

---

# **Mean Face**

## **Bildbasierte Computergrafik**

Lali Nurtaev

Fakultät für Informatik und Ingenieurwissenschaften  
TH Köln  
Email: [lali.nurtaev@mail.th-koeln.de](mailto:lali.nurtaev@mail.th-koeln.de)

Frechen, 11. Februar 2022

---

# Inhaltsverzeichnis

<b>1 Einleitung</b>	<b>3</b>
<b>2 Facial Landmarks</b>	<b>3</b>
<b>3 Delaunay Triangulation</b>	<b>3</b>
<b>4 PCA</b>	<b>4</b>
<b>5 Averaging und Morphing</b>	<b>5</b>
<b>6 Ergebnis</b>	<b>6</b>
<b>Literatur</b>	<b>8</b>

---

# 1 Einleitung

Die Begriffe „Mean Face“ oder auch „Average Face“ beschreiben ein Durchschnittsgesicht, welches aus mindestens zwei Gesichtern ermittelt wird. Zur Bestimmung des Durchschnittsgesichts existieren verschiedene Methoden. Die Ermittlung erfolgt aber immer auf demselben Weg: Zwischen zwei Pixeln aus unterschiedlichen Bildern wird ein durchschnittlicher Pixelwert ermittelt. Aus diesen setzt sich das Ergebnis zusammen. Die Vorgehensweise für die Ermittlung kann variieren, jedoch wurde für das zugehörige Projekt folgende durchgeführt: Zunächst folgt eine Facial Landmark Detection, um die wichtigen Merkmale im Gesicht zu ermitteln. Danach wird anhand dieser eine Delaunay Triangulation durchgeführt. Im Anschluss wird das den durchschnittlichen Pixeln ein Morphing erzeugt. Die Durchführung wird anhand zwei Bilder erläutert, kann aber auf weitere angewandt werden. Die Schritte der Facial Landmark Detection und Delaunay Triangulation werden in GitHub im Projekt ”1. Versuchäufgeführt. Die vollständige Ermittlung des Mean Face ist im Projekt ”BCG\_Projekt“ hinterlegt. Die Codeausschnitte sind aus den Tutorials der Webseite <https://learnopencv.com/> entnommen.

## 2 Facial Landmarks

Für die Ermittlung eines Durchschnittsgesichts muss zunächst ein Gesicht auf einem Bild erkannt werden. Die Bibliotheken *dlib* und *OpenCV* enthalten Machine Learning und Computer Vision Algorithmen in den Programmiersprachen C++ und Python wie die Gesichtserkennung (Face Detection) [1]. Sie wird ausgeführt, um ein Gesicht bei schlechten Lichtverhältnissen eingrenzen zu können. Für diese wird eine Erkennung der *Facial Landmarks* bzw. der Gesichtsmerkmale ausgeführt. Gesichtsmerkmale sind in allen Gesichtern zu finden und bilden die Augen, Nase, Mund, Augenbrauen und die Gesichtslinie. Diese werden über 68 Gesichtsmerkmalpunkte in x- und y-Koordinaten definiert (siehe Abbildung 1).

Der Facial Landmark detector wurde mit Trainingsdaten angereichert, die manuell mit Koordinaten beschriftet wurden. Die Gesichtsmerkmale werden in zwei Beispielbildern erkannt und mit grünen Punkten abgebildet (siehe Abbildung 2).

## 3 Delaunay Triangulation

Delaunay Triangulation beschreibt die Unterteilung von Dreiecken, welche aus einer Menge von Punkten gebildet wird, bei dem sich kein Punkt in einem Dreieck befindet. Dabei werden die Innenwinkel der Dreiecke möglichst groß gehalten. Die Triangulation ermöglicht eine große Menge an Punkten übersichtlich zu vernetzen [3]. Für die Gesichtserkennung eignet sich das Verfahren, um das Gesicht besser zu unterteilen und im Anschluss ein das Morphing zu vereinfachen. Anhand der ermittelten 64 Landmarks wird die Triangulation für beide Gesichter ausgeführt (siehe Abbildung 3).

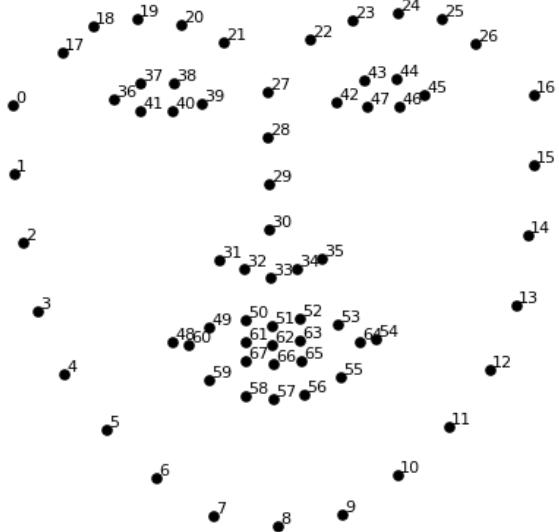


Abbildung 1: 68 Facial Landmarks (Quelle: [2])



Abbildung 2: Facial Landmarks bei beiden Bildern (Quelle: Eigene Darstellung)

## 4 PCA

Die Hauptkomponentenanalyse oder Principal Component Analysis (PCA) ist ein Verfahren zur Dimensionsreduktion von Daten. Das Ziel der PCA ist es mehrdimensionale Daten zu reduzieren oder komprimieren und gleichzeitig relevante Informationen zu bewahren. Um das Ziel der PCA zu erreichen, wird das Konzept der Varianz, welches aus der Statistik kommt, verwendet. Die Varianz misst die Streuung der Daten und kodiert die Informationen, die in den Daten enthalten sind. Der Versuch der PCA Anwendung wurde im Skript "1. Versuch eingefangen und nicht weiter verfolgt.[4].

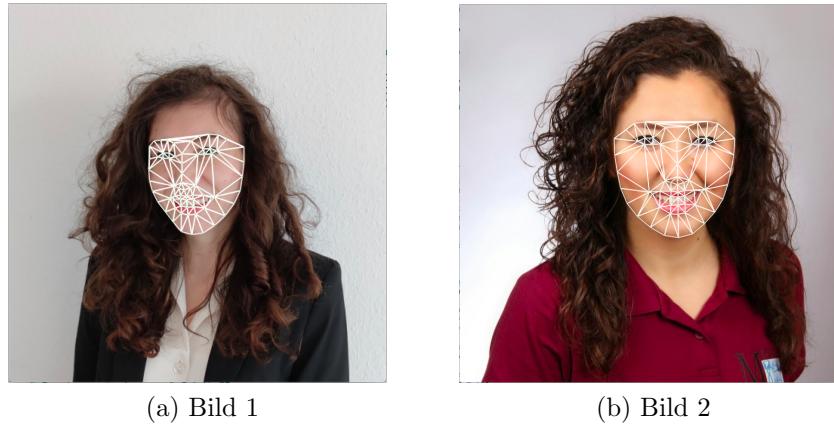


Abbildung 3: Delaunay Triangulation bei beiden Bildern (Quelle: Eigene Darstellung)



Abbildung 4: Anwendung PCA mit 97% (Quelle: Eigene Darstellung)

## 5 Averaging und Morphing

Der letzte Schritt für die Ermittlung des Durchschnittsgesichts ist das Morphing. Beim Morphing entsteht aus zwei Bildern durch Überblendung ein neues Bild. Dabei wird die Überblendung durch einen Parameter  $\alpha$  gesteuert, welcher einen Wert zwischen 0 und 1 annimmt. Wenn  $\alpha = 0$  bei zwei gegebenen Bildern I und J ist, entspricht das Endergebnis dem Bild I. Bei  $\alpha = 1$  entspricht das Endergebnis dem Bild J und bei  $\alpha = 0.5$  entsteht eine gleichmäßige Überblendung der Bilder I und J. Diese Überblendung wird für jedes Pixel in den Bildern angewandt. Für jedes Pixel in Bild I muss das Gegenpixel in Bild J gefunden werden. Die neu entstandene Pixelposition wird mit folgender Gleichung berechnet [5]:

$$x_m = (1 - \alpha)x_i + \alpha x_j \quad (1)$$

$$y_m = (1 - \alpha)y_i + \alpha y_j \quad (2)$$

---

Danach kann durch eine affine Transformation eine Rotation, Verschiebung und Skalierung auf die Delaunay Dreiecke angewendet werden. So kann eine Beziehung von zwei Bildern herausgefunden werden. Die affine Transformation wird berechnet, um die Dreiecke aus Bild I auf das gemorphte Bild abbilden zu können [6]. Sodurch können alle Pixel in den Dreiecken aus beiden Bildern in das gemorphte Bild transformiert werden. Über folgende Gleichung wird die Intensität der Pixel festgelegt [5]:

$$M(x_m, y_m) = (1 - \alpha)I(x_i, y_i) + \alpha J(x_j, y_j) \quad (3)$$

Über die neuen Pixelpositionen wird die affine Transformation angewandt, indem die Delaunay Dreiecke aus Bild 1 und Bild 2 den neu berechneten Dreieckspunkten transformiert werden. Das gemorphte Bild der zwei Gesichter ist in Abbildung 5 zu sehen. Ohne die affine Transformation und Verzerrung der Dreiecke, würde eine einfache Überblendung kein neues Gesicht erstellen.

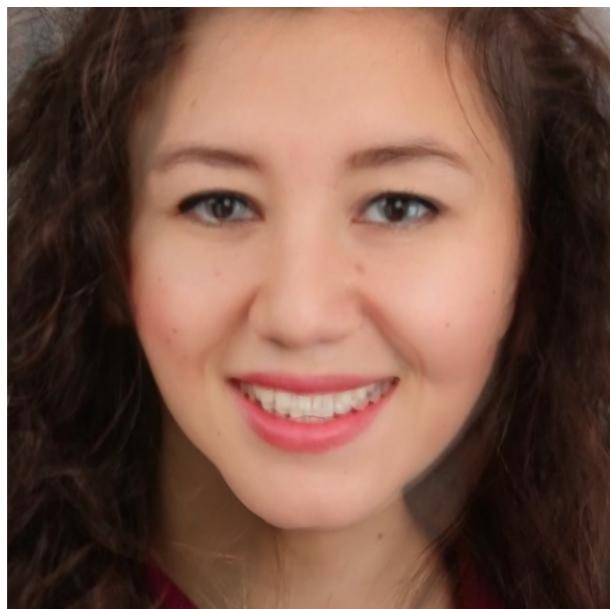


Abbildung 5: Morphing beider Bilder (Quelle: Eigene Darstellung)

## 6 Ergebnis

Die Ermittlung des Durchschnittsgesichts wurde mit den Methoden der Facial Landmarks, Delaunay Triangulation und dem Face Morphing im Projekt angewandt. Eine PCA konnte für ein Bild angewandt, aber nicht für die Ermittlung des Durchschnittsgesichts verwendet werden. Der Code kann für eine beliebige Anzahl an Bildern angewandt werden für die Ermittlung eines Durchschnittsgesichts. In Abbildung 6 sind die zwei Beispielbilder und das Ergebnis dargestellt.

Der Link zum GitHub-Repository, in welchem das Projekt hinterlegt ist: [https://github.com/lnurtaev/mean\\_face](https://github.com/lnurtaev/mean_face)

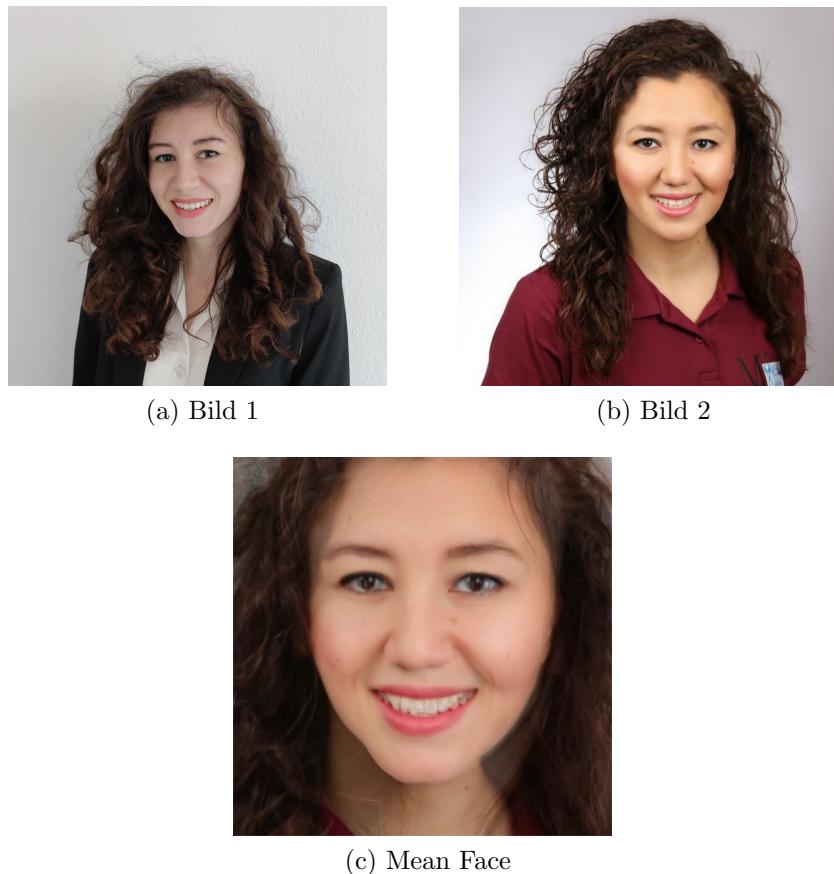


Abbildung 6: Durchschnittsgesicht aus beiden Bildern (Quelle: Eigene Darstellung)

## Literatur

- [1] N. Boyko, O. Basystiuk, and N. Shakhovska, “Performance evaluation and comparison of software for face recognition, based on dlib and opencv library,” in *2018 IEEE Second International Conference on Data Stream Mining Processing (DSMP)*, 2018, pp. 478–482.
- [2] P. Korshunov and S. Marcel, “Speaker inconsistency detection in tampered video,” 09 2018, pp. 2375–2379.
- [3] S. Mallick. (2015, 11) Delaunay triangulation and voronoi diagram using opencv ( c++ / python ). Accessed: 2022-02-03. [Online]. Available: <https://learnopencv.com/delaunay-triangulation-and-voronoi-diagram-using-opencv-c-python/>
- [4] ——. (2018, 1) Eigenface using opencv (c++/python). Accessed: 2022-02-04. [Online]. Available: <https://learnopencv.com/eigenface-using-opencv-c-python/>
- [5] ——. (2016, 3) Face morph using opencv — c++ / python. Accessed: 2022-02-04. [Online]. Available: <https://learnopencv.com/face-morph-using-opencv-cpp-python/#id1444128263>

- 
- [6] OpenCV. (2022, 2) Affine transformations. Accessed: 2022-02-04. [Online]. Available: [https://docs.opencv.org/3.4/d4/d61/tutorial\\_warp\\_affine.html](https://docs.opencv.org/3.4/d4/d61/tutorial_warp_affine.html)