Xuan Linh Do
Matrikelnr: 2138196
Media Systems
Dozent: Dr. Andreas Plaß
Titel des Papers: Gesichts- und Punkteerkennung

# Paper: Gesichts- und Punkteerkennung

## Einleitung

Es gibt bereits zahlreiche Apps und Programme, die mit der Gesichtserkennung arbeiten. Ein Beispiel wäre die App Snapchat, die unter anderem einen Videofilter hat, der das Gesicht des Benutzers erkennt und diesen verändert, sodass der Benutzer Hasenohren und lange Schneidezähne bekommt. Sobald der Benutzer eine Kaubewegung macht, sieht man eine Animation auf dem Handy wie der Benutzer eine virtuelle Karotte aufisst.
Aufgrund der Tatsache, dass die Technik der Gesichtserkennung immer populärer wird, gibt es auch schon einige Algorithmen, die für diesen Zweck entworfen worden sind. OpenCV verfügt auch bereits über mehrere Algorithmen zur Gesichtserkennung. Auf einige Ideen und Algorithmen soll in diesem Paper eingegangen werden, wobei dabei nicht ins Detail gegangen werden soll wie die Algorithmen aussiehen und funktionieren. Vielmehr soll beleuchtet werden, wieso gewisse Methoden und Algorithmen nicht für das Projekt in Frage kamen und welche für das Projekt am meisten Sinn gemacht haben.

## Probleme der Gesichtserkennung

Zu unterscheiden ist bei der Gesichtserkennung zwischen verschiedenen Teilproblemen. Für das Projekt ist das Problem der Face Detection und das der Face Point Localization wichtig. Bei der Face Detection geht es darum, auf einem Bild den Bereich zu finden, auf dem sich ein menschliches Gesicht befindet. Das Face Point Localization (bzw. Facial Feature Detection) beschreibt das Problem, bestimmte Gesichtsfeatures (im Paper Face Key Points genannt) zu lokalisieren, beispielsweise die Eckpunkte von den Augen oder dem Mund. Diese Informationen sind zwingend notwendig, damit später unterschieden werden kann, ob unter anderem der Mund geöffnet ist oder nicht.

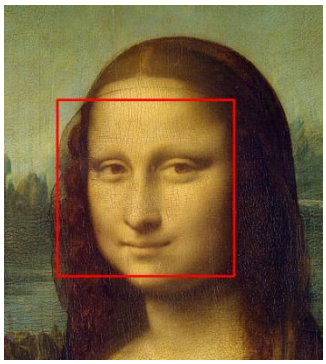

Abbildung 2 Face Detection

Abbildung 1 Face Point Localization

## Erwähnenswerte Methoden für die Face Detection

- Durch Color Detection: Wohl einer der naheliegendsten Methoden, vom Bild den Farbbereich zu extrahieren, deren Farbe zur Hauttonfabe des Gesichts

zugeordnet werden kann. Programmiertechnisch funktioniert dies ungefähr so: Das Bild wird Pixel für Pixel durchgegangen und dabei jeweils die RGB-Farbwerte ausgelesen. Passen die RGB-Werte in dem RGB-Farbbereich, den man vorher als Werte für das Gesicht definiert hat, so wird das Pixel weiß. Gehört es nicht zu diesem definierten Farbbereich, wird der Pixel einfach schwarz gefärbt. Dies funktioniert alles noch sehr einfach, lässt sich aber noch deutlich verbessern, zum Beispiel durch das Nutzen von Mittelwerten oder anderen Farbräumen wie YcbCr oder HSI/HSV (siehe dazu [i]). Nachteile an dieser Methode sind jedoch, dass es nicht mit allen Typen von Hautfarbe funktioniert und stark von der Beleuchtung abhängig ist.

Abbildung 3 Face Color Detection

- Durch Bewegung: Bei Videos kann man annehmen, dass das Gesicht das einzige ist, was sich bewegt oder verändert. Dieses Wissen kann man sich zu nutze für die Face Detection machen. Dafür werden die verschiedenen Videoframes miteinander verglichen. Anhand verschiedener Kriterien (siehe dazu [ii]), kann dann festgestellt werden, in welchem Bereich eine Bewegung stattgefunden hat. Nachteil an dieser Methode ist jedoch, dass kein Gesicht erkannt werden kann, wenn sich außer dem Gesicht noch was anderes in dem Video bewegt.
- Durch modelbasierte Gesichtserkennung mithilfe der Technik der Kantenerkennung (siehe [iii]) und/oder der Hausdorff Distance (siehe [iv]).

## Face Detection durch Haar-like Feature

Bei Haar-like Features wird - sehr grob erklärt - das Bild mit einem Detection Window durchgegangen, in diesem Detection Window rechteckige Bereiche anhand der Unterschiede in der Pixelintensität gebildet und anhand dieser mehrere Subregionen im Bild kategorisiert und so Haar-like Features kalkuliert.

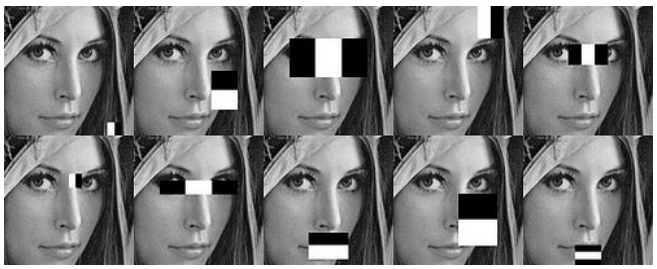

Abbildung 4 Face Detection mit Haar-like Features

Der sogenannte Viola-Jones-Algorithmus für die Gesichtserkennung mit Haar-Features ist bereits in OpenCV implementiert. Mit diesem lassen sich Haar-Features für die Gesichtserkennung durch Machine Learning trainieren, wofür als Trainingsdateien viele Beispielbilder von Gesichtern benötigt werden. Anhand der zahlreichen Bildern werden Beschreibungsdateien errechnet und als .xml Dateien

gespeichert. Praktischerweise kann man sich auch den Schritt des doch sehr zeitaufwändigen Trainings ersparen, da es in OpenCV bereits Beschreibungsdateien, auch Haar Cascades genannt, gibt. Mit der Funktion `cv2.CascadeClassifier()` wird die Haar Cascade (also die xml Datei) geladen. Anschließend wird mithilfe der Anwendung der Funktion `detectMultiScale()` auf den CascadeClassifier die entdeckten Gesichter als Liste von Rechtecken zurückgegeben. Das Nutzen von Haar-like Features erweist sich in der Praxis als sehr schnell und damit effektiv.

## Face Detection mithilfe der Dlib C++ Library

Die Dlib C++ Library beinhaltet Algorithmen für Machine Learning, unter anderem findet sich dort ein Face Detector, der mit dem Classic Historgram of Oriented Gradients (HOG) Feature kombiniert mit einem linearen Classifier, einer Bilder Pyramide, und dem Sliding Window Detection Scheme arbeitet. Der Detector funktioniert in der Praxis auch schnell genug. Doch zum Dlib Face Detector wurden beim Recherchieren im Internet vergleichsweise weniger Beispiele und Informationen gefunden als zu dem von OpenCV.

## Vorzeitiges Fazit für die Face Detection

Sowohl der Face Detector von OpenCV als auch der von Dlib Library scheinen gut und schnell zu funktionieren. Da der Gesichtsdetektor von OpenCV populärer ist und sich zu ihm mehr im Internet finden lässt als der von der Dlib Library, schien es besser sich für den OpenCV Gesichtdetektor zu entscheiden.

## Face Point Localization

Sobald das Gesicht aus dem Bild herausgeschnitten wurde, ist das nächste Problem des Projekts die Face Key Points zu ermitteln.

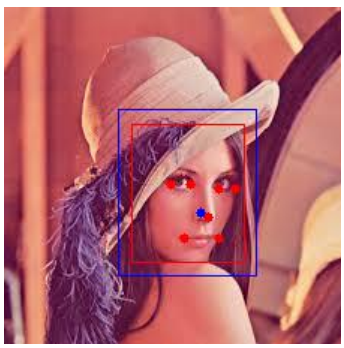

**Abbildung 5 Ergebnisse des Flandmark Point Detector**

- Der Viola-Jones-Detektor bietet auch die Möglichkeit diese Face Key Points zu finden. Jedoch funktioniert der Algorithmus zu diesem Zweck nicht robust und genau genug.
- Der Flandmark Point Detector findet die vier Ecken von den Augen, die zwei Ecken des Mundes, die Mitte der Nase und die Mitte des Gesichts. Für das Projekt sind dies jedoch zu wenig Face Key Points, die zurückgegeben werden.

Xuan Linh Do
Matrikelnr: 2138196
Media Systems
Dozent: Dr. Andreas Plaß
Titel des Papers: Gesichts- und Punkteerkennung

- Der Clandmark Point Detector ist eine Fortführung vom Flandmark Point Detector und soll diesen ersetzen. So gibt er deutlich mehr Face Key Points zurück als sein Vorgänger. Jedoch funktioniert er bisher nur für Linux und ist kaum dokumentiert. Damit ist er für das Projekt noch nicht nutzbar.
- Die Dlib C++ Library beinhaltet ebenfalls eine Landmark Detection. Dieser ist in der Lage 68 verschiedene Face Key Points zu finden und deren Position zu bestimmen (siehe Abbildung). Für das Projekt würden aber wahrscheinlich lediglich die Punkte 18-27 ( Augenbrauen), 37-48 (Augen), 49-68 (Mund) relevant sein.

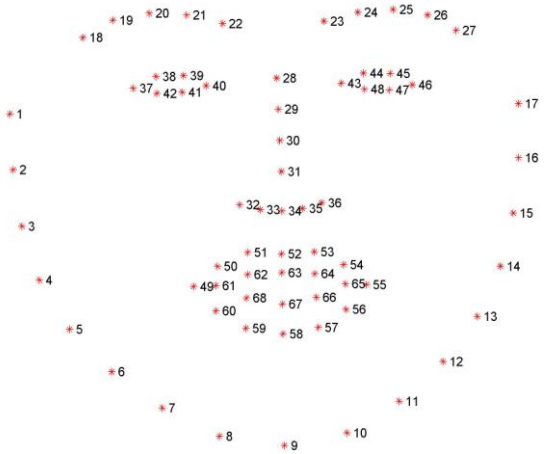

**Abbildung 6 Die 68 Face Key Points vom Landmark Detector aus der Dlib Library**

- Das CLM-Framework kann sogar zusätzlich zum Bestimmen der Face Key Points noch die Kopfposition zurückgeben. Das CLM-Frameworks benötigt ebenfalls für die Benutzung die Dlib Library. Beim Testen hat sich jedoch herausgestellt, dass das CLM-Framework langsamer ist als die Dlib library.
- Das OpenFace Tool ist eine Fortführung des CLM-Framworks und bietet eine bessere Erkennung mithilfe von neuronalen Netzen und weitere Funktionen wie das Erkennen der Blickrichtung.

## Fazit Face Point Localization

Für das Projekt schien es am besten einfach die Landmark Detection von der Dlib Library zu verwenden. So reicht sie zum Zweck des Projekts völlig aus und funktioniert auch schneller als das CLM-Framework und das OpenFace Tool.

## Die endgültige Entscheidung

Anfangs fiel die Entscheidung bei der Face Detection auf die Methode mit den Haar-like Features, da bereits Implementierungen in OpenCV zu finden sind, sie am Populärsten von allen Methoden erscheint, die meisten Tutorials dafür im Internet zu finden sind und zudem schnell und effektiv funktioniert. Bei der Face Point Localization fiel die Entscheidung deutlich aufgrund ihrer Robustheit, Schnelligkeit und besseren Anwedung auf die Dlib Library. Aus diesem Grund schien es doch logischer bei der Face Detection ebenfalls die Dlib library zu verwenden. Videos mit der Thematik, die beiden Face Detectoren zu vergleichen(siehe [v]), bekräftigten diesen Entschluss. So wirkt der Dlib Detector im Vergleich doch zuverlässiger und schlug weniger Fehlalarme.

Xuan Linh Do
Matrikelnr: 2138196
Media Systems
Dozent: Dr. Andreas Plaß
Titel des Papers: Gesichts- und Punkteerkennung

---

Einzelnachweise

[i] Seite „Face Detection in Color Images“. In web.archive.org. Abgerufen 06. November 2016 von http://web.archive.org/web/20091018190530/http://geocities.com/jaykapur/face.html
Seite „Skin Color Based Algorithm for Face Detection” In youtube.com. Abgerufen 06. November 2016 von „https://www.youtube.com/watch?v=HbqG1By2kas

[ii] Seite „Motion Detection”. In web.archive.org. Abgerufen 06. November http://web.archive.org/web/20080522171806/http:/www.ansatt.hig.no/erikh/papers/hig98_6/node2.html

[iii] Seite „ Real-Time Face Detection Using Edge-Orientation Matching “. In link.springer.com. Abgerufen 06. November 2016 von http://link.springer.com/chapter/10.1007%2F3-540-45344-X_12

[iv] PDF „Robust Face Detection Using the Hausdorff Distance ”. In facedetection.com. Abgerufen 06. November https://facedetection.com/wp-content/uploads/AVBPA01BioID.pdf

[v] Video „dlib vs OpenCV face detection ”. In youtube.com. Abgerufen 06. November https://www.youtube.com/watch?v=LsK0hzcEyHI
Video „ Face detection: comparison of methods”. In youtube.com. Abgerufen 06. November https://www.youtube.com/watch?v=9yfdHr3qLyU

Weitere Informationen

1 Seite „ Face Detection Algorithms & Techniques”. In facedetection.com. Abgerufen 06. November https://facedetection.com/algorithms/
2 Seite „ Eine einfache Gesichtserkennung mit OpenCV und scikit-learn”. In eliteinformatiker.de. Abgerufen 06. November http://eliteinformatiker.de/2013/08/28/eine-einfache-gesichtserkennung-mit-opencv-und-scikit-learn
3 Seite „Facial Landmark Detection“ In learnopencv.com. Abgerufen 06. November https://www.learnopencv.com/facial-landmark-detection/
4 Seite -Kein Titel. In dlib.net. Abgerufen 06. November http://dlib.net/face_landmark_detection_ex.cpp.html

Abbilungsverzeichnis

Abb 1 http://intelligentbee.com/wp-content/uploads/2016/02/mona.png
Abb 2 http://res-1.cloudinary.com/demo/image/upload/c_limit,h_540,w_770/WomanFaceSample.jpg
Abb 3 Screenshot vom Video in https://www.youtube.com/watch?v=HbqG1By2kas
Abb 4 http://farm7.static.flickr.com/6072/6088364490_37bdbf776b_d.jpg
Abb 5 http://i.stack.imgur.com/1IN4p.png
Abb 6 http://ibug.doc.ic.ac.uk/media/uploads/images/annotpics/figure_68_markup.jpg