SHKOLLA AUSTRIAKE

# Diplomarbeit

# Gesichtserkennung - Rei

Autor: Rei Hoxha

# Inhaltsverzeichnis

# Kapitel 1

# Gesichtserkennung - Rei

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

## 1.1 Allgemeines

Die Gesichtserkennung ist der Teil des Systems, mit dem ich mir 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.1.1 zu sehen.

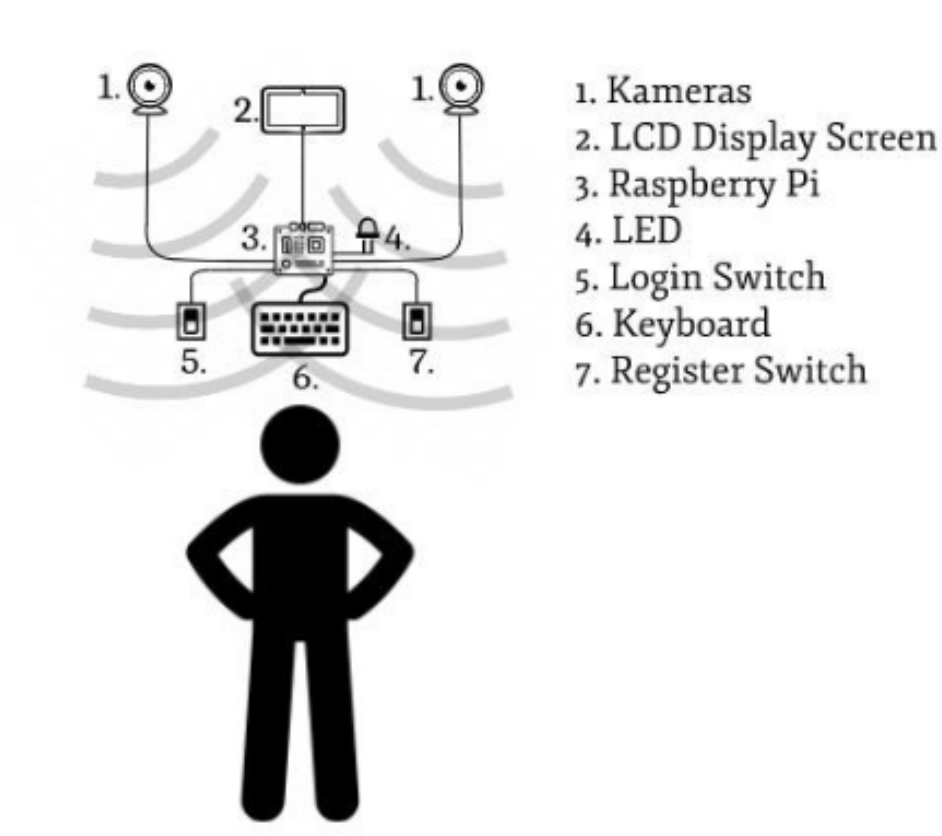


Abbildung 1.1: Grobe Skizze des Systems

## 1.2 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 umgefasst.

### 1.2.1 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*[1]
   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.[4]

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 GPIO[2] 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.

[1] Light Emitting Diode
[2] Generated Input Output

8. *Tastatur*
   Eine Tastatur ist ein Input Gerät, dass durch das Eindrücken 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.1.2 zu sehen.

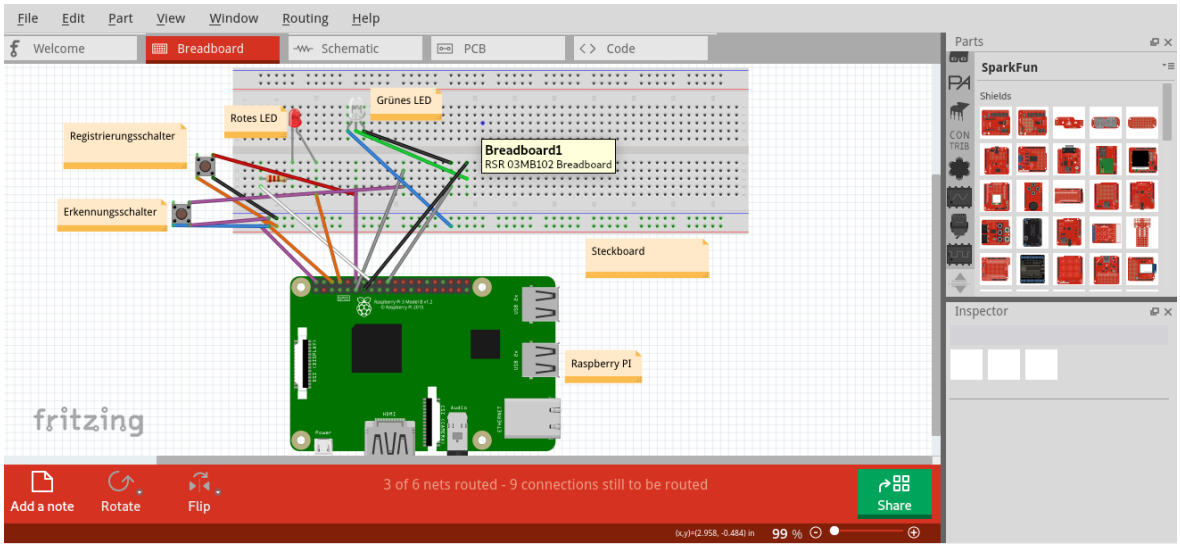


Abbildung 1.2: Schaltplan des Systems

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 Erkennungstaster[3] 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

[3]Auch 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.

### 1.2.2 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. [3]

- 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.1.2). 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 Fritzing kostenlos im Linux angeboten wird, was für Windows nicht der Fall ist.[1]

### 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.1.3

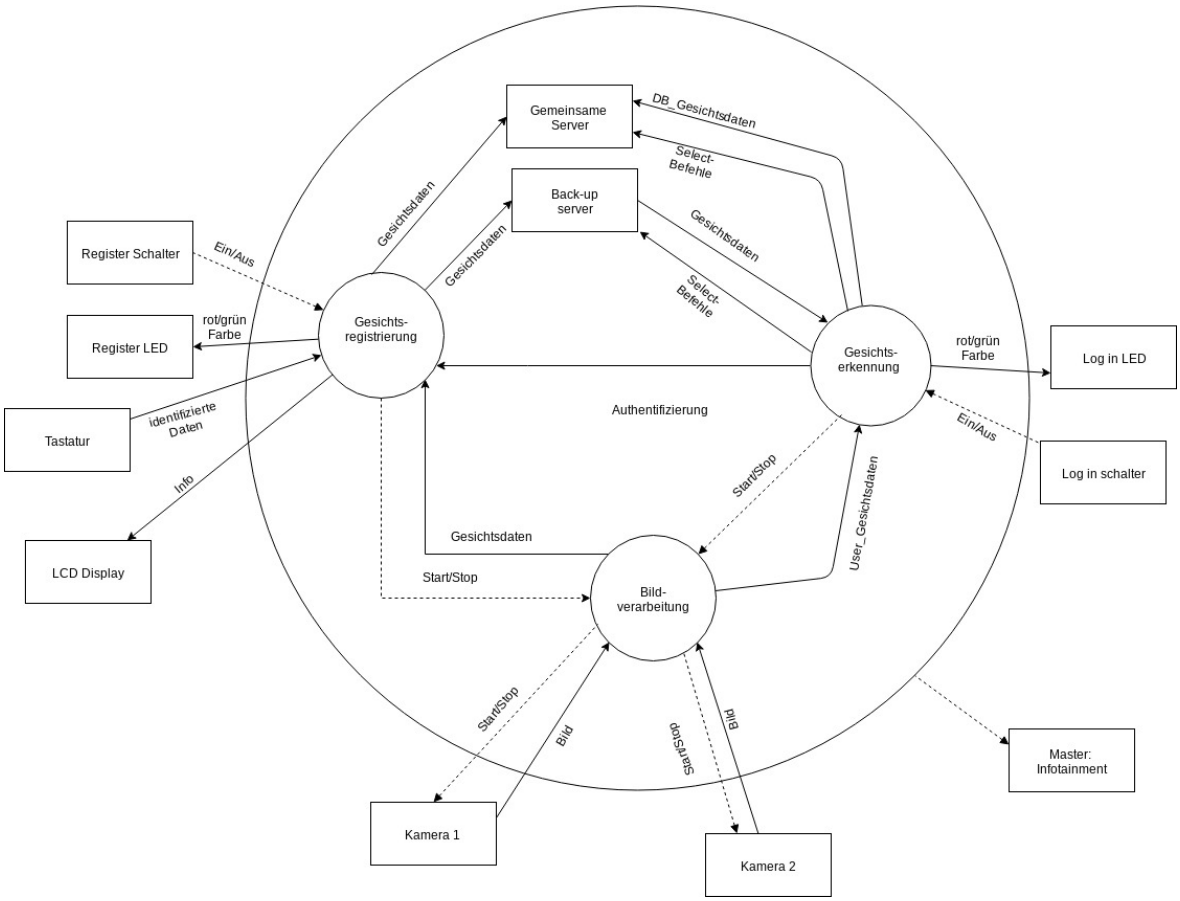


Abbildung 1.3: Big Picture

Die erste Ebene sieht man unter Abb.1.4. Sie ist ein detaillierte Version des Big Picture, das sich nur auf dem 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.

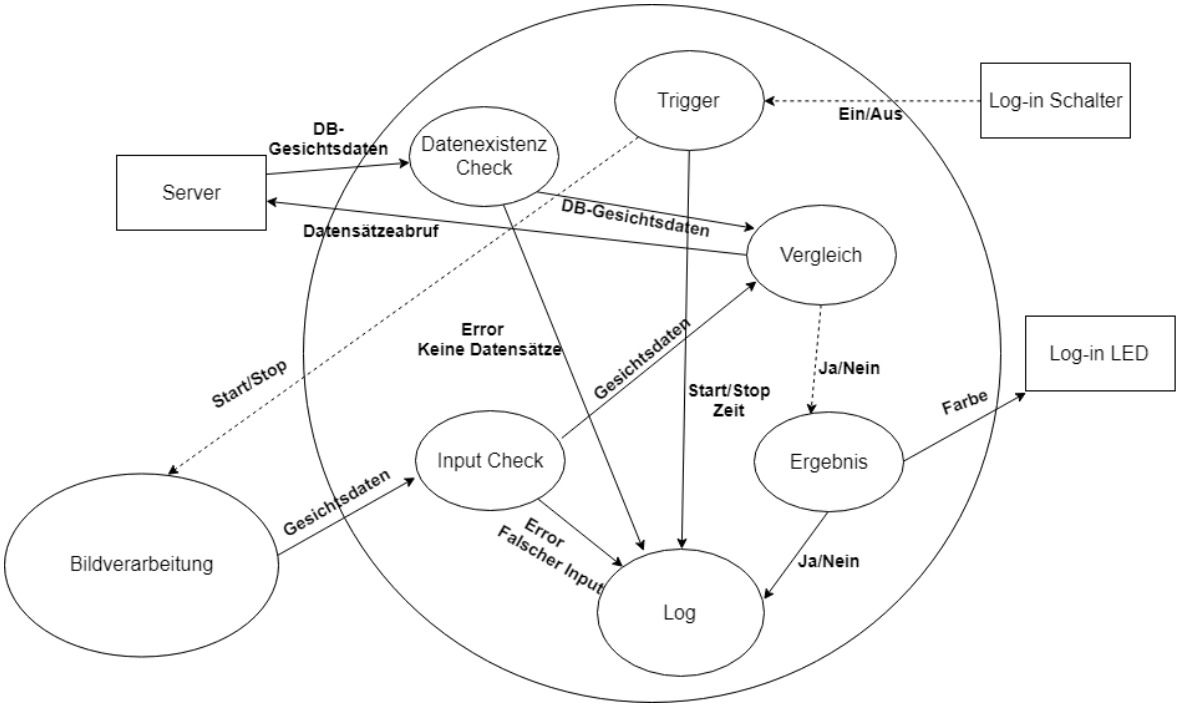


Abbildung 1.4: Erste Ebene

**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 Bildvergleichung 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 gemachtes Bild ist noch nicht bereit zum Vergleich. Um die Punkte richtige zu extrahieren, muss das Bild zuerst gekroppt bzw. geschnitten sein. Das heißt, die Dimensionen des Bildes werden reduziert, und somit werden die andere Objekten oder die andere Objekten 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.

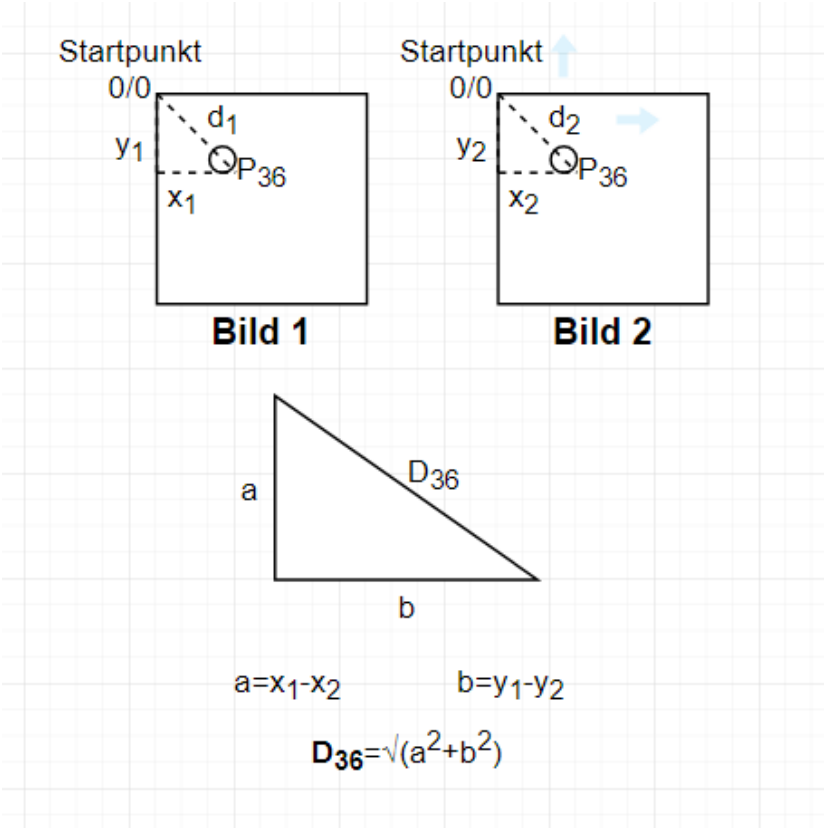


Abbildung 1.5: Vergleichsweg

Auf Abbildung 1.5 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 vergleicht 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 x1,x2,...xn zu ermitteln, muss man deren Produkt bilden und von diesem die n-te Wurzel ziehen."[2]

Auf Abbildung 1.6 wird das Formel, das sich ergibt gezeigt:

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

Abbildung 1.6: 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

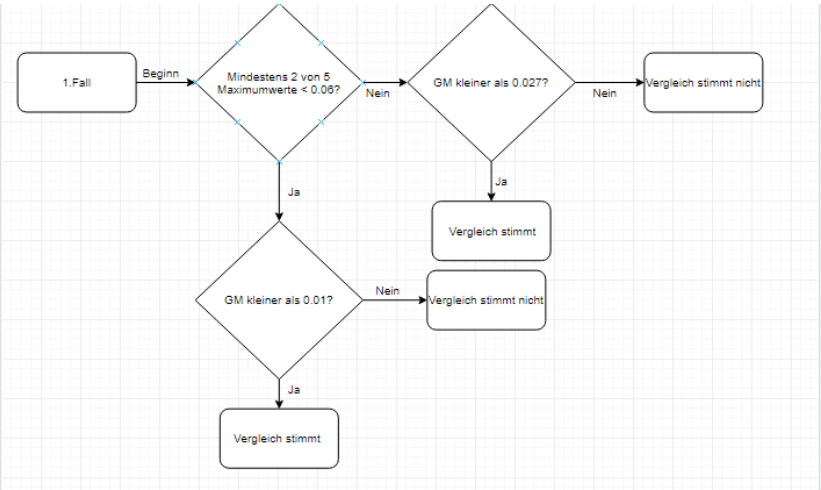


Abbildung 1.7: 1_Fall

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 1.7 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

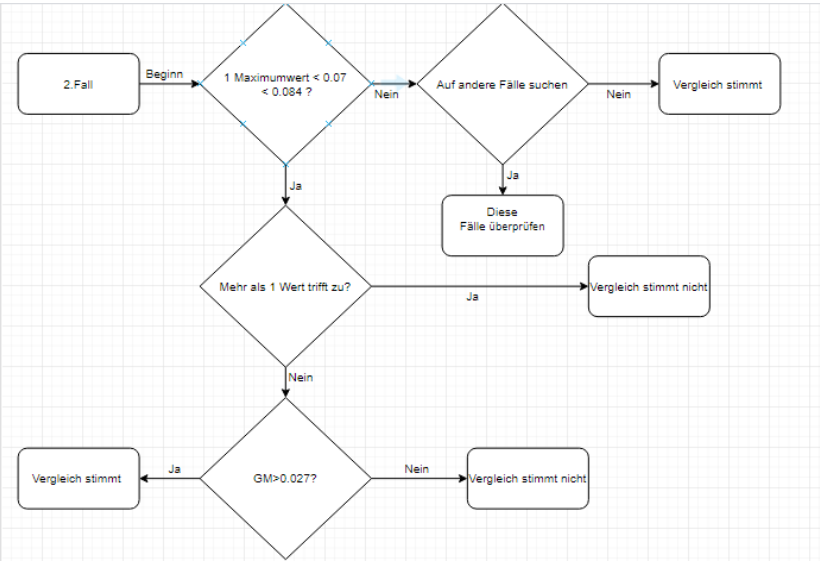


Abbildung 1.8: 2_Fall

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 1.8 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
```

```
                    status33=True
                    status4=True
#Wenn JA :Wenn gmT kleiner als 0.027 ist passt,
                    sonst FALSCH
            if count == 1:
                    global status33
                    status33=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

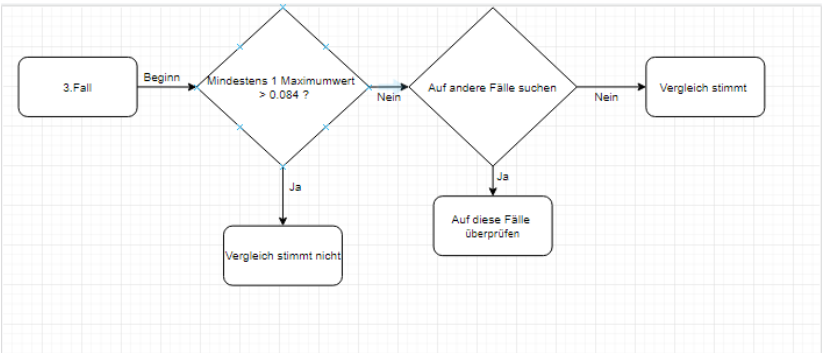


Abbildung 1.9: 3_Fall

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

Auf Abbildung 1.9 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 eingegeben

   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 verschieden 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 gedruckt'
log.error('This event occurred: %s' % test_var)
```

## 1.3 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 Pfaden 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 arbeitet, die aber von einem 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.

## 1.4 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 Ausfallsfunktion die bei dem Zustand logisch-1 einen Ausfall ausdrückt, bei logisch-0 liegt ein funktionsfähiges System vor. Sie gehört zu den "Top-DownAnalyseformen 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.

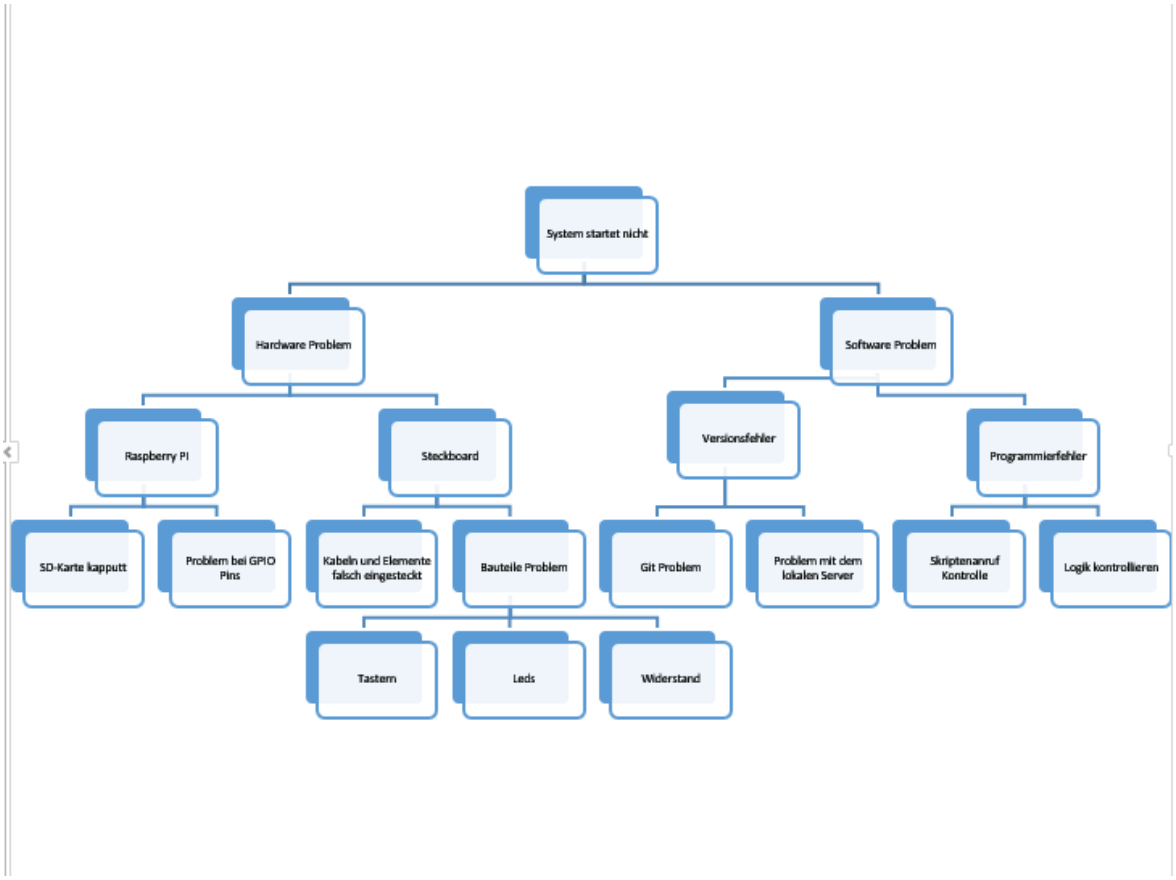


Abbildung 1.10: Fehlerbaum

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 1.10 wird ein Fehler in der Gesichtserkennung Teil des Systems beschrieben. Zuerst muss es überprüft werden, ob der Fehlern bei der Hardware oder Software liegt.

Hardware Problem?
Ist es ein Problem in Hardware, wird der Raspberry Pi und die Steckboard getestet. Beim Raspberry kann es entweder Probleme mit dem SD-Karte oder mit der GPIO Pins geben. Bei 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 Programmierfehlern. Diese Programmierfehlern können beim Logik(Algorithmen) oder beim Syntax auftreten(Skripten Aufruf, Variablen usw.)

## 1.5 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.

## 1.6 Was wurde gelernt und welche Verbesserungsmöglichkei 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 sie 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 2

# 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 Große der Arbeitspakete beim Beginn nicht richtig abgeschätzt weil wir uns nicht gut ausgekannt 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. 2.2 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 2.1: 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 erfolgsmäßig zu erkennen. | ✓ |
| Aufbau des Systems wurde erledigt. | ✓ |
| Es ist ein Benutzer-freundliches System mittels einer minimalen Anzahl von Schalter und einem Display erstellt. | ✓ |

Abbildung 2.2: Planung vs Realisierung

# Kapitel 3

# Evaluierung und Resümee

## 3.1 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

# Tabellenverzeichnis

# Literatur

## Aus dem Netz


[1] URL: https://fritzing.org/learning/.

[2] URL: https://de.serlo.org/mathe/stochastik/daten-datendarstellung/daten-kenngroessen/geometrisches-mittel.

[3] *Einführung in Computer Vision mit OpenCV und Python.* URL: https://blog.codecentric.de/2017/06/einfuehrung-in-computer-vision-mit-opencv-und-python/.

[4] *Unterschied Taster als Schließer, Taster als Wechsler und Schalter.* URL: https://dein-elektriker-info.de/taster-als-schliesser-und-taster-als-wechsler/.