 0.21875
1. 0.4291666666666664 0.1625	18. 0.4625 0.103125	35. 0.6041666666666666 0.1953125	52. 0.5916666666666667 0.215625
2. 0.4375 0.190625	19. 0.4875 0.0984375	36. 0.4770833333333336 0.1328125	53. 0.6104166666666667 0.21875
3. 0.4458333333333336 0.215625	20. 0.5166666666666667 0.1	37. 0.49375 0.1265625	54. 0.6291666666666667 0.2234375
4. 0.4604166666666664 0.2390625	21. 0.5416666666666666 0.1078125	38. 0.5125 0.125	55. 0.6125 0.23125
5. 0.4833333333333334 0.259375	22. 0.5895833333333333 0.10625	39. 0.53125 0.1328125	56. 0.5958333333333333 0.2359375
6. 0.5125 0.2734375	23. 0.6166666666666667 0.0984375	40. 0.5125 0.1375	57. 0.5791666666666667 0.2375
7. 0.5479166666666667 0.2859375	24. 0.6458333333333334 0.09375	41. 0.49375 0.1375	58. 0.5645833333333333 0.2359375
8. 0.5833333333333334 0.2875	25. 0.6729166666666667 0.096875	42. 0.6083333333333333 0.13125	59. 0.5458333333333333 0.2328125
9. 0.61875 0.2828125	26. 0.6916666666666667 0.1125	43. 0.625 0.121875	60. 0.5333333333333333 0.2265625
10. 0.65 0.2703125	27. 0.56875 0.1265625	44. 0.64375 0.121875	61. 0.5625 0.225
11. 0.675 0.253125	28. 0.5708333333333333 0.1453125	45. 0.6604166666666667 0.128125	62. 0.5791666666666667 0.2265625
12. 0.6916666666666667 0.23125	29. 0.5729166666666666 0.1640625	46. 0.6458333333333334 0.1328125	63. 0.59375 0.225
13. 0.7 0.20625	30. 0.5729166666666666 0.184375	47. 0.6270833333333333 0.134375	64. 0.6229166666666667 0.2234375
14. 0.7041666666666667 0.1796875	31. 0.5479166666666667 0.196875	48. 0.525 0.2265625	65. 0.59375 0.2234375
15. 0.7104166666666667 0.153125	32. 0.5625 0.1984375	49. 0.54375 0.2203125	66. 0.5791666666666667 0.225
16. 0.7125 0.125	33. 0.5770833333333333 0.2	50. 0.5625 0.2171875	67. 0.5645833333333333 0.2234375

Abbildung 5.11: Gesichtsdaten

Die Merkmale werden in einer numpy Array gespeichert für die Speicherung in der Datenbank.

5.3 Herausforderungen, Probleme und deren Lösung

Die größte Herausforderung lag bei der Planung der Software und den Methoden die zum Extrahieren von den Gesichtsschlüsselpunkten dienten.

Es hat mich viel Zeit gekostet, bis ich eine geeignete Lösung gefunden habe und das hat mir viel Stress gemacht.

Eine weitere Herausforderung war das Verknüpfen von den Entwicklungsumgebungen und die Kooperation zwischen den Teammitgliedern.

Die verwendete Systeme waren all zu unterschiedlich und es konnte keine Standardisierung zwischen ihnen gefunden werden. Also ist viel Zeit beim Installieren und Konfigurieren investiert worden. Ein Grund dafür ist die mangelnde Erfahrung mit den verwendeten Technologien.

Viele Sachen, wie z.b.: die Einteilung der Arbeit, die genaue Spezifikationen, die sehr grobe Planung, war am Beginn auch wegen den Kommunikationslücken unklar.

Als das Projekt weiter fortschritt, hat sich der Bedarf an Genauigkeit und Präzision auch enorm gesteigert. Es hat immer wieder Fälle beim Finden von Gesichter gegeben, in denen die fertig gestellten neuronale Netze nicht so gut funktioniert haben .

Es war keine gute Idee sich nur von einem Prädiktor abhängig zu machen. Es war deutlich dass, die Einsetzung von mehreren neuronalen Netzen Erhöhung der Treffsicherheit ergeben hätte.

Eine einfache Normalisierung genügte auch nicht die Umwandlung der Bilder in Graustufe), sondern es war eine komplexere Lösung dafür notwendig.

Bis das funktioniert hat, hat es viel Zeit und Mühe gekostet. Dafür waren auch relativ fortgeschrittene mathematische Kenntnisse nötig. Dabei habe ich viel Try and Error angewendet.

Es hat auch Schwierigkeiten beim Qualitätsmanagement gegeben. Die Erfahrung fehlte, somit waren gewisse Maßstäbe und Standards auch nicht bekannt.

5.3.1 Lösungen

Während der Arbeit habe ich viel recherchiert und mich genau über alles Mögliche informiert.

Es wurde viel herumprobiert und experimentiert. Es wurde auch sehr viel getestet, damit die Ziele auch qualitativ erfüllt wurden.

Die Kommunikation hat sich steigendem mit Bedarf über die Zeit stark verbessert und dadurch sind auch die Unklarheiten abgeklärt worden.

5.4 Qualitätssicherung, Controlling

Die Qualität wurde durch verschiedene Methoden gesichert. Eine von denen war die Methode 5xWarum.

„Fünf warum“ ist eine iterative Methode, die Fragen als Basis hat und die Beziehungen zwischen Fragen und Problemen.

Es geht hier um die Verschachtelung der Ursachen und das Herausfinden von ihnen durch iterative Fragetechnik, da viele Probleme nicht nur eine einzige Ursache haben. Die Methode ruft jedes Mal eine andere Folge von Fragen auf.[25]

Die aufgetauchten Probleme haben viele Ursachen, die nicht mit dem ersten Blick sichtbar sind. 5x warum hilft durch diese verschachtelten Ursachen das grundlegende Problem zu entdecken.

Eine Problemstellung: „Segmentation fault“ beim Finden von Keypoints mit FAST¹¹.“

1. **Was ist überhaupt ein „Segmentation Fault“?** Ein Segmentierungsfehler tritt auf, wenn ein Programm versucht, auf einen Speicherort zuzugreifen, auf den es nicht zugreifen darf, oder wenn versucht wird, auf einen Speicherort auf nicht zulässige Weise zuzugreifen (z. B. beim Versuch, an einen schreibgeschützten Speicherort zu schreiben, oder einen Teil des Betriebssystems zu überschreiben).
2. **Warum passiert das, warum versucht mein Program auf einen Speicherort zuzugreifen, auf den es nicht zugreifen darf?**

Problem ergibt sich in dieser Zeile: “kp = fast.detect(image, None)”

Wahrscheinlich durften die Daten die fast.detect zurückgibt nicht in kp gespeichert werden.

ODER

Das Paket in dem die FAST Algorithmus drinnen ist ist nicht (richtig) installiert worden. Warum? Alle nötigen Pakete wurden in einer gesamten Installation geholt und FAST war nicht da.

3. **Warum dürfen die Daten die fast.detect zurückgibt nicht in kp gespeichert werden ?**

Die Daten die fast.detect zurückgibt, dürfen nicht in kp gespeichert werden, weil kp kein Array ist.

4. **Warum ist kp kein Array?**

Kp ist kein array, weil man es in Python manuell angeben muss.

5. **Warum wurde es nicht manuell angegeben?**

¹¹Features from Accelerated Segment Test

Es wurde nicht manuell gegeben weil, die Methode fast.detect nicht gut recherchiert wurde. Man sollte mehr darüber in die Dokumentation nachschauen.

Konklusion

Durch die 5x Warum Methode ist es zu den Ergebnis gekommen: Es musste ja mehr untersucht werden bevor man eine solche Methode implementiert. Die „5xWarum“ Methode hat dabei geholfen, dass die eigentliche Ursache des Problems herausgefunden wurde.

Die nächste Schritte sind: entweder eine andere Methode, die schon installiert ist, verwenden oder die benötigten Pakete für FAST manuell installieren.

5.5 Ergebnisse

Ich habe ungefähr 180 Stunden Zeit ins gesamt für die Diplomarbeit investiert und folgendes erreicht.

- Gesichtserkennung und das Zuschneiden von Gesichter wurden fertig implementiert.
- Bilder sind normalisiert worden.
- Gesichtsschlüsselpunkte wurden extrahiert.

5.6 Evaluierung und Resümee

In diesem Kapitel geht es um die Reflexion, Evaluierung der Ergebnisse. Was von der Planung abgewichen ist usw.

5.6.1 Planung vs. Realisierung

Es wurde am Anfang so geplant, dass die Klassifikatoren und die neuronale Netze alle selbst gemacht werden müssten. Über die Zeit wurde mir klar, dass so was zu kompliziert und fast unmöglich war. Die Genauigkeit der Gesichtserkennungsalgorithmen könnte nie erreicht werden, wenn es vom Beginn erarbeitet wurde. Geplant waren auch gewisse Ziele, die über die Diplomarbeit-Mitglieder falsch verteilt worden waren. Z.b.: Ein Ziel von mir war die Vektorumwandlung, die später so geschätzt wurde, dass es eigentlich zu Rei's Teil gehörte.

5.6.2 Ergebnisse

Im technischen Aspekt habe ich viel gelernt. Diese Diplomarbeit hat viele Kenntnisse mit sich gebracht. Zahlreiche neue Bibliotheken wurden verwendet. Ein ganz unbekanntes Framework ist kennengelernt worden. Ich habe viel Erfahrung mit Computer

Vision bekommen. Da meine Aufgaben zum Teil sehr anspruchsvoll waren, habe ich viel Zeit und Mühe für die Realisierung investiert. Soft Skills wurden auch dazugewonnen. Ich habe erlernt wie wichtig Teamwork und Kommunikation ist. Wenn ich die ganze Diplomarbeit noch einmal machen würde, wäre vieles anders.

Kapitel 6

Gesichtserkennung

Am Beginn wird eine allgemeinere Übersicht dieses Diplomarbeitskapitels gemacht bzw. dargestellt, damit verstanden wird, worum es in diesem Teil geht.

6.1 Allgemeines

Die Gesichtserkennung ist der Teil des Systems, mit dem ich mich beschäftige. Es wird durch einen Vergleich von Bildern überprüft, ob die Person vorher schon registriert worden ist oder nicht, und ob seine Gesichtsdaten schon am Server existieren oder nicht. Nur wenn die Vergleichsergebnisse positiv sind, wird die Person erkannt. Der Person werden die Ergebnisse durch Anzeiger kommuniziert. Dieses Teil des Projektes erfordert eine Arbeit mit Datenbanken, Gesichtsvergleichsalgorithmen und mit vielen Systemtests. Teil meiner Aufgabe ist auch der Aufbau des Systems und alles was mit Hardware zu tun hat. Alle Hardware Komponenten werden in den folgenden Kapiteln klar, verständlich und deutlich erklärt. Eine grobe Skizze des Systems ist in Abb.6.1 zu sehen.

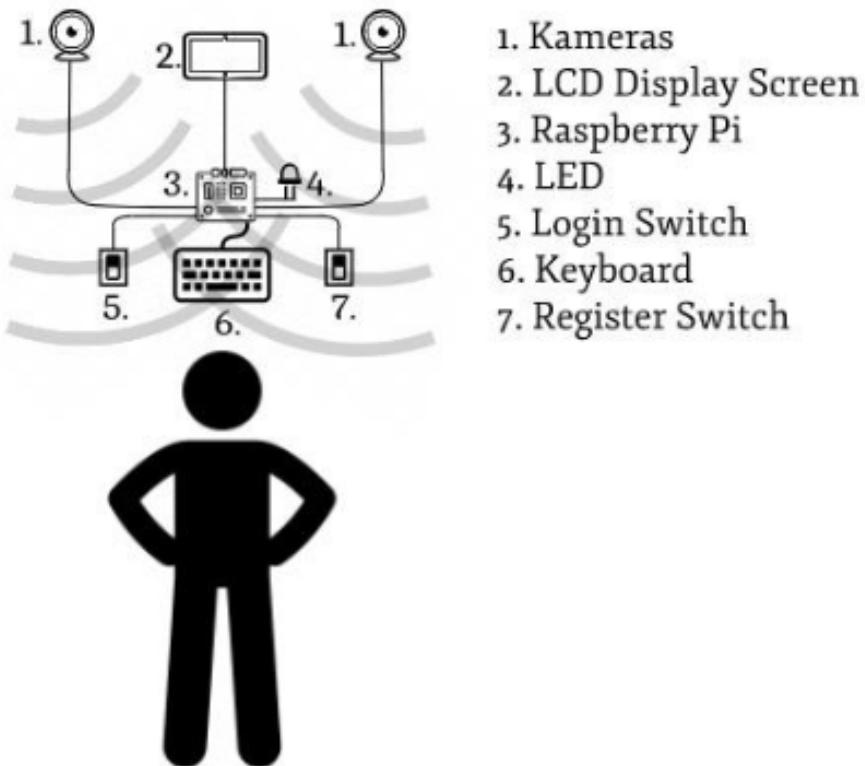


Abbildung 6.1: Grobe Skizze des Systems

6.2 Technische Lösung

In diesem Abschnitt wird eine feinere und detaillierte Übersicht im Bezug auf die technische Lösung gegeben. Alles was mit Technik zu tun hat, wird hier erklärt.

6.2.1 Hardware und Aufbau

Zuerst wird der Aufbau des Systems beschrieben, zusammen mit allen Komponenten, die zu verwenden sind. Ohne die Hardware würde nichts funktionieren, weil die Software ohne die Hardware nicht funktionieren kann.

Bauteile und HW-Komponenten

Die Bauteile, die für dieses System verwendet worden sind, sind die folgende:

1. *Steckboard bzw. Steckplatine*

Die Steckplatine ist eine Komponente der Hardware des Systems. Sie wird für die Entstehung der elektrischen Verbindung von verschiedenen elektrischen Bauteile benötigt, um elektrische Schaltungen zu bauen oder um verschiedene Tests und

Experimenten zu machen. In dieser Steckplatine werden alle anderen Komponenten platziert, damit die Verbindung erstellt werden kann und damit das System funktionieren kann.

2. Kabel bzw. Leiter

Damit die verschiedenen Komponenten, die in der Steckplatine platziert sind, miteinander verbunden werden können, braucht man unbedingt Kabel. Mithilfe von Kabeln können elektrische Impulse und Signale fließen, damit die Energie und die Information übertragen werden. Die verwendeten Kabel sind aus Kupfer und vom Typ Male-Male als auch vom Typ Male-Female. Die Kabeln vom Typ Male-Female werden verwendet, um die Verbindungen zwischen den Elementen in der Steckplatine und dem Raspberry Pi zu ermöglichen. Auf der anderen Seite werden die Male-Male Kabeln verwendet um die Verbindungen innerhalb der Steckplatine zu ermöglichen.

3. LEDs¹²

LEDs sind elektronische Halbleiter Elemente, die Licht produzieren können, wenn sie Spannung kriegen. Ein LED besteht aus zwei Beinen. Das längere Bein ist die Anode, die den Pluspol symbolisiert. Das andere Bein ist die Kathode und symbolisiert den Minuspol. Durch die Beine wird der Kontakt mit der Steckplatine hergestellt.[13]

4. Widerstand

Ein Widerstand ist ein elektrisches Bauteil, das zur Reduzierung von Strom verwendet wird, damit ein Gleichgewicht zwischen Strom und Spannung gesichert werden kann. Die Einheit ist Ohm.[14]

5. Taster

Ein Taster wird wie ein Schalter gedrückt, mit dem Zweck Impulse oder Signale zu schicken. Im Gegenteil zu einem Schalter wird der Taster nach der Betätigung wieder in der Basiszustand zurückgestellt. Ein Plusleiter, Minusleiter und ein Datenleiter sind bei einem Taster vorhanden.[23]

6. Raspberry Pi

Raspberry Pi ist ein Minicomputer, der in diesem Projekt den normalen Computer ersetzt. Der verwendete Raspberry Pi, Version 3, hat 4 USB-Anschlüsse, einem Netzteil, eine SD-Karte, 16 GPIO¹³ Pins und eine VGA Schnittstelle. Die 3.-Bit SD-Karte ist ein wichtiges Element, weil dort alle Daten und Informationen gespeichert sind.

7. Bildschirm

Ein Bildschirm ist eine Anzeige, die für die visuelle Darstellung von verschiedenen Informationen oder Daten(wie Videos, Fotos, Statistiken usw.) verwendet wird. Ein Bildschirm wird zu den heutigen Zeiten sehr häufig verwendet, aufgrund der hohen Benutzerfreundlichkeit, die angeboten wird.[15]

¹²Light Emitting Diode

¹³Generated Input Output

8. Tastatur

Eine Tastatur ist ein Input Gerät, dass durch das Drücken von Tasten den Benutzer die Eingabe von Daten oder Befehle ermöglicht.[16]

9. Kamera

Es wird eine Kamera benötigt, die die Fotos der Gesichter der Personen macht. Sie werden auch im Raspberry Pi integriert bzw. mit dem Raspberry Pi verbunden. Die Kameras sind vom Typ Aukey.

Schaltplan und Erklärung des Aufbaus

Die Hardware Komponenten werden in einer Steckplatine platziert. Dort werden die Verbindungen mit den anderen Komponenten sowie mit dem Raspberry Pi hergestellt. Die elektrische Schaltung wird durch einen Schaltplan beschrieben. Dieser Schaltplan wurde mit Hilfe eines Programms, „Fritzing“, erstellt und spielt eine sehr wichtige Rolle bei der Organisation und Planung des Schaltkreises. Der Schaltplan ist auf Abb.6.2 zu sehen.

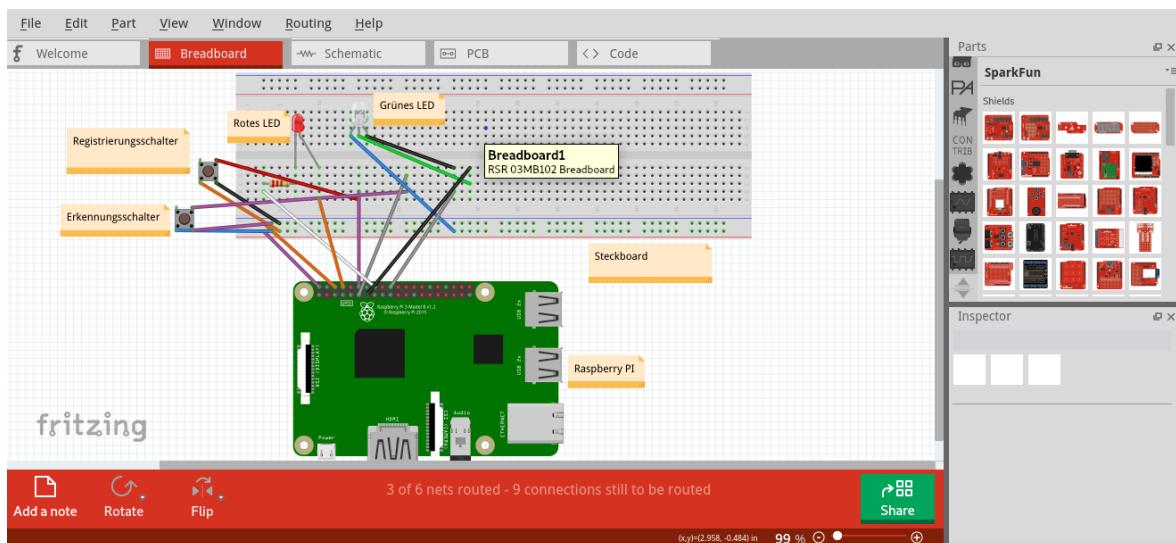


Abbildung 6.2: Schaltplan des Systems

Wie der Schaltplan zeigt, besteht das System aus zwei LEDs, zwei Tastern, einem Widerstand, einer Steckplatine und einem Raspberry Pi. Sehr wichtig für den Aufbau der Schaltung sind die GPIO Pins. Diese Pins sind in dem Raspberry Pi platziert und können als Input, als Output oder Spannungspins verwendet werden. Der erste Taster dient für die Gesichtsregistrierung und besteht aus einem Pluspol, einem Datenleiter und aus dem Minuspol. Der Pluspol Anschluss wird mittels Steckplatine mit dem 5-Volt Pin des Raspberry Pi verbunden, während der Minuspol-Anschluss mit dem Minuspol der Steckplatine verbunden wird. Der Datenanschluss ist mit dem GPIO18 Pin verbunden. Wie bei dem Registrierungs-Taster wird auch bei dem Erkennungstaster¹⁴ der Minuspol-Anschluss mit einem Pin des Minusbereichs der Steckplatine verbunden.

¹⁴ Auch als Login Taster genannt

Der Pluspol-Anschluss gehört zu dem 5-Volt Pin des Raspberry PI, während der Datenanschluss mit dem GPIO17 Pin verbunden ist. Die rote LED wird verwendet wenn die Registrierung oder Erkennung der Benutzer im System nicht erfolgreich war. Das längere Bein(die Anode) wird mit dem GPIO23 Pin des Raspberry verbunden, während die Kathode mit dem Minuspol-Bereich der Steckplatine verbunden wird. Die grüne LED ist eine RGB(Red Green Blue) LED. Diese LED kann die Farbe ändern, und wird im Fall einer erfolgreichen Registrierung oder Erkennung des Benutzers im System verwendet. Sie hat im Gegensatz zu der normalen LED 3 Anschlüsse. Der Minus-Anschluss wird mit dem Minuspol Bereich der Steckplatine verbunden und der Pluspol-Anschluss wird mit dem Pluspol Bereich der Steckplatine verbunden. Mit dem GPIO27 Pin des Raspberry wird der Datenanschluss verbunden. Die GPIO Pins sind extrem wichtig für die Integration der Tastern, LEDs und der anderen Bauteile in dem technischen und logischen Teil bzw. in der Software und in den verwendeten Skripten.

6.2.2 Software

Nachdem der Aufbau und die verwendete Hardware des Systems beschrieben wurden, wird nun die Software beschrieben. In diesem Unterkapitel wird alles was mit dem logischen Teil der Umsetzung zu tun hat besprochen: das verwendete Betriebssystem, Programmiersprachen, Frameworks, Technologien und Planungsmethoden. Es wird jede Aufgabe zusammen mit der zugehörigen Lösung im Detail beschrieben und jedes programmierte Skript erklärt.

Verwendete Technologien

In diesem Teil des Projekts, das Gesichtserkennung heißt, werden für die Umsetzung 2 verschiedene Technologien verwendet, die in den anderen Teilen nicht oder anders verwendet werden.

- Open CV

Das wichtigste Framework für dieses Projekt ist OpenCV. OpenCV ist eine Softwarebibliothek, die für Computer-Vision und maschinelles Lernen verwendet wird. Die Bibliothek verfügt über mehr als 2500 optimierte Algorithmen, die sowohl klassische als auch moderne Computer Vision- und maschinelle Lernalgorithmen umfassen. Diese Algorithmen können verwendet werden, um Gesichter zu registrieren und zu erkennen, um Objekte zu identifizieren, menschliche Handlungen in Videos zu klassifizieren und Kamerabewegungen zu verfolgen. Dieses Framework wurde deswegen gewählt, weil die Vielfältigkeit der angebotenen Optionen und Paketen einfacher größer ist als bei anderen Frameworks. Ein anderer Vorteil ist das OpenCV Open-Source ist. Die größte Herausforderung ist die lange und komplizierte Installation auf Linux. [19]

- Fritzing

Fritzing ist ein Programm das für die graphische Darstellung des Schaltkreises des Systems verwendet wurde. Durch dieses Programm wurde auch der Schaltplan des Systems erstellt(Abb.6.2). Fritzing bietet eine sehr große Menge von

elektronischen Komponenten und ein PCB¹⁵, eine Steckplatine und einen Schematic View. In diesem Fall wurde die Breadboard View gewählt, weil dort alles übersichtlicher ist. Was noch gut ist, ist das Fritzing kostenlos in Linux angeboten wird, was für Windows nicht der Fall ist.[17]

Lösungsweg- Beschreibung und Erklärung

Der Lösungsweg für die Umsetzung der Aufgaben der Gesichtserkennung ist streng mit der vorherigen Planung verbunden. Deshalb wurde nicht nur das Big Picture(Große Sicht des Systems von Außen), die in die vorherigen Kapiteln beschrieben wurde, erklärt, sondern auch die Erste Ebene des Gesichtserkennungsteils.

Die erste Ebene sieht man unter Abb.6.3. Sie ist ein detaillierte Version des Big Picture, das sich nur auf den Erkennungsteil konzentriert. Man spricht von einer Iteration, die hier passiert ist. Folgend wird die Vorgehensweise und der Lösungsweg dieser Aufgabe mit Hilfe der 1.Ebene erklären und beschrieben.

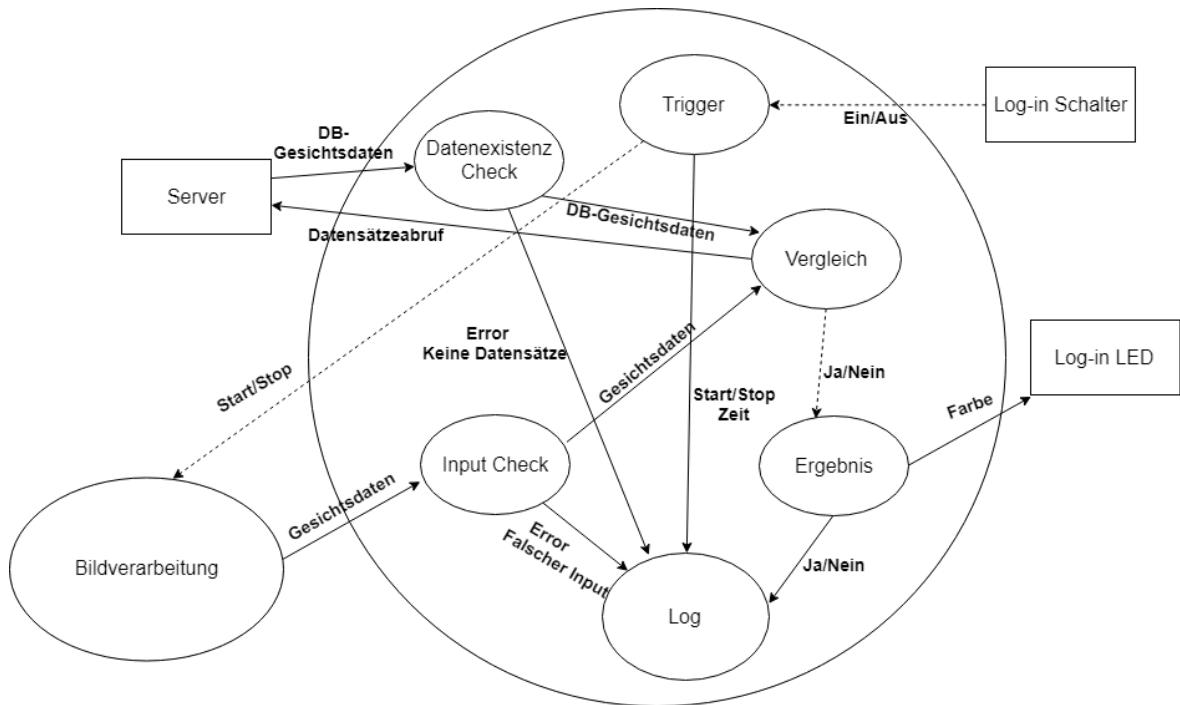


Abbildung 6.3: Erste Ebene

Die wichtigste Elemente der ersten Ebene sind auf der Abbildung 6.4 erklärt:

¹⁵Printed Circuit Board

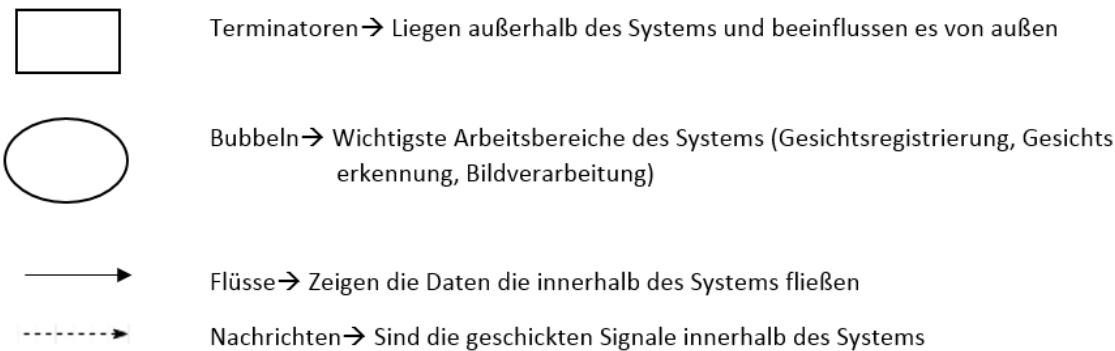


Abbildung 6.4: Erste Ebene Erklärung

Einzelne Arbeitsschritte des Erkennungsprozesses:

In diesem Unterkapitel wird der Vergleich der Gesichter der Personen detailliert erklärt. Es wurden die einzelne Schritte der Arbeit sowie die verwendeten Methoden beschrieben.

1. Erkennungstaster wird gedrückt

Will der Benutzer sich im System einloggen, muss er zuerst den Erkennungstaster drücken. Wird dieser Taster gedrückt, dann kriegt der Benutzer eine Anmeldung, bei der er seine Emailadresse eingeben muss. Um diese Aufgabe zu erledigen, muss das Paket RPi.GPIO importiert werden. Auf diese Weise ist der Zugriff auf den GPIO Pins möglich, damit der Taster Zustand, oder der LED Zustand erkannt werden. Das Paket mysqlDb wird auch benötigt, um Zugriff auf die Datenbank zu haben. Der untere Codeabschnitt 6.1 zeigt wie das programmiert wird:

```

1 if GPIO.input(17):
2     exec(open('Existiert_nExistiert.py').read())
3

```

Listing 6.1: Integration des Tasters

2. Benutzer gibt seine Emailadresse ein

Es ist wichtig zu erwähnen, dass es für die Umsetzung dieses Systems notwendig ist, die verschiedenen Skripte miteinander zu verbinden. Deshalb müssen sie innerhalb anderer Skripte aufgerufen werden. Das wird durch Variablen gemacht. Deshalb ist das Importieren der Pakete *sys* und *os* nötig. Der Benutzer gibt seine Emailadresse an, die danach verwendet wird, um in der Datenbank schneller auf die Benutzerbilddaten zuzugreifen. Es wurde deshalb die Emailadresse(und nicht Vorname, Nachname usw.) gewählt, weil diese Adresse immer eindeutig ist, das heißt, es ist unmöglich, dass zwei Benutzern dieselbe Adresse haben, und dasselbe kann für die andere Benutzerdaten(Vorname, Nachname usw.) nicht gesagt werden. Der untere Codeabschnitt 6.2 zeigt wie das programmiert wird:

3. Die Existenz der eingegebenen Emailadresse des Benutzers in der Datenbank wird geprüft

```

1  for x in myresult:
2      if x[2]==rei.emailiperkrahasm:
3          bool=True
4      else:
5          bool=False
6      os.system('./Log_Erkennung_Email_Nicht_Gefunden.py')

```

Listing 6.2: Überprüfung der Existenz der Emailadresse

Wie es vorher erwähnt wurde, ist dieser Schritt deswegen wichtig, weil falls die eingegebene Email nicht in der Datenbank existiert, muss das System den Bildvergleich nicht machen. Der Benutzer kriegt stattdessen eine Anmeldung, die ihm sagt, dass er noch einmal versuchen kann seine Emailadresse einzugeben. Wird diese neue Adresse auch nicht in der Datenbank gefunden, kann danach dieser Benutzer mit dieser Adresse sich nicht mehr versuchen sich beim System anzumelden.

4. Bild wird gemacht und temporär gespeichert

Gleich nachdem der Benutzer sein Email eingegeben hat, macht die Kamera das Bild. Dieses Bild wird temporär gespeichert, weil es nach dem Vergleich mit dem Bild, das schon in der Datenbank liegt, nicht mehr benötigt wird. Sonst würde es extrem viele Bilddaten in der Datenbank geben, die nur einmal verwendet werden. Stattdessen wird nur der Pfad des Bildes zusammen mit den extrahierten Punkten in der Datenbank gespeichert.

5. Gemachtes Bild wird zugeschnitten

Das von der Kamera gemachtes Bild ist noch nicht bereit zum Vergleich. Um die Punkte richtige zu extrahieren, muss das Bild zuerst zugeschnitten werden. Das heißt, die Dimensionen des Bildes werden reduziert, und somit werden die anderen Objekte auf dem Bild nicht berücksichtigt (Sie würden die Performance der Vergleichsprozess negativ beeinflussen). Das Zuschneiden des Bildes wurde mit Hilfe eines Programms, das Teil der Arbeit der Bildverarbeitungsgruppe ist, gemacht. Wird das Bild zugeschnitten, wird es automatisch auch umbenannt. Das Bild nimmt das Schlüsselwort „New“+die Emailadresse der Person als Name. Der Datentyp ist .jpg .

6. Die Gesichtspunkte werden extrahiert

Das zugeschnittene Bild des Gesichts der Person besteht aus verschiedene Punkte. Insgesamt sind es 68 Gesichtspunkte. Alle diese Punkte müssen für den Vergleichsprozess extrahiert werden. Das heißt, jeder Punkt muss ein X und ein Y Wert haben. Diese Koordinaten müssen ermittelt werden. Dies ist auch eine Aufgabe

der Bildverarbeitungsbranche, und ist extrem wesentlich für das Funktionieren des Erkennungsprozesses.

7. *Wird dieselbe Emailadresse in der Datenbank gefunden, werden die zugehörige Bildinformationen geladen*

Falls aber die vom Benutzer eingegebene Emailadresse schon in der Datenbank liegt, weißt das System Bescheid, dass ein Bilderdatenvergleich stattfinden muss. Um das zu erreichen, werden alle Bildinformationen geladen. Unter Bildinformationen sind die 68 extrahierten Gesichtspunkte zu verstehen. Der kurze Codeabschnitt 6.3 unten zeigt genau wie dies funktioniert:

```

1  if b=="existiert":
2      mycursor.execute(
3          """select * from info i \
4              join person p \
5                  on i.idP=p.idP \
6                  where p.email=%s"""%var1)
7      myresult=mycursor.fetchall()
8      for x in myresult:
9          print(x)
10

```

Listing 6.3: Bildinformationen werden selektiert, wenn Email gefunden wird

Das heißt, die extrahierte Punkte des von der Kamera gemachten Bildes und die extrahierte Punkte, die in der Datenbank gespeichert sind und die den Person mit derselbe Emailadresse wie die vom Benutzer eingegebene Adresse gehören , stehen jetzt endlich zur Verfügung. Ab hier wird angenommen, dass die eingegebene Emailadresse mit mindestens einer in der Datenbank vorhandenen Emailadresse gleich ist, damit der Vergleichsprozess erklärt werden kann.

8. *Die extrahierten Punkte werden in 5 verschiedene Bereiche geteilt*

Bevor der Vergleichsprozess beginnt, müssen die 68 extrahierten Punkte geteilt werden. Die 5 großen Bereiche sind das Gesicht, die Nase, das rechte Auge, das linke Auge und der Mund. Jeder Bereich hat seine eigenen Punkte. Die ersten 28 Punkte gehören dem Gesicht, die nächste 9 gehören der Nase, die nächste 6 dem rechten Auge, die nächste 6 dem linken Auge und die letzte 19 gehören dem Mund. Durch dieser Prozess wird die Vergleichsqualität stark erhöht, weil die Punkte der Nase werden genau mit der Punkte der anderen Nase verglichen, und nicht mit dem von dem Augen beispielsweise.

Der untere Codeabschnitt 6.4 zeigt wie dieser Schritt für die ersten 27 Punkte programmiert wird(nur um eine Idee zu haben, wie es funktioniert):

```

1  for pika in range(0,27):
2      dis=math.sqrt(abs((vleratx.item(pika)
3          -float(res[vx])))*
4          abs((vleratx.item(pika)-float(res[vx])) ) + (

```

```

5   abs((vleraty.item(pika)-float(res[vy])))*
6   abs((vleraty.item(pika)-float(res[vy])) ) ))

```

Listing 6.4: Teilung der extrahierten Punkte in 5 Bereiche

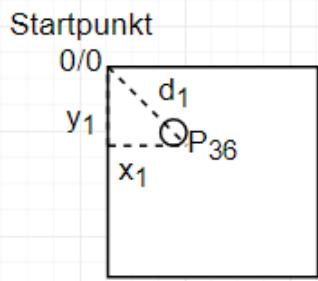
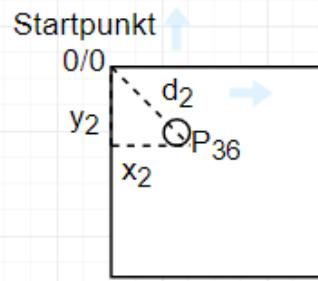
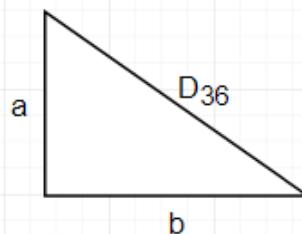
9. Die Bildinformationen der beiden Bilder werden verglichen

Die wichtigste Aufgabe ist der Vergleich der Bildinformationen. Es werden die extrahierten Gesichtspunkte der bereits gemachten Bilder mit der Gesichtspunkte in der Datenbank verglichen. Selbstverständlich müssen, wie es vorher erwähnt wurde, die beiden Personen, deren Gesichtspunkte verglichen werden dieselbe Emailadresse haben. Das wichtigste Paket, Open CV, wird für diesen Vergleich benötigt. Nachdem dieses Paket importiert wurde, kann der Vergleich beginnen. Es gibt verschiedene Schritte, die erfolgen, und die nun erklärt werden:

- Startpunkt bzw. Origin wird festgelegt

In jeden Gesichtsbild wird ein Startpunkt bzw. eine Origin für den Vergleich festgelegt. Dieser Punkt ist der Punkt mit der Koordinaten (0/0). Es wird der Abstand von jedem anderen Gesichtspunkt zu dieser Origin berechnet. Dieser Abstand wird dann mit dem entsprechenden Abstand des zweiten Bildes verglichen. Diese Abstände müssen miteinander übereinstimmen(zum Beispiel ein Punkt vom rechten Augen im ersten Bild muss mit dem zugehörigen Punkt im zweiten Bild verglichen werden, und nicht mit einem Punkt von dem Augen in ersten Bild).

- Die Abstände jedes anderen Punktes des gemachten Bildes vom Startpunkt werden berechnet.
- Nachdem der Startpunkt festgelegt wurde, beginnt der Vergleich der Abstände des gemachten Bildes(Punkt-Startpunkt) und der Abstände, die in der Datenbank gespeichert sind.


Bild 1

Bild 2


$$a = x_1 - x_2 \quad b = y_1 - y_2$$

$$D_{36} = \sqrt{a^2 + b^2}$$

Abbildung 6.5: Vergleichsmethode zweier Punkte

Auf Abbildung 6.5 wird die verwendete Vergleichsmethode dargestellt. Der Startpunkt hat die Koordinaten (0/0) und befindet sich links oben. Jeder Punkt hat seine bestimmten Koordinaten, zum Beispiel, der Punkt 36 des ersten Bildes hat die Koordinaten x1 und y1, und der Punkt 36 des anderen Bildes hat die Koordinaten x2 und y2. Um den Abstand des Punktes 36 zu berechnen, wird die pythagoreische Formel verwendet. Die Differenz der x-Koordinaten und die Differenz der beiden y-Koordinaten sind die beiden Katheten(a und b auf der Abbildung). Um die Hypotenuse zu finden wird das Quadrat beider Katheten addiert, und am Ende wird die Wurzel des Ergebnisses berechnet. Das Ergebnis ist der Abstand des Punktes 36 vom Startpunkt. Dieselbe Methode wird für jeden Punkt verwendet. Das obige Beispiel vergleicht die Abstände der Punkte von 2 verschiedenen Bildern. Das Vergleichsprinzip ist dasselbe wie bei dem Vergleich der Punkteabstände des gemachten Bildes und der in der Datenbank gespeicherten Punkteabstände. Es ist einfach leichter erklärbar und verstehbar, wenn 2 Bildern verglichen werden. Die Abstände aller Punkte aller Bereiche werden in einer Variable gespeichert.

- Es wird der maximale Abstand für jeden einzelnen Bereich berechnet

Nachdem die oben erklärten Abstände kalkuliert wurden, wird der maximale Abstand für jeden Bereich(Gesicht, Nase, linke Auge, rechte Auge, Mund)

benötigt. Diese Maximumwerte sind wesentlich für den Vergleich. Warum das so ist wird später erklärt.

Der untere Codeabschnitt 6.5 zeigt wie dieser Schritt für die ersten 27 Punkte implementiert ist:

```

1  MAX = [0,0,0,0,0]
2  for pika in range(0,27):
3      dis=math.sqrt(abs((vleratx.item(pika)
4          -float(res[vx])))*
5          abs((vleratx.item(pika)-float(res[vx])) ) + (
6              abs((vleraty.item(pika)-float(res[vy])))*
7                  abs((vleraty.item(pika)-float(res[vy])) ) )
8      if(dis < min):
9          min = dis
10     if(dis > max):
11         max = dis
12     global MAX
13     MAX[0]=max
14     print("Max:\n%s\n"%MAX[0])
15     min=10.0
16     max=0.0
17     dis=0

```

Listing 6.5: Berechnung der Maximumwerte

Zuerst wurde eine Array MAX erstellt, das 5 Elemente enthält. Jedes Element gehört zu einem der 5 Bereiche. Beispielsweise der Maximumwert an der Stelle 0(erste Element) ist der Maximumabstand, der bei dem Gesichtsbereich gefunden wurde. Die Funktion Max in Python, die den größte Wert automatisch findet, wurde verwendet. Jeder Abstand wird mit dem Maximumwert verglichen(Maxwert beginnt am Anfang bei 0). Ist er größer, dann wird dieser Abstand der neue Maximumwert. So wird der Maximumwert ermittelt.

- Es wird das geometrische Mittel der Abstände aller Bereiche berechnet

”Das geometrische Mittel der Abstände aller Punkte (nicht in einzelnen Bereichen) ist auch wichtig für den Vergleich. Es gibt 68 Gesichtspunkten insgesamt, das heißt, 68 Abstände von dem Startpunkt. Um das geometrische Mittel von n Zahlen x_1, x_2, \dots, x_n zu ermitteln, muss man deren Produkt bilden und von diesem die n-te Wurzel ziehen.”[18]

Auf Abbildung 6.6 wird die Formel, die sich ergibt gezeigt:

$$G(x_1, x_2, \dots, x_n) = \bar{x}_{\text{geom}} = \sqrt[n]{x_1 \cdot x_2 \cdots x_n}$$

Abbildung 6.6: geometrisches Mittel

Der untere Codeabschnitt 6.6 zeigt wie dieser Schritt für die ersten 27 Punkte implementiert ist(nur um eine Idee zu haben, wie es funktioniert):

```

1   for pika in range(0,27):
2       dis=math.sqrt(abs((vleratx.item(pika)
3           -float(res[vx])))* abs(
4               (vleratx.item(pika)-float(res[vx])) ) + (
5                   abs((vleraty.item(pika)-float(res[vy])))*
6                       abs((vleraty.item(pika)-float(res[vy])) ) ))
7       gm *= dis
8       vx+=2
9       vy+=2
10      gm=gm** (1/27)
11      print("GM:%s\n"%(gm))
12      gm=1
13      dis=0

```

Listing 6.6: Berechnung des geometrischen Mittels für die erste 27 Punkte

- Die ermittelte Maximumwerte und die geometrischen Mittel der Abstände der Punkte des gemachten Bildes werden mit der Maximumwerten und den geometrische Mitteln der Abstände der, in der Datenbank, gespeicherten Punkte verglichen

(a) Beschreibung

Wenn die angesprochene Maximumwerte und die geometrische Mittel der Abstände der Punkte des gemachten Bildes und die geometrische Mittel der Abstände der in der Datenbank gespeicherten Punkte gleich oder sehr ähnlich sind, dann wird von der selben Person gesprochen und die Erkennung ist erfolgreich. Deshalb werden einige Abgrenzungen bestimmt, wann die Erkennung als erfolgreich gilt und wann nicht.

(b) Mögliche Fälle

Insgesamt gibt es 5 Bedingungen bzw. 5 Fälle, die eintreten können und die wichtig sind. Diese Bedingungen wurden mit Hilfe vieler gemachten Tests, über die später gesprochen wird, bestimmt. Ist einer dieser fünf Bedingungen wahr, ist die Erkennung nicht erfolgreich und es wird nicht von derselben Person gesprochen. Diese Fälle werden unten mit Hilfe von Bildern erklärt. Es werden die Maximumwerten jedes Bereichs(Gesicht, Nase, linke Auge, rechte Auge, Mund, also 5 insgesamt), sowie das geometrische Mittel aller 68 Punkten berücksichtigt.

- i. 1.Fall: Mindestens 2 Maximumwerte sind größer als 0.06

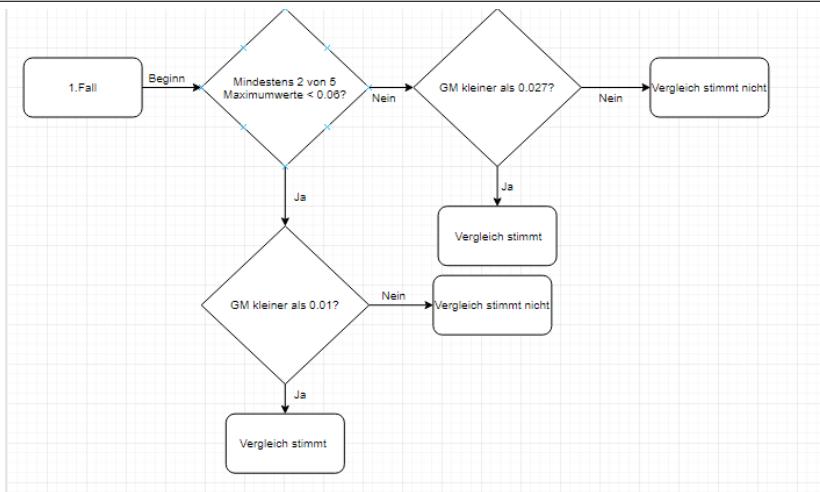


Abbildung 6.7: 1.Fall

Sind 2 oder mehrere Maximumwerte größer als 0.06, wird das geometrische Mittel betrachtet. Ist der Wert größer als 0.01, gibt es keine Übereinstimmung. Wenn weniger als 2 Werte größer als 0.06 sind, muss auch das geometrische Mittel kleiner als 0.027 sein, um eine erfolgreiche Erkennung zu haben.

Auf Abbildung 6.7 ist ein Flussdiagramm dargestellt, das die Abfolge dieser Bedingung beschreibt.

Der Codeabschnitt 6.7, der zeigt wie das funktioniert:

```

1  global count
2  count = 0
3  for i in range(0, len(MAX)):
4      if MAX[i] > 0.06:
5          count+=1
6          status1=True
7          status11=True
8          status2=True
9          status22=True
10
11 #Wenn JA: Nur wenn gmT kleiner als 0.01
12 sind sie richtig, sonst FALSCH
13
14 if count ==1:
15     global status11
16     status11=False
17 if count >= 2:
18     global status22
19     status22=False
20     if gmT > 0.01:
21         global status1
22         status1 = False
23
24 #Wenn NEIN: gmT muss kleiner als 0.027 sein,
  
```

```

25         um RICHTIG zu sein
26     else:
27         if gmT > 0.027:
28             global status2
29             status2 = False

```

Listing 6.7: 1.Fall

- ii. 2.Fall: Genau einen Maximumwert größer als 0.07 ist und kleiner als 0.084

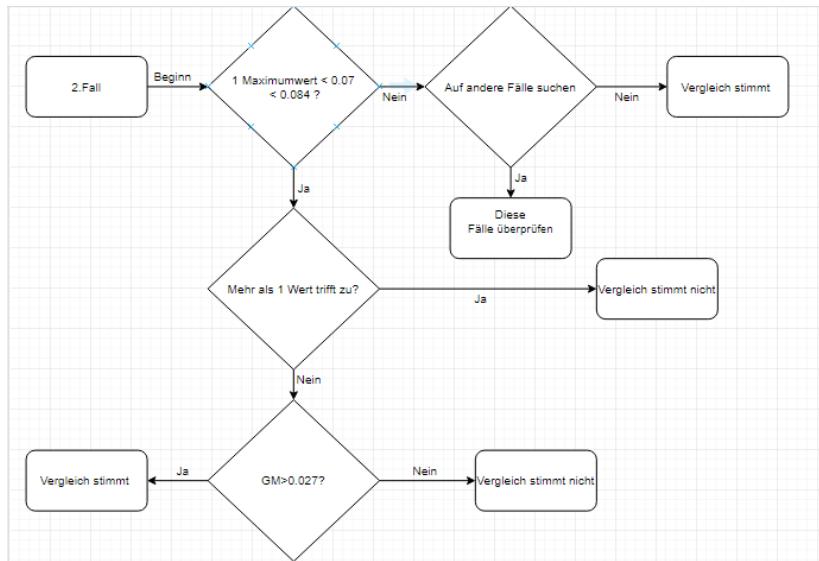


Abbildung 6.8: 2.Fall

Stimmt dieser Bedienung, wird wieder überprüft, ob das geometrische Mittel kleiner als 0.027 ist oder nicht. Wenn ja, dann gibt es keine Übereinstimmung. Ist aber dieser Bedienung wahr für mehr als einen Wert, gibt es keine Übereinstimmung.

Auf Abbildung 6.8 ist ein Flussdiagramm dargestellt, das die Abfolge dieser Bedingung beschreibt.

Der Codeabschnitt 6.8, der zeigt wie das funktioniert:

```

1  count = 0
2  for i in range(0,len(MAX)):
3      if MAX[i] > 0.07 and MAX[i]< 0.084:
4          count+=1
5          status3=True
6          status33=True
7          status4=True
8
9      #Wenn JA :Wenn gmT kleiner als 0.027 ist passt,
10     sonst FALSCH
11

```

```

12     if count == 1:
13         global status33
14         status33=False
15     if gmT > 0.027:
16         global status3
17         status3 = False
18
19 #Wenn 2 oder mehr: FALSCH
20
21 elif count > 1:
22     global status4
23     status4 = False

```

Listing 6.8: 2.Fall

iii. 3.Fall: Mindestens einen Maximumwert ist größer als 0.084

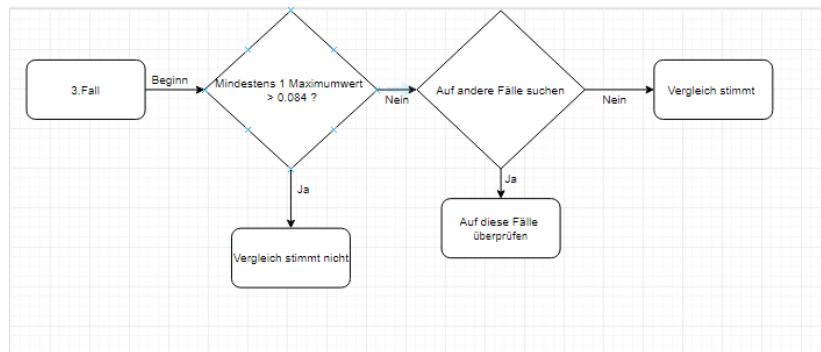


Abbildung 6.9: 3_Fall

Trifft dieser Bedienung zu, stimmt die Erkennung nicht und der Person wird nicht vom System erkannt.

Auf Abbildung 6.9 ist ein Flussdiagramm dargestellt, das die Abfolge dieser Bedingung beschreibt.

Der Codeabschnitt 6.9, der zeigt wie das funktioniert:

```

1  for i in range(0,len(MAX)):
2      if MAX[i] > 0.084:
3          count+=1
4          status5=True
5
6 #Wenn JA: FALSCH
7
8      if count >=1:
9          global status5
10         status5=False

```

Listing 6.9: 3.Fall

(c) Andere Abgrenzungen und Entscheidungen

Trifft eine von diesen 2 unterstehenden Kombinationen zu, ist weitere Überprüfung (geometrisches Mittel) nicht mehr nötig und es gibt keine Übereinstimmung.

- i. Ein Maximumwert größer als 0.06 und ein Maximumwert größer als 0.07
- ii. Zwei Maximumwerte größer als 0.06 und ein Maximumwert größer als 0.07

Der Codeabschnitt 6.10, der zeigt wie das funktioniert:

```

1 if status1==False or status2==False
2 or status3==False or status4==False
3 or status5==False or
4 (status11==False and status33==False)
5 or (status22==False and status33==False):
6 print("Problem!!!")
7 os.system
8 ('./Log_Erkennung_Person_Existiert_Nicht.py')

```

Listing 6.10: Andere Abgrenzungen und Entscheidungen

- Ist die Erkennung erfolgreich, kriegt der Benutzer eine Information, dass er im System erfolgreich angemeldet ist.

Wenn keinen Fall bzw. Bedienung zutrifft, ist die Authentifizierung erfolgreich. Der Benutzer kriegt danach eine Information, die ihm Bescheid gibt, dass die Authentifizierung erfolgreich war und dass er weitermachen darf. Dazu wurde eine gelbe LED verwendet, die leuchtet.

Der Codeabschnitt 6.11, der zeigt wie das funktioniert:

```

1 else:
2 b="existiert"
3 print("Eingeloggt!!!")
4 os.system('./Log_Erkennung_Person_Existiert.py')

```

Listing 6.11: Erkennung ist erfolgreich

Andere wichtige Punkte und Abgrenzungen

Es gibt auch einige Punkte, die erwähnt werden müssen, und die zur Verwendung des Systems wesentlich sind:

1. Distanz der Benutzer von der Kamera

Damit der Vergleich und die Erkennung der Gesichter erfolgreich und so genau wie möglich ist, gibt es eine Distanz, die der Benutzer von der Kamera stehen muss. Diese Distanz ist 1 Meter, dem Benutzer wird das durch einer Linie auf dem Boden angezeigt.

2. Unbekannte Emailadresse eingegeben

Wenn der Benutzer nach dem Drücken des Tasters eine falsche Emailadresse eingibt, wird er informiert, dass er eine zweite Chance bekommen wird, die richtige Emailadresse einzugeben(Vielleicht hat er zum Beispiel einen Fehler beim Eintippen gemacht). Ist die Adresse noch immer unbekannt oder inkorrekt, beginnt der Vergleichsprozess nicht und der Benutzer kann sich nicht anmelden.

3. Kein eindeutiges Gesicht von der Kamera nach der Bildaufnahme gefunden

Wenn nach der Bildaufnahme kein eindeutiges Gesicht erkannt wird, wird ein zweites Bild gemacht. Der Benutzer wird durch eine Meldung erinnert, dass er gerade und an der Linie stehen muss.

4. Mehrere Gesichter von der Kamera nach der Bildaufnahme gefunden

Wenn nach der Bildaufnahme mehrere Gesichter erkannt werden, wird wieder ein zweites Bild gemacht. Werden wieder mehrere Gesichter erkannt, kann das System nicht überprüfen, ob sich der Benutzer anmelden darf oder nicht (Nicht Ziel).

5. Benutzer vom System nicht erkannt

Ist die Authentifizierung nicht erfolgreich, leuchtet eine rote LED um den Benutzer zu zeigen, dass er sich nicht anmelden darf. Er kann aber alles nochmals probieren(Erkennungstaster drücken usw.), oder er kann sich registrieren.

6. Verarbeitungszeit des Systems

Jedes intelligente System braucht einer Verarbeitungszeit. Der Erkennungsprozess braucht ungefähr 7 bis 8 Sekunden. Das ist aber auch von der Geschwindigkeit des Benutzers abhängig.

Das Testen

Eine der wichtigsten Arbeitspakete des Erkennungsteils war das Testen des Vergleichsverfahrens. Um die Vergleichsbedingungen und Begrenzungen, die weiter oben erwähnt wurden, zu finden und zu bestimmen, war es wesentlich und extrem notwendig, viele Tests zu machen. Zuerst hat der Prozess des Testens mit verschiedene Bilder einer Person begonnen. Danach wurden Bilder verschiedener Personen verwendet, um eine

Toleranz zu finden, die in der Bedienungen verwendet werden kann. Nachdem die Gesichtspunkte von zwei Bildern getestet und verglichen waren, sind die Ergebnisse auch mit den in der Datenbank gespeicherten Punkten getestet worden. Am Ende des Testens sind die vorher erklärte Bedingungen definiert und formuliert worden.

Das Logging

Jedes große System braucht Logging. Mit Hilfe einer Logdatei (englisch: log file) können alle oder bestimmte Aktionen von Prozessen auf einem Computersystem automatisch protokolliert werden. Das heißt, diese Logdatei kann vom Benutzer verwendet werden, um Fehlermeldungen zu erhalten, um den Fehler zu verstehen, um den Zeitpunkt verschiedener wichtiger Ereignissen zu erfahren und vieles mehr. Alle diese Vorteile und Merkmale des Logging waren mehr als genug zum entscheiden, dass die Protokollierung der Ereignisse des Systems ein wichtiges Ziel sind. Das war der Grund, warum der Gesichtserkennungsteil auch protokolliert.

Es wurden die wichtigsten Ereignisse mitprotokolliert:

1. Der Zeitpunkt, wann der Erkennungstaster gedrückt wurde

Es ist sehr wichtig für die Admins des Systems zu wissen, wann der Erkennungstaster gedrückt wurde, bzw. wann die Gesichtserkennungsprozess beginnt. Diese Zeit kann dafür verwendet werden, um die totales Dauer des Erkennungsprozesses zu bestimmen.

2. Der Zeitpunkt, wann der Benutzer erfolgreich angemeldet war

Dieses Ereignis ist ein sehr wichtiges, weil es der Schlusspunkt des Erkennungsteils ist. Es kann auch als Beweis verwendet werden, dass der Benutzer im System schon registriert war.

3. Der Zeitpunkt, wann der Benutzer sich nicht erfolgreich anmeldet

Wie vorher erwähnt, ist eine Log-Datei eine sehr gute Methode, wichtige Fehlermeldungen sehr schnell zu erhalten. In diesem Fall wird gespeichert, wann der Benutzer sich nicht erfolgreich anmeldet.

4. Der Zeitpunkt, wann die eingegebene Emailadresse in der Datenbank nicht gefunden wird

Gibt der Benutzer eine falsche oder eine nicht existierende Emailadresse ein, wird dieses Ereignis geloggt. Das ist extrem wichtig für die Administratoren, weil ohne eine vom Benutzer eingegebene Emailadresse, die in der Datenbank schon existiert, beginnt die Vergleichsprozess für die Erkennung gar nicht. Auf diesem Weg wissen die Administratoren, wo der Fehler ist, und der Benutzer wird darüber informiert.

Wie wurde das Logging gemacht?

Das Logging des Gesichtserkennungsprozesses war nicht sehr komplex in Python. Es gab eine Tabelle in der Datenbank, die für dieser Funktion geeignet war. In dieser Tabelle wurden alle Informationen, die protokolliert werden müssen, gespeichert. Mit Hilfe einer Python Skript wurde dann die Verbindung aller Skripte, die die notwendige Informationen senden mit dieser Log Tabelle, die in der Datenbank liegt, gemacht. Es wurde danach ein Programm für jeder Ereignis, das oben erwähnt war, erzeugt. Diese Programme ermöglichen die Verbindung mit der Datenbank, und zeigten eine Anmeldung. Beispielweise wurde in der Datenbank notiert, dass es ein Problem bei der Gesichtserkennung gab(Taster nicht gedrückt..). In den Hauptskript wurde dann dieses einzelne Programm aufgerufen.

Die Codeabschnitte 6.12 und ?? zeigen, wie das programmiert wurde:

Der Taster wird gedrückt:

```
1 if GPIO.input(17):
2     os.system('~/Log_Erkennung_TasterGed.py')
```

Listing 6.12: Log für gedrücktes Taster

Zeitpunkt wird gespeichert:

```
1 logging.basicConfig(filename=log_file_path)
2 if(log_to_db):
3     logging.getLogger('').addHandler(logdb)
4 log=logging.getLogger('Gesichtserkennung')
5 log.setLevel(log_error_level)
6 test_var='Taster gedruckt'
7 log.error('This event occurred: %s' % test_var)
```

Listing 6.13: Log für den Zeitpunkt

6.3 Herausforderungen, Probleme, und deren Lösung

Während dieser Arbeit musste man einige Problemen und Herausforderungen lösen.

- *OpenCV Installation*

Die größten Probleme hat es bei der Installation von OpenCV gegeben. Diese Installation hat sehr lang gedauert und es war sehr schwer zu bestimmen, welche Paketen ausgelassen werden sollten und welche nicht. Dieses Problem wurde nach vielen Tests gelöst, durch das Kopieren von einer anderen SD-Karte, auf der OpenCV bereits installiert war. CMake und Make waren sehr wichtige Pakete, mit denen diese Aufgabe erledigt wurde.

- *Fritzing*

Das Problem beim Fritzing war, dass jedes Mal wenn das Programm beendet wurde, es nicht mehr geöffnet werden konnte. Es fehlten entweder die Pfade oder die Bauteile, die für die Schaltung notwendig waren. Die Lösung war eigentlich sehr leicht. Das Programm wurde komplett gelöscht und dann wieder im System installiert, und danach wurde die ganze Schaltung gemacht, ohne das Programm zu beenden. Sonst wären dieselben Probleme wieder aufgetreten.

- *Kurzschluss beim Raspberry Pi*

Was noch passiert ist, ist das es beim Anfassen vom Raspberry PI einen Kurzschluss gab. Der Raspberry PI war kaputt und musste ersetzt werden. Glücklicherweise konnte die SD-Karte erfolgreich kopiert werden und deshalb sind alle Daten und Informationen gespeichert.

- *Paket FaceRecognition Installation*

Die Installation des Paketes FaceRecognition konnte nicht erfolgreich gemacht werden, weil die dlib Pakete auch gebraucht wurden. Wahrscheinlich aufgrund des zu geringen RAMs kann man diese Pakete nicht installieren. Dieses Problem wurde noch nicht gelöst.

- *Probleme bei der Aufruf der Skripten*

Am Beginn war das Umgehen mit dem Aufruf der Skripten sehr schwer. Es war schwer zu verstehen wie das genau funktioniert, weil die große Variablenanzahl die Arbeit kompliziert machte. Dieser Anzahl wurde reduziert und es wurden viele Recherchen über die korrekten Verwendung des *sys* Paketes gemacht um das Problem zu lösen.

- *Schlechtere Vergleichsqualität*

Am Beginn des Testens des Vergleichs waren die Ergebnisse nicht stabil. Es konnte keinen Zusammenhang gefunden werden. Beispielsweise waren die Maximumswerte der Abstände bei Bildern, die derselben Person gehörten größer als bei Bildern, die verschiedenen Personen gehörten. Nach viele Tests wurde der Grund verstanden. Die 68 Punkte mussten in 5 verschiedene Teile(Gesicht, Nase, linke Auge, rechte Auge, Mund) zerlegt werden. In dieser Weise wird ein Punkt, der dem Mund gehört nicht mit einem Punkt, der der Nase gehört verglichen.

- *Probleme bei Einteilung der Gesichtspunkte in 5 Bereichen*

Der Prozess der Einteilung der Gesichtspunkte in dem verschiedenen Bereiche war auch schwierig. Ein großes Array mit 68 Elementen wurde verwendet. Es war sehr kompliziert auf einem Index dieses Array zu zugreifen. Deshalb wurde dieses große Array in 5 kleinere Arrays zerlegt. Auf diese Weise war es viel leichter zu verstehen, wie die Indizien funktionieren und wie man dort zugreifen kann.

- Probleme bei der Berechnung des geometrischen Mittels

Die Berechnung des geometrischen Mittels war auch nicht leicht. Nachdem die Formel gefunden war, musste es in dem Programm integriert werden. Die Schwierigkeit war bei dem Ziehen von der N-Wurzel. Sollte das $n = 67$ oder 68 sein? Das war von dem Index abhängig. Am Ende wurde verstanden, dass das Produkt aller Abstände hoch $1/68$, die richtige Lösung ist.

- Probleme mit der Datenbank

Probleme hat es noch beim Zugriff auf der in der Datenbank gespeicherten Gesichtspunkte gegeben. Es war sehr kompliziert zu verstehen, wie und wo die Punkte genau gespeichert waren (Auf welchem Index die X-Werte, auf welchem die Y-Werte). Es wurde deshalb Hilfe von dem Mitarbeiter, der sich mit der Datenbank beschäftigt hat, gebraucht, und danach war alles klarer und verständlich.

- Probleme mit GIT

Am Ende müssen die Probleme mit Git erwähnt werden. Es gab viele Fälle, wo es Konflikten mit den Dateien gab. Man muss immer zuerst Pull machen, und danach Push. Das ist aber oft nicht passiert. Es wurde in einer Datei gearbeitet, die aber von einem anderen Person schon geändert wurde. Deshalb gab es Probleme bei dem Push. Oft mussten die ganzen Dateien gelöscht werden, und danach wieder vom Git geholt werden. Es wurde am Ende in der Gruppe darüber gesprochen und das Problem gelöst.

6.4 Projektmanagement und Controlling

In Bezug auf Projektmanagement und Controlling wurde die Methode des Fehlerbaums verwendet. Diese ist eine berühmte Methode um die Aufwandschätzung zu berechnen und um eine Fehlerursache zu finden. Das Problem wird in kleinen Teile geschnitten damit es klarer wird. Es wurde eine detaillierte Soll-Ist Analyse gemacht, die bei der neuen Aufgabeteilung und einer Fehlerbaumanalyse zu erstellen sehr geholfen hat.

Die Fehlerbaumanalyse wird verwendet, um die Zuverlässigkeit von technischen Systemen zu testen. Die Fehlerbaumanalyse nimmt als Ausgangspunkt nicht eine einzelne Systemkomponente, sondern das potenziell gestörte Gesamtsystem. Die Fehlerbaumanalyse baut auf der sogenannten negativen Logik auf. Das heißt, der Fehlerbaum beschreibt eine Ausfallsfunktion die bei dem Zustand logisch-1 einen Ausfall ausdrückt, bei logisch-0 liegt ein funktionsfähiges System vor. Sie gehört zu den "Top-Down"-Analyseformen im Risikomanagement. In einem ersten Schritt wird daher das Gesamtsystem detailliert und exakt beschrieben. Darauf aufbauend wird analysiert, welche primären Störungen eine Störung des Gesamtsystems verursachen oder dazu beitragen können. Ausgangspunkt ist hierbei zunächst ein einziges unerwünschtes Ereignis,

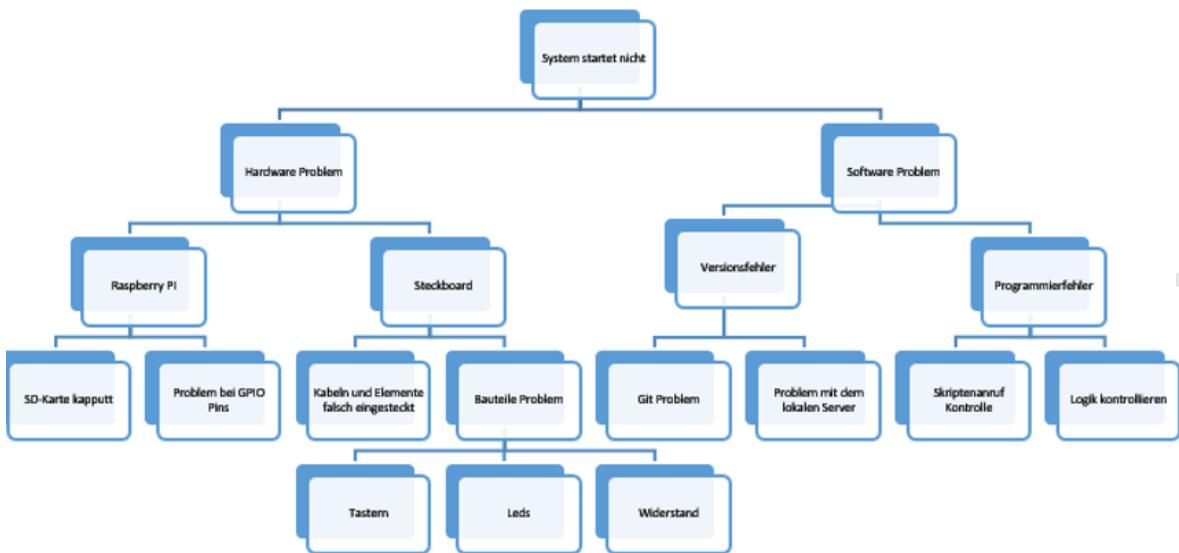


Abbildung 6.10: Fehlerbaum

welches an der Spitze des Fehlerbaums steht, das sogenannte Top-Ereignis. Das Top-Ereignis resultiert in der Regel aus einer Risikoanalyse bzw. Szenarioanalyse. In der einfachsten Form besteht er aus folgenden Elementen: Entscheidungsknoten (E), die Entscheidungen kennzeichnen, Zufallsknoten, die den Eintritt eines zufälligen Ereignisses darstellen sowie aus Ergebnisknoten (R), die das Ergebnis von Entscheidungen oder Ereignissen darstellen.

Auf der Abbildung 6.10 wird ein Fehler in den Gesichtserkennungsteil des Systems beschrieben. Zuerst muss überprüft werden, ob der Fehler bei der Hardware oder Software liegt.

Hardware Problem

Ist es ein Problem in Hardware, wird der Raspberry Pi und die Steckboard getestet. Beim Raspberry kann es entweder Probleme mit dem SD-Karte oder mit der GPIO Pins geben. Bei dem Steckboard sind entweder die Kabel und die Verbindungen falsch eingesteckt, oder die Bauteile sind kaputt(Taster, Widerstand, LED, Kamera).

Software Problem

Wenn es ein Softwareproblem ist, sind entweder die Versionen vermischt(Problem mit GIT), oder es gibt Programmierfehlern. Diese Programmierfehlern können bei der Logik(Algorithmen) oder bei der Syntax auftreten(Skripten Aufruf, Variablen usw.)

6.5 Ergebnisse

Die finale Ergebnisse sind die folgende:

1. Systemaufbau
Das ganze System wurde zusammen mit der Hardware aufgebaut.
2. Digitale Darstellung des Systemaufbaus
Der Schaltplan des Systems wurde durch Fritzing digital dargestellt.
3. Erkennung der Gesichter
Es wurde eine Gesichtserkennung erreicht, die in 80 Prozent der Fälle funktioniert.
4. Aufnahme der Benutzer Gesichtsdaten mithilfe der Emailadresse
Gibt der Benutzer seine Emailadresse ein, werden die zugehörigen Gesichtsdaten von der Datenbank geladen, um den Vergleich zu machen.

6.6 Was wurde gelernt und welche Verbesserungsmöglichkeiten gibt es

Während dieser Diplomarbeit wurde viel wichtiges gelernt und verstanden. Die wichtigsten Lektionen werden unten genannt und erklärt.

- Zeitmanagement

Die Wichtigkeit der Zeit wurde wirklich verstanden. Man hatte nicht mehr die Möglichkeit, Dinge immer für den letzte Moment zu lassen, weil es einfach zu viel ist. Ohne einen detaillierten Zeitplan und ohne eine systematische Arbeit würde dieses Projekt nicht fertig sein.

- Wie man mit OpenCV und mit Bilderdaten arbeitet

Es wurde gelernt, wie man überhaupt mit Bildinformationen arbeitet, und wie das Paket OpenCV funktioniert. In Zukunft wenn es um Arbeiten mit Bilderdaten geht, ist das Basiswissen vorhanden.

- Wichtigkeit des Testens des Systems

Wie wichtig der Test des Prozesses ist wurde auch hier verstanden. Vorher wurde gedacht, dass das Testen des Systems nur eine zusätzliche und unwichtige Arbeit ist. Es war aber nur mit Hilfe des Testens möglich, den Vergleichsprozess erfolgreich zu realisieren und Fehler zu finden. Ein System, das vorher nicht getestet wurde, kann kein gutes System sein.

- Gruppenarbeit und Verantwortlichkeiten übernehmen

Die Rolle der Gruppe ist auch sehr wichtig. Geht es den Mitarbeitern miteinander gut und sie sind gut gelaunt, sind sie auch motivierter zum Arbeiten. Die Kommunikation soll nie fehlen und die Entscheidungen des Projektleiters sollen akzeptiert werden. Man muss auch seine Verantwortlichkeiten selbst übernehmen und seine Arbeit mit seiner Stärken erledigen. Das wurde alles auch während der Arbeit gelernt.

- Wichtigkeit der Planung und der Begrenzungen

Es wurde schon gelernt, dass die Planung eines Projekts sehr wichtig war, aber mit Hilfe dieser Diplomarbeit wurde ganz genau verstanden, warum das so ist. Ohne die Big Pictures und die Erste Ebene wurde der Aufwand der Arbeit viel grösser, und die Zeit würde sicherlich nicht genügen. Die Begrenzungen sind auch wichtig, weil ohne die würde man Zeit an etwas verlieren, das am Ende nicht realisierbar oder nicht nötig ist.

- Wie man mit Fehlern umgeht

Außerdem wurde verstanden und gelernt, wie man mit Fehlern umgehen soll. Pessimismus und das Aufgeben sind nicht die Lösungen. Man muss optimistisch sein und viele verschiedene Wege suchen, wie man den Fehlern finden und dann reparieren kann. Fremde Hilfe ist auch irgendwann notwendig.

Verbesserungsmöglichkeiten sind auch erkannt worden. Diese sind:

- Besseres Zeitmanagement

Die Arbeit musste ein bisschen früherer begonnen werden, und es muss besser verteilt werden. Es muss verstanden werden, dass sich hinter einem schlechten Projekt eine schlechte Zeitmanagement versteckt.

- Backup Verfahren

Nächstes Mal wird die Arbeit öfter gespeichert. Back-Ups werden täglich gemacht, damit nichts verloren geht. Man denkt, dass das lokale Speichern genug ist. Was ist aber, wenn der PC kaputt wird? Das wird in die nächste Arbeiten berücksichtigt.

Kapitel 7

Ausblick

In diesem Kapitel wird kurz erklärt, was mit dem Projekt in der Zukunft passieren wird?

7.1 Wie geht es weiter ?

Dieses Projekt wird nicht in der Schule implementiert, weil die Schule es nicht angenommen hat. Dieses Projekt war zu kompliziert, um in die Schulinfrastruktur implementiert zu werden. Nach der Diploma Präsentation wird dieses Projekt abgeschlossen.

7.2 Was passiert mit dem Ergebnis?

Das Ergebnis wird von der Schule betrachtet, aber nichts besonders anders und nicht weiterentwickelt von dieser Gruppe.

7.3 was passiert mit dem Prototyp?

Der Prototyp wird im Lager der Schule eingeschlossen, da kein Interesse daran hat.

7.4 Gibt es Folgeprojekte?

Momentan gibt es keine, aber die anderen Gruppen mit dem Schwerpunkt Systemtechnik können bzw. haben die Möglichkeit mit dem Projekt weiterzumachen.

7.5 Gibt es Interessanten?

Sowohl von der Schule als auch von anderen Unternehmen gibt es keine Interessengruppen, die dieses Projekt finanzieren möchten.

Kapitel 8

Handbuch

Dieses System ist einfach zum Benutzen und man muss kein technologisches Wissen zur Verwendung haben. Das System besteht aus zwei Teilen:

- Einloggen bzw. Erkennung
- Registrierung

Damit das System weiß welche Funktion angerufen wird, gibt es zwei Taster eine für jede Funktion. Diese Taster sind erkennbar durch die Namenzeichen, die sie haben. Wenn jemand eingeloggt bzw. registriert will, soll der Taster der beibiegen Funktion gedrückt werden.

Einloggen

Der Benutzer kann sich selbst einloggen, ohne die Hilfe des Admins. Nach dem Drucken des Tasters soll der Benutzer seine erforderlichen Daten eingeben. Danach muss er vor der Kamera stehen. Nachdem dass das System ein Bild von ihm macht, wird überprüft und erkennt ob dieser Benutzer schon registriert ist. Am Ende wird im Bildschirm gezeigt ob der Benutzer eingeloggt ist oder nicht. Als Hilfe sind auch die LEDs, die wenn alles in Ordnung läuft, soll ein grüne LED leuchten, sonst leuchtet die rote LED als Fehlermeldung.

Registrierung

Immer wenn ein neuer Benutzer registriert wird, sollte ein Admin dabei sein. Die Rolle des Admins ist zur Kontrolle ob alles wie geplant geht. Nach dem Drucken des Tasters sollen die folgenden Anforderungen von dem Benutzer eingegeben werden. Das System wird im Bildschirm zeigen, was der Benutzer eingeben sollte. Als letzter Schritt muss der Benutzer vor der Kamera bleibend stehen, damit ein Bild von ihm gemacht wird. Am Ende wird im Bildschirm gezeigt ob der Benutzer registriert wurde oder nicht. Als Hilfe sind auch die LEDs, die wenn alles in Ordnung läuft, soll ein grüne LED leuchten, sonst leuchtet die rote LED als Fehlermeldung.

Bemerkungen

Damit das System ohne Probleme läuft, muss man auf ein paar Situation beachten. Es gibt viele Kleidungen bzw. Zubehörteile, die das Registrieren bzw. Einloggen schwierig oder fast unmöglich machen. Deshalb soll ein Benutzer vor der Kamera:

- Keine Brille tragen.

- Stehend bleiben und nicht bewegen.
- Stehend bleiben und nicht bewegen.
- Keine Maske tragen.

Abbildungsverzeichnis

2.1	Big picture	5
3.1	Structured Software Design bzw. erste Ebene	9
3.2	Risikoanalyse in Excel	16
4.1	MySQL Logo [2]	19
4.2	MariaDB Logo [3]	19
4.3	Python Logo [4]	20
4.4	OpenCV Logo [5]	20
4.5	Datenbank - RDM	22
4.6	Reale Person	25
4.7	Ausgabe: Person Detected	25
4.8	Bild eines Kindes	26
4.9	Ausgabe: Nicht Detected	26
4.10	Zeitdauer 1	26
4.11	Zeitdauer 2	26
4.12	Zeitdauer 3	27
4.13	Zeitdauerl 4	27
4.14	Mysql-Server is missing	28
4.15	Linke und Rechte Bild [11]	28
4.16	Beispiel Disparity Map [11]	28
4.17	3D Gesicht Matlab [12]	29
4.18	Jordi Linke Bild 1	30
4.19	Jordi Linke Bild 1	30
4.20	Gute Disparity Map	30
4.21	Jordi Linke Bild 2	30
4.22	Jordi Linke Bild 2	30
4.23	Schlechte Disparity Map	31
4.24	Plan-Do-Check-Act-Zyklus [1]	33
5.1	dlib logo [35]	36

5.2	Bildverarbeitung Structed Design	37
5.3	Erste Ebene Erklärung	38
5.4	Haar Features[43]	39
5.5	Haar	40
5.6	Output: Box auf Gesicht	41
5.7	Unterschied: normalisiert, nicht normalisiert	44
5.8	Unterschied: normalisiert, nicht normalisiert	45
5.9	Output: Zugeschnittenes Bild	46
5.10	Gesichtsschlüsselpunkte [34]	47
5.11	Gesichtsdaten	47
6.1	Grobe Skizze des Systems	53
6.2	Schaltplan des Systems	55
6.3	Erste Ebene	57
6.4	Erste Ebene Erklärung	58
6.5	Vergleichsmethode zweier Punkte	62
6.6	geometrisches Mittel	63
6.7	1_Fall	65
6.8	2_Fall	66
6.9	3_Fall	67
6.10	Fehlerbaum	74

Tabellenverzeichnis

4.1 Technologien	18
----------------------------	----

Datenblätter und Applikationsschriften

[1] MA BSc Bekim Alibali. *ITP QM ALB Skript SchülerTeil1.*

Aus dem Netz

- [2] URL: <https://www.mysql.com/>.
- [3] URL: <https://mariadb.org/>.
- [4] URL: <https://www.python.org/>.
- [5] URL: <https://opencv.org/>.
- [6] URL: <https://www.idiap.ch/dataset/3dmad>.
- [7] URL: <http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html>.
- [8] URL: http://www.vision.caltech.edu/Image_Datasets/Caltech_10K_WebFaces/.
- [9] URL: <http://www.jdl.ac.cn/peal/>.
- [10] URL: <https://www.nist.gov/itl/products-and-services/color-feret-database>.
- [11] URL: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_calib3d/py_depthmap/py_depthmap.html.
- [12] URL: <https://www.mathworks.com/matlabcentral/fileexchange/69497-matlab-3d-face-recognition-code>.
- [13] URL: <https://www.simplyscience.ch/teens-liesnach-archiv/articles/was-ist-eine-leuchtdiode.html>.
- [14] URL: <https://www.rohm.de/electronics-basics/resistors/resistor-basics>.
- [15] URL: <https://www.elektronik-kompendium.de/sites/com/0808181.htm>.
- [16] URL: <https://www.elektronik-kompendium.de/sites/com/0808142.htm>.

- [17] URL: <https://fritzing.org/learning/>.
- [18] URL: <https://de.serlo.org/mathe/stochastik/daten-datendarstellung/daten-kenngrößen/geometrisches-mittel>.
- [19] *Einführung in Computer Vision mit OpenCV und Python.* URL: <https://blog.codecentric.de/2017/06/einfuehrung-in-computer-vision-mit-opencv-und-python/>.
- [20] *Einstiegerguide: Was ist OpenCV?* URL: <https://viscircle.de/einstiegerguide-was-ist-opencv/>.
- [21] *Git.* URL: <https://git-scm.com/docs>.
- [22] Kitware inc. *CMake.* 2000. URL: <https://cmake.org/overview/>.
- [23] *Unterschied Taster als Schließer, Taster als Wechsler und Schalter.* URL: <https://dein-elektriker-info.de/taster-als-schliesser-und-taster-als-wechsler/>.
- [24] Eric W. Weisstein. *Affine Transformation.* MathWorld. URL: <http://mathworld.wolfram.com/AffineTransformation.html>.

Der ganze Rest

- [25] *5x Warum.* <https://www.quality.de/lexikon/5xwarum/>. [Online; accessed 2019]. 2017.
- [26] Bekim Alibali. “Scrum”.
- [27] Stephan Augsten. *Was ist Python?* <https://www.dev-insider.de/was-ist-python-a-843060/>. [Online; accessed 2019]. 12/7/2019.
- [28] Martin H. Badicke. “MariaDB-Monitor?” In: (7. März 2017).
- [29] MA BSc Bekim Alibali. “Projektmanagement Teil5”.
- [30] G. Bradski. “The OpenCV Library”. In: *Dr. Dobb's Journal of Software Tools* (2000).
- [31] Jackson Cooper. *Python's time.sleep() – Pause, Stop, Wait or Sleep your Python Code.* <https://www.pythongcentral.io/pythons-time-sleep-pause-wait-sleep-stop-your-code/>. [Online; accessed Tuesday 23rd July 2013]. 2013.
- [32] Ben Croston. *RPi.GPIO Python Module.* <https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/>. [Online; accessed 2014]. 2014.
- [33] EDUCBA. *Differences Between Linux vs Windows.* <https://www.educba.com/linux-vs-windows/>. [Online; accessed 2019]. 2017.
- [34] Vahid Kazemi und Josephine Sullivan. “One millisecond face alignment with an ensemble of regression trees”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition* (2014), S. 1867–1874.
- [35] Davis E. King. “Dlib-ml: A Machine Learning Toolkit”. In: *Journal of Machine Learning Research* 10 (2009), S. 1755–1758.
- [36] Stefan Luber. “Was is Python?” In: *Nico Litzel* (4. Juli 2018).

- [37] CHIP Digital GmbH Niels Held. *Linux-Umstieg: So einfach gelingt der Windows-Wechsel.* [https://www\(chip.de/artikel/Linux-Umstieg-So-einfach-gelingt-der-Windows-Wechsel-2_140047889.html](https://www(chip.de/artikel/Linux-Umstieg-So-einfach-gelingt-der-Windows-Wechsel-2_140047889.html). [Online; accessed 2007]. 2007.
- [38] Mindfire solutions. *Python: 7 Important Reasons Why You Should Use Python.* <https://medium.com/@mindfiresolutions.usa/python-7-important-reasons-why-you-should-use-python-5801a98a0d0b/>. [Online; accessed 2019]. 3/10/2017.
- [39] Guido Stepken. “MySQL Datenbankhandbuch”. In: Abruf: <http://www.littleidiot.de/mysql> (am 23.04. 2008) (1999).
- [40] Dominik Stocklasser. “Architektur”.
- [41] Dominik Stocklasser. “Aufbau von Systemen”.
- [42] Jeff Tranter. *Control Raspberry Pi GPIO Pins from Python.* <https://www.ics.com/blog/control-raspberry-pi-gpio-pins-python>. [Online; accessed Wednesday, July 31, 2019]. 2019.
- [43] Paul Viola und Michael Jones. “Robust Real-time Object Detection”. In: *International Journal of Computer Vision*. 2001.
- [44] Pauli Virtanen u. a. “SciPy 1.0—Fundamental Algorithms for Scientific Computing in Python”. In: *arXiv e-prints*, arXiv:1907.10121 (Juli 2019), arXiv:1907.10121. arXiv: 1907.10121 [cs.MS].