

Höhere technische Schule für Informationstechnologie
Shkolla e mesme profesionale private për teknologji informacioni

Österreichische 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, 5. 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.

Rei Hoxha

Ort, Datum
Egli Hasmegaj

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 wird dadurch erhöht. Alle Gesichter sollen von den aktuellen Schülern und Lehrer erkannt werden. Es soll auch zwischen einer reellen Person und einem Foto den Unterschied berücksichtigt werden.

Wir haben uns für diese Idee entschieden, weil Sicherheit heute hoch interessant und relevant ist. Es geht hier um einen kontrollierten Zugang in Institutionen mittels Gesichtserkennung zu ermöglichen, da Gesichter eindeutig für jede Person sind. Die größten Herausforderungen und Voraussetzungen des Projekts befinden sich in dem Planungsprozess. Eine andere Voraussetzung ist das Gebrauch von zwei Kameras, damit der Unterschied zwischen einer reellen Person und einem Foto berücksichtigt 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, rooms. Further it achieves the goal of differentiating between a real person and a photo being identified.

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 institucioni 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 angenommen, dass es das Sicherheitssystem einer Institution verbessert und gleichzeitig den Zugang zu bestimmten Orten und Räumen kontrolliert. Ferner wird das Ziel erreicht, zwischen einer realen Person und einem identifizierten Foto zu unterscheiden.

Danksagung

Inhaltsverzeichnis

1	Allgemeines	1
2	Planung	2
2.1	Projektziele	2
2.1.1	Ziele	2
2.1.2	Nicht Ziele	3
2.1.3	Optionale Ziele	3
2.2	Projektplanung	3
2.3	Projektmanagmentmethode	4
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	Umsetzung - Jordi	18
4.1	Allgemeine Beschreibung	18
4.2	Technische Lösungen	21
4.2.1	Datenkbank	21
4.3	Herausforderungen	24
4.3.1	Datenkbank - MariaDB	24
4.3.2	Python Scripting	24
4.3.3	OpenCV	25
4.4	Qualitätssicherung	25

5	Ergebnisse - Jordi	27
5.1	Datenbank	27
5.2	Tieferkennung	27
6	Bildverarbeitung	28
6.1	Allgemeines	28
6.1.1	Entwicklungsumgebung und Technologien	28
6.1.2	Frameworks und Bibliotheken	29
6.2	Technische Lösungen	30
6.2.1	Lösungsweg - Structed Software design	30
6.2.2	Gesichtsdetektion	32
6.2.3	Normalisierung	34
6.2.4	Gesichtsdetektion und Zuschneiden	34
6.2.5	Gesichtsschlüsselpunkte Extraktion	35
6.2.6	Zusammenfassung	36
6.3	Herausforderungen, Probleme und deren Lösung	36
6.4	Qualitätssicherung, Controlling	37
6.5	Ergebnisse	38
7	Gesichtserkennung - Rei	39
7.1	Allgemeines	39
7.2	Technische Lösung	40
7.2.1	Hardware und Aufbau	40
7.2.2	Software	43
7.3	Herausforderungen, Probleme, und deren Lösung	58
7.4	Projektmanagement und Controlling	60
7.5	Ergebnisse	62
7.6	Was wurde gelernt und welche Verbesserungsmöglichkeiten gibt es	62
8	Planung vs Realisierung	65
9	Evaluierung und Resümee	68
9.1	Wertschöpfung und Lessons Learned	68

Kapitel 1

Allgemeines

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

Die Idee, ein System zu entwickeln dass Gesichter erkennt und registriert, ist daraus entstanden wegen folgenden Grunds. Es wurde vom Auftraggeber dieses System gefordert, um einen kontrollierten Zugang in der Schule zu ermöglichen. Es ist gedacht, die Sicherheit der Schule dadurch zu erhöhen und die Überwachung effizienter machen. Hauptziel ist es, alle Gesichter von den aktuellen Schülern und Lehrer zu registrieren und zu erkennen. Das System sollte auch den Unterschied zwischen einer reellen Person und einem Foto berücksichtigen. Es ist auch gefordert, dass die betreffende Person keine Maske, Brille oder Hüte bei der Gesichtserkennung trägt. Die Erkennung von Gesicht erfolgt auch nicht beim Bewegen von Person. Das Team besteht aus Aron Terzeta, Egli Hasmegaj, Rei Hoxha und Jordi Zmiani. Aufgaben sind wie folgend geteilt.

- Aron beschäftigt sich hauptsächlich mit der Gesichtsregistrierungsteil und Tiefenschärfe des Bildes herauszuholen.
- 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 herausholen. Aufbereitung der Daten für Abgleich.
- Rei: User-Gesichtsdaten von Bildverarbeitung-Funktion holen, Vergleichen von Gesichtsdaten, System aufbauen.
- Jordi: Datenbankdesign: Eine DB einrichten, Entwurf der Struktur der DB, DB in MySQL implementieren, Zugriffsberechtigungen festlegen, Error-checking.

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

Ziele, nicht Ziele und optionale Ziele

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 wurde 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 anderen Fotos in der Datenbank, um zu sehen, ob die Person schon registriert wurde).
7. Max. 500 Personen in einer Datenbank speichern.
8. 10 Tests, jeder Test in einer anderen Raumkondition, um alle Betriebskonditionen zu testen.

9. Datenbankdesign
10. Error checking
11. Safe Mode (eine Batterie, Back-ups in einem lokalen Server)
12. Min. Arbeitsvorbereitung (Min. Gesichtsdetektionszeit)
13. Admin account (Register-Rechte nur für Schüler und Lehrer eingeben)

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, Hüte 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. Integration in dem Infotainment-System.
4. Licht neben der Kamera (Night Vision implementieren damit die Erkennung/Registrierung auch dann funktioniert, wenn es dunkel ist.)

2.2 Projektplanung

Unsere Big Picture ist unser erstes grobes Design, das die Lösungsskizze des Projekts beschreibt. Es gibt bestimmte Gründe, warum Big Picture und Structed 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 Structed Design folgt das Top-Down Prinzip, das heißt die Funktionen werden hierarchisch zerlegt (Jede Funktion wird in die folgenden Ebenen detaillierter beschreibt).

Structed Design und Big Picture haben keine Begrenzung. Dort können eindeutig alle Funktionen, Schnittstellen, Signalen und Daten beschreibt werden, sodass von allem leicht zu verstehen ist. Sehen Sie auf Abb. 2.1

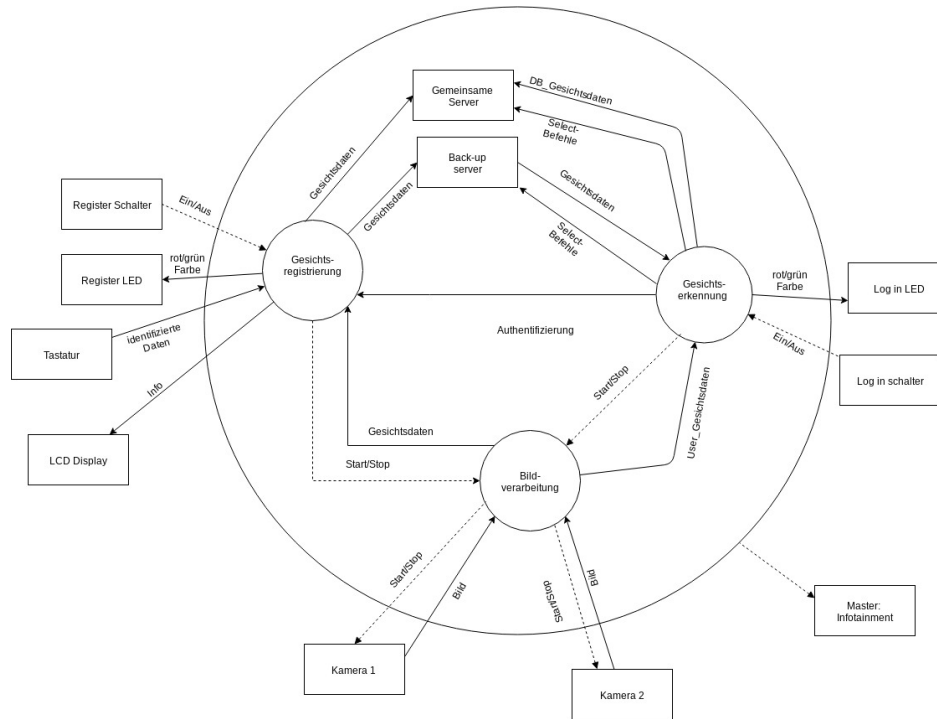


Abbildung 2.1: Big picture

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. Sehen Sie auf Abb. 2.2

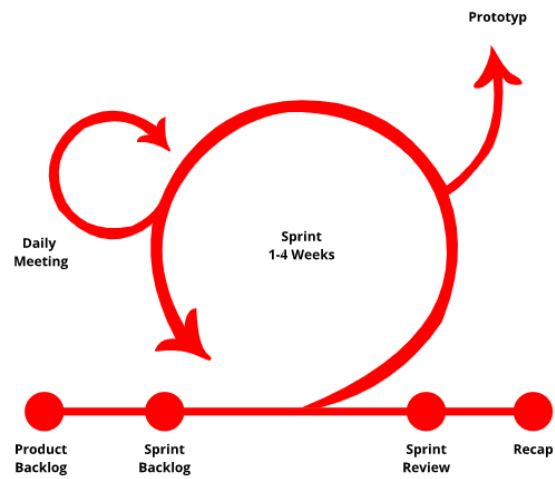


Abbildung 2.2: Scrum

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 Gesichter 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 herausgeholt 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" könnte als eine Backup-Strategie verwendet, wenn das System abstürzt. Git ist ein Versionsverwaltungssystem, das verschiedene Versionen einer Software auf einem Server speichert. Auf dem Server gibt es dann verschiedene Versionen des Systems und die Daten können von einer bestimmten Version dann zurückgeholt werden. Es wird meistens bei der Implementierung-Phase verwendet, um die Veränderungen des Source-Codes zu verwalten.

cmake¹ ist ein Paket, das gebraucht wird, wenn man mit OpenCV² 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.” [8]

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 [21]

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.” [11] 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

¹ein erweiterbares Open-Source-System, das den Erstellungsprozess in einem Betriebssystem und auf compilerunabhängige Weise verwaltet.

²eine Bibliothek von Programmierfunktionen, die hauptsächlich auf Computer Vision in Echtzeit abzielen.

diese.” [24]

Der wichtigste Terminator ist der ”Register-Schalter”. Er initialisiert das ganze System. Schalter in der Technik ist nichts anderes, nur ein Gerät zum Ein- und Ausschalten des Stroms 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-LED³ 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 LED⁴. Eine RGB LED hat drei Grundfarben, rot, grün und blau. Mit diesen drei Farben kann man alle Farben erstellen. Es könnte auch zwei LEDs geben, rot und grün, aber es ist effektiver, ein RGB LED zu verwenden.

Eine spezielle Eigenschaft des Systems ist die Verwendung einer Tastatur. Sie wird verwendet, weil die einzelnen Personen ihren Namen, bzw. Email eingeben müssen. Die andere spezielle Eigenschaft ist die Verwendung eines LCD-Screens⁵. 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-Screens, 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-Screen 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 Infos 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.

³Light-emitting Diode, fließt Strom durch, strahlt sie Licht aus

⁴rot, grün,blau LED

⁵Liquid Crystal Display, präsentiert die elektrischen Signale in Form von visuellen Bildern.

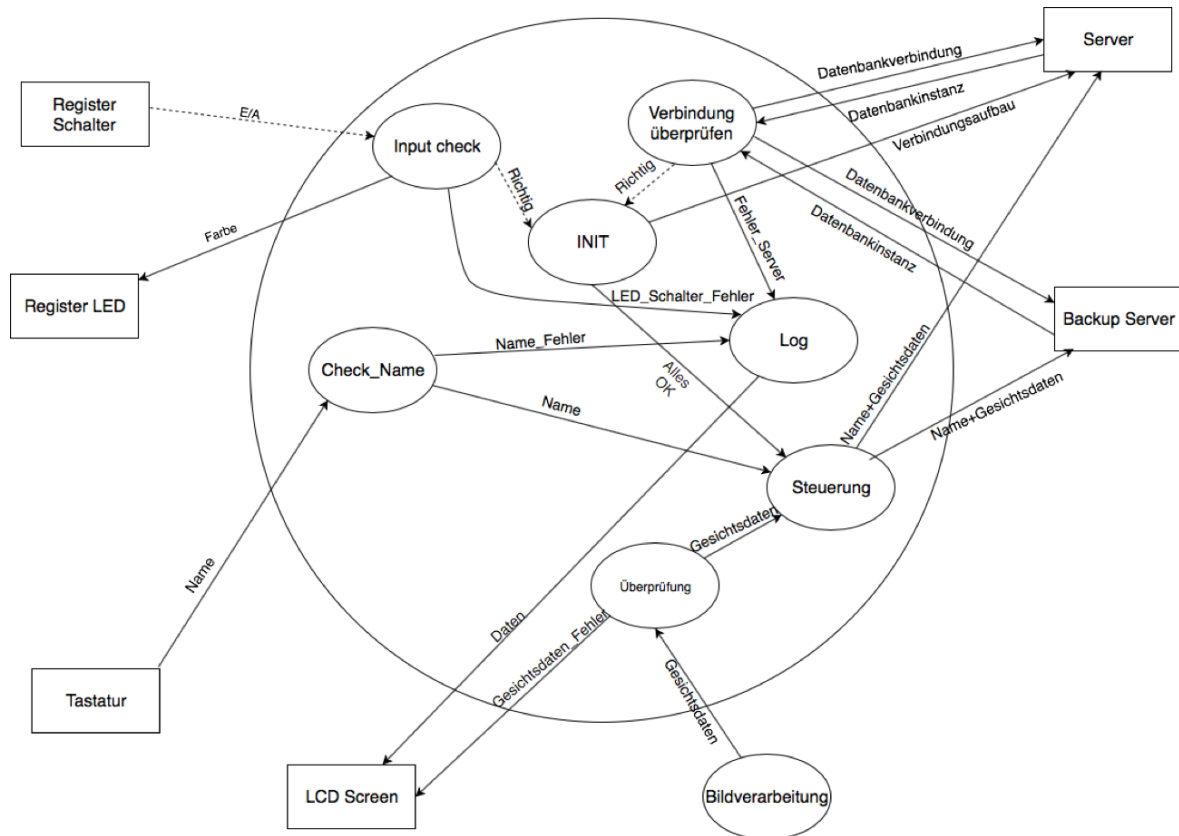


Abbildung 3.1: Structed Software Design bzw. erste Ebene

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 werden 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 Abb. 3.1

Listing 3.1: Packages zu importieren

```
import RPi.GPIO as GPIO
import time
import os
import subprocess
import sys
```

- RPi.GPIO ist ein Paket, das verwendet wird, um Zugriff auf die sogenannten GPIO-Ports⁶ 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. [25]
- time ist ein Paket in Python. Von diesem Paket wird nur die Funktion "sleep" verwendet. Diese Funktion pausiert das python-Programm. [15]

⁶Eingehende und Ausgehende digitale Signale, am Eingangsport können verwendet werden, um digitalen Messwerte mitzuteilen, die von Sensoren empfangen werden

- os ist das wichtigste 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 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 Abb. 3.2 ist der Python-Code.

Listing 3.2: GPIO-Ports Konfiguration

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(18,GPIO.IN)
GPIO.setup(17,GPIO.IN)
GPIO.setup(27,GPIO.OUT)
```

Es gibt verschiedene Betriebsarten für GPIO wie BCM und Board. Ich verwende BCM⁷, weil ich das Paket RPi.GPIO verwende. Mit diesem Paket darf nur die Betriebsart “BCM” verwendet werden. [16] Für die Registrierung der Schüler und Schülerinnen bzw. Lehrer und Lehrerinnen ist es nötig, dass der Admin sich bei dem System einloggt. Die Überprüfung, ob der Admin da ist oder nicht, wird mit einem Vergleich von zwei Bildern gemacht. Ein Bild von Admin 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.

⁷Broadcom Pin Number

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 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 Admin sich bei dem System einloggt. Ist der Admin 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 Admin ist. Die Rolle wird vom Admin geschrieben. 1 steht für Admin, 2 für Schüler und 3 für Lehrer und Lehrerinnen. Sie wird eigentlich nur für die Verwaltung des Systems verwendet. Wenn der Admin 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 davon, 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 Managment 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 Verzeichnisse 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 Missverständnisse 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 Teammitglied 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 bekommt, 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.[13] 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 Einführung	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(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 Administrators 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:

- Admin-Konto. Die Registrierung einer neuen Person geht nicht ohne ein Admin-Konto. Der Admin muss bei dem System eingeloggt sein, damit der Registrierungsteil funktioniert. Der Admin 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

Umsetzung - Jordi

4.1 Allgemeine Beschreibung

Ein wichtiger Teil jedes Projekt ist die Datenbank, und dass ist der Teil wo ich am meisten konzentriert bin. Andere Teilebereiche wo ich teilnehmen habe, sind bei der Extraktion der Gesichtspunkte, wo mein Job, den Tiefe von Bildern zu finden, damit ein 2D vs. 3D unterschied geben sollte, ist.

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 echte Multi-User, Multi-Treaded SQL Datenbank und wird von allen großen Providern oder auch Suchmaschinenbetreibern eingesetzt. MySQL ist eine CLi-



Abbildung 4.1: MySQL Logo
[2]

ent/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 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.[23]

MySQL hat ein breites Anwendungsspektrum und wird meistens in Verbindung mit PHP, Linux, Python usw. verwendet. Der Grund zu der Auswahl von MySQL steht bei der einfachen Bedienung und Verwaltung von Datenbank. Da es sich bei 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.[12]



Abbildung 4.2: MariaDB Logo
[3]

Es ist hauptsächlich in Linux implementiert, da es das Standard-DBS ist. Es bietet eine einfache Verbindung mit Linux-Dateien wie Python, C ++ und vielen anderen Programmiersprachen. Da dies der Standard-Linux-DBS ist und unsere Software unter Linux mit Python-Skripten geschrieben wurde, ist MariaDB unsere Wahl für DBS.

Python

Python ist eine Programmiersprache, die dank ihrer klaren Syntax und einfachen Lesbarkeit leicht zu erlernen ist und sich sehr vielseitig einsetzen lässt. Für die gängigen

Betriebssysteme ist Python frei verfügbar. Die üblichen Programmierparadigmen wie die objektorientierte oder funktionale Programmierung werden unterstützt.

Bei Python handelt es sich um eine Programmiersprache mit einer klaren Syntax und guten Lesbarkeit. Sie gilt als leicht zu erlernen und ist in den gängigen Betriebssystemen interpretierbar. Python unterstützt mehrere Paradigmen der Programmierung wie die funktionale, objektorientierte oder aspektorientierte Programmierung und ist auch als Skriptsprache nutzbar.[20] Die Sprache weist ein offenes, gemeinschaftsbasier-



Abbildung 4.3: Python Logo
[4]

tes Entwicklungsmodell auf, das durch die gemeinnützige Python Software Foundation gestützt wird, die de facto die Definition der Sprache in der Referenzumsetzung CPython pflegt.

Der Grund, warum die Software in Python geschrieben ist, ist, dass Python eine einfache und schnelle Möglichkeit bietet, Datenbankskripte und Gesichtserkennungsprogramme zu erstellen.

OpenCV

OpenCV (Open Source Computer Library) ist eine Open-Source-Bibliothek für Computer Vision und Machine-Learning-Software. OpenCV wurde entwickelt, um eine gemeinsame Infrastruktur für Computer Vision-Anwendungen bereitzustellen und die Nutzung der maschinellen Wahrnehmung in den kommerziellen Produkten zu beschleunigen. Als BSD-lizenziertes Produkt macht OpenCV es Unternehmen leicht, den Code zu nutzen und zu ändern. Es verfügt über C++, Python, Java und MATLAB-Schnittstellen und unterstützt Windows, Linux, Android und Mac OS.[6]



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 implementiert werden. OpenCV ist eine plattformübergreifende Bibliothek und da es viele Deep-Learning-Algorithmen und Frameworks enthält, musste es unsere Wahl für die Objekterkennung sein.

4.2 Technische Lösungen

4.2.1 Datenbank

Wie bereits erwähnt, haben wir uns für MySQL und MariaDB als DBS entschieden, wobei meistens das zweite mehr zum Einsatz kommt. Es ist wichtig zu beachten, dass die Datenbank nicht über die Befehlszeile oder die MySQL/MariaDB-Konsole implementiert wurde, sondern mithilfe von Python-Skripten. In diesem Fall würde jeder gegebene Befehl in den Python-Dateien gespeichert, und wenn etwas schief ging, könnten die Skripte einfach verbessert und ein 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 könnten wir uns eine Vorstellung davon machen, wie die 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
- The AR Face Database, The Ohio State University
- Caltech Faces
- CAS-PEAL Face Database
- The Color FERET Database, USA

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

Der nächste Schritt bestand darin, in Python nachzuforschen, wie die Schnittstelle mit MariaDB verbunden wird. Da es von vielen Linux-Benutzern empfohlen wurde, Python als beste Wahl für die Verbindung zu MariaDB zu wählen, wurde der Programmierer aufgefordert, den Code mithilfe von Methoden und try and except-Anweisungen zu schreiben, um einige saubere und perfekte Skripte zu erstellen. Auf diese Weise wäre der Code sehr einfach zu analysieren und zu verbessern, aber am wichtigsten wäre es, Fehlermeldungen zu beseitigen.

Beschreibung der Tabellen

Nach der anfangs Planung wurden drei Tabellen definiert, die folgenden Tabellen:

- Person
- Info
- Log

Tabelle Person

Die erste Tabelle, Person, enthält alle User mit ihren grundlegenden Informationen, wie Vorname, Nachname, Rolle, Email. Die Spalte Rolle ist damit geeignet, damit ein Unterschied zwischen Personen geben sollte, weil es zwei Gruppen, die Admins und des normalen Users. Die Admin bekommen die Zahl 1 bei der Rolle, während die anderen Users 0. Bei den Spalten, wo der Datentyp varchar verwendet wurde, gibt es ein Grund wiese varchar und nicht char. Das ist so denn char den ganzen Platz reserviert, wo andererseits der Datentyp varchar nur die Länge des gespeicherten Strings reserviert, obwohl die Spalte ein varchar mit der Länge 50 sein kann.

Die Spalte idP ist eine eindeutige Spalte, die es zu der Identifizierung von der Datenzeile gilt, deshalb wird auch als Primar Schlüssel definiert. Es wird auch auto increment festgelegt, damit die nächste Zeile automatisch die nächste Zahl bekommt.

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

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

Die Spalte Rolle enthält der 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.

Tabelle Info

Die zweite Tabelle enthält die Bilder der Personen, die während der Zeit gemacht wurden, und auch die Grund Information der Bilder, wie Punkten und der Path von dem Bild. Die Spalten der folgenden Tabellen sind:

Die Spalte idF ist eine eindeutige Spalte, die es zu der Identifizierung von der Datenzeile gilt, deshalb wird auch als Primar Schlüssel definiert. Es wird auch auto increment festgelegt, damit die nächste Zeile automatisch die nächste Zahl bekommt.

Die Spalte idP ein fremder Schlüssel, der als Primar Schlüssel für die Person Tabelle geeignet ist. Diese fremden Schlüssel sind dazu, damit eine Verbindung zwischen diese zwei Tabellen geben kann. Diese zwei Tabelle haben eine 1 zu mehreren Beziehungen, wo eine Person mehrere Bilder haben kann. Das wurde so geplant, damit je mehr Bilder eine Person hat, desto sicher, besser und funktionierbar der Vergleich sein kann.

Jeder Punkt besteht aus eine 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 dezimal werden, weil die Koordinaten kommastellen Zahlen sind. Nur für den ersten Punkt werden genau die Koordinaten gespeichert, für die anderen zeigt die X- und Y- Koordinate einen Vektor, das heißt es wird der Distanz von diesem Startpunkt aufgenommen.

Die Spalte imagePath speichert die Path der Bilder und wird mit dem Datentyp varchar mit einer Länge von 50 gespeichert, weil Varchar zum Speichern von Strings geeignet ist. Die Bilder wurden als Path und nicht als ganze Image gespeichert, weil ob diese Bilder in der Datenbank gespeichert werden, wurde die Datenbank überfüllt mit Daten. Alternativ was medium blob aber dieser Datentyp brauch zu viel Speicherplatz von unserer Datenbank. Es konnte eine Overflow von der Datenbank geben.

Log Tabelle

Die dritte Tabelle ist nichts anders als eine Tabelle, wo alle Änderungen in der Datenbank protokolliert werden. In diesen Tabellen sollten die Grund Informationen gespeichert werden müssen, wie z.B wer hat was, wann gemacht. Diese frage muss die Tabelle beantwortet.

Die Spalte idL ist eine eindeutige Spalte, die es zu der Identifizierung von der Datenzeile gilt, deshalb wird auch als Primar Schlüssel definiert. Es wird auch auto increment festgelegt, damit die nächste Zeile automatisch die nächste Zahl bekommt.

Die Spaltet User soll der Person speichert, wer die Änderungen durchgeführt hat. Hier wird der Datenbank User gespeichert, weil um etwas zu verändern soll die Person zuerst in der Datenbank einloggen. Die Spalte hat den Datentyp Varchar mit der Länge 50, weil die Funktion Current User, da eingetragen wird.

Die Spalte Date als der Name zeigt, kümmert um die Speicherung von der 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 Triggers um die eintragen von Daten.

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 Internet Blogs gelöst werden.

4.3.1 Datenbank - MariaDB

Die meisten Problemen sind bei der Datenbank getroffen, das ist so passiert wegen der kleinen Unterschiede zwischen MySQL und MariaDB. Manchmal wurde mit dem Programmierregeln von MySQL, obwohl MariaDB in Verwendung war. Zwischen diese zwei Datenbankverwaltungssystem gibt es kleine Unterschiede, die zu Problemen führen. Den betroffenen Fehlern werden hinunter besprochen:

ERROR 1452: Cannot add or update a child row: a foreign key constraint fails.
Das Problem hier war es, dass es keine Tabelle geändert werden konnte, wegen die Beziehungen von Tabellen. In Prinzip sollte zuerst das Kind Tabelle gelöscht werden, und dann die Eltern Tabelle, weil sonst wird eine Konflikt geben, wozu bei der Kind Tabelle einen Fremd Schlüssel definiert ist, der jetzt nicht mehr existiert.

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 getroffen, wenn einen Trigger Erstellung Script ausgeführt wurde. Der Fehler hat hier beim Delimiter gestanden, weil wenn ein Trigger durch ein Script erstellt wird, braucht man keine Delimiter einsetzen, sondern nur wenn es manuell durch der Command Line sollte der Delimiter festgelegt. Hier kann man sehen, die Änderungen zwischen MariaDB und MySQL, in MySQL sollte dieser Code ohne Probleme ausgeführt werden.

Mysql-Server is missing// Dieser Fehler zeigte, dass keine MySQL Server auf dem Betriebssystem installiert war. Dieses Problem ist getroffen, wegen die Änderungen in Linux System. MySQL Server wurde nicht mehr supportiert, und jetzt war im Betrieb MariaDB, und statt mysql-server zu installieren, sollte mariadb-server durch den Befehl `sudo apt-get install mariadb-server` installiert.

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, weil wenn man in Kali Linux programmiert und gearbeitet hat, musste den MariaDB Server immer noch einmal durch den Befehl `service mariadb restart` gestartet werden. Im Vergleich zu Debian Betriebssystem, die auf RaspberryPi installiert war, musst den Server nicht gestartet werden, weil es automatisch im Betrieb war.

4.3.2 Python Scripting

Python ist beliebige Programmiersprachen von vielen Users, aber Python unterscheidet sich von anderen Programmiersprachen durch den Code Struktur. Die meisten Fehlern

wurde bei der Ubetragung von den Parametern zwischen den verschiedenen Funktionen.

TypeError: not all arguments converted during string formatting python

Der folgende Code nimmst als Parameter die Daten von dem neuen User, damit der User in der Datenbank registriert wird. Aber wenn man nur die Methode zu probieren mochte, und gibt als Parameter nur der Name, als eine einfache Registrierung, dann wurde dieser Fehler getroffen.

```
insertvalues1(curs, 'jordi', 'zmiani', 'jorزمi14@htl-shkoder.com', 1)
```

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

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 sollte keine Variable bzw. Funktion mit diesem Namen erstellt werden. Aber trotzdem wurde dieser Fehler generiert, weil wenn ein Parameter auf den falschen Datentyp geparkt wird, dann wird dieses Problem dargestellt.

4.3.3 OpenCV

OpenCV Bibliotheken können manchmal schwierig zu installiert werden, weil es so viele Möglichkeit zum Einrichten von OpenCV gibt, die aber von dem Betriebssystem abhängen. Am meisten Fällen wenn die Installation Fehler produziert, musste OpenCV von Beginn noch einmal eingerichtet werden.

Unable to locate package libjasper-dev

Diese Bibliotheken ermöglichen ein paar Funktion von den OpenCV, das ist eine von vielen Bibliotheken, die eingerichtet werden sollten. In diesem Fall konnte diese Bibliothek nicht installiert werden, weil es die Repository geändert hat, und deshalb musste die Bibliotheken von einer älteren Version heruntergeladen werden. Mit den folgenden Befehlen wurden diese Bibliotheken installiert.

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

4.4 Qualitätssicherung

Einer der wohl bekanntesten Berater/-innen, Lehrer/-innen und Autoren/Autorinnen (über 200 Veröffentlichungen) 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

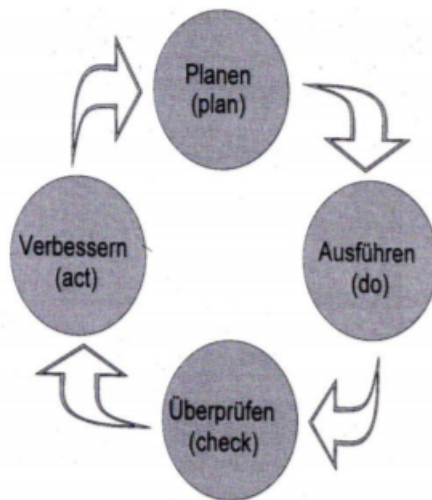


Abbildung 4.5: Plan-Do-Check-Act-Zyklus
[1]

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 4.5 wird die PDCA Zyklus dargestellt, und es ist ganz sichtbar dass die Qualitätsmethode sehr gut für die Verbesserung von Software und Programmen ist. Deshalb wurde auch diese Methode gewählt, weil durch Error und Checking konnten die Fehler gefunden, analysiert und verbessert. Das passt wirklich gut beim Team, wenn sie keine Erfahrungen mit bestimmenden Programmiersprachen bzw. Softwareprogrammen haben.

Oben bei den Herausforderungen war klar, dass wenn Probleme getroffen wurden, wurden sie durch Hilfe von Tutorials bzw. Net Blogs gelöst. Durch verschiedene Zyklen beachtet man auch bei den Risiken, weil wenn etwas nicht funktionier, wird schnell das Problem gefunden und später wird eine andere Lösungswegen bestimmt.

Kapitel 5

Ergebnisse - Jordi

5.1 Datenbank

Für den Bereich Datenbank wurde ordentlich jede Woche programmiert, wo seit dem Anfang wurde es Recherche über verschiedenen Arten von Gesichtserkennung Datenbank, und später hat die Planung und Design von Datenbankentwurf begonnen. Es gab auch viele Änderungen abhängig von anderen Teilbereichen des Projektes. Die Datenbank musst immer geändert werden, und nach den Gesichtsdaten anpassen, damit alles in Ordnung ist. Die folgenden Punkten der Datenbank wurden realisiert:

- Datenbankdesign
- Erstellen von Tabellen
- Log Table
- Triggers
- Zugriffsrechte

Die Datenbank sollte fast fertig sein, nur kleine Änderungen konnten durchgeführt werden, wie für die Backup von Daten.

5.2 Tieferkennung

Für den anderen Bereich Tieferkennung wurde gearbeitet aber nicht mit einem hohen Niveau wegen der Probleme beim Gesicht Erkennung, weil Tieferkennung ein letztes Ziel von dem Projekt ist. Hier wurde natürlich Recherche gemacht, und ein kleines Stück Programm geschrieben, nur zum Ausprobieren von Bibliothek und OpenCV.

Kapitel 6

Bildverarbeitung

6.1 Allgemeines

Diese Diplomarbeit besteht aus vielen unterschiedlichen Modulen, die um bestimmte Ziele bzw. Aufgaben zu lösen gut aufgeteilt sind.

Sehr wichtiger Teil dieses Projektes ist der Bildverarbeitungsteil.

In dem folgenden Abschnitt der Ausarbeitung wird es genau erklärt und beschrieben wie diese Aufgabe gelöst wurde und was dafür verwendet wurde.

Für die Umsetzung wurden folgende Technologien gebraucht.

6.1.1 Entwicklungsumgebung und Technologien

Die gewählte Entwicklungsumgebung war grundsätzlich eine virtuelle Maschine die Linux auf einem Windows System geboten hat.

Linux im Gegensatz von Windows ermöglicht volle Kontrolle über Updates und Upgrades und dadurch könnten komplexe Aufgaben einfacher erledigt werden.

Alles wird leicht und bequem durch Konsole eingegeben.

So wurden die Entwicklung, das Testen von den genutzten Algorithmen und anderen Technologien vielmals erleichtert. [17]

Zusätzlich ist Raspberry Pi als Backup System verwendet worden das auch mit einem lauffähigen Linux Betriebssystem (Debian) funktioniert.

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 allgemeine Programmiersprache auf hohem Niveau. 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 Modulen durch seine große und robuste Standardbibliothek an, von denen man ruhig, abhängig vom Bedarf, auswählen kann. Sehr leicht kann es in der Dokumentation der Python-Standardbibliothek nachgesehen werden um sich besser mit den gezielten Funktionalitäten auszukennen.

[22]

”Git ist ein freies und Open Source verteiltes Versionskontrollsystem, das entwickelt wurde, um alles von kleinen bis zu sehr großen Projekten mit Geschwindigkeit und Effizienz abzuwickeln.” [7]

Die Versionierung der ganzen Software Änderungen erfolgte durch Git.

Es hat sich nützlich erweist indem die Arbeit dadurch zwischen die Projektmitgliedern sehr gut koordiniert und verwaltet wurde.

6.1.2 Frameworks und Bibliotheken

Schlüsselwort von unserem Projekt war das Framework bzw. die Bibliothek „OpenCV“.

OpenCV ist eine open-source Bibliothek für Computer Vision. Also, ganz allgemein kann sie als eine Bibliothek für Bildverarbeitung betrachtet werden.

Sie kümmert sich unter anderem um die Manipulation von Bildern, die Analyse von denen und um die daraus bestimmte Muster bzw. Objekte, für verschiedenen Zwecken einzusetzen. Ganz berühmte Anwendungsgebieten sind Gesichtserkennung und Stereo Vision.

Stereo Vision bedeutet die Extrahierung von Informationen aus einem Bild in eine 3-dimensionalen 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. Module die als feststehende Bibliotheken implementiert sind.

Es ist Cross-Platform, in C/C++ geschrieben und unterstützt auch Python.

[14]

SciPy hat sich nützlich, beim Lösen von gewisse mathematische Angaben und bei zusätzliche Installationen von Bibliotheken zu helfen.

SSciPy ist eine kostenlose und Open-Source-Python-Bibliothek für wissenschaftliches und technisches Rechnen.” [27]

Dies sind einige der Kernpakete von SciPy die nutzbar für das Projekt waren:

NumPy legt die Basis für das wissenschaftliche Rechnen mit Python.

Es unterstützt große mehrdimensionale Arrays und Matrizen. Es enthält viele Mathematische Funktionen auf hoher Ebene 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



Abbildung 6.1: dlib logo [19]

zum Erstellen komplexer Software zur Lösung von der realen Welt getauchte Probleme, beinhaltet.

Es wird vielseitig eingesetzt, in Robotik, Mobilgeräte und unter anderem in Computer Vision.[19]

Es wurde prinzipiell bei der Extrahierung der so genannten „Facial Landmarks“ gebraucht.

6.2 Technische Lösungen

Bei der Gesichtsregistrierung und Gesichtserkennung Diplomarbeit war es die schwierigste und herausforderndste 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 keine Lücken bei der Implementierung kommt.

Aus diesem Grund musste die Arbeitsteilung gut geregelt werden, damit jedes Teammitglied sich auf einen bestimmten Modul der Arbeit konzentrieren konnte.

Es wurde auch das 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 hat durch die Methode „Structed Design“ erfolgt.

Structed Design ist eine Erweiterung von der „Big Picture“ Methode, die dazu dient, ein technisches System mit ihren Schnittstellen mit außen grob zu beschreiben.

Es ist in verschiedenen Ebenen unterteilt, von außen beginnend.

Unten wird die erste Ebene der Structed Design von der Bildverarbeitungsteil dargestellt.

6.2.1 Lösungsweg - Structed Software design

Wie es in der Abb.6.2 sichtbar ist, ist die Bildverarbeitungsteil in diese Einheiten unterteilt:

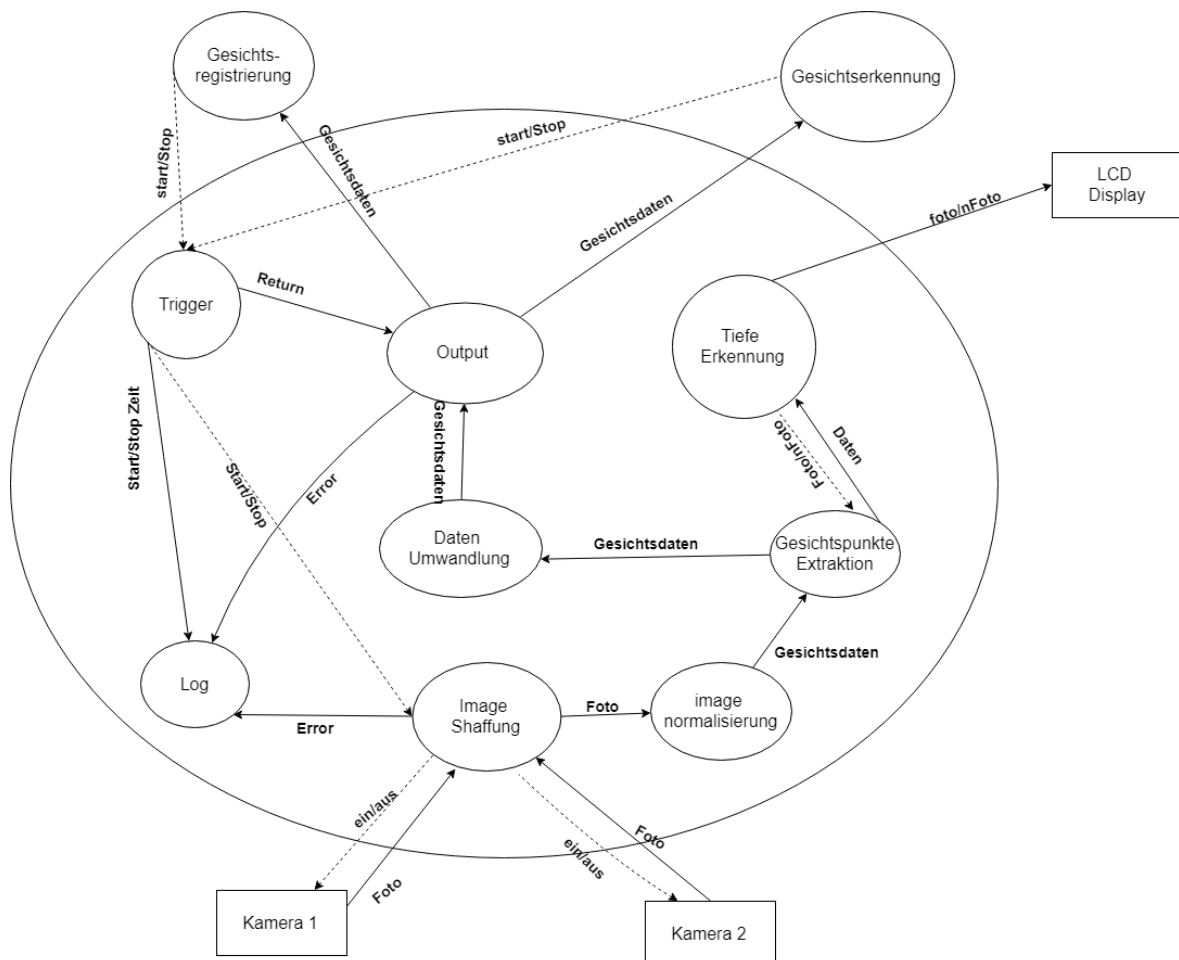


Abbildung 6.2: Bildverarbeitung Structed Design

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

Die Schnittstellen von außen sind das Gesichtserkennungsmodul und die Gesichtsregistrierungsmodul.

Diese schicken eine Anforderung an den Trigger der dann die Bildaufnahme Modul initialisiert. Unabhängig von welchen von diesen beiden Schnittstellen die Anfrage kommt, macht das Bildaufnahme Modul aus der zwei Kameras ein Foto.

Implementierungsnah wurde bis jetzt nur eine Kamera verwendet, da die Stereo Vision noch nicht funktional ist.

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 in dem Normalisierung Abschnitt erklärt.

Danach folgt die Erkennung von Gesichtern, das Ausschneiden von denen und die Extrahierung der Gesichtsschlüsselpunkte.

Sie werden dann zur Abgleich oder Registrierung je nach Bedarf, bereitet und geschickt.

Hinweis: Bei der Umsetzung wurden bestimmte Bereiche mit einander verknüpft, was zu Folgendes führte: was bei der Planung steht, stimmt nicht völlig mit der Methoden zur Umsetzung überein. Weitere Unterschiede werden im Punkt 13b. genauer beschrieben.

6.2.2 Gesichtsdetektion

Das erste Ereignis, dass erfüllt werden muss bei der Gesichtserkennung ist das Finden von Gesichtern, also die Feststellung, wo sich die Gesichter im Bild befinden.

Dafür ist die „Haar Cascade Classifiers“ pre-trained Classifier gebraucht worden.

Gesichtsdetektion 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 gemacht haben.

Die Arbeit heißt „Rapid Object Detection using a Boosted Cascade of Simple Features“. [26]

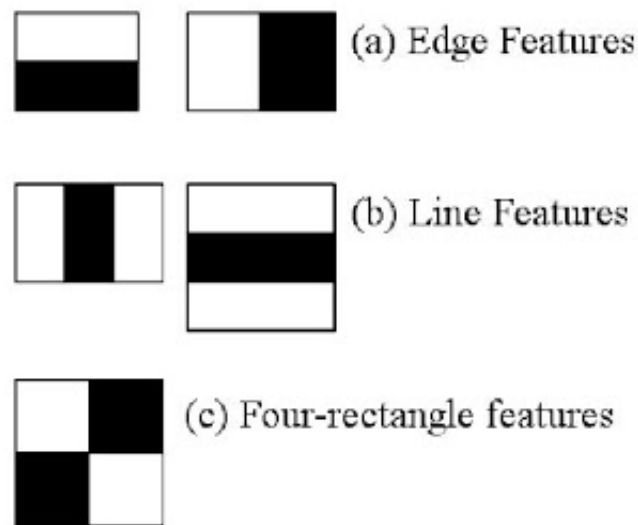


Abbildung 6.3: Haar Features[26]

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 die Haar Merkmale benutzt. Auf Abbildung 6.3 sind sie dargestellt. Jedes Haar Merkmal ist nichts anders als ein Wert, durch das Subtrahieren der Summe der Pixel unter dem weißen Rechteck von der Summe der Pixel unter dem schwarzen Rechteck, erhält. Diese Summen berechnet man durch integrale Bilder, die zur Vereinfachung zu Summenberechnungen dient.

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

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

Umsetzung in Code

Durch die Verwendung von classifiers die OpenCV bietet, setzen wir das um.

```
//haar cascade classifier laden
1 face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_def
```

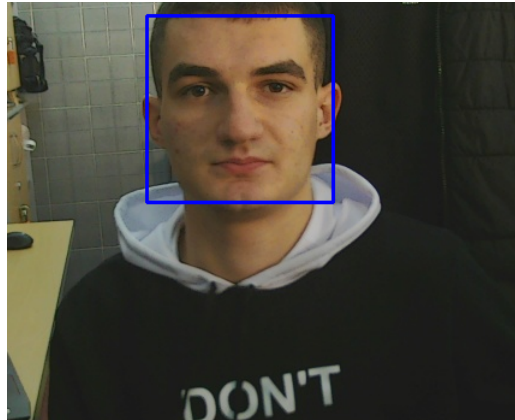



Abbildung 6.4: Box auf Gesicht

```

2         gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
3         faces = face_cascade.detectMultiScale(gray, 1.1, 4)
4         for(x, y, w, h) in faces:
//parameter: image, startpunkt, endpoint(w..width, h..height), farbe, D
5             cv2.rectangle(image, (x, y), (x+w, y+h), (255,0,0), 2)

```

opencv und numpy importieren

Bild als input einfügen

Bild einlesen

Bild in „gray scale“ umwandeln wodurch die Rechenleistung reduziert wird. (Da es keine Farben zuhanden sind).

Parametern: src(source image), dst(destination image), code(conversion code)

Hier benutzt man die Cascade Classifier um Gesichter zu finden.

Parameter:

image, objects, scaleFactor = 1.1, minNeighbors = 3, flags = 0, minSize = new cv.Size(0, 0), maxSize = new cv.Size(0, 0))

Zeichenrahmen für das Gesicht zeichnen.

Bild zeigen durch image Fenster

Press any key zum schließen vom Fenster

Die Funktionalität des Code ist genau auf Abb.6.4 ersichtlich.

6.2.3 Normalisierung

6.2.4 Gesichtsdetektion und Zuschneiden

Nachdem Gesichter gefunden werden, muss man daraus eigene Bilder machen, die zur Erleichterung von der Extrahierung der Schlüsselpunkte dienen.

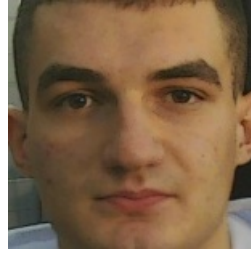


Abbildung 6.5: Abgeschnittenes Gesicht

nrFace bestimmt die Anzahl von Gesichtern, die von den cascade classifier gefunden wurden. Nur wenn dieser Wert größer als Null ist, soll die Logik von der Crop Funktion weiterlaufen.

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

Zeilen 1-6 berechnen das Zentrum von Bild neu und in Zeile 7 wird ein neues Image mit den berechneten Parametern angelegt.

Mit imwrite wird das Image gespeichert. Auf Abb.6.5 sieht man deutlich nur das Gesicht.

6.2.5 Gesichtsschlüsselpunkte Extraktion

Nun kommt es zu dem wichtigsten Punkt meines Teils der Diplomarbeit, die Gesichtsschlüsselpunkte Extraktion.

Dazu wurden die „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 Landmarken?

Das ist eine Teilmenge des Problems der Formvorhersage.

Bei einem vorgegebenen Eingabebild (und normalerweise einer ROI⁸, 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 und Sullivan in ihrem Paper: One Millisecond Face Alignment with an Ensemble of Regression Trees entwickelt.[18]

Dieser Methode werden viele Trainingsdaten hinzugefügt, womit es eine Kombination von Regressionsbäumen trainiert wird, um die Positionen der „Facial Landmarks“ direkt aus den Pixelintensitäten selbst zu beurteilen.[18]

Dadurch werden unsere gewünschten 68 Gesichtsmarkierungen mit hoher Vorhersagbarkeit extrahiert und als x,y Koordinaten weitergegeben. Die Indizes der 68 Koordinaten sind in der Abb. dargestellt:

Nachdem wir der „Predictor“ geladen haben, speichern wir sie in einem Formobjekt mit 68(x,y) Koordinaten.

Es wird hier nicht die resize Funktion verwendet, weil es 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.

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

6.2.6 Zusammenfassung

6.3 Herausforderungen, Probleme und deren Lösung

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

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

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

Die verwendeten 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 neuen Technologien.

Viele Sachen waren am Beginn auch unklar wegen den Kommunikationslücken und auch die sehr grobe Planung war problematisch.

⁸Region of interest

Lösungen:

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

Die Kommunikation hat sich mit der Zeit stark verbessert und dadurch wurden auch die Unklarheiten abgeklärt.

6.4 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 die Ursachen und die. Es geht hier um die Verschachtelung der Ursachen und die Herausfindung von denen durch iterative Fragetechnik, da viele Probleme nicht nur eine einzige Ursache haben. Die Methode ruft jedes Mal eine andere Folge von Fragen auf. [10] 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 FAST⁹.“

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

⁹Features from Accelerated Segment Test

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?

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 haben dabei geholfen dass die eigentliche Ursache des Problems herausgefunden worden ist.

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

6.5 Ergebnisse

Ich habe ungefähr 150 Stunden Zeit bis jetzt für die Diplomarbeit investiert und folgendes erreicht.

Face Detection wurde fertig implementiert.

Bilder sind halbwegs normalisiert worden.

Gesichtsschlüsselpunkte wurden extrahiert und die Merkmalvektoren wurden angegeben.

Kapitel 7

Gesichtserkennung - Rei

Am Beginn wird eine allgemeinere Übersicht dieses Diplomarbeit Kapitels gemacht bzw. dargestellt. Damit wird es verstanden, worum es überhaupt in diesem Teil geht.

7.1 Allgemeines

Die Gesichtserkennung ist der Teil des Systems, mit dem ich mich beschäftige. Es wird durch einen Vergleich überprüft, ob die Person vorher schon registriert worden ist oder nicht, und ob sein Gesichtsdaten schon am Server existieren oder nicht. Nur wenn die Vergleichsergebnisse positiv sind, darf die Person rein kommen. Der Person werden die Ergebnisse durch Anzeigern kommuniziert. Dieses Teil des Projektes erfordert eine Arbeit mit Datenbanken, Gesichtsvergleichsalgorithmen und mit vielen System Tests. 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.7.1 zu sehen.

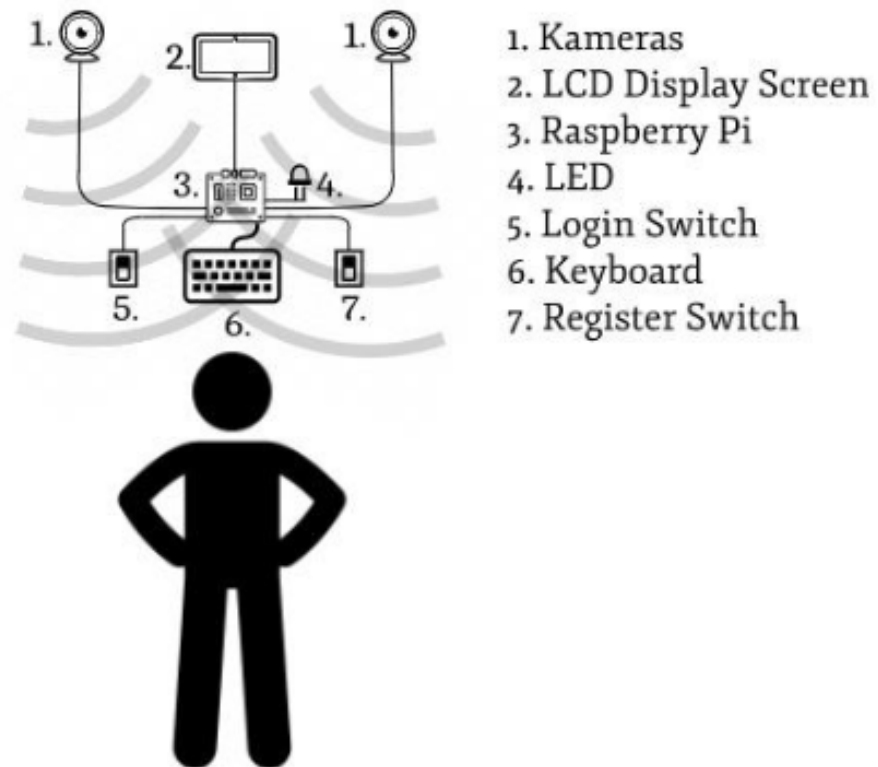


Abbildung 7.1: Grobe Skizze des Systems

7.2 Technische Lösung

In diesem Section wird eine feinere und detaillierte Übersicht im Bezug auf der technischen Lösung gemacht. Alles was mit Technik zu tun hat, wird hier erklärt und umfasst.

7.2.1 Hardware und Aufbau

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

Bauteile und HW-Komponenten

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

1. Steckboard bzw. Steckplatine

Die Steckplatine ist die wichtigste Komponente der Hardware unseres Systems. Sie wird für die Entstehung der elektrischen Verbindung von verschiedenen elektrischen Bauteile benötigt, um elektrische Schaltungen zu bauen oder um ver-

schiedene Tests und Experimenten zu machen. In dieser Steckplatine werden alle anderen Komponenten platziert, damit die Verbindung erstellt werden kann und damit das System laufen 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 die Elemente in der Steckplatine und den Raspberry Pi zu ermöglichen. Auf der anderen Seite werden die Male-Male Kabeln verwendet um die Verbindungen innerhalb der Steckplatine zu ermöglichen.

3. *LEDs*¹⁰

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 erstellt.

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.

5. *Taster*

Ein Taster wird wie ein Button 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.[9]

6. *Raspberry Pi*

Raspberry Pi ist ein Minicomputer, der in diesem Projekt den normalen Computer ersetzt. Der verwendete Raspberry, Version 3, hat 4 USB-Anschlüsse, einem Netzteil, eine SD-Karte, 16 GPIO¹¹ Pins und einem VGA Schnittstelle. Die 3.-Bit SD-Karte ist das wichtigste 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.

¹⁰Light Emitting Diode

¹¹Generated Input Output

8. Tastatur

Eine Tastatur ist ein Input Gerät, dass durch das Eindringen von Tastern den Benutzer die Eingabe von Daten oder Befehle ermöglicht.

9. Kamera

Letztens werden 2 Kameras benötigt, die die Fotos der Gesichter der Personen machen werden. Sie werden auch im Raspberry integriert bzw. mit dem Raspberry verbunden. Die Kameras sind auf 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 erstellt. Die elektrische Schaltung wird durch einen Schaltplan beschrieben. Dieser Schaltplan wurde mit Hilfe eines Programms, das Fritzing heißt, erstellt und spielt eine sehr wichtige Rolle bei der Organisation und Planung des Schaltkreises. Der Schaltplan ist auf Abb.7.2 zu sehen.

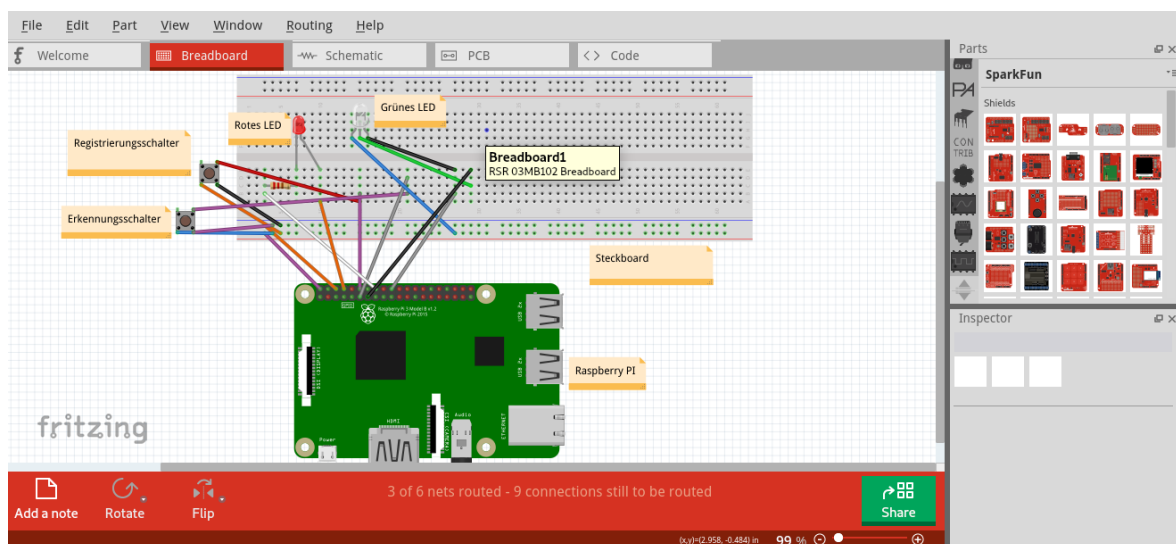


Abbildung 7.2: Schaltplan des Systems

Wie der Schaltplan zeigt, besteht das System aus zwei LEDs, zwei Tastern, einem Widerstand, eine 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 Spannung Pins 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 Erkennungstaster¹² der Minuspol Anschluss mit einem Pin des Minusbereichs der Steckplatine verbunden. Der Pluspol Anschluss gehört zu dem 5-Volt Pin des Raspberry PI, während der

¹²Auch als Login Taster genannt

Datenanschluss mit dem GPIO17 Pin verbunden sind. 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 in dem Minuspol Bereich der Steckplatine eingeschlossen 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.

7.2.2 Software

Nachdem der Aufbau und die verwendete Hardware des Systems beschrieben wurden, ist die Software dran. In diesem Unterkapitel wird alles was mit der logische Teil der Umsetzung zu tun hat: das verwendete Betriebssysteme, Programmiersprachen, Frameworks, Technologien und Planungsmethoden. Es wird jede Aufgabe zusammen mit der zugehörigen Lösung im Detail beschrieben und jedes programmiertes 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 sind.

- 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 einfach 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. [OpenCVR]

- 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.7.2). Fritzing bietet eine sehr große Menge von elektronischen Komponenten und ein PCB, 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 Fritzting kostenlos im Linux angeboten wird, was für Windows nicht der Fall ist.[Fritzing]

Lösungsweg- Beschreibung und Erklärung

Der Lösungsweg für die Umsetzung der Aufgaben der Gesichtserkennung ist streng mit einer guten vorherigen Planung verbunden. Deshalb wurde nicht nur das Big Picture(Großes Sicht des Systems nach Außen), die in die vorherigen Kapiteln beschrieben wurde erklärt, sondern auch die Erste Ebene des Gesichtserkennungsteils. Die Big Picture sieht man unter Abb.7.3

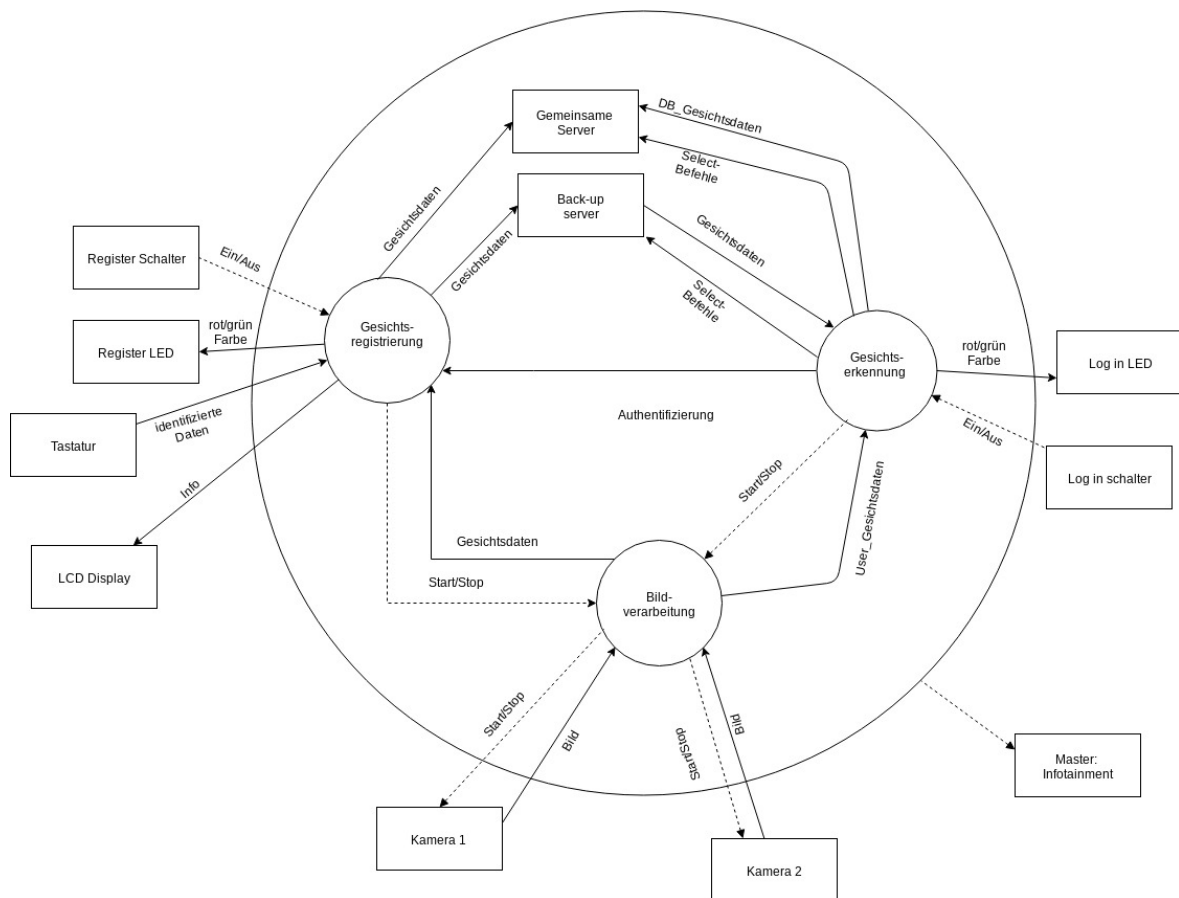


Abbildung 7.3: Big Picture

Die erste Ebene sieht man unter Abb.7.4. Sie ist eine detaillierte Version des Big Picture, das sich nur auf den Erkennungsteil konzentriert. Man spricht von einer Iteration, die hier passiert ist. Folgend wird es die Vorgehensweise und den Lösungsweg dieser Aufgabe mit Hilfe der 1.Ebene erklären und beschreiben.

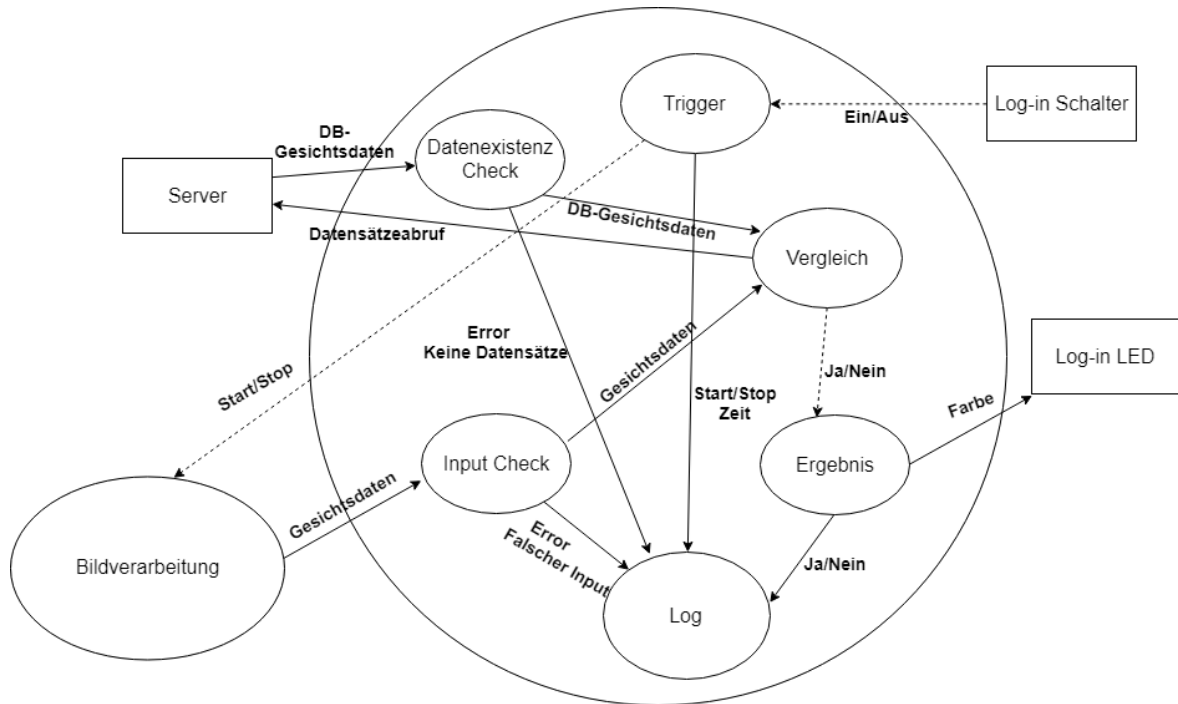


Abbildung 7.4: Erste Ebene

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.

```

if GPIO.input(17):
    exec(open('Existiert_nExistiert.py').read())
  
```

2. Benutzer gibt seine Emailadresse ein

Es ist wichtig zu erwähnen, dass für die Umsetzung dieses Systems es notwendig ist, die verschiedenen Skripten miteinander zu verbinden. Deshalb müssen sie innerhalb anderer Skripten angerufen werden. Das wird durch Variablen gemacht. Deshalb ist das Importieren des Pakets sys 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 UNIQUE 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.

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

```
for x in myresult:
    if x[2]==rei.emailiperkrahasim:
        bool=True
    else:
        bool=False
os.system(' ./ Log-Erkennung-Email-Nicht-Gefunden.py ')
```

Wie es vorher erwähnt wurde, ist dieser Schritt deswegen wichtig, weil falls die eingegebene Email nicht in der Datenbank existiert, muss das System die Bildvergleichen 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 im System nicht mehr versuchen 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 Bilderdaten 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 gekroppt*

Das von der Kamera gemachte Bild ist noch nicht bereit zum Vergleich. Um die Punkte richtig zu extrahieren, muss das Bild zuerst gekroppt bzw. geschnitten sein. Das heißt, die Dimensionen des Bildes werden reduziert, und somit werden die anderen Objekte oder die anderen Objekte nicht berücksichtigt(Sie würden die Performance der Vergleichsprozess sehr viel beeinflussen). Das Cropping des Bildes wurde mit Hilfe eines Programms, das Teil der Arbeit der Bildverarbeitungsgruppe ist. Wird das Bild gekroppt, wird es automatisch auch umbenannt. Das Bild nimmt der Schlüsselwort „New“+die Emailadresse der Person als Name. Der Datentyp ist .jpg .

6. *Die Gesichtspunkte werden extrahiert*

Das gekroppte Bild des Gesichts der Person besteht aus verschiedene Punkte. Insgesamt sind 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. Diese ist auch eine Aufgabe der Bildverarbeitungsbranche, und ist extrem wesentlich für die Kontinuität der Erkennungsprozess.

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 unten zeigt genau wie dies funktioniert.

```
if b=="existiert":
    mycursor.execute(
        """select * from info i \
        join person p \
        on i.idP=p.idP \
        where p.email='%s';""" % var1)
    myresult=mycursor.fetchall()
    for x in myresult:
        print(x)
```

Das heißt, die extrahierte Punkte des von der Kamera gemachten Bildes und die extrahierte Punkte, die in der Datenbank gespeichert waren und die dem Person mit dieselbe Emailadresse wie die vom Benutzer eingegebene Adresse gehören, stehen jetzt endlich zur Verfügung. Von hier wird es angenommen, dass die eingegebene Emailadresse mit mindestens eine in der Datenbank Emailadresse gleich ist, damit der Vergleich Prozess 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, die rechte Auge, der linke Auge und der Mund. Jeder Bereich hat seine eigenen Punkte. Die erste 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 der Vergleichs Qualität sehr stark erhöht, weil die Punkte der Nase werden genau mit der Punkte der anderen Nase verglichen, und nicht mit die von dem Augen beispielsweise.

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

```
for pika in range(0,27):
    dis=math.sqrt(abs(( (vleratx.item(pika)-float(res[vx]))) *
    abs( (vleratx.item(pika)-float(res[vx])) ) + (
    abs((vleraty.item(pika)-float(res[vy]))) *
    abs((vleraty.item(pika)-float(res[vy])) ) ))
```

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 man erfolgen muss, und die nach unten erklärt werden:

- Startpunkt bzw. Origin wird festgelegt

In jedes Gesichtsbild wird ein Startpunkt bzw. Origin für den Vergleich festgelegt. Dieser Punkt ist der Punkt mit der Koordinaten (0/0). Es wurde so gedacht, dass der Abstand von jedem anderem Gesichtspunkt mit diesem Origin kalkuliert wurde. Dieser Abstand wird dann mit dem Abstand der Origin des zweiten Bildes und sein anderer Punkt. Die zwei Punkte, dessen Abstand mit den jeweiligen Startpunkten der Bilder kalkuliert ist, müssen miteinander einpassen(zum Beispiel ein Punkt der rechten Augen im ersten Bild muss mit dem zugehörigen Punkt in dem zweiten Bild verglichen werden, und nicht mit einem Punkt von den Augen).

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

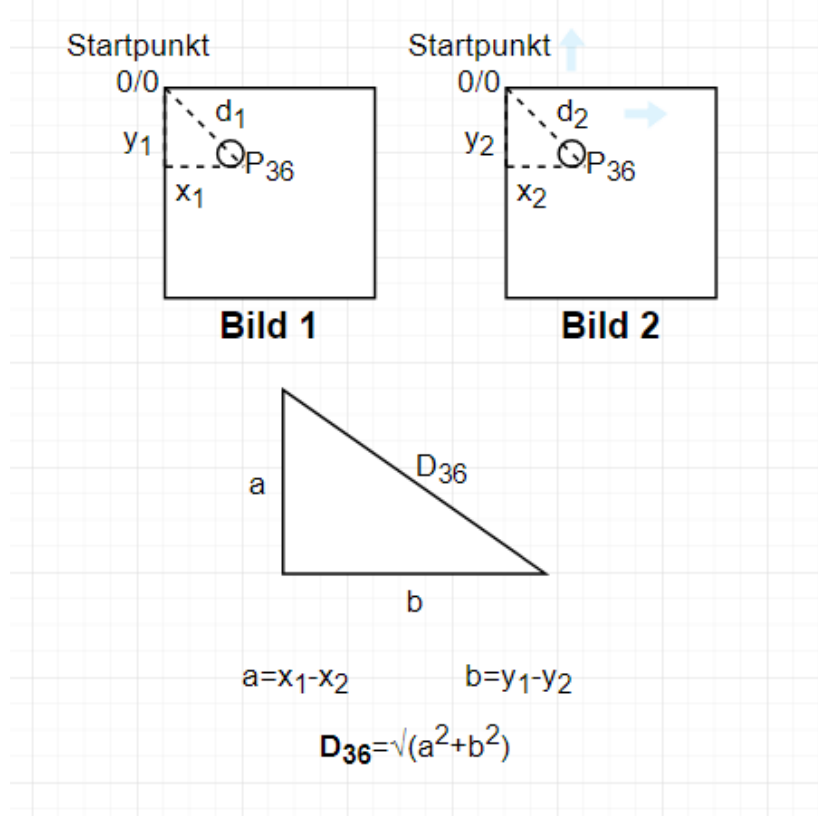


Abbildung 7.5: Vergleichsweg

Auf Abbildung 7.5 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 die 36. Punkt des anderen Bildes hat die Koordinaten x₂ und y₂. Um den Abstand des Punkt 36 zu berechnen, wird die pythagoreische Formel verwendet. Die Differenz der x-Koordinaten und die Differenz der beiden y-Koordinaten sind die beide Katheten(a und b auf die Abbildung). Um die Hypotenuse zu finden wird das Quadrat beider Katheten addiert, und am Ende wird die Wurzel des Ergebnisses gefunden. Die gerechnete Zahl ist der Abstand der Punkt 36 von dem Startpunkt. Dieselbe Methode wird für jeder Punkt verwendet. Das obige Beispiel verglich die Abstände der Punkte von 2 verschiedenen Bilder. Der Grund war folgendes, dass der Weg der Vergleich zwischen der Punkte Abstände des gemachten Bildes und der in der Datenbank gespeicherten Punkte Abstände genau gleich wie im Beispiel erklärt wurde ist. Es ist einfach leichter erklärbar und verstehbar, wenn 2 Bildern verglichen wurden. Die Abstände jeder Punkt jeder Bereich wurde in einer Variable gespeichert.

- Es wird der maximale Abstand für jeder einzelne Bereich gefunden

Nachdem die nach oben erwähnte und erklärte Abstände kalkuliert wurden,

wurden die maximalen Abstände für jeder Bereich(Gesicht, Nase, linke Auge, rechte Auge, Mund) benötigt. Diese Maximumwerte sind wesentlich für den Vergleich. Warum das so ist wird später erklärt.

Die untere Codeabschnitt zeigt wie dieser Schritt für die ersten 27 Punkte programmiert wird

```
MAX = [0,0,0,0,0]
for pika in range(0,27):
    dis=math.sqrt(abs((vleratx.item(pika)-float(res[vx]))) *
    abs((vleratx.item(pika)-float(res[vx])) ) + (
    abs((vleraty.item(pika)-float(res[vy]))) *
    abs((vleraty.item(pika)-float(res[vy])) ) ))
    if(dis < min):
        min = dis
    if(dis > max):
        max = dis
global MAX
MAX[0]=max
print("Max:%s\n"%(MAX[0]))
min=10.0
max=0.0
dis=0
```

Zuerst wurde eine Array Max erstellt, der 5 Elementen enthält. Jedes Element gehört einem von den 5 Bereichen. Beispielsweise der Maximumwert an der Stelle 0(erste Element) ist die Maximumabstand, das bei der Gesichtsbereich gefunden war. Die Funktion Max in Python verwendet. Jeder Abstand wird mit der Maxwert verglichen(Maxwert beginnt am Anfang bei 0). Ist es größer, dann wird dieser Abstand der neue Maximumwert. Am Ende wird einen Maximumwert ermittelt.

- Es wird das geometrische Mittel der Abstände aller Bereiche kalkuliert

"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." [geometrischesM]

Auf Abbildung 7.6 wird das Formel, das sich ergibt gezeigt:

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

Abbildung 7.6: geometrisches Mittel

Die untere Codeabschnitt zeigt wie dieser Schritt für die ersten 27 Punkte

programmiert wird(nur um eine Idee zu haben, wie es funktioniert)

```
for pika in range(0,27):
dis=math.sqrt(abs((vleratx.item(pika)-float(res[vx]))) * abs(
(vleratx.item(pika)-float(res[vx])) ) + (
abs((vleraty.item(pika)-float(res[vy]))) *
abs((vleraty.item(pika)-float(res[vy])) ) ))
gm *= dis
vx+=2
vy+=2
gm=gm**(1/27)
print("GM:%s\n"%gm))
gm=1
dis=0
```

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

(a) Beschreibung

Wenn die angesprochene Maximumwerte und die geometrische Mitteln der Abstände der Punkte des gemachten Bildes und die geometrische Mitteln Abstände des in der Datenbank gespeicherten Punkte gleich oder sehr ähnlich sind, dann wird es von dem selber Person gesprochen und die Erkennung muss erfolgreich sein. Deshalb werden einige Abgrenzungen bestimmt, wann die Erkennung als erfolgreich benannt wird und wann nicht.

(b) Mögliche Fälle

Insgesamt gibt es 5 Bedingungen bzw. 5 Fälle, die eintreten können und die wichtig sind. Ist einer dieser fünf Bedingungen wahr, ist die Erkennung nicht erfolgreich und es wird nicht von derselben Person gesprochen. Diese Fälle werden nach unten mit Hilfe von Bildern erklärt. Es werden die Maximumwerten jeder Bereich(Gesicht, Nase, linke Auge, rechte Auge, Mund, also 5 insgesamt), sowie der geometrische Mittel aller 68 Punkten berücksichtigt.

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

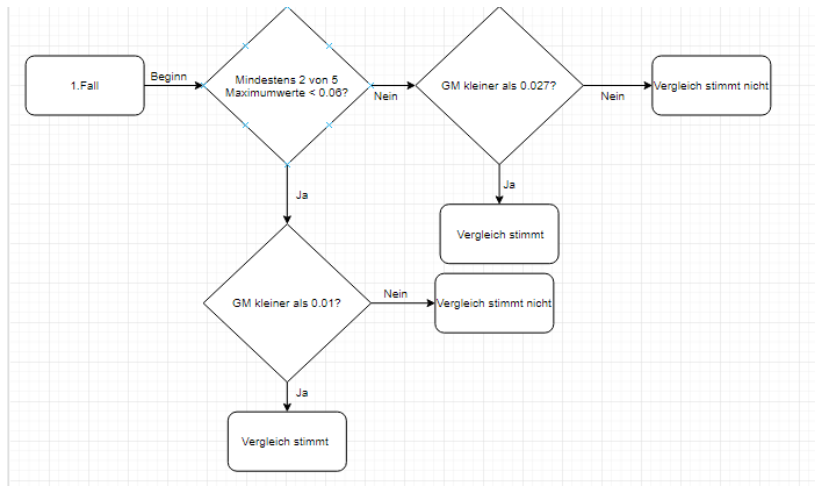


Abbildung 7.7: 1. Fall

Sind 2 oder mehrere Maximumwerte größer als 0.06, wird das geometrische Mittel beobachtet. Ist das größer als 0.01, stimmt die Erkennung nicht. 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 7.7 ist ein Flussdiagramm dargestellt, das die Abfolge dieser Bedingung beschreibt.

Nach unten ist die Codeabschnitt, die wie das funktioniert zeigt, ersichtlich.

```

global count
count = 0
for i in range(0, len(MAX)):
    if MAX[i] > 0.06:
        count+=1
        status1=True
        status11=True
        status2=True
        status22=True
  
```

```

#Wenn JA: Nur wenn gmT kleiner als 0.01 sind sie richtig ,
sonst FALSCH
    if count ==1:
        global status11
        status11=False
    if count >= 2:
        global status22
        status22=False
    if gmT > 0.01:
  
```

```
global status1
status1 = False
```

```
#Wenn NEIN: gmT muss kleiner als 0.027 sein ,
um RICHTIG zu sein
```

```
else:
```

```
    if gmT > 0.027:
```

```
        global status2
```

```
        status2 = False
```

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

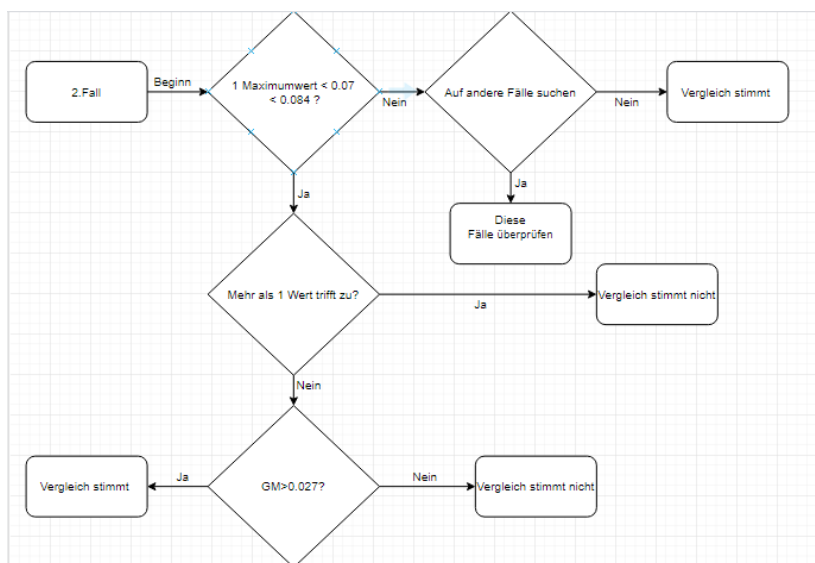


Abbildung 7.8: 2.Fall

Stimmt dieser Bedienung, wird es wieder überprüft, ob das geometrische Mittel kleiner als 0.027 ist oder nicht. Wenn ja, dann ist die Erkennung erfolgreich. Ist aber dieser Bedienung wahr für mehr als einen Wert, ist der Vergleich gleich als nicht erfolgreich genommen.

Auf Abbildung 7.8 ist ein Flussdiagramm dargestellt, das die Abfolge dieser Bedingung beschreibt.

Nach unten ist die Codeabschnitt, die wie das funktioniert zeigt, ersichtlich.

```
count = 0
for i in range(0, len(MAX)):
    if MAX[i] > 0.07 and MAX[i] < 0.084:
        count+=1
        status3=True
```

```

        status3=True
        status4=True
#Wenn JA :Wenn gmT kleiner als 0.027 ist passt ,
        sonst FALSCH
        if count == 1:
            global status3
            status3=False
        if gmT > 0.027:
            global status3
            status3 = False
#Wenn 2 oder mehr: FALSCH
        elif count > 1:
            global status4
            status4 = False

```

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

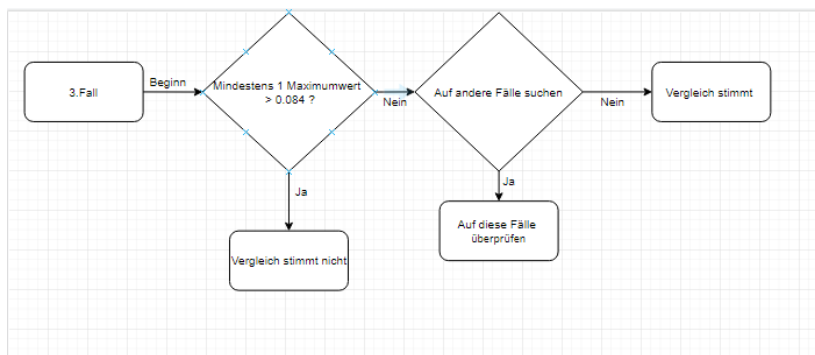


Abbildung 7.9: 3_Fall

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

Auf Abbildung 7.9 ist ein Flussdiagramm dargestellt, das die Abfolge dieser Bedingung beschreibt.

Nach unten ist die Codeabschnitt, die wie das funktioniert zeigt, ersichtlich.

```

for i in range(0, len(MAX)):
    if MAX[i] > 0.084:
        count+=1
        status5=True
#Wenn JA: FALSCH
    if count >=1:
        global status5
        status5=False

```

(c) Andere Abgrenzungen und Entscheidungen

Treffen eine von die 2 untere Kombinationen zu, ist weitere Überprüfung (geometrisches Mittel) nicht mehr nötig und die Erkennung ist nicht erfolgreich.

- i. 1 Maximumwert größer als 0.06 und 1 Maximumwert größer als 0.07
- ii. 2 Maximumwerte größer als 0.06 und 1 Maximumwert größer als 0.07

Nach unten ist die Codeabschnitt, die wie das funktioniert zeigt, ersichtlich.

```
if status1==False or status2==False
or status3==False or status4==False
or status5==False or
  (status11==False and status33==False)
or (status22==False and status33==False):
    print(" Vergleich nicht erfolgreich ,
          sie sind nicht eingeloggt!!!")
    os.system
    ( './Log-Erkennung_Person_Existiert_Nicht.py')
```

- *Ist die Erkennung erfolgreich, kriegt der Benutzer eine Anmeldung, die ihm informiert, dass er im System erfolgreich angemeldet ist.*

Wenn keinen Fall bzw. Bedienung zutrifft, ist die Erkennung erfolgreich. Der Benutzer kriegt danach eine Anmeldung, die ihm Bescheid gibt, dass die Anmeldung erfolgreich war und dass er weitermachen darf. Dazu wurde ein gelbes LED verwendet, das leuchten wird.

Nach unten ist die Codeabschnitt, die wie das funktioniert zeigt, ersichtlich.

```
else :
    b="existiert"
    print(" Vergleich erfolgreich , Sie sind eingeloggt!!!")
    os.system( './Log-Erkennung_Person_Existiert.py')
```

Andere wichtige Punkte und Abgrenzungen

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

1. Distanz der Benutzer von der Kamera

Damit der Vergleich und die Erkennung der Gesichter erfolgreich und je genau wie möglich sind, wurde eine Distanz, dass der Benutzer von der Kamera erhalten muss. Diese Distanz ist 1 Meter, und dem Benutzer wird das durch einer Linie auf dem Boden gezeigt.

2. Nicht gefundene bzw. unbekannte Emailadresse eingeben

Wenn der Benutzer nach dem Drucken des Tasters eine falsche Emailadresse eingibt, wird er eine Anmeldung kriegen, die ihm informiert, dass er eine zweite Chance kriegen wird, die richtige Emailadresse einzugeben (Vielleicht hat er zum Beispiel ein 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 bzw. detektiert

Wenn nach der Bildaufnahme kein eindeutiges Gesicht detektiert wird, wird ein zweites Bild gemacht. Der Benutzer wird durch einer Anmeldung erinnert, dass er gerade und in der Linie stehen muss.

4. Mehrere Gesichter von der Kamera nach der Bildaufnahme gefunden bzw. detektiert

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

5. Die Erkennung stimmt nicht, Benutzer vom System nicht erkannt

Ist die Erkennung nicht erfolgreich, wird ein rotes LED leuchten und der Benutzer versteht, dass er nicht anmelden darf. Er kann aber alles nochmals probieren (Erkennungstaster drücken usw. ...), oder er kann sich zuerst registrieren.

6. Verarbeitungszeit des Systems

Jedes intelligente System braucht einer Verarbeitungszeit. Der Erkennungsprozess braucht ungefähr 7 bis 8 Sekunden Zeit. 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 nach 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 verschiedenen Personen verwendet, um eine Toleranz zu finden, die in der Bedienungen verwendet sein kann. Nachdem die Gesichtspunkte von 2 Bilder getestet und verglichen waren, sind die Ergebnisse auch mit den in der Datenbank gespeicherten Punkten getestet. Am Ende des Testens sind die vorher erklärte Bedienungen definiert und formuliert.

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. Das heißt, diese Logdatei kann vom Benutzer verwendet werden, um Fehleranmeldungen zu kriegen, um den Fehlerart zu verstehen, um den Zeitpunkt verschiedener wichtiger Ereignissen zu wissen und vieles mehr. Alle diese Vorteile und Merkmale der Logging waren mehr als genug zum entschieden, dass die Protokollierung der Ereignisse des Systems eine wichtige Ziel sein wird. Das war der Grund, warum das Gesichtserkennungsteil auch protokolliert wurde.

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 darum verwendet werden, um die totales Dauer der Erkennungsprozess zu bestimmen.

2. Der Zeitpunkt, wann der Benutzer erfolgreich angemeldet war:

Dieser Ereignis ist der wichtigste von allem, 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 nicht erfolgreich angemeldet war

Wie vorher erwähnt war, ist eine Log-Datei eine sehr gute Methode, wichtige Fehlermeldungen sehr schnell zu kriegen. In diesem Fall wird es gespeichert, wann der Benutzer nicht erfolgreich angemeldet war.

4. Der Zeitpunkt, wann die eingegebene Emailadresse in der Datenbank nicht steht, das heißt Email existiert nicht

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

Wie wurde das Logging gemacht?

Das Logging der Gesichtserkennungsprozess 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 Skripten, die die notwendige Informationen erhaltenen mit dieser Log Tabelle, die in der Datenbank liegt, gemacht. Es wurde danach ein Programm für jeder Ereignis, der nach oben erwähnt war, gemacht. Diese Programme ermöglichten die Verbindung mit der Datenbank, und zeigten eine Anmeldung. Beispielweise wurde es in der Datenbank notiert, dass es ein Problem bei Gesichtserkennung gab(Taster nicht gedrückt..). In das Hauptskript wurden dann diese einzelnen Programme gerufen. Für diese Programme wurde die Klasse

Nach unten ist die Codeabschnitt, die wie das funktioniert zeigt, ersichtlich.

Der Taster wird gedrückt:

```
if GPIO.input(17):  
    os.system(' ./ Log-Erkennung-TasterGed.py ')
```

Zeitpunkt wird gespeichert:

```
logging.basicConfig(filename=log_file_path)  
if(log_to_db):  
    logging.getLogger('').addHandler(logdb)  
log=logging.getLogger('Gesichtserkennung')  
log.setLevel(log_error_level)  
test_var='Taster gedrückt'  
log.error('This event occurred: %s' % test_var)
```

7.3 Herausforderungen, Probleme, und deren Lösung

Während dieser Arbeit musste man mit einigen Problemen und Herausforderungen konfrontiert werden.

- *OpenCV Installation*

Die größte 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 war.

- *Fritzing*

Das Problem beim Fritzing war folgendes, dass jedes Mal wenn das Programm beendet wurde, konnte es nicht mehr geöffnet werden. Es fehlten entweder die Pfade oder die Bauteile, die für um die Schaltung zu bauen 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 ein Kurzschluss gab. Der Raspberry PI war kaputt und musste ersetzt werden. Glücklicherweise konnte die SD-Karte erfolgreich in kloniert werden und deshalb sind alle Daten und Informationen gespeichert.

- *Pakete FaceRecognition Installation*

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

- *Probleme bei der Aufruf der Skripten*

Am Beginn war das Umgehen mit der Skripten Aufrufe sehr schwer. Es war schwer zu verstehen wie das genau funktionierte, weil die große Variablenanzahl die Arbeit kompliziert machte. Dieser Anzahl wurde reduziert und es wurden viele Recherchen an der korrekten Verwendung von der sys Pakete 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 Maximumwerte der Abstände bei Bilder, 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 erstellt. Es war sehr kompliziert, in dem Index diesem Array zu zugreifen. Deshalb wurde dieses große Array in 5 kleinere Arrays zerlegt. In dieser Weise war es viel leichter zu verstehen, wie die Indexe funktionierten und wie man dort Zugriff haben 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 $1/68$, das richtige Ergebnis war.

- 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 waren die Punkte genau gespeichert (Auf welchem Index die X-Werte, auf welchem die Y-Werte). Es wurde deshalb Hilfe von dem Mitarbeiter, der sich mit der Datenbank beschäftigt hätte, gebraucht, und danach war alles klarer und verstehbar.

- Probleme mit GIT

Letztens müssen die Probleme mit Git erwähnt werden. Es gab viele Fälle, wo es Konflikten mit der Dateien gibt. Man muss immer zuerst Pull machen, und danach Push. Das hat aber oft nicht passiert. Es wurde in einer Datei gearbeitet, die aber von einer anderen Person schon geändert war. Deshalb gab es Probleme bei der Push. Oft mussten die ganze Dateien gelöscht werden, und danach wieder vom Git abgeholt. Es wurde am Ende zwischen der Gruppe darüber gesprochen, und endlich war das Problem gelöst.

7.4 Projektmanagement und Controlling

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

Die Fehlerbaumanalyse wird verwendet, um die Zuverlässigkeit von technischen Systemen zu testen. Die Fehlerbaumanalyse nimmt als Ausgangspunkt – im Gegensatz zur FMEA – nicht eine einzelne Systemkomponente, sondern das potenziell gestörte Gesamtsystem. Die Fehlerbaumanalyse baut auf der sogenannten negativen Logik auf. Das heißt, der Fehlerbaum beschreibt eine Ausfallfunktion 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, welches an der Spitze des Fehlerbaums steht, das sogenannte Top-Ereignis.

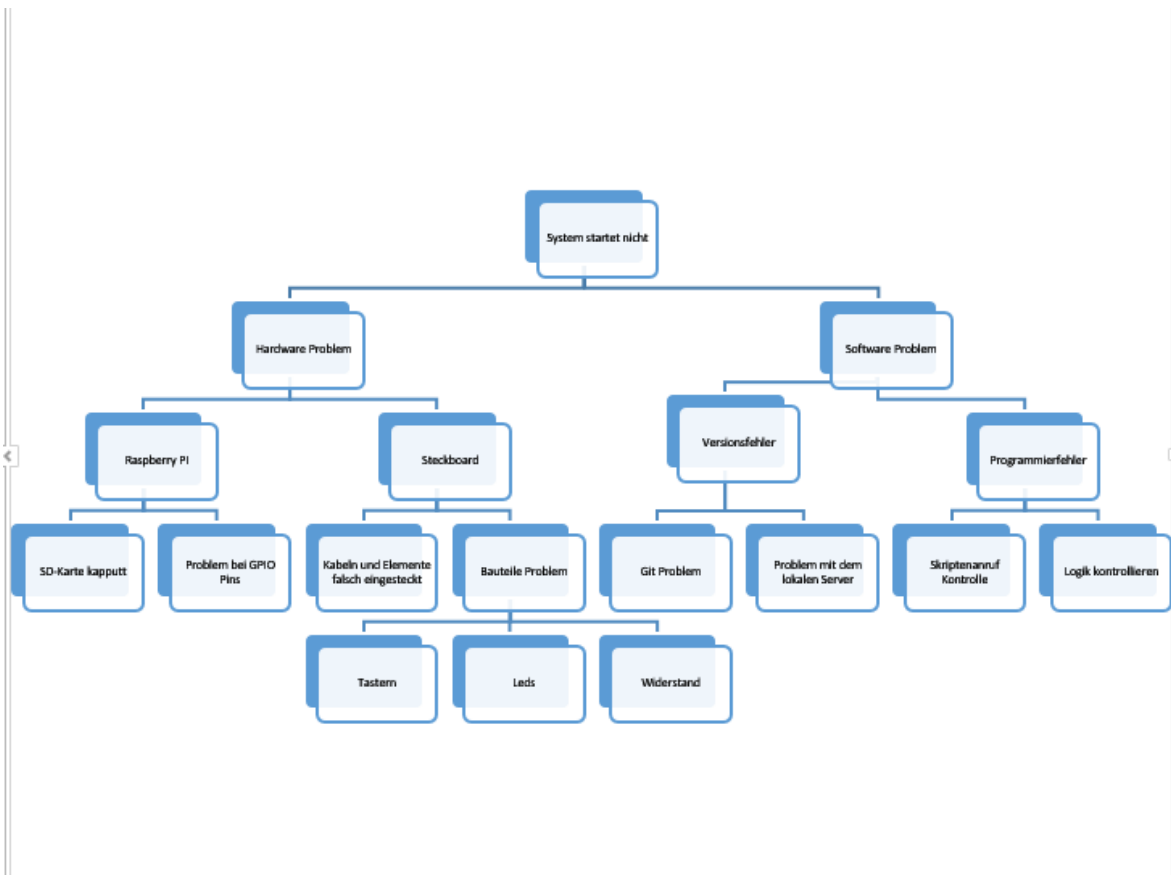


Abbildung 7.10: Fehlerbaum

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 7.10 wird ein Fehler in der Gesichtserkennung Teil des Systems beschrieben. Zuerst muss es 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 der SD-Karte oder mit den GPIO Pins geben. Bei der Steckboard sind entweder die Kabeln und die Verbindungen falsch eingesteckt, oder sind die Bauteile kaputt (Taster, Widerstand, LED, Kamera).

Software Problem?

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

7.5 Ergebnisse

Die finale Ergebnisse sind die folgenden:

1. Systemaufbau
Das ganze System wurde zusammen mit der ganzen 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 80 Prozent der Fälle funktionieren wird.
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.

7.6 Was wurde gelernt und welche Verbesserungsmöglichkeiten gibt es

Während dieser Diplomarbeit war vieles wichtiges gelernt und verstanden. Die wichtigsten Lektionen werden nach unten genannt und erklärt.

- Zeitmanagement

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

- Eingehen mit OpenCV und arbeiten mit Bilderdaten

Es wurde gelernt, wie man überhaupt mit Bildinformationen arbeitet, und wie das wichtige Paket OpenCV funktioniert. In der Zukunft wenn es um arbeiten mit Bilderdaten geht, ist das Basiswissen schon da.

- Wichtigkeit des Testens des Systems

Wie wichtig der Testen Prozess ist wurde auch hier verstanden. Vorher wurde es 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 Fehlern zu finden. Ein System, das vorher nicht getestet ist, kann nicht ein gutes System sein.

- Gruppenarbeit und Verantwortlichkeiten übernehmen

Die Rolle der Gruppe ist auch sehr wichtig. Geht den Mitarbeitern miteinander gut und sind sie alle gut gelaunt, sind sie auch mehr motiviert 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 fertig machen. 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 Picture und die Erste Ebene wurde der Aufwand der Arbeit viel grösser, und die Zeit wurde sicherlich nicht genügend sein. Die Begrenzungen sind auch wichtig, weil ohne die wurde man Zeit an etwas verlieren, das am Ende nicht realisierbar ist oder nicht nötig ist.

- Wie man mit der Fehlern umgeht

Letztens wurde verstanden und gelernt, wie man mit der 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.

Die Verbesserungsmöglichkeiten sind auch bewusst. Diese sind:

- Besseres Zeitmanagement

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

- Backup Verfahren

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

Kapitel 8

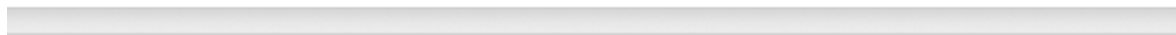
Planung vs Realisierung

Planung vs Realisierung Viele Sachen sind anders umgesetzt worden als sie geplant waren. Die Aufgabenteilung wurde anders geplant aber bei der Implementierung wurde es als vernünftiger angesehen, dass die verändert werden. Es war die Wichtigkeit und die Größe der Arbeitspakete beim Beginn nicht richtig abgeschätzt weil wir uns nicht gut auskannten haben. Konkretes Beispiel ist das Datenbankdesign. Zuerst war diese Aufgabe an Aron zugeteilt, und erst später wurde festgestellt, dass Jordi besser für das geeignet war und so hat er sie übernommen. Gewisse Ziele wurden auch entsprechend angepasst und sogar weggenommen, weil sie als unnötig galten. Zum Beispiel: „Licht neben der Kamera“ wird nicht mehr berücksichtigt. Wir wollten auch eine minimale Verarbeitungszeit erreichen, aber dadurch wir fertig trainierte Modelle und Formvorhersager verwenden, auf denen wir nicht zugreifen können, ist es also schwierig so was zu machen. Es war früher auch geplant nur Gesichtsdaten in der Datenbank zu speichern aber erst wenn es in der Implementierungsphase gekommen ist, haben wir gemerkt, dass wir auch Fotos von den Personen (ins Besondere die von den Administratoren) in der Datenbank speichern sollen. Das wurde aber noch nicht fest entschieden. Auf Abb. 8.2 sind die erreichten und nicht erreichten Ziele.

Ziel	Status
Datenbankdesign wurde erstellt.	✓
Datenbank wurde eingerichtet.	✓
Zugriffsberechtigungen wurden implementiert.	✓
Gesichts-Schlüsselpunkte wurden extrahiert.	✓
Position/Verhältnisse der Hauptmerkmale wurden relativ zueinander herausgeholt.	✓
Daten werden für Gesichtserkennung aufbereitet.	✓
Gesichtserkennung wurde implementiert. Data streams von den extrahierten Bildern werden mit erheblichen Daten die auf der Datenbank gespeichert sind.	✓
Gesichter von zwei Administratoren wurden auf der Datenbank gespeichert für den Zweck der Admin Account.	✓
Admin Account (Register-Rechte nur für Schüler und Lehrer eingeben)	✓
Ein Maximum von 500 Personen ist gedacht, in der Datenbank gespeichert zu werden.	
10 Tests wurden gemacht, jeder Test in einer anderen Raumkondition. Alle Betriebskonditionen wurden getestet.	

Abbildung 8.1: Planung vs Realisierung

"Gefälschte" Gesichter wurden gegenüber "echte / legitime" Gesichtern erkannt. Unterschied zwischen einer reellen Person und einem Bild bzw. einer Visuellen Darstellung von ihnen gefunden.	
Error checking ausgeführt. Es wird signalisiert wenn ein Gesicht nicht 100% identifiziert werden kann.	



Safe Mode erstellt. Backup Server wurde erstellt und eingerichtet.	
Min. Verarbeitungszeit (min. Zeit für Gesichtsdetektion) eingestellt. Es dauert min x Sekunden um ein Gesicht erfolgreich zu erkennen.	✓
Aufbau des Systems wurde erledigt.	✓
Es ist ein Benutzer-freundliches System mittels einer minimalen Anzahl von Schalter und einem Display erstellt.	✓

Abbildung 8.2: Planung vs Realisierung

Kapitel 9

Evaluierung und Resümee

9.1 Wertschöpfung und Lessons Learned

Also, nach einige Wochen sind wir zu Idee gekommen, dass die Planung eine große Rolle in diesem Projekt spielt, weil das Team musst genau spezifizieren was jeder macht, und auch die wichtigsten Punkte, wie z.B der Lösungsweg. Bei den nächsten Treffungen soll mehr über den Problemen diskutiert werden, damit klar für jeden was genau zu tun ist. Eine chaotische Arbeit wird keine guten Ergebnisse bringen, weil auch eine Software bzw. Programm, dass ohne Struktur geschrieben wird, wird schwierig zum Lesen und Verstehen. `~~~~~ ad5d4a4503bd63c93088f9dff686e29b86128b49`

Abbildungsverzeichnis

2.1	Big picture	4
2.2	Scrum	5
3.1	Structed Software Design bzw. erste Ebene	9
3.2	Risikoanalyse in Excel	16
4.1	MySQL Logo	18
4.2	MariaDB Logo	19
4.3	Python Logo	20
4.4	OpenCV Logo	20
4.5	Plan-Do-Check-Act-Zyklus	26
6.1	dlib logo [19]	30
6.2	Bildverarbeitung Structed Design	31
6.3	Haar Features[26]	33
6.4	Box auf Gesicht	34
6.5	Abgeschnittenes Gesicht	35
7.1	Grobe Skizze des Systems	40
7.2	Schaltplan des Systems	42
7.3	Big Picture	44
7.4	Erste Ebene	45
7.5	Vergleichsweg	49
7.6	geometrisches Mittel	50
7.7	1.Fall	52
7.8	2.Fall	53
7.9	3.Fall	54
7.10	Fehlerbaum	61
8.1	Planung vs Realisierung	66
8.2	Planung vs Realisierung	67

Tabellenverzeichnis

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

Literatur

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] *Einsteigerguide: Was ist OpenCV?* URL: <https://viscircle.de/einsteigerguide-was-ist-opencv/>.
- [7] *Git*. URL: <https://git-scm.com/docs>.
- [8] Kitware inc. *CMake*. 2000. URL: [url](#).
- [9] *Unterschied Taster als Schließer, Taster als Wechsler und Schalter*. URL: <https://dein-elektriker-info.de/taster-als-schliesser-und-taster-als-wechsler/>.

Der ganze Rest

- [10] *5x Warum*. <https://www.quality.de/lexikon/5xwarum/>. [Online; accessed 2019]. 2017.
- [11] Stephan Augsten. *Was ist Python?* <https://www.dev-insider.de/was-ist-python-a-843060/>. [Online; accessed 2019]. 12/7/2019.
- [12] Martin H. Badicke. “MariaDB-Monitor?” In: (7. März 2017).
- [13] MA BSc Bekim Alibali. “Projektmanagement Teil5”.
- [14] G. Bradski. “The OpenCV Library”. In: *Dr. Dobb’s Journal of Software Tools* (2000).
- [15] Jackson Cooper. *Python’s time.sleep() – Pause, Stop, Wait or Sleep your Python Code*. <https://www.pythoncentral.io/pythons-time-sleep-pause-wait-sleep-stop-your-code/>. [Online; accessed Tuesday 23rd July 2013]. 2013.

- [16] Ben Croston. *RPi.GPIO Python Module*. <https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/>. [Online; accessed 2014]. 2014.
- [17] EDUCBA. *Differences Between Linux vs Windows*. <https://www.educba.com/linux-vs-windows/>. [Online; accessed 2019]. 2017.
- [18] 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.
- [19] Davis E. King. “Dlib-ml: A Machine Learning Toolkit”. In: *Journal of Machine Learning Research* 10 (2009), S. 1755–1758.
- [20] Stefan Luber. “Was is Python?” In: *Nico Litzel* (4. Juli 2018).
- [21] 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. [Online; accessed 2007]. 2007.
- [22] 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.
- [23] Guido Stepken. “MySQL Datenbankhandbuch”. In: *Abruf: http://www.littleidiot.de/mysql (am 23.04. 2008)* (1999).
- [24] Dominik Stocklasser. “Architektur”.
- [25] 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.
- [26] Paul Viola und Michael Jones. “Robust Real-time Object Detection”. In: *International Journal of Computer Vision*. 2001.
- [27] 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].

```
#!/usr/bin/python3
```

```
import cv2
cam = cv2.VideoCapture(0)
cv2.namedWindow("test")

img_counter = 0
while True:
    ret, frame = cam.read()
    cv2.imshow("test", frame)
    if not ret:
        break
    k = cv2.waitKey(1)
    if k%256 == 27:
        print("Escape hit, closing...")
        break
    elif k%256 == 32:
        #space
        img_name = "frame{}.png".format(img_counter)
        cv2.imwrite(img_name, frame)
        print("{} written!".format(img_name))
        img_counter+=1

cam.release()

cv2.destroyAllWindows()
```

```
#!/usr/bin/python3
import MySQLdb
import time
import logging
db_server='localhost'
db_user='aronterzeta'
db_password='aronterzeta'
db_dbname='test'
db_tbl_log='log'
log_file_path='/home/pi/diplomarbeit/python-scripts/fehlerlog.txt'
log_error_level='DEBUG'
log_to_db=True
```

```
class LogDBHandler(logging.Handler):
    def __init__(self, sql_conn, sql_cursor, db_tbl_log):
        logging.Handler.__init__(self)
        self.sql_cursor=sql_cursor
        self.sql_conn=sql_conn
        self.db_tbl_log=db_tbl_log
    def emit(self, record):
```



```

tm=time.strftime("%Y-%m-%d_%H:%M:%S", time.localtime(record.created_at))
self.log_msg=record.msg
self.log_msg=self.log_msg.strip()
self.log_msg=self.log_msg.replace('\'', '\\\'')
sql='insert into '+self.db_tbl_log + ' (log_level, ' + \
    'log_levelname, log, created_at, created_by)' + \
    ' values (' + \
    ' ' + str(record.levelno) + ', ' + \
    '\\\' + str(record.levelname) + '\\\' , ' + \
    '\\\' + str(self.log_msg) + '\\\' , ' + \
    '\\\' + tm + '\\\' , ' + \
    '\\\' + str(record.name) + '\\\' )'
try:
    self.sql_cursor.execute(sql)
    self.sql_conn.commit()
except MySQLdb.Error as e:
    print (sql)
    print (" Critical DB Error")
if(log_to_db):
    log_conn=MySQLdb.connect(db_server, db_user, db_password, db_dbname, 30)
    log_cursor=log_conn.cursor()
    logdb=LogDBHandler(log_conn, log_cursor, db_tbl_log)

logging.basicConfig(filename=log_file_path)
if(log_to_db):
    logging.getLogger('Aron').addHandler(logdb)
log=logging.getLogger('Gesichtsregistrierung')
log.setLevel(log_error_level)
test_var='Admin ist eingeloggt'
log.error('Die Nachricht ist: %s' % test_var)

#!/usr/bin/python3
import MySQLdb
import time
import logging
db_server='localhost'
db_user='aronterzeta'
db_password='aronterzeta'
db_dbname='test'
db_tbl_log='log'
log_file_path='/home/pi/diplomarbeit/python_scripts/fehlerlog.txt'
log_error_level='DEBUG'
log_to_db=True

class LogDBHandler(logging.Handler):
    def __init__(self, sql_conn, sql_cursor, db_tbl_log):
        logging.Handler.__init__(self)

```

```
        self.sql_cursor=sql_cursor
        self.sql_conn=sql_conn
        self.db_tbl_log=db_tbl_log
    def emit(self,record):
        tm=time.strftime("%Y-%m-%d_%H:%M:%S", time.localtime(record.created_at))
        self.log_msg=record.msg
        self.log_msg=self.log_msg.strip()
        self.log_msg=self.log_msg.replace('\\', '\\\\')
        sql='insert into '+self.db_tbl_log + '_(log_level,_' + \
            'log_levelname,log,created_at,created_by)_' + \
            'values_(' + \
            ' + str(record.levelno) + ',_' + \
            '\\'+ str(record.levelname) + '\\',_' + \
            '\\'+ str(self.log_msg) + '\\',_' + \
            '\\'+ tm + '\\',_' + \
            '\\'+ str(record.name) + '\\'),'
        try:
            self.sql_cursor.execute(sql)
            self.sql_conn.commit()
        except MySQLdb.Error as e:
            print (sql)
            print ("Critical DB_Error")
if(log_to_db):
    log_conn=MySQLdb.connect(db_server ,db_user ,db_password ,db_dbname,30)
    log_cursor=log_conn.cursor()
    logdb=LogDBHandler(log_conn ,log_cursor ,db_tbl_log)

logging.basicConfig(filename=log_file_path)
if(log_to_db):
    logging.getLogger('Aron_Terzeta').addHandler(logdb)
log=logging.getLogger('Gesichtsregistrierung')
log.setLevel(log_error_level)
test_var='Eine_Person_list_in_der_Datenbank_registriert'
log.error('Die_Nachricht_list:_%s' % test_var)

#!/usr/bin/python3
import MySQLdb
import time
import logging
db_server='localhost'
db_user='aronterzeta'
db_password='aronterzeta'
db_dbname='test'
db_tbl_log='log'
log_file_path='/home/pi/diplomarbeit/python_scripts/fehlerlog.txt'
log_error_level='DEBUG'
log_to_db=True
```

```
class LogDBHandler(logging.Handler):
    def __init__(self, sql_conn, sql_cursor, db_tbl_log):
        logging.Handler.__init__(self)
        self.sql_cursor=sql_cursor
        self.sql_conn=sql_conn
        self.db_tbl_log=db_tbl_log
    def emit(self, record):
        tm=time.strftime("%Y-%m-%d_%H:%M:%S", time.localtime(record.created))
        self.log_msg=record.msg
        self.log_msg=self.log_msg.strip()
        self.log_msg=self.log_msg.replace('\\', '\\\\')
        sql='insert into '+self.db_tbl_log + '_(log_level,_' + \
            'log_levelname,log,created_at,created_by)_' + \
            'values_(' + \
            '' + str(record.levelno) + ',_' + \
            '\\'+ str(record.levelname) + '\\',_' + \
            '\\'+ str(self.log_msg) + '\\',_' + \
            '_,_' + tm + '\\',_' + \
            '\\'+ str(record.name) + '\\')'
        try:
            self.sql_cursor.execute(sql)
            self.sql_conn.commit()
        except MySQLdb.Error as e:
            print (sql)
            print ("Critical DB Error")
if(log_to_db):
    log_conn=MySQLdb.connect(db_server,db_user,db_password,db_dbname,30)
    log_cursor=log_conn.cursor()
    logdb=LogDBHandler(log_conn,log_cursor,db_tbl_log)

logging.basicConfig(filename=log_file_path)
if(log_to_db):
    logging.getLogger('Aron_Terzeta').addHandler(logdb)
log=logging.getLogger('Gesichtsregistrierung')
log.setLevel(log_error_level)
test_var='Fehler bei der Registrierung einer neuen Person'
log.error('Die Nachricht ist: %s' % test_var)

#!/usr/bin/python3
import MySQLdb
import time
import logging
db_server='localhost'
db_user='aronterzeta'
db_password='aronterzeta'
db_dbname='test'
```

```
db_tbl_log='log'
log_file_path='/home/pi/diplomarbeit/python_scripts/fehlerlog.txt'
log_error_level='DEBUG'
log_to_db=True

class LogDBHandler(logging.Handler):
    def __init__(self, sql_conn, sql_cursor, db_tbl_log):
        logging.Handler.__init__(self)
        self.sql_cursor=sql_cursor
        self.sql_conn=sql_conn
        self.db_tbl_log=db_tbl_log
    def emit(self, record):
        tm=time.strftime("%Y-%m-%d_%H:%M:%S", time.localtime(record.created))
        self.log_msg=record.msg
        self.log_msg=self.log_msg.strip()
        self.log_msg=self.log_msg.replace('\\', '\\\\')
        sql='insert_into_'+self.db_tbl_log+'_'+(log_level, '_' + \
            'log_levelname,log,created_at,created_by)+'_ + \
            'values_(' + \
            '' + str(record.levelno) + ',_' + \
            '\\'+ str(record.levelname) + '\\',_' + \
            '\\'+ str(self.log_msg) + '\\',_' + \
            '_,_' + tm + '\\',_' + \
            '\\'+ str(record.name) + '\\')'
        try:
            self.sql_cursor.execute(sql)
            self.sql_conn.commit()
        except MySQLdb.Error as e:
            print (sql)
            print ("Critical_DB_Error")
if(log_to_db):
    log_conn=MySQLdb.connect(db_server,db_user,db_password,db_dbname,30)
    log_cursor=log_conn.cursor()
    logdb=LogDBHandler(log_conn,log_cursor,db_tbl_log)

logging.basicConfig(filename=log_file_path)
if(log_to_db):
    logging.getLogger('Aron').addHandler(logdb)
log=logging.getLogger('Gesichtsregistrierung')
log.setLevel(log_error_level)
test_var='Der_Schalter_fuer_Gesichtsregistrierung_ist_gedrueckt.'
log.error('Die_Nachricht_ist:_%s' % test_var)

#!/usr/bin/python3.5
import MySQLdb

#def drop_tables(curs):
```



```
variable2=input()
print("email_eingeben_bitte:")
variable3=input()
print("Role_eingeben_bitte:")
variable4=int(input())
insert_values1(curs, variable1, variable2, variable3, variable4)
conn.commit()
os.system(' ./ fehlerlog_Gesichtsregistrierung_Personregistration.py ')
a="inserted_values"
print(a)
#print("inserted values")
except:
    conn.rollback()
    os.system(' ./ fehlerlog_Gesichtsregistrierung_notPersonregistration.py ')
    a="failed_to_insert"
    #print("failed to insert")
#try:
#    for row in read_values(curs):
#        print('Rei')
#    print("Read values")
#except:
#    conn.rollback()
#    print("failed to read values")

#!/usr/bin/python3.5
import RPi.GPIO as GPIO
import time
import os
import subprocess
import sys
from getpass import getpass
#from Person_Registrierung import (variable3)
#os.system("./enter.sh")
GPIO.setmode(GPIO.BCM)

GPIO.setup(23, GPIO.OUT)#AronLed
GPIO.setup(18, GPIO.IN)#AronSchalter
GPIO.setup(17, GPIO.IN)#ReiLED
GPIO.setup(27, GPIO.OUT)#ReiSchalter
a=""
b=""
correctpaswadmin="@r0nt3rz()"
#image2=sys.argv[1]
#foto2=sys.argv[1]
while True:
    if GPIO.input(18):
        os.system(' ./ fehlerlog_Gesichtsregistrierung.py ')
```

```
print("Password_für_admin_please:")
inputpasw=getpass()
if(inputpasw == correctpaswadmin):
    os.system('./fehlerlog_Gesichtsregistrierung_admin.py')
#os.system('./Test_'+image2+".jpg")
#exec(open('./Vergleich_2_Fotos.py').read())
#if(c=="matched"):
#os.system('./connection.py')
exec(open('Person_Registrierung.py').read())
print(variable3)
if (a=="inserted_values"):
    os.system('./Test_'+(variable3)+".jpg")
#os.system('./selectID_UpdateBild.py'+(variable3))
os.system('./68fLandmarks.py_%s'%(variable3))
GPIO.output(23,GPIO.HIGH)
time.sleep(1.5)
GPIO.output(23,GPIO.LOW)
#exit()
else:
    print("Fehler")
    GPIO.output(23,GPIO.LOW)
#exit()
else:
    print("Admin_ist_nicht_da")
else:
    #os.system('./connection.py')
    #GPIO.output(23,GPIO.LOW)
    print("Schalter_für_Gesichtsregistrierung_nicht_gedrueckt")
    time.sleep(0.5)
    #exit()
if GPIO.input(17):
    os.system('./Log_Erkennung_TasterGed.py')
    exec(open('reitest3.py').read())
    if b=="existiert" and skaftyra=="face":
        GPIO.output(27,GPIO.HIGH)

        time.sleep(1.5)
        GPIO.output(27,GPIO.LOW)
    else:
        print("nicht_existiert")
        GPIO.output(27,GPIO.LOW)
else:
    print("Schalter_für_Gesichtserkennung_nicht_gedrueckt")
    time.sleep(0.5)
```

```
#!/usr/bin/python3
```

```
import MySQLdb
import sys
import os
import subprocess
variable3=sys.argv[1]
conn=MySQLdb.connect('localhost','aronterzeta','aronterzeta','test')
mycursor=conn.cursor()
conn.close()
```