h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

Hochschule Darmstadt
- Fachbereich Informatik -

Landmarkdetektion für dreidimensionale Gesichtsdaten

Abschlussarbeit zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.)
vorgelegt von

Martin Böckeler

Referent:	Prof. Dr. Thomas Horsch
Korreferent:	Dr. Xuebing Zhou

Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Die Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 1. Februar 2013 _____

Martin Böckeler

Abstakt

Die 3D-Gesichtserkennung hat in den vergangenen Jahren aufgrund gestiegener Rechenleistung und sinkender Sensorpreise immer weiter an Popularität gewonnen. Dabei spielt die automatische Landmarkdetektion eine wichtige Rolle. Durch sie lässt sich das Posenproblem während den Aufnahmen auslöschen. Da jedoch bei der Datengewinnung von 3D-Objekten auch unter optimalen Umgebungsbedingungen Aufnahmefehler entstehen, müssen diese normalerweise erst durch rechen- und zeitintensive Vorverarbeitungsschritte eliminiert werden, bevor Landmarks präzise detektierbar sind. Viele vorgestellte Ansätze zur 3D-Landmarkdetektion müssen daher einen Kompromiss zwischen einer geringen Ausführungszeit oder einer hohen Präzision eingehen. Diese Arbeit stellt ein effizientes, 3D-Landmarkdetektionsverfahren vor, dass diesen Kompromiss nicht eingehen muss. Das Verfahren ist aus einer Kombination von Classifiern, einer kaskadierten Suche und dem Ausnutzen anatomischer Gesichtseigenschaften in der Lage, ohne Preprocessing eine präzise und schnelle Detektion zu gewährleisten. Dabei ist es weitestgehend Posen und Mimik unabhängig und ermöglicht die Detektion von bis zu 10 Landmarks, in 410 msec. Die Detektionsrate beträgt über 99%. Die Detektionswerkzeuge sind auf Haar-Like Feature basierende Classifier, welche in speziellen, erstmalig in dieser Arbeit eingesetzten Gradient Darstellungen der 3D-Daten nach Landmarks suchen. Ein spezieller Trainingsprozess lehrt die Classifier zwischen einem beliebigen Bildausschnitt und einem Landmark zu differenzieren. Das Training und die Generierung der Classifier erfolgt mit den plattformübergreifenden, für die Programmiersprache C/C++ geschriebenen Open Source Computer Vision Programmbibliotheken.

Inhaltsverzeichnis

Erklärung **2**

Abstakt **3**

Abbildungsverzeichnis **6**

Tabellenverzeichnis **8**

1. Einleitung **9**
1.1. Biometrie 9
1.2. Gesichtserkennung 9
1.2.1. Vergleich der 2D- und 3D-Landmarkdetektion 10
1.2.2. Verfahren zur Datengewinnung von 3D-Objekten 10
1.2.3. Probleme bei der Datengewinnung von 3D-Gesichtern 12
1.3. Motivation 14
1.4. Herausforderung 14
1.4.1. Sensorleistung 14
1.4.2. Rechenleistung 15
1.5. Ziel 15
1.5.1. Rahmenbedingungen 17
1.6. Aufbau der Arbeit 17

2. Stand der Technik **18**
2.1. Gesichtserkennung 19
2.2. Landmarkdetektion 19

3. Gradient-Image-basierendes Landmarkdetektionsverfahren **21**
3.1. Gradient Image 22
3.1.1. Unterschiedliche Gradientendarstellung der verschiedenen Landmarks . 22
3.2. Viola-Jones Classifier 24
3.3. Sliding-Window 25

3.4. Zuverlässige Landmarkdetektion in beschränkten Suchregionen 26
3.4.1. Nasendetektion zum Setzen der Suchregionen 26
3.4.2. Suchregion der Alare 26
3.4.3. Suchregion der Pupillen 26
3.4.4. Suchregion Endocanthion 27
3.4.5. Suchregion Exocanthion 28
3.4.6. Suchregion Cheilion 29
3.5. Filterung falsch detektierter Landmarks 31
3.6. Clustering 32
3.7. Archivierung der Ergebnisse 34

4. Trainingsverfahren 35
4.1. Generierung der Trainingsdaten 35
4.1.1. GroundTruth Editor 35
4.1.2. Positive- und negative Images 36
4.1.3. Komprimieren positiver und negativer Images 37
4.2. Parameter des Trainingsprozesses 37

5. Testergebnisse und Evaluierung 41
5.1. Laufzeit 41
5.2. Präzision 44
5.2.1. Gegenüberstellung der einzelnen Classifier 46

6. Zusammenfassung und Ausblick 49
6.1. Zusammenfassung 49
6.2. Ausblick 49

7. Schlusswort 52

A. Userinterface des GroundTruth Editor 53

B. Struktur eines GroundTruth Editor Datensatzes 55

Stichwortverzeichnis 57

Literaturverzeichnis 59

Abbildungsverzeichnis

1.1. Streifenlichttopometrie und 3D Laserscan 11
1.2. Vergleich der Umgebungsbeleuchtung während einer aktiven 3D-Gesichtsaufnahme 12
1.3. Aufnahmefehler/Spikes im Bereich der Augenregion 13
1.4. Allgemeine Landmarkübersicht und Übersicht der detektierten Landmarks . . . 16

3.1. Gesamtablauf zur Landmarkdetektion 21
3.2. 3D-Abbildung einer Gesichtsoberfläche aus einem Datensatz der FRGC Datenbank 21
3.3. Beispielhafte Berechnung des x-Gradienten einer Bildzeile 23
3.4. Übersicht der verschiedenen Gradientendarstellungen 24
3.5. 2 Haar-Like Features, welche besonders gut die Augen beschreiben 25
3.6. Sliding-Window eines Classifiers, welches ein Bild nach Haar-Like Features durchsucht 25
3.7. Detektierte Nase und Suchbereich der Pupillen 27
3.8. Erstellen des Suchbereichs der rechten Endocanthion 28
3.9. Erstellen des Suchbereichs der rechten Exocanthion 29
3.10. Detektierte Nase und allgemeiner Suchbereich der Cheilions 30
3.11. Erstellen des Suchbereichs der rechten Cheilion 30
3.12. Funktionsweise der Treshold-Funktion 31
3.13. Die 1. Iteration des k-means++ Algorithmus 33
3.14. Vergleich eines guten und schlechten Detektionsergebnisses 34

4.1. Negatives und positives Image für das Training des Nasen-Classifiers 36
4.2. Übersicht der in OpenCV verwendeten Haar-Like Features 39

5.1. Laufzeiten der einzelnen Classifier 43
5.2. Sliding-Window eines Classifiers welches 2 Bilder nach Haar-Like Features durchsucht 44
5.3. Landmarkdetektion ohne das Eingrenzen der Suchregion 45
5.4. Präzisionsvergleich zwischen dem entwickelten Verfahren und der Detektion ohne vorherigem Setzen der Suchbereiche 47

5.5. True negative und false positive rate des entwickelten Verfahrens 48

6.1. Horizontales Setzen der Suchregionen bei 45° rotiertem Kopf 51

A.1. Screenshot des GroundTruth Editors 54

B.1. GroundTruth-Datensatz detektierter Landmarks 56

Tabellenverzeichnis

4.1. Übersicht der positiven Images des ersten Datensatzes 38
4.2. Parameter des Haartrainings der einzelnen Classifier 40

5.1. Skalierung des Sliding-Windows 42
5.2. Definition der Begriffe: false negative, true negative, false positive, true positive 47

1. Einleitung

1.1. Biometrie

Die Biometrie befasst sich mit den biometrischen Charakteristika eines Individuums. Ein biometrisches Charakteristika definiert der ISO/IEC 2382-37 Standard dabei wie folgt: *Ein biologisches oder verhaltensabhängiges Charakteristikum eines Individuums, von welchem sich zur Unterscheidung verwendbare, reproduzierbare biometrische Merkmale ableiten lassen, die zum Zwecke der biometrischen Erkennung/ automatischen Erkennung einsetzbar sind.* Ein Biologische Charakteristika ist unter anderem der Fingerabdruck, die Iris oder die DNA eines Menschen. Verhaltenscharakteristika beschreiben dagegen erlernte Fähigkeiten wie den Gang oder eine Unterschrift.

1.2. Gesichtserkennung

Es gibt eine Vielzahl von unterschiedlichen biometrischen Charakteristika, angefangen von dem Geruch, bis hin zum Venenbild in der Hand. Um diese aufzunehmen muss der Benutzer jedoch in direkten Kontakt mit einer Sensorfläche kommen, oder explizit mit einem biometrischen System kooperieren, was die Benutzerakzeptanz schmälert. Die Gesichtserkennung bietet in diesen Bereichen viele Vorteile gegenüber anderen Aufnahmeverfahren. Zum einen brauch der Benutzer in keinen körperlichen Kontakt mit dem System kommen, zum anderen können dennoch gute Proben genommen werden, auch wenn der Proband nicht vollständig kooperiert. Die aufgenommenen Proben lassen sich anschließend zur Identifikation oder Verifikation von Personen einsetzen. Bei einer Verifikation werden die Proben mit einer vorher, von dem Benutzer angegebenen Identität, verglichen. Sie stellt einen 1:1 Vergleich dar. Die Identifikation vergleicht die genommene Probe mit allen in der Datenbank gespeicherten, um die Identität einer Person zu bestimmen.

1.2.1. Vergleich der 2D- und 3D-Landmarkdetektion

2D

Die Mehrzahl der auf dem Markt erhältlichen Systeme zur Landmarkdetektion benutzen Intensitätsbilder des Gesichts [SCAS07]. Diese Intensitätsbilder können auch als 2D-Gesichtsbilder oder Fotos bezeichnet werden, in denen jeder Pixel einen Farbwert speichert. Ein 3D- Gesichtsbild speichert dagegen die Position jedes Messpunktes im dreidimensionalen Raum $\mathbb{R}^3$. Ein Vorteil der 2D- gegenüber der 3D-Landmarkdetektion liegt in der einfacheren und schnelleren Aufnahme der Gesichtsbilder. Die Nachteile der 2D-Landmarkdetektion bestehen darin, dass diese anfälliger auf Veränderungen der Umgebungsbeleuchtung und Veränderung der Oberflächentextur, durch z.B. Schminke reagiert [SA06]. So nimmt Datenbank übergreifend die Leistungsfähigkeit zweidimensionaler Verfahren, zunehmend durch unterschiedliche Lichtverhältnisse während den Aufnahmen, ab [DSA08].

3D

In der Regel wird behauptet, dass die 3D-Gesichtserkennung unabhängig von dem Umgebungslicht sei. An sich stimmt diese Behauptung auch, da die Oberfläche eines Gesichts unabhängig von der Umgebungsbeleuchtung ist. Jedoch wird das Gesicht immer erst von handelsüblichen Digitalkameras aufgenommen, die es als 2D-Bild darstellen. Beleuchtungsfehler in dem 2D-Gesichtsabbild übertragen sich auf die rekonstruierte 3D-Oberfläche. Solche Aufnahmefehler werden auch als Artefakte bezeichnet. Die Art der Artefakte unterscheiden sich bei 2D und 3D, sind aber oftmals ähnlich. Die Frage, bei welcher Technik mehr oder schwerwiegendere Artefakte entstehen, lässt sich dabei nicht pauschal beantworten und ist Anwendungs- und Sensorabhängig.

1.2.2. Verfahren zur Datengewinnung von 3D-Objekten

Biometrische Daten können auf unterschiedlichste Weise gewonnen werden, je nach verwendeten Einsatzzweck. Bei der 3D-Gesichtserkennung gibt es 3 Hauptansätze um ein Abbild der Gesichtsoberfläche zu erhalten.

Aktives Verfahren

Bei dem aktiven Verfahren wird eine Szene mit strukturiertem Licht oder einem Laserstrahl ausgeleuchtet und das vom Objekt Zurückgeworfene Licht mit einer Kamera aufgenommen. Das am häufigsten eingesetzte Verfahren, welches auf strukturiertem Licht beruht ist dabei die Streifenprojektion/Streifenlichttopometrie, bei der die geometrische Position von Kamera und Projektor wie bei dem 3D-Laserscan in einem zuvor festgelegten Verhältnis zueinander stehen. Die Oberfläche eines Objektes wird durch den Projektor mit codiertem Streifenlicht beleuchtet. Die Kamera nimmt das einfallende Licht und dessen, durch die Objektoberfläche verzerrte Struktur wieder auf. Durch die Verzerrung der Lichtstruktur kann ein 3D-Objekt berechnet werden. Siehe dazu Abbildung 1.1 Seite 11. Die Vorteile des aktiven Verfahrens

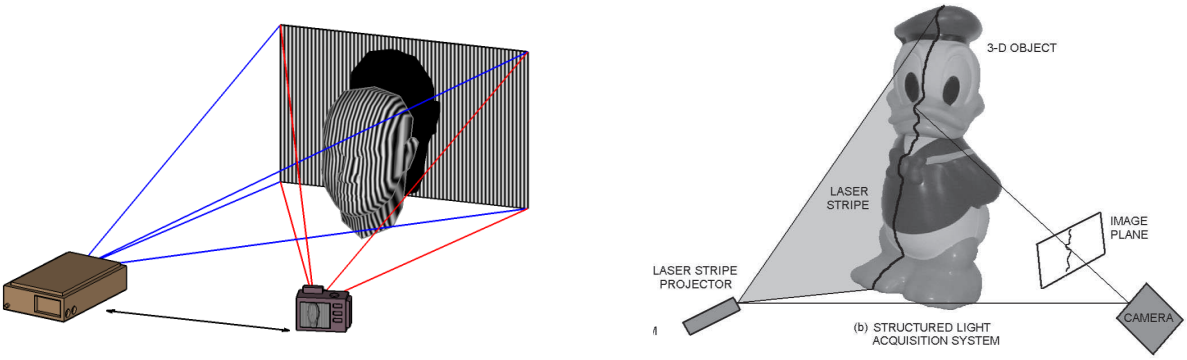


Abbildung 1.1.: Gewinnung dreidimensionaler Daten. links die Aktive Streifenprojektion, Bildquelle [Alc12]. Rechts ein Laserscanner, Bildquelle [MBO04].

gegenüber dem passiven sind hierbei unter anderem die Reichweite. Während bei dem passiven Verfahren die Objekte bis zu einem Abstand von 0,3 m zuverlässig gescannt werden können, liegt die Reichweite bei dem aktiven Verfahren bei ungefähr 1,0 m [KON]. Zusätzlich ist das Aufnahmeergebnis in puncto Genauigkeit dem passiven Verfahren überlegen. Typischerweise treten bei diesem Verfahren jedoch Aufnahmefehler im Bereich der Augen und der Augenbrauen auf. Außerdem muss die Luminanz des Projektors/Lasers genau abgestimmt werden. Ist diese zu hoch eingestellt, kann es zu gesundheitlichen Risiken der Augen kommen oder es treten geschlossene Augen während einer Aufnahme auf. Ist das Projektionslicht dagegen zu schwach eingestellt, ist die Aufnahme anfälliger für Umgebungslicht. Siehe dazu Abbildung 1.2 Seite 12, die auf der linken Seite eine Aufnahme mit gutem Umgebungslicht darstellt und auf der rechten Seite eine Aufnahme mit massiven Aufnahmefehlern, hervorgerufen durch ein Studiopotlight in 1,5 m Entfernung zum Kopf. Die Gesichtsdaten aus der in dieser Arbeit verwendeten Datenbank sind von einem aktiven 3D Laser Scanner, dem Minolta Vivid 900/910 Laser gewonnen.

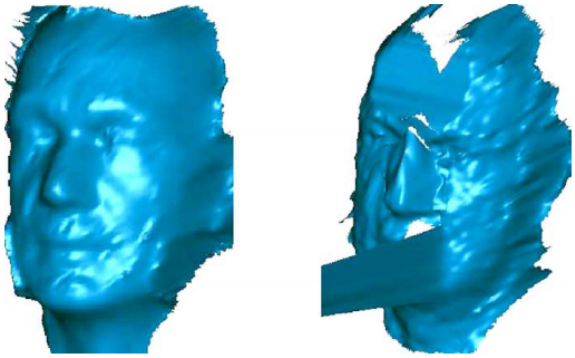


Abbildung 1.2.: Vergleich der Umgebungsbeleuchtung während einer aktiven 3D-Aufnahme des Gesichts. Auf der linken Seite eine qualitativ hochwertige Aufnahme, auf der rechten Seite eine qualitativ schlechte Aufnahme. Bildquelle [BCF06].

Passives Verfahren

Bei dem passiven Stereoverfahren nehmen zwei Kameras eine Szene aus unterschiedlichen Blickwinkeln auf. Die Position der Kameras zueinander muss auch hier wieder vorher kalibriert werden. Aus der aufgenommenen Szene lassen sich dann einzelne Punkte und deren Koordinaten berechnen. Die Vorteile dieses Verfahrens sind dessen Aufnahmegeschwindigkeit. Diese ist höher als die des aktiven Verfahrens, da nur ein einzelnes Bild gemacht werden muss. Ein Nachteil dieses Verfahrens ist, dass in Regionen in denen nicht viel unterschiedliche Texturen vorhanden sind, es oft zu Aufnahmefehlern kommt und die betroffenen Region zu glatt dargestellt werden.

Hybrid Verfahren

Das Hybridverfahren kombiniert das Streifenprojektionsverfahren mit dem passiven Stereoverfahren. Dabei wird das Objekt mit strukturiertem Licht beleuchtet und mit mehreren verschiedenen Kameras, von unterschiedlichen Blickwinkeln aufgenommen. Ziel ist es, die Vorteile beider Verfahren zu kombinieren.

1.2.3. Probleme bei der Datengewinnung von 3D-Gesichtern

Bei der Datengewinnung von 3D-Gesichtern kommt es immer wieder durch Licht und Bewegungen des Gesichts während einer Aufnahme zu Aufnahmefehlern.

Licht

Während der Aufnahme eines Gesichts kann es trotz idealer Umgebungsbeleuchtung immer wieder zu Artefakten, kommen. Diese werden durch Reflexionen und einer ungünstigen Pose hervorgerufen. Reflexionen entstehen durch Lichtspiegelungen der Sensorbeleuchtung durch Hautfett, der Lederhaut des Auges oder der Haare und drücken sich durch „Spikes“ aus. Siehe Abbildung 1.3 Seite 13. Während es sich bei Spikes um einzelne, unkorrekte Datenpunkte handelt, sind sogenannte „Holes“, ganze Areale, in denen Daten nicht erfassbar sind.

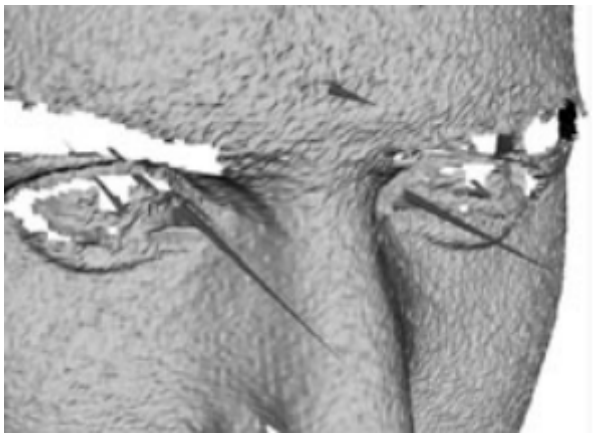


Abbildung 1.3.: Messfehler bei der aktiven 3D Aufnahme im Bereich der Augen. Hervorgerufen durch Lichtspiegelungen der Sensorbeleuchtung mit der Lederhaut des Auges. Bildquelle [Spr11].

Bewegungen während der Aufnahme

Je kürzer die Aufnahmedauer umso besser die Aufnahmeergebnisse, da durch Bewegungen des Gesichts die Aufnahmen verfälschen. Die Testperson sollte also nahezu unbeweglich vor dem Aufnahmegerät verharren bis die Aufnahme vollständig beendet ist. Verfälschte Aufnahmen durch eine unruhige Testperson treten bei beiden Verfahren jedoch meist nur bei der Identifikation von Personen auf. Bei der Verifikation kann man davon ausgehen, dass das Subjekt kooperiert und sich möglichst wenig bewegt.

1.3. Motivation

Die automatische Positionsbestimmung von Landmarks ist ein wichtiges und herausforderndes Forschungsgebiet. Sie sind in vielen Bereichen der Computer Vision und der Biometrie, wie der Gesichtserkennung oder der Identifikation und Verifikation von Personen unerlässlich. Anhand von Landmarks lassen sich rotierte oder gedrehte Gesichter in eine normalisierte Position bringen, um gewonnene Proben mit denen einer Datenbank vergleichen zu können. Die meisten bisherigen Verfahren zur Landmarkdetektion beruhen auf 2D-Images. Diese sind jedoch sehr anfällig für Umgebungsbeleuchtung, der Oberflächentextur des Gesichts, die sich z.B. stark durch Make-up ändert, der Pose und der Mimik. In den vergangenen Jahren hat aus diesen Gründen und der gestiegenen Rechenleistung daher die Popularität der 3D-Gesichtserkennung immer weiter zugenommen.

1.4. Herausforderung

Die Herausforderung bei der 3D-Landmarkdetektion ist eine präzise und gleichzeitig schnelle Detektion der Landmarks zu erreichen. Dies wird allerdings zurzeit noch durch die Sensorleistung der Aufnahmesysteme und der Rechenleistung heutiger Computersysteme begrenzt.

1.4.1. Sensorleistung

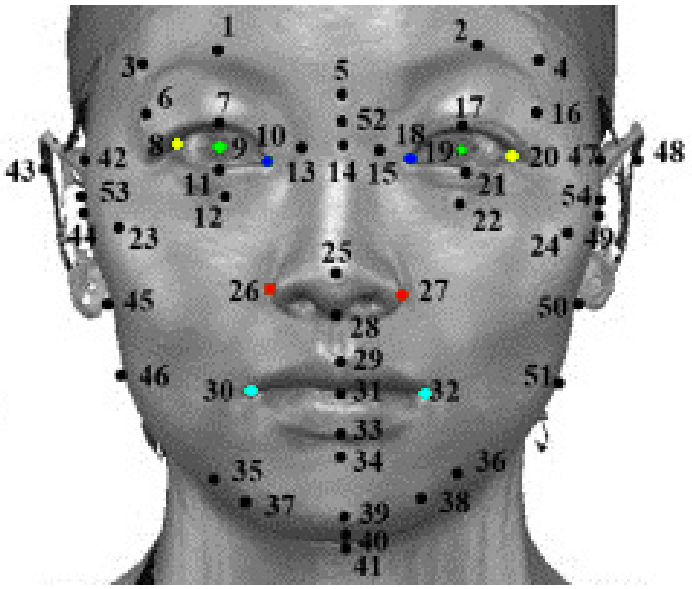
Die heutigen Scannersysteme sind auch bei optimalen Umgebungsbedingungen nicht in der Lage, fehlerfreie Gesichtsaufnahmen zu erstellen. Die Fehlerquellen sind dabei sehr vielfältig und reichen von der Laserintensität, dem Abstand der Gesichtsoberfläche zum Sensor, den optischen Komponenten eines Systems wie der Linse oder den internen Spiegeln der Kamera bis hin zur Herstellersoftware der Systeme, welche die aufgenommenen Daten digitalisiert. Sun et al. [SLML09] gibt einen detaillierten Überblick über die Fehlerquellen eines 3D-Scannersystems. Die Störungen treten oft in den Bereichen der Augen, der Nase oder der Mundwinkel auf, welche gleichzeitig auch die Hauptansatzpunkte der Landmarkdetektion im Gesicht darstellen.

1.4.2. Rechenleistung

Zur Beseitigung von Aufnahmestörungen und um so eine präzise Positionsbestimmung der Landmarks garantieren zu können, nutzen die meisten Verfahren Vorverarbeitungsschritte wie eine lineare oder bikubische Interpolation. Diese rechenintensiven Vorverarbeitungsschritte verlangsamen allerdings den Prozess der Landmarkdetektion. Daher müssen die meisten Verfahren zur 3D-Landmarkdetektion immer einen Kompromiss zwischen einer geringen Programmlaufzeit oder einer präzisen Positionsbestimmung der Landmarks eingehen. Um in Zeiten von biometrischen Pässen und biometrischen Eingangskontrollen eine Benutzerakzeptanz zu erreichen, ist jedoch beides, eine hohe Präzision und eine geringe Detektionszeit erforderlich.

1.5. Ziel

Ziel dieser Arbeit ist es, ein Landmarkdetektionsverfahren zu entwickeln, welches in 3D Gesichtsdaten die Position verschiedener Landmarks zuverlässig bestimmt und ohne aufwendige Vorverarbeitungsschritte auskommt. Dabei soll das Verfahren möglichst unabhängig von Pose und Mimik sein. Abbildung 1.4 Seite 16 zeigt eine Übersicht aller Landmarks des Gesichts. Die in dieser Arbeit detektierten Landmaks sollen die Punkte: 8 Exocanthion rechts, 9 Pupille rechts, 10 Endocanthion rechts, 18 Endocanthion links, 19 Pupille links, 20 Exocanthion links, 26 Alare rechts, 27 Alare links, 30 Cheilion rechts und 32 Cheilion links sein. Diese Landmarks sind für die Detektion ausgewählt, da ihre Position genau definierbar ist. So lässt sich z.B. die Postion der Nasenspitze, bei Personen mit einer flachen Nase, nicht eindeutig bestimmen. Ein anderer Vorteil dieser Punkte ist ihre Robustheit gegenüber bestimmten Posen. Die Detektion beider Ohrläppchen ist z.B. unmöglich, sobald die Person bei der Aufnahme ein bisschen zur Seite guckt. In dem nachfolgenden Text sind die Begriffe aus Abbildung 1.4 Seite 16 als Synonyme der Landmarks verwendet. Des weiteren soll eine Performanceanalyse die Leistungsfähigkeit des Systems einordnen.



1 Superciliare rechts	19 Pupille links	37 Kinn rechts
2 Superciliare links	20 Exocanthion links	38 Kinn links
3 Frontotemporale links	21 Palpebrale inferius links	39 Kinn
4 Frontotemporale links	22 Infraorbitale links	40 Menton
5 Glabella	23 Lat. zygomatie rechts	41 Nacken
6 Zygofrontale rechts	24 Lat. zygomatie links	42 Otobasion superius rechts
7 Palpebrale superius rechts	25 Pronasale	43 Postaurale rechts
8 Exocanthion rechts	26 Alare rechts	44 Tragion rechts
9 Pupille rechts	27 Alare links	45 Otobasion inferius rechts
10 Endocanthion rechts	28 Subnasale	46 Gonion rechts
11 Palpebrale inferius rechts	29 Labiale superius	47 Otabasion superius links
12 Infraorbitale rechts	30 Cheilion rechts	48 Postaurale links
13 Nasal root point rechts	31 Stomion	49 Tragion links
14 Sellion	32 Cheilion links	50 Otobasion inferius links
15 Nasal root point links	33 Labiale inferius	51 Gonion links
16 Zygofrantale links	34 Sublabiale	52 Nasion
17 Palpebrale superius links	35 Äußeres Kinn rechts	53 Zygion rechts
18 Endocanthion links	36 Äußeres Kinn links	54 Zygion links

Abbildung 1.4.: Eine Übersicht aller im Gesicht befindlichen Landmarks. Die in dieser Arbeit detektierten Landmarks sind die Punkte: 8, 9, 10, 18, 19, 20, 26, 27, 30 und 32. Bildquelle [LBJ12].

1.5.1. Rahmenbedingungen

Als Datenbasis der Arbeit dient die FRGC v2 Datenbank. Die Detektion beruht auf Classifiern, die in einer visuellen Darstellung der FRGC-Daten, so genannten Gradient-Images, nach den Landmarks suchen. Das Training und die Implementierung des Systems ist eine Kombination aus der Programmiersprache C/C++ mit den Open Source Computer Vision (OpenCV) Programmbibliotheken.

1.6. Aufbau der Arbeit

Der Aufbau dieser Arbeit gliedert sich in 7 Kapitel. Kapitel 1 gibt eine kurze Einführung in die Biometrie und erläutert die Vorteile der 3D-Gesichtserkennung gegenüber anderen Verfahren. Es beinhaltet die Motivation einer Implementierung eines Landmarkdetektionssystems, sowie die damit verbundenen Herausforderungen. Das Ende der Einleitung bildet die Zielsetzung mit den Rahmenbedingungen. Kapitel 2 stellt 5 Verfahren über die bisher verfolgten Ansätze der 3D-Landmarkdetektion vor und geht kurz auf die damit verbundenen Nachteile ein. Kapitel 3 erläutert das in dieser Arbeit entwickelte Verfahren, gibt Hintergrundinformationen zu den verwendeten Algorithmen und liefert zum besseren Verständnis derer Funktionsweisen kleinere Beispiele. Das Trainingsverfahren der eingesetzten Classifier, dass die Generierung der Trainingsdaten und die Vorstellung der Trainingsprozessparameter inkludiert, ist in Kapitel 4 beschrieben. Die Laufzeit und Präzision des fertigen Projekts nimmt Kapitel 5 näher unter die Lupe. Kapitel 6 zieht ein Resümee der erreichten Ziele und gibt ein Ausblick für zukünftige Erweiterungen. Den Abschluss dieser Arbeit bildet das Schlusswort in Kapitel 7.

2. Stand der Technik

Das Forschungsgebiet, welches sich mit 3D-Gesichtsdaten beschäftigt, hat in den vergangenen Jahren kontinuierlich an Popularität gewinnen können. Zum einen liegt das an der gestiegenen Rechenleistung heutiger Computersysteme und zum anderen an dem preislichen Aspekt, der die Sensoren zur Datenerfassung betrifft. Bei ständig sinkenden Sensorpreisen sind diese gleichzeitig in der Lage immer bessere Aufnahmeergebnisse zu erzielen. Infolge dieses Trends steht der Wissenschaft im zunehmenden Maße immer umfangreichere Datenbanken zur Verfügung. Dieses Kapitel gibt einen kurzen Überblick über den Stand der Technik in den Bereichen Gesichtserkennung/Gesichtsdetektion und Landmarkdetektion.

FRGC Datenbank

Die FRGC Datenbank war ein wichtiger Schritt für die 3D-Landmarkdetektion. Bis 2003 umfassten die Datenbankvolumina von nicht mehr als 100 verschiedener aufgenommener Personen [BCF06]. Erst als 2002 der Face Recognition Ventor Test flächendeckend die schlechte Leistungsfähigkeit der kommerziell auf dem Markt erhältlichen 3D-Gesichtserkennungssysteme aufdeckte, startete auf diesem Gebiet ein umfassendes Forschungsprojekt. Das Projekt, dass den Namen Face Recognition Grand Challange (FRGC) trug, lief von 2004 bis 2006 und hatte das Ziel, weiterentwickelte Technologie für die 3D-Gesichtserkennung bereitzustellen [Nat11]. Die Technologie sollte mit der, zu dem damaligen Zeitpunkt, bereits bestehenden Forschung der U.S. Regierung kompatibel sein. Aus dem FRGC-Projekt ist die FRGC v2 Datenbank hervorgegangen, die unter anderem auch dieser Arbeit den Zugang zu 4096 einheitlichen Gesichtsscanns bereitstellt.

2.1. Gesichtserkennung

Yang et al. [YKA02] definiert in seiner Arbeit den Begriff der Gesichtserkennung, der sinngemäß lautet: *Ziel der Gesichtserkennung ist es herauszufinden, ob in einem beliebigen Bild Gesichter vorhanden sind und bei Präsenz eines Gesichts, dessen Position sowie dessen Abmessungen zu bestimmen.* Colombo et al. [CCS06] nutzt zur Gesichtserkennung den Ansatz der Kurvenanalyse. Das Verfahren untersucht Objektoberflächen in einem Range-Image und detektiert Anhöhungen, die in der Regel bei der Nasenspitze und den beiden Augen zu finden sind. In einem weiteren Schritt untersucht ein auf dem PCA-Algorithmus basierender Classifier, ob es sich bei den jeweiligen Triplets um ein Gesicht handelt. Von insgesamt 150 vorhanden Gesichtern liegt die Erfolgsrate der gefundenen bei 82%. Der Nachteil der auf Kurvenanalyse basierender Verfahren liegt in den rechenintensiven Vorverarbeitungsschritten, welche Artefakte, hauptsächlich im Bereich der Augen und Mundregion eliminieren, bevor die eigentliche Analyse startet. Außerdem vernichtet das Glätten der Gesichtsoberfläche viele lokale Kurveninformationen, da es die Kurveninfomation von Nachbarschaftspunkten angleicht. Pan et al. [PWW03] arbeitet mit Profilen von Objekten in Range Images. Um Gesichter zu erkennen richtet das Verfahren alle Objekte eines Bildes mittels ICP-Algorithmus vertikal symmetrisch aus, findet die Symmetrielinie der Objekte und vergleicht die daraus berechneten Profile mit zuvor gespeicherten Gesichtsprofilen. Eine Glättung der Oberflächenstruktur ist hierbei nicht nötig, da der Vergleich der Profile mittels k-th Hausdorff Distanz geschieht. Die Performance des Verfahrens ist auf der 3D_RMA Datenbank getestet, die Teil des M2VTS Projektes ist. Die equal error rate liegt bei 6,67% (DBs1m+s2m) bzw. 8,89% (DBs1a+s2a). Obwohl dieses Verfahren keine vorverarbeitende Bildglättung vornimmt, ist das Finden der Symmetrieliene sehr zeitaufwändig. So dauert es auf einem P4 mit 2,0 GHz 2,2 Sekunden um 2 Profile miteinander zu vergleichen.

2.2. Landmarkdetektion

Bei der Landmarkdetektion versucht man die Position einzelner, bestimmter Punkte eines Gesichts zu bestimmen. Dibeklioglu et al. [DSA08] präsentiert einen datenbankübergreifenden Algorithmus, der auf statistischen Modellen basiert. Dabei sucht der Algorithmus in einem Bild nach Punkten, die in ihren lokalen Eigenschaften denen der statistischen Werten jedes einzelnen Landmarks am meisten ähneln. Die Leistungsfähigkeit des Algorithmus ist auf der Bosphorus-3D Gesichtsdatenbank und auf der FRGC Datenbank getestet. Der Schwellwert, ab dem ein Landmark als falsch detektiert gilt, liegt bei 3 Pixel. Das beste Ergebnis mit einer De-

tektionsrate von 100% für die Endocanthion ist bei Verwendung der BOSv1 Datenbank für das Training und der BOSv2 Datenbank für das Testing gemessen. Die schlechteste Detektionsrate von 48,42% tritt bei der Cheilion in Verwendung der FRGCv1 Datenbank zum Training und der BOSv1 Datenbank zum Testen auf. Perakis et al. [PPTK10] stellt 2 weitere Methoden vor um Landmarks mittels Shape-Indizes und Extrusionskarten oder mittels Shape-Indizes und Spin-Images zu lokalisieren. Bei der ersten Methode sind die Minima der Shape-Indizes Kandidaten für die Außenkanten der Augen und des Mundes, die Maxima sind Kandidaten für die Nasen- und Kinnspitze. Als Distanzmetrik findet die normalisierte Procrustes Distanz Verwendung. Die zweite Methode unterteilt mit Hilfe von Spin-Image-Templates die Maxima und Minima der Shape-Indizes in 5 weitere Klassen. So findet eine Unterscheidung zwischen Endocanthion und Exocanthion, sowie eine zwischen Nasen- und Kinnspitze statt. Zur Bestimmung der Landmarks verwendet diese Methode eine Kombination aus der normalisierten Procrustes Distanz und der durchschnittlichen Ähnlichkeitsdistanz der Spin-Images. Methode 1 erreicht auf der FRGC Datenbank eine Abweichung kleiner 10 Pixel von $\approx$ 83% und Methode 2 eine von 97,23%.

Gradient-Images und Classifier

Die nachfolgende Arbeit hat ihre Hauptimpulse aus dem 2011 veröffentlichtem Paper von Ajmal Mian [Mia11]. Mian beschreibt dort wie er mit Hilfe von Classifiern, Gradient- und Range-Images die Nasenspitze sowie die beiden Exocanthions aus den Datensätzen der FRGC Datenbank detektiert. Zur Detektion der Exocanthions ist das Range-Image und die y-Gradient Darstellung, zur Detektion der Nase zusätzlich die x-Gradient Darstellung verwendet worden. Die Abweichung, ab der ein Landmark noch als richtig detektiert gilt, liegt bei 20 mm. Die Detektionsrate der rechten Exocanthion ist mit 99,68%, die der linken Exocanthion mit 99,90% und die der Nasenspitze mit 99,18% angegeben.

3. Gradient-Image-basierendes Landmarkdetektionsverfahren

In diesem Kapitel wird das Gradient-Image-basierendes Landmarkdetektionsverfahren vorgestellt, sowie dessen Vor- und Nachteile besprochen. Ein Gesamtüberblick des Ablaufs ist in Abbildung 3.1 Seite 21 dargestellt. Dieser gliedert sich in die Hauptbestandteile der Gene-

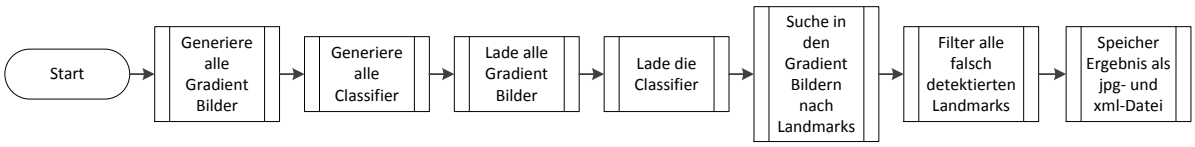


Abbildung 3.1.: Gesamtablauf zur Landmarkdetektion. Nachdem alle Gradient-Bilder und Classifier generiert sind, können diese zur Landmarkdetektion verwendet werden. Die Abspeicherung der Ergebnisse erfolgt als .jpg und .xml Datei.

rierung der Gradient-Bilder, der Erstellung der Classifiern, der Landmarkdetektion durch die Classifier und der Abspeicherung der Ergebnisse in Form von .jpg- und .xml-Dateien. Als Rohdatenquelle kommt die FRGC Datenbank zum Einsatz. Abbildung 3.2 Seite 21 zeigt Oberfläche eines Gesichts, dreidimensional abbildet.

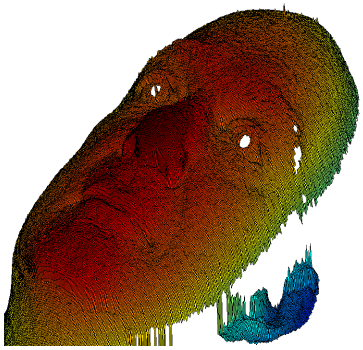


Abbildung 3.2.: 3D-Abbildung einer Gesichtsoberfläche aus einem Datensatz der FRGC Datenbank.

3.1. Gradient Image

Der erste Schritt in diesem Verfahren ist aus den Rohdaten des Scanners ein Grauwertbild zu erstellen, dessen Farbwerte von 0 bis 255 reichen. Dieses geschieht durch den Gradienten der Scannerdaten, der die Tiefendifferenz benachbarter Messpunkte auswertet. Die Gradientendarstellung kann in x- oder y-Richtung vollzogen werden, wobei der Nullpunkt in der linken oberen Bildecke liegt. Siehe dazu Abbildung 3.3 auf Seite 23, welche beispielhaft die Gradientenberechnung einer kleinen 5×4 Matrix beschreibt. Der x-Gradient wertet von jedem Messpunkt ausgehend die z-Differenz des linken und rechten Nachbarn ($\Delta x = 2$), der y-Gradienten die des darüber liegenden und des darunterliegenden Nachbarn aus. Da am Rand der Matrix nicht beide Nachbarn vorhanden sind, berechnet der Gradient den Wert des eignen Pixels und des vorhandenen Nachbarn als Differenz. Am Rand der Matrix ist Δx und Δy somit 1. Durch Abbilden der Ergebnismatrix auf den Wertebereich 0-255 entsteht das Gradient-Image.

3.1.1. Unterschiedliche Gradientendarstellung der verschiedenen Landmarks

Die Landmarkdetektion kann auf verschiedenen Gradient Bildern angewendet werden. Die Endocanthions, Nasenflügel und Mundwinkel lassen sich auf dem y-Gradienten am besten erkennen. Daher arbeitet das Verfahren bei diesen Landmarks mit der y-Gradient Darstellung. Die Suche der rechten Exocanthion erfolgt in dem x-Gradient Image, die der linken Exocanthion in dem gespiegeltem x-Gradient Image. Zum Detektieren der Pupillen findet erstmalig in dieser Arbeit ein xAbs-Gradient Image Verwendung. Das Erstellen des xAbs-Gradienten erfolgt, indem von dem x-Gradienten jeweils die absoluten z-Differenzen zur Bilddarstellung verwendet werden. Da es bei der Datenerfassung durch Lichtspiegelungen in der Region der Pupillen immer zu Bereichen kommt, in denen die Daten nicht erfasst werden können, sind diese Pixel bei dem Gradient Image entweder komplett Schwarz oder komplett Weiß. Durch Invertieren aller schwarzen Pixel auf den Wert 255 (Weiß) entsteht eine gut erkennbare, weiße Pupille, die sich deutlich zu ihrer dunklen Umgebung abgrenzt. Abbildung 3.4 auf Seite 24 zeigt eine Übersicht der verschiedenen Gradientendarstellungen.

Bildmatrix

	x0	x1	x2	x3	x4
y0	0	2	4	16	2
y1	2	3	8	1	1
y2	15	11	3	6	9
y3	0	4	9	5	2

x-Gradient

	x0	x1	x2	x3	x4
y0	2	2	7	-1	-14
y1	1	3	-1	-3,5	0
y2	-4	-6	-2,5	3	3
y3	4	4,5	0,5	-3,5	-3

y-Gradient

	x0	x1	x2	x3	x4
y0	0	2	4	16	2
y1	2	3	8	1	1
y2	15	11	3	6	9
y3	0	4	9	5	2

$(x0, y0):$

$\Delta x_{x0,y0} = 1$

$\Delta z_{x0,y0} = 2 - 0 = 2$

$\Delta F_{x0,y0} = \frac{\Delta z_{x0,y0}}{\Delta x_{x0,y0}} = \frac{2}{1} = 2$

$(x1, y0):$

$\Delta x_{x1,y0} = 2$

$\Delta z_{x1,y0} = 4 - 0 = 4$

$\Delta F_{x1,y0} = \frac{4}{2} = 2$

$(x2, y0):$

$\Delta x_{x2,y0} = 2$

$\Delta z_{x2,y0} = 16 - 2 = 14$

$\Delta F_{x2,y0} = \frac{14}{2} = 7$

$(x3, y0):$

$\Delta x_{x3,y0} = 2$

$\Delta z_{x3,y0} = 2 - 4 = -2$

$\Delta F_{x3,y0} = \frac{-2}{2} = -1$

$(x4, y0):$

$\Delta x_{x4,y0} = 1$

$\Delta z_{x4,y0} = 2 - 16 = -14$

$\Delta F_{x4,y0} = \frac{-14}{1} = -14$

Abbildung 3.3.: Berechnung des x-Gradienten der ersten Bildzeile. An dem rechten und linken Rand bei (x0,y0) und (x4,y0) gibt es in x-Richtung nur einen Nachbarpixel, daher ist an diesen Punkten $\Delta x = 1$. Δz entsteht durch Subtraktion der beiden Nachbarpixel in x-Richtung, bzw. am Rand durch Subtraktion des eigenen Pixelwertes mit einem vorhanden Nachbarpixel in x-Richtung.

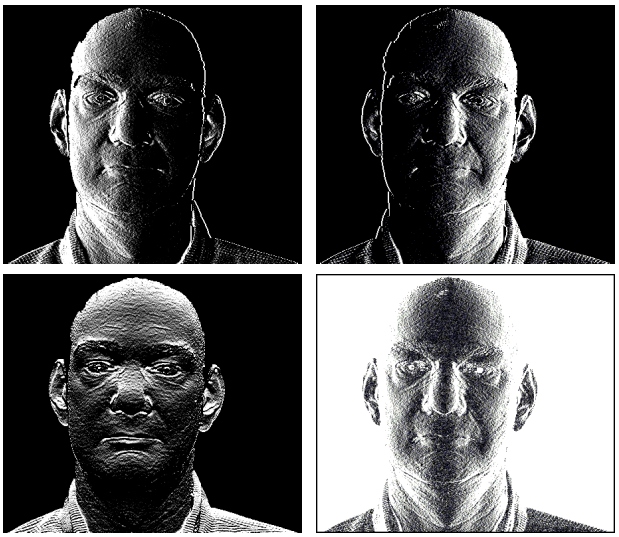


Abbildung 3.4.: Übersicht der verschiedenen, zur Landmarkdetektion eingesetzten Gradientendarstellungen. Von links oben nach rechts unten: x-Gradient zur Detektion der rechten Exocanthion, x-Gradient-Gespiegelt zur Detektion der linken Exocanthion, y-Gradient zur Detektion der Nasenflügel, der Mundwinkel und der Endocanthions, xAbs-Gradient zum Erkennen der Pupillen.

3.2. Viola-Jones Classifier

Integralbild

2001 veröffentlichten die Autoren Paul Viola und Michael Jones ein neues Verfahren zur Visuellen Objektdetektion [VJ01], dass nicht direkt in einem Bild nach einem Objekt sucht, sondern zuerst ein Integralbild von diesem erstellt. Das Integralbild kann als eine Lookup-Tabelle angesehen werden, die es ermöglicht, die Grauwertsumme eines beliebig großen, rechteckigen Bildausschnittes, anhand von nur 4 Tabelleneinträgen zu berechnen. Durch diese effiziente Berechnung lässt sich von den einzelnen Pixelwerten eines Bildes abstrahieren und zur Beschreibung von Objektstrukturen simple Features, so genannte Haar-Like Features einsetzen. Siehe Abbildung 3.5 Seite 25.

AdaBoost Algorithmus

Eine weitere Kernkomponente des Viola-Jones Verfahrens ist der eingesetzte AdaBoost Lernalgorithmus. Dieser erstellt für die Detektion eines Objekts einen schwachen Classifier. Der

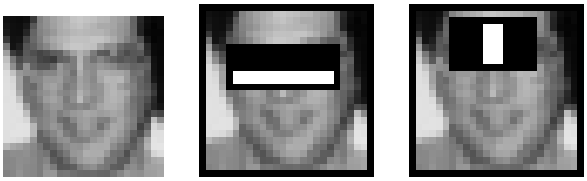


Abbildung 3.5.: 2 Haar-Like Features, welche besonders gut die Augen beschreiben. Die Berechnung der Haar-Like Features erfolgt durch Subtraktion der Grauwertsumme des weißen Bereichs, von der Grauwertsumme des schwarzen Bereichs. Bildquelle [VJ01].

Algorithmus Prüft aus einer großen Menge von möglichen Haar-Like Features, welche Features das gesuchte Objekt am besten beschreiben.

Kaskadierter Classifier

Durch Hintereinanderschaltung und Kaskadierung mehrerer schwacher Classifier ist ein effizienter Classifier generierbar, dessen Struktur als Entscheidungsbaum mit mehreren Stufen angesehen werden kann. Jede Stufe des Classifiers testet, wie groß die Ähnlichkeit eines Bildausschnittes mit einem Haar-Like Feature ist. Überschreitet die Ähnlichkeit einer Region einen gewissen Schwellwert, wird in einer weiteren Stufe die Ähnlichkeit mit einem anderen Haar-Like Feature untersucht. Unterschreitet die Region in einer Stufe jedoch einen gewissen Schwellwert, bricht das Testen ab und die Region wird verworfen. Wenn alle Stufen des Entscheidungsbaums erfolgreich durchlaufen sind, ist ein Objekt detektiert.

3.3. Sliding-Window

Um in einem Bild Landmarks mit dem Viola-Jones Classifier zu detektieren, durchläuft ein kleines Fenster (Sliding-Window) das Bild von links nach rechts und von oben nach unten. Siehe Abbildung 3.6 Seite 25. Dieser Prozess wiederholt sich solange, bis das gesamte Bild durchlaufen

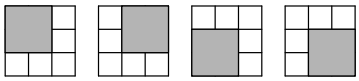


Abbildung 3.6.: Ein 2×2 Pixel großes Sliding-Window eines Classifiers, welches ein 3×3 Pixel großes Bild nach Haar-Like Features durchsucht.

ist. Da die Landmarks nur eine vergleichsweise ähnliche Struktur besitzen, diese sich jedoch

in ihrer Größe unterscheiden, wird nach jedem Durchlauf das Sliding-Window aufskaliert. Das Detektionsergebnis der Classifier sind die x-, y-Koordinaten der linken oberen Ecke des Sliding-Windows, sowie dessen Höhe und Breite. Die endgültige Position eines Landmarks, welche nur aus x- und y-Koordinaten besteht, ist der Mittelpunkt dieses Rechtecks.

3.4. Zuverlässige Landmarkdetektion in beschränkten Suchregionen

3.4.1. Nasendetektion zum Setzen der Suchregionen

Bei jedem Landmark das gesamte Bild zu durchlaufen, wie es z.B. Mian [Mia11] in seinem Ansatz handhabt, kostet unnötig Zeit. Idealerweise durchläuft dieses Verfahren nur ein einziges Mal das gesamte Bild, nämlich bei dem Detektieren der Nase. Die Nase hat einige besondere Eigenschaften, die es erlauben einen zuverlässigen Nasen-Classifier zu trainieren. Zum einen verändert sich ihre Form während der Mimik kaum und zum anderen wird die Nase nie unabsichtlich verdeckt. Eine Region, welche sich durch Mimik dagegen sehr stark verändert, ist der Mund. Typische Körperteile, die sehr oft durch lange Haare oder das Drehen des Kopfes verdeckt werden, sind z.B. die Ohren.

3.4.2. Suchregion der Alare

Wird eine Nase detektiert und ein Rechteck zurückgegeben, startet eine kaskadierte Suche innerhalb des detektierten Rechtecks nach den Nasenflügeln. Bei keiner detektierten Nase, sucht der Alare-Classifier in dem gesamten Bild nach den Nasenflügeln. Da für jedes Landmarkpaar jeweils nur der rechte Classifier trainiert ist, lassen sich durch horizontale Spiegelung des Bildes und erneutes Suchen die Positionen der linken Landmarks bestimmen.

3.4.3. Suchregion der Pupillen

Die Suchregion der Pupillen hat eine Höhe von 130% des Suchbereiches der Nasenflügel/der detektierten Nase und ist direkt über dieser in das xAbs-Gradient Bild positioniert. Die Breite der Suchregion entspricht der Bildbreite. Siehe Abbildung 3.7 Seite 27. Falls die Detektion der

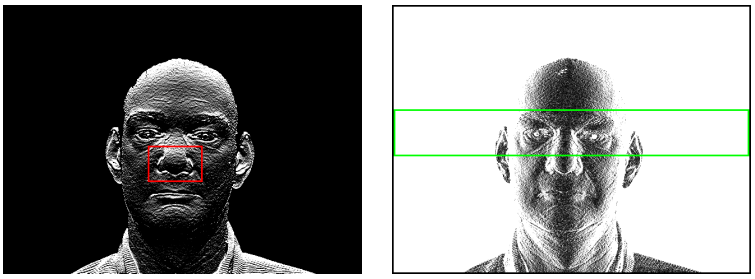


Abbildung 3.7.: Nach der Detektion der Nase (links), leitet sich die Suchregion der Pupillen ab (rechts).

Nase keinen Erfolg bringt, beläuft sich der Suchbereich der Pupillen auf das gesamte Bild. Bei der nachfolgenden Detektion der anderen Landmarks verkleinern sich der Suchbereiche ein weiteres mal. Die Beschreibung dieses Prozesses erfolgt nur für die rechten Landmarks, verläuft bei den linken jedoch identisch. Einziger Unterschied ist, dass die Zuweisung spiegelverkehrt von den hier beschriebenen Prozessen stattfindet.

3.4.4. Suchregion Endocanthion

Für die Endocanthion verläuft die Zuweisung der Suchregion wie folgt: Sind beide Pupillen detektiert, schrumpft der Suchbereich der Pupillen von der x-Koordinate der rechten Pupille bis zu der x-Koordinate des Mittelpunktes beider Pupillen. Falls nicht beide Pupillen, jedoch die rechte Pupille und der linke Nasenflügel detektiert sind, schrumpft die Suchregion zwischen diesen beiden Punkten. In dem Fall, dass die Koordinaten der rechten Pupille nicht, jedoch die des linken Nasenflügels vorhanden sind, erstreckt sich der Suchbereich neben dem linken Nasenflügel. Ist auch dieser nicht vorhanden, befindet sich der Suchbereich neben der linken Pupille. Bei Abwesenheit aller Koordinaten ist die Suchregion der gesamte Bereich, im dem zuvor nach den Pupillen gesucht wurde. Siehe Abbildung 3.8 Seite 28.

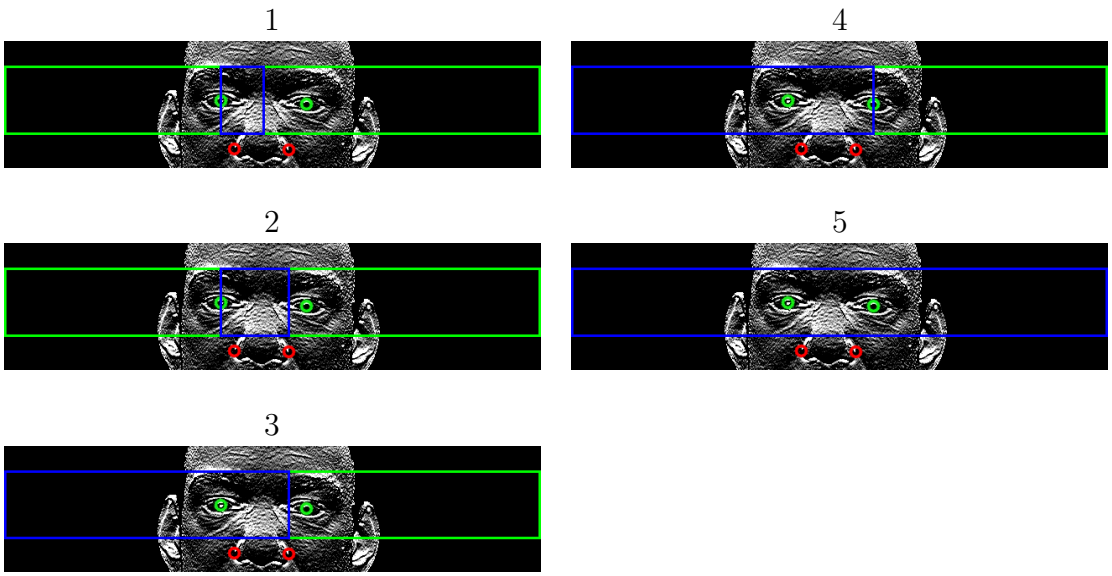


Abbildung 3.8.: Erstellen des Suchbereichs der rechten Endocanthion. Von der rechten Pupille bis zum Mittelpunkt der Pupillen, von der rechten Pupille bis zur linken Alare, von der linken Alare bis zum Bildrand, von der linken Pupille bis zum Bildrand und die gesamte Suchregion der Pupillen.

3.4.5. Suchregion Exocanthion

Zur Detektion der Exocanthions ist als Ausgangspunkt der Suchbereich der Pupillen definiert, welcher sich je nach zuvor gefunden Landmarks verkleinert. Falls die rechte Pupille detektiert ist, liegt die Suchregion der rechten Exocanthion neben der rechten Pupille. Wenn die rechte Pupille nicht detektiert ist, aber die rechte Alare, liegt der Suchbereich links neben dieser. Ist auch die rechte Alare nicht detektiert, jedoch die linke Alare, sucht der Classifier in dem Bereich neben der linken Alare. Falls nur die linke Pupille detektiert ist, liegt die Suchregion neben der linken Pupille. Konnte keine Pupille und kein Nasenflügel detektiert werden, ist der Suchbereich der gleiche wie für die Pupillen. Siehe Abbildung 3.9 Seite 29.

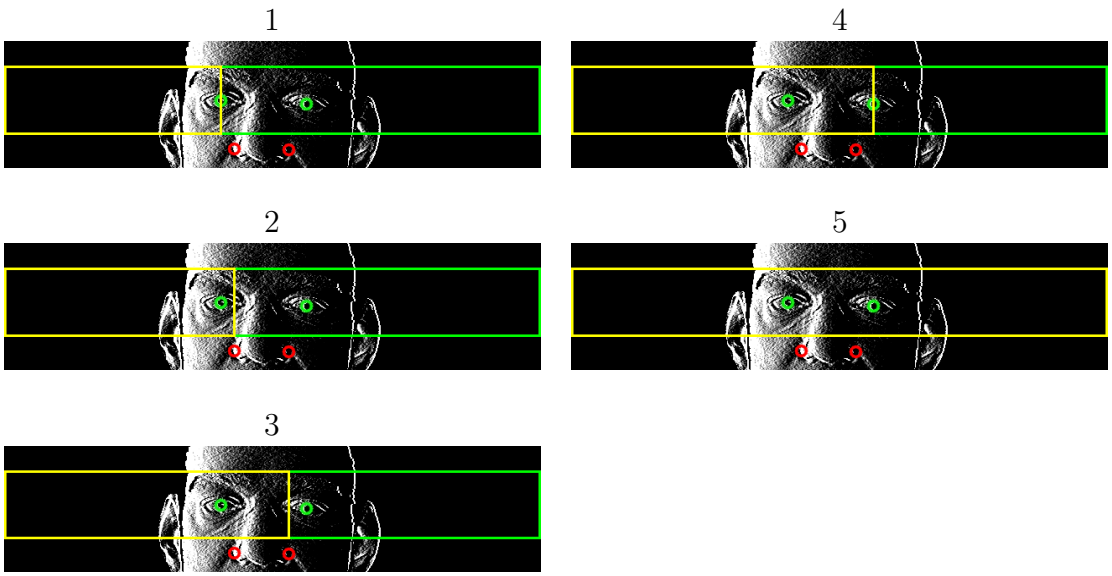


Abbildung 3.9.: Erstellen des Suchbereichs der rechten Exocanthion. Von der rechten Pupille bis zum Bildrand, von der rechten Alare bis zum Bildrand, von der linken Alare bis zum Bildrand, von der linken Pupille bis zum Bildrand und die gesamte Suchregion der Pupillen.

3.4.6. Suchregion Cheilion

Die letzten beiden Landmarks deren Suchregionen gesetzt werden, sind die rechte und linke Cheilion. Für diese ist vorläufig ein allgemeiner Suchbereich definiert, der direkt unterhalb der Detektierten Nase positioniert ist und auch deren Höhe übernimmt. Die Breite dieses Suchbereichs ist ebenfalls die komplette Bildbreite. Siehe Abbildung 3.10 Seite 30. Ausgehend von dem allgemeinen Suchbereich findet das Setzen des finalen statt. Der Suchbereich der rechten Cheilion beginnt immer bei der rechten Exocanthion. Falls die rechte Exocanthion nicht detektiert ist, beginnt der Suchbereich bei dem linken Bildrand. Sind beide Nasenflügel detektiert, liegt das Ende der Suchregion zwischen der rechten Alare und dem Mittelpunkt beider Nasenflügel. Wenn die Nasenflügel nicht detektiert sind, jedoch die beiden Pupillen, endet die Suchregion auf $\frac{3}{4}$ der Strecke zwischen der rechten Pupille und dem Mittelpunkt der Pupillen. Waren nicht beide Nasenflügel oder Pupillen detektierbar, ist die Suchregion zwischen der rechten Exocanthion und der linken Alare. Ist auch diese nicht detektiert, liegt der Suchbereich zwischen der rechten Exocanthion und der linken Pupille. In dem Fall das keiner der gerade aufgeführten Landmarks gefunden ist, bleibt der allgemeine Suchbereich, als Suchbereich der Cheilions bestehen. Siehe Abbildung 3.11 Seite 30.

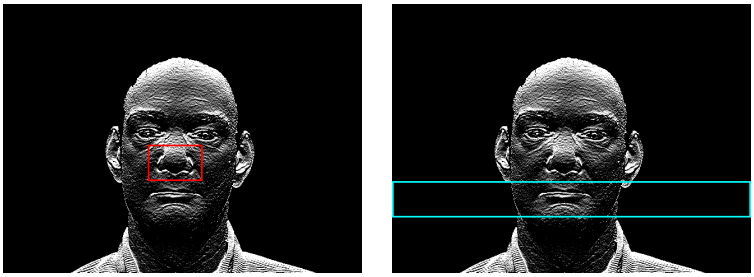


Abbildung 3.10.: Nach der Detektion der Nase (links), leitet sich der allgemeine Suchbereich der Cheilions ab (rechts).

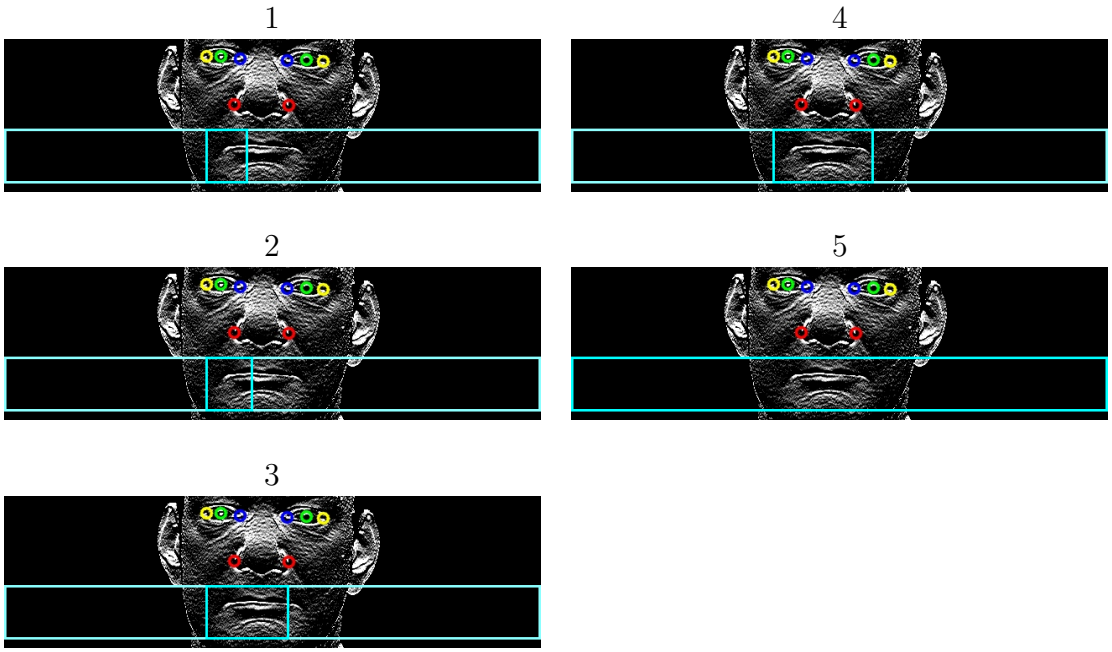


Abbildung 3.11.: Erstellen des Suchbereichs der rechten Cheilion. Von der rechten Exocanthion bis zu dem Punkt zwischen der rechten Alare und dem Mittelpunkt der Nasenflügel, von der rechten Exocanthion bis zu $\frac{3}{4}$ der Strecke zwischen rechter Pupille und dem Mittelpunkt der Pupillen, von der rechten Exocanthion bis zur linken Alare, von der rechten Exocanthion bis zur linken Pupille und der allgemeine Suchbereich der Mundwinkel.

3.5. Filterung falsch detektierter Landmarks

Findet ein Classifier bei der Detektion ein Landmark, wird dessen Position in einem Vektor zwischengespeichert. Das Sliding-Window des Classifiers durchläuft dabei mehrmals in verschiedenen Skalierungen das Bild und füllt somit den Vektor mit Koordinaten. Da die trainierten Classifier immer eine Vielzahl von Punkten zurückgeben, die nicht das gesuchte Objekt darstellen, kommt bei der Landmarkdetektion die Treshold-Funktion zum Einsatz. Die Treshold-Funktion erzeugt um den Mittelpunkt eines Landmarks eine kleine 7×7 Pixel große Matrix und überprüft, ob der durchschnittliche Grauwert aller Pixel innerhalb der kleinen Matrix größer als ein bestimmter Grenzwertes ist. Liegt der Durchschnitt der Matrix oberhalb des Grenzwertes, wird die Position des Landmarks gespeichert. Ist der Durchschnitt kleiner oder gleich dem Grenzwert, verwirft die Funktion den Punkt. Der Grenzwert der Pupillen ist 190. Siehe Abbildung 3.12 Seite 31. Der Grenzwert aller anderen Landmarks ist 0, da die Funktion bei ihnen nur Prüft, ob der Landmark (Mittelpunkt des Sliding-Windows) nicht im Hintergrund liegt, welcher komplett schwarz ist und einen Durchschnittswert von 0 zur Folge hat. Sind mehr als 2 Punkte bei der Suche gefunden, teilt das Programm diese in 2 Cluster

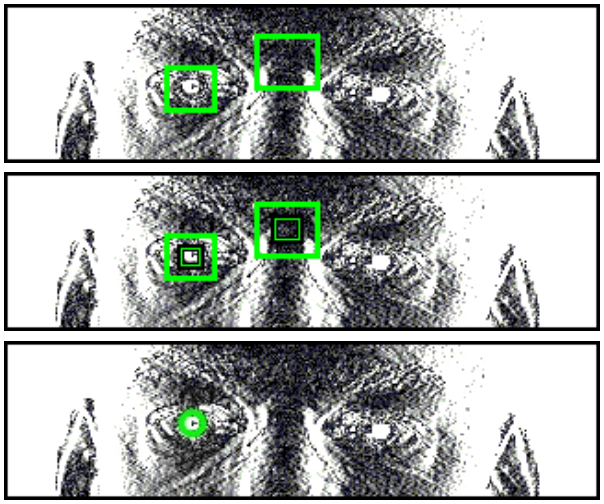


Abbildung 3.12.: Funktionsweise der Treshold-Funktion. Auf dem oberen Bild sind 2 Landmarks zu sehen, die von dem Classifier detektiert wurden. In dem mittleren Bild wird der Mittelpunkt der Landmarks gesucht und um den Mittelpunkt eine kleine 7×7 Pixel große Matrix gelegt. Übersteigt der Mittelwert der kleinen Matrizen einen Treshold von 190, wird die Position des Landmarks gespeichert (unteres Bild).

auf und setzt die finalen Positionen der Landmarks. Ist nur ein Punkt detektiert, wird dieser dem linken oder rechten Landmark zugewiesen.

3.6. Clustering

Ein weiterer wichtiger Grund den Suchbereich auf diese Weise einzuschränken, liegt neben der geringeren Ausführungszeit des Programms darin, dass sich manche Landmarks zu sehr ähneln und es dadurch zu vielen falschen Detektionen kommt. So weisen z.B. die rechte Endocanthion und die linke Exocanthion eine zu ähnliche Struktur auf, als dass die trainierten Classifier diese zuverlässig unterscheiden können. Zu viele falsch detektierte Landmarks sind bei dem Clustering ein Problem, da es nicht mehr nur einzelne Ausläufer gibt, sondern eine ganze Reihe falsch detektierter Landmarks, welche so die Position des finalen Clusters verschieben. Das Clustering wird mit der bereits in OpenCV implementierten k-means-Funktion durchgeführt, wobei der k-means++ Algorithmus die Initialkoordinaten der Clusterzentren (Zentroiden) bestimmt. Das Hauptziel des k-means++ Algorithmus besteht darin, die absolute Distanz der einzelnen Punkte zu ihren jeweiligen, nächst gelegenen Zentroiden zu minimieren. Dieses ist durch

$$\phi = \sum_{x \in X} \underset{c \in C}{min} ||x - c||^2$$

ausgedrückt, wobei X die Menge aller Punkte in $\mathbb{R}^n$ und x ein Punkt der Menge X ist. C ist die Menge der Zentroiden in $\mathbb{R}^n$ und c ein Punkt aus C. Den logischen Ablauf des iterationsbasierten Clusteringverfahrens erläutert das folgende Beispiel. Angenommen es sollen 5 Punkte, $p_1 = (1,1),\ p_2 = (1,2),\ p_3 = (2,1),\ p_4 = (3,4)$ und $p_5 = (5,4)$ in 2 Cluster unterteilt werden, so wird im ersten Schritt zufällig ein Initialpunkt gewählt, welcher das Zentrum des ersten Clusters bildet. Zufällig wäre das $p_3 = c_1 = (2,1)$, siehe dazu Abbildung 3.13 Seite 33. Im zweiten Schritt wählt der Algorithmus mit der Wahrscheinlichkeit

$$P(p_x) = \frac{D(p'_x)^2}{\sum_{x \in X} D(p_x)^2}$$

den nächsten zentralen Punkt. Wobei $D(p'_x)^2$ die Distanz des zu untersuchenden Punktes zu seinem nächst gelegenen Zentroiden (hier c_1) und $\sum_{x \in X} D(p_x)^2$ die Summe der Distanzen aller Punkte zu ihren jeweiligen nächst gelegenen Zentroiden darstellt. $D(p_1)$ ist die Distanz von p_1 zu $c_1 = 1$. Analog ist $D(p_2)$ die Distanz von p_2 zu $c_1 = 1,414$, $D(p_4) = 3,162$ und $D(p_5) = 4,243$. Somit ist $\sum_{x \in X} D(p_x)^2 = 1^2 + 1,414^2 + 3,162^2 + 4,243^2 \approx 31$. Die Wahrscheinlichkeit, dass p_1 zum nächsten Zentrum des zweiten Clusters wird ist $P(p_1) = \frac{1^2}{31}$. Die Wahrscheinlichkeit, dass p_2 zum Zentrum des nächsten Clusters wird ist $P(p_2) = \frac{1,414^2}{31}$ usw. Da der Algorithmus den Punkt mit der größten Wahrscheinlichkeit zum Zentrum des nächsten Clusters wählt und diese bei p_5 mit $P(p_5) = \frac{18}{31}$ am größten ist, wird p_5 zu c_2. Es sind die ersten 2 Zentroiden ausgewählt. Die nachfolgenden Schritte wiederholen sich in einer Schleife so lange, bis entweder 10 Iterationen erreicht sind, oder bis sich die x-, y-Koordinaten der Zentroiden

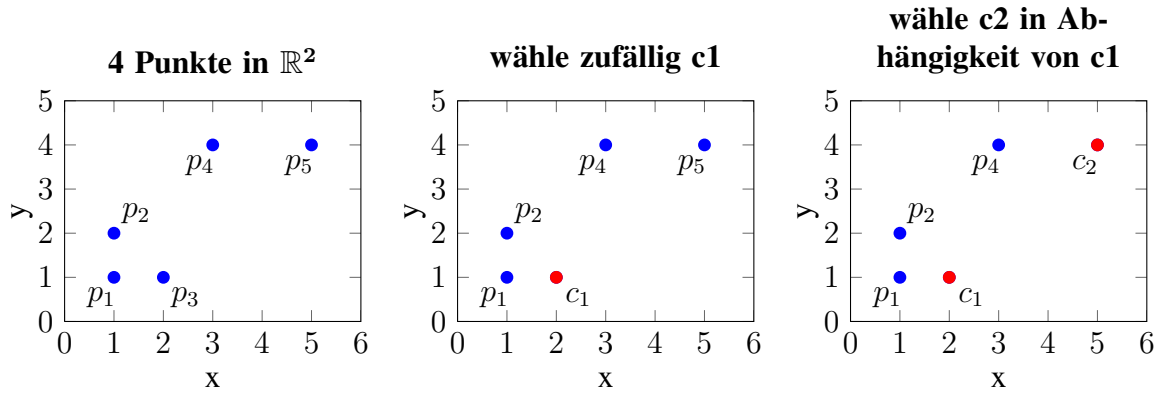


Abbildung 3.13.: Die 1. Iteration des k-means++ Algorithmus. Der Algorithmus bestimmt zufällig den ersten Centroid aus 5 Punkten. Abhängig von den Distanzen der restlichen Punkte zum ersten Centroid wird der zweite Centroid gewählt. Die Wahl fällt auf den Punkt $p5$ welcher am weitesten von c1 entfernt ist.

um weniger als 1 Pixel verschieben. Schritt 1: Errechne von jedem Punkt die Distanz zu den beiden Zentroiden. Die Spalten der Matrix stehen dabei für die Punkte. Die Zeilen stehen für die Distanz von dem Punkt bis zum Zentroiden.

$$D^0 = \begin{array}{c} \begin{array}{ccccc} p_1 & p_2 & p_3 & p_4 & p_5 \end{array} \\ \begin{pmatrix} 1 & 1,414 & 0 & 3,162 & 4,243 \\ 5 & 4,472 & 4,243 & 2 & 0 \end{pmatrix} \end{array} \begin{array}{l} \\ \text{Distanz zu } c_1 \\ \text{Distanz zu } c_2 \end{array}$$

Schritt 2: Jeder Punkt wird demjenigen Zentroiden/Cluster zugewiesen, zu dem er am nächsten liegt. Die 5 Punkte werden wie folgt zugewiesen, dabei steht eine 1 für zugewiesen und eine 0 für nicht zugewiesen.

$$G^0 = \begin{array}{c} \begin{array}{ccccc} p_1 & p_2 & p_3 & p_4 & p_5 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array} \begin{array}{l} \\ c_1 \\ c_2 \end{array}$$

Die Punkte p_1, p_2, und p_3 gehören zu Cluster 1 wobei die Punkte p_4 und p_5 zu Cluster 2 gehören. Der letzte Schritt der Iteration besteht darin, den neuen Mittelpunkt dieser beiden Cluster zu bestimmen. Der Mittelpunkt ist der Durchschnitt aller Punkt-Koordinaten des jeweiligen Clusters. Zentroid c_1 lässt sich durch die Koordinaten $c_1 = (\frac{p_1x+p_2x+p_3x}{3}, \frac{p_1y+p_2y+p_3y}{3}) = (\frac{1+1+2}{3}, \frac{1+2+1}{3}) = (1,33; 1,33)$ und Zentroid c_2 durch $c_2 = (\frac{3+5}{2}, \frac{4+4}{2}) = (4; 4)$ bestimmen. Die nachfolgende Iteration berechnet wieder die Distanz der einzelnen Punkte $p_1 - p_5$ zu c_1 und c_2, weist die Punkte den Clustern zu und berechnet die Zentroiden erneut. Durch das vorherige Eingrenzen des Suchbereichs plus dem anschließenden Clustering erreicht das Programm eine geringe Ausführungszeit und eine zusätzlich hohe Detektionsgenauigkeit. Wie gut die Performance genau ist beleuchtet das Kapitel Testergebnisse und Evaluierung.

3.7. Archivierung der Ergebnisse

Nach der Detektion aller Landmarks zeichnet das Programm diese zur visuellen Darstellung in den y-Gradienten des Bildes ein und speichert sie in einem eigenen Verzeichnis ab. Abbildung 3.14 S. 34 zeigt ein sehr gutes Detektionsergebnis bei dem alle Landmarks zuverlässig gefunden wurden und ein sehr schlechtes, bei dem die linke Pupille, sowie die beiden Exocanthions durch die zugekniffenen Augen nicht detektierbar sind. Die linke Cheilion ist zudem zu ungenau detektiert, da sich durch die Mimik der linke Mundwinkel zu stark von den Trainingsdaten unterscheidet. Die Abbildung macht sehr gut deutlich, wo die Grenzen dieses Verfahrens liegen.

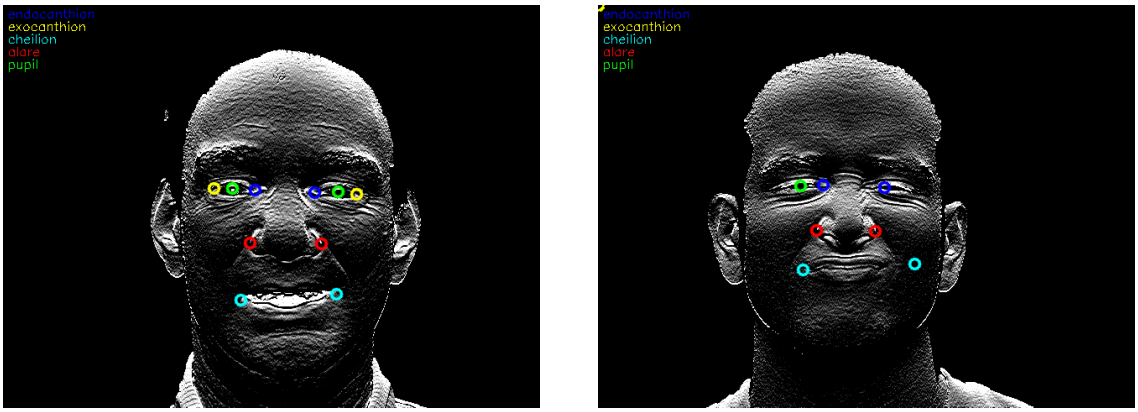


Abbildung 3.14.: Vergleich eines guten und schlechten Detektionsergebnisses. Auf der linken Seite ein sehr gutes Detektionsergebnis, alle Landmarks wurden präzise gefunden. Auf der rechten Seite ein schlechtes Ergebnis. Durch die zugekniffenen Augen fehlt die linke Pupille sowie die beiden Exocanthions. Die starke Mimik im Mundbereich lässt nur eine ungenaue Detektion der linken Cheilion zu.

Da der Grundgedanke in der Similarität einzelner Landmarks begründet ist, fallen durch die Mimik stark verzerrte, oder durch eine spezielle Pose nicht sichtbare Punkte durch das Detektionsraster. Es ist daher nahezu unmöglich einen soliden Classifier zu trainieren, der mit allen Hindernissen dieser Art zurecht kommt. Um mit den detektierten Landmarks weiter arbeiten zu können, was nicht Bestandteil dieser Arbeit ist, wird der gesamten Datensatz als xml-Datei abgespeichert, welche der Struktur eines GroundTruth Editor Datensatzes nachempfunden ist. Die Tags der einzelnen Landmarks sind dabei die „abbreviation“, das „masking“, der Name des Landmarks, der Radius sowie dessen x- und y-Koordinaten. Die „abbreviation“ ist eine Abkürzung des Landmarks. Der Verdeckungsgrad eines Landmarks wird durch „masking“ beschrieben und kann die Werte „None“, „Minor“, „Major“ und „Full“ annehmen. Siehe Anhang A Userinterface des GroundTruth Editor und Anhang B Struktur eines GroundTruth Editor Datensatzes.

4. Trainingsverfahren

In diesem Kapitel geht es um den Trainingsprozess der Classifier, welcher die Classifier schult, Landmarks von anderen Bildinhalten zu unterscheiden. Das Kapitel beschreibt, was alles benötigt wird um die eingesetzten Classifier zu generieren und wie lange das Training dauert, bis die gewünschte Detektionsgenauigkeit erreicht ist. Dabei lässt sich das Training in zwei Abschnitte unterteilen. Der erste Abschnitt befasst sich mit dem Bereitstellen der für das Training benötigten Bildinformationen und der zweite mit dem eigentlichen Training.

4.1. Generierung der Trainingsdaten

Um ein Classifiertraining zu starten, muss dieses zuvor auf Spezielles Bildmaterial Zugang haben, dass sich in positive- und negative-Bilder unterteilen lässt. Positive Bilder enthalten nur das gesuchte Objekt, während der Inhalt negativer Bilder frei gewählt werden kann und nur der Einschränkung unterliegt, dass in diesem nicht das gesuchte Objekt vorhanden ist.

4.1.1. GroundTruth Editor

Für das Erstellen positiver Bilder müssen alle Landmarks aus den unterschiedlichen Gradient-Images ausgeschnitten werden. Da das manuelle Ausschneiden sehr viel Zeit in Anspruch nimmt und für die Performance-Analyse Referenzdaten vorhanden sein müssen, war der GroundTruth Editor bei der Arbeit behilflich. Der GroundTruth Editor ist ein von der Forschungseinrichtung CASED zur Verfügung gestelltes Java-Programm, welches das manuelle Setzen einzelner Landmarks in ein Bild ermöglicht. Durch Ziehen eines Kreises um ein Landmark erfolgt die Speicherung seiner Position in einer xml-Datei. Die Anzahl der markierten Datensätze aus der FRGC-Datenbank beläuft sich auf die ersten 500 Stück.

4.1.2. Positive- und negative Images

Anhand der gespeicherten Positionen verschiedener Landmarks in den xml-Dateien wurde die Größe und Position der positiven Images errechnet. Die positiven Bilder für den Nasendetektor wurden aus den y-Gradienten ausgeschnitten, da der Classifier in diesen nach der Nase sucht. Sie haben eine Höhe von 90% der Distanz zwischen Endocanthion und Subnasale (Nasenansatz kurz oberhalb der Oberlippe). Ihre Höhe beträgt 150% ihrer Breite. Die Zentrierung der Bilder in x-Richtung richtet sich nach der Nasenspitze, die als Mittelpunkt zwischen linker und rechter Alare definiert ist. Das Positionieren der Unterkante zwischen Nasenansatz und Operlippe legt das Fenster genau über die Nase. Durch zufälliges Setzen der Grauwerte aller Pixel in der ausgeschnittenen Region ließen sich direkt negative Images erzeugen. Siehe Abbildung 4.1 Seite 36 welche ein negatives und ein positives Image abbildet. Diese Methode fand auch bei der Generierung negativer Images für das Training der anderen Classifier Verwendung. Zu den

Abbildung 4.1.: negatives und positives Image für das Training des Nasen-Classifiers. Erzeugung des negativen Images (links) durch zufälliges Setzen der Grauwerte jedes Pixels in der ausgeschnittenen Region. Der rote Rahmen ist in dem negativen Image beim Trainieren des Classifiers nicht vorhanden und dient hier nur der Illustration.

bereits 500 erstellten negativen Bildern kamen weitere 1500 Grauwertbilder unterschiedlichster Herkunft aus dem Internet. Um die Anzahl positiver Bilder für das Training zu erhöhen, wurden die ausgeschnittenen Nasen gespiegelt. Bei dem Training des Nasendetektors kamen so insgesamt 1000 positive und 2000 negative Images zum Einsatz. Bei dem Classifiertraining der Endocanthion, der Alare, der Cheilion und der Pupille ist ein Verhältnis von 1:4 von positiven zu negativen Images gewählt. Das Training des Exocanthion-Classifier hat ein Verhältnis von 1:6 von positiven zu negativen Images. Die Trainingsvorbereitung dieser Classifier gestaltete sich etwas anders als das Training des Nasen-Classifiers. Da die in dieser Arbeit detektierten Landmarks alle 2 mal im Gesicht vorkommen, wurde jeweils das linke positive Image gespiegelt. Der Grund dafür ist, dass sich dadurch beim Training die Symmetrie und Ausrichtung der ausgeschnittenen Landmarks angleicht und dies eine bessere Trainingsgrundlage bereitstellt. Die jeweiligen Landmarks entsprechen genau dem Mittelpunkt der Bildausschnitte aus den unter-

schiedlichen Gradient-Bildern. Die Bildausschnitte der linken und rechten Endocanthion, der linken und rechten Alare, sowie der linken und rechten Cheilion sind aus den y-Gradient Images entnommen worden. Die Abmessungen der positiven Images der Endocanthions entspricht der Distanz zwischen Endocanthion und Exocanthion mal 0,8. Abbildung 4.1 Seite 38 zeigt einen Überblick der ausgeschnittenen Landmarks. Die Abmessungen der positiven Images der Alare sind durchweg 13×25 Pixel groß. Die Höhe und Breite positiver Images der Cheilions ist die Hälfte des Abstandes zwischen Cheilion und der Labiale Superius (Mittelpunkt der Oberlippe). Um die positiven Images der Pupille zu erstellen, sind Teile aus den xAbs-Gradienten ausgeschnitten. Breite und Höhe entsprechen 0,8 mal den Abstand zwischen Endocanthion und Exocanthion. Die Differenz von Pupille zu Exocanthion mal 130% ergibt die Höhe und Breite der positiv Images der Exocanthions. Für die rechte Exocanthion wurde aus den x-Gradient Images und für die linke Exocanthion aus den gespiegelten x-Gradient Images ausgeschnitten.

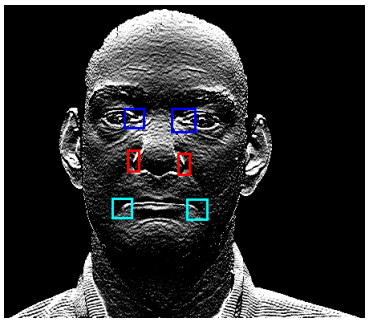
4.1.3. Komprimieren positiver und negativer Images

OpenCV stellt für das Training der Classifier ein eigenes Tool namens „haartraining“ bereit. Das Tool erwartet als weiteren Eingabeparameter neben den positiven und negativen Images eine Datei im speziellen Dateiformat „.vec“. Die Datei enthält alle positiven Images auf eine einheitliche Größe skaliert. Um das .vec-File zu erzeugen wird eine Beschreibung aller positiven Images benötigt, die den Speicherort sowie deren Abmessungen enthält. Dem OpenCV Tool „createsamples“ wird die Beschreibung und die Größe, auf dem die positiven Images skaliert werden sollen, übergeben. Daraufhin erstellt „createsamples“ das .vec-File. Die Skalierungsgröße der positiven Images für den Nasen-Classifier beträgt 36×24 Pixel und ist in der Ratio gleich der ausgeschnittenen Nasen. Diese Ratio beizubehalten ist wichtig, da sonst die Images bei der Skalierung verzerren. Per Kommandozeile kann das haartraining-Tool gestartet werden. Ein anderer Parameter ist neben dem vec-File die Beschreibungsdatei der negativen Images. Die Beschreibungsdatei der negativen Images enthält den Speicherort der Dateien.

4.2. Parameter des Trainingsprozesses

Weitere Parameter für das Haartraining sind neben dem .vec-Files: $nstages$, $nsplits$, $minhitrate$, $maxfalsealarm$, $npos$, $nneg$, w, h, $sym/\ nonsym$, mem, $mode$. Die maximale Anzahl der Stufen eines Classifiers gibt $nstages$ an. $nsplits$ beschreibt deren interne Verzweigung. Sobald

Übersicht der positiven Images des ersten Datensatzes für das Haartraining

y-Gradient Image			xAbs-Gradient Image
			
Endocanthion rechts, links	Alare rechts, links	Cheilion rechts, links	Pupille rechts, links

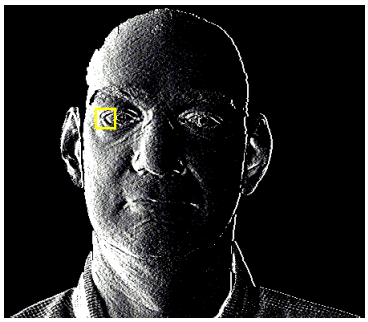
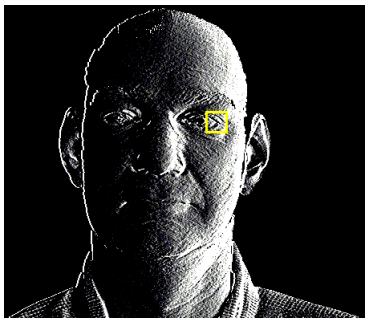
x-Gradient Image	xMirrored-Gradient Image
	
Exocanthion rechts	Exocanthion links

Tabelle 4.1.: Übersicht der positiven Bilder die für das Training der Classifier benutzt wurden. Hier die positiven Bilder des ersten Datensatzes von 500 verwendeten. Zu beachten ist, dass die ausgeschnittenen linken Landmarks für das Training gespiegelt werden um so die Symmetrie und Ausrichtung der Landmarks anzugleichen. Dies bietet dem Classifier eine bessere Trainingsgrundlage.

nsplits einen wert größer 2 erhält, wird jede Stufe intern wieder als Baumstruktur aufgebaut. Der einfachste Classifier mit $nsplits = 1$ besteht in jeder Stufe aus einer Entscheidung: Ist die Ähnlichkeit der Region über dem sich das Sliding-Window befindet mit einem Haar-Like Feature zu gering, wird an dieser Stelle abgebrochen. Überschreitet die Ähnlichkeit der Region einen gewissen Schwellwert, wird zur nächsten Stufe übergegangen. Das Ausloten der Schwellwerte und das Auswählen der Haar-Like Features in den einzelnen Stufen ist die Hauptaufgabe des Trainings. Der maximale Wert von falschen Detektionen, also dass eine Region beim Testen eine Stufe passiert, obwohl sie das gesuchte Objekt nicht enthält, gibt $maxfalsealarm$ an. Die Anzahl positiver Bilder und negativer Bilder die beim Training verwendet werden sollen, gibt *npos* bzw. *nneg* an. Die Parameter w und h stehen für die Breite und Höhe der positiven Bilder im .vec-File und müssen sich mit der gewählten Skalierungsgröße bei der .vec-File Generierung decken. Durch *sym* oder *nonsym* werden die Classifier auf vertikal symmetrische oder vertikal asymmetrische Objekte trainiert. Das Training auf vertikal Symmetrische Objekte verbraucht weniger RAM und weniger Rechenzeit. *mem* reserviert für das Training den Speicherbereich und *mode* weist dem Training die Art der verwendeten Haar-Like Features zu. Es existieren 3 verschiedene Modi. BASIC, CORE und ALL. Siehe dazu Abbildung 4.2 Seite 39. Die *overall hitrate* des Classifiers berechnet sich durch

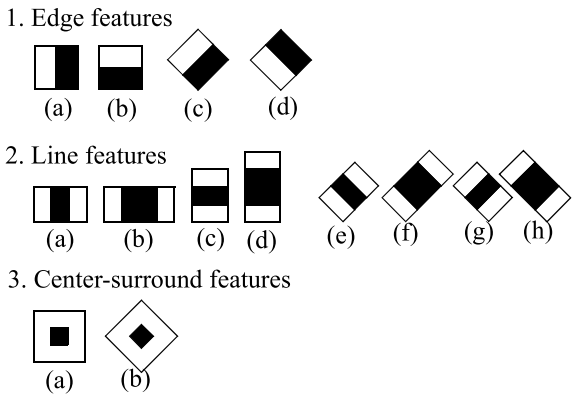


Abbildung 4.2.: Übersicht der in OpenCV verwendeten Haar-Like Features. Modus CORE verwendet die Features 1a, 1b, 2a, 2c. Modus BASIC verwendet CORE + 2b, 2d und 3a. Im Modus ALL werden alle Features genutzt. Die Classifier in dieser Arbeit sind mit dem Modus ALL trainiert. Bildquelle [LM02].

$overall\ hitrate = minhitrate^{nstages}$. Im Gegensatz dazu steht die $overall\ maxfalsealarmrate$, welche als $overall\ maxfalsealarmrate = maxfalsealarm^{nstages}$ berechenbar ist. Das Training eines Classifiers beendet automatisch, sobald die Kaskade den Wert der *overall hitrate* überschreitet und den der $overall\ maxfalsealarmrate$ unterschreitet. Die *overall hitrate* und $overall\ maxfalsealarmrate$ stehen dabei in einem Verhältnis zueinander. So hat ein Classifier mit einer hohen *overall hitrate* auch automatisch eine hohe $overall\ maxfalsealarmrate$. Es

kann kein Classifier trainiert werden, der jedes Feature korrekt erkennt und dabei nicht einen einzigen Fehler begeht. Mit einem Fehler ist in diesem Sinne das Passieren eines nicht gesuchten Features durch alle Stufen des Classifiers gemeint. Analog dazu kann ein Classifier mit einer niedrigen false positive rate nie eine hohe true positive rate erreichen. Das Training des Nasen-Classfiers ist mit 14 Stufen, 2 Splits, einer $minhitrate$ von 0.999, einer $maxfalsealarmrate$ von 0,5 und als $nonsym$ vollzogen worden. Dem Training waren 1024 MB RAM zugewiesen und alle Haar-Features als Modus gewählt. Das System, welches das Training vollzog, besteht aus einem Intel Core 2 Duo E8400 @ 3.00GHz mit 4 GB RAM und dem Betriebssystem Windows 7 in der 64 bit Version. Insgesamt dauerte das Training des Nasen-Classifiers auf diesem System 4,04 Stunden und brach nach Stufe 7 ab, da die $overall\ maxfalsealarmrate$ erreicht war (unterschritten). Für jedes Training der Landmarks sind die folgenden Parameter benutzt: $nstages = 14$, $nsplits = 2$, $minhitrate = 0,999$, $maxfalsealarm = 0,5$, $sym/nonsym = nonsym$, $mem = 1024$ und $mode = ALL$. Die abweichenden Parameter, nach wie vielen Stufen das Training endete, da die $overall\ maxfalsealarmrate$ erreicht war und die Trainingsdauer der einzelnen Classifier listet Tabelle 4.2 Seite 40 auf.

Classifier	npos	nneg	w × h	Verhältnis pos zu neg	Stufen des Classifiers	Dauer des Trainings (in h)
Nase	1000	2000	36 × 24	1:2	7	4,04
Endocanthion	952	3808	18 × 18	1:4	9	1,86
Alare	944	3776	13 × 25	1:4	5	1,39
Cheilion	892	3568	14 × 14	1:4	8	0,71
Pupil	964	3856	24 × 24	1:4	7	3,78
Exocanthion	762	4572	14 × 14	1:6	9	1,04

Tabelle 4.2.: Parameter des Haartrainings der einzelnen Classifier. Die Spalte „w × h“ ist deckungsgleich mit der Skalierungsgröße der positiven Images bei der .vec-File Erzeugung. Durch zugekniffene Augen, einer schlechten Pose oder aufgrund der Mimik werden manche Landmarks verdeckt/ verzerrt. So sind nur bei dem Training des Nasen-Classifiers genau 1000 positive Images verfügbar. Die Stufen des Classifiers sind mit dem Abbruch des Trainings, durch erreichen der $overall\ maxfalsealarmrate$, gleichzusetzen.

5. Testergebnisse und Evaluierung

Das Kapitel Testergebnisse und Evaluierung befasst sich mit der Programmlaufzeit und der Präzision, mit der die Landmarks detektierbar sind. Es stellt einen Vergleich her zwischen dem Detektieren der Landmarks mit vorherigem Setzen der Suchregionen, oder dem Detektieren der Landmarks auf dem gesamten Bild.

5.1. Laufzeit

Die Untersuchung der Programmlaufzeit ist ein wichtiger Teil der Evaluierung, da sie die Verwendbarkeit eines Systems für verschiedene Szenarien ausleuchtet. Man stelle sich z.B. ein Verfahren vor, dass an einem Flughafen oder einer Grenze zum Einsatz kommen soll. In einer solchen Umgebung herrscht eine hohe Durchgangsrate. Daher ist es wichtig, die Programmlaufzeit so gering wie möglich zu halten, da sonst der reibungslose Betrieb gestört wird. Um aussagekräftige Ergebnisse zu erhalten, stellen die nachfolgenden Zeitmessungen einen Durchschnittswert von 100 Programmdurchläufen dar. Das Testsystem mit dem diese Zeiten gemessen wurden, besteht aus einer Intel Core 2 Duo CPU @ 2.16 GHz, 4 GB RAM und dem Betriebssystem Windows 7 in der 64 bit Version.

Begrenzen des Suchbereichs vs freie Suche

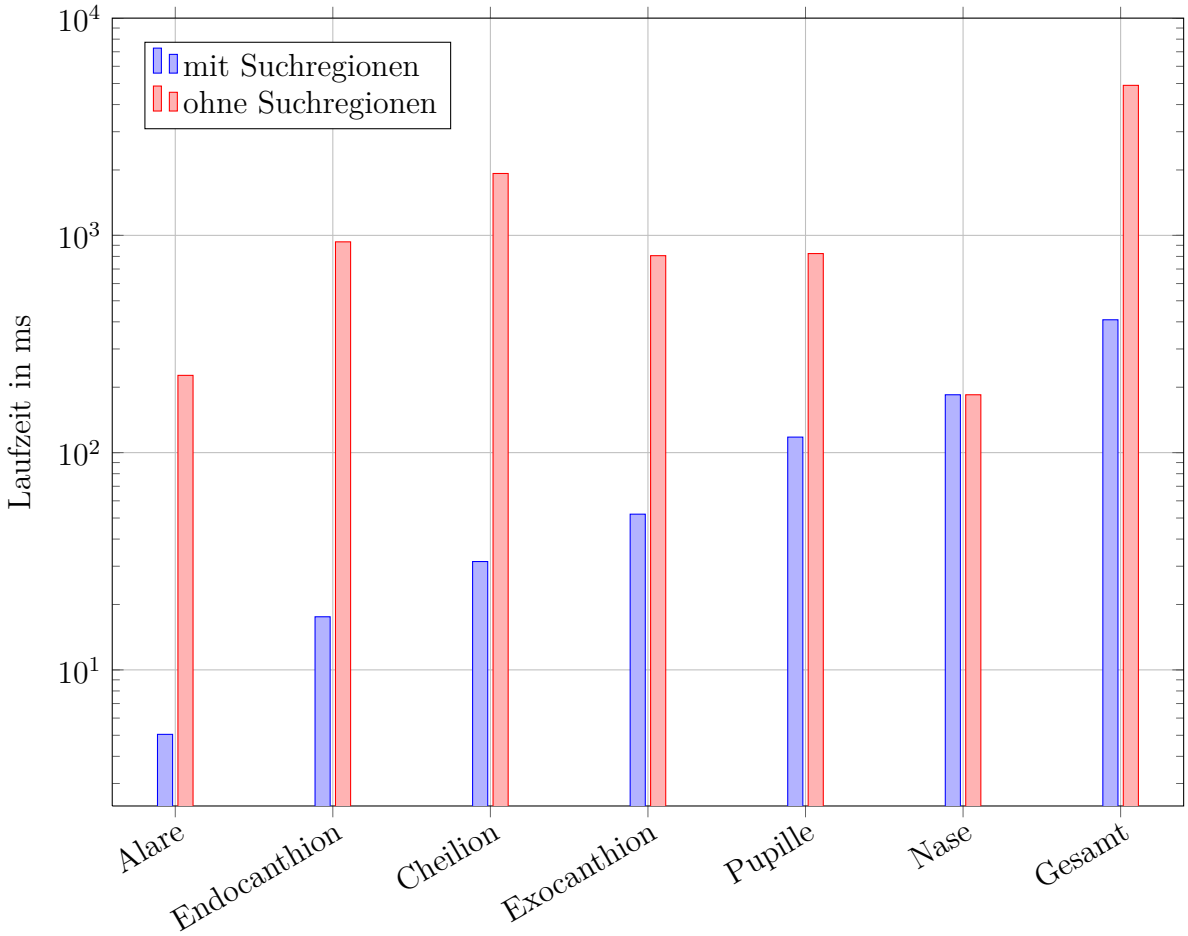
Bei der Gegenüberstellung mit dem Setzen der Suchregionen und ohne dem Setzen der Suchregionen sind bei beiden Vorgehensweisen die Skalierungen des Sliding-Windows, dessen Startgröße und die maximale Skalierungsgröße gleich. Siehe Tabelle 5.1 Seite 42. Die Laufzeitmessungen stellt Abbildung 5.1 Seite 43 dar. Auffällig ist bei den Laufzeitmessungen, dass bei der entwickelten Vorgehensweise mit dem Setzen der Suchregionen, der Classifier der Alare seine Detektion am schnellsten beendet, obwohl dieser im Durchschnitt einen größeren Bereich untersuchen muss. Der Grund ist die Startgröße des Sliding-Windows von 13×25 und dessen

Classifier	Sliding-Window Skalierungs-faktor in %	min Sliding-Window w × h	max Sliding-Window w × h	Anz. der Skalierungen bis max erreicht
Nase	110	57 × 38	141 × 94	10
Pupillen	110	20 × 20	60 × 60	10
Exocanthion	110	12 × 12	40 × 40	12
Cheilion	105	14 × 14	40 × 40	22
Endocanthion	110	16 × 16	50 × 50	11
Alare	105	13 × 25	15 × 27	2

Tabelle 5.1.: Skalierung des Sliding-Windows der einzelnen Classifier. Der Skalierungsfaktor gibt an, wie sich das Sliding-Window in seinen Abmessungen verändert nachdem es einen Suchbereich durchlaufen hat. 105% bedeutet, dass es sich um 5% vergrößert. Erreicht die Breite oder die Höhe des Sliding-Windows einen Wert, der größer ist als die der Suchregion, bricht an dieser Stelle die Detektion ab.

Endgröße von nur 15 × 27 Pixeln. Der Classifier durchläuft eine Region zur Detektion der Landmarks also nur 2 mal, während bei den anderen Classifiern die Anzahl der Skalierungen/Durchläufe deutlich höher ist. Doch nicht nur die Anzahl der Skalierungen während der Detektion beeinflusst die Laufzeit. Dies fällt, auf wenn die Laufzeiten der Classifier ohne dem Setzen der Suchregionen analysiert werden. Bei diesen ist zu sehen, dass der Classifier der Alare eine größere Durchlaufzeit als der Nasen-Classifier besitzt obwohl, dieser sein Sliding-Window 8 mal öfter skaliert. Eine Antwort auf diese Beobachtung gibt die Größe des Sliding-Windows bei der Detektion. Das Sliding-Window des Nasen-Classifiers ist größer und muss deshalb nicht so viele Regionen Scannen, wie das Sliding-Window des Alare-Classifiers. Bei der Leistungsfähigkeit betreffend der Detektionslaufzeit lässt sich abschließend sagen, dass sich diese durch das Setzen der Suchregionen um den Faktor $11,9947 \approx 12$ verringert. Ein direkter Vergleich mit dem von Mian [Mia11] vorgestellten Ansatz ist nur bedingt möglich. Ein Grund hierfür ist die unterschiedliche Systemkonfiguration bei der Mian eine durchschnittliche Programmlaufzeit von 210 ms angibt. Die einzige gelistete Komponente ist die CPU, die mit 2,4 GHz taktet. Ein anderer Grund liegt in der geringeren Anzahl von nur 3 detektierten Landmarks (die beiden Exocanthions und die Nasenspitze) im Vergleich zu der in dieser Arbeit 10 detektierten. Der letzte Faktor, der einen direkten Vergleich unmöglich macht, ist in der Implementierung zu finden. Die Sliding-Windows der Classifier von Mians Verfahren vergrößern sich nur ein einziges mal, bevor die Suche nach dem nächsten Landmark startet. Im Grunde lässt sich sagen, dass auch bei Mians Ansatz das Setzen von Suchregionen eine schnellere Ausführungszeit bewirkt, da die Classifier beider Verfahren auf dem Viola-Jones Ansatz beruhen und diese unter gleichen Rahmenbedingungen die gleiche Ausführungszeit benötigen.

Programmlaufzeit mit dem Setzen der Suchbereiche vs ohne dem Setzen der Suchbereiche



	Verfahren mit eingeschränkten Suchbereichen		Verfahren ohne eingeschränkten Suchbereichen	
Classifier	∅ Suchbereich	∅ Laufzeit in ms	∅ Suchbereich	∅ Laufzeit in ms
Nase	640 × 480	184,67	640 × 480	184,67
Pupillen	640 × 89	117,88	640 × 480	824,63
Exocanthion	262 × 89	52,09	640 × 480	806,4
Cheilion	90 × 69	31,53	640 × 480	1927,08
Endocanthion	84 × 89	17,56	640 × 480	933,63
Alare	104 × 69	5,05	640 × 480	226,81
Gesamt:		408,78		4903,21

Abbildung 5.1.: Die Laufzeiten der einzelnen Classifier logarithmisch dargestellt. Obwohl der Classifier der Alare im Schnitt eine größere Region untersuchen muss als die der Cheilion oder der Endocanthion, ist seine Laufzeit geringer, da er das Sliding-Window nur 2 mal aufskalieren muss.

5.2. Präzision

In der Regel hat ein solider Classifier mit einer geringen false negative rate und einer hohen true positive Rate mehrere tausend positive wie negative Images als Trainingsgrundlage. Da jedoch das Erstellen positiver Images einen manuellen Arbeitsschritt erfordert, ist der Prozess sehr zeitintensiv. Die Generierung, der in dieser Arbeit verwendeten Classifier, ließ sich beschleunigen, indem als Trainingsgrundlage nur 500 (durch die horizontale Spiegelung 1000) positive Images genutzt wurden. Ein weiterer Faktor, welcher gleichermaßen die Präzision und die Trainingszeit beeinflusst, sind die beim Trainingsprozess vorgegebenen Stufen. Ein solider Classifier hat um die 20 Stufen. OpenCV liefert mit dem „haarcascade_frontalface_alt_tree“ Classifier sogar einen Classifier aus, welcher zur Gesichtsdetektion auf Farbbildern verwendet wird und aus 46 Stufen besteht. Das bedeutet, dass der Classifier bei dem Training eine $overall\ maxfalsealarmrate$ von $0,5^{46} = 1,421 \cdot e^{-14}$ erreicht. Um die Trainingszeit zu verkürzen sind die generierten Classifier allerdings auf 14 Stufen trainiert und haben folglich eine größere $overall\ maxfalsealarmrate$ von $0,5^{14} = 6,104 \cdot e^{-5}$. Da diese Zahl immer noch sehr klein ist, hat es den Anschein, als ob 14 Stufen ausreichen um einen soliden Classifier zu generieren, in dessen Detektionsergebnis wenig false positives enthalten sind. Das folgende Beispiel soll jedoch verdeutlichen, was es mit dieser Zahl auf sich hat und wie sich die Präzision der Classifier in ihr widerspiegelt. In Abbildung 5.2 Seite 44 ist jeweils ein Sliding-Window mit einer Größe von 2×2 Pixeln zu sehen. Durchläuft das Sliding-Window ein 3×3 Pixel großes Bild, muss es 4 Regionen scannen. Bei einem 4×4 Pixel großen Bild, muss es 9 Regionen scannen. Die untersuchten Regionen in Abhängigkeit von der Sliding-Window Größe und der

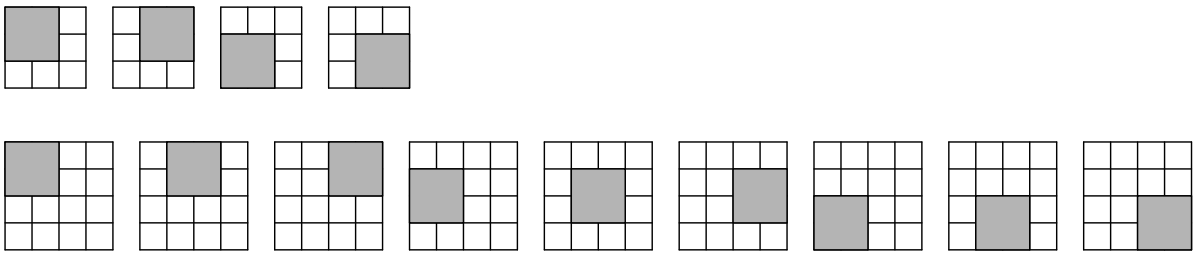


Abbildung 5.2.: Ein 2×2 Pixel großes Sliding-Window eines Classifiers, welches in der oberen Reihe ein 3×3 Pixel großes Bild und in der unteren Reihe ein 4×4 Pixel großes Bild nach Haar-Like Features durchsucht. Für das 3×3 Pixel große Bild müssen 4, für das 4×4 Pixel große Bild müssen 9 Regionen gescannt werden.

Bildgröße lassen sich durch die Formel $(Image_w - SW_w + 1) \cdot (Image_h - SW_h + 1)$ darstellen, wobei SW für Sliding-Window, w für width und h für height steht. Der Cheilion-Classifier z.B. hat als Startgröße des Sliding-Windows 14×14 Pixel und muss dieses 22 mal um 5% vergrößern bis es seine Maximalgröße von 40×40 Pixeln erreicht hat. Durchsucht der Clas-

sifier ein 640×480 Pixel großes Bild, hat er insgesamt 6.191.670 Regionen auf Haar-Features untersucht. Bei einer Wahrscheinlichkeit von $0,5^{14}$ sind statistisch $6191670 \cdot (0,5^{14}) = 377,19$ false positives detektiert. Siehe Abbildung 5.3 Seite 45. Aus diesen vielen Punkten die 2 Fina-

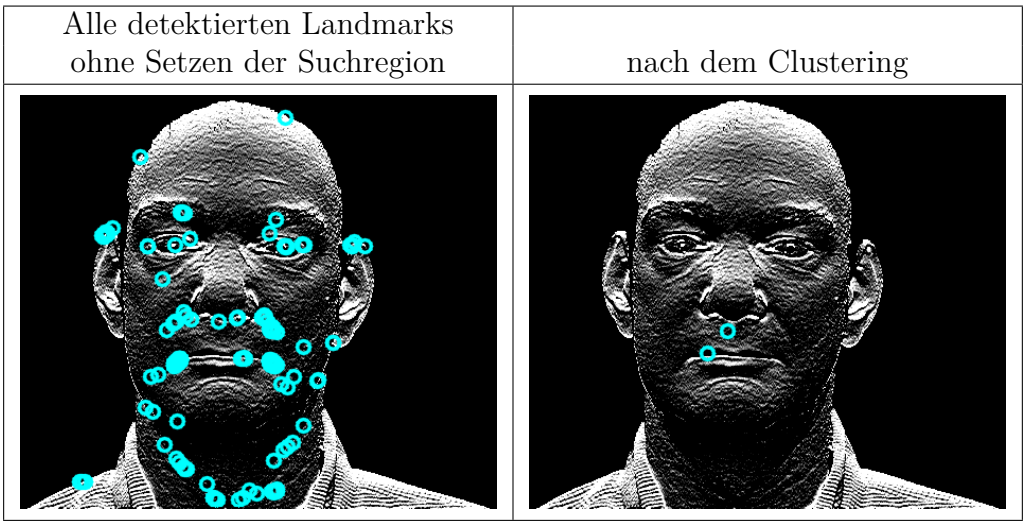


Abbildung 5.3.: Detektion der rechten und linken Cheilion ohne den Suchbereich einzugrenzen. Da der Hintergrund komplett Schwarz ist und keine Informationen enthält, kann der Classifier hier keine Landmarks detektieren. Die Abbildung zeigt auf der linken Seite 228 Detektierte Landmarks. Nach dem Clustering sind die beiden finalen Punkte auf der rechten Seite zu sehen.

len Positionen der Landmarks zu bestimmen, wäre ein ungenügendes Ergebnis im Bezug auf die Detektionsgenauigkeit. Die Mundwinkel sind mit dem GroundTruth Editor für die linke Cheilion bei (355;345) und für die rechte Cheilion bei (261;343) markiert. Bei der Detektion ohne dem Setzen der Suchbereiche ist die finale Position der linken Cheilion bei (301;312) und die der rechten Cheilion bei (282;333). Das macht eine Abweichung von $63,285 \approx 63$ Pixeln für die linke und eine Abweichung von $23,707 \approx 24$ Pixeln für die rechte Cheilion. Im Vergleich dazu ist bei einem Durchlauf mit dem Setzen der Suchregionen die finale Position der linken Cheilion bei (354;344) und die der rechten bei (262;341). Das macht eine Abweichung von 1 Pixel für die linke und eine Abweichung von 2 Pixeln für die rechte Cheilion mit dem Setzen der Suchbereiche. Bei diesem Datensatz ist deutlich zu sehen, wie stark sich die Präzision der finalen Landmarkpositionen durch das Setzen der Suchregionen verbessert.

5.2.1. Gegenüberstellung der einzelnen Classifier

Um eine detaillierte Präzisionsübersicht aller Classifier zu erhalten, ist ein Vergleich zwischen den manuell gesetzten Ground Truth-Daten und den Detektionsergebnissen von 500 Datensätzen entstanden. Dabei sind die jeweils rechten und linken Landmarks pro Classifier zusammengefasst, sodass für jeden Classifier 1000 detektierte Punkte und 1000 manuell gesetzte Punkte zum Vergleich vorliegen. Auf der einen Seite steht die *Detektionsrate* und auf der anderen die Abweichung in Pixeln. Die *Detektionsrate* hat ein Intervall von 0-100% und die Abweichung in Pixeln ein Intervall von 0-20 Pixel. Eine Abweichung größer 20 Pixel ist als falsch detektierter Landmark angesehen. Detektiert ein Classifier 250 Landmarks mit einer Abweichung von 0 Pixeln und 250 Landmarks mit einer Abweichung von 1 Pixel, ist bei einer Abweichung von 0 Pixel die *Detektionsrate* = 25% und bei einer Abweichung von einem Pixel die *Detektionsrate* = 50%. Die Auswertungsergebnisse sind in Abbildung 5.4 Seite 47 zu sehen. Die Performance der Classifier, welche mit dem Setzen der Suchbereiche erreicht wird, liegt bei allen, mit Ausnahme des Exocanthion-Classifiers, sehr dicht beieinander. Die 4 Classifier der Alare, Pupille, Endocanthion und Cheilion detektieren über 90% der Landmarks mit einer Abweichung von 4 Pixel. Der Grund, weshalb der Exocanthion-Classifier eine schlechtere Performance besitzt, liegt an seinem Trainingsprozess. Um die Homogenität der positiven Images zu erhöhen, sind nur die Exocanthions von „normal“ geöffneten Augen ausgeschnitten worden. Bei zu weit geschlossen Augen während der Detektion ist die Exocanthion durch ihre starke Formverzerrung nicht mehr detektierbar. Die Analyse der Classifier, die eine Detektion ohne das vorherige Definieren der Suchbereiche vollziehen, zeigt einen sehr deutlichen Performanceabfall. Einzig der Classifier der Pupille kann bei einer Abweichung von 20 Pixeln an die Performance des Exocanthion-Classifiers mit dem Setzen der Suchregionen heranreichen, erreicht diese aber nicht. Eine Separierung von rechten und linken Leistungsmessungen vollzieht die hier aufgestellte Analyse bewusst nicht. Grund dafür ist, dass die Präzision mit der die jeweiligen Landmarks detektierbar sind, in der Regel sehr dicht beieinander liegen. Da ab einer Abweichung von 21 Pixeln die Landmarks als nicht detektiert gelten, definiert die Tabelle 5.2 Seite 47 die Begriffe false negative, true negative, false positive und true positive, um die fehlenden Informationen, die aus Abbildung 5.4 Seite 47 nicht hervorgehen, darzustellen. Diese zeigt nur die true positives, bzw. die daraus ableitbaren false negatives. Um zu veranschaulichen, wie sich die false positive rate und die true negative rate berechnen lassen, sei ein kurzes Beispiel genannt. Sind 100 Pupillen wegen geschlossener Augen nicht mit dem GroundTruth Editor markiert und detektiert ein Classifier aus diesen 100 Landmarks fälschlicherweise 20 Stück, hat er eine false positive rate von 20%. Erkennt er aus diesen 100 nicht markierten Landmarks korrekterweise 80 Pupillen nicht, besitzt er eine true negative rate von 80%. Abbildung 5.5 Seite 48 ergänzt diese Informationen. Die Classifier der Alare und Cheilion sind nicht abgebildet, da

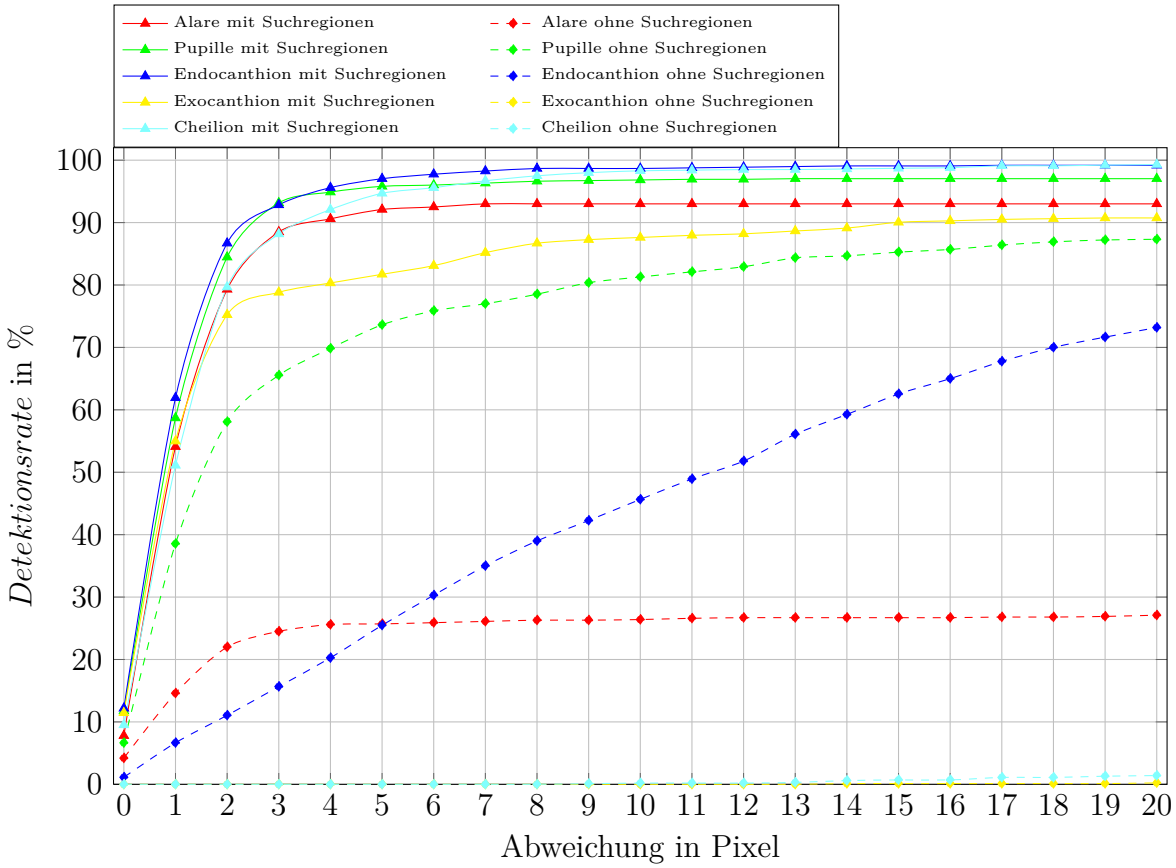


Abbildung 5.4.: Präzisionsvergleich zwischen dem entwickelten Verfahren und der Detektion ohne dem vorherigen Setzen der Suchregionen. Grundlage sind die Detektionsergebnisse und Ground Truth-Informationen von 500 Datensätzen.

	Landmark wird detektiert und Abweichung < 21 Pixel	Landmark wird detektiert und Abweichung ≥ 21 Pixel	Landmark wird nicht detektiert
GroundTruth Daten für Landmark gesetzt	true positive	false negative	false negative
GroundTruth Daten für Landmark nicht gesetzt	false positive	false positive	true negative

Tabelle 5.2.: Definition der Begriffe: false negative, true negative, false positive und true positive.

diese Landmarks niemals verdeckt werden und daher keine true negatives oder false positives bei der Detektion entstehen können. Auch bei der true negative und der false positive rate kann

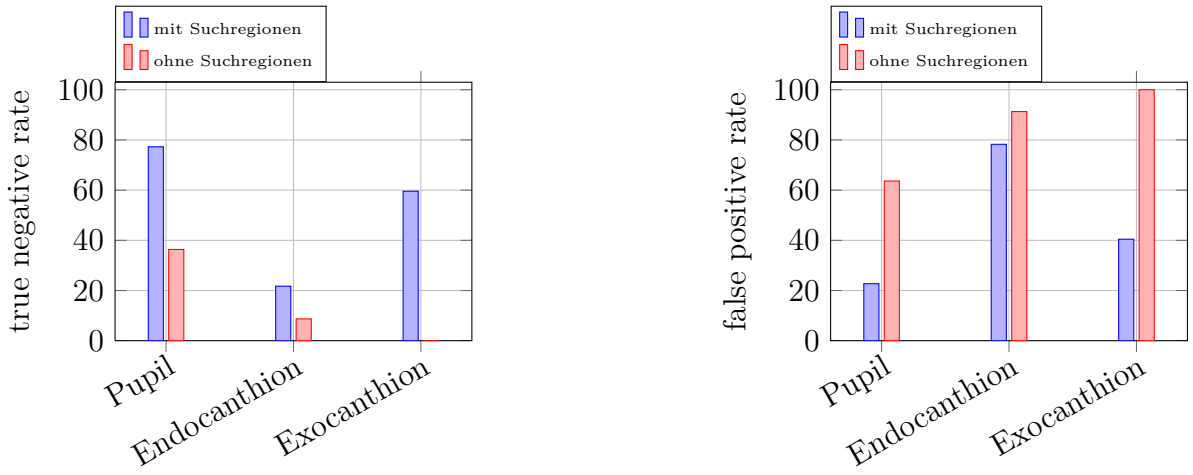


Abbildung 5.5.: True negative und false positive rate des entwickelten Verfahrens. Zu sehen ist die Performance des Pupillen-, Endocanthion- und Exocanthion-Classifiers. Eine true negative rate von 100% und eine false positive rate von 0% wäre das Optimum eines jeden Classifiers.

das vorherige Setzen der Suchregionen die Performance der Classifier deutlich verbessern. Einzig die Classifier der Endocanthions liegen dichter beieinander. Bei dem Exocanthion Classifier ist sehr deutlich erkennbar, dass dieser trotz der Abwesenheit der gesuchten Landmarks diese bei jedem Suchlauf detektiert, falls er auf dem gesamten Bild sucht. Abschließend ist zu der Leistungsfähigkeit der einzelnen Classifier zu sagen, dass diese separat betrachtet sehr gering ausfällt. Durch Kombination mit einem starken Nasen-Classifiers, welcher nach einem Objekt mit großem Alleinstellungsmerkmal sucht und dadurch die Suchbereiche der anderen Classifier vorgibt, verbessert sich die Laufzeit und Präzision der gesamten Landmarkdetektion.

6. Zusammenfassung und Ausblick

6.1. Zusammenfassung

Im Verlauf dieser Arbeit ist durch die Berücksichtigung der anatomischen Eigenschaften des Gesichts und der Kaskadierten Suche ein zuverlässiges Landmarkdetektionssystem für dreidimensionale Gesichtsdaten entstanden. Das Zusammenspiel aus separat betrachtet, schlechten Classifiern, macht es Möglich, die Suchbereiche von Landmarks schon vor ihrer Detektion einzugrenzen. Dadurch konnten die mit nur 500 Bilder Trainierten Classifier die Systemperformance drastisch erhöhen. Aufnahmefehler, welche in anderen Forschungsprojekten Hindernisse darstellen, wandeln die in dieser Arbeit erstmalig eingesetzten xAbs-Gradient Darstellungen zum Vorteil um. Alle im Vorfeld bekannten Herausforderungen konnten überwunden und so die gesteckten Ziele erreicht werden. Das Verfahren ist schnell, präzise und zu einem bestimmten Grad Posen und Mimik unabhängig. Zudem konnte das finale Detektionsvolumen von 10 Landmarks erreicht werden.

6.2. Ausblick

Das entwickelte Verfahren kombiniert eine geringe Ausführungszeit mit einer hohen Präzision und stellt somit ein Verfahren für Einsatzszenarien vor, in denen beides gefordert ist. Das Potential des entwickelten Ansatzes ist jedoch bei weitem nicht ausgeschöpft.

Erhöhung von Präzision und Verringerung der Laufzeit

Durch eine Verbesserung der einzelnen Classiefer lässt sich auch die Gesamtperformance des Systems erhöhen. Dazu muss beim Training der Classifier angesetzt werden, dass bei der vorgestellten Implementierung nur maximal 500 positive Images zur Verfügung hat. Durch eine

Erhöhung der positiven und negativen Trainingsimages ist der Schwellwert, ab dem eine Region die Stufe eines Classifiers passiert, feiner justierbar. Aber nicht nur die Anzahl der Training-Images können zukünftige Arbeitsschritte erhöhen, sondern auch die Anzahl der Stufen eines jeden einzelnen Classifiers. Dadurch bringen die Classiefier weit aus weniger false positives hervor, da sich die Wahrscheinlichkeit mit der eine Region ohne Landmark alle stufen passiert, verringert. Um die Performance im Bezug zur Laufzeit zu verbessern, kann, bei guten Classifiern vorausgesetzt, auch deren Skalierung erhöht werden.

Datenbankübergreifende Entwicklung

Um eine vollständige Leistungsanalyse des Verfahrens zu vollziehen, sind Datenbankübergreifende Messungen von Nöten. Die dargestellten Analysen begrenzen sich wie auch das Training der Classifier auf die FRGC Datenbank, die nur einen Teil der verfügbaren 3D Gesichtsdatenbanken widerspiegelt. Um ein flächendeckendes System zu entwickeln, müssen daher auch andere Datenbanken bei dem Training und der Implementierung mit einbezogen werden.

Vollständige Evaluierung durch Testingdaten

Es ist anzumerken, dass die Evaluierung die gleichen Datensätze verwendet, die auch bei dem Training der Classifier genutzt worden sind. Eine Untersuchung mit unabhängigen Daten ist zu diesem Zeitpunkt nicht möglich, da die Ground Truth-Informationen zu diesen noch nicht vorliegen. Dieser Umstand beeinflusst die Präzisionsmessungen des Programms, welche mit realen Testdaten voraussichtlich schlechter sind als die hier dargestellten. Die Vollständige Evaluierung gibt tiefere Einblicke in die Systemperformance und stellt die realen Arbeitsbedingungen besser dar.

Erhöhung der Robustheit gegenüber der Pose

Zur logischen Struktur der Implementierung ist zu erwähnen, dass sich bis zum jetzigen Zeitpunkt nur Landmarks von nicht zu stark rotierten Gesichtern detektieren lassen, da die Suchbereiche der jeweiligen Landmarks wie der Pupille oder der Mundwinkel Horizontal ausgelegt sind. Siehe Abbildung 6.1 Seite 51. Die Lösung dieses Problems könnte die Detektion eines Landmarkpaares, wie der beiden Pupillen oder der beiden Alares bilden. Durch ihre Höhendifferenz ist dann die Rotation des Kopfes und der Suchregionen berechenbar.

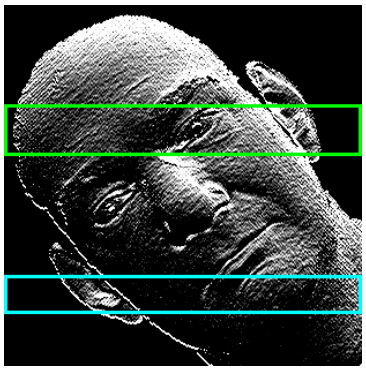


Abbildung 6.1.: Durch die 45° Rotation des Kopfes und das horizontale Setzen der Suchregionen liegt die rechte Pupille und die linke Cheilion nicht mehr in dem jeweiligen Suchbereich.

Größeres Detektionsvolumen

Ein naheliegender Schritt ist weiterhin die Anzahl der zu detektierenden Landmarks zu erhöhen. Bis zum jetzigen Zeitpunkt sind 10 Landmarks korrekt detektiert. Das Gesicht beinhaltet allerdings über 50 Landmarks. Siehe Kapitel 1.5 Seite 16. Ab einer gewissen Anzahl korrekt detektierter Landmarks ist eine Implementierung in Richtung Identifikations-/ Verifikationssystem denkbar. Durch Verarbeitungsschritte zur Eliminierung von Aufnahmestörungen im Bereich der detektierten Landmarks ist die Tiefenkomponente der Messpunkte in $\mathbb{R}^3$ mit ein beziehbar, um durch relative Abstandsmessungen Daten des Gesichts mit anderen zu vergleichen.

7. Schlusswort

Das letzte Wort dieser Arbeit geht an die Wissenschaft im allgemeinen. Bei der Entwicklung der Landmarkdetektion für 3D-Gesichtsdaten ist aufgefallen, dass viel manuelle Arbeitsschritte sehr störend, zeitaufwendig und vor allem unnötig sein könnten. Das Beispiel auf dem diese Aussage beruht, bezieht sich auf das manuelle setzen der GroundTruth-Daten für die 500 Datensätze, um die positiven Images zu erstellen. Durch einheitliche Standards und vor allem eine exzessivere Nutzung des Internets in diesem Bereich, könnten sich immer wiederkehrende Aufgaben, welche von jeder Forschungsgruppe von neuem vollzogen werden müssen, umgehen. So könnten die GroundTruth Informationen auf einer Plattform im Internet angeboten werden.

A. Userinterface des GroundTruth Editor

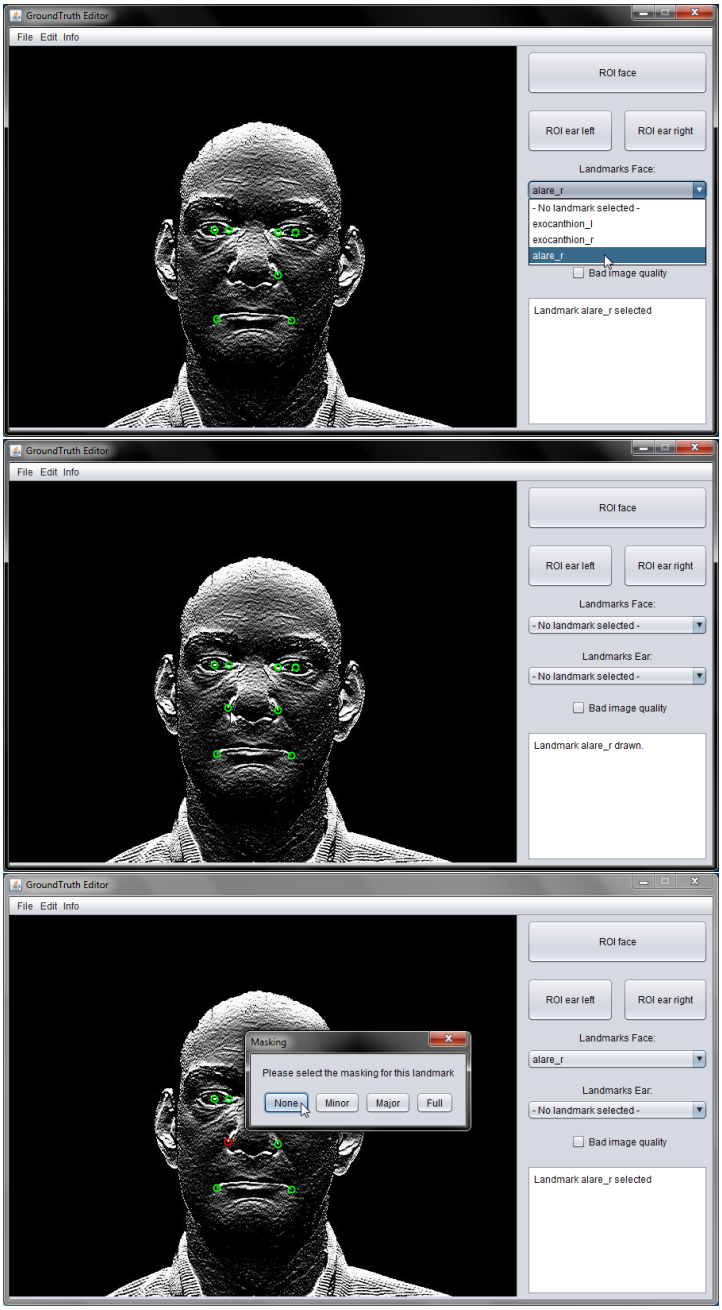


Abbildung A.1.: Screenshot des GroundTruth Editors. Von oben nach unten: Auswahl des Lanmarks, einzeichnen des Landmarks in das Image, Auswahl des Verdeckungsgrades des Landmarks.

B. Struktur eines GroundTruth Editor Datensatzes

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<groundtruth>
<image_quality bad_quality="false"/>
<roi_face X1="0" X2="0" Y1="0" Y2="0"/>
<roi_ear_left X1="0" X2="0" Y1="0" Y2="0"/>
<roi_ear_right X1="0" X2="0" Y1="0" Y2="0"/>
<image_mask X1="0" X2="0" Y1="0" Y2="0"/>
<landmarks_face>
	<landmark abbreviation="En" masking="None" name="endocanthion_l" radius="8" x="400" y="255"/>
	<landmark abbreviation="En" masking="None" name="endocanthion_r" radius="8" x="333" y="258"/>
	<landmark abbreviation="Ex" masking="None" name="exocanthion_l" radius="8" x="440" y="248"/>
	<landmark abbreviation="Ex" masking="None" name="exocanthion_r" radius="8" x="292" y="254"/>
	<landmark abbreviation="P" masking="None" name="center point of pupil_l" radius="8" x="42" y="24"/>
	<landmark abbreviation="P" masking="None" name="center point of pupil_r" radius="8" x="31" y="25"/>
	<landmark abbreviation="Al" masking="None" name="alare_l" radius="8" x="405" y="322"/>
	<landmark abbreviation="Al" masking="None" name="alare_r" radius="8" x="338" y="325"/>
	<landmark abbreviation="ch" masking="None" name="cheilion_l" radius="8" x="417" y="370"/>
	<landmark abbreviation="ch" masking="None" name="cheilion_r" radius="8" x="326" y="375"/>
</landmarks_face>
</groundtruth>
```

Abbildung B.1.: GroundTruth-Datensatz detektierter Landmarks. Alle detektierten Landmarks befinden sich in dem Tag „<landmarks_face>“. Der äußere Tag „<groundtruth>“ ist für den bei der Forschungseinrichtung CASED entwickelten GroundTruth Editor vorgesehen. Das Attribut „abbreviation“ Steht für das Kürzel des Landmarks, „masking“ für dessen Verdeckungsgrad welcher die Werte „None“, „Minor“, „Major“ und „Full“ annehmen kann. Die x- und y-Koordianten geben die Position des Landmarks im Bild an.

Stichwortverzeichnis

Algorithmus
- AdaBoost, 24
- ICP, 19
- PCA, 19

Artefakt, 10, 13, 19
- Hole, 13
- Spike, 13

Aufnahmeverfahren
- aktiv, 11
- hybrid, 12
- passiv, 12

Biometrie, 9
- Charakteristika, 9

Clustering, 32
- Cluster, 31
- Zentroid, 32

Detektionsrate, 46

Extrusionskarten, 20

false negative, 44, 46
false positive, 45, 46
false positive rate, 46
FRGC, 18
- Datenbank, 17–19, 35

Gradient, 22
- Image, 22
- x-Gradient, 22, 37
- xAbs-Gradient, 22, 37
- y-Gradient, 22, 34

GroundTruth Editor, 34, 35

Haar-Like Feature, 24, 25
- Modi, 39
  - ALL, 39
  - BASIC, 39
  - CORE, 39

Identifikation, 9, 13
Integralbild, 24
Interpolation
- bikubisch, 15
- linear, 15

k-means, 32
- k-means++ Algorithmus, 32

Landmark, 14
Landmarkübersicht, 15

Ratio, 37

Shape-Indize, 20
Sliding-Window, 25, 31, 41, 42
Sping-Image, 20
Streifenlichttopometrie, 11
Streifenprojektion, 11

Trainingsbild
- negativ, 35

positiv, 35, 37
Treshold-Funktion, 31
true negative, 46
true negative rate, 46
true positive, 44, 46

vec-File, 37
Verifikation, 9, 13

Literaturverzeichnis

[Alc12] Alces Technology: *3D depth capture and Structured Light.* `http://main.alcestech.com/2012/01/3d-depth-capture-and-sensing.html`. Version: Jan 2012. – zuletzt besucht am 21.01.2013

[AV07] Arthur, David ; Vassilvitskii, Sergei: k-means++: The Advantages of Careful Seeding. In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* 18 (2007), S. 1027 – 1035

[BCF06] Bowyer, Kevin ; Chang, Kyong ; Flynn, Patrick: A survey of approaches and challenges in 3D and multi-modal 3D + 2D face recognition. In: *Computer Vision and Image Understanding* 101 (2006), S. 1 – 15

[CCS06] Colombo, Alessandro ; Cusano, Claudio ; Schettini, Raimondo: 3D face detection using curvature analysis. In: *Pattern Recognition* 39 (2006), S. 444 – 455

[DSA08] Dibeklioglu, Hamdi ; Salah, Albert A. ; Akarun, Lale: 3D Facial Landmarking under Expression, Pose, and Occlusion Variations. In: *Journal of Physics D-applied Physics* (2008)

[KON] KONICA MINOLTA: *VIVID 910 3D Laser Scanner.* `http://sensing.konicaminolta.us/products/vivid-910-3d-laser-scanner/`. – zuletzt besucht am 25.01.2013

[LBJ12] Luximon, Yan ; Ball, Roger ; Justice, Loriane: The 3D Chinese head and face modeling. In: *Computer-Aided Design* 44 (2012), S. 40 – 47

[LM02] Lienhart, Rainer ; Maydt, Jochen: An Extended Set of Haar-like Features for Rapid Object Detection. In: *International Conference on Image Processing* 3 (2002), S. 1 – 900

[MBO04] Mian, Ajmal S. ; Bennamoun, Mohammed ; Owens, Robyn: Automated 3D model-based free-form object recognition. In: *Sensor Review* 24 (2004), S. 206 – 215

[Mia11] Mian, Ajmal: Robust Realtime Feature Detection in Raw 3D Face Images. In: *Applications of Computer Vision WACV, IEEE Workshop* (2011), S. 220–226

[Nat11] National Institute of Standards and Technology: *Overview of the FRGC*. `http://www.nist.gov/itl/iad/ig/frgc.cfm`. Version: Feb 2011. – zuletzt besucht am 22.01.2013

[PPTK10] Perakis, Panagiotis ; Passalis, Georgios ; Theoharis, Theoharis ; Kakadiaris, Ioannis A.: 3D Facial Landmark Detection & Face Registration: A 3D Facial Landmark Model & 3D Local Shape Descriptors Approach / Computer Graphics Laboratory, University of Athens. 2010. – Forschungsbericht

[PWW03] Pan, Gang ; Wu, Yijun ; Wu, Zhaohui: Investigating Profile Extracted from Range Data for 3D Face Recognition. In: *International Conference on Systems, Man and Cybernetics* 2 (2003), S. 1396 – 1399

[SA06] Salah, Albert A. ; Akarun, Lale: 3D Facial Feature Localization for Registration. In: *Workshop on Multimedia Content Representation, Classification and Security* 4105 (2006), S. 338 – 345

[SCAS07] Salah, Albert A. ; Cinar, Hatice ; Akarun, Lale ; Sankur, Bülent: Robust Facial Landmarking for Registration. In: *Annals of Telecommunications* 62 (2007), S. 1608 – 1633

[SLML09] Sun, Xianfang ; L.Rosin, Paul ; Martin, Ralph R. ; Langbein, Frank C.: Noise Analysis and Synthesis for 3D Laser Depth Scanners. In: *Graphical Models* 71 (2009), S. 34 – 48

[Spr11] Spreeuwers, Luuk: Fast and Accurate 3D Face Recognition. In: *International Journal of Computer Vision* 93 (2011), S. 389 – 414

[VJ01] Viola, Paul ; Jones, Michael: Rapid Object Detection using a Boosted Cascade of Simple Features. In: *Computer Vision and Pattern Recognition* 1 (2001), S. 511 – 518

[YKA02] Yang, Ming-Hsuan ; Kriegman, David J. ; Ahuja, Narendra: Detecting Faces in Images: A Survey. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), S. 34 – 58