TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Mensch-Maschine-Kommunikation

Blickrichtungsunabhängige Erkennung von Personen in Bild- und Tiefendaten

Andre Störmer

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr.-Ing. Thomas Eibert
Prüfer der Dissertation:	1. Univ.-Prof. Dr.-Ing. habil. Gerhard Rigoll
	2. Univ.-Prof. Dr.-Ing. Horst-Michael Groß, Technische Universität Ilmenau

Die Dissertation wurde am 16.06.2009 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 30.10.2009 angenommen.

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über
http://dnb.d-nb.de abrufbar.

ISBN 978-3-86853-297-5

Sternstr. 18, 80538 München
Tel.: 089/66060798
www.dr.hut-verlag.de

1. Auflage 2009

Vorwort

Die vorliegende Arbeit ist das Ergebnis meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Mensch-Maschine-Kommunikation an der Technischen Universität München. Dem Leiter dieses Lehrstuhls, Herrn Prof. Dr.-Ing. habil. Gerhard Rigoll, gilt mein außerordentlicher Dank für die wunderbare Möglichkeit sich intensiv mit den vorgestellten Themen, auch im Rahmen interessanter Projekte, zu befassen. Ein sehr angenehmes Arbeitsklima, wertvolle Anregungen bei der Definition wissenschaftlicher Zielsetzungen, entgegengebrachtes Vertrauen und viel Freiraum für die Bearbeitung und Auswahl der vorgestellten Methoden, sorgen dafür, dass ich mich gerne an diese Zeit erinnern werde.

Ebenso gilt mein herzlicher Dank Herrn Prof. Dr.-Ing. Horst-Michael Groß, Leiter des Fachgebiets Neuroinformatik und Kognitive Robotik an der Technischen Universität Ilmenau. Da ich einen entscheidenden Anteil meines Werdeganges in Ilmenau verbracht habe, und meine Diplomarbeit in seinem Fachgebiet abgeschlossen habe, wo ich mich nicht nur fachlich, sondern auch menschlich sehr wohl gefühlt habe, wurden viele Grundsteine dieser Arbeit in Ilmenau gelegt.

Weiterhin möchte ich mich bei all meinen Kollegen bedanken. Neben den vielen positiven menschlichen Aspekten möchte ich mich insbesondere auch für den fachlichen Austausch in angeregten Diskussionen bedanken. Besonderer Dank gilt hier meinem langjährigen Zimmerkollegen Sascha Schreiber für nahezu tägliche Diskussionen und gemeinsam bearbeitete Forschungsthemen. Des weiteren danke ich F. Wallhoff für die Förderung in den ersten Monaten am Lehrstuhl, D. Arsic für die vielen gemeinsam bearbeiteten Projektthemen und den damit verbundenen Ausstausch, sowie M. Kaiser, J. Schenk und S. Schwärzler für die motivierenden Diskussionen, insbesondere während der Zeit des Verfassens dieser Arbeit.

Abschließend möchte ich meinen Eltern danken, die mich stets gefördert haben und meiner Freundin Grit, die mich immer unterstützt und mir stets Rückhalt gegeben hat.

München, November 2009 Andre Störmer

Kurzfassung

In dieser Arbeit werden vier verschiedene Probleme der Personenerkennung behandelt. Als Erstes wird eine Authentifikation auf frontalen Gesichtsbildern umgesetzt, die entscheiden soll, ob Passagiere, die ein Flugzeug betreten, die gleichen sind, die eingecheckt haben. Als Zweites werden Methoden zur Bearbeitung einer blickrichtungsunabhängigen Identifikation vorgeschlagen. Hier sollen Personen, egal welche Orientierungen ihre Köpfe gerade haben, erkannt werden. Danach wird aufgezeigt, wie ein Paarvergleich von Gesichtsbildern umgesetzt werden kann, bei dem für zwei Bilder entschieden werden soll, ob jeweils die gleiche Person zu sehen ist. Der letzte Anwendungsfall geht von Tiefendaten mehrerer Personen mit verschiedenen Blickwinkeln und Gesichtsausdrücken aus. Hier wird eine vollautomatische Verarbeitungskette erläutert, die die Identität der Personen feststellen kann.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1

Einleitung

Die computerbasierte Erkennung von Personen in digitalen Bildquellen hat in den vergangenen Jahren stetig an Bedeutung gewonnen. Durch ständige Weiterentwicklung von Bildaufnahmeverfahren und Datenverarbeitungstechnologien konnten wichtige Fortschritte bei der automatischen Auswertung von Bildern errungen werden. Viele Verfahren zur Erkennung von Gesichtern werden bereits in einem breiten Anwendungsspektrum genutzt, als Beispiele seien hier die Zutrittskontrolle in ihren verschiedensten Ausprägungen, sowie die videobasierte Überwachung öffentlicher Plätze genannt. Eine weitere weit verbreitete Anwendung ist die Gesichtserkennung bei der Einreise in diverse Länder aufgrund von Daten, die auf dem elektronischen Chip des Reisepasses gespeichert sind. Auch die automatische Fahndung, bei der Passanten öffentlicher Plätze gegen gesuchte Kriminelle verglichen werden oder zivile Sicherheitsfragen, bei denen biometrische Merkmale nach und nach klassische Zugangsberechtigungsmethoden, wie Schlüssel und Passwort ergänzen oder gar ersetzen, gewinnen immer mehr an Bedeutung.

Ein sinnvoller Einsatz dieser Technologien ergibt sich, zusätzlich zu dem klassischen Einsatz in Überwachungs- und Sicherheitsszenarien, auch in der Mensch-Maschine-Kommunikation. Hier ist es insbesondere von Interesse zu wissen, wer gerade mit der Maschine interagiert. Wenn dieses ermittelt werden kann, können nutzerspezifische Profile angelegt und abgerufen werden, um zum Beispiel personalisierte Bedienoberflächen aufzurufen. Gerade im Bereich der Mensch-Maschine Kommunikation bietet sich der Einsatz der Gesichtserkennung an, da die dazu notwendige Sensorik, also Kameras, bereits einen breiten Einzug in die typischen Anwendungsgebiete, zum Beispiel in Form von Webcams an Computerarbeitsplätzen, erreicht haben, und außerdem für viele andere Anwendungen sinnvoll genutzt werden können. Dies unterscheidet die Gesichtserkennung von vielen anderen erfolgreichen Identifizierungstechnolgien, wie Fingerabdruck-, Iris- oder Handvenenerkennung, in denen spezielle Sensoren eingesetzt werden, die oft nur diese eine Anwendung bedienen.

Der aktuelle Stand in dem Forschungsbereich Gesichtserkennung ist, dass die Aufgabe, frontale neutral schauende Gesichter unter guten Beleuchtungsbedingungen zu erkennen, als grundsätzlich gelöst gilt. Im Laufe der Jahre hat sich eine große Anzahl verschiedener Gesichtserkennungsansätze entwickelt, die in der Lage sind, sehr gute Ergebnisse auf solchen normierten Gesichtsbildern zu erzeugen. Interessanterweise haben sich bereits einige Folgen dieser Technologien in das alltägliche Leben eingeschlichen. Als bekanntes Beispiel sind hier die Fotorichtlinien für Reisepässe zu nennen, die sich an einen internationalen Standard anlehnen, der von vielen Nationen berücksichtigt wird. In Aufnahmen für Passbilder ist die Person frontal und emotionslos abzulichten, zusätzlich wird der Bildausschnitt, der das Gesicht enthalten darf, genau definiert. Es ist zwar nicht bekannt, welche Algorithmen von den einzelnen Nationen zur Erkennung herangezogen werden, grundsätzlich ist aber anzumerken, dass die meisten Gesichtserkennungsalgorithmen von einer solchen strikten Normierung profitieren und somit bessere Erkennungsergebnisse liefern.

Das Ziel der aktuellen Forschung im Bereich Gesichtserkennung ist es allerdings, robuste Algorithmen zu entwickeln, die die Erkennungsaufgabe, ein Gesicht zu identifizieren auch bei nicht-frontalen Aufnahmen und nicht normierten Beleuchtungsverhältnissen löst. Diese Aufgabenstellung ist stark an den realen Einsatz von Kamerasystemen in unterschiedlichen Umgebungen angelehnt. Typische Überwachungskameras sind nur selten auf "Augenhöhe" von Passanten angebracht sondern meist so weit oben wie möglich, mit einem steilen Blickwinkel auf die Szene. Trotzdem möchte man die Daten automatisch auswerten können und Aussagen über die Identität von Personen treffen. Beleuchtungsbedingungen sind nur in komplett geschlossenen Räumen zu kontrollieren, oder nur auf sehr eng begrenztem Raum. Normalerweise wird dieses durch aktive Beleuchtungskomponenten erzielt, die natürlich Kosten verursachen. In Szenen, die sich außerhalb von Gebäuden abspielen, ist das Tageslicht ein nicht zu kontrollierender Einflussfaktor und einige Anwendungen gehen von einer bewegten Kamera aus, so dass sich die Szene und damit auch die Beleuchtungsbedingungen ständig ändern. Eine algorithmische Herangehensweise an die durch Blickwinkel und Beleuchtung entstehenden Probleme ist also erforderlich. Grundsätzlich besteht die Aufgabe also darin, die bereits im Labor funktionierenden Ansätze auch in der Realwelt einsetzen zu können. Um dieses zu erreichen, werden bestehende Verarbeitungsschritte analysiert und angepasst oder ersetzt.

Als Erstes gilt es also, sich zunächst einen Überblick über die verschiedenen bereits zur Verfügung stehenden Verfahren zu verschaffen.

1.1 Was ist Gesichtserkennung?

Zunächst gilt es, die Aufgabenstellung detailliert zu formulieren und zu konkretisieren, welches Ziel erreicht werden soll. In dieser Arbeit wird der Fragestellung nachgegangen, menschliche Gesichter, die durch bildgebende Sensoren (hier Grauwert- und Farbkameras, sowie 3D Scanner / Tiefenkamera) erfasst wurden, in Bezug auf ihre Identität zu beurteilen. Diese Aufgabe kann, je nach Stabilität der Umgebungs- und Aufnahmebedingungen mehr oder weniger anspruchsvoll sein. Einflussfaktoren, die eine erhebliche Rolle bei der Bearbeitung dieser Thematik haben, sind:

- Beleuchtung
  Bei jeder Bildverarbeitungsaufgabe spielen die Beleuchtungsbedingungen eine erhebliche Rolle. Grundsätzliche Schwierigkeiten bei der Gesichtserkennung entstehen insbesondere dann, wenn keine homogene Beleuchtung vorliegt, sondern starke Beleuchtungsgradienten, Schatten oder Überbelichtung für unterschiedliche Grundhelligkeiten in den verschiedenen Regionen des Gesichtes sorgen.

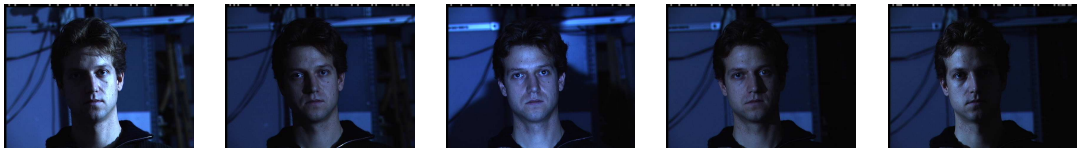


Abbildung 1.1: Die Beleuchtungsbedingungen sorgen für Unterschiede in der Wahrnehmung des gleichen Gesichts (PIE-Datenbank [SBB03]).

- Kopfpose
  Die geometrische Orientierung des Kopfes in einer gegebenen Ansicht stellt ebenfalls eine Herausforderung da. Bei variabler Kopfpose ergibt sich die zusätzliche Aufgabenstellung, festzustellen, welche Kopf- oder Gesichtsregionen miteinander korrespondieren, also miteinander verglichen werden müssen. Diese Problematik wird oft ausgeblendet, indem vorausgesetzt wird, nur mit frontalen Gesichtsaufnahmen zu arbeiten.

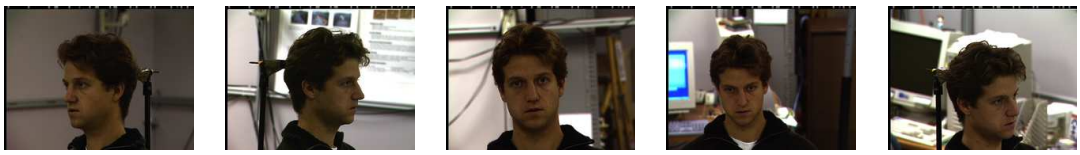


Abbildung 1.2: Unterschiedliche Kopfposen (PIE-Datenbank [SBB03]).

- Styling (Kosmetik, Frisur, Bartwuchs, Brille)
  Änderungen im Styling einer Person wirken sich natürlich direkt auf ihr äußeres Erscheinungsbild aus. Sie stellen damit eine klassische Störquelle bei der Gesichtserkennung dar, insbesondere wenn ein Anwendungsfall vorliegt, bei dem gespeicherte Datensätze über einen längeren Zeitraum verwendbar sein sollen.

Abbildung 1.3: Unterschiedliches Styling, verschiedene Frisuren, mit oder ohne Sonnenbrille und Schal (aus AR-Datenbank [MB98]).

- Verdeckung
  Je nach Anwendungsfall kann es vorkommen, dass Objekte, andere Personen oder Teile der Kleidung das zu bearbeitenden Gesicht teilweise verdecken. Wenn dieses der Fall ist, muss die Verarbeitungskette eine gewisse Robustheit gegenüber dieser Störung vorweisen.

- Mimik
  Der Mensch ist in der Lage, verschiedenste Gesichtsausdrücke zu erzeugen. Diese bewirken eine deutliche Änderung des optischen Eindrucks, selbst, wenn Aufnahmen der gleichen Person vorliegen. Hier gilt es, die Einflüsse von Mimik dahingehend zu untersuchen, welche Gesichtsregionen weitgehend invariant gegenüber Mimikveränderung sind, oder ob eine elastische Modellierung des Gesichtes hilfreich sein kann.

Abbildung 1.4: Verschiedene Gesichtsausdrücke (PIE-Datenbank [SBB03]).

- Sonstiges
  Es gibt viele weitere Einflüsse, die es von Fall zu Fall zu berücksichtigen gilt. Die Komplexität des Hintergrundes spielt bei vielen Algorithmen eine Rolle. Weiterhin gilt es je nach Anwendungsfall die Fragestellung zu berücksichtigen,

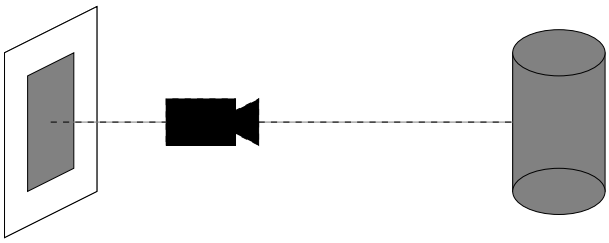


Abbildung 1.5: Die Aufnahme eines Bildes entspricht der Projektion der dreidimensionalen Realität auf eine zweidimensionale Fläche, dieses bedeutet eine Informationsreduktion.

ob mehrere Gesichter gleichzeitig in einem Bild vorkommen können und ob Robustheit gegenüber dem menschlichen Alterungsprozess verlangt wird.

Natürlich treten diese Einflussfaktoren nicht immer getrennt von einander auf (wie es in den Beispielbildern der Fall ist, siehe Abbildungen 1.1, 1.2, 1.3 und 1.4), sondern können durchaus in den verschiedensten Kombinationen oder auch alle zugleich auftreten. Ziel dieser Arbeit ist es, verschiedene Lösungsansätze zu präsentieren, wie das Problem der Gesichtserkennung unter Auftreten diverser Einflussfaktoren, insbesondere der variablen Pose, bearbeitet werden kann.

Um diese doch sehr weite Aufgabenstellung technisch zu interpretieren, wird zunächst einmal überlegt, welches die Eingangsgrößen zu einem System sind, welches dieses Problem bearbeiten soll. Ein bildgebender Sensor ist dadurch charakterisiert, dass er ein zweidimensionales Feld von ein- oder mehrkanäligen Messgrößen liefert. Diese sind typischerweise entweder Grauwerte (ein Kanal), Farbwerte (drei Kanäle) oder Tiefenwerte (ein Kanal). Kombinationen von texturbeschreibenden Kanälen (Grau oder Farbwerte) und Tiefenwerten sind möglich. Das gemessene zweidimensionale Feld wird als Bild $I(x, y)$ bezeichnet, es hat eine gegebene Auflösung, die angibt wieviele Bildelemente, genannt Pixel, es in Breite und Höhe hat. An jeder Position (x, y) im Bild I hat dieses Bild pro Kanal einen konkreten diskreten Messwert, es handelt sich also um ein orts- und wertdiskretes zweidimensionales Signal. Ein Bild $I(x, y)$ ist dabei die zweidimensionale Projektion einer dreidimensionalen Realität (siehe Abbildung 1.5), aus diesem Zusammenhang ergibt sich die Schwierigkeit, komplexe Objekterkennungsaufgaben zu lösen, da viele geometrische Objekteigenschaften durch die Projektion auf eine zweidimensionale Fläche verloren gehen.

In der folgenden Sektion wird nun betrachtet, wie die einzelnen Schritte der Verarbeitungskette für die Aufgabe Gesichtserkennung beschrieben werden können.

Abbildung 1.6: Die Verarbeitungskette vom Sensor bis hin zur Entscheidung in der Erkennerstufe.

1.2 Die Verarbeitungskette

Eine aktuelle Definition der Verarbeitungskette zur Erkennung von Gesichtern besteht aus den typischen Schritten Detektion, Ausrichtung und Erkennung, und wird mit dem Fachbegriff "Detection-Alignment-Recognition pipeline" (DAR-pipeline) beschrieben (siehe Abbildung 1.6). Diese Definition der DAR-pipeline ist an den aktuellen Stand der Technik angelehnt, und beschreibt die Aufgabe, eine vollautomatische Verarbeitung vom Sensor bis hin zur Entscheidung zu erzielen. Die einzelnen Bestandteile dieser Kette werden nun im einzelnen genauer beschrieben. Vorverarbeitungsschritte werden hier nicht als eigene Stufe angesehen, sondern als integraler Bestandteil jeder einzelnen Stufe. Die Leistung jeder dieser einzelnen Verarbeitungsschritte wird typischerweise durch objektive Maße bewertet, für Interessierte sei hier auf den Anhang A verwiesen.

1.2.1 Detektion

Die Detektion von Gesichtern beschreibt die Aufgabe, zu bestimmen, ob und wo sich Gesichter im Bild befinden. Im Allgemeinen werden in diese Verarbeitungsstufe Sensordaten eingegeben. Diese werden nach Anwendung klassischer Vorverarbeitungsschritte mit dem eigentlichen Detektionsalgorithmus ausgewertet. Als Ergebnis liegen typischerweise die Koordinatenangaben zu den Regionen vor, in denen sich Gesichter befinden. Auf die gegebene Definition des Bildes $I(x, y)$ als zweidimensionales Signal angewendet, bedeutet dieses, dass diejenigen Anteile des Signals gefunden werden, die für die Aufgabe Gesichtserkennung von Bedeutung sind. Das ist in der Regel der Ausschnitt, der das Gesicht selbst abbildet. Falls eine direkte Detektion eines Gesichtes je nach Aufgabenstellung nicht möglich ist, kann die interessierende Region auch der Kopf oder der Oberkörper einschließlich Kopf sein.
Eine gängige Beschreibungsform dieser Regionen ist ein umschließendes Rechteck (engl. "bounding box") mit $x \in [x_l, .., x_r]$ und $y \in [y_u, .., y_o]$, mit den linken und rechten Rändern x_l und x_r sowie den unteren und oberen Rändern y_u und y_o. Andere Beschreibungsformen verwenden die Koordinaten der vier Eckpunkte oder die Koordinaten des mittleren Punktes, sowie Breite und Höhe.
Ein Detektor ist oft nur in der Lage, grob zu lokalisieren, da, um den Suchaufwand möglichst gering zu halten, viele Freiheitsgrade nicht berücksichtigt werden. Typischerweise berücksichtigen Detektoren Skalierung und Translation nur in wenigen diskreten Stufen, Rotation wird oft ganz vernachlässigt. Alle diese Transformatio-

nen beschränken sich meistens auf die Bildebene.
Der durch den Detektor gefundene Ausschnitt liefert also eine grobe Lokalisation des weiter zu untersuchenden Objektes, hier des Gesichtes.

1.2.2 Ausrichtung

Die Ausrichtung (engl. "Alignment") beschreibt den Verarbeitungsschritt der Feinanpassung einer gefundenen Gesichtsregion an eine Referenz, somit das Erzeugen eines normierten Gesichtsauschnitts. Ausrichtungsschritte schätzen in der Regel die geometrischen Transformationen (Rotation, Verschiebung, Skalierung) und/oder perspektivische Abbildungen, um einen Ausschnitt optimal auf eine Referenz abzubilden. Diese Referenz ist entweder etwas Statisches, wie zum Beispiel der Durchschnitt aller Gesichter, die Referenz kann aber auch ein parametrisches Modell sein (siehe ASM, AAM). Wichtige Teilprobleme des Ausrichtens sind das Finden korrespondierender Bildpunkte oder Bildregionen und die Berechnung von Transformationen, um diese aufeinander abzubilden. Ziel ist es, durch Herausnormieren dieser Transformationen einen normierten Gesichtsausschnitt zu erhalten, der möglichst ausschließlich personenspezifische Merkmale enthält.

Für die klassische Gesichtserkennung auf Frontalbildern unter kontrollierten Umgebungsbedingungen reduziert sich dieser Verarbeitungsschritt zu einem Normieren des gegebenen Ausschnittes in Bezug auf Größe und Lage in der Bildebene. Je mehr Freiheitsgrade in Bezug auf Gesichtspose, Mimik und Beleuchtung in den Bildern zugelassen werden, desto anspruchsvoller wird dieser Verarbeitungsschritt. Typischerweise ist der Ausrichtungsschritt eng mit dem nächsten Schritt, der Erkennung, verknüpft, da er das Ziel hat, die gegebenen Daten an die Bedürfnisse des Erkenners anzupassen. In vielen Algorithmen der Gesichtserkennung ist dieser Schritt implizit enthalten.

1.2.3 Erkennung

Die Erkennung befasst sich mit der Thematik gegebene Eingangsgrößen zu klassifizieren, sie also einer bestimmten Klasse zuzuordnen. In der Thematik Gesichtserkennung unterscheidet man üblicherweise zwischen drei verschiedenen Aufgabenstellungen: Gesichtsidentifikation, Gesichtsauthentifikation und unabhängiger paarweiser Vergleich von Gesichtsbildern.

Gesichtsidentifikation

Bei der Identifikation liegt eine Datenbank zugrunde, in der Informationen über bekannte Gesichter hinterlegt sind. Üblicherweise gehört der Aufbau einer solchen

Datenbank mit zum Erkennungsalgorithmus und wird bei den Verfahren maschinellen Lernens auch als Training bezeichnet. Die Identifikation ist die Aufgabe, ein unbekanntes Gesicht demjenigen in der Datenbank zuzuordnen, welches ihm am ähnlichsten ist. Man bezeichnet dieses auch als Ausführung eines $1 : N$-Vergleiches, da aus N Klassen der Datenbank diejenige ausgewählt wird, welche am besten zu dem einen vorliegenden Gesichtsbild passt.

Gesichtsauthentifikation

Bei der Authentifikation geht man von dem Szenario aus, dass eine Person sich ausweist, also behauptet eine bestimmte Person zu sein, zu der bereits Informationen in der Datenbank hinterlegt sind. Nun wird überprüft, ob diese Behauptung stimmt, also ob die Person diejenige ist, für die sie sich ausgibt. Hier wird also ein $1 : 1$-Vergleich ausgeführt, also ein Gesicht wird mit einem in der Datenbank hinterlegtem Gesicht oder Gesichtsmodell verglichen und es wird entschieden, ob es das gleiche Gesicht ist oder nicht. Auch für die Authentifikation gibt es einen Trainingsprozess, eine Datenbank für in Frage kommende Personen wird vorher erstellt. Zusätzlich gehört hier zum Trainingsprozess, dass eine geeignete Schwelle auf einem Ähnlichkeits- oder Distanzmaß gefunden wird, ab der eine Person als authentifiziert gilt.

Paarweiser Vergleich von Gesichtsbildern

Beim paarweisen Vergleich (engl. “pair matching”), wird davon ausgegangen, dass zwei oder mehr Datensätze vorliegen, die völlig unbekannt sind. Es gilt zu entscheiden, ob diese Datensätze der gleichen Klasse angehören oder nicht. Diese Erkennungsaufgabe ist an typische Passkontrollszenarien angelehnt. So gibt es ein Passbild im Ausweis und aktuelle Bilder der sich ausweisenden Person. Es soll entschieden werden, ob die Person die gleiche ist, wie die auf dem Passbild. Im Gegensatz zu den beiden anderen Erkennungsaufgaben liegt aber kein klassenspezifisches Trainingsmaterial zugrunde, nur domänenspezifisches Trainingsmaterial, also allgemeingültiges Wissen über die Klasse Gesicht.

Die Aufgabe ähnelt der Authentifikation, unterscheidet sich aber dadurch, dass kein explizites klassenspezifisches Training unter bekannten Bedingungen vorgenommen werden kann, sondern dass die Aufnahmebedingungen, unter denen die Bilder erzeugt wurden, völlig unbekannt sind. Des weiteren spielt hier die Thematik des Lernens anhand nur eines Beispiels eine Rolle. Da nicht bekannt ist, welche Bilder der selben Klasse angehören, können keine Trainingsdatensätze zusammengefügt werden und so muss vor der Entscheidungsfindung davon ausgegangen werden, das jedes Bild eine andere Klasse beinhalten kann.

2

Überblick über bestehende Verfahren

In diesem Kapitel wird ein kurzer Überblick über gängige Verfahren aus den Bereichen Gesichtsdetektion, Ausrichtung und Gesichtserkennung gegeben. Hier werden allerdings nur die theoretischen Grundlagen erläutert, eventuelle Untersuchungen im Hinblick auf die Leistungsfähigkeit werden später in den Kapiteln vorgenommen, in denen der tatsächliche Einsatz dieser Algorithmen beschrieben wird oder in denen sie zum Vergleich herangezogen werden. Der Übergang zwischen Detektion, Ausrichtung und Erkennung ist oft fließend. Viele Algorithmen, die der Detektion oder Erkennung zugeordnet werden, enthalten Ausrichtungsaspekte, deshalb ist die hier vorgenommen Einteilung nicht als feste Kategorisierung zu verstehen, sondern eher als grobe Zuordnung.

2.1 Gesichtsdetektion

Gesichtsdetektion beschreibt die Aufgabe, in einem vorliegenden Bild die Ausschnitte zu finden, die Gesichter enthalten. Typische Repräsentationen der Ergebnisse sind sogenannte "bounding boxes", also die jeweilige Region umschließende Rechtecke. Die geläufigsten Verfahren zur Detektion von Gesichtern in Digitalbildern sind der auf AdaBoost und Haar ähnlichen Merkmalen basierende Ansatz von Viola und Jones [VJ01], sowie der Ansatz von Rowley [RBK98] der eine konkrete Struktur eines künstlichen neuronalen Netzes (Multi Layer Perceptron) anwendet, um Gesichter zu finden. Diese beiden Detektionsalgorithmen werden im Folgenden kurz vorgestellt. Ein Überblick über weitere verschiedene Ansätze kann man in [YKA02] finden. Neben den spezifisch für Gesichtsdetektion entwickelten Ansätzen, spielen hier natürlich auch die allgemeinen Detektionsansätze, wie Vorder- und Hintergrundsegmentierung durch Hintergrundsubtraktion, Detektion von bestimmten Farben im Bild oder Detektion von Bewegung durch zeitliche Differenzbildung, eine Rolle, die

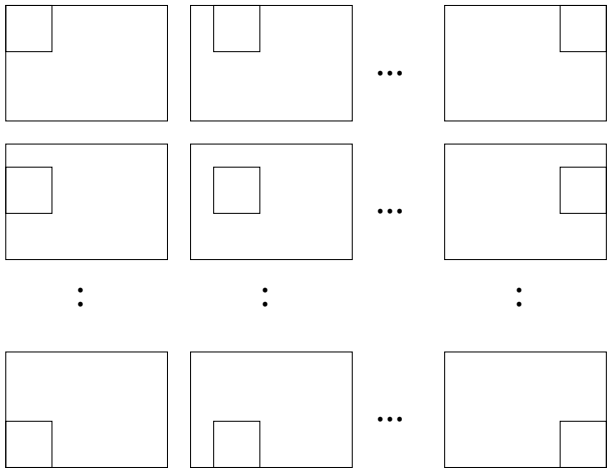


Abbildung 2.1: Prinzip des "sliding window". Durch ein sich über das Bild schiebendes Fenster werden alle möglichen Bildausschnitte gleicher Größe bearbeitet.

oft mit den hier aufgeführten Ansätzen kombiniert werden. An dieser Stelle sei auf [Jäh01], [AR05] und [Rus02] verwiesen, welches umfangreiche Standardwerke zur Bildverarbeitung sind.

2.1.1 Detektion mit Neuronalen Netzen

Bei diesem Detektionsansatz wird von einem Neuronalen Netz [Pat97] ausgegangen, das einen einzelnen Bildausschnitt zu den beiden Klassen "Gesicht" oder "kein Gesicht" zuordnet. Ein "sliding window", ein sich über das Bild schiebendes Fenster (siehe Abbildung 2.1), schneidet alle möglichen Bildausschnitte aus, die zunächst helligkeitsnormiert werden und anschließend einzeln klassifiziert werden. Dieses Prinzip wird mit einer Auflösungspyramide verknüpft (siehe Abbildung 2.2), um zusätzlich in der Lage zu sein, verschieden große Gesichter zu finden. Eine Auflösungspyramide ist dabei ein ursprüngliches Bild I und die k mit dem Faktor $f_n = s^n; n = 1, .., k$ abgetasteten Bilder $I_1, .., I_k$. Hierbei nimmt s einen Wert zwischen 1 und 2 an. Typischerweise werden die Bilder vor der Unterabtastung tiefpassgefiltert.

Nachdem nun geklärt ist, wie alle möglichen Ausschnitte verschiedener Größe extrahiert werden können, wird sich nun dem neuronalen Netz zugewendet, das die Ausschnitte klassifiziert. Hierbei handelt es sich um ein klassisches Multi Layer Perceptron, in diesem Fall mit einer Input-Schicht (Input Layer) von 20 mal 20 Elementen, einer Hidden-Schicht (Hidden Layer) und einem Output-Neuron, das die Entscheidung "Gesicht" oder "kein Gesicht" repräsentiert. Als effektive Architektur für das Neuronale Netz hat sich die von Henry A. Rowley [RBK98][Row99] herausge-

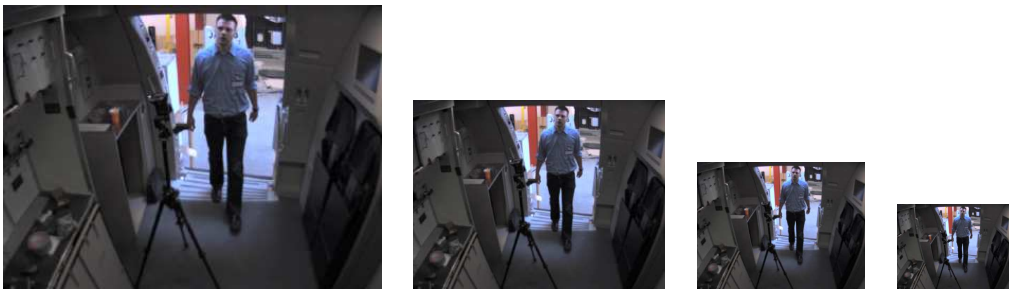


Abbildung 2.2: Auflösungspyramide: Das ursprüngliche Bild wird in verschieden Größen skaliert.

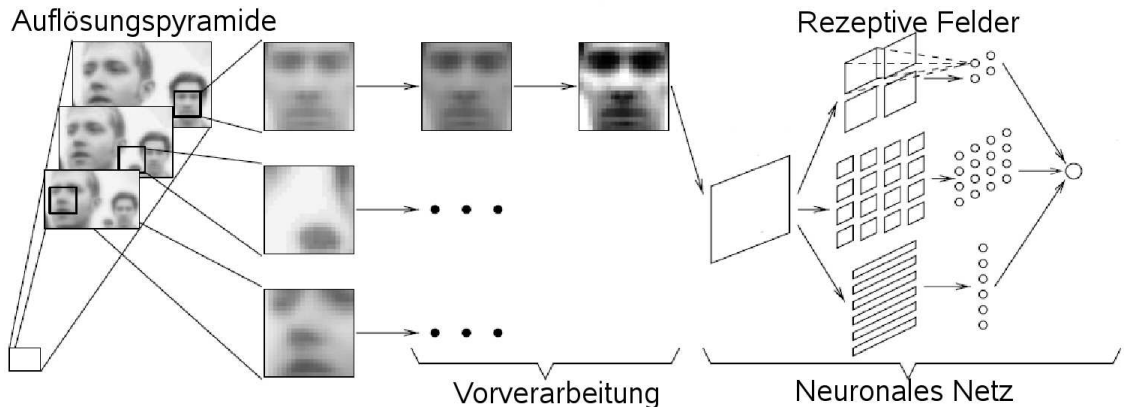


Abbildung 2.3: Gesichtsdetektion mit neuronalen Netzen (aus [RBK98]).

stellt, der eine konkrete Struktur von rezeptiven Felder vorschlägt (siehe Abbildung 2.3). Es werden drei verschiedenen Formen von rezeptiven Feldern verwendet, wobei jedes rezeptive Feld mit einem Neuron der Hidden-Schicht verbunden ist. Ein rezeptives Feld ist dabei einfach eine bestimmt geformte Untermenge der Input-Schicht. Die verschiedenen rezeptiven Felder, die genutzt werden, sind 4 jeweils 10×10 Pixel große Quadrate, 16 Quadrate der Größe 5×5 Pixel und 6 sich teilweise überlappende horizontale Streifen von 20×5 Pixeln. Der Grundgedanke ist, dass jedes dieser rezeptiven Felder auf bestimmte Strukturen hindeuten kann, die für die Gesichtserkennung von Interesse sind. So können die streifenförmigen Felder auf Augen- oder Mundregionen schließen lassen, während die quadratischen Felder eher auf einzelne lokale Merkmale wie Augen, Nase oder Mundecken hindeuten.

Das verwendete Neuronale Netz wird anhand manuell selektierter Beispielgesichter trainiert. Jedes dieser Bilder wird an Augen, Nase und Mund markiert. Anhand dieser Positionen wird ein Ausschnitt erzeugt und auf eine fixe Größe normiert. Anschließend werden aus jedem dieser Bilder 15 zusätzliche Trainingsbilder erzeugt,

indem rotiert, verschoben, skaliert und gespiegelt wird. Als Negativbeispiele werden zunächst unzählige zufällige Ausschnitte aus Bildern, die keine Gesichter enthalten, verwendet. Alle Ausschnitte, die das neuronale Netz als Eingabe erhält, werden vorher helligkeitsnormiert und ein Histogrammausgleich wird durchgeführt. Als Trainingsverfahren wird Backpropagation mit Trägheitsterm (standard error backpropagation with momentum) [HKP91] verwendet, eine grundlegende Erklärung dieses Verfahrens findet sich im Anhang C. Das Training selbst findet nun in mehreren Iterationen statt, zunächst werden neben den Positivbeispielen die bereits erwähnten zufälligen ausgewählten Negativbeispiele verwendet. Nachdem das Training konvergiert ist, wird das trainierte neuronale Netz als Detektor auf Bilder angewendet, die keine Gesichter enthalten. Alle falsch positiven Detektionen werden den Negativbeispielen hinzugefügt. Dieses Verfahren wird mehrfach angewendet, um nach und nach möglichst wenig falsch positive Detektionen zu erzielen.

Ein weiterer Beitrag Rowleys war es, sogenannte "Merging" Strategien zu entwickeln, also Verfahren um detektierte Gesichtshypothesen im Bild über Nachbarschaftsbeziehungen in Ort und Skalierung zu einzelnen detektierten Gesichtern zusammenzufassen. Ein einfacher Ansatz, basierend auf einem einzelnen neuronalen Netz, ist es, die Anzahl sich in einer bestimmten Nachbarschaft in Ort und Skalierung überlappender Detektionen zu zählen. Wenn die Anzahl über einer vordefinierten Schwelle ist, werden alle Hypothesen gemittelt und diese Region gilt als detektiertes Gesicht. Falls die Schwelle nicht überschritten wird, werden die Hypothesen verworfen und somit als Fehldetektion interpretiert. Dieses Prinzip wird "overlap eliminiation" genannt.
Eine andere Möglichkeit besteht darin, mehrere neuronale Netze auf ähnliche Art und Weise zu trainieren, allerdings unterscheidet sich die zufällige Initialisierung und im Trainingsprozess wird eine zufällige Auswahl der Trainingsbeispiele eingeführt. Nun können die Detektionsergebnisse der einzelnen Netze verknüpft werden, dies geschieht zum Beispiel durch logische UND Verknüpfung der mit den verschiedenen neuronalen Netzen detektierten Hypothesen.

Insgesamt stellt die Gesichtsdetektion nach Rowley ein realzeitfähiges System zur Detektion von Gesichtern dar, das inzwischen in vielen praktischen Anwendungen zum Einsatz kommt.

2.1.2 Detektion durch eine Kaskade von Klassifikatoren auf Haar ähnlichen Merkmalen

Die Gesichtsdetektion mittels AdaBoost und Haar ähnlichen Merkmalen wurde von P. Viola und M. Jones (siehe [VJ04] und [VJ01]) eingeführt. Neben dem eigentlichen Detektionsvorgang, also dem Klassifizieren aller möglichen Ausschnitte, ob

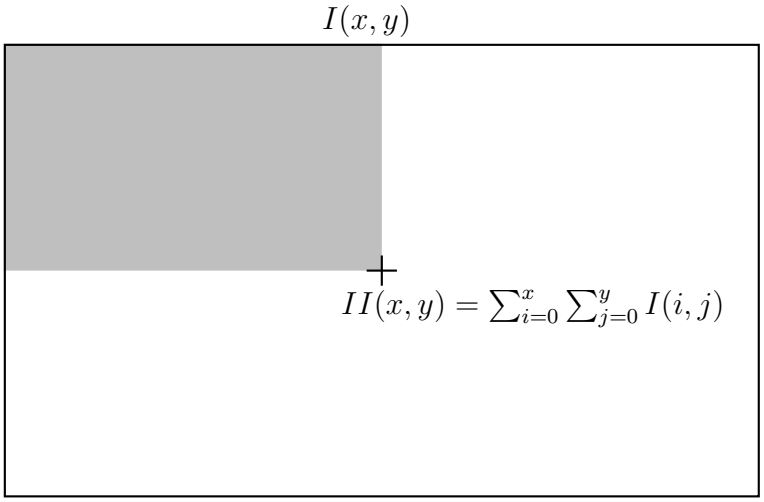


Abbildung 2.4: Berechnung des Integralbildes: Die Summe aller Pixel im Bild I innerhalb des Rechtecks von der linken oberen Ecke $(0, 0)$ bis zur Position (x, y) ist der Wert des Integralbildes II an der Stelle (x, y)

sie der Klasse Gesicht angehören oder nicht, wurde dieser Ansatz auch durch die äußerst effektive Berechnung der Merkmale populär. Diese Berechnung basiert auf sogenannten Integralbildern, die es ermöglichen schnell und effektiv die Summe einer beliebigen rechteckigen Fläche innerhalb eines Bildes zu berechnen. Um das Integralbild II zu erzeugen, wird die Summe aller Pixelwerte innerhalb des Rechtecks von der linken oberen Ecke bis zur Pixelposition (x, y) aus dem Originalbild I berechnet und an die Position (x, y) im Integralbild II geschrieben (siehe Abbildung 2.4).

$$II(x, y) = \sum_{i=0}^{x} \sum_{j=0}^{y} I(i, j) \tag{2.1}$$

Um die Berechnung effektiv zu gestalten, will man verhindern, dass für jeden Pixel neu über das jeweilige Rechteck aufsummiert werden muss. Da in den linken und oberen Nachbarpixeln im Integralbild bereits Teilsummen des aktuellen zu berechnenden Rechtecks sind, können diese genutzt werden. Wenn der Aufsummierungsprozess in zwei Teilschritte zerlegt wird, in spaltenweises und zeilenweises Aufsummieren, kann ein besonders effektiver Weg zur Berechnung des Integralbildes formuliert werden, der jeden Pixel insgesamt nur zweimal verrechnen muss, um das komplette Integralbild zu erstellen. Dies geschieht, indem man zunächst kumulative Zeilensummen ZS berechnet und anschließend die kumulativen Zeilensummen spaltenweise aufaddiert.

$$ZS(x, y) = \sum_{i=0}^{x} I(i, y) \tag{2.2}$$

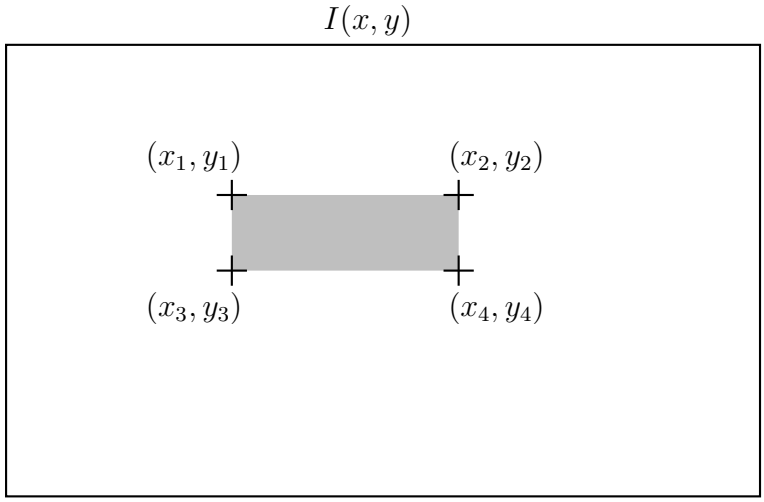


Abbildung 2.5: Berechnung der Summe alle Pixel in einem Reckeck: Die Werte im Integralbild II an den vier Eckpunkten werden miteinander verrechnet.

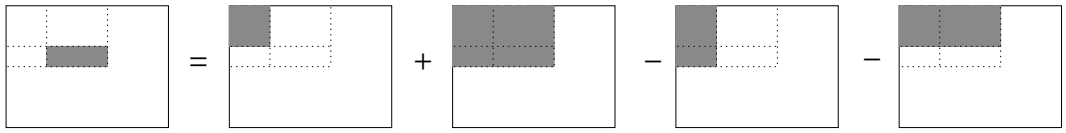


Abbildung 2.6: Berechnung einer Rechteckfläche durch Verrechnung der kumulativen Flächen der Eckpunkte

$$II(x, y) = \sum_{j=0}^{y} ZS(x, j) \tag{2.3}$$

Die Berechnung kumulativer Summen kann iterativ formuliert und somit effizient umgesetzt werden, so ist $ZS(x + 1, y) = ZS(x, y) + I(x + 1, y)$ und $II(x, y + 1) = II(x, y) + ZS(x, y + 1)$.

Ein Integralbild II wird berechnet, um die Summe der Pixel innerhalb eines beliebigen Rechtecks im Bild I unter Verwendung von möglichst wenig Rechenoperationen berechnen zu können. Wenn die Positionen der vier Eckpunkte $(x_i, y_i); i = (1, .., 4)$ bekannt sind, kann die Summe RS aller Pixel eines Rechtecks unmittelbar aus den Werten an den Positionen der Eckpunkte ermittelt werden, die im Integralbild gespeichert sind (siehe Abbildungen 2.5, 2.6 und Formel 2.4).

$$RS = II(x_4, y_4) + II(x_1, y_1) - II(x_2, y_2) - II(x_3, y_3) \tag{2.4}$$

Die effektive Berechnung der Pixelsumme innerhalb eines Rechtecks wird benötigt, um Haar ähnliche Merkmale ("haar like features") zu berechnen. Haar ähnliche Merkmale sind intuitiv ähnlich den allgemein bekannten Haar-Wavelets, welche sich

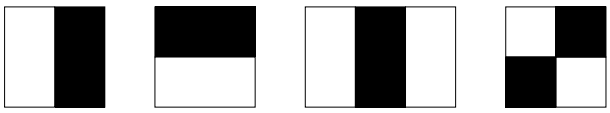


Abbildung 2.7: Typische Haar ähnliche Merkmale bestehend aus 2,3 oder 4 Rechtecken

aus der Theorie der orthogonalen Funktionen von Alfred Haar (siehe [Haa10]) ergeben, daher der Name. Ein einfaches auf Rechtecken basierendes Haar ähnliches Merkmal ist definiert als die Differenz der Pixelsummen innerhalb verschiedener rechteckiger Regionen, welche sich in beliebiger Größe und Position im Bild befinden können. Gängigerweise werden Merkmale basierend auf 2, 3 oder 4 rechteckigen Regionen verwendet. Die Werte, die bei der Berechnung eines Haar ähnlichen Merkmals herauskommen, beschreiben eine bestimmte Charakteristik des Bildes. So können zum Beispiel Kanten, Änderungen in der Textur oder die Grenze zwischen hellen und dunklen Bildregionen erfasst werden. Eine übliche graphische Darstellung der Haar ähnlichen Merkmale sind weiße und schwarze Rechtecke, die beschreiben welche Bildregionen positiv (weiß) und welche negativ (schwarz) gewichtet werden (siehe Abbildung 2.7). Der Merkmalswert f basierend auf einem Haar ähnlichen Merkmal nimmt also den Wert an, den die Differenz der Summen aller Pixel in den weiß dargestellten Regionen minus die Summe aller Pixel in den schwarz dargestellten Regionen ergibt.

Mittels eines Haar ähnlichen Merkmales i und eines Schwellwertes kann nun ein sogenannter schwacher Klassifikator h_i erzeugt werden, der entscheiden soll, ob es sich bei einem untersuchten Ausschnitt $\vec{x}$ um ein Gesicht handelt oder nicht. Hierbei wird der Merkmalswert $f_i(\vec{x})$ einfach mit einer Schwelle l_i verglichen, die aufgrund der vorhandenen Trainingsbeispiele so gewählt wurde, dass die Klassifikationsleistung maximal wird.

$$h_i(\vec{x}) = \begin{cases} 1 & \text{falls } p_i f_i(\vec{x}) < p_i l_i \\ 0 & \text{sonst} \end{cases} \tag{2.5}$$

Dieser schwache Klassifikator entspricht geometrisch interpretiert einer trennenden Hyperebene im Merkmalsraum. Der Vorzeichenfaktor p_i bestimmt dabei nur, auf welcher Seite der Hyperebene sich welche Klasse befindet. Da eine solche Klassifikation anhand eines einzelnen Merkmals kaum sehr aussagekräftig ist, werden viele dieser schwachen Klassifikatoren erzeugt und zu einem sogenannten starken Klassifikator zusammengefasst.

Die Methode, mit der aus mehreren schwachen Klassifikatoren ein starker Klassifikator erstellt wird, nennt sich Adaptive Boosting, oder kurz AdaBoost. AdaBoost ist ein maschinelles Lernverfahren, welches durch Yoav Freund und Robert Schapire in [FS95] formuliert wurde. Ziel ist es, einen starken Klassifikator zu finden, der sich

als Kombination von T unterschiedlich stark gewichteten schwachen Klassifikatoren h ergibt.

$$H(x) = \begin{cases} 1 & \text{falls } \sum_{t=1}^{T} \alpha_t h_t(\vec{x}) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{sonst} \end{cases} \tag{2.6}$$

Im Folgenden wird die leicht modifizierte Version des AdaBoost vorgestellt, die von Viola und Jones zur Erzeugung ihres Gesichtsdetektors verwendet wurde. Gegeben sind dabei n Trainingsbeispiele $\vec{x}_1, ..\vec{x}_n$ und die Klassenzugehörigkeiten $y_1, .., y_n$ mit $y_i \in \{1, 0\}$, die beschreiben, ob in den Trainingsbeispielen, in diesem Fall Bildausschnitte, Gesichter enthalten sind oder nicht. Es gibt m positive und k negative Trainingsbeispiele. Jedes positive Trainingsbeispiel ist zunächst gewichtet mit $w_{1,i} = \frac{1}{2m}$ und jedes Negative mit $w_{1,i} = \frac{1}{2k}$. Grundgedanke des Verfahrens ist es, nach jeder Trainingsiteration $t = 1, .., T$ falsch klassifizierte Trainingsbeispiele höher zu gewichten, und diese Gewichtung bei der Auswahl weiterer schwachen Klassifikatoren mit zu berücksichtigen. Der Gesamtalgorithmus läuft wie folgt ab:

Für $t = 1, .., t$:

- Normiere die Gewichte, so dass sie eine Wahrscheinlichkeitsverteilung ergeben:

$$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}} \tag{2.7}$$

- Trainiere einen schwachen Klassifikator h_j für jedes Merkmal j, und evaluiere den Klassifikationsfehler bezogen auf die aktuelle Gewichtung der Trainingsbeispiele.

$$\epsilon_j = \sum_i w_i \mid h_j(x) - y_i \mid \tag{2.8}$$

- Finde den schwachen Klassifikator h_t, für den ϵ_t minimal ist.

- Setze das Gewicht α_t mit dem der schwache Klassifikator in den starken Klassifikator eingeht:

$$\alpha_t = \log\left(\frac{1 - \epsilon_t}{\epsilon_t}\right) \tag{2.9}$$

- Gewichte alle Trainingsbeispiele neu:

$$w_{t+1,i} = w_{t,i}\left(\frac{\epsilon_t}{1 - \epsilon_t}\right)^{1-e_i} \tag{2.10}$$

  e_i ist 0 wenn das Beispiel $\vec{x}_i$ richtig klassifiziert wurde, 1 falls es falsch klassifiziert wurde. Richtig klassifizierte Beispiele werden also schwächer gewichtet, falsch klassifizierte somit durch die Normierung im nächsten Schritt implizit stärker.

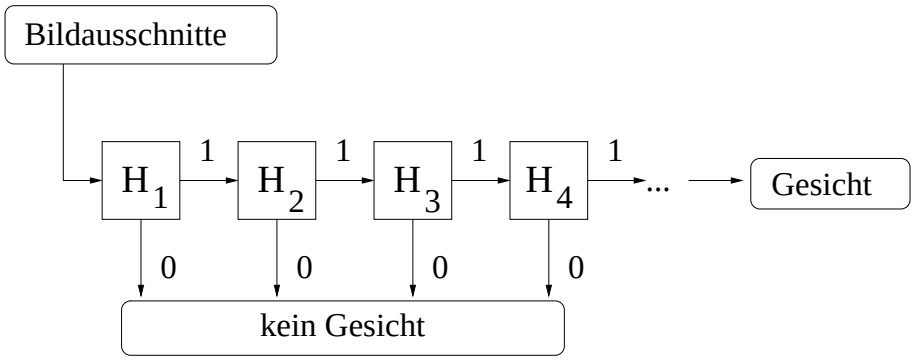


Abbildung 2.8: Kaskade von Klassifikatoren, es werden nur Ausschnitte weiterverarbeitet, die von der vorhergehenden Kaskadenstufe positiv (1) bewertet wurden, zurückgewiesene Ausschnitte (0) werden nicht weiter untersucht.

Nach Ende der Iterationen wird der starke Klassifikator $H(x)$ erzeugt:

$$H(x) = \begin{cases} 1 & \text{falls } \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{sonst} \end{cases} \tag{2.11}$$

Ein Detektor, der mit diesem Verfahren trainiert wird, erzeugt bereits sehr gute Ergebnisse. Allerdings geht die Anzahl der verwendeten Merkmale und somit der schwachen Klassifikatoren, die bei einem solchen Trainingsverfahren typischerweise recht hoch ist, linear in die Berechnungszeit ein. Um das zu verhindern, wurde aus diesem Basisalgorithmus eine Methode entwickelt, statt eines starken Klassifikators eine sogenannte Kaskade von Klassifikatoren zu entwickeln, die vom Grundprinzip stark an den Entscheidungsbaum angelehnt ist. Ziel ist es, höhere oder vergleichbare Detektionsperformanz bei gleichzeitiger Reduzierung der Rechenzeit zu erlangen.

Dieses wird realisiert, indem Zielvorgaben betreffend Detektionsrate und Falsch-Positiv-Rate gegeben werden. Das Ziel ist, die Detektionsraten der einzelnen Kaskadenstufen nahe 1 zu halten, gleichzeitig sollen möglichst niedrige Falsch-Positiv-Raten erzielt werden, die aber typischerweise durchaus in der Größenordnung von 40% sein können. Die Idee ist es, alle Ausschnitte, die Gesichter enthalten, durchzulassen, dabei aber möglichst viele Ausschnitte, die keine Gesichter enthalten, herauszufiltern. Spätere Stufen in der Kaskade können sich dann auf die übrigbleibenden Fälle spezialisieren. Ein Rechnenzeitvorteil wird dadurch erzielt, dass möglichst einfache Klassifikatoren, die aber viel aussortieren, möglichst frühe Kaskadenstufen einnehmen, und so überhaupt nur möglichst wenig Beispiele mehrere Kaskadenstufen durchlaufen. Ein Algorithmus zum Training dieser Kaskade kann wie folgt formuliert werden:

- Es werden Werte für fp, die maximal akzeptierbare Falsch-Positiv Rate pro Kaskadenstufe und für d, die minimale Detektionsrate pro Kaskadenstufe, vorgegeben
- Die insgesamt zu erzielende Falsch-Positiv Rate FP_{ziel} wird vorgegeben.
- P = Menge der positiven Trainingsbeispiele
- N = Menge der negative Trainingsbeispiele
- $FP_0 = 1.0$; $D_0 = 1.0$
- $i = 0$
- Solange $FP_i > FP_{ziel}$
  - $i \leftarrow i + 1$
  - $n_i = 0; FP_i = FP_{i-1}$
  - Solange $FP_i > fp \cdot FP_{i-1}$
    - $n_i = n_i + 1$
    - Benutze P und N um einen starken Klassifikator mit n_i schwachen Klassifikatoren unter Verwendung von AdaBoost zu trainieren.
    - Evaluiere die momentane Kaskade von Klassifikatoren um die aktuelle Gesamt-Falsch-Positiv Rate FP_i und die aktuelle Gesamtdetektionsrate D_i zu erhalten.
    - Verringere den Schwellwert innerhalb des i-ten Klassifikators bis die aktuelle Kaskade mindestens eine Gesamtdetektionsrate von $d \cdot D_{i-1}$ hat (erhöht gleichzeitig die Falsch-Positiv Rate).
  - $N \leftarrow \emptyset$, leere Negativbeispiele
  - Wenn $F_i > F_{ziel}$ evaluiere die aktuelle Kaskade von Klassifikatoren anhand der Bildausschnitte, die keine Gesichter enthalten. Füge alle Fehldetektionen den Negativbeispielen N hinzu.

Die Gesamtanzahl an Kaskadenstufen und die Anzahl der zu verwendenden schwachen Klassifikatoren werden durch den Anwender vorgegeben. In mehreren Tests wurde nachgewiesen, dass eine Kaskade von Klassifikatoren eine ungefähr gleiche Detektionsrate wie ein einzelner starker Klassifikator mit insgesamt gleicher Anzahl von schwachen Klassifikatoren erreicht, allerdings ist diese Kaskade wesentlich schneller. Dieses ist plausibel, da ja nur ein Bruchteil der untersuchten Ausschnitte alle Kaskadenstufen durchläuft.

Die Gesichtsdetektion durch Haar ähnliche Merkmale und eine Kaskade von Klassifikatoren ist eines der meist genutzten Verfahren und zeichnet sich besonders durch

die effiziente Rechenzeit aus. Hier ist nicht nur der Klassifikator hinsichtlich Rechenzeit optimiert, sondern auch die Merkmalsextraktion. Durch die effiziente Berechnung der Haar ähnlichen Merkmale im Integralbild, die jeweils nur die Eckpunkte der zu untersuchenden Rechteckflächen benötigt, kann auf eine Auflösungspyramide und auf ein "sliding window" verzichtet werden. Hier wird jeweils nur die Position der Eckpunkte der Rechteckflächen der Haar ähnlichen Merkmale verschoben und skaliert, damit können Umformungsschritte, die das ganze Bild betreffen, eingespart werden.

2.2 Ausrichtung

Der Ausrichtungsschritt ist sicherlich der Schritt, der in der gesamten Verarbeitungskette in den letzten Jahren am stärksten an Bedeutung gewonnen hat. Die beiden Aufgaben Detektion und Erkennung werden schon seit vielen Jahren untersucht, wurden in der Vergangenheit allerdings oft als Einzelprobleme betrachtet. Nicht selten wurden Erkennungsergebnisse auf ausschließlich manuell definierten Ausschnitten untersucht. Seitdem das Anwendungsziel aber immer mehr darin besteht, eine vollautomatische Verarbeitungskette vom Sensor bis hin zur Entscheidung zu erlangen, wird immer mehr verstanden, dass eine ordentliche maßgeschneiderte Anbindung von Detektor und Erkenner notwendig ist, um ordentliche Ergebnisse zu produzieren. Die eigentliche Aufgabe dieses Ausrichtungsschrittes, nämlich den Gesichtsausschnitt zu normieren, spielt besonders bei der blickrichtungsunabhängigen Erkennung eine Rolle, da hier der detektierte Gesichtsausschnitt auf eine frontale Ansicht ausgerichtet werden soll. In der Vergangenheit wurde dieser Schritt oft mit einer einfachen Skalierung auf eine bestimmte Bildgröße und einem einfachen Helligkeitsausgleich abgehandelt, es hat sich allerdings gezeigt, dass komplexere Herangehensweisen oft zu besseren Ergebnissen führen.

2.2.1 Active Shape Modell

Active Shape Modelle (ASM), frei übersetzt "Aktive Form Modelle", gehören zur Klasse der "Point Distribution Models" und wurden von T. F. Cootes eingeführt [CTCG95][CT93]. Sie beschreiben eine Form aufgrund der Verteilung charakteristischer Punkte. Üblicherweise wird diese Verteilung durch ein parametrisches Modell beschrieben. Die "aktive" Komponente des Modells ist die Fähigkeit, sich an ein gegebenes Muster durch Veränderung anzupassen, das heißt ein Matching Verfahren ist integraler Bestandteil des Modells.

Um ein Active Shape Modell zu erstellen wird ein Satz Trainingsdaten benötigt, in denen die charakteristischen, formbeschreibenden Punkte markiert sind. Typischerweise sind diese charakteristischen Punkte an Positionen, die eindeutig identifizierbar sind, zum Beispiel an Positionen an denen sich Kanten im Bild kreuzen

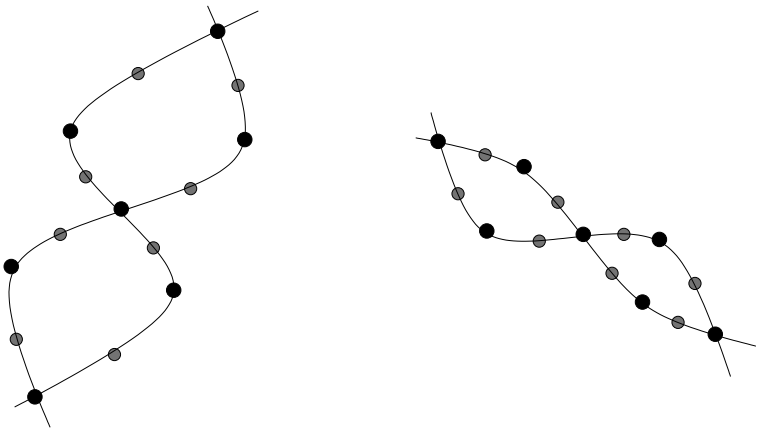


Abbildung 2.9: Beispielformen mit den darin enthaltenen Labelpunkten. Direkt identifizierbare Punkte aufgrund sich kreuzender Kanten oder lokalen Extrema in der Krümmung sind durch schwarze Punkte dargestellt. Zusätzliche Punkte, typischerweise Streckenteiler entlang Kanten, sind grau dargestellt.

oder eine besonders starke Krümmung aufweisen. Gerne werden zusätzliche Punkte gleichabständig zwischen zwei eindeutig identifizierbaren Punkten eingefügt, um eine präzisere Beschreibung der Form zu erhalten (siehe Abbildung 2.9). Die charakteristischen Punkte werden oft Labelpunkte oder Markierungspunkte genannt.

Ein Formdatensatz $\vec{s}$ aus den N Trainingsbeispielen wird im Folgenden durch einen Vektor $\vec{s} = (x_1, y_1, .., x_l, y_l)^T$, in dem die Koordinaten aller l Markierungspunkte enthalten sind, beschrieben. Um nun aus den Trainingsbeispielen ein Formmodell zu erstellen, wird zunächst ein Normierungsschritt vorgenommen, der aus allen Trainingsdaten $\vec{s}_i$ jeweils die Transformationen Rotation, Translation und Skalierung herausnormiert. Ein iterativer Algorithmus, der abwechselnd die mittlere Form $\overline{\vec{s}}$ berechnet, und anschließend alle Datensätze mittels der Methode der kleinsten quadratischen Fehler auf diese mittlere Form ausrichtet, kann wie folgt formuliert werden.

- Berechne die Mittlere Form $\overline{\vec{s}} = \sum_{i=1}^{N} \frac{\vec{s}_i}{N}$
- Berechne für jeden Datensatz $\vec{s}_i$ die Rotation, Translation und Skalierung, die den Fehler $e_i = |\overline{\vec{s}} - \vec{s}_i|^2$ minimiert, so dass $\overline{\vec{s}} \approx R_s \vec{s}_i + \vec{t}$.
- Wende die Rotation, Translation und Skalierung auf $\vec{s}_i$ an: $\vec{s}_i \leftarrow R_s \vec{s}_i + \vec{t}$
- Wiederhole alle Schritte bis sich die mittlere Form $\overline{\vec{s}}$ nicht mehr signifikant von der der vorhergehenden Iteration unterscheidet.

Das Ermitteln der Rotation, Translation und Skalierung zwischen zwei Datensätzen wird oft auch als Prokrustes Analyse bezeichnet, auf eine mögliche Berechnung dieser

Transformation wird in Anhang B eingegangen. Nachdem nun alle Trainingsdaten ausgerichtet sind, kann das eigentliche Formmodell berechnet werden, indem die Statistik der Trainingsbeispiele ausgewertet wird. Dies geschieht durch eine Principal Compenent Analysis (PCA), eine Hauptkomponentenanalyse. Nachdem der Mittelwert, wie bereits beschrieben, berechnet wurde, wird zunächst A_s, die Matrix der N Mittelwert freien Formvektoren erstellt.

$$A_s = (\vec{s}_1 - \overline{\vec{s}}, \vec{s}_2 - \overline{\vec{s}}, \cdots, \vec{s}_N - \overline{\vec{s}}) \tag{2.12}$$

Anschließend wird die Kovarianzmatrix C_s ermittelt.

$$C_s = \frac{1}{N-1} A_s A_s^T \tag{2.13}$$

Als abschließender Schritt des Modellaufbaus werden nun die Eigenvektoren $\vec{e}_{s,i}$, sowie die dazugehörigen Eigenwerte $\lambda_{s,i}$ dieser Kovarianzmatrix C_s bestimmt. Anschließend werden die Eigenvektoren nach der Größe ihrer Eigenwerte sortiert, nur die k Bedeutendsten werden verwendet. Als Ergebnis liegen nun also der Mittelwert und die Hauptkomponenten, also die k bedeutendsten Eigenvektoren, vor. Als parametrisches Modell zur Beschreibung einer Form $\vec{s}$ erhält man :

$$\vec{s} = \overline{\vec{s}} + \sum_{i=1}^{k} w_{s,i} \vec{e}_{s,i} \tag{2.14}$$

Eine Form $\vec{s}$ wird also durch den Mittelwert $\overline{\vec{s}}$, sowie durch die Gewichtungsfaktoren $w_{s,i}$ der k bedeutendsten Formeigenvektoren $\vec{e}_{s,1}, .., \vec{e}_{s,k}$ beschrieben. Um die Gewichtungsfaktoren für eine bereits bekannte Form $\vec{s}$ zu ermitteln, wird diese in den durch die Eigenvektoren $\vec{e}_{s,i}$ aufgespannten linearen Unterraum projiziert:

$$\vec{w}_s = \begin{pmatrix} w_{s,1} \\ \vdots \\ w_{s,k} \end{pmatrix} = E_{s,k}^T (\vec{s} - \overline{\vec{s}}) \tag{2.15}$$

Hierbei ist $E_{s,k}$ die Matrix, die die k bedeutendsten Eigenvektoren als Spaltenvektoren enthält.

Nachdem nun geklärt ist, wie eine Form parametrisch beschrieben werden kann, wird sich nun der Aufgabenstellung zugewendet, das Formmodell an eine aktuelle Beobachtung anzupassen. Da das Modell ja aufgrund händisch markierter Punkte ermittelt worden ist, muss nun zusätzlich ein automatisch im Bild zu ermittelndes Kriterium gefunden werden, welches die Lage dieser Punkte bestimmt. Hier gibt es unterschiedliche Ansätze, wie z.B. Gradientenverläufe, saliente Punkte oder Texturinformationen. Im ursprünglichen Ansatz nach Cootes [CTCG95] wird eine iterative Matchingstrategie wie folgt beschrieben:

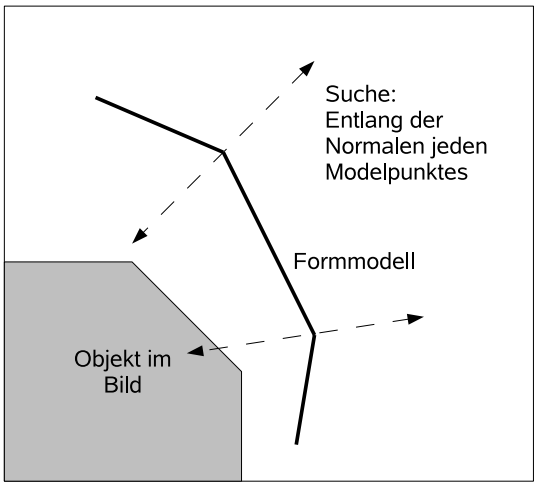


Abbildung 2.10: Suchschritt: Jeder Punkt wird an die Position der stärksten Kante entlang der Normalen verschoben

- Für jeden Punkt (x_i, y_i) der Form $\vec{s}$:
  - Finde die stärkste Kante entlang der Normalen des Formmodells, die durch den Punkt (x_i, y_i) geht (siehe Fig. 2.10).
  - Verschiebe den Punkt (x_i, y_i) zu der stärksten Kante entlang der Normalen
- Berechne Rotation, Translation und Skalierung zwischen der durch die neu gefundenen Punkten beschriebenen und der mittleren Form.
- Normiere die gefundenen Transformationen aus der Form heraus und beschreibe die daraus resultierenden Punkte mit dem Formmodell $\vec{s} = \overline{\vec{s}} + \sum_{i=1}^{k} w_{s,i}\vec{e}_{s,i}$.
- Wende einschränkende Bedingungen auf die Gewichtungsparameter $\vec{w}_s$ an, so dass $w_{s,i}$ im Bereich von $\pm 3\sigma_i$ der Beispieldaten ist.
- Wiederhole den kompletten Ablauf, bis Konvergenz erreicht ist.

Hier ist $\sigma_i = \sqrt{\lambda_{s,i}}$ die Standardabweichung der Trainingsdaten in Richtung des i-ten Eigenvektors $\vec{e}_{s,i}$. Diese kann unmittelbar aus den Eigenwerten bestimmt werden, da der Eigenwert $\lambda_{s,i}$ nichts anderes ist, als die Varianz in Richtung des dazugehörigen Eigenvektors $\vec{e}_{s,i}$. Um Konvergenz festzustellen, wird typischerweise die relative Veränderung zur vorhergehenden Iteration gemessen, wenn diese kleiner als eine vorgegebene Schwelle ist, wird der iterative Algorithmus abgebrochen. Eine andere, auch oft verwendete Methode ist es, eine feste Anzahl an Iterationen zu berechnen.

2.2.2 Iterative Closest Point Algorithmus

Der Iterative Closest Point Algorithmus (ICP) [BM92][RL01][ESE08][YB07] ist ein iterativer Ausrichtungsalgorithmus, dessen Ziel es ist, zwei ungeordnete Punktwolken durch eine geometrische Transformation aufeinander abzubilden. Hierbei unterscheidet man zwischen einem auszurichtenden Datensatz A und einem Referenzdatensatz B. Ziel ist es, eine Abbildungsfunktion zu finden, die A auf B abbildet. Der Begriff "ungeordnete" Punktwolke beschreibt dabei, dass keinerlei Sortierung der Punkte vorliegt und es nicht bekannt ist, welcher Punkt aus Datensatz A mit welchem Punkt aus Datensatz B korrespondiert. Hier liegt ein grundsätzlicher Unterschied zu der sogenannten Prokrustes Analyse (Procrustes Analysis) vor, bei der Punktkorrespondenzen bereits bekannt sind.

Beim ICP wird gängigerweise eine "rigide" Abbildung, also eine nicht formverändernde Transformation, die sich aus Rotation R und Translation $\vec{t}$ zusammensetzt, verwendet, teilweise wird auch noch eine Skalierung s hinzugenommen. Wenn nun angenommen würde, dass die innere Sortierung der Punktwolken $A = (\vec{a}_1, \vec{a}_2, ..., \vec{a}_k)$ und $B = (\vec{b}_1, \vec{b}_2, ..., \vec{b}_k)$ bekannt wäre, und man wüsste, dass jeweils $\vec{a}_i$ mit $\vec{b}_i$ korrespondierte, könnte man die Aufgabenstellung als Lösung des Gleichungssystems

$$B = sRA + \vec{t} \tag{2.16}$$

formulieren. Hier ist s ein Skalierungsfaktor, R die Rotationsmatrix und $\vec{t}$ ein Verschiebungsvektor. Da allerdings genau diese Punktkorrespondenzen nicht bekannt sind, versucht der Iterative Closest Point Algorithmus diese zu schätzen, indem er jedem Punkt in B den nächstgelegenen Punkt in A zuordnet. Als typisches Abstandsmaß wird der Euklidische Abstand verwendet.

Eine allgemeine Formulierung des ICP ist Folgende, gegeben seien die unsortierten Punktmengen A und B:

- Setze $A_0 = A$ für die erste Iteration $i = 0$
- Wiederhole
  - Sortieren: Finde für jeden Punkt in B den nächsten Punkt in A_i. Sortiere die Punkte in A_i um, so dass $\vec{a}_{i,j}$ der Punkt aus A_i mit dem niedrigsten Abstand zu $\vec{b}_j$ ist.
  - Auswählen: Erzeuge die Punktmengen A_i^* und B^*, die aus den n Punktpaaren mit den niedrigsten Abständen bestehen.
  - Löse das Gleichungssystem

    $$B^* = sRA_i^* + \vec{t} \tag{2.17}$$

    um Schätzungen für Skalierung s, Rotation R und Translation $\vec{t}$ zu erhalten.

– Wende die geschätzte Transformation an und erzeuge den Datensatz für die nächste Iteration

$$A_{i+1} = sRA_i + \vec{t} \tag{2.18}$$

– Wenn Konvergenz erreicht ist, breche ab, ansonsten gehe zur nächsten Iteration $i = i + 1$

Diese allgemeine Formulierung enthält mehrere offene Fragestellungen, die auf verschiedene Art und Weise beantwortet werden können. Beim klassischen ICP werden für alle Punkte in B korrespondierende Punkte aus A verwendet, somit ist n die Anzahl der Punkte in B. Beim Trimmed Iterative Closest Point Algorithmus [CSSK02] wird davon ausgegangen, dass die vorhandenen Punktwolken sich nur teilweise überlappen, somit wird n deutlich niedriger gewählt als die Gesamtanzahl der Punkte in B. Eine der Kernproblematiken beim ICP ist die konkrete Lösung des Gleichungssystems $B^* = sRA_i^* + \vec{t}$, welche eine Form des Prokrustes Problems, nämlich das Abbilden zweier geordneter Punktmengen aufeinander, darstellt. Typischerweise wird hier mit der Methode der kleinsten quadratischen Abweichung gearbeitet, es gibt allerdings eine ansehnliche Menge unterschiedlicher konkreter Ansätze, die dieses Problem bearbeiteten. An dieser Stelle sei auf die Arbeiten von Eggert u.a. und Fitzgibbon verwiesen [ELF97][Fit01], die verschiedene Lösungsansätze vorstellen. Die im Rahmen dieser Arbeit verwendete Quaternionenmethode nach Horn [Hor87] wird in Anhang B ausführlich beschrieben. Auch die Abbruchbedingung dieses Algorithmus, also die Frage, wann Konvergenz erreicht ist, kann auf verschiedene Art und Weise beantwortet werden. Varianten umfassen zum Beispiel eine festgelegte Anzahl an Iterationen oder die Erklärung von Konvergenz, wenn sich die Summe der Abstände zwischen allen korrespondierenden Punktpaaren nicht weiter verringert.

Einsatzgebiete des ICP

Der ICP hat sich vor allem als Standardalgorithmus bei der Vorverarbeitung von 3D-Scans durchgesetzt. Hierbei gibt es zwei typische Anwendungsfälle. Der Erste, das Ausrichten zweier Datenscans aufeinander, die örtlich verschiedene, sich aber überlappende Ausschnitte der gleiche Szene beinhalten, wird auch oft mit dem englischen Fachbegriff "Registration" versehen. Der zweite Anwendungsfall ist typischerweise in der Objekt- oder Personenerkennung beheimatet. Hier werden inhaltlich verschiedene Objekte oder Personen, also im Fall Gesichtserkennung verschiedene Individuen, auf eine Referenz ausgerichtet, um eine Richtungsnormierung zu erhalten. Der englische Fachbegriff für dieses ist "Alignment" und wird hier im Rahmen dieser Arbeit mit "Ausrichtung" übersetzt.

2.3 Erkennung

Aus der unzähligen Menge von verbreiteten Gesichtserkennungsverfahren werden hier nun diejenigen beschrieben, die im Laufe dieser Arbeit entweder als Referenz im Zuge experimenteller Vergleiche gedient haben, oder die als Basis für Veränderungen und somit als neue Varianten von Erkennern gedient haben. Neben den Verfahren "Eigenfaces", "Active Appearance Modelle" und "Pseudo 2D - Hidden Markov Modelle", die im Folgenden dargestellt werden, gibt es noch eine große Menge weiterer, durchaus prominenter Verfahren, die in der Gesichtserkennung als "state of the art" gelten. Hierbei sollten sicherlich noch das "Elastic Bunch Graph Matching" [WFKv97] und die "3D Morphable Models" [BV99][BR02] namentlich erwähnt werden. Eine umfangreiche Übersicht über diverse Verfahren findet man in [ZCRP03], [BCF06] und [FSB97].
Oft ist es nicht klar und eindeutig, ob ein Verfahren tatsächlich als Erkennungsverfahren gewertet werden sollte, in vielen Fällen wie zum Beispiel bei den Eigenfaces oder Active Appearance Modellen, handelt es sich im Grunde genommen um Merkmalsextraktionsverfahren, die zunächst nur eine parametrische Beschreibung liefern. Diese wird dann letztendlich unter Verwendung klassischer Mustererkennungsverfahren, oft einfache Abstandsklassifikatoren, den einzelnen Klassen zugeordnet. Da diese Verfahren aber in der gängigen Literatur typischerweise den Erkennungsverfahren zugeordnet werden, wird dieses auch hier so vorgenommen.

2.3.1 Eigenfaces

Das Verfahren der sogenannten Eigenfaces [SK87][TP91a][TP91b] ist sicherlich die bekannteste Methode unter den Gesichtserkennungsalgorithmen. Das Vorgehen selbst ist denkbar einfach: Die vorliegenden Daten werden in einen linearen Unterraum projiziert, der vorher aufgrund der Statistik von Beispielen ermittelt wurde. Anschließend wird in diesem Unterraum mit einem Abstandsklassifikator entschieden. Der lineare Unterraum wird von den Eigenfaces aufgespannt, welches die mittels Hauptkomponentenanalyse (engl. "Principal Component Analysis" (PCA)) aus den Trainingsdaten ermittelten Hauptkomponenten sind. Im Folgenden werden die einzelnen Verarbeitungsschritte genauer erklärt.

Ausgegangen wird von einem Trainingsdatensatz $X = \vec{x}_1, .., \vec{x}_N$ mit N Spaltenvektoren $\vec{x}_i, i \in [1; N]$. In den Trainingsdaten befinden sich mehrere bereits ausgeschnittene Gesichter, die alle die gleichen Bilddimensionen haben. In Abbildung 2.11 sind einige Beispiele dargestellt. Ein Spaltenvektor $\vec{x}_i$ ist dabei ein zu einem Vektor umgeformtes Bild eines Gesichtes, welches zum Training herangezogen werden soll. Die Umformung des Bildes geschieht dabei durch Aneinanderhängen der Bildspalten, so dass ein Spaltenvektor entsteht (siehe Abbildung 2.12).

Abbildung 2.11: Trainingsdaten aus dem FERET Datensatz. Die Gesichtsausschnitte werden alle auf die gleiche Größe normiert.

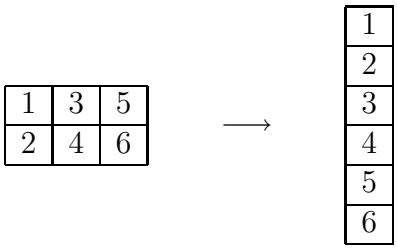


Abbildung 2.12: Umformung einer 2D Datenstruktur in einen Spaltenvektor. Die Spalten der 2D-Struktur werden einfach untereinander gehängt, so dass ein Spaltenvektor entsteht.

Zunächst wird der Mittelwert bestimmt:

$$\overline{\vec{x}} = \frac{\sum_{i=1}^{N} \vec{x}_i}{N} \tag{2.19}$$

Anschließend wird A, die Matrix der mittelwertfreien Daten, erzeugt.

$$A = [\vec{x}_1 - \overline{\vec{x}}, .., \vec{x}_N - \overline{\vec{x}}] \tag{2.20}$$

Als nächstes wird die Kovarianzmatrix C der Mittelwert freien Daten berechnet.

$$C = \frac{1}{N-1} A A^T \tag{2.21}$$

Die Eigenvektoren $\vec{e}$ der Kovarianzmatrix C sind diejenige, die die Gleichung

$$\lambda \vec{e} = C\vec{e} \tag{2.22}$$

Abbildung 2.13: Mittelwert aller Trainingsdaten, wenn $\overline{\vec{x}}$ in eine 2D-Bildstruktur zurückgewandelt wird.

Abbildung 2.14: Die Eigenfaces mit den größten Eigenwerten.

erfüllen. Da es sich bei der Kovarianzmatrix C immer um eine reelle symmetrische Matrix handelt, sind alle Eigenvektoren $\vec{e}$ orthogonal zueinander. Wenn diese Eigenvektoren zurück in eine 2D-Datenstruktur gewandelt werden, kann man erkennen, dass in ihnen Gesichtsmerkmale enthalten sind, daher ergibt sich der Name "Eigenfaces" (siehe Abbildung 2.14).

Der lineare Unterraum, der durch den Mittelwert $\overline{\vec{x}}$ und k Eigenvektoren $E_k = (\vec{e}_1, .., \vec{e}_k)$ aufgespannt wird, wird nun verwendet, um Gesichter zu beschreiben. Dieses geschieht, indem ein Gesicht $\vec{x}$ in diesen linearen Unterraum projiziert wird, es entsteht der Vektor der Projektionen $\vec{w}$:

$$\vec{w} = E_k^T(\vec{x} - \overline{\vec{x}}) \tag{2.23}$$

Die Anzahl k der zu verwendenden Eigenvektoren ist wählbar und wird oft an ein Qualitätskriterium gebunden. Typischerweise werden die Eigenvektoren $\vec{e}_i$ nach der Größe ihrer Eigenwerte λ_i sortiert, anschließend werden die ersten k Eigenvektoren verwendet, wobei k über das Festlegen einer Mindestqualität q ermittelbar ist.

$$q(k) = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{j=1}^{N} \lambda_j}. \tag{2.24}$$

q ist dabei ein Wert $\in [0, 1]$; je näher er an 1 ist, desto höher ist die geforderte Qualität. Die Rückprojektion $\vec{x}_r$ von dem durch die Eigenvektoren aufgespannten Gesichtsraum zurück in den Bildraum ergibt sich durch:

$$\vec{x}_r = \overline{\vec{x}} + \sum_{j=1}^{k} w_i \cdot \vec{e}_i = \overline{\vec{x}} + \cdot E_k \vec{w} \tag{2.25}$$

Diesen Prozess des Hin- und Rücktransformierens kann man sich auch so vorstellen, dass Gesichter durch den Mittelwert und eine Linearkombination der Eigenfaces beschrieben oder erzeugt werden (siehe Abbildung 2.15).

Die eigentliche Gesichtserkennung findet in dem durch die Eigenvektoren und den Mittelwert aufgespannten linearen Unterraum statt. Klassifiziert wird dabei typischerweise durch ein Abstandsmaß zwischen der Projektion $\vec{w}$ eines aktuell vorliegenden Gesichtes und den Projektionen $\vec{w}_{DB}$ aller bekannten Gesichter, die in einer Datenbank hinterlegt wurden. Hierbei werden viele verschiedene Abstandsmaße angewandt. Eines der einfachsten ist die Euklidische Distanz. Hierbei wird die Länge

Abbildung 2.15: Beschreibung von Gesichtern durch Linearkombination der Eigenfaces

des Differenzvektors zwischen $\vec{w}$ und $\vec{w}_{DB}$ berechnet.
Euklidischer Abstand:

$$d_{eukl}(\vec{w}, \vec{w}_{DB}) = \sqrt{(\vec{w} - \vec{w}_{DB})^T(\vec{w} - \vec{w}_{DB})} \tag{2.26}$$

Ein anderes Abstandsmaß, welches sich aus der multivariaten Normalverteilung ergibt, ist die Mahalanobis-Distanz. Hierbei werden die verschiedenen Vorzugsrichtungen der Verteilung unterschiedlich stark gewichtet, nämlich in dem Sinne, dass Richtungen, in denen die Beispieldaten stark streuen, schwächer in die Gesamtdistanz eingehen als Richtungen, in denen die Daten wenig streuen. Graphisch bilden die Punkte gleicher Mahalanobis-Distanz von einem Zentrum, je nach Dimensionalität, einen n-dimensionalen Ellipsoiden, während es bei der euklidischen Distanz eine n-dimensionale Kugel ist. Ist die Kovarianzmatrix die Einheitsmatrix, so entspricht die Mahalanobis-Distanz dem euklidischen Abstand.
Mahalanobis-Distanz:

$$d_{mah}(\vec{w}, \vec{w}_{DB}) = \sqrt{(\vec{w} - \vec{w}_{DB})^T C^{-1}(\vec{w} - \vec{w}_{DB})} \tag{2.27}$$

Die Entscheidung in einer Identifikationsaufgabe wird folgendermaßen getroffen. Der Abstand d zwischen einem zu identifizierenden Datensatz und allen Datenbankeinträgen bekannter Gesichter wird berechnet. Der unbekannte Datensatz wird derjenigen Person zugeordnet, zu der der Abstand minimal ist.
Dieser sehr verbreitete Ansatz wurde bereit in vielen Variationen verwendet, in [PMS94] wurde so auch untersucht, ob und wie man verschiedene Blickrichtungen mit diesem Ansatz modulieren kann.

2.3.2 Active-Appearance-Modelle

Bei den Active Appearance Modellen (AAM) [ETC98][CET01][CT04] handelt es sich um eine logische Weiterentwicklung oder Kombination der Eigenfaces und der Active Shape Modelle. Bei den AAM werden Textur- und Formvariationen jeweils getrennt durch Hauptachsentransformationen untersucht. Der Aufbau eines solchen Modells

Abbildung 2.16: Ein Beispiel für ein mit Markerpunkten versehenes Bild, welches zur Erzeugung eines Active Appearance Modells herangezogen wird (aus [SS04]).

unterteilt sich in mehrere Einzelschritte, die hier im Folgenden einzeln behandelt werden.

Datenbasis

Die notwendige Datenbasis zur Erzeugung eines AAM sind mit Markerpunkten versehene Gesichtsbilder. Diese Markerpunkte wurden so gewählt, dass sie eindeutige Merkmale im Gesicht beschreiben. Das können charakteristische Punkte wie Augenecken, Nasenspitze oder Mundwinkel sein, zusätzlich werden oft auch streckenteilende Punkte entlang Kanten zwischen solchen eindeutigen Punkten verwendet. Angenommen, es liegen N Bilder mit jeweils L Markerpunkten vor, so werden daraus N formbeschreibende Vektoren $\vec{s}$, die jeweils die Dimensionalität $2L$ haben, da L Punkte mit x- und y-Koordinate vorliegen.

Formmodell

Das in Active Appearance Modellen zum Einsatz kommenden Formmodell ist gleich dem Formmodell, welches bereits bei den Active Shape Modellen vorgestellt wurde, also ein anhand von Beispielen der gegebenen Datenbasis mittels PCA berechnetes Modell, bestehend aus mittlerer Form $\overline{\vec{s}}$ und den k bedeutendsten Hauptachsen $\vec{e}_{s,1}, .., \vec{e}_{s,k}$. Die formale Beschreibung und Berechnung des Formmodells ist die gleiche wie bei den ASM. Der Unterschied besteht darin, dass im Kontext der Active Appearance Modelle eine zusätzliche Modellierung für die Textur stattfindet und anschließend eine Matchingmethode formuliert wird, die sowohl Textur- als auch Formmodell verwendet. Daraus ergibt sich, dass bei der Wahl der formbeschreibenden Punkte für Active Appearance Modelle durchaus nicht nur Punkte entlang Kanten verwendet werden, sondern auch charakteristische Punkte in der Textur

Verwendung finden können (siehe Abbildung 2.16). Nachdem das Formmodell erstellt wurde, liegt, genau wie bei den Active Shape Modellen, eine Möglichkeit zur Formbeschreibung durch Linearkombination vor:

$$\vec{s} = \overline{\vec{s}} + \sum_{i=1}^{k} w_{s,i}\vec{e}_{s,i} \tag{2.28}$$

Hier ist $\vec{s}$ die zu beschreibende Form, $\overline{\vec{s}}$ die mittlere Form und $\vec{e}_{s,i}$ die bedeutendsten Eigenvektoren, auch "Moden" genannt, die jeweils mit den Formparametern $w_{s,i}$ gewichtet werden. Um einen Eindruck über die Modellierungsmöglichkeiten zu gewinnen, wird im Folgenden dargestellt, wie sich eine Gewichtung der ersten drei Moden mit jeweils ± 3 Standardabweichungen in der Form auswirkt (siehe Abbildung 2.17).

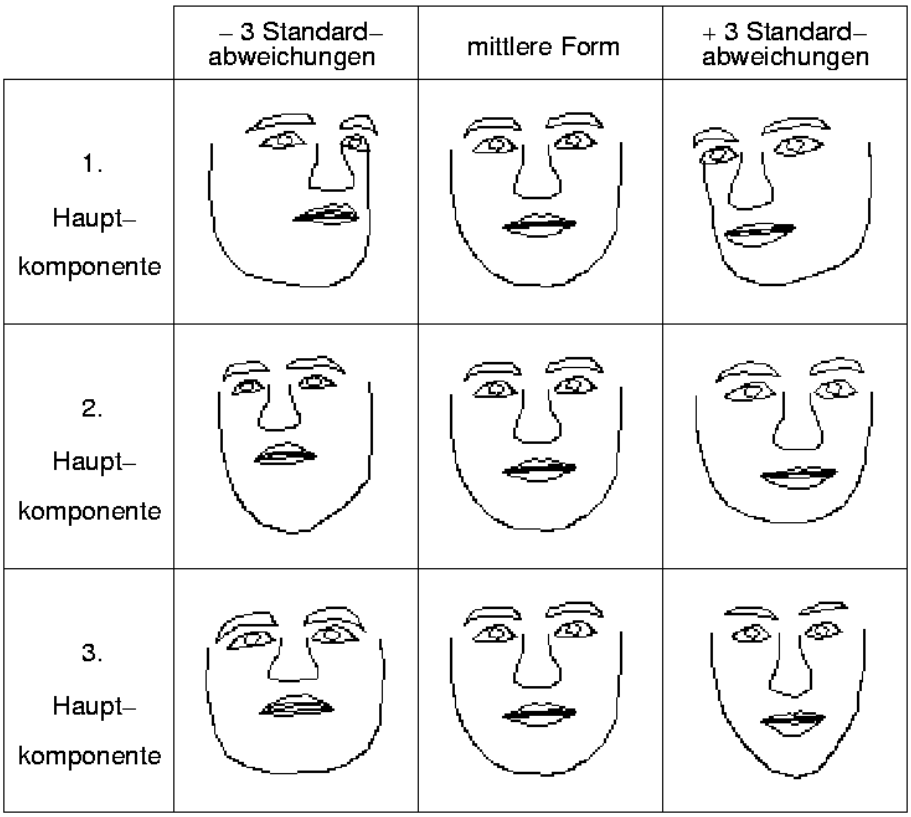


Abbildung 2.17: Einfluss der drei bedeutendsten Eigenvektoren auf die Formbeschreibung (aus [Stö04]).

Warping

Warping beschreibt das Umformen einer Textur von einer konkreten Form zu einer anderen. Im Kontext der AAM wird es dazu verwendet, Gesichter von einer gegebe-

nen Form auf die mittlere Form zu warpen oder von der mittleren auf eine gegebene Form. Die mittlere Form des Gesichts, beschrieben durch Punkte, wird trianguliert, um so ein Netz aus Dreiecken zu generieren. Wenn die Form nun verändert wird, bedeutet dies eine Positionsänderung der Punkte, somit auch eine Formveränderung der Dreiecke. Das eigentliche Warping geschieht nun durch lineares Texturmapping von den ursprünglichen Dreiecksformen auf die Neuen. Da jedes Dreieck einzeln bearbeitet wird, das Ganze also stückweise abgearbeitet werden kann, spricht man auch von "piecewise linear texture mapping". Der erste Schritt, um einen Warping-Algorithmus im Kontext der Active Appearance Modelle zu realisieren, besteht darin, dass man die Punktwolke der mittleren Form trianguliert. Ein typischer Triangulationsalgorithmus ist die Delaunay-Triangulation [For97], die das sogenannte Umkreiskriterium verwendet. Ein gültiges Dreieck als Ergebnis der Triangulation ist dabei so definiert, dass kein weiterer Punkt der zu triangulierenden Punktmenge innerhalb des Umkreises der Eckpunkte des Dreiecks liegen darf (siehe Abbildung 2.18). Eine einfache Variante der Delaunay-Triangulation überprüft also alle mög-

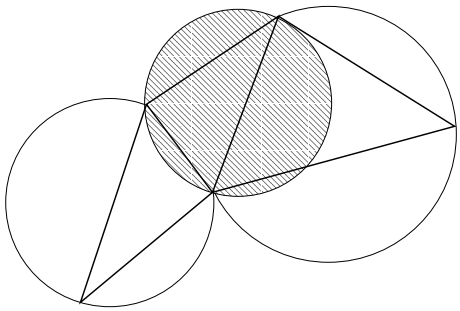


Abbildung 2.18: Delaunay-Kriterium: Kein weiterer Punkt darf innerhalb des Umkreises, der durch die Eckpunkte eines Dreiecks definiert ist, liegen (aus [Stö04]).

lichen Tupel, bestehend aus drei Punkten, ob sie ein gültiges Dreieck nach diesem Kriterium bilden.

Das resultierende Netz, bestehend aus Dreiecken, die durch Triangulation der Punkte der mittleren Form ermittelt wurden, ist die Basis für das Warping. Wenn sich nun die Form des Gesichts verändert, verschieben sich die Positionen der einzelnen Punkte. Ein modifiziertes Netz von Dreiecken wird erstellt, indem die Beziehungen, zwischen welchen Punkten Dreieckskanten bestehen, erhalten bleiben. Es wird also nicht neu trianguliert, sondern die Kanten werden nur relativ zur Positionsveränderung der Punkte mit verschoben. Anschließend wird für jedes einzelne Dreieck die in ihm enthaltene Textur auf das korrespondierende, modifizierte Dreieck mittels Texturmapping abgebildet. Dabei wird jeder Intensitätswert der Textur an die gleiche relative Position im veränderten Dreieck geschrieben (siehe Abbildung 2.19). Um Abtastungseffekte zu vermeiden, wird gängigerweise für jede Pixelposition im

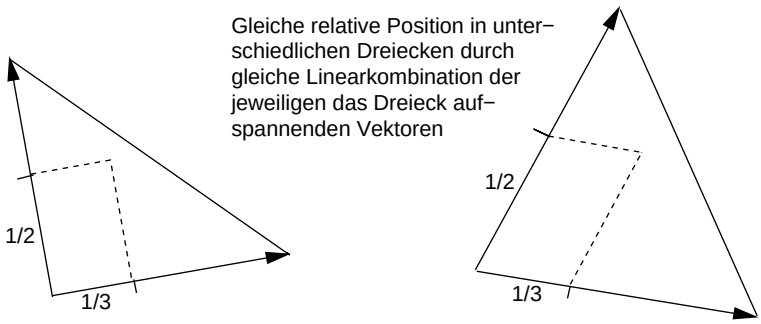


Abbildung 2.19: Texturmapping, durch Schreiben von Intensitätswerten an die gleiche relative Position im formveränderten Dreieck (aus [Stö04]).

neuen Dreieck, die korrespondierende Position im ursprünglichen Dreieck berechnet, anschließend wird der Intensitätswert durch Interpolation ermittelt. Dieses wird mit dem Fachausdruck "backward resampling" oder "backward mapping" bezeichnet. Das Warping für ein komplettes Gesicht wird dann ausgeführt, indem alle Texturen innerhalb von Dreiecken der triangulierten Mittelwertform per Texturmapping auf die jeweils veränderte Dreiecksform gebracht werden.

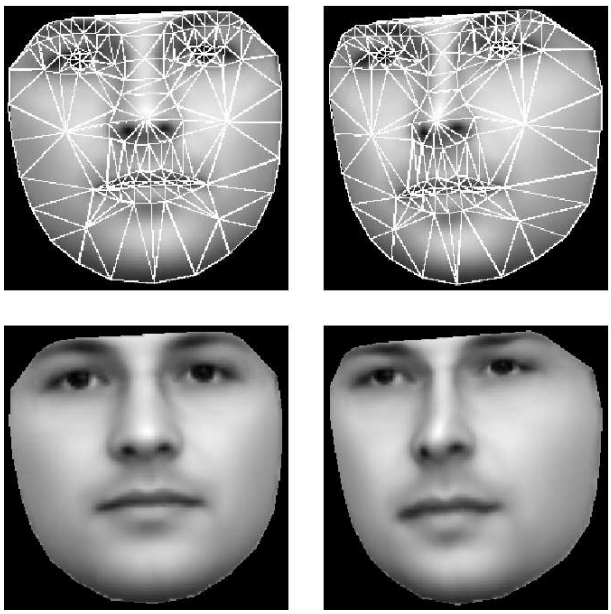


Abbildung 2.20: Warping durch Texturmapping jedes einzelnen Dreiecks der mittleren Form (links) auf das Korrespondierende in der modifizierten Form (rechts)(aus [SS04]).

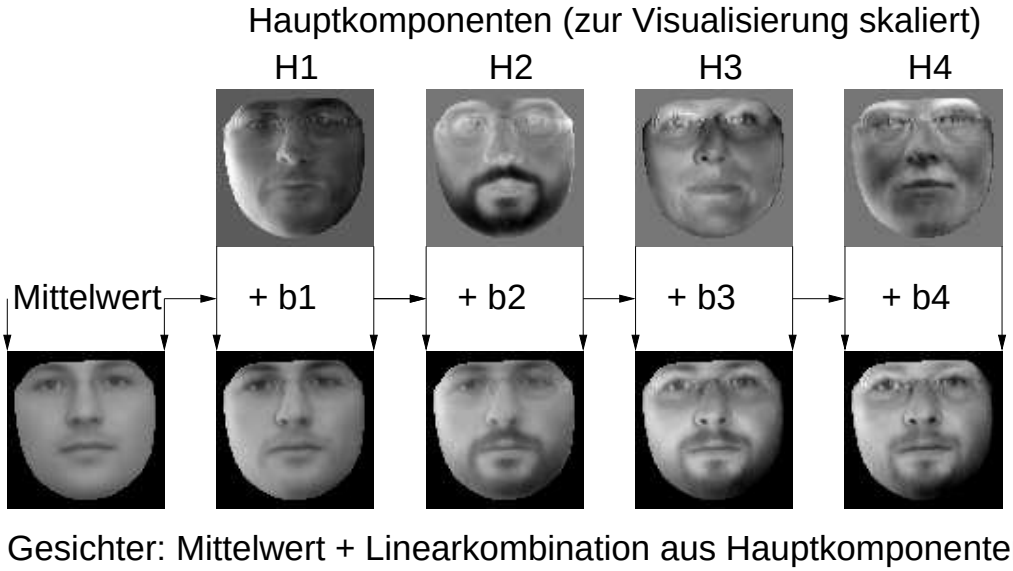


Abbildung 2.21: Erzeugung von Gesichtstexturen durch Linearkombination von Mittelwert und den bedeutendsten Hauptkomponenten H_i, welche sich aus der Rückformung der Eigenvektoren $\vec{e}_{g,i}$ in 2D-Bilder ergeben (aus [Stö04]).

Texturmodell

Das Textur- oder Grauwertmodell funktioniert genau so, wie der Ansatz der Eigenfaces (siehe 2.3.1). Der einzige Unterschied ist, dass bevor die PCA ausgeführt wird, alle Gesichter mit der beschriebenen Warpingmethode auf die mittlere Gesichtsform gewarpt werden, somit Formeinflüsse weitgehend aus der Textur heraus normiert werden. Anschließend wird, genau wie bei den Eigenfaces, der Mittelwert $\overline{\vec{g}}$ und die k bedeutendsten Eigenvektoren $\vec{e}_{g,1}, \cdots, \vec{e}_{g,k}$ mit den k größten Eigenwerten $\lambda_{g,1}, \cdots \lambda_{g,k}$ bestimmt. Somit kann eine Gesichtstextur $\vec{g}$ durch eine Linearkombination aus Mittelwert $\overline{\vec{g}}$ und Eigenvektoren $\vec{e}_{g,i}$ beschrieben werden (Abbildung 2.21):

$$\vec{g} = \overline{\vec{g}} + \sum_{j=1}^{k} w_{g,i} \cdot \vec{e}_{g,i} \tag{2.29}$$

Zusammenwirken

Mittels Form- und Texturmodell kann nun für ein bestehendes, mit Markerpunkten versehenes Gesicht eine parametrische Beschreibung erzielt werden, indem zunächst die Form parametrisch beschrieben wird (Abbildung des Formvektors $\vec{s}$ in einen linearen Unterraum (Formmodell)), anschließend die Textur innerhalb der beschriebenen Form auf die mittlere Form gewarpt wird, und dann für die resultierende

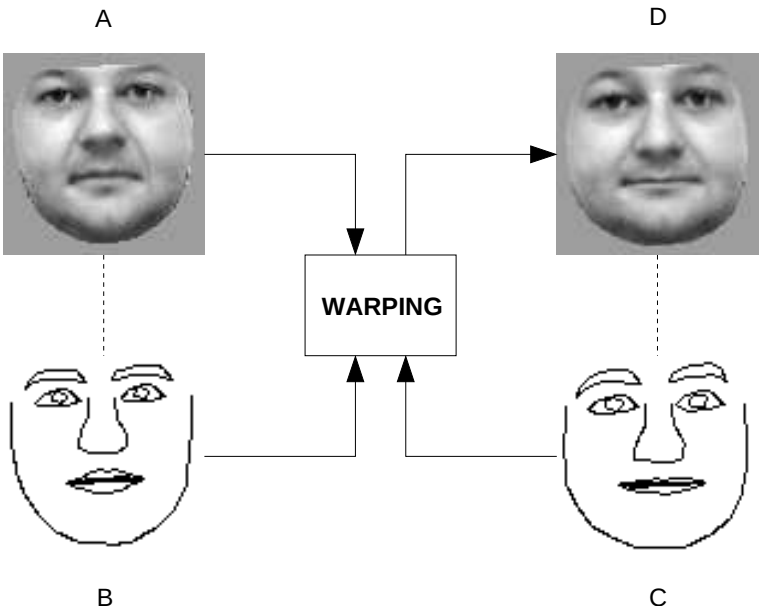


Abbildung 2.22: Zusammenwirken von Form und Texturmodell durch Warping am Beispiel einer Gesichtssynthese: Eine durch das Texturmodell erzeugte Textur (A) wird von der mittleren Form (B) auf eine durch das Formmodell erzeugte Form (C) gewarpt, um das resultierende Gesicht (D) zu erzeugen (aus [SS04]).

Textur eine parametrische Beschreibung erzeugt wird (Abbildung des Texturvektors $\vec{g}$ in einen linearen Unterraum (Texturmodell)).
Der Umkehrprozess, nämlich das Erzeugen eines synthetischen Gesichtes aus einem Parametervektor, geschieht, indem zunächst die auf die mittlere Form bezogene Textur durch Linearkombination erzeugt wird. Anschließend wird die Form durch das Formmodell erzeugt. Das resultierende Gesicht wird nun erzeugt, indem die Textur von der mittleren Gesichtsform auf die resultierende Form gewarpt wird (siehe Abbildung 2.22).

Da in der Regel davon ausgegangen wird, dass Form und Textur eines Gesichtes miteinander korreliert sind, wird gerne noch eine weitere PCA auf dem zusammengesetzten Parametervektor $\vec{w} = (\vec{w}_s^T, \vec{w}_g^T)^T$ aus Form- und Texturparametern durchgeführt. Daraus ergeben sich wiederum Mittelwert $\overline{\vec{w}}$ und Eigenvektoren $\vec{e}_{a,i}$. Ein Gesicht kann anschließend durch Linearkombination dieser erzeugt werden:

$$\vec{w} = \overline{\vec{w}} + \sum_{i=1}^{k} w_{a,i}\vec{e}_{a,i} \tag{2.30}$$

Die Parameter $w_{a,i}$ werden als Appearance-Parameter bezeichnet und beschreiben ein Gesicht vollständig in Form und Textur (siehe 2.23).

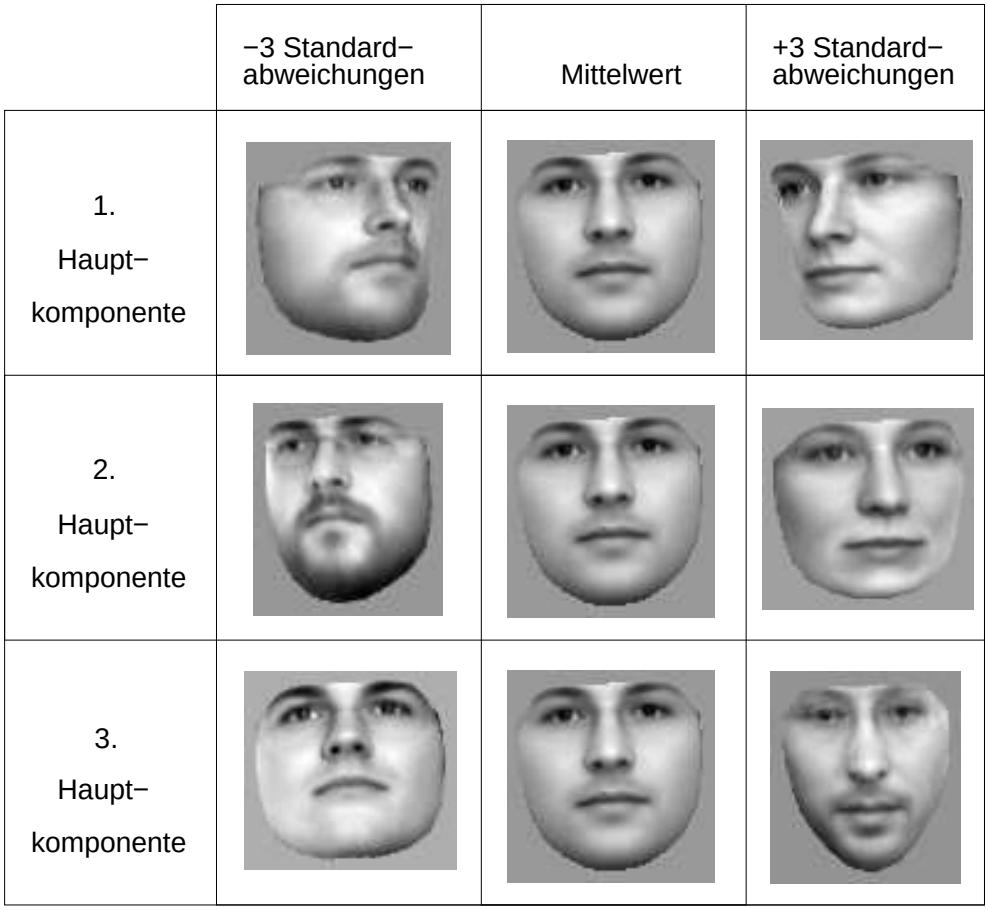


Abbildung 2.23: Erzeugung unterschiedlicher Gesichter durch Änderung der Appearance Parameter $w_{a,1}, \cdots, w_{a,3}$ im Bereich von ± 3 Standardabweichungen (aus [Stö04]).

Matching

Für das Matching eines Active Appearance Modells, also das Schätzen der Appearance-Parameter bei einem gegebenen Gesichtsbild ohne Markerpunkte, wurden in den letzten Jahren mehrere Verfahren entwickelt. Das Grundprinzip bei den meisten dieser Verfahren ist es, die Energie des Differenzbildes δI zwischen Originalbild I_{orig} und durch das Modell synthetisiertem Gesicht I_{synth} zu minimieren.

$$\delta I = I_{orig} - I_{synth} \tag{2.31}$$

Diese Differenzbildung wird typischerweise nicht im Original-Bildraum gemacht, sondern im auf die mittlere Form normierten Raum (siehe Abbildung 2.24). Es wird also die Textur aus dem Originalbild extrahiert, basierend auf der aktuellen Formschätzung. Diese wird auf die mittlere Gesichtsform gewarpt und mit der Textur,

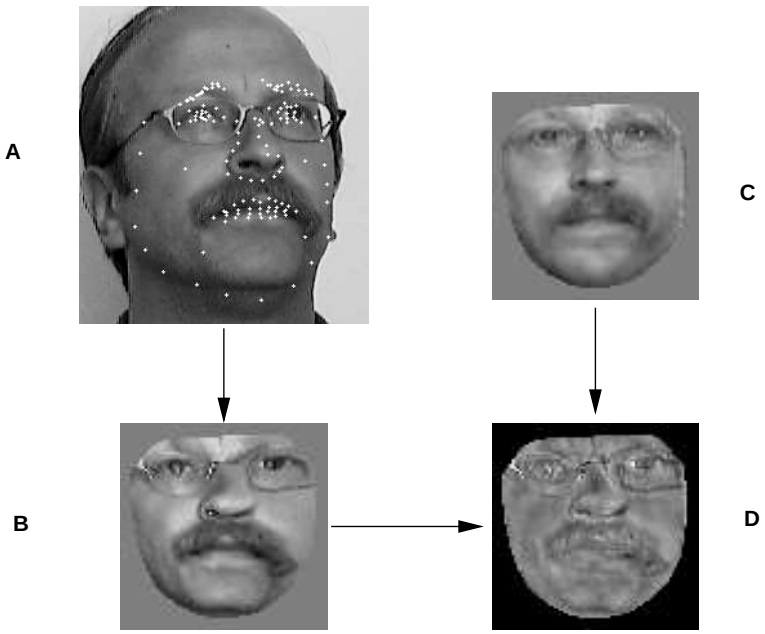


Abbildung 2.24: Berechnung des Residuums im formnormierten Raum. Die Textur aus dem Originalbild (A) wird basierend auf der aktuellen Formschätzung auf die mittlere Form gewarpt (B) und mit der Texturschätzung (C) verrechnet, um das Differenzbild (D) zu erzeugen (aus [SS04]).

die aus dem Texturmodell erzeugt wird, verrechnet.

Um Gesichter in beliebiger Position und Größe im Bild synthetisieren zu können, muss der bestehende Appearance Parametervektor noch um 4 weitere Parameter erweitert werden, die eine Similarity-Transformation im 2D beschreiben, also Translation, Rotation und Skalierung (siehe Anhang B). Aus den Appearance Parametern $\vec{w}_a$ und diesen Transformationsparametern ergibt sich ein Parametervektor $\vec{p}$, aus dem sich eindeutig ein Gesicht synthetisieren lässt. Die Differenzbildung im formnormierten Raum ergibt sich dann also zu

$$\vec{r}(\vec{p}) = \vec{g}_{orig}(\vec{p}) - \vec{g}_{synth}(\vec{p}). \tag{2.32}$$

Hier ist $\vec{r}$ das Differenzbild in Vektorform, auch Residuum genannt, $\vec{g}_{orig}$ die aus dem Originalbild extrahierte, auf die mittlere Form gewarpte Textur, und $\vec{g}_{synth}$ die mittels der aktuellen Texturparameter synthetisierte Textur. Ziel des Optimierungsverfahren ist es, die Energie $|\vec{r}|^2$ des Residuums $\vec{r}$ zu minimieren. Eine Taylor-Erweiterung erster Ordnung von Gleichung 2.32 ist:

$$\vec{r}(\vec{p} + \delta\vec{p}) = \vec{r}(\vec{p}) + \frac{\partial\vec{r}}{\partial\vec{p}}\delta\vec{p} \tag{2.33}$$

Dabei ist $\frac{d\vec{r}_i}{d\vec{p}_j}$ das ij-te Element der Matrix $\frac{\partial\vec{r}}{\partial\vec{p}}$ der partiellen Ableitungen.
Angenommen, es gibt eine Parameterschätzung $\vec{p}$ mit dazugehörigem Residuum $\vec{r}(\vec{p})$, dann gilt es $\delta\vec{p}$ so zu wählen, dass $|\vec{r}(\vec{p}+\delta\vec{p})|^2$ minimiert wird. Durch Gleichsetzen von Gleichung 2.33 mit Null und Umstellen kommt man zu:

$$-\vec{r}(\vec{p}) = \frac{\partial\vec{r}}{\partial\vec{p}}\delta\vec{p} \tag{2.34}$$

durch linksseitiges Einmultiplizieren von $\frac{\partial\vec{r}}{\partial\vec{p}}^T$ kommt man zu:

$$-\frac{\partial\vec{r}}{\partial\vec{p}}^T \vec{r}(\vec{p}) = \frac{\partial\vec{r}}{\partial\vec{p}}^T \frac{\partial\vec{r}}{\partial\vec{p}}\delta\vec{p} \tag{2.35}$$

und somit durch Umstellen zu

$$\delta\vec{p} = -P\vec{r}(\vec{p}) \qquad \text{wobei} \qquad P = \left(\frac{\partial\vec{r}}{\partial\vec{p}}^T \frac{\partial\vec{r}}{\partial\vec{p}}\right)^{-1} \frac{\partial\vec{r}}{\partial\vec{p}}^T \tag{2.36}$$

Unter Verwendung eines Standardverfahrens zur Optimierung, wäre es notwendig, $\frac{\partial\vec{r}}{\partial\vec{p}}$ in jedem Schritt neu zu berechnen, was eine sehr aufwendige Operation wäre. Hier wird allerdings angenommen, dass es dadurch, dass es in einem normalisierten Bereich berechnet wird (formnormierte und grauwertnormierte Gesichter), vorher fix approximiert werden kann.
$\frac{\partial\vec{r}}{\partial\vec{p}}$ wird aus den Trainingsdaten durch numerische Differentiation geschätzt. Jeder Parameter wird für verschiedene Trainingsbeispiele vom bekannten Optimum verschoben, die jeweilige partielle Ableitung wird numerisch berechnet und über die Trainingsbeispiele gemittelt. Anschließend wird daraus, wie in Gleichung 2.36 beschrieben, die Prädiktormatrix P berechnet.

Das hier beschriebene Standardverfahren zum Matching von AAM wurde in vielen Varianten modifiziert oder neu formuliert. Eine Übersicht über Varianten des Originalansatzes findet man in [CKn02], neue Varianten von Matchingalgorithmen werden ständig weiterentwickelt [CWT00][MB04][PM08] und sind nach wie vor ein aktuelles Forschungsgebiet. Im Rahmen dieser Arbeit wurde die hier beschriebene Basisversion verwendet.

2.3.3 Hidden Markov Modelle

Ein Hidden Markov Modell (HMM) ist ein stochastisches Modell, das sich durch zwei Zufallsprozesse beschreiben lässt. Der erste Zufallsprozess entspricht dabei einer Markov-Kette, die durch Zustände und Übergangswahrscheinlichkeiten gekennzeichnet ist. Die Zustände der Kette sind von außen jedoch nicht direkt sichtbar ("hidden",

verborgen). Stattdessen erzeugt ein zweiter Zufallsprozess zu jedem Zeitpunkt beobachtbare Ausgangssymbole gemäß einer zustandsabhängigen Wahrscheinlichkeitsverteilung. Die Bezeichnung "hidden" bezieht sich bei einem HMM auf die Zustände der Markov-Kette während der Ausführung, nicht auf die Parameter des Markov-Modells. Die Aufgabe besteht häufig darin, aus der Sequenz der Ausgabesymbole (Beobachtungen) auf die Sequenz der verborgenen Zustände zu schließen. Wichtige Anwendungsgebiete sind die Sprach-, Gesten- und Handschrifterkennung. Allgemein werden HMM in vielen verschiedenen Mustererkennungsaufgaben eingesetzt und haben so auch Einzug in die Gesichtserkennung gefunden. Im Folgenden wird eine kurze Erläuterung zum Grundverständnis durchgeführt, für Interessierte sei an dieser Stelle auf die Beschreibung von HMM durch Rabiner [Rab90] hingewiesen.

Ein HMM λ wird beschrieben durch N Zustände $Q = [S_1, \ldots, S_N]$und eine $N \times N$ dimensionale Matrix A von Zustandsübergangswahrscheinlichkeiten $A = [a_{ij}]$. Deren Elemente a_{ij} beschreiben jeweils die Wahrscheinlichkeit vom Zustand S_i zum Zustand S_j zu wechseln, ein solcher Zustandsübergang wird auch Transition genannt.

$$a_{ij} = p(q_{t+1} = S_j \mid q_t = S_i); 1 \leq i \leq N; 1 \leq j \leq N. \tag{2.37}$$

q_t beschreibt dabei den Zustand zum Zeitpunkt t der Zustandssequenz $q = q_1, \ldots, q_T; q_t \in Q$. Die Wahrscheinlichkeitsverteilung der Beobachtungen o im Zustand S_j, wird beschrieben durch $b_j(o)$. Für diskrete Beobachtungen (o ist Element einer endlichen Menge möglicher Symbole) ist die Verteilung der Wahrscheinlichkeiten diskret, für kontinuierliche Beobachtungen ($o \in \mathbb{R}$) ist diese Verteilung kontinuierlich. Die Wahrscheinlichkeitsverteilungen von Beobachtungen in allen Zuständen werden in $B = [b_j(o)]$ gespeichert.

Die Einsprungswahrscheinlichkeiten $\pi = \{\pi_j\}$ sind jeweils die Wahrscheinlichkeiten, dass die Zustandssequenz im Zustand S_j beginnt..

$$\pi_j = p(q_1 = s_1); 1 \leq j \leq N. \tag{2.38}$$

Durch die Größen π, A und B ist ein Hidden Markov Modell $\lambda(A, B, \pi)$ vollständig beschrieben. Die Hauptaufgabe beim Einsatz eines HMM als Klassifikator ist es, die Produktionswahrscheinlichkeit $P(O|\lambda)$ einer Beobachtungssequenz $O = (o_1, .., o_T)$ für ein gegebenes Modell λ zu bestimmen.

$$P(O|\lambda) = \sum_{q \in Q^T} \prod_{t=1}^{T} b_{q_t}(o_t) \pi_{q_1} \prod_{t=2}^{T} a_{q_{t-1} q_t} \tag{2.39}$$

Neben dem Aufsummieren der Wahrscheinlichkeiten aller möglicher Pfade, seien hier als Berechnungsmöglichkeiten für diese Produktionswahrscheinlichkeit der "Forward-Backward"-Algorithmus und der Viterbi-Algorithmus [Vit67], der den wahrscheinlichsten Pfad ermittelt, und somit ermöglicht, die Gesamtproduktionswahrscheinlichkeit durch dessen Wahrscheinlichkeit zu approximieren, genannt [Rab90]. Ein

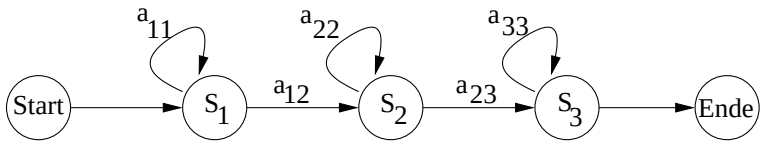


Abbildung 2.25: Ein HMM in der Links-Rechts-Modell Form. Das Modell wird von links nach rechts durchlaufen, jeder Zustand wird dabei verwendet.

Pfad ist hierbei eine konkrete Sequenz von Zuständen.

Ein weiterer wichtiger Aspekt bei der Verwendung von HMM ist das Trainieren, also das Bestimmen der Parameter λ, A und B, für eine gegebene Anzahl von Beobachtungssequenzen die durch ein HMM mit gegebener Struktur von Zuständen und Zustandsübergängen modelliert werden sollen. Hierfür wird der sogenannte Baum-Welch Algorithmus verwendet, der ein Spezialfall des "Expectation-Maximization" Algorithmus ist und in [Bil98] detailliert erläutert wird.

Die grundsätzliche Idee, beim Einsatz von Hidden Markov Modellen zur Klassifikation von Mustern ist es, für jede Klasse ein solches HMM zu erstellen. Wenn nun ein unbekanntes Muster vorliegt, wird die Produktionswahrscheinlichkeit dieses Musters für jedes HMM bestimmt, und es wird derjenigen Klasse zugeordnet, für die die Produktionswahrscheinlichkeit des jeweiligen klassenspezifischen HMM am größten ist. Für den Anwendungsfall Gesichtserkennung bedeutet dieses, dass für jedes Individuum, welches erkannt werden soll, eine eigene Klasse mit jeweils einem HMM modelliert wird.

Eine in der Mustererkennung gängige Form des HMM ist das sogenannte Links-Rechts-Modell, auch Vorwärtsmodell genannt (Abbildung 2.25). Dieses beschreibt eine konkrete Struktur von Zuständen und Zustandsübergängen, die dadurch geprägt ist, dass die Zustandssequenz in genau einem Zustand beginnen kann und in genau einem Zustand terminiert. Dazwischen können nur Selbsttransitionen (Verbleiben im gleichen Zustand) oder Vorwärtstransitionen (Übergang zum nächsten Zustand) geschehen, alle anderen Übergangswahrscheinlichkeiten sind Null. Dieses Modell ist an zeitliche Signalverläufe angelehnt, kann aber auch als Modell für einen örtlichen Signalverlauf in genau einer Richtung interpretiert werden.

Pseudo-Zweidimensionale Hidden Markov Modelle

Pseudo-Zweidimensionale Hidden Markov Modelle (P2D-HMM) sind eine bestimmte Zustands- und Zustandsübergangsstruktur der klassischen HMM, um diese auch für zweidimensionale Signalverläufe nutzen zu können, siehe Abbildung 2.26. Die

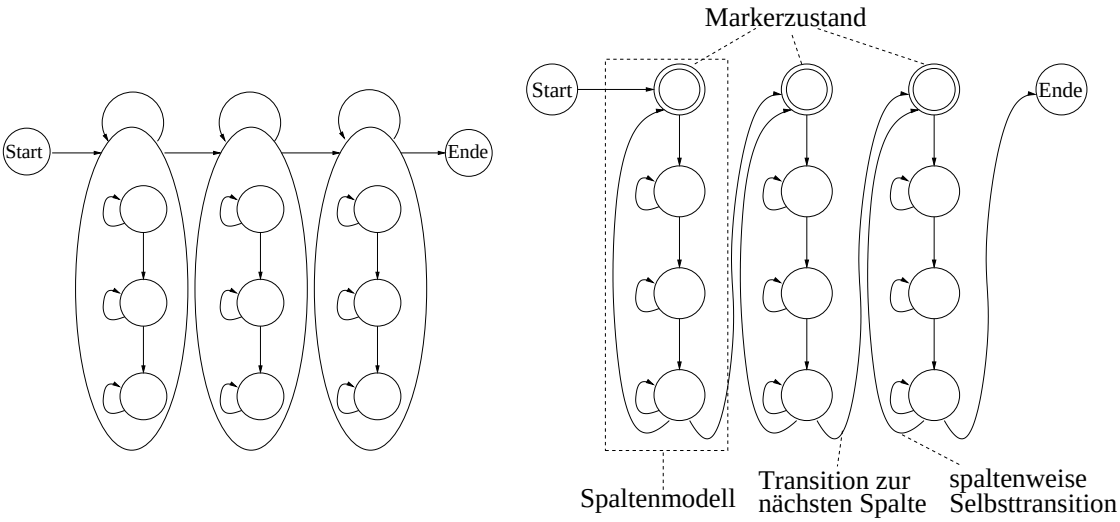


Abbildung 2.26: Ein P2D-HMM (links) und das Ersatzschaltbild als 1D-HMM (rechts). Ein P2D-HMM ist eine verschachtelte Struktur von Links-Rechts-Modellen. Neben Selbst- und Vorwärtstransitionen sind spaltenweise Selbst- und Vorwärtstransitionen erlaubt.

Grundidee dabei ist es, jede Spalte eines zweidimensionalen Signalverlaufs durch ein Links-Rechts-Modell zu modellieren, den Gesamtverlauf des Signals dann wiederum als Links-Rechts-Modell von Spalten. Dadurch entsteht die verschachtelte Struktur eines Links-Rechts-Modells, dessen Zustände wieder eigenständige Links-Rechtsmodelle sind (Spaltenmodelle). Jedes Spaltenmodell ist ein Links-Rechts-Modell mit einem zusätzlichen Markerzustand als ersten Zustand. Transitionen vom letzten Zustand jedes Spaltenmodells zur diesem Markerzustand (spaltenweise Selbsttransition) und zum Markerzustand der nächsten Spalte (spaltenweise Vorwärtstransition) werden eingefügt.
Der Markerzustand ist dabei ein spezieller Zustand, der eine maximale Beobachtungswahrscheinlichkeit für genau einen speziellen Markerwert hat, der künstlich zu Beginn jeder Spalte des Signalverlaufs zur Beobachtungssequenz hinzugefügt wird. Dieser Markerwert ist ein beliebiger Wert, der im eigentlichen Wertebereich des Signals nicht vorkommt. Für alle anderen Beobachtungen ist die Beobachtungswahrscheinlichkeit in diesem Markerzustand Null. Mit dieser Maßnahme wird gewährleistet, dass eine Ausrichtung zwischen Signalverlauf und zweidimensionalen Zustandsmodell stattfindet, da immer nur bei Auftreten dieser Markerwerte, also beim Beginn einer neuen Spalte im Signalverlauf, ein Übergang zum nächsten Spaltenmodell innerhalb des P2D-HMM stattfinden kann. Somit wird erzwungen, dass tatsächlich ganze Bildspalten von den Spaltenmodellen modelliert werden.

Da es sich bei den P2D-HMM nur um eine konkrete Struktur der HMM handelt, können genau die gleichen Algorithmen zum Trainieren (Bestimmen der Modellparameter) und Erkennen (Berechnung der Produktionswahrscheinlichkeiten) verwendet werden, wie bereits erwähnt.

HMM und P2D-HMM in der Gesichtserkennung

HMM wurden bereits in mehreren verschiedenen Varianten in der Gesichtserkennung verwendet, siehe [SY94], [NH98], [Nef99] und [EMR00]. Erste Ansätze versuchten normale eindimensionale Links-Rechts-Modelle einzusetzen, deren Beobachtungen die Grauwertverläufe ganzer Bildspalten oder -zeilen waren. Später wurden dann verschiedene Formen der P2D-HMM entwickelt, die mittels Zeilen- oder Spaltenmodelle diese Grauwertverläufe modellierten. Es hat sich allerdings gezeigt, dass eine solche Modellbildung aufgrund Intensitätswerten von Pixeln sich als komplex erweist, da sehr viele Zustände verwendet werden müssen, um einen solchen, relativ feinschrittigen Signalverlauf zu modellieren. Dies führt zu dem Nachteil, dass sehr viele Trainingsdaten verwendet werden müssten, um eine hinreichend gute Schätzung der Modellparameter zu erreichen. Deshalb werden nach aktuellem Stand der Technik mehrere Pixel zu Blöcken zusammengefasst, aus denen Blockmerkmale ermittelt werden. Gerne werden sich überlappende Blöcke verwendet, um die sprunghafte Änderung von Merkmalswerten von Block zu Block zu vermeiden. Die Merkmale, die aus diesen Blöcken extrahiert werden, müssen keine Grauwerte sein, in der Praxis werden die verschiedensten Mermalstypen verwendet. Insgesamt findet durch diese Maßnahme der blockweisen Mermalsextraktion eine Abstraktion von Bildpunkten zu Bildregionen statt, die durch die jeweiligen Merkmale beschrieben werden. Dadurch kann eine Modellierung durch die generelle Abfolge von Bildregionen geschehen, wobei in den Beobachtungswahrscheinlichkeiten der Zustände modelliert ist, welche Merkmale in den Bildregionen jeweils auftreten.

Einzelne Aspekte der im Rahmen dieser Arbeit verwendeten Varianten dieses Basisansatzes werden in den jeweiligen Kapiteln noch einmal erläutert. Dort werden die auch jeweils verwendeten Merkmalsextraktionsmechanismen vorgestellt.

3

Frontale Gesichtserkennung für die Zutrittskontrolle

In diesem Kapitel wird ein konkretes Gesichtserkennungssystem erläutert, dass im Rahmen dieser Arbeit für einen realen Anwendungsfall entwickelt wurde. Der gegebene Anwendungsfall ist hierbei die Überprüfung der Identität beim Betreten eines Flugzeuges (engl. "Boarding"), und wurde im Kontext von SAFEE ("Security of Aircraft in the Future European Environment") [Gau05], einem vierjährigen Forschungsprojekt der Europäischen Union in sechsten Rahmenprogramm, umgesetzt. Die Grundidee hierbei ist, dass Personen, die fliegen möchten, beim Check-In aufgenommen werden, ein Gesichtsmodell wird erstellt und auf der Bordkarte gespeichert. Beim Durchlaufen der diversen Sicherheitskontrollen wird jeweils eine korrekte Überprüfung auf der Bordkarte vermerkt. Beim Betreten des Flugzeuges soll nun kontrolliert werden, ob alle Sicherheitskontrollen durchgeführt wurden, und ob es sich um die richtige Person handelt.

Aus technischer Sicht stellt dieses System mehrere Herausforderungen dar, insbesondere was die Bearbeitungszeiten der prinzipiellen Verarbeitungsschritte und den zur Verfügung stehenden Speicherplatz auf der Bordkarte angeht. Als Zielvorgabe ist hier jeweils die Dauer von maximal 1 Sekunde sowohl für das Erstellen des Gesichtstemplates als auch für die Erkennung beim Betreten des Flugzeuges zu erreichen. Weiterhin gilt es, mögliche Fehlerquellen, die in der Praxis auftreten können zu berücksichtigen. Diese sind insbesondere Fehler bei der Gesichtsdetektion und variable Beleuchtungsbedingungen zwischen dem Ort der Modellerstellung (Check-In Schalter) und der Erkennung (Eingangsbereich des Flugzeuges). Eine weitere technische Herausforderung, stellt die Wahl der Kameraposition dar, hier gilt es für Check-In und Boarding möglichst ähnliche Blickwinkel zu erreichen. Dies ist aber durch die enge Bauweise im Eingangsbereich eines Flugzeuges eine durchaus herausfordernde Aufgabe.

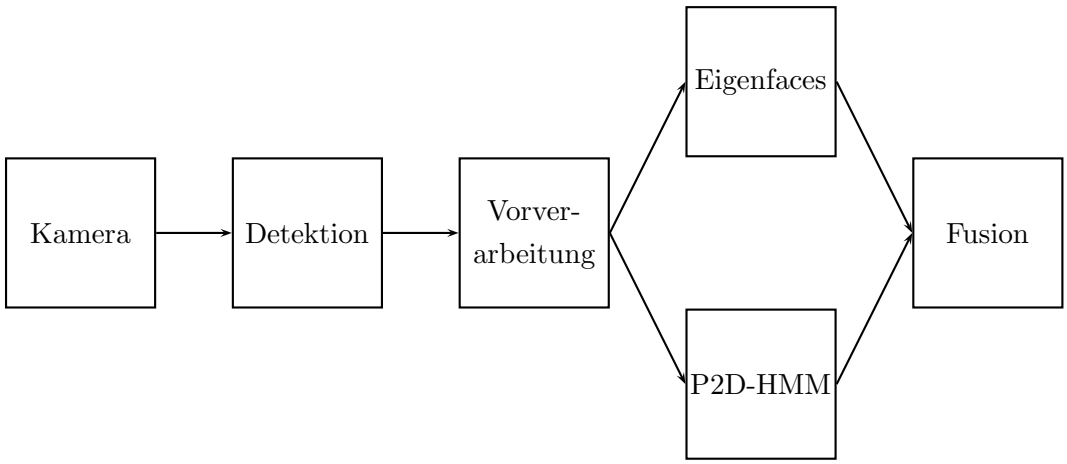


Abbildung 3.1: Verarbeitungskette im Zugangskontrollszenario im Flugzeug.

3.1 Grundkonzept

Da das resultierende System eine hohe Robustheit gegenüber möglichen Fehlerquellen haben soll, wurde das Grundkonzept entwickelt, nicht nur mit einer Gesichtsaufnahme sondern mit einer kurzen Sequenz von Gesichtsaufnahmen zu arbeiten. Dieses bietet mehrere Vorteile:

- Einzelne Fehldetektionen und Störungen haben geringere Auswirkungen
- Es stehen mehr Trainingsdaten für das Mustererkennungssystem zur Verfügung
- Nebenaufgaben, wie zum Beispiel das Überprüfen, ob ein statisches Bild vor die Kamera gehalten wird, um das System zu überlisten, können gelöst werden
- Im Testfall können diverse Fusionsstrategien verwendet werden, um die Ergebnisse der einzelnen Bilder zusammenzuführen

Zusätzlich zu der Verwendung mehrerer Bilder wurde auch entschieden, zwei verschiedene Erkennungsverfahren zu verwenden und deren Ergebnisse zu fusionieren. Die beiden verwendeten Erkennungsverfahren sind “Eigenfaces” und Pseudo-2-Dimensionale Hidden Markov Modelle (P2D-HMM). Die dadurch entstehende Verarbeitungskette ist in Abbildung 3.1 abgebildet.

3.2 Wahl der Kameraposition und Optik

Die Wahl der Kameraposition stellt in der gegebenen Anwendung eine besondere Herausforderung dar. Da der Eingangsbereich des Flugzeuges, in dem die Über-

prüfung der Identität stattfinden soll, eine sehr enge Bauweise hat, gleichzeitig die bestehenden Durchgangswege zum Betreten der Flugzeugkabine, des Cockpits oder der Toiletten nicht verbaut werden dürfen, steht nur eine beschränkte Auswahl von Montagemöglichkeiten zur Verfügung. Da das Gesamtkonzept vorsieht, dass der Passagier seine Bordkarte an ein Lesegerät hält, gilt es gleichzeitig, diese Position zu berücksichtigen. Hier ist es vorgesehen, dass dieses Lesegerät rechts, unmittelbar hinter der Eingangstür angebracht wird. Die Möglichkeiten für verschiedene Kamerapositionen sind (siehe Abbildung 3.2):

- Montage an der Decke der Kabine: Es entsteht ein mehr oder weniger steiler Blick von oben auf die zu authentifizierenden Personen, je nachdem wie weit die Kamera vom Eingang entfernt angebracht wird. Hier wurden konkret zwei Positionen ausgetestet, eine direkt über der gegenüberliegenden Tür und eine im Flugzeuginneren an der Decke montierte Kamera, siehe Abbildung 3.3.

- Montage an der Mittelkonsole zwischen den beiden Türdurchgängen von Eingangsbereich zu Kabinenbereich: Hier wurden zwei Positionen, eine am vorderen und eine am hinteren Ende der Mittelkonsole getestet. Es entsteht ein leicht schräger Blick, siehe Abbildung 3.4

- Montage vor dem ersten Durchgang zum Kabinenbereich direkt rechts der Tür, siehe Abbildung 3.5. Eine günstige Variante, vor allem dadurch, dass der Bordkarten-Leser kurz davor angebracht wurde. Die Passagiere bleiben typischerweise, während sie die Bordkarte an den Leser halten, kurz stehen.

Zusätzlich zur Kameraposition steht noch die Wahl der zu verwendenden Optik aus. Hier gilt es die Fragestellung zu untersuchen, ob eine Großaufnahme ("herangezoomt", längere Brennweite der Linse) des interessierenden Bereiches günstiger ist, oder ob ein weiteres Blickfeld von Vorteil ist ("herausgezoomt", kürzere Brennweite). Es gilt zu berücksichtigen, dass verschieden große Flugzeugpassagiere erkannt werden müssen, es gilt also einen eher größeren Bereich einzuplanen, in denen Gesichter auftreten können.
Weiterhin hat es sich gezeigt, dass außerhalb und innerhalb des Flugzeuges extrem unterschiedliche Beleuchtungsbedingungen auftreten können. Deshalb ist es vorteilhaft, ein Blickfeld anzustreben, welches sich zu größten Teilen auf einen Beleuchtungsfall konzentriert. Ein weiteres Problem entsteht dadurch, dass beim Boarding des Flugzeuges meist viele Personen schnell hintereinander einsteigen wollen. Dieses wirkt sich nachteilig auf die sonst sehr guten Blickfelder der weiter entfernten Kamerapositionen (z.B. über gegenüberliegender Tür, siehe Abbildung 3.3) aus, da sich Personen, die das Flugzeug vorher betreten haben, noch im Blickfeld befinden können und somit die gerade am Bordkarten-Leser befindliche Person verdecken können. Als bester Kompromiss zwischen den verschiedenen Kameraposition hat sich die am nächsten am Eingang liegende herausgestellt (siehe Abbildung 3.5), deren Blickfeld

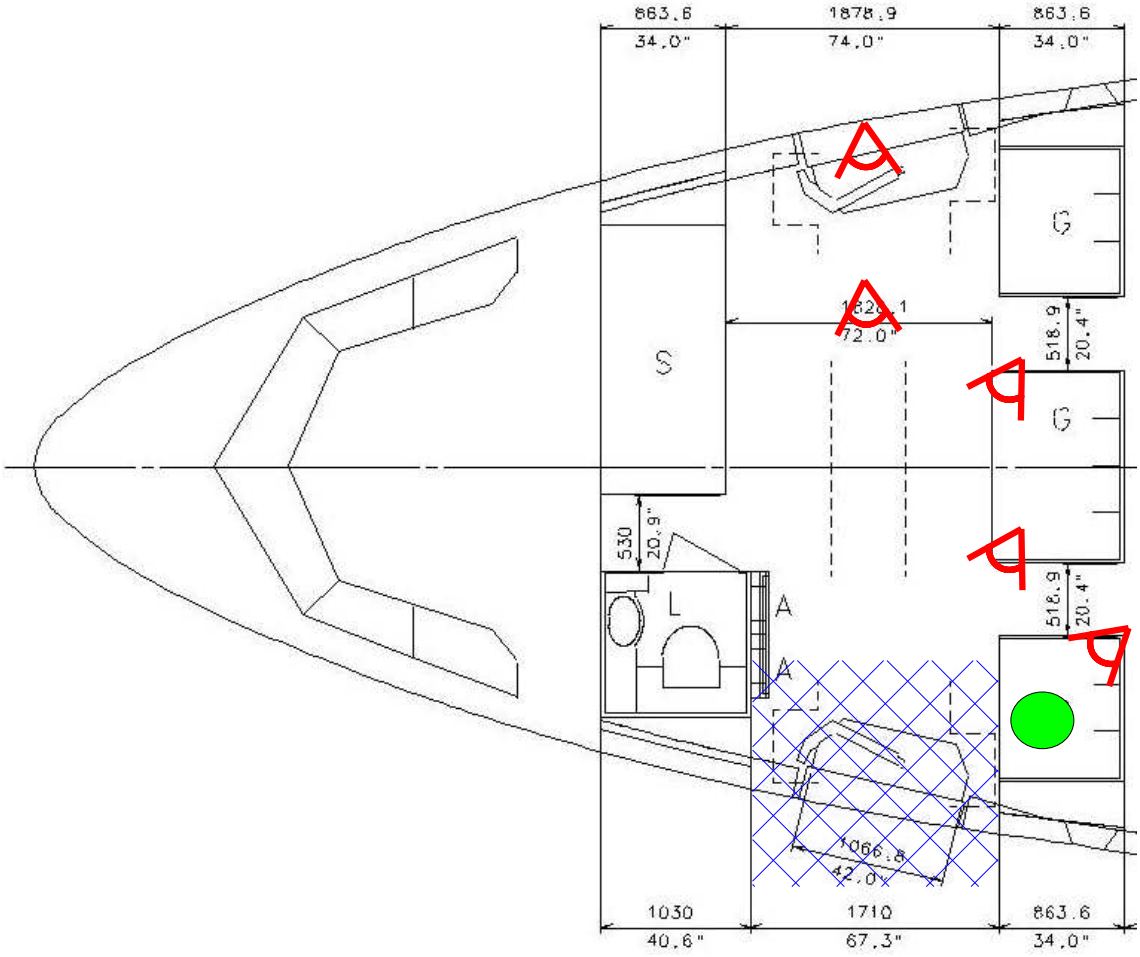


Abbildung 3.2: Mögliche Kamerapositionen (rot), der interessierende Eingangsbereich ist durch die blauen gekreuzten Linien dargestellt, die gewünschte Position des Lesegerätes für die Bordkarten ist durch den grünen Kreis dargestellt.

Abbildung 3.3: Erfassung des Eingangsbereich durch Kamerapositionen an der Decke.

Abbildung 3.4: Erfassung des Eingangsbereich durch Kamerapositionen an der Mittelkonsole zwischen den Durchgängen zum Kabinenbereich.

Abbildung 3.5: Erfassung des Eingangsbereich durch Kameraposition unmittelbar rechts neben der Eingangstür.

die Person an besten einfängt, während die Bordkarte ausgelesen wird. Das Blickfeld ist in diesem Fall so, dass keine weitere Person sich zwischen der Kamera und der zu erkennenden Person befindet. Leicht nachteilig ist, dass sich die Beleuchtungsverhältnisse von außerhalb und innerhalb des Flugzeuges für diese Kameraposition in einer Übergangssituation befinden. Ein weiterer durch Wahl dieses Blickfeldes entstehender Vorteil ist, dass die Aktivierung des Bordkarten-Lesers gleichzeitig als Auslöser des Erkennungsprozesses verwendet werden kann. Somit kann auf ein kontinuierliches Erkennen und Detektieren von Gesichtern verzichtet werden, welches insbesondere im vorliegenden Szenario problematisch ist, da ja laufend Gesichter den beobachteten Ausschnitt neu betreten oder verlassen.

3.3 Detektion

Als Detektionsmethode wurde ein Gesichtsdetektor basierend auf neuronalen Netzen verwendet (siehe 2.1.1), so wie er von Henry A. Rowley vorgeschlagen wurde

[RBK98]. Die konkrete Realisierung des Detektionsmoduls umfasst mehrere Anpassungen und Verbesserungen, die gezielt für das gegebene Flugzeugszenario entwickelt wurden. Die ergriffenen Maßnahmen umfassen u.a. eine Hintergrundsubtraktion mit der leeren Szene, eine Vorklassifikation interessanter Regionen durch einen Hautfarbenklassifikator, und das Einbetten des Detektors in einen Partikelfilter [IB98a]. Insgesamt konnte so ein robustes Detektionsmodul erzielt werden. Da im Rahmen dieser Arbeit keine Untersuchungen im Hinblick auf Modifikationen im Detektionsalgorithmus vorgenommen werden, wird an dieser Stelle auf eine ausführliche Erläuterung verzichtet. Interessierte finden eine genauere Beschreibung des hier eingesetzten Detektionsmoduls in [ASR07].
Die im Folgenden durchgeführten Untersuchungen sind unabhängig von der Detektionsmethode, so könnten für den Detektionsschritt natürlich auch andere "state of the art" Verfahren eingesetzt werden.

In den folgenden Sektionen wird nun detailliert erläutert, wie die Verarbeitungsschritte auf den gefundenen Gesichtsausschnitten aussehen, um am Ende der Verarbeitungskette eine Entscheidung bezüglich der Authentifikationsaufgabe zu erreichen.

3.4 Vorverarbeitung der gefundenen Gesichtsausschnitte

Die mit dem Gesichtsdetektor gefundenen Gesichtsausschnitte werden in Hinblick auf mehrere Aspekte normiert, um so anschließend eine möglichst robuste Klassifikation zu ermöglichen. Die zu normierenden Aspekte sind die Bildgröße, die Beleuchtung und eine Maskierung des Hintergrundes. Die im Rahmen dieser Arbeit durchgeführten Untersuchungen zur Vorverarbeitung orientieren sich dabei an den bereits in der Literatur vorhandenen Erkenntnissen [HPA02][JL05][QGSR01][GB03].

3.4.1 Wahl des Bildausschnitts

Um den Bildausschnitt zu normieren, wird auf die Daten zurückgegriffen, mit denen das neuronale Netz des Gesichtsdetektors trainiert wurde. Im vorliegenden Fall wurde der Detektor mit 20×20 Pixel großen Bilder trainiert. Diese wurden so ausgeschnitten, dass die Augen jeweils auf fixen Positionen lagen. Wenn der Gesichtsdetektor nun einen Ausschnitt liefert, werden die Augenpositionen näherungsweise implizit mitgeliefert. Die relativen Augenpositionen, die beim Detektortraining verwendet wurden, werden als Schätzungen für die Augenposition des aktuell detektierten Gesichts verwendet. Ein fest auf dem Augenabstand d_a basierender Ausschnitt wird nun ausgeschnitten und auf eine feste Größe skaliert (siehe Abbildung 3.6).

Die Koordinaten der Punkte für das linke Auge $\vec{p}_{la}$, und das rechte Auge $\vec{p}_{ra}$, jeweils als Ortsvektor, werden dabei für die Berechnung des Ausschnitts herangezogen. Zunächst werden der Augenabstand d_a und der auf die Länge 1 normierte Richtungsvektor $\vec{v}_a$ berechnet, der vom linken auf das rechte Auge zeigt.

$$d_a = |\vec{p}_{ra} - \vec{p}_{la}| \tag{3.1}$$

$$\vec{v}_a = \frac{\vec{p}_{ra} - \vec{p}_{la}}{d_a} \tag{3.2}$$

Anschließend wird ein dazu senkrecht stehender Richtungsvektor $\vec{v}_b$ ermittelt, der Richtung Mund zeigt, indem der Vektor $\vec{v}_a$ mit einer Rotationsmatrix multipliziert wird.

$$\vec{v}_b = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \vec{v}_a \tag{3.3}$$

Der Mittelpunkt $\vec{m}_a$ zwischen den Augen wird als Ankerpunkt für den auszuschneidenden Ausschnitt verwendet.

$$\vec{m}_a = \frac{\vec{p}_{la} + \vec{p}_{ra}}{2} \tag{3.4}$$

Als Parameter für die Wahl des Ausschnittes stehen die Höhe h und die Breite b zur Verfügung. Beide Parameter sind als auf den Augenabstand bezogene Faktoren zu verstehen. Der Ausschnitt selbst wird nun dadurch erstellt, dass ausgehend von dem Augenmittelpunkt ein Ausschnitt mittels der beiden Richtungsvektoren $\vec{v}_a$ und $\vec{v}_b$ so aufgespannt wird, dass ein resultierender Ausschnitt der Höhe $h \cdot d_a$ und der Breite $b \cdot d_a$ entsteht, für den der Augenmittelpunkt gleichzeitig der Mittelpunkt des Ausschnittes ist (siehe Abbildung 3.6). Typischerweise wählt man einen Ausschnitt, der etwas höher als breit ist, um die ungefähr elliptische Kopfform abzubilden.

Einige konkrete Ausschnitte mit verschiedene Breiten und Höhenparametern sind im Folgenden beispielhaft in Abbildung 3.7 abgebildet.

3.4.2 Helligkeitskorrekturen und Maskierung

Nachdem der Ausschnitt nun festgelegt wurde, wird als nächster Vorverarbeitungsschritt eine Maskierung und eine Helligkeitskorrektur vorgenommen. Die Maskierung hat dabei die Aufgabe, Randbereiche der Bildes, die für die hier auftretende Erkennungsaufgabe nicht von Interesse oder gar störend sind, auszublenden. Da das zu klassifizierende Gesicht keine dem Ausschnitt entsprechende rechteckige Form hat, sind in den Randbereichen der Ausschnitte oft Hintergrundelemente. Die Intensität oder Helligkeit dieser Randregionen kann in ganz anderen Wertebereichen sein als der interessierende Bereich. Es können also störende Einflüsse auf weitere Verarbeitungsschritte auftreten, besonders auf solche, die auf eine Normierung des Wertebereichs abzielen, wie die im Folgenden beschriebenen Helligkeitskorrekturen. Deshalb

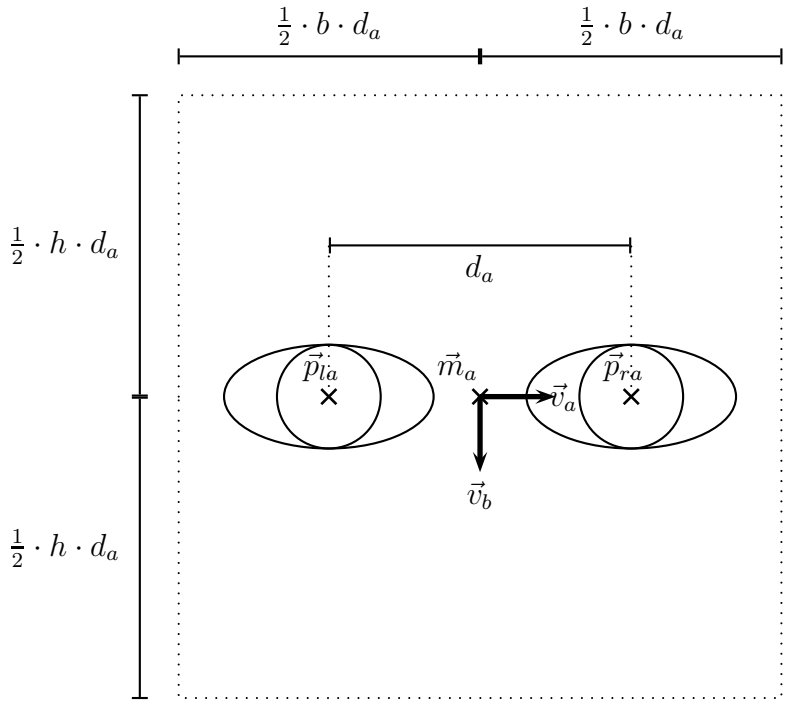


Abbildung 3.6: Der in diesem Ansatz verwendete Ausschnitt definiert sich basierend auf dem Augenmittelpunkt $\vec{m}_a$, dem Augenabstand d_a und der Richtungsvektoren $\vec{v}_a$ und $\vec{v}_b$. Diese Größen sind alle aus der Schätzung der Augenpositionen zu ermitteln. Zusätzlich werden die einstellbaren Parameter Breite b und Höhe h verwendet.

Abbildung 3.7: Erzeugung von Gesichtsausschnitten mit verschiedenen Parametern für Breite und Höhe; hier verwendete Parameterpaare (vom linken zum rechten Bild): $(b = 1.5, h = 2.8)$, $(b = 1.6, h = 2.4)$, $(b = 1.8, h = 2.8)$, $(b = 2.0, h = 2.5)$.

Abbildung 3.8: Maskierung der Randbereiche, Randpixel werden ausgeblendet, indem eine elliptische Maske diese auf den Wert Null (schwarz) setzt.

wird hier eine elliptische Maskierung verwendet, die alle Pixelwerte außerhalb eines durch Ellipsenform definierten Bereiches auf den Wert 0 setzt, siehe Abbildung 3.8.

Nachdem eine Maskierung den interessierenden Bereich hervorhebt, soll nun dessen Wertebereich normiert werden. Diese Maßnahme bezeichnet man in der Bildverarbeitung als Helligkeitskorrektur. In den hier durchgeführten Untersuchungen werden zwei verschiedene Helligkeitskorrekturen verglichen, der klassische Histogrammausgleich und eine Methode, die eine Vorverarbeitung durch Kombination von Zeilen- und Spaltenprojektionen mit dem ursprünglichen Bild erzielt.

Histogrammausgleich

Der Histogrammausgleich ist eine bekannte und verbreitete Vorverarbeitungsmethode [Jäh01][AR05][Rus02], die sich in vielen Anwendungen bewährt hat. Der Grundgedanke dieser Methode ist es, den Wertebereich eines Datensatzes dadurch zu normieren, dass die Häufigkeitsverteilung der einzelnen Werte so modifiziert wird, dass sie möglichst ähnlich einer Gleichverteilung wird, die den gesamten zur Verfügung stehenden Wertebereich ausschöpft. Anders ausgedrückt bedeutet dies, dass n Prozent der Bildpunkte in etwa n Prozent der Grauwerte verwenden sollen. Dieses Ziel lässt sich durch sogenannte Summenhäufigkeiten erreichen. Zu jeden Grauwert g bestimmt man die absolute Häufigkeit $a(g)$ seines Vorkommens im Bild. Wenn a_{ges} die Gesamtanzahl der Bildpunkte ist, dann ist die relative Häufigkeit $r(g)$ eines Grauwertes g:

$$r(g) = a(g)/a_{ges} \tag{3.5}$$

Die Summenhäufigkeit $s(g)$ ist dann die Summe aller Häufigkeiten vom Grauwert 0 bis zum Grauwert g. Die absolute Summenhäufigkeit $s_a(g)$ ergibt sich zu

$$s_a(g) = \sum_{i=0}^{g} a(i) \tag{3.6}$$

und die relative Summenhäufigkeit $s_r(g)$ ist

$$s_r(g) = \sum_{i=0}^{g} r(i). \tag{3.7}$$

Abbildung 3.9: Helligkeitsverbesserung durch Histogrammausgleich, in den Bildpaaren sind jeweils links der bereits maskierte Orginalausschnitt und rechts die histogrammausgeglichenen Ausschnitte.

Eine Gleichverteilung von Grauwerten würde zu einer linearen relativen Summenhäufigkeitsfunktion mit einer Steigung von $1/g_{max}$ führen, wenn von einem Wertebereich $[0..g_{max}]$ ausgegangen wird. Ziel ist es nun, jeden Grauwert g so abzubilden, dass die Summenhäufigkeitsfunktion der abgebildeten Grauwerte g^* der einer Gleichverteilung, nämlich g^*/g_{max}, entspricht.

$$s_r(g) = \sum_{i=0}^{g} \frac{a(i)}{a_{ges}} = \frac{\sum_{i=0}^{g} a(i)}{a_{ges}} \stackrel{!}{=} \frac{g^*}{g_{max}} \tag{3.8}$$

Daraus ergibt sich für jeden Grauwert g die Abbildung auf den neuen Grauwert g^* mittels folgender Abbildungsfunktion:

$$g^* = g_{max} \frac{\sum_{i=0}^{g} a(i)}{a_{ges}} \tag{3.9}$$

In Abbildung 3.9 werden einige Beispiele von histogrammausgeglichenen Gesichtsausschnitten (mit bereits angewendeter Maskierung) abgebildet.

Helligkeitskorrektur durch Zeilen- und Spaltenprojektionen

Der Helligkeitsausgleich durch Verrechnung des ursprünglichen Bildes mit seinen Zeilen- und Spaltenprojektion (engl. "Projection Combine")[WZ02][CV03][CZZ04] ist systemtheoretisch an das Projektions-Schnitt Theorem angelegt, welches die Beziehungen zwischen Projektionen und Schnitten im Orts- und Ortsfrequenzbereich beschreibt. Dabei gilt, dass eine Projektion auf eine Achse im Ortsbereich genau die Ortsfrequenzen beschreibt, die entlang einer dazu orthogonalen Achse im Ortsfrequenzbereich liegen. Bezogen auf Zeilen- und Spaltenprojektionen bedeutet dieses, dass Zeilenprojektionen die mittleren vertikalen Ortsfrequenzen beschreiben, Spaltenprojektionen die mittleren horizontalen Ortsfrequenzen. Der für die Helligkeitsnormierung zugrunde liegende Gedanke ist der, dass die mittleren Ortsfrequenzen charakteristisch zur Beschreibung der Beleuchtungssituation im Gesicht sind.

Da sich diese beiden Projektionen im rechteckigen Pixelraster sehr einfach durch Berechnung der Zeilen- oder Spaltensummen des Bildes I bestimmen lassen, bietet sich eine Verwendung an.

Ausgehend von einem Bild I der Größe $X \times Y$ Pixel ist die Zeilenprojektion $\vec{p_z}$ der Spaltenvektor der sich aus den Zeilensummen ergibt:

$$\vec{p_z} = \left(\sum_{x=1}^{X} I(x,1), ..., \sum_{x=1}^{X} I(x,Y) \right)^T \tag{3.10}$$

Die Spaltenprojektion $\vec{p_s}$ ist der Zeilenvektor der sich aus den Spaltensummen ergibt:

$$\vec{p_s}^T = \left(\sum_{y=1}^{Y} I(1,y), ..., \sum_{y=1}^{Y} I(X,y) \right) \tag{3.11}$$

Aus der Multiplikation von Zeilenprojektion (Spaltenvektor) und Spaltenprojektion (Zeilenvektor) ergibt sich die Projektion P

$$P = \vec{p_z} \cdot \vec{p_s}^T \tag{3.12}$$

Diese Projektion hat die Dimensionalität $X \times Y$, also die gleiche Anzahl von Pixeln, wie der Ausschnitt I, für den sie berechnet wurde.

Eine Helligkeitskorrektur wird nun dadurch vorgenommen, dass eine gewichtete Summe aus Originalbild I und seiner Projektion P gebildet wird:

$$I_{pc}(x,y) = I(x,y) + \alpha P(x,y) \tag{3.13}$$

Da diese Verrechnung den Wertebereich beeinflusst, wird anschließend noch eine Wertebereichskalierung vorgenommen, die linear auf den gesamten Wertebereich [0..255] abbildet ("linear stretching").

$$I_{pc}(x,y) \leftarrow (I_{pc}(x,y) - \min_{xy} I_{pc}(x,y)) \cdot \frac{255}{\max_{xy} I_{pc}(x,y) - \min_{xy} I_{pc}(x,y)} \tag{3.14}$$

Ein positives α verstärkt dabei durch Beleuchtungsunterschiede auftretende Helligkeitsschwankungen im Bild, ein negatives α schwächt dieses ab. Da allerdings die einfache Projektion auf mittlere horizontale und vertikale Ortsfrequenzen nur eine sehr grobe Näherung für Beleuchtungsvariation ist, und natürlich auch Eigenschaften des abgebildeten Individuums enthält, werden teilweise auch personenspezifische Charakteristika kompensiert. Deshalb wird dieses Verfahren zur Beleuchtungskompensation in der Regel nur dann eingesetzt, wenn man davon ausgeht, dass starke Helligkeitsgradienten in der Szene auftreten.

Einige Beispiele für Ergebnisse dieser Kompensation sind in Abbildung 3.11 zu sehen. Wie man sieht, kompensiert ein negatives α inhomogene Beleuchtung, während ein positives α diese noch verstärkt.

Die Kombination von "projection combine" mit der Eigenfacemethode ist unter extremen Beleuchtungsbedingungen sehr erfolgreich und wird in der Literatur auch als $(PC)^2A$ bezeichnet, siehe [WZ02][CV03][CZZ04].

Abbildung 3.10: Ein Gesichtsausschnitt unter extremen Beleuchtungsbedingungen und seine Projektion, die sich aus dem Produkt der Zeilen- und der Spaltenprojektion ergibt.

Abbildung 3.11: Die Ausschnitte nach der Beleuchtungskorrektur durch "Projection Combine". Die hier verwendeten Parameter α sind von links nach rechts -1, -0.5, +0.5 und +1.

3.5 Eigenfaces

Der grundsätzliche Eigenfaces Algorithmus (siehe [TP91a][TP91b]) wurde bereits in Kapitel 2 vorgestellt. Hier wird nun auf die Besonderheiten im vorliegenden Anwendungsfall eingegangen. Insbesondere der hier zum Einsatz kommende Gesichtsausschnitt, mit allen charakteristischen Merkmalen, wie Positionierung des Ausschnitts, Blickwinkel und typische Helligkeitsverteilung muss beim Erstellen der Eigenfaces berücksichtigt werden. Dieses lässt sich dadurch bewerkstelligen, dass die zum Training verwendeten Daten, also jene die zur Berechnung der zu verwendenden Hauptachsen herangezogen werden, auf möglichst ähnliche Weise ermittelt werden, wie die, die bei der Erkennungsaufgabe anliegen. Da allerdings für die Aufgabenstellung "Erkennung beim Betreten des Flugzeuges" keine Datenbank vorliegt und es ohne riesigen Aufwand kaum möglich ist, eine dafür repräsentative Datenbank zu erstellen, muss mit anderen bereits vorhandene Daten gearbeitet werden. Weiterhin soll in der gegebenen Anwendung die Erstellung eines Gesichtsmodells online geschehen, dass heißt, Personen, die am Schalter für einen Flug einchecken, sollen möglichst ohne längere Wartezeit ihre Bordkarte erhalten, auf der ja die Gesichtsrepräsentation gespeichert sein soll. Deshalb wurde hier entschieden, die Berechnung der Hauptachsentransformation offline auf ähnlichen Daten, wie sie in der Anwendung vorkommen, vorzubereiten. Im Anwendungsfall werden dann die bereits vorberechneten Eigenfaces zur Projektion verwendet. Diese Projektion ist dann die Gesichtsrepräsentation, die für die Erkennungsaufgabe verwendet wird.
Hier wurde dazu die FERET Datenbank [PMRR00] verwendet, die eine Vielzahl

von Personen mit unterschiedlichen Gesichtsausdrücken und in einigen unterschiedlichen Posen enthält. Zum Training verwendete Gesichtsausschnitte wurden mit den gleichen Vorverarbeitungsschritten ermittelt, wie sie bereits beschrieben wurden. Annahme hierbei ist es, dass die Datenbank hinreichend viele verschiedene Gesichter enthält, so dass die daraus berechneten Hauptachsen eine hinreichend gute Basis zur Beschreibung aller möglichen Gesichter sind.
Die zur Verfügung stehenden 1196 Gesichter in dieser Datenbank werden ausgeschnitten, normiert und auf 64×64 Pixel skaliert, anschließend wird jedes ausgeschnittene Gesichtsbild durch Aneinanderhängen der Bildspalten zu einem $N = 4096$-dimensionalen Spaltenvektor umgeformt. Dann wird die Hauptkomponentenanalyse für diese 1196 Spaltenvektoren durchgeführt bei der N zueinander orthogonale Eigenvektoren berechnet werden[1]. Nun stellt sich die Frage, wie viele dieser Eigenvektoren notwendig sind, um ein hinreichend gutes Beschreibungssystem für Gesichter zu erzeugen. Um diese Frage zu beantworten, wird untersucht, wie das Verhältnis der Varianz in Richtung der k verwendeten Eigenvektoren zur Gesamtvarianz der Trainingsdatenbasis ist. Da die Varianz in Richtung des i-ten Eigenvektors $\vec{e_i}$ durch seinen Eigenwert λ_i beschrieben wird, kann dieses Verhältnis $q(k)$ als Qualitätsmaß durch folgenden Ausdruck mathematisch beschrieben werden:

$$q(k) = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{N} \lambda_i} \tag{3.15}$$

Zur Umsetzung dieses Qualitätsmaßes werden die Eigenvektoren nach Größe ihrer Eigenwerte absteigend sortiert. In der Tabelle 3.1 wird für einige ausgewählte Qua-

Gefordertes $q(k)$	ohne Vorverarb.	Maskierung	HA	PC
0.90	60	41	86	122
0.95	133	100	178	232
0.98	273	223	336	400
0.99	393	335	459	519

Tabelle 3.1: Anzahl k der zu verwendenden Eigenvektoren, um ein vorgegebenes Qualitätsmaß q zu erzielen. Verschiedene Vorverarbeitungsschritte haben Einfluss auf die Varianz der Daten. "HA" steht für Histogrammausgleich (mit Maskierung), "PC" für "Projection Combine" (mit Maskierung).

litätsmaße q für die auf der FERET Datenbank ausgeführte PCA angegeben, wie

[1]Im vorliegenden Fall mit 1196 Trainingsbeispielen entstehen theoretisch nur 1195 Eigenvektoren mit einem Eigenwert ungleich 0, die numerischen Verfahren zur Berechnung beachten dies aber typischerweise nicht, und erzeugen so viele orthogonale Eigenvektoren, wie die Anzahl N der Dimensionen der Beispiele ist.

viele Eigenvektoren verwendet werden müssen, um diese zu erreichen. Es zeigt sich, dass die Vorverarbeitungsschritte offensichtlich starken Einfluss auf die Varianz der Daten haben. Sie sind in der Lage klassenspezifische Varianz zu erhöhen und klassenunspezifische Varianz zu verringern. Dieser Einfluss auf die Varianz zeigt sich insbesondere bei der Maskierung, bei der eine deutliche Reduktion zu erkennen ist. Die Helligkeitskorrekturen hingegen verstärken die Erkennbarkeit von klassenspezifischen Merkmalen und erhöhen somit die Varianz. Insgesamt ist die Vorverarbeitung als vorteilhaft zu deuten, da ja genau diese Erhöhung der klassenspezifischen Varianz bei gleichzeitiger Reduktion der klassenunspezifischen Einflüsse die Unterscheidbarkeit der einzelnen Klassen erhöht.

3.5.1 Untersuchungen zur Vorverarbeitung

Nun werden die verschiedenen Vorverarbeitungsschritte verglichen. Dafür wird für jeden Vorverarbeitungsschritt getrennt untersucht, welchen Einfluss er hat, anschließend wird die beste zu erzielende Kombination ermittelt. Als Maß, ob eine Vorverarbeitung sinnvoll ist, wird die Identifikationsrate, siehe Anhang A, herangezogen. Für diese Untersuchungen wurden jeweils 60 Eigenvektoren verwendet, als Abstandsklassifikator wurde die Mahalanobis-Distanz unter Verwendung der inversen Kovarianzmatrix verwendet, die aus allen Trainingsbeispielen ermittelt wurde.

In Tabelle 3.2 wird der Einfluss der Größe des Ausschnittes dargestellt. Hierfür wurden verschiedene Parameterpaare für Breite b und Höhe h ausgetestet, mit und ohne Maskierung. Da die Maskierung einen ähnlichen Einfluss hat, wie die Ausschnittsgröße, sie blendet Bildpunkte aus, wurden diese zwei Aspekte kombiniert untersucht.

Breite b	Höhe h	Erkennungsrate mit Maskierung	Erkennungsrate ohne Maskierung
1.5	2.0	0.7479	**0.76**
1.5	2.8	0.7806	-
2.0	2.0	0.7709	-
2.0	2.8	0.8061	0.7139
2.0	**3.0**	**0.8133**	0.7115
2.0	3.5	0.7503	0.6764
2.5	2.5	0.7187	-
2.5	3.0	0.6921	-
2.5	3.5	0.6813	-

Tabelle 3.2: Vergleich der Ergebnisse eines einfachen Eigenface Erkenners unter Verwendung verschiedener Ausschnitte.

Es zeigt sich, dass die Maskierung sich deutlich positiv auswirkt, vor allem bei den größeren Ausschnitten, die Randbereiche des Kopfes enthalten. Nur bei sehr kleinen Ausschnitten, die ausschließlich die inneren Elemente des Gesichtes zeigen, wirkt sich die Maskierung negativ aus, allerdings sind hier insgesamt die Erkennungsergebnisse auf einem niedrigeren Niveau. Insgesamt kann hier aufgrund der erreichten Ergebnisse empfohlen werden, die Maskierung weiterhin einzusetzen. Da dieser Trend eindeutig ist, wird auf weitere Experimente ohne Maskierung verzichtet, da jedes Erkennungsexperiment ja ein zeitaufwendiges Training auf der gesamten Datenbank voraussetzt.

Weiterhin ist ein grundsätzlicher Trend, je größer der Ausschnitt, desto besser das Ergebnis, zu erkennen, der allerdings nur bis zu einer bestimmten Größe gilt. Für Ausschnitte über diese Größe hinaus fallen die Erkennungsergebnisse rapide ab. Das lässt sich leicht dadurch erklären, dass es natürlich vorteilhaft ist, soviel personenspezifisches wie möglich im Ausschnitt enthalten zu haben, also möglichst den ganzen Kopf. Ab einer gewissen Größe steigt dann allerdings der Anteil des Hintergrundes, der selbstverständlich keinerlei personenspezifische Merkmale enthält. Dadurch dass die Ausschnitte auf eine fixe Größe von 64×64 Pixel skaliert werden, nimmt also ab einer bestimmten Größe der Anteil der unspezifischen Informationen an der Gesamtinformation zu, der Anteil der personenspezifischen Information somit ab, dieses wirkt sich natürlich auf die Erkennungsergebnisse aus. Insgesamt haben Ausschnitte mit einer Breite von $b = 2.0$ und einer Höhe von $h = 3.0$ mal Augenabstand die besten Ergebnisse erzeugt, so dass diese Parameter nun festgesetzt werden und nun, basierend auf dem dadurch definierten Ausschnitt, die Helligkeitskompensation und direkte Einstellungen des Abstandsklassifikators untersucht werden (Anzahl der Eigenvektoren, Abstandsmaß).

Um den Einfluss der Helligkeitskorrektur zu untersuchen, wird also die bisher beste Ausschnittsgröße unter Verwendung der verschiedenen Korrekturverfahren untersucht. Hier wurden die beiden beschriebenen Methoden Histogrammausgleich und Ausgleich durch Verwendung der Zeilen- und Spaltenprojektionen ("Projection Combine") verglichen (siehe Tabelle 3.3). Es zeigt sich, dass der Histogrammausgleich im Vergleich die besten Ergebnisse liefert. Das komplexere Helligkeitskorrekturverfahren, die Kombination mit Projektionen, ist für extremere Beleuchtungsbedingungen konzipiert, als sie in der gegebenen Datenbank auftreten und kann in der gegebenen Versuchsreihe seine Wirkung nur begrenzt entfalten. Im Folgenden wird der Histogrammausgleich für die weiteren Experimente verwendet.

Methode	Erkennungsrate
Histogrammausgleich	**0.8630**
Projection Combine (+1.0)	0.8448
Projection Combine (+0.5)	0.8424
Projection Combine (-0.5)	0.8521
Projection Combine (-1.0)	0.8375
ohne Helligkeitskorr.	0.8133

Tabelle 3.3: Erkennungsraten bei verschiedenen Helligkeitskorrekturen als Vorverarbeitung.

3.5.2 Untersuchungen zum Abstandsklassifikator

Nachdem die allgemeine Vorverarbeitung untersucht wurde, gilt es nun den Abstandsklassifikator, der anhand der Projektionen der Gesichter auf die Hauptachsen entscheidet, einzustellen. Hier gilt es festzulegen, wie viele Eigenvektoren und welches Abstandsmaß zu verwenden sind, auch im Hinblick darauf, dass für jede Anwendung ein Kompromiss zwischen Rechenzeit, Speicherbedarf und Erkennungsleistung gefunden werden muss.

Da bei der Projektion auf k verwendete Eigenvektoren, eine Datenreduktion des Gesichtsausschnitts auf k Gewichte durchgeführt wird und dieses k wesentlich kleiner ist, als die ursprünglichen Bilddimensionen, ist der Speicherplatzbedarf für den Eigenface Ansatz sehr vorteilhaft. Es wird nur ein Parameter pro Eigenvektor benötigt. Auch die Rechenzeit der Berechnung von Projektionen oder der Abstandsmaße zwischen Projektionen ist sehr gering, hier geht die Anzahl k der verwendeten Vektoren linear ein. Also geht es hier zunächst nur darum, die Erkennungsleistung zu maximieren. Deshalb wird nun verglichen, wie sich die Erkennungsrate auf der FERET Datenbank in Abhängigkeit der Anzahl der verwendeten Eigenvektoren verhält. Dazu werden mehrere Erkennungsexperimente mit jeweils verschiedener Anzahl k durchgeführt, die Ergebnisse sind in Tabelle 3.4 dargestellt.

Es zeigt sich, dass die Anzahl verwendeter Eigenvektoren k einen erheblichen Einfluss auf die Klassifikationsleistung hat. Wenn zu viele Eigenvektoren zum Aufspannen des linearen Unterraums verwendet werden, werden offensichtlich zunehmend unwichtige Merkmale beschrieben, wodurch die Klassifikationsleistung sinkt. Werden zu wenig Eigenvektoren verwendet, können diskriminative charakteristische Details, die nur selten auftreten, und somit nicht in den ersten k Eigenvektoren enthalten sind, nicht im Merkmalsraum abgebildet werden. Gute Erkennungsraten in der Größenordnung von ca. 86% konnten unter Verwendung von etwa 60 bis 120 Eigenvektoren erzielt werden. Es zeigt sich weiterhin deutlich, dass die Verwendung

Anzahl k der Eigenfaces	Mahalanobis-Distanz	Euklidische Distanz
20	0.7527	0.7515
40	0.7855	0.7745
50	0.8496	-
60	0.8581	0.8157
80	0.8606	0.8255
100	**0.8618**	0.8339
120	0.8581	0.8339
140	0.8533	0.8388
160	0.8509	0.8375
180	0.8388	0.8412
200	0.8303	0.**8424**
300	0.7890	0.8218
400	0.7381	-
500	0.6848	-

Tabelle 3.4: Ergebnisse der Erkennung mittels Eigenface Methode in Abhängigkeit der Anzahl k der verwendeten Eigenfaces. Die Vorverarbeitung wurde hier bereits optimiert, es wurde ein Bildausschnitt der Höhe $h = 3.0$ und der Breite $b = 2.0$ verwendet, dazu eine Maskierung und ein Histogrammausgleich.

der Mahalanobis-Distanz für diese Anzahl von Eigenvektoren deutlich vorteilhaft ist, und insgesamt die besseren Erkennungsergebnisse hervorbringt.

3.5.3 Identifikation und Authentifikation

Für die Identifikationsaufgabe wurde zunächst eine Datenbank angelegt, in der die zu erkennenden Klassen hinterlegt sind. Für jede Klasse i, also für jedes Individuum, wird die Projektion $\vec{w}_i$ auf die Eigenvektoren gespeichert. Wenn nun ein zu identifizierender Bildausschnitt vorliegt, wird für diesen die Projektion $\vec{w}$ berechnet, und die Projektion $\vec{w}_j$ aus der Datenbank ermittelt, für die der Abstand zu $\vec{w}$ minimal ist. Der untersuchte Gesichtsausschnitt wird dann der Klasse j zugeordnet.

$$j = \underset{i=1..K}{\mathbf{arg\,min}}\, d(\vec{w}_i, \vec{w}) \tag{3.16}$$

Im Unterschied zur Identifikation liegen bei der Authentifikation nur genau zwei Projektionen $\vec{w}_a$ und $\vec{w}_b$ vor. Die erste Projektion $\vec{w}_a$ beschreibt die Identität, für die sich die Person ausgibt, die zweite, $\vec{w}_b$, diejenige, die aus einem vorliegenden Bildausschnitt ermittelt wurde. Um nun die Entscheidung zu treffen, ob die Person

diejenige ist, für die sie sich ausgibt, wird überprüft, ob eine Mindestähnlichkeit zwischen $\vec{w}_a$ und $\vec{w}_b$ vorliegt. Dieses wird durch einen Schwellwertklassifikator auf dem Distanzmaß d realisiert:

$$A(\vec{w}_a, \vec{w}_b) = \begin{cases} \textbf{gleiche Person} & \textbf{für} \quad d(\vec{w}_a, \vec{w}_b) < T \\ \textbf{verschiedene Person} & \textbf{für} \quad d(\vec{w}_a, \vec{w}_b) \geq T \end{cases} \tag{3.17}$$

Der Schwellwert T wird dabei aus den Trainingsdaten ermittelt, und ist in der Regel der Wert, der die Equal Error Rate erreicht (siehe Anhang A), also der Wert an dem sich Falsch-Positiv-Rate und Falsch-Negativ-Rate schneiden. Da die Einstellung des Schwellwertes T eine Ergebnisoptimierung hinsichtlich des Datensets darstellt, auf dem sie vorgenommen wird, wird sie in der Praxis ausschließlich auf den Trainingsdaten vorgenommen, der ermittelte Schwellwert wird dann in der Anwendungsphase übernommen.

Im Folgenden wird die Authentifikation auf der FERET Datenbank für den Fall "Jeder gegen Jeden" durchgeführt. Hier wird also jedes Bild aus dem Testset mit allen vorhandenen Bildern aus den Trainingsdaten verglichen. Dieses sorgt dafür, dass die Anzahl der Bildpaare, die unterschiedliche Gesichter zeigen, deutlich größer ist, als die Anzahl der Positivbeispiele, dies ist natürlich eine herausfordernde Authentifikationsaufgabe. In einigen anderen Studien werden teilweise pro Positivbeispiel nur ein Negativbeispiel durch Zufallsauswahl ausgewählt, dadurch lassen sich natürlich viel bessere Raten erzeugen. Um allerdings die Evaluierung im Gegensatz zu einer Zufallsauswahl reproduzierbar zu gestalten, wurde hier die Entscheidung getroffen, alle aus der Datenbank erzeugbaren Bildpaare zu evaluieren. Anhand der Diagramme (siehe Abbildung 3.12 und Abbildung 3.13) in dem die Falsch-Positiv- und die Falsch-Negativ Raten aufgezeigt werden, wird die Equal Error Rate (EER) ermittelt. Die hier aufgezeigten Kurven wurden jeweils durch 100 relative Schwellwerte T auf der Mahalanobis-Distanz oder auf der euklidischen Distanz im Bereich $T \in [0, 1]$ ermittelt, die sich jeweils zwischen dem maximalen und dem minimalen auftretenden Distanzwerten befinden.

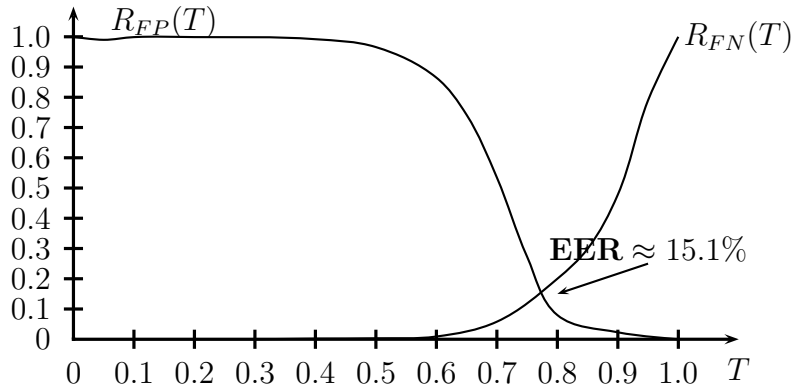


Abbildung 3.12: Bestimmung der EER unter Verwendung der Mahalanobis-Distanz durch Veränderung des Schwellwertes T. Die sich zueinander gegenläufig verhaltenden FP-Rate R_{FP} und FN-Rate R_{FN} schneiden sich bei der EER.

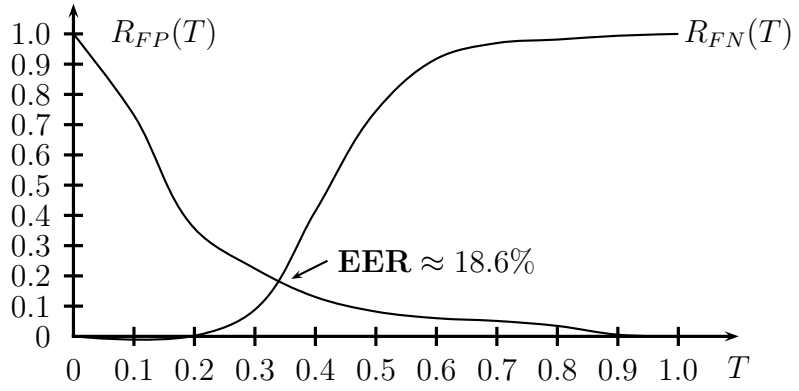


Abbildung 3.13: Bestimmung der EER unter Verwendung der Euklidischen Distanz durch Veränderung des Schwellwertes T. Die sich zueinander gegenläufig verhaltenden FP-Rate R_{FP} und FN-Rate R_{FN} schneiden sich bei der EER.

3.6 Pseudo 2-Dimensionale HMM

Für den HMM Ansatz wurden die gleichen Vorverarbeitungsmechanismen verwendet, wie für die Eigenfaces. Um das optimale Modell für die vorliegende Anwendung zu finden, wurden verschiedene Modellstrukturen, also Strukturen mit verschiedener Anzahl von Zuständen, sowie unterschiedliche Merkmalsextraktionsverfahren und Repräsentationen der Beobachtungswahrscheinlichkeiten, verwendet. Um HMM sinnvoll zu trainieren, benötigt man pro Klasse mehrere Beispiele. Da in der FERET Datenbank für die verwendete Ansicht leider nur eine Mindestanzahl von einem Bild pro Person vorliegt, wird zunächst ein personenunabhängiges Gesichtsmodell unter Verwendung aller zur Verfügung stehenden Bilder erzeugt. Dieses personenunabhängige Modell wird dann zu einem personenabhängigen Modell adaptiert, indem es mit Bildern der jeweiligen Person nachtrainiert wird, die erzeugt werden, indem das eine Trainingsbeispiel zusätzlich durch zufällige Translation um ±3 Pixel, Rotation um ±5° und Skalierung um ±5% variiert wird.

3.6.1 Blockweise Merkmalsextraktion

Um einen vorliegenden Gesichtsausschnitt mit einem HMM bearbeiten zu können, muss das zweidimensionale Bild in eine eindimensionale Abfolge von Merkmalsvektoren umgeformt werden (siehe 2.3.3). Um dieses zu Erreichen, wird das Bild in sich um d_o Pixel überlappende Blöcke der Größe $s_b \times s_b$ Pixel zerlegt. Die Merkmalsextraktion wird dann auf jedem der N_b Blöcke einzeln vorgenommen, daraus entsteht jeweils ein Merkmalsvektor $\vec{f_i}$ pro Block B_i. Wenn das Bild die Höhe von H und die Breite von B Pixeln hat, dann ist die Gesamtanzahl N_b der Blöcke die Anzahl der Blöcke pro Spalte N_{bs} mal der Anzahl der Blöcke pro Zeile N_{bz}

$$N_b = \frac{B - d_o}{s_b - d_o}\frac{H - d_o}{s_b - d_o} = N_{bz} \cdot N_{bs} \tag{3.18}$$

Ein resultierender Gesamtmerkmalsvektor, also die Sequenz der einzelnen Merkmalsvektoren, wird dann dadurch erzeugt, dass die blockweisen Merkmalsvektoren $\vec{f_i}$ von oben nach unten spaltenweise aneinander gehängt werden. Zu Beginn jeder Spalte wird ein eindeutiger Wert als Marker eingefügt, anschließend werden die resultierenden spaltenweisen Merkmalssequenzen von links nach rechts aneinandergehängt. Dieser Vorgang ist in Abbildung 3.14 beschrieben.

Dieses Grundschema kann für alle möglichen verschiedenen Merkmalstypen verwendet werden. Im folgenden werden nun die Merkmale vorgestellt, die tatsächlich für die gegebene Aufgabenstellung verwendet wurden.

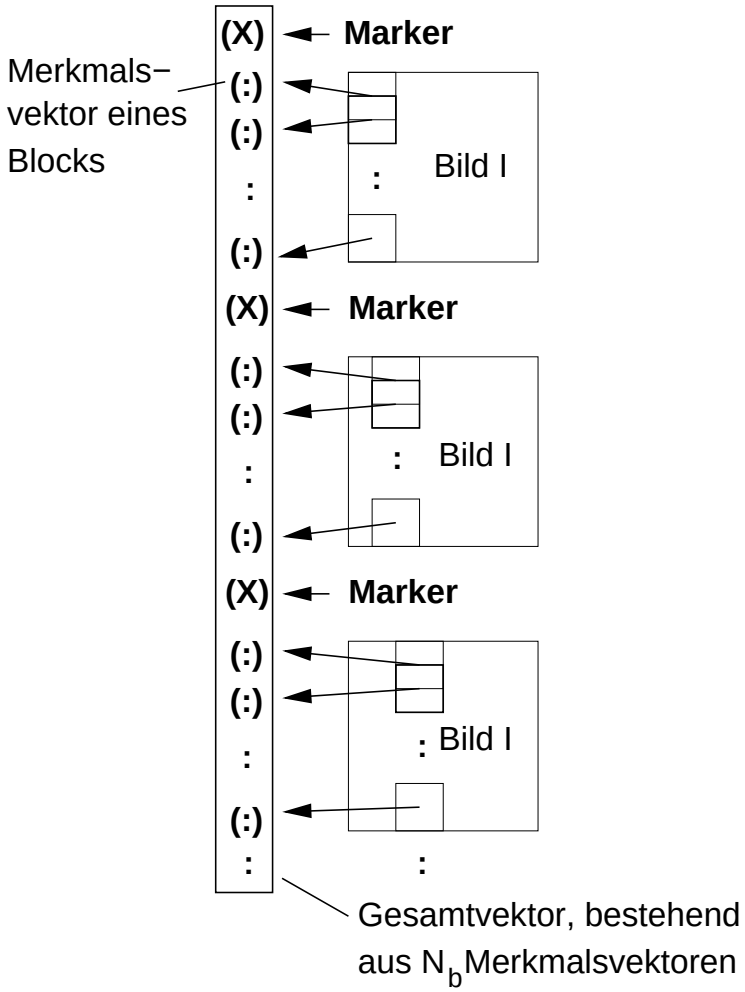


Abbildung 3.14: Erzeugen eines Gesamtmerkmalsvektors als Sequenz der einzelnen blockweise extrahierten Merkmalsvektoren.

DCT Merkmale

Die Diskrete Kosinustransformation (DCT, engl. "Discrete Cosine Transformation") ist eine weit verbreitete und sehr bekannte Transformation. Der Grundgedanke ist es, ein gegebenes Signal durch eine Linearkombination orthogonaler Basisfunktionen zu beschreiben. Diese Basisfunktionen setzen sich in diesem Fall durch Produkte horizontaler und vertikaler Kosinusfunktionen zusammen.

$$C_{i,j} = \frac{2C(i)C(j)}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} I(x,y) \cos\left(\frac{(2x+1)i\pi}{2N}\right) \cos\left(\frac{(2y+1)j\pi}{2N}\right) \tag{3.19}$$

hier ist $C_{i,j}$ der DCT-Koeffizient für die i-te horizontale und j-te vertikale Kosinusfrequenz. $C(i)$ und $C(j)$ sind konstante Vorfaktoren, mit

$$C(n) = \begin{cases} \frac{1}{\sqrt{2}} & \textbf{für} \quad n = 0 \\ 1 & \textbf{für} \quad n \neq 0 \end{cases} \tag{3.20}$$

Für $(i,j) = (0,0)$ handelt es sich um den Gleichanteil, je höher i oder j werden, desto höher wird die horizontale oder vertikale Kosinusfrequenz. Da es sich bei der Berechnung von DCT-Koeffizienten um eine weit verbreite Standardmethode handelt, wird auf eine weitere ausführliche Erläuterung verzichtet, für Interessierte sei auf die Literatur verwiesen [ANR74][BL90].

DCTmod2 Merkmale

DCTmod2 Merkmale sind DCT-Koeffizienten, für die die Koeffizienten für den Gleichanteil $C_{0,0}$, sowie für die niedrigste horizontale und vertikale Kosinusfrequenz ($C_{1,0}$ und $C_{0,1}$) durch die horizontalen und vertikalen partiellen Ableitungen ersetzt werden. Diese Ableitung wird typischerweise durch die Differenz zwischen benachbarten Blöcken approximiert, so dass statt $C_{0,0}, C_{1,0}, C_{0,1}$ im neuen DCTmod2-Merkmalsvektor $\Delta_x C_{0,0}, \Delta_y C_{0,0}, \Delta_x C_{1,0}, \Delta_y C_{1,0}, \Delta_x C_{0,1}, \Delta_y C_{0,1}$ steht, dann die Merkmalssequenz wieder mit den normalen DCT-Koeffizienten fortgesetzt wird.
DCTmod2 Merkmale sind besonders wegen Ihrer Robustheit gegenüber gerichteter Beleuchtung eingeführt worden und führen in gezielten Untersuchungen zu besseren Ergebnissen als DCT-Koeffizienten (siehe [SP03]). Deshalb wurden sie auch hier getestet.

3.6.2 Modellierung der Beobachtungswahrscheinlichkeit

Die Beobachtungswahrscheinlichkeit $b_j(o)$ eines jeden Zustandes S_j kann entweder diskret oder kontinuierlich modelliert werden. Im kontinuierlichen Fall werden gängigerweise Gaußmixturen $\mathrm{N}(\mu, \Sigma)$ mit dem Mittelwert μ und der Kovarianzmatrix Σ

verwendet, um von einem Eingangswert auf dessen Beobachtungswahrscheinlichkeit abzubilden:

$$b_j(o) = \sum_{k=1}^{K} c_{jk} \cdot \mathrm{N}(o, \mu_{jk}, \Sigma_{jk}) \tag{3.21}$$

Die kontinuierliche Modellierung des Merkmalsraums bietet sich vor allen dann an, wenn die tatsächliche Verteilung der Merkmale normalverteilt ist oder aus Normalverteilungen N zusammengesetzt ist. Je höher die Anzahl der verwendeten Mixturen, desto komplexer ist natürlich auch die abbildbare Verteilung, aber ebenso die Berechnung.

Eine zweite Möglichkeit der Modellierung ist die Verwendung diskreter Beobachtungswahrscheinlichkeiten. Hier wird der Merkmalsraum durch Vektorquantisierung auf eine endliche Anzahl von Symbolen abgebildet. Jedes dieser Symbole wird nun mit einer Beobachtungswahrscheinlichkeit versehen, somit bildet sich eine diskrete Wahrscheinlichkeitsverteilung.

$$b_j(o) = [b_j[o_1], .., b_j[o_C]] \tag{3.22}$$

Da die Berechnung der Beobachtungswahrscheinlichkeit $b_j[o]$ für eine gegebene Beobachtung o sich zu einem Ablesen der diskreten Wahrscheinlichkeiten $[b_j[o_1], .., b_j[o_C]]$ reduziert, kann ohne erhöhten Rechenaufwand eine hohe Anzahl von Symbolen verwendet werden. Diese Anzahl wird in der Regel Kodebuchgröße C genannt.
Das Problem, die auftretende Merkmalsverteilung geeignet zu quantisieren, wird oft mit Unüberwachtem Lernen (engl. “unsupervised learning”) angegangen. Die wohl bekannteste Methode ist der k-means Algorithmus [Mac67], der ausgehend von einer Initialisierung, immer abwechselnd Meßpunkte Clusterzentren zuordnet und dann aufgrund dieser Zuordnung die Position der Clusterzentren aktualisiert. Die Zuordnung wird dabei durch den geringsten Abstand ermittelt und die Position des Clusterzentrums durch Mittelwertbildung der ihm zugeordneten Meßpunke berechnet. Dieses Standardverfahren wurde auch hier in dieser Arbeit verwendet, dabei wird jedes Cluster als ein beobachtbares Symbol interpretiert. Um die Wahrscheinlichkeit der Beobachtung o im Zustand S_j zu ermitteln wird o zunächst quantisiert ($o \rightarrow [o_1, ..o_C]$), anschließend wird die Wahrscheinlichkeit des resultierenden Clustersymbols in der Wahrscheinlichkeitstabelle $b_j[o]$ nachgeschlagen.
Ein klassisches Dilemma bei der Verwendung diskreter Beobachtungswahrscheinlichkeiten ist, dass diese durch relative Häufigkeiten der in den Trainingsdaten vorkommenden Meßwerte geschätzt werden. Je höher die Kodebuchgröße C, also je mehr Clusterzentren verwendet werden, desto weniger Trainingsbeispiele pro Clusterzentrum, daraus folgt eine weniger signifikante Schätzung der relativen Häufigkeiten. Andererseits gilt, je komplizierter die Merkmalsverteilung, desto mehr Clusterzentren werden benötigt, um diese geeignet beschreiben zu können, zumindest, wenn

man darauf verzichten möchte Expertenwissen über die Merkmalsverteilung explizit bei der Quantisierung zu berücksichtigen. Diese beiden Problematiken sind zueinander gegenläufig, es gilt also durch Experimente die beste Kodebuchgröße zu ermitteln.
Für den vorliegenden Fall wurde eine diskrete Modellierung der Beobachtungswahrscheinlichkeit verwendet, da bereits nach wenigen Versuchen klar wurde, dass die zu erzielenden Ergebnisse trotz klar geringerer Rechenzeit gleich oder besser waren als eine Modellierung mit kontinuierlichen Beobachtungswahrscheinlichkeiten. Diese Beobachtung korrespondiert mit den Ergebnissen früherer Untersuchungen (siehe [Wal06], [WER01]). Genauere Angaben zu den durchgeführten Experimenten und den daraus gewonnen Erkenntnissen finden sich in Sektion 3.6.4.

3.6.3 Parameter des Modells

Bei der Anwendung von P2D-HMM sind unzählige Einstellmöglichkeiten vorhanden. Die im Rahmen dieser Arbeit untersuchten Parameter sind die Anzahl der Zustände des Modells, die Anzahl der verwendeten Kodebucheinträge, also die Anzahl der Quantisierungsstufen zur Modellierung der Beobachtungswahrscheinlichkeit, Art der Merkmale und der Überlappungsgrad der Blöcke bei der Merkmalsextraktion.
Um eine geeignete Wahl der Anzahl der Zustände des Modells zu erzielen, wurden verschiedene Konfigurationen ausprobiert. Die gewählte Notation ist dabei folgende: Ein P2D-HMM mit N Spaltenmodellen die jeweils mit M Zuständen modelliert werden, wird im folgenden als $N \times M$- Modell bezeichnet, Markerzustände werden dabei nicht mitgezählt.
Es wird ebenfalls berücksichtigt, dass eine erhöhte Anzahl von Zuständen zu stark erhöhten Rechenzeiten beim Training und Erkennen führt. Aufgabenstellung hierbei ist es, den besten Kompromiss zwischen Rechenaufwand und Erkennungsleistung zu finden. Da in der gegebenen Aufgabenstellung nicht nur die Erkennung online stattfinden soll, sondern auch das Training (Check-In am Schalter, Speicherung des Modells auf der Bordkarte), ist diese Rechenzeit für die gegebene Anwendung stark eingeschränkt. Im Folgenden werden einige Experimenten aufgelistet, die durchgeführt wurden, um die Modellparameter einzustellen.

3.6.4 Experimente

Zunächst werden hier einige durchgeführte Ergebnisse aufgelistet, dabei wurden Versuche mit verschiedenen Parametereinstellungen für die Merkmalsextraktion, Anzahl der Zustände und Kodebuchgröße verwendet. Als Erstes werden einige direkte Vergleiche angestellt, ob die Merkmalsextraktion mit DCTmod2 Merkmalen wirklich einen Vorteil gegenüber den klassischen DCT Merkmalen bietet (Tabelle 3.5).

Da in allen Versuchen DCTmod2-Merkmale zu deutlich besseren Erkennungsergebnissen führen (siehe Tabelle 3.5), werden die folgenden Untersuchung alle un-

Merkmal	Überlappung	Kodebuchgröße	Zustände	Erkennungsrate
DCT	75%	500	5×5	0.9163
DCTmod2	75%	500	5×5	0.**9657**
DCT	75%	2000	4×4	0.9102
DCTmod2	75%	2000	4×4	0.**9264**
DCT	75%	2000	7×7	0.9364
DCTmod2	75%	2000	7×7	0.**9816**

Tabelle 3.5: Vergleich der beiden Merkmalsextraktionsmethoden, insgesamt schneidet DCTmod2 deutlich besser ab als DCT.

Merkmal	Überlappung	Kodebuchgröße	Zustände	Erkennungsrate
DCTmod2	50%	500	5×5	0.9046
DCTmod2	**75%**	500	5×5	0.**9657**

Tabelle 3.6: Vergleich der Erkennungsergebnisse bei verschiedenen Überlappungsgraden.

ter Verwendung dieser Merkmalsextraktion durchgeführt. Als nächstes werden nun Experimente mit verschiedenen Überlappungsgraden ausgeführt, also um wie viele Pixel die Blöcke, aus denen die Merkmale extrahiert werden, benachbarte Blöcke überlappen. Verglichen werden hier die Überlappungsgrade von 50% und 75%. Bei einer Blockgröße von 8×8 Pixeln bedeutet dies 4 oder 6 Pixel Überlappung.

In Tabelle 3.6 erkennt man deutlich, dass eine Überlappung von 6 Pixeln, also $6/8 = 75\%$ gegenüber geringeren Überlappungsgraden vorzuziehen ist. Eine Überlappung von 7 Pixeln also 7/8 wäre theoretisch auch möglich, wird aber hier nicht untersucht, da sich der Überlappungsgrad natürlich direkt auf die Länge der extrahierten Merkmalsequenz auswirkt. Ausgehend von einem 64×64 Pixel großem Bild, werden bei einer Blockgröße von 8 Pixeln bei einer Überlappung von 50% genau 15×15 Merkmalsvektoren extrahiert, bei 75% bereits 29×29 und bei 7/8 wären es sogar 57×57 Merkmalsvektoren. Da die Länge der Merkmalssequenz starken Einfluss auf Trainings- und Erkennungszeiten hat, sind diese Dimensionen für den praktischen Einsatz nicht sinnvoll.

Als nächstes wird in Tabelle 3.7 der Einfluss der Kodebuchgröße C betrachtet. Es zeigt sich, dass eine höhere Kodebuchgröße tendenziell bessere Ergebnisse liefert, allerdings nimmt die Erkennungsrate bei zu großer Anzahl Kodebucheinträge wieder ab. Die beste Leistungsfähigkeit wurde hier bei der Verwendung von 2000 Einträgen erreicht (siehe Tabelle 3.7).

Merkmal	Überlappung	Kodebuchgröße	Zustände	Erkennungsrate
DCTmod2	75%	500	5×5	0.9657
DCTmod2	75%	1000	5×5	0.9687
DCTmod2	75%	**2000**	5×5	**0.9799**
DCTmod2	75%	3000	5×5	0.9615

Tabelle 3.7: Vergleich der Erkennungsergebnisse unter Verwendung verschiedener Kodebuchgrößen.

Betrachtungen zum Speicherbedarf und zur Rechenzeit

Da in der Anwendung "Authentifikation im Flugzeug" geplant ist, das Modell jeder Person auf der Bordkarte zu speichern, werden im Folgenden allgemeine Betrachtungen zum Speicherbedarf eines Gesichtstemplates gemacht. Weiterhin wird verglichen, wie hoch die Rechenzeit zur Berechnung der Produktionswahrscheinlichkeit eines bereits vorverarbeiteten Merkmalsvektors bei einem gegebenen Modell ist. Dazu werden jeweils 1000 solcher Produktionswahrscheinlichkeiten berechnet und die benötigte Zeit darüber gemittelt, um die Größenordnungen abzuschätzen.

Um den Speicherbedarf zu ermitteln wird sich im Folgenden überlegt, wie viele Parameter gespeichert werden müssen, in Abhängigkeit der Modelleinstellungen. Zunächst wird betrachtet, wie viel Speicherplatz ein Zustand benötigt. Die Parameter innerhalb eines Zustandes im diskreten HMM sind die Emmisionswahrscheinlichkeiten für jedes mögliche Symbol. Im vorliegenden Fall sind das so viele Symbole, wie die Kodebuchgröße C. Für jeden Kodebucheintrag muss also pro Zustand eine Gleitkommazahl gespeichert werden. Weiterhin müssen alle Übergangswahrscheinlichkeiten gespeichert werden. Das bedeutet für jeden Zustand innerhalb eines Spaltenmodells eine Selbsttransition und eine Vorwärtstransition, im Fall des letzten Zustandes einer Spalte ist diese Vorwärtstransition die spaltenweise Selbsttransition. Zusätzlich entsteht pro Spaltenmodell eine spaltenweise Vorwärtstransition. Bei N Spaltenmodellen mit jeweils M Zuständen bedeutet dies, dass weitere $2MN + N$ Gleitkommazahlen gespeichert werden müssen. Insgesamt ergibt sich also, dass bei einem $N \times M$ Modell,

$$n = N \cdot M \cdot C + 2MN + N \tag{3.23}$$

Gleitkommazahlen gespeichert werden müssen. Wenn davon ausgegangen wird, dass pro Gleitkommazahl 4 Byte benötigt werden, ergibt sich für die ausgewählten Beispiele jeweils folgender Speicherbedarf (siehe Tabelle 3.8).

Durch geeignete Kompressionsverfahren können diese Werte noch deutlich reduziert werden, vor allem da ja viele Einträge von Emmisionswahrscheinlichkeiten Null

Zustände	Kodebuchgröße	Speicherbedarf
3×3	1000	36108 Byte
5×5	500	50220 Byte
5×5	1000	100220 Byte
7×7	2000	392420 Byte

Tabelle 3.8: Speicherbedarf für einige beispielhafte P2D-HMM-Konfigurationen.

Zustände	Kodebuchgr.	Überl.	Rechenzeit Authentif.	Rechenzeit Identif.	Rechenzeit Training
5×5	2000	75%	32 ms	38.56 s	1.46 s
5×5	500	75%	21 ms	24.80 s	0.52 s
7×7	500	75%	27 ms	32.17 s	0.80 s

Tabelle 3.9: Rechenzeiten zur Berechnung der Produktionswahrscheinlichkeiten bei Authentifikation (1 : 1 Vergleich) und Identifikation (1 : 1196 Vergleich), außerdem Rechenzeit für das Training eines Modells. Das Modell wurde anhand von 4 Merkmalssequenzen (aus 4 Gesichtsausschnitten) mit 5 Iterationen trainiert.

sind. In der Praxis konnten diese Werte auf ca. ein Zehntel reduziert werden. Da nach aktuellen Stand der Technik nicht unbegrenzt Speicherplatz auf der Bordkarte, die ähnlich einer Chipkarte konzipiert ist, vorhanden ist, kann für die gegebene Anwendung leider nicht mit den Parametereinstellungen, die die beste Erkennungsrate liefern, gearbeitet werden.
Im Vergleich dazu braucht der Eigenface Ansatz übrigens nur die Projektion auf die Eigenvektoren zu speichern, das bedeutet eine Gleitkommazahl pro verwendetem Eigenvektor. Bei den oben ermittelten 100 zu verwendenden Hauptachsen resultiert dies in einem Speicherbedarf von 400 Byte.

Ein weiterer in der Praxis auftretender Aspekt schränkt weiter die in der tatsächlichen Anwendung zu verwendenden Parametereinstellungen ein. Die Rechenzeit darf in einer Online-Anwendung nicht beliebig lang sein. Deshalb wird im folgenden die Rechenzeit für einige Beispielkonfigurationen (siehe Tabelle 3.9) betrachtet. Wie man sieht, sind die Rechenzeiten für die Authentifikation durchaus verträglich mit einer Onlineanwendung, der einzige limitierende Aspekt, der problematisch sein könnte, ist die Dauer des Trainings. Dies ist ein Sonderfall für die hier gegebene Anwendung, da ja in diesem speziellen Fall das Training auch in einer Onlinephase ablaufen soll. Wenn die Flugzeugpassagiere einchecken, sollen sie möglichst ohne längere Verzöge-

rungen ihre Bordkarte erhalten. Für viele andere Anwendungen, läuft das Training, also das Erzeugen der Gesichtsmodelle in einer Offline-Phase ab, so dass dort keine Restriktionen bzgl. der Rechenzeit wären. Für eine Identifikationsanwendung mit einer Datenbankgröße von mehr als 1000 Personen sind die Rechenzeiten des P2D-HMM übrigens weit entfernt davon, für praktische Anwendungen nutzbar zu sein, da diese im Bereich von mehreren Sekunden liegen.
Die zu erzielenden Vorgabewerte von max. 1 Sekunde für Training und Authentifikation sind allerdings in Reichweite, so dass man insgesamt sagen kann, dass der Speicherplatzbedarf auf der Bordkarte für die gegebene Anwendung der deutlich stärker einschränkende Aspekt ist.

Eine technische Lösung für das Speicherplatzproblem wäre es, auf der Bordkarte nur einen Schlüssel, z.B. eine laufende Nummer zu hinterlegen, und das Gesichtstemplate dann über andere Wege, per Netzwerk oder Einlesen eines Datenspeichers, an Bord des Flugzeuges zu bringen. Allerdings ist es nach aktueller rechtlicher Lage nicht ohne weiteres möglich, biometrische Informationen von Personen abzuspeichern, deshalb wurde im Hinblick auf Rechtsverträglichkeit für die Lösung plädiert, dass die Personen selbst im Besitz ihrer biometrischen Daten, also in diesem Fall der Bordkarte sind und die darauf enthaltenen Daten nirgendwo anders abgespeichert werden dürfen. Ein sehr guter Kompromiss zwischen Speicherplatzbedarf, Rechenzeit und Erkennungsleistung wurde für die Parametereinstellung 5×5 Zustände, 500 Quantisierungsstufen im Kodebuch und eine Überlappung von 75% erreicht.

Authentifikation mit HMM

Es wurde in Kapitel 2 bereits erwähnt, dass zur Identifikation das Modell λ_i aus K Modellen $[\lambda_1, .., \lambda_K]$ gesucht wird, welches die höchste Produktionswahrscheinlichkeit $i = \mathbf{arg\ max}_{j=1..K}\, p(O|\lambda_j)$ für eine gegebene Beobachtungssequenz O liefert.

Für die Authentifikation wird auch wieder mit dieser Produktionswahrscheinlichkeit gearbeitet, diesmal wird allerdings überprüft, ob eine Mindestproduktionswahrscheinlichkeit erreicht wird, diese wird durch einen Schwellwert T eingestellt. Dieser Schwellwert wird während des Trainings durch Vergleich von 100 verschiedenen Einstellmöglichkeiten zwischen der maximalen und minimalen Produktionswahrscheinlichkeit, die bei Anwendung der Daten auf die P2D-HMM vorkommt, ermittelt. Dieses wird als relativer Schwellwert $T \in [0, 1]$ dargestellt. In Abbildung 3.15 wird die EER für ein P2D-HMM mit 7×7 Zuständen, 2000 Kodebucheinträgen und einer Überlappung der Merkmalsblöcke von 75% (6 Pixel bei 8×8 Pixel Blockgröße) ermittelt.

Die EER bei der sich die FP- und FN-Rate schneiden liegt bei etwa 0.038 (für 7×7 Zustände, Kodebuchgröße 2000 und 75% Überlappung). Im Abbildung 3.16

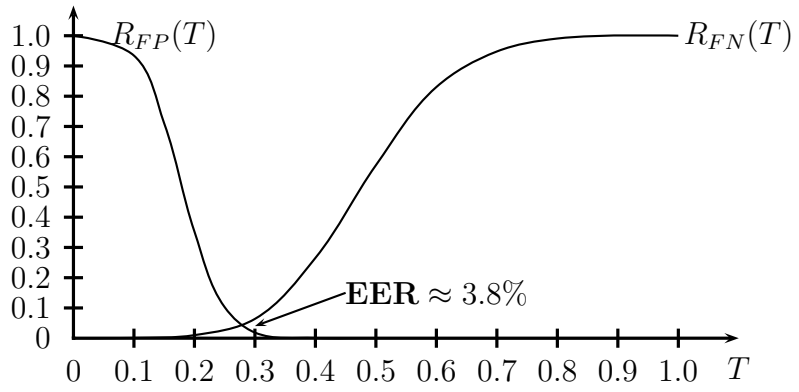


Abbildung 3.15: Bestimmung der EER für die Authentifikation mittels P2D-HMM durch Veränderung des Schwellwertes T. Die sich zueinander gegenläufig verhaltenden FP-Rate R_{FP} und FN-Rate R_{FN} schneiden sich bei der EER. Konfiguration: 7×7 Zustände, Kodebuchgröße 2000 und 75% Überlappung

wird nun noch das Diagramm für die im Folgenden verwendete Konfiguration (5×5 Zustände, Kodebuchgröße 500 und 75% Überlappung) dargestellt, die den besten Kompromiss zwischen Speicherplatzbedarf, Rechenzeit und Erkennungsleistung darstellt. Hier wurde immerhin noch eine EER von 0.045 erreicht.

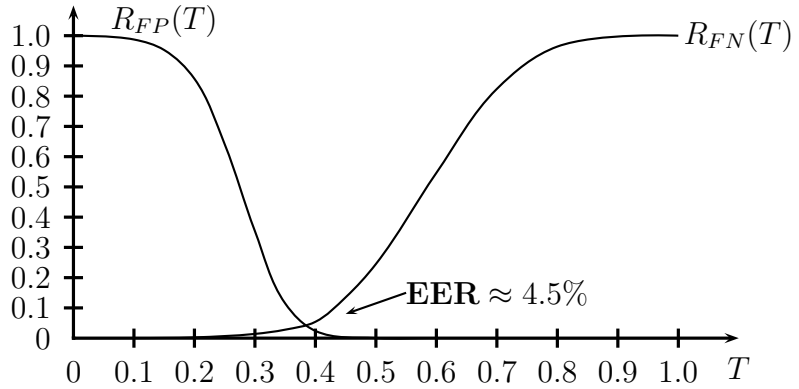


Abbildung 3.16: Bestimmung der EER einer Authentifikation mittels P2D-HMM durch Veränderung des Schwellwertes T. Die sich zueinander gegenläufig verhaltenden FP-Rate R_{FP} und FN-Rate R_{FN} schneiden sich bei der EER. Konfiguration: 5x5 Zustände, Kodebuchgröße 500 und 75% Überlappung

3.7 Zusammenführung von Eigenfaces und HMM zur Authentifikation

Um robustere Ergebnisse zu erzeugen, werden die beiden Ansätze Eigenfaces und P2D-HMM nun miteinander kombiniert. Dabei wird berücksichtigt, das P2D-HMM die deutlich besseren Erkennungszahlen auf der Datenbank vorliegen hat, allerdings können die besten Parametereinstellungen dieser Untersuchungen nicht übernommen werden, da ja wie bereits erwähnt, Einschränkungen bezüglich Speicherplatz und Rechenzeit bestehen. Auch wenn die Eigenface-Methode schlechtere Erkennungsergebnisse liefert, so bietet sie dennoch die Möglichkeit, sinnvoll in die vorliegende Anwendung integriert zu werden, da sie schnell und speicherplatzeffektiv ist.

3.7.1 Auswertung auf der FERET Datenbank

In der vorliegenden Evaluierung wurde ein P2D-HMM mit 5×5 Zuständen und unter der Verwendung von 500 Kodebucheinträgen verwendet, da dieses einen guten Kompromiss zwischen Rechenzeiten und Speicherplatzbedarf darstellt. Nun wird geschaut, ob durch Kombination mit der Eigenfacemethode unter Verwendung von 100 Eigenvektoren, die ja im Vergleich zum P2D-HMM die Speicherplatz- und Rechenzeitressourcen kaum belastet, die Erkennungsleistung gesteigert werden kann, um so ein in der Praxis anwendbares System zu erzeugen.

Um eine Fusion der beiden Ergebnisse zu erzielen, wurden sowohl die Produktionswahrscheinlichkeiten, sowie die Mahalanobis-Distanzen auf einen Wertebereich zwischen 0 und 1 normiert, indem linear zwischen dem maximal auftretenden und minimal auftretenden Wert aller möglichen Paarungen der Trainingsbeispiele skaliert wurde. Im Fall der Distanzmaße wurde der Wertebereich noch invertiert ($d \leftarrow (1-d); d \in [0,1]$), da ja im Gegensatz zur Produktionswahrscheinlichkeit, niedrigere Werte auf eine höhere Ähnlichkeit hindeuten. Anschließend werden diese beiden Werte miteinander verbunden, um eine Entscheidung zu erzielen. Dabei wird folgende Notation verwendet: Die normierte Produktionswahrscheinlichkeit einer aus einem Bild I_k extrahierten Merkmalssequenz für das Modell der Klasse i wird als $p_i(I_k)$ bezeichnet. Die normierte und dann invertierte Mahalanobis-Distanz der Projektion eines Bildes I_k zu der Projektion des Klassenbeispiels der Klasse i wird als $d_i(I_k)$ bezeichnet. Im Folgenden gilt es eine Funktion $f(p_i(I_k), d_i(I_k)) \rightarrow \{0,1\}$ zu finden, die auf die Entscheidung "Person i ist authentifiziert" oder "Person i ist nicht authentifiziert" abbildet.

Es werden verschiedene Fusionsvarianten getestet, beide kombinieren die beiden Werte d_i und p_i zu einem Skalar, anschließend wird anhand einer Schwelle entschie-

	$p_i(I_k) + d_i(I_k)$	$p_i(I_k) \cdot d_i(I_k)$	$\min(p_i(I_k), d_i(I_k))$	$\max(p_i(I_k), d_i(I_k))$
EER	0.047	**0.034**	0.045	0.15

Tabelle 3.10: Die verschiedenen EER in Abhängigkeit der Fusionsstrategie.

den. Für die angegebenen Fusionsstrategien wurde jeweils die EER ermittelt, die Ergebnisse sind in Tabelle 3.10 hinterlegt. Es zeigt sich, dass eine Fusion durch ein Produkt ein verbessertes Ergebnis erzeugt.

Die Fusion der beiden Erkennungsverfahren durch ein einfaches Produkt erreicht eine bessere Equal Error Rate von 3.4% als beide Einzelverfahren für sich genommen (Eigenfaces: 15.1%, 5×5-P2D-HMM mit $C = 500$: 4.5%). Durch diese im Hinblick auf Rechenzeit und Speicherbedarf effiziente Kombination ist das Gesamtergebnis sogar besser als die im Rahmen dieser Arbeit gefundene beste P2D-HMM Parametereinstellung. Diese konnte mit deutlich mehr, fast doppelt sovielen Zuständen (7×7) und einem wesentlich größerem Kodebuch ($C = 2000$ Quantisierungsstufen) eine Equal Error Rate von 3.8% erreichen. Das bedeutet, dass dieses kombinierte System aus Eigenfaces und einem P2D-HMM mit weniger Zuständen (5×5) und einer kleineren Kodebuchgröße ($C = 500$) besser in der Authentifizierungsleistung ist, obwohl es deutlich weniger Ressourcen benötigt. So benötigt es zum Beispiel nur ca. 13% des Speicherplatzes (siehe Tabelle 3.8).

Wenn man die Beschränktheit des Speicherplatzes und der Rechenzeit vernachlässigt, kann das grundsätzliche Prinzip der Fusion zweier Ansätze natürlich auch mit besseren Parametereinstellungen der Einzelverfahren durchgeführt werden, um die Authentifizierungsleistung zu steigern. Bei der Fusion des P2D-HMM mit der Parametereinstellung, die im Rahmen dieser Arbeit die beste Erkennungsleistung hervorbrachte (7×7 Zustände, 2000 Kodebucheinträge, 75% Überlappung der Merkmalssequenz), und dem Eigenfaceansatz über $p_i(I_k) \cdot d_i(I_k)$, also durch das Produkt der normierten Produktionswahrscheinlichkeit und der normierten und invertierten Mahalanobis-Distanz, wurde eine EER von 1.9% erreicht. An dieser Stelle sei noch mal darauf hingewiesen, dass sich diese Leistungsbewertung auf den anspruchsvollen Vergleich "jeder gegen jeden" in der Datenbank bezieht, also deutlich mehr Vergleiche durchgeführt werden, als in der ebenfalls gängigen Evaluationsstrategie, gleich viele positive und negative Beispiele zu verwenden.

3.7.2 Ergebnisse im Anwendungsszenario

Das angestrebte Szenario "Zutrittskontrolle beim Betreten des Flugzeuges" wurde im Rahmen des SAFEE-Projektes in einer Online-Demonstration in einem Simula-

Abbildung 3.17: Der SAFEE-Simulationsaufbau für den Test der Authentifikation unter Realbedingungen.

tionsaufbau eines Flugzeuges (siehe Abbildung 3.17) mit 32 Personen evaluiert. Der Ablauf der durchgeführten Demonstration wird dabei in den Abbildungen 3.18 und 3.19 gezeigt. Die dabei verwendeten Parametereinstellungen, sind diejenigen, die sich bei den Evaluierungen auf der FERET Datenbank als günstig erwiesen haben.

Bei dieser doch recht geringen Anzahl von Vergleichen wurden alle Personen korrekt verifiziert ($EER = 0\%$), allerdings wurden hier nur stichprobenartig Eindringsversuche getestet, also Personen, die sich als jemand anderes ausgeben, also eine falsche Bordkarte besitzen. Da die biometrischen Daten der Personen aus rechtlichen Gründen nicht gespeichert werden konnten, bleibt es bei dieser qualitativen Einschätzung der Ergebnisse, quantitative Auswertungen mit unterschiedlichen Einstellungen sind ohne Speicherung der Daten leider nicht möglich und wurden deshalb, wie bereits beschrieben auf der FERET Datenbank ausgeführt.

Bei der in einem Modellaufbau eines Flugzeugeingangs stattgefundenen Evaluierung wurde großer Wert auf Rechenzeit, und auf den Einfluss auf den klassischen Boardingprozess gelegt. Die objektiven Kenngrößen bezüglich der Erkennungsleistung des Algorithmus waren weniger von Belang, da eine grundsätzliche, aussagekräftige Evaluation des zugrunde liegenden Algorithmus bereits offline auf der FERET Datenbank ausgeführt wurde. Es bleiben die Aussagen übrig, dass das System die geforderten Einschränkungen in Bezug auf Rechenzeiten (jeweils eine Sekunde für Check-In und Boarding) und Speicherplatzbedarf (<12kByte) eingehalten hat und somit tatsächlich in realen Anwendungen einsetzbar ist.

Eine weitere Maßnahme, die das System in der praktischen Anwendung ro-

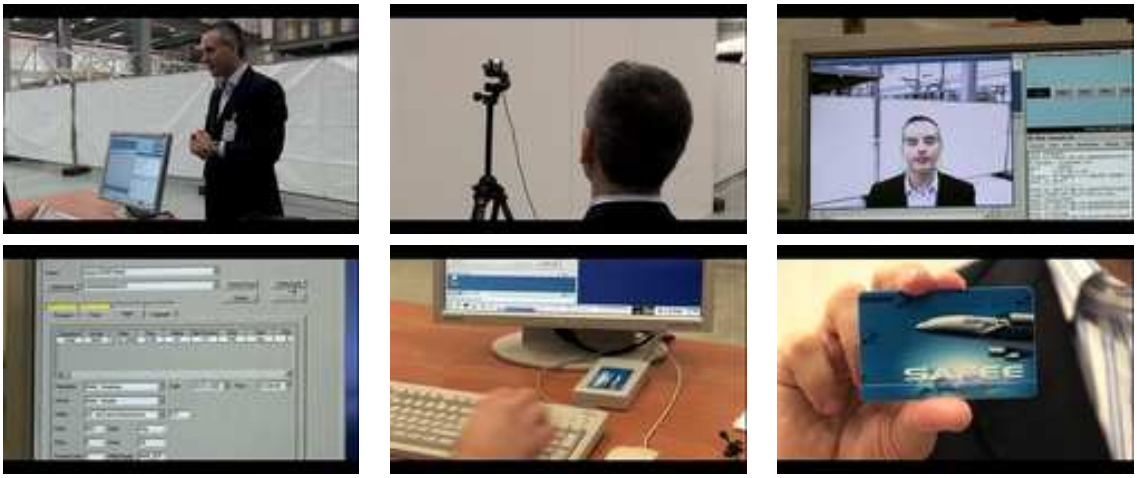


Abbildung 3.18: Beim Check-In wird die Person aufgefordert in die Kamera zu schauen, ein Gesicht wird detektiert, das Gesichtstemplate wird berechnet und auf der Bordkarte gespeichert, die die Person erhält.

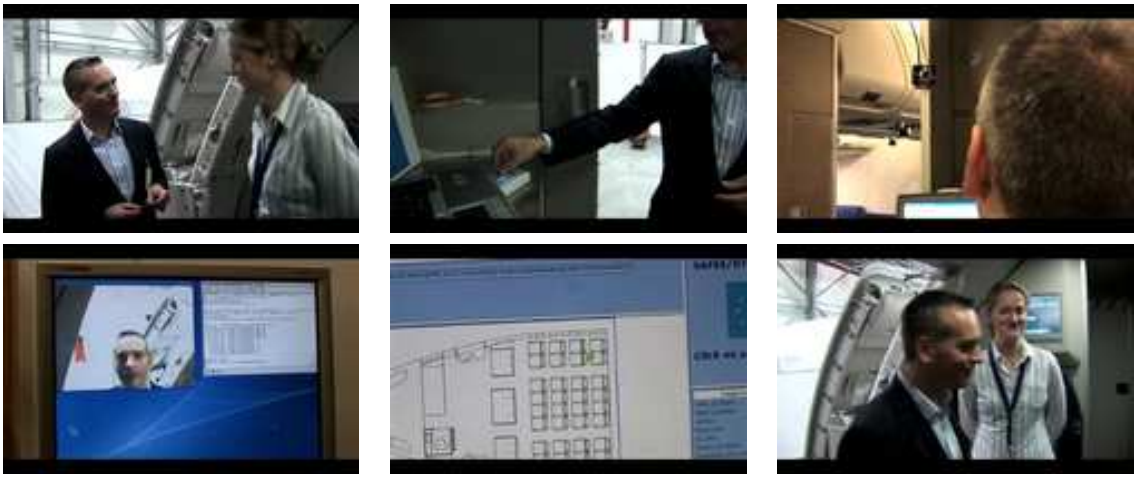


Abbildung 3.19: Beim Boarding wird die Person durch das Bordpersonal aufgefordert, die Bordkarte auf den Leser zu legen und in die Kamera zu schauen. Das Gesicht wird detektiert und mit dem auf der Karte gespeicherten Template verglichen. Eine Entscheidung wird in Form von grünen oder roten Punkten am jeweiligen Sitzplatz im Flugzeug angezeigt, daraufhin wird die Person hereingelassen oder nicht.

bust gegenüber einzelnen Fehldetektionen gemacht hat, ist die Verwendung mehrerer Ausschnitte, die zeitlich unmittelbar hintereinander aufgenommen wurden. Im Erkennungsfall wurde dann einfach der beste der 4 hier zur Verfügung stehenden Ausschnitte gewählt, d.h. der mit dem geringsten Abstand zwischen den Projektionen (Eigenface-Ansatz) und den höchsten Produktionswahrscheinlichkeiten (P2D-HMM). Dadurch wurde eine zusätzliche Robustheit geschaffen.

3.8 Zusammenfassung

In diesem Kapitel wurden zunächst einmal grundsätzliche Untersuchung zum Thema Gesichtserkennung auf Frontalansichten vorgenommen. Dabei wurde ein System umgesetzt, das den aktuellen Stand der Technik in einer praktischen Anwendung einsetzt.
Dabei war das Ziel unter anderem, festzustellen, inwieweit eine gezielte Anpassung des Gesichtsausschnitts an den Erkenner zu einer Verbesserung der Ergebnisse führen kann. Es hat sich gezeigt, dass die Wahl des Ausschnittes, der Vorverarbeitung und der verwendeten Merkmale einen erheblichen Einfluss hat. Daraus begründet sich für weitere Anwendungen eine gezielte Entwicklung eines Ausrichtungsschrittes, der diese recht aufwendige Einstellung von Parametern automatisieren soll und gezielt für jeden Ausschnitt die optimalen Parameter anhand eines formulierbaren Kriteriums bestimmt. Es sei noch darauf hingewiesen, dass in diesem Abschnitt Rotationen in der Bildebene und in der Tiefe nicht behandelt wurden, da sie in dem gegebenen Anwendungsfall nur eine untergeordnete Rolle spielen.
Weiterhin wurde untersucht, ob zwei unterschiedliche Verfahren so kombiniert werden können, dass sie unter gegebenen Einschränkungen bezüglich Rechenzeit und Speicherplatzbedarf Erkennungsraten liefern, die für einen praktischen Einsatz erforderlich sind. Dieses konnte erreicht werden, das Ergebnis des kombinierten Ansatzes war sogar trotz erheblicher Einschränkungen bezüglich Rechenzeit und Speicherplatz besser, als die Erkennungsergebnisse der einzelnen Ansätze ohne Einschränkungen auf ihre Parametereinstellungen.

4

Blickrichtungsunabhängige Gesichtserkennung

In diesem Kapitel werden zwei neuartige Ansätze zur blickrichtungsunabhängigen Personenerkennung vorgestellt. Als Anwendungsfall liegt hier das Erkennen des Nutzers eines Computers vor, der auf einem Drehstuhl an einem Schreibtisch sitzt. Es gilt dieser beobachteten Person eine Identität zuzuweisen. Das Aufnahmesystem ist eine handelsübliche Webcam, die auf einem Computermonitor montiert ist. Die Aufgabe besteht nun darin, die Person, die am Schreibtisch sitzt, zu erkennen, ganz egal in welche Richtung der Kopf orientiert ist, also welche Blickrichtung die Person hat. Diese Aufgabenstellung stellt sowohl hohe Ansprüche an die Detektion als auch an die Erkennungsaufgabe. Für erste Untersuchungen wurde zunächst eine Datenbasis mit möglichst wenig Einflüssen der Umgebung, also unter Laborbedingungen, erstellt. Als Detektionsverfahren diente dort zunächst ein einfaches heuristisches Verfahren. Da dieses allerdings unter komplexeren Umgebungsbedingungen, wie in der Datenbasis, die die Bedingungen des Anwendungsfalls nachbildet, versagte, wurde später eine innovative Kombination aus den Verfahren Active Shape Models und Partikelfilter verwendet, die es auch ermöglichte, Köpfe in komplexen Szenen zu detektieren. Dieses Detektionsverfahren wurde von Schreiber entwickelt [Sch09] und dient hier als Werkzeug, um die auszuwertenden Kopfausschnitte zu finden.

Zwei verschiedene neuartige Erkennungsansätze, wie die mit dem gegebenen Detektionsverfahren gefundenen Ausschnitte hinsichtlich der Identität der darin enthaltenen Person ausgewertet werden können, wurden im Rahmen dieser Arbeit entwickelt und werden im Folgenden vorgestellt. Beim ersten Ansatz handelt es sich um eine hierarchische Kombination von Active Shape Modell und Active Appearance Modell [SSR06], der zweite Ansatz verwendet eine Hidden Markov Modell Struktur, die im Folgenden als zyklisches P2D-HMM bezeichnet wird [SSR08].

Grundsätzliche Lösungsansätze zur Erkennung nicht frontaler Gesichter können auf vielfältige Art und Weise formuliert werden:

- Einzelne Erkenner für verschiedene Posen:
  Hierzu wird mit einer ersten oft personenunabhängigen Erkennungsstufe festgestellt, um welche Pose es sich handelt, in der zweiten Erkennungsstufe ist dann für jede einzelne Pose ein eigener Erkenner für jede zu erkennende Person notwendig.

- Erkenner mit Konfidenzen:
  Bei diesem Ansatz kann der Erkennungsalgorithmus selbständig feststellen, ob es sich um eine für ihn geeignete Pose handelt. Zusätzlich zu dem Erkennungsergebnis liefert er für diesen Aspekt ein Konfidenzmaß. Dieser Ansatz wird gerne für generative Modelle verwendet. Das aus dem Modellwissen erzeugte synthetische Abbild wird dabei mit dem originalen Bildausschnitt verglichen, um dieses Konfidenzmaß zu berechnen. Der Erkenner kann dann zwar nicht zu jeder Kopfpose ein gültiges Erkennungsergebnis liefern, aber selbständig feststellen, ob es sich um eine für ihn geeignete Pose handelt.

- Ein Erkenner für alle Posen:
  Um dieses Ziel zu erreichen, wird ein Rundumsichtsmodell des Kopfes erstellt. Der Erkenner ordnet eine Erkennungsanfrage aus beliebigen Blickwinkel selbständig zu, indem er eigenständig ermittelt, welcher Teil der Rundumsicht zum Vergleich herangezogen werden muss.

Für alle diese Aufgaben gilt, dass der zu untersuchende Bildausschnitt bereits detektiert worden ist. Ein “state of the art” Kopfdetektor, der dieses leisten kann, wird in Sektion 4.2 kurz dargestellt.
Im Folgenden werden zunächst die verwendeten Daten vorgestellt, anschließend wird kurz auf mögliche Ansätze zur Detektion eingegangen. Danach wird der im Rahmen dieser Arbeit entwickelte Ansatz der Auswertung des gegebenen Ausschnittes durch ein Active Appearance Modell vorgestellt und ausgewertet. Weiterhin wird ein zweiter neu entwickelter Ansatz, die zyklischen P2D-HMM, durch Vergleich mit klassischen P2D-HMM vorgestellt und auf beiden zur Verfügung stehenden Datensätzen ausgewertet.

Abbildung 4.1: Beispiele aus der Datenbank, der Kopf wird in verschiedenen Rotationen um die z-Achse erfasst. Diese Bilder beschreiben Aufnahmen aus dem Blickwinkel der ersten Kameraposition unter Laborbedingungen.

Abbildung 4.2: Beispiele für Aufnahmen aus dem Blickwinkel der zweiten Kameraposition. Diese Aufnahmen wurden als Testdaten verwendet.

4.1 Datenbasis und Aufgabenstellung

Um eine genauere Vorstellung der zu lösenden Aufgabe zu bekommen, wird hier die verwendete Datenbasis beschrieben. Die entwickelten Ansätze wurden zuerst auf einer eher einfachen Datenbasis, die unter stabilen Umgebungsbedingungen (gleiche Beleuchtung, homogener Hintergrund) erzeugt wurde, ausgetestet. Nachdem hier erste hoffnungsvolle Ergebnisse erzielt werden konnten, wurde ein anspruchsvoller Anwendungsfall mit Aufnahmen aus realen Büroumgebungen, wechselnden Beleuchtungsbedingungen und bewusst schlechter Aufnahmequalität mit Webcams erzeugt.

4.1.1 Aufnahme unter Laborbedingungen

Eine erste Datenbank wurde in einem Laborraum aufgenommen. Hierfür wurden alle 39 Personen, die in der Datenbank enthalten sind, aufgefordert, auf einem fest positionierten Drehstuhl mindestens eine volle Drehung durchzuführen. Eine zweite Aufnahme wurde unter gleichen Voraussetzungen, allerdings mit veränderter Kameraposition, vorgenommen. Die erste Kameraposition war horizontal zum Gesicht (siehe Abbildung 4.1), die zweite Position ergab einen erhöhten Blickwinkel, so dass der Kopf von schräg oben aufgenommen wurde (siehe Abbildung 4.2).

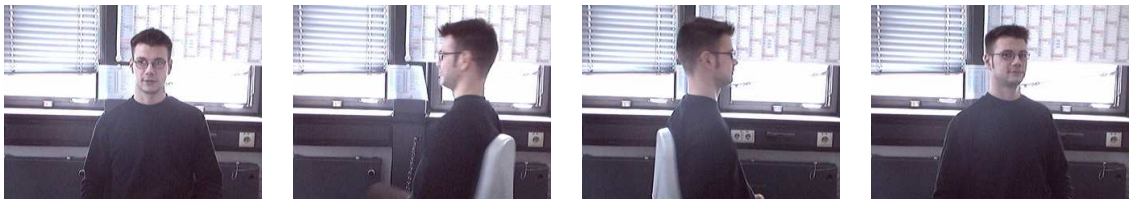


Tabelle 4.1: Beispiele für Aufnahmen aus dem Büroszenario.

Mittels dieser Datenbank wurden erste grundlegende Studien durchgeführt, wie sich Rotation um die z-Achse modellieren lässt, und wie stark der Einfluss einer erhöhten Kameraposition ist. Die hier vorliegenden Aufnahmen wurden mit einer PAL Kamera (interlaced, 768×576 Pixel, 25 fps) unter stabilen Umgebungsbedingungen aufgenommen. Es ist darauf hinzuweisen, dass die Aufnahmen zwischen horizontalem Blickwinkel und dem erhöhten Blickwinkel direkt hintereinander gemacht wurden. Dadurch hat sich ergeben, dass die Personen in beiden Aufnahmen die gleiche Kleidung tragen und die Umgebungsbedingungen identisch sind. Sowohl für die Trainings- als auch für die Testansichten stehen pro Person etwa 200 Bilder in den verschiedenen Blickwinkeln zur Verfügung.

4.1.2 Aufnahme unter Realbedingungen

Eine zweite Datenbank wurde unter anspruchsvolleren Bedingungen aufgenommen. Hier wurde ein normales Büro als Umgebung verwendet, im vorliegenden Fall ein Eckbüro mit zwei Fensterfronten, wodurch erhebliche Einflüsse auf die Beleuchtung erfolgten. Weiterhin waren die Drehstühle, auf denen die Personen Platz nehmen und sich drehen sollten, nicht fest positioniert, sondern wie üblich, frei beweglich, dadurch ergibt sich zusätzlich eine räumliche Translation der Drehachse. Als Aufnahmegerät wurde eine handelsübliche Webcam (VGA, 640×480 Pixel, 15 fps) verwendet, mit den dadurch entstehenden erheblichen Einschränkungen der Bildqualität. Die Kamera war an dem auf dem Schreibtisch positionierten Monitor befestigt, wie es für Büroarbeitsplätze üblich ist. Beispiele für Bilder aus diesem Datenset sind in Abbildung 4.1 abgebildet.

Wenn man die beiden Datensets miteinander vergleicht, sieht man deutlich die erhöhte Schwierigkeit der Erkennungsaufgabe in dem zweiten Datenset. Nicht nur, dass Umgebungseinflüsse deutlich variabler sind, auch die Auflösung der interessierenden Bildregion "Gesicht" ist deutlich niedriger. Beim Erstellen dieses zweiten Datensets wurde gezielt darauf geachtet, dass Trainings- und Testset unabhängig voneinander sind. Dieses wurde dadurch realisiert, dass die verschiedenen Aufnahmen einer Per-

son an unterschiedlichen Tagen zu unterschiedlichen Tageszeiten gemacht wurden. Dadurch ergibt sich automatisch, dass die Personen in jeder Aufnahme unterschiedlich gekleidet sind, und dass verschiedene Beleuchtungsbedingungen, je nach Tageszeit und Wetter, herrschen. Insgesamt wurden in diesem Datenset 25 verschiedene Personen aufgenommen, wobei pro Person etwa 100 Bilder in verschiedenen Blickwinkeln zur Verfügung stehen.

4.2 Detektion und Tracking variabler Kopfposen

Um nun die Erkennungsaufgabe "Personenidentifikation" zu lösen, wird zunächst verlangt, dass der Kopf als Eingabe für eine Erkennungsstufe zur Verfügung steht. Da vermieden werden soll, dass andere personenvariante Merkmale, wie zum Beispiel die Bekleidung zum Zeitpunkt der Datenbankaufnahme, gelernt werden und die Ergebnisse somit verfälschen, werden zunächst Detektionsmöglichkeiten, die den Kopf möglichst passgenau ausschneiden, vorgestellt.

Einfache Detektion durch Hintergrundsubtraktion und a-priori Wissen

Eine einfache und effektive Methode Köpfe in Bildern zu detektieren, ist es, die Annahme zu treffen, dass der Kopf am höchsten Punkt des Vordergrundobjektes beginnt. Unter Annahme eines homogenen Hintergrundes kann die äußere Hülle des Vordergrundobjektes leicht ermittelt werden, aus der äußeren Hülle kann dann leicht auf den Kopf geschlossen werden, siehe Abbildung 4.3.

Im vorliegenden Fall, siehe Abbildung 4.3, wird einfach der Betrag des Gradienten (II.) dazu verwendet, alle Kanten im Bild zu finden, anschließend werden diese Kanteninfomationen über eine Schwellwertbildung ausgedünnt und unterabgetastet (III). Als nächster Schritt wird die äußere Hülle bestimmt, indem in jeder Zeile nur die am weitesten links und rechts liegenden Punkte übrig bleiben (IV.). Danach wird die untere Grenze des Kopfes dadurch bestimmt, dass der Hals als die Region angenommen wird, an der die Breite des Objektes ausgehend vom obersten Punkt zum erstenmal größer wird, nachdem sie kleiner geworden ist, siehe Abbildung 4.4.

Dieser sehr einfache Detektionsansatz ist hinreichend für die in der ersten Datenbank enthaltenen Bilder (in allen Bildern wurde der Kopf detektiert), versagt aber natürlich für komplexere Szenen. Im Folgenden wird kurz ein weiterer Detektionsansatz, der dem aktuellen Stand der Technik entspricht, vorgestellt, der zur Extraktion der Kopfregion auf dem zweiten, anspruchsvolleren Datenset verwendet wurde.

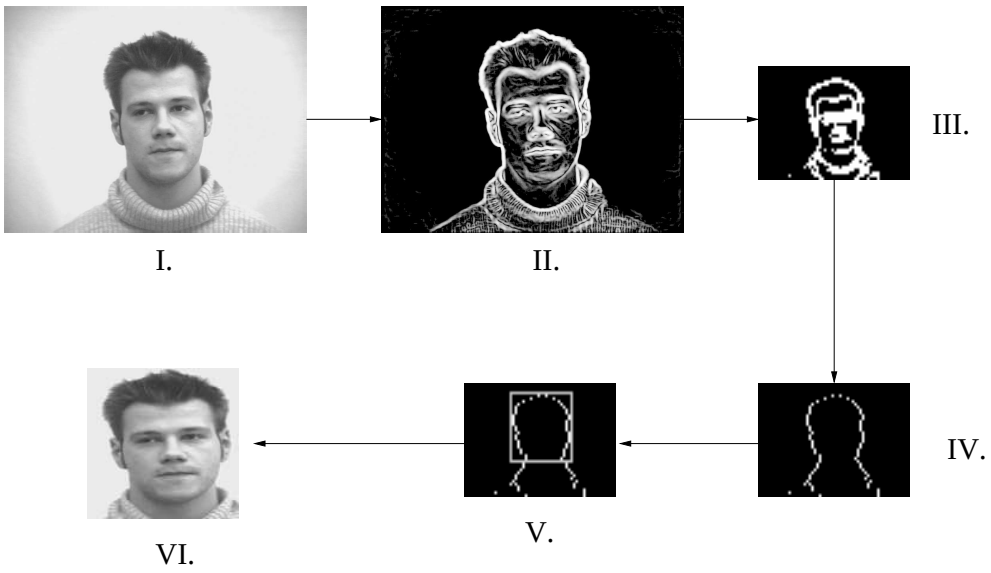


Abbildung 4.3: Eine einfache Methode den Kopfausschnitt zu detektieren basierend auf der Annahme, dass der Kopf am höchsten Punkt der äußeren Hülle des Vordergrundobjektes beginnt und bis zur ersten Verengung am Hals reicht.

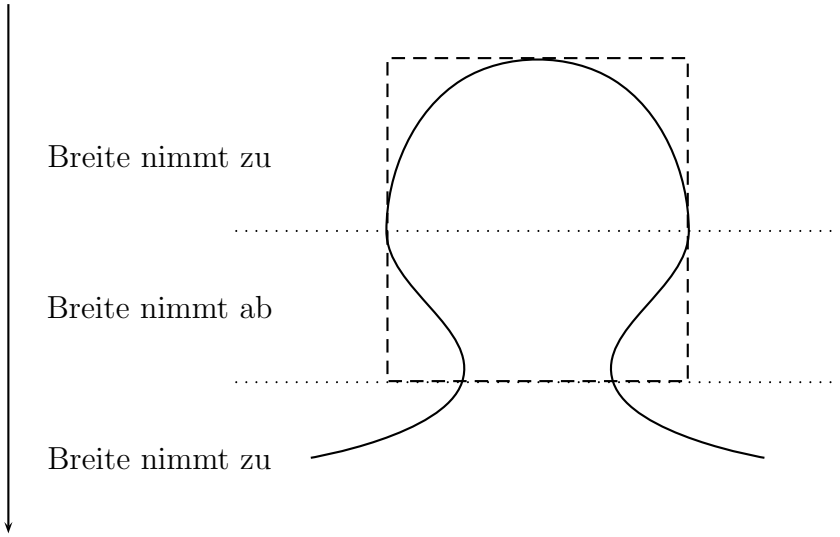


Abbildung 4.4: Einfache Heuristik, um den Kopf auszuschneiden: Der Kopf ist als die Region definiert, die vom obersten Punkt der äußeren Hülle bis zur ersten Verengung geht, die Breite des Ausschnittes ist die bis dahin aufgetretene maximale Breite.

Tracking von Köpfen in komplexen Szenen mittels Active Shape Modellen

Als robuste Methode, die Kopfregion zu detektieren und in einer Bildfolge zu verfolgen wurden Active Shape Modelle, die in ein Partikelfilter eingebettet wurden, verwendet. Im folgenden wird eine kurze Erklärung des hier verwendeten Grundprinzips vorgenommen, für Interessierte sei an dieser Stelle auf die Arbeit von Schreiber [Sch09] verwiesen, die sich ausführlich mit der hier vorgestellten Trackingmethode beschäftigt. Hier wurde ein ASM erzeugt, welches die Form der Kopf-Schulter Region aus allen möglichen Blickrichtungen modellieren sollte. Dieses bietet sich deshalb an, da die grundlegende Form auch für verschiedene Blickrichtungen recht ähnlich ist, somit ist eine rotationsinvariante Detektion möglich.

Als Unterschied zu der Standardvariante der Active Shape Modelle (siehe 2.2.1), in der die Grauwertverläufe der Pixel entlang der Normalen auf der durch das ASM definierten Kontur angepasst werden, werden in der Variante nach Schreiber Gradienteninformationen aus dem Bild extrahiert. Dadurch wird bei der Positionsanpassung des ASM nicht nur die Information, ob eine Kante vorliegt, sondern auch die Richtung der Kante berücksichtigt. Die modifizierte Updating-Methode der aktuellen ASM Schätzung ergibt sich also zu:

- Berechne den Gradienten des gegebenen Bildes durch Anwendung eines Sobel-Operators in x- und y-Richtung.
- Berechne die Normalenvektoren zu jeder Labelpunktposition der aktuellen ASM- Schätzung.
- Berechne den Winkel zwischen Normalenvektor und Gradienten an jeder Pixelposition entlang der Normalen. Wähle den Pixel mit dem kleinsten Winkel als neue Position für jeden Labelpunkt.
- Berechne die Modellparameter für Rotation, Skalierung und Translation durch die Methode der kleinsten quadratischen Abweichung zwischen neuen und aktuellen Labelpositionen
- Berechne die formbeschreibenden Parameter durch Projektion auf die formbeschreibenden Eigenvektoren.
- Wiederhole, bis die Form sich nicht mehr signifikant ändert.

Im Vergleich zum Standard-ASM nach Cootes (siehe [CTCG95]) wird also nicht für jeden Labelpunkt die minimale Distanz zwischen zwei Grauwertverläufen gesucht, sondern das maximale Skalarprodukt zwischen Normalenvektor der Form und Gradient der Grauwerte. Für eine angepasste Form wird ein Qualitätsmaß θ berechnet, indem die Summe aller Skalarprodukte über alle Labelpunkte aufsummiert wird.

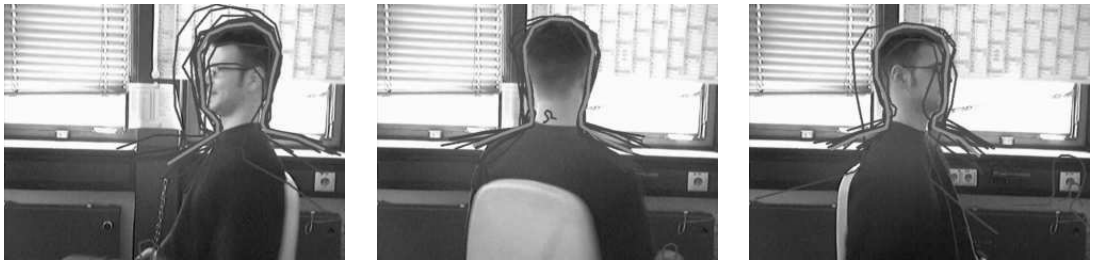


Abbildung 4.5: Beispiele für Kopftracking mittels ASM. Die Verschiedenen Hypothesen sind in dunkelgrau eingezeichnet, der Mittelwert aller Hypothesen in hellgrau (aus [SSR06]).

Um eine robustes und stabiles Tracking zu erzielen, werden diese ASM in ein Partikelfilter (ICondensation, [IB98b]) eingebettet. Dabei ist die Idee, verschiedene Hypothesen von ASM-basierten Formschätzungen zu verfolgen, und die Wahrscheinlichkeitsverteilung w_t für Köpfe zu jedem Zeitpunkt t zu modellieren. Das Ziel ist es, basierend auf aktuellen Beobachtungen o_t, die Position der Köpfe durch die a-posteriori Wahrscheinlichkeit $p(w_t|o_{1..t})$ zu verfolgen. Diese a-posteriori Wahrscheinlichkeit kann iterativ durch

$$p(w_t|o_{1..t}) \leftarrow p(o_t|w_t) \int p(w_t|w_{t-1})p(w_{t-1}|o_{1..t-1})dw_{t-1} \tag{4.1}$$

bestimmt werden. Die a-posteriori Wahrscheinlichkeit $p(w_t|o_{1..t-1})$ zum Zeitschritt $t-1$ führt durch Prädiktion mittels der Dynamik $p(w_t|w_{t-1})$ zu einer a-priori Wahrscheinlichkeit für den aktuellen Zeitschritt t. Diese wird mit der aktuellen Messung $p(o_t|w_t)$, die durch Ermittelung der Qualitätsmaße θ der angepassten ASM erzeugt wird, multipliziert. Daraus ergibt sich dann die aktuelle a-posteriori Wahrscheinlichkeit $p(w_t|o_{1..t})$. Insgesamt kann durch Verwendung eines solchen Partikelfilters zum einen die Genauigkeitsanforderungen an das einzelne ASM gesenkt werden, zum anderen kann auf eine vollständige Suche zu jedem Zeitschritt t zugunsten der Verfolgung der n besten Hypothesen verzichtet werden und dadurch Rechenzeit gespart werden. Gleichzeitig erhöht sich die Robustheit dieses Trackingansatzes, im Vergleich zu einer einzelnen ASM Struktur.

In Abbildung 4.5 und Abbildung 4.6 werden einige Beispiele für detektierte Kopfausschnitte mittels des beschriebenen Ansatzes gezeigt. Die Herausforderung, die sich nun stellt, ist, diese Ausschnitte jeweils nach der Identität der darin enthaltenen Personen zu klassifizieren.

Wie man anhand der Ergebnisse erkennen kann, ist dieses Verfahren trotz schlechter Bildqualität und komplexen Hintergrund in der Lage, die Kopfausschnitte robust

Abbildung 4.6: Beispiele für detektierte Kopfregionen durch den ASM Ansatz, wenn die das Kopf-Schulter Modell umschließende rechteckige Region ausgeschnitten wird. Aufgrund des komplexen Hintergrundes sind nicht alle detektierten Ausschnitte perfekt, aber die meisten sind gut zu verwenden. Diese Ausschnitte sind der Ausgangspunkt für eine Erkennungsaufgabe.

zu detektieren. Diese Detektionsstufe liefert aber nicht nur diese Kopfausschnitte, weiterhin können die detektierten Positionen der Labelpunkte des ASM natürlich auch von weiterverarbeitenden Schritten verwendet werden. Hier bietet sich als erster Ansatz die Verwendung eines Active Appearance Models an, da dessen Formmodell das gleiche Prinzip beinhaltet wie ein ASM und deshalb die Labelpunkte unmittelbar ausgewertet werden können.

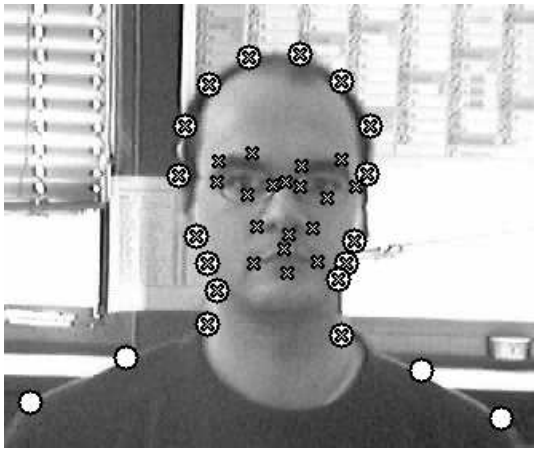


Abbildung 4.7: Labelpunkte: Die Kreise stellen die Labelpunkte des ASM dar, die Kreuze die des AAM. Punkte, die die äußere Kopfform beschreiben, werden von beiden Modellen benutzt (aus [SSR06]).

4.3 Ausrichtung und Erkennung mit Active Appearance Modellen

Active Appearance Modelle (siehe Kapitel 2) haben sich als geeignet erwiesen, formveränderliche komplexe Objekte zu beschreiben. Ein Nachteil der AAM ist allerdings, dass eine recht präzise Initialisierung benötigt wird, damit das Iterative Matchingverfahren gut funktioniert, da dieses immer zum nächsten lokalen Minimum hinkonvergiert. Dieses Problem wird in diesem Ansatz dadurch gelöst, dass das Detektionsergebnis des beschriebenen ASM Ansatzes als Initialisierung verwendet wird. Eine Einführung in die Grundlagen der AAM wurde bereits in 2.3.2 vorgenommen, hier wird nun gezielt geschaut, wie die zur Verfügung stehenden Detektionsergebnisse durch Anwendung eines AAM sinnvoll genutzt werden können.

Um ein Active Appearance Modell geeignet an ein Active Shape Modell anbinden zu können, sollte man dafür sorgen, dass das Ergebnis des ASM möglichst gut durch das AAM genutzt werden kann. Das kann dadurch erreicht werden, dass einige der Labelpunkte, die bei der Erstellung des ASM verwendet wurden, ebenfalls bei der Erstellung des AAM berücksichtigt werden (siehe Abbildung 4.7). Im hier vorliegenden Fall sind das Punkte, die auf der Außenhülle der Kopfform liegen.
Um nun das AAM basierend auf k Trainingsbildern $(I_1, .., I_k)$ zu erzeugen, die mit den formbeschreibenden Vektoren $(\vec{s}_1, .., \vec{s}_k)$ versehen sind, die jeweils der Auflistung der n Labelpunkte entsprechen $\vec{s} = (x_1, y_1, .., x_n, y_n)^T$, werden folgenden Schritte abgearbeitet.

- Berechne die mittlere Form $\overline{\vec{s}}$ und die Hauptachsen der Formveränderungen, die Eigenvektoren $E_s = (\vec{e}_{s,1}, .., \vec{e}_{s,i})$, durch Berechnung einer PCA auf den formbeschreibenden Vektoren $(\vec{s}_1, .., \vec{s}_k)$.
- Warpe die Textur aller Trainingsbeispiele auf die mittlere Form $\overline{\vec{s}}$, dadurch wird eine Formnormierung aller Texturen vorgenommen.
- Berechne die mittlere formnormierte Textur $\overline{\vec{g}}$ und die Hauptachsen der Texturveränderung, die Eigenvektoren $E_g = (\vec{e}_{g,1}, .., \vec{e}_{g,j})$, durch Berechnung der PCA auf den formnormierten Texturvektoren $(\vec{g}_1, .., \vec{g}_k)$.

Ein gegebenes Bild I als Trainingsbeispiel kann nun durch Projektion $\vec{w}_s$ der gegebenen Labelpunkte auf die formbeschreibenden Eigenvektoren, anschließendes Warping auf die mittlere Form und Projektion $\vec{w}_g$ dieser formnormierten Textur auf die texturbeschreibenden Eigenvektoren parametrisch beschrieben werden $\vec{w} = (\vec{w}_s^T, \vec{w}_g^T)^T$. Hier besteht ein kleiner Unterschied zu den in Kapitel 2 eingeführten AAM, da auf eine weitere PCA auf dem kombinierten Form- und Texturvektor verzichtet wurde. Hier wird eine direkte Beschreibung über die Formparameter $\vec{w}_s$ und die Texturparameter $\vec{w}_g$ verwendet. Diese Form des AAM wird als "independent AAM" bezeichnet, da Form und Textur als unkorreliert, also unabhängig behandelt werden.

Ein automatischer Suchalgorithmus, der diese Parameter für ein unbekanntes Gesicht schätzt, wird durch Berechnung einer linearen Prädiktionsmatrix P erzeugt. Diese schätzt die vorzunehmenden Änderungen $\Delta\vec{w}$ am Parametervektor $\vec{w}$ durch Multiplikation mit dem Residuum $\vec{r} = \vec{g}_{orig} - \vec{g}_{synth}$, welches die Differenz der formnormierten Textur $\vec{g}_{orig}$ des Originalbildes und der durch die aktuellen Modellparameter geschätzten formnormierten Textur $\vec{g}_{synth} = E_g\vec{w}_g$ ist. Der Prädiktor soll also bei gegebenem Residuum $\vec{r}$ die vorzunehmende Parameteränderung $\Delta\vec{w}$ schätzen:

$$\Delta\vec{w} = -P\vec{r} \tag{4.2}$$

Durch Erzeugen bekannter $\Delta W = (\Delta\vec{w}_1, \cdots, \Delta\vec{w}_K)$ und Residuen $R = (\vec{r}_1, .., \vec{r}_K)$ durch Synthetisierung mit verschiedenen Modellparametern kann der Prädiktor P mittels multivariater linearer Regression geschätzt werden.

$$P = -\Delta W R^T (RR^T)^{-1} \tag{4.3}$$

Im vorliegenden Fall wurden 34 Labelpunkte verwendet, um das Formmodell des AAM zu erzeugen, davon waren 16 Labelpunkte auch in dem Formmodell des ASM enthalten. Im Vergleich zu anderen AAM Anwendungen, in denen hochaufgelöste Gesichtmodelle erzeugt werden, ist diese Anzahl vergleichsweise niedrig. Da allerdings in der vorliegenden Anwendung niedrig aufgelöste Bilder in schlechter Qualität vorliegen, ist diese Anzahl ausreichend.

Um nun ausgehend von einer ASM Schätzung eine geeignete Initialisierung des AAM Matching Algorithmus zu erzielen, wird der Formvektor $\vec{s}^*$, der von der ASM Schätzung erzeugt wird auf die Eigenvektoren des AAM Formmodells projiziert. Dabei werden die Elemente, die sich auf die Koordinaten der fehlenden, nur im AAM, nicht aber im ASM befindlichen Labelpunkte beziehen, weggelassen, dieses wird durch die modifizierte Eigenvektormatrix E_* beschrieben.

$$\vec{w}_s \approx E_*^T \vec{s}_* - \overline{\vec{s}}^* \tag{4.4}$$

Bei genauerer Betrachtung bedeutet das Weglassen der im ASM nicht vorhandenen Koordinaten eine Ersetzung dieser Labelpunkte durch die im mittleren Formvektor enthaltenen Positionen. Nachdem diese Formschätzung vorgenommen wurde, werden die Texturparameter $\vec{w}_g$ durch Warpen auf die mittlere Form und Projektion der Textur bestimmt. Die dadurch resultierende Parameterschätzung $\vec{w} = (\vec{w}_s^T, \vec{w}_g^T)^T$ wird anschließend in der Matchingprozedur für das AAM als Initialschätzung verwendet:

- I. Berechne das Residuum $\vec{r} = \vec{g}_{synth}(\vec{w}_g) - \vec{g}_{orig}(I, \vec{w}_s)$
- II. Schätze die vorzunehmende Parameteränderung $\Delta\vec{w} = -P\vec{r}$
- III. Wende Parameteränderung an $\vec{w} \leftarrow \vec{w} + \alpha\Delta\vec{w}$. Überprüfe ob die Energie $\vec{r}^2$ des Residuums kleiner geworden ist, falls ja behalte die Änderung bei, falls nicht, ändere α in definierten Schrittweiten (z.B $\alpha \in \{1, 2, 0.5\}$).
- Wiederhole I.-III. solange, bis für kein α eine Verbesserung erzeugt werden konnte.

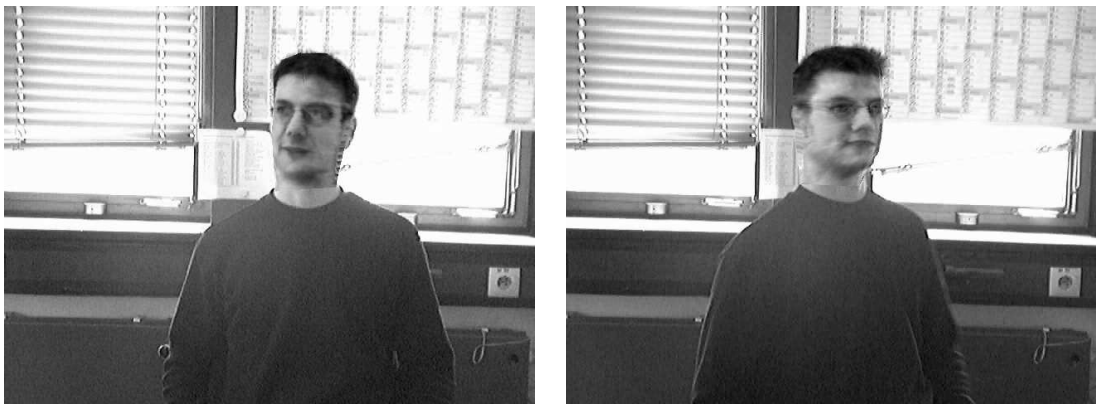


Abbildung 4.8: Im Bild resynthetisierte Suchergebnisse, die durch die Kombination von ASM und AAM erreicht wurden.

Abbildung 4.9: Weitere Beispiele für resynthetisierte Gesichter. Auch hier wurden die dazu notwendigen Parameter durch die vollautomatische Suche ermittelt.

In Abbildung 4.8 werden einige Beispiele für im Bild an Originalposition resynthetisierte Gesichter dargestellt, die unter Verwendung der durch das beschriebene Suchverfahren ermittelten Parameter erzeugt wurden. In Abbildung 4.9 werden weitere resynthetisierte Suchergebnisse angezeigt, diesmal nur die Kopfausschnitte.

Wie man an den resynthetisierten Beispielen sehen kann, können AAM für einen sehr weiten Blickrichtungsbereich eine sehr gute, komprimierte Beschreibung aller wichtigen Informationen in Form eines Parametervektors $\vec{w}$ liefern. Allerdings hat diese Beschreibungsform auch ihre Grenzen. Wenn man sich das Grundprinzip des AAM anschaut, nämlich die Beschreibung durch Punkte als Formverteilung und die dazwischenliegende Textur, so kann dieses Prinzip nur solange funktionieren, solange alle Punkte im Bild zu sehen sind. Sobald durch die gegebene Perspektive Punkte verdeckt werden, wie beispielsweise in einer Profilansicht, in der die Hälfte aller Punkte auf der Kamera abgewandten Seite liegen würden, muss das Prinzip zwangsläufig versagen, siehe Abbildung 4.10.

Ob eine Blickrichtung allerdings für eine Beschreibung mittels AAM geeignet ist, kann durch das Ergebnis der Suche selbst bestimmt werden. Wenn das der Fall ist, hat die Textur des gefundenen Ausschnittes $\vec{g}_{orig}$ und die durch die ermittelten Parameter resynthetisierte Textur $\vec{g}_{synth}$ eine hohe Ähnlichkeit. Diese Ähnlichkeit kann durch ein geeignetes Maß, zum Beispiel durch den normierten Kreuzkorrelationsko-

Abbildung 4.10: Eine Beschreibung durch AAM ist nur bis zu einem bestimmten Blickwinkel sinnvoll, da diese Methode versucht, alle in der Modellerstellung trainierten Punkte zu lokalisieren. Eindeutig erkennbar ist, dass der resynthetisierte Ausschnitt in ungeeigneten Blickwinkeln zunehmend unähnlicher zu dem Originalausschnitt wird.

effizienten *cc*, beschrieben werden:

$$cc(\vec{g}_{orig}, \vec{g}_{synth}) = \frac{\vec{g}_{orig} \cdot \vec{g}_{synth}}{|\vec{g}_{orig}| \cdot |\vec{g}_{synth}|} \tag{4.5}$$

Der Wert dieses normierten Kreuzkorrelationskoeffizienten *cc* liegt zwischen 0 und 1, nur für sehr ähnliche Texturvektoren $\vec{g}_{orig}$ und $\vec{g}_{synth}$ liegt dieser nahe 1. Ein einfacher Schwellwertklassifikator auf diesem Koeffizienten *cc* kann nun bestimmen, ob es sich um eine "gute" Beschreibung des Ausschnittes handelt, somit um eine geeignete Blickrichtung, oder nicht.

4.3.1 Identifikation mit AAM Parametern

Für alle Blickrichtungen, die nach Anwendung des Schwellwertklassifikators als gut gewertet wurden, soll nun die Identität der beobachteten Person erkannt werden. Dafür wird der Vektor der Form- und Texturparameter $\vec{w}$ klassifiziert. Da es sich um einen Vektor mit fest definierter Länge handelt, kann ein statischer Klassifikator verwendet werden. In dem Parametervektor sind aber nun sowohl Parameter enthalten, die die Form, damit auch implizit die Blickrichtung beschreiben, wie auch solche, die die Textur beschreiben. Man kann davon ausgehen, dass sowohl Formaspekte als auch die Textur zur Bestimmung der Indentität eine Rolle spielen, aufgrund der

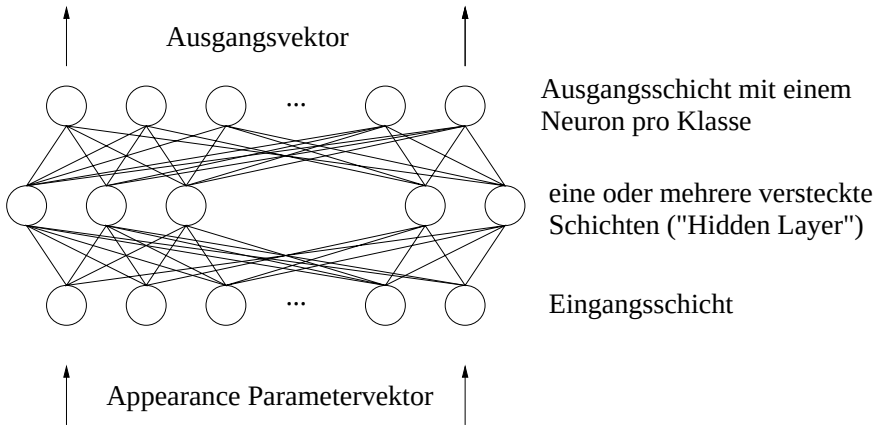


Abbildung 4.11: Ein mehrschichtiges neuronales Netz zur Klassifikation der Identität basierend auf dem Appearance-Parametervektor $\vec{w}$.

statistischen Natur des Modells kann aber nicht klar interpretiert werden, welcher Parameter wie stark für eine Identifikationsaufgabe geeignet ist. Deshalb wurde hier ein Lernverfahren angewandt, welches die Gewichtung der der einzelnen Parameter automatisch lernt. Der dazu verwendete Ansatz ist ein mehrschichtiges neuronales Netz, welches den Parametervektor $\vec{w}$ als Eingangsgröße hat, sowie für jede gelernte Klasse ein Ausgangsneuron (siehe Abbildung 4.11).
Das neuronale Netz wird durch die jeweiligen berechneten Parameter $\vec{w}$ der Trainingsdaten mittels Backpropagation (siehe Anhang C) trainiert. Der zu erreichende Wert am Ausgang ist dann "1" für das Output-Neuron, das die jeweilige Klasse beschreibt und "-1" für jedes andere Output-Neuron. Im vorliegenden Fall wurde ein Netz mit 168 Input-Neuronen ($\vec{w}$ mit 68 Form- und 100 Texturparametern), 85 Neuronen in der Hidden-Schicht, und 25 Output-Neuronen (eins für jede Identitätsklasse) verwendet.
Ergebnisse werden in Abbildung 4.12 im Hinblick auf den Schwellwert $T \in [0, 1]$ aufgeplottet, der bestimmt, wie groß der normierte Kreuzkorrelationskoeffizient cc mindestens sein muss, damit der Identifikationsprozess durchgeführt wird. Nur für höhere T werden tatsächlich sinnvolle Erkennungsraten erzeugt, da ja nur dann eine parametrische Beschreibung vorliegt, die den zu beurteilenden Ausschnitt tatsächlich sinnvoll repräsentiert.
Wie man an der Abbildung 4.12 erkennen kann, wurde eine Erkennungsrate von 100% erreicht, wenn ein sehr hoher Qualitätsschwellwert T eingestellt wurde. Dies bedeutet aber gleichzeitig, dass nur eine geringe Anzahl von Eingangsbildern überhaupt klassifiziert wird. Ein guter Kompromiss ist ein Schwellwert von 0.92, in diesem Fall wurden immer noch 15% der Frames klassifiziert, mit einer Erkennungsrate von 92.5%. Wenn man bedenkt, dass bei den hier genutzten Bildsequenzen eine große Anzahl von Profil- oder Hinterkopfansichten enthalten ist, ist dies ein gutes

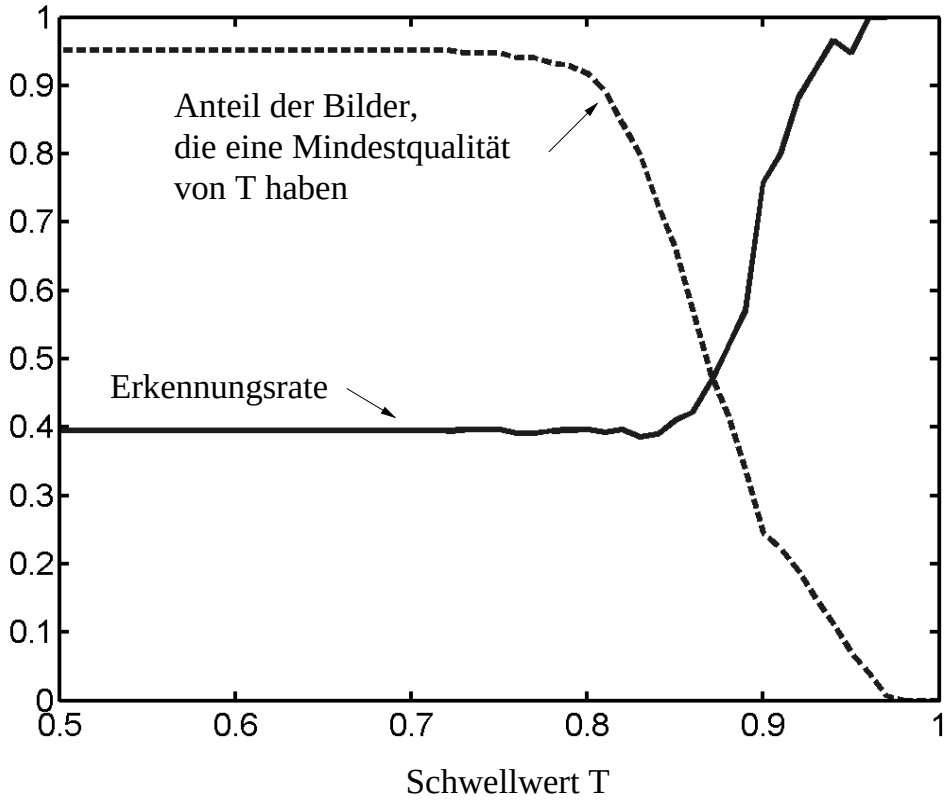


Abbildung 4.12: Erkennungsrate durch Anwendung eines MLP zur Klassifikation auf dem Parametervektor $\vec{w}$ (durchgezogene Linie). Der Anteil der Bilder, in denen eine parametrische Beschreibung des Gesichtsausschnitts mit der Mindestqualität T erreicht wurde, ist durch die gestrichelte Linie dargestellt.

und in Anwendungen einsetzbares Ergebnis. Bei einer Kombination von Identifikation und Tracking, wie es hier der Fall ist, ist im Grunde genommen ein einziger guter Frame ausreichend, um die beobachtete Person mit einer Identitätsinformation zu versehen. Nur bei Neuinitialisierung des Trackingalgorithmus wäre eine wiederholte Erkennung notwendig. Das Interessante an dem hier vorgestellten Ansatz ist, dass er durch Anwendung eines Konfidenzmaßes selbst in der Lage ist, zu entscheiden, wie gut ein Ausschnitt für die Erkennungsaufgabe geeignet ist. Wenn man aufgrund der Erstellung der Daten, nämlich die Aufforderung an Personen eine Rotation auf einem Drehstuhl vorzunehmen, davon ausgeht, dass die vorkommenden Blickwinkel in etwa gleichverteilt sind, bedeuten diese 15% näherungsweise, dass von den 360° einer vollen Rotation etwa ein Bereich von 54° oder, ausgehend dass das der frontale Sektor ist, ein Blickwinkelbereich von ±27° gut klassifiziert werden konnte.

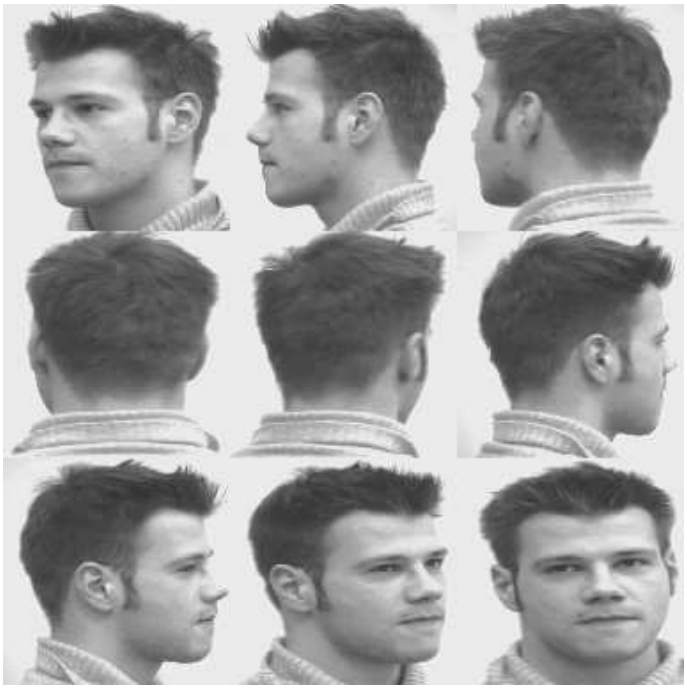


Abbildung 4.13: Detektionsergebnisse des heuristischen Detektionsansatzes. Diese Bilder sollen im folgenden durch P2D-HMM klassifiziert werden.

4.4 HMM-basierte Erkennung

Im Folgenden werden nun verschiedene Varianten von P2D-HMM-basierten Erkennungssystemen für die Aufgabe, einem gegebenen Bildausschnitt eine Identität zuzuweisen, getestet. Die dabei untersuchten Bildausschnitte wurden dabei mit den beiden bereits vorgestellten Detektionsansätzen ermittelt.
Erste Untersuchungen finden dabei auf der unter Laborbedingungen aufgenommenen Datenbank statt. Der bereits vorgestellte, auf Heuristik basierende Detektor liefert dabei die auszuwertenden Kopfausschnitte, einige Beispiele sind in Abbildung 4.13 dargestellt. Später, nachdem das hier entwickelte Konzept vorgestellt wurde, werden dann vergleichende Untersuchungen auf dem zweiten Datenset, das unter realen Bedingungen in einer Büroumgebung aufgenommen wurde, durchgeführt.

4.4.1 Grundsätzliche Überlegungen

Als erstes werden einige grundlegende Überlegungen zu der Problemstellung gemacht. Dadurch, dass nur zweidimensionale Projektionen der in Wirklichkeit dreidimensionalen Gesichter vorliegen, sind je nach Ansicht unterschiedliche Teilinfomationen des Gesichtes vorhanden. Wenn man sich vereinfacht vorstellt, dass Augen, Nase, Mund und Ohren charakteristische Informationen für einen Kopf darstellen,

so ist die Aufgabenstellung hier, eine Identität zuzuordnen, egal ob alle diese Merkmale (z.B Frontalansicht), oder nur eine Teilmenge (z.B Profilansicht) vorhanden ist. Hier ist übrigens der Hauptunterschied zu dem vorherigen Ansatz mittels AAM, in dem durch ein Konfidenzmaß überprüft wurde, ob alle Merkmale vorhanden sind, und nur in diesen Fällen klassifiziert wurde.
Ein weiterer interessanter Aspekt der Problematik ist, dass die relative Positionierung der Merkmale zueinander ungefähr gleich bleibt, auch wenn aus extremen Perspektiven auf die Bildebene projiziert wird. Das bedeutet, die Augen sind immer über dem Mund und zwischen den Ohren, die Nase zwischen den Augen usw., anders ausgedrückt, die relative Reihenfolge der Merkmale ist gleich bleibend. Dies ist interessant und wichtig für die Modellierung mittels HMM, da aufgrund dieser Charakteristik ein (Pseudo-2 Dimensionales) links-rechts Modell verwendet werden kann.
Als letzte offene Fragestellung gilt es zu klären, ob es eine geeignete rotationsinvariante Repräsentation der Merkmale gibt, die ähnliche Merkmalswerte für unterschiedliche Ansichten liefert, also ob z.B. die Nase in frontaler Ansicht eine ähnliche Merkmalswertverteilung wie die gleiche Nase in einer Profilansicht hat.
Ausgehend von einer Modellierung mit DCT-Koeffizienten kann man die Beobachtung machen, dass vertikale Ortsfrequenzen von einer Rotation um die z-Achse (Rotation in der Tiefe) nur wenig betroffen sind, allerdings hat eine solche Rotation Auswirkung auf horizontale Ortsfrequenzen. Wenn man sich ein Muster mit gleichabständigen horizontalen und vertikalen Streifen als Textur eines Rotationskörpers um die z-Achse, z.B. eines Zylinders vorstellt, dann werden in der Projektion auf die Bildebene die Abstände der vertikalen Streifen zum Rand des Zylinders hin geringer, während die Abstände der horizontalen Streifen gleich bleiben. Dieses bedeutet also, dass vertikale Frequenzen gleich bleiben, während horizontale Frequenzen zum Rand des Zylinders zunehmen (siehe Abbildung 4.14).

Abbildung 4.14: Bei Projektion auf die Bildebene verändern sich horizontale Frequenzen einer Textur je nach Winkel, vertikale Ortsfrequenzen sind jedoch invariant.

Diese Betrachtungen sind für die tatsächliche Anwendung natürlich nur näherungsweise gültig, da davon ausgegangen wird, dass die Oberflächennormalen der Texturflächen des beobachteten Gesichtes genau den Normalen des Zylinders an diesen Stellen entspricht. Bei komplexen geometrischen Körpern wie Gesichtern, sind allerdings die Oberflächennormalen durchaus abweichend. In der Nasenregion zum Beispiel weichen die Oberflächennormalen deutlich von denen eines Zylinders ab, für die eher glatten Regionen des Gesichtes, wie Stirn, Wangen, oder auch Mundregion passt die grobe Näherung durch einen Zylinder allerdings recht gut.
Im vorliegenden Fall wird davon ausgegangen, dass eine Modellierung mit DCTmod2 Koeffizienten verwendet werden kann, da Koeffizienten, die sich auf Vertikalfrequenzen beziehen, unabhängig von der Rotation stabil sein sollten, und Horizontalfrequenzen sich nicht sprunghaft, sondern kontinuierlich in Abhängigkeit der Rotation verändern. Da die DCTmod2 Koeffizienten die Änderung der DCT-Koeffizienten zwischen benachbarten Blöcken enthalten, also eine relative Beziehung der Ortsfrequenzen benachbarter Blöcke beschreiben, wird eine näherungsweise Rotationsinvarianz bezüglich Rotationen um die z-Achse angenommen. Deshalb wird für die blickrichtungsunabhängige Erkennung die gleiche Merkmalsextraktion verwendet, die bereits für frontale Ansichten erfolgreich eingesetzt werden konnte.

4.4.2 Modellierung mit klassischen P2D-HMM

Als Erstes wird versucht die beobachteten Ausschnitte mit klassischen P2D-HMM zu modellieren. Dabei wird ein Modell pro Klasse trainiert, darin sind dann allerdings alle verschiedenen Blickrichtungen enthalten. Bei grundsätzlicher Betrachtung bedeutet dies, dass mehrere Blickrichtungen parallel in das Modell trainiert werden. Aufgeschlüsselt auf die links-rechts Modellierung von Spaltenmodellen, heißt das, dass verschiedene Spaltenansichten in ein und dasselbe Spaltenmodell trainiert werden. Bei einen Links-Rechtsmodell bedeutet dies zum Beispiel, dass in dem ersten Spaltenmodell sowohl der Spaltenausschnitt, der das linke Ohr beschreibt (in einer Frontalansicht), als auch der Ausschnitt, der Nase und Mund enthält (in einer Profilansicht), enthalten ist.
Erste Versuche werden nun auf der Datenbasis durchgeführt, die unter Laborbedingungen erstellt wurde. In Abbildung 4.15 sind Beispiele für ausgeschnittene Trainings und Testdaten dargestellt. Diese Ausschnitte wurden alle durch den heuristischen Detektionsansatz ermittelt, der unter Laborbedingungen durchgehend gute Ausschnitte ermitteln konnte.

Zunächst werden in den Tabellen 4.2 und 4.3 die Erkennungsergebnisse unter Verwendung verschiedener Konfigurationen aufgelistet. Es zeigt sich, dass die P2D-HMM für diese blickrichtungsunabhängige Erkennung bei Verwendung von allen Blickrichtungen als Trainingsmaterial überraschend gut abschneiden.

Abbildung 4.15: Trainingsbeispiele (horizontale Ansicht, obere Reihe) und Testbeispiele (Aufnahme von schräg oben, untere Reihe) für die gegebene Erkennungsaufgabe. Die Unterschiede zwischen Trainings- und Testdaten sind je nach Person, Kopfhaltung und Körpergröße unterschiedlich stark ausgeprägt.

Anzahl der Spaltenmodelle	Anzahl der Zustände pro Spaltenmodell	Erkennungsrate
11	11	0.949
9	9	0.958
7	**7**	**0.960**
5	5	0.944
3	3	0.889
1	1	0.536

Tabelle 4.2: Klassische P2D-HMM zur Modellierung verschiedener Blickrichtungen. Hier wurde mit allen Blickrichtungen, die vorkommen können, trainiert (50% Überlappung, 100 Quantisierungsstufen).

Interessant ist die Beobachtung, dass selbst mit nur einem Zustand eine Erkennungsrate von ca. 53% erreicht werden konnte und das, obwohl die Merkmale mit nur 100 Kodebucheinträgen relativ grob quantisiert wurden. Dies bedeutet nichts anderes, als dass die relativen Häufigkeiten der verschiedenen Quantisierungsstufen der vektorquantisierten DCTmod2-Koeffizienten-Vektoren schon eine aussagekräftige Beschreibung in Hinblick auf die Identität der im jeweiligen Ausschnitt zu sehenden Person sind, ohne dass eine örtliche Modellierung ihres Auftretens vorgenommen wird. Mit weiterer Hinzunahme von Zuständen ist das Modell zunehmend in der Lage nicht nur die Häufigkeiten, sondern auch die relative örtliche Position der Merkmale zu modellieren. Dies führt zu einer Verbesserung der Erkennungsleistung bis zu einer bestimmten Modellgröße, ab der die Erkennungsraten wieder sinken. Das lässt sich

Kodebuchgr.	Erkennungsrate
10	0.499
50	0.847
100	0.960
500	**0.989**

Tabelle 4.3: Erkennungsraten eines 7×7 P2D-HMM unter Verwendung unterschiedlicher Kodebuchgrößen.

dadurch erklären, dass bei einer konstanten Länge der Beobachtungssequenz, hier bei Ausschnitten der Größe 64×64 Pixel und bei 50% Überlappung von 8×8 Pixel großen Blöcken werden 15×15 Merkmalsvektoren extrahiert, mit zunehmender Komplexität des Modells die Warpingmöglichkeiten innerhalb der Merkmalssequenz abnehmen. Bei 15×15 Zuständen würde genau ein Merkmalsvektor auf einen Zustand abgebildet, es würde gar keine Warpingmöglichkeit mehr bestehen und man könnte genauso gut den statischen Abstand zwischen beobachteter Merkmalsequenz und der Merkmalssequenzen aller Trainingsbeispiele berechnen.

Es ist bemerkenswert und verwunderlich, dass klassische P2D-HMM so gute Erkennungsergebnisse (bis zu 98.9%) auf den verschiedenen Blickrichtungen dieses Datensets erzielen können. Dieses lässt sich vermutlich dadurch erklären, dass die Unterschiede zwischen Trainings und Testdaten doch insgesamt recht gering sind und dass eine riesige Anzahl an Trainingsdaten pro Person zur Verfügung stand, die alle vorkommenden Blickrichtungen enthielt. Die nur leicht unterschiedlichen Elevationen in der Blickrichtung der Testdaten scheinen keinen größeren Einfluss auf die Erkennungsleistung zu haben. Insgesamt ist die Ähnlichkeit der Personen zwischen Trainings- und Testset sehr groß und da für jede Person alle verschiedenen zur Verfügung stehenden Blickrichtungen des Trainingssets verwendet wurden, muss man davon ausgehen, dass die Modelle die Merkmalssequenzen der verschiedenen Blickrichtungen mehr oder weniger auswendig gelernt haben. Ob die Modelle in der Lage sind zu generalisieren, wird nun getestet, indem die Modelle mit deutlich weniger Ansichten trainiert werden. Die folgenden Versuche finden deshalb mit nur 8 Trainingsbeispielen pro Klasse statt, eine frontale Ansicht, die beiden Profilansichten, die Hinterkopfansicht und die vier dazwischen liegenden Ansichten. Ziel ist es zu überprüfen, ob die Modelle hinreichend gut generalisieren können. Ergebnisse sind in Tabelle 4.4 dargestellt.

Wie man in Tabelle 4.4 sieht, sinkt die Erkennungsleistung auf ein niedrigeres Niveau. Da in allen 8 Trainingsbildern die charakteristischen Gesichtsmerkmale in unterschiedlicher horizontaler Reihenfolge auftreten, können beim Training des P2D-

Zustände	Kodebucheintr.	Erkennungsrate
5×5	500	0.837
3×3	500	0.803
1×1	500	0.695
1×3	500	0.821

Tabelle 4.4: Erkennungsrate der P2D-HMM unter Verwendung von nur 8 Blickrichtungen zum Training.

HMM nicht sinnvoll Beobachtungen bestimmten Zuständen zugeordnet werden, weil ja der Merkmalsstrom nur entsprechend der im links-rechts P2D-HMM erlaubten Zustandsübergänge durchlaufen werden kann. Ein paralleles Auswendiglernen der Trainingsbeispiele (die Reklassifikationsrate beträgt 100%) reicht nur noch bedingt aus, um Blickrichtungen, die nicht in den Trainingsdaten auftreten, sinnvoll zuzuordnen. Diese Erkenntnis lässt sich anhand der Ergebnisse in Tabelle 4.4 bestätigen. Die Erkennungsleistung eines einzelnen Spaltenmodells mit 1×3 Zuständen ist besser, als die von mehreren Spaltenmodellen (3×3 Zustände). Daran lässt sich erkennen, dass nur die vertikale Reihenfolge von Merkmalen modelliert werden kann, eine horizontale Abfolge, die ja direkt abhängig von Blickwinkel ist, kann mit dieser Struktur nur sehr eingeschränkt modelliert werden.

4.4.3 Modellierung durch zyklische P2D-HMM

Nachdem festgestellt wurde, dass bei der Modellierung mit klassischen P2D-HMM die Bildspalten, die unterschiedliche Merkmale enthalten, je nach Reihenfolge ihres Auftretens und nicht je nach darin enthaltenem Merkmal, in Spaltenmodelle trainiert werden, wird nun nach einer Möglichkeit gesucht, eine Modellierung vorzunehmen, die egal welche Rotation vorliegt, Bildspalten mit gleichartigen Merkmalen immer auf die gleichen Spaltenmodelle abbildet. Dafür müssen mehrere Kriterien erfüllt werden. Zum einen kann nicht mehr mit einem klassischen Links-Rechts-Modell gearbeitet werden, das nur eine Einsprungsmöglichkeit hat, und somit immer im ersten Spaltenmodell beginnt, sondern es muss ermöglicht werden, in jedes Spaltenmodell einzuspringen. Dieses wird dadurch gewährleistet, dass die Einsprungswahrscheinlichkeiten $\pi = \{\pi_j\}$ über alle Spaltenmodelle gleichverteilt werden. Weiterhin muss es für das Modell möglich sein, jede beliebige Rotation abzubilden, also müssen die Vorwärtstransitionen einen geschlossenen Ring bilden. Dafür muss eine Transition vom letzten Spaltenmodell zum ersten Spaltenmodell eingefügt werden. Nun muss es noch möglich sein, dass das Modell in jedem beliebigen Spaltenmodell enden kann. Insgesamt ergibt sich damit eine Struktur, die einen geschlossenen Ring von Spaltenmodellen bildet, die nur in einer Richtung durchlaufen werden kann, aber in jedem

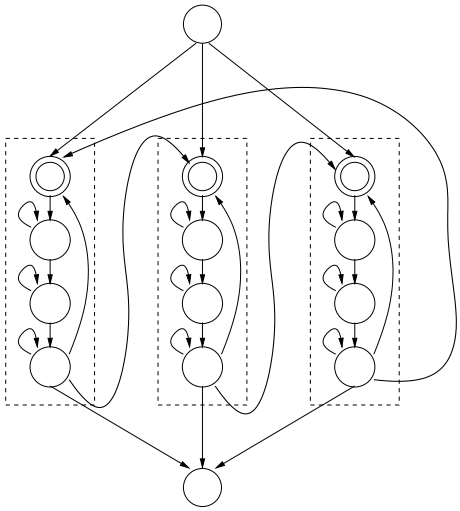


Abbildung 4.16: Ein 3×3 zyklisches P2D-HMM.

einzelnen Spaltenmodell beginnen und enden kann (siehe Abbildung 4.16). Diese Struktur wird als zyklisches P2D-HMM bezeichnet.

Wenn man sich dieses Modell räumlich vorstellt, so entspricht es einem geschlossen Ring von Spaltenmodellen, welches sich zu einer zylinderartigen Struktur ergibt. Im Grunde genommen wird also der Kopf unter der Vorstellung modelliert, das sich seine Textur auf einen umschließenden Zylinder projiziert (siehe Abbildung 4.17).

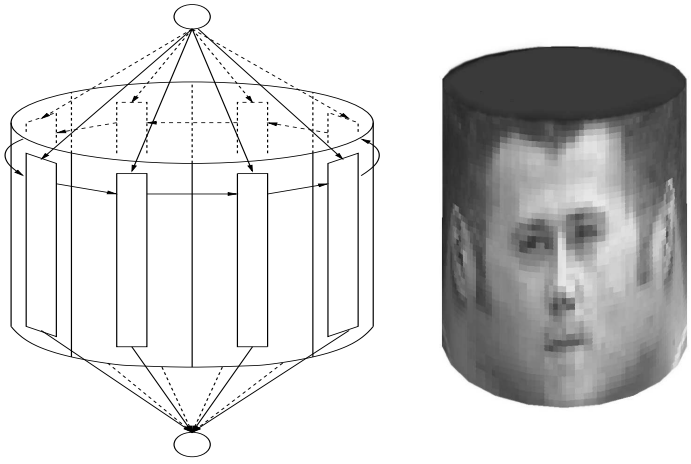


Abbildung 4.17: Interpretation des zyklischen P2D-HMM als Zylinder (aus [SSR08]). Die Rechtecke abstrahieren hier die Spaltenmodelle.

Abbildung 4.18: Eine künstlich generierte Kopf-Rundumansicht, die durch Aneinanderhängen der immer gleichen Bildspalte innerhalb einer Bildsequenz, die eine Kopfdrehung enthält, generiert wird (aus [SSR08]).

Abbildung 4.19: Erzeugung von Trainingsdaten für die Initialisierung des Modells durch manuelles Ausschneiden einer vollen Kopfdrehung. Im vorliegenden Fall beginnt und endet diese bei der Hinterkopfansicht.

Training zyklischer P2D-HMM

Entscheidend beim Training dieser zyklischen Modelle ist es, diese so zu initialisieren, dass tatsächlich gleiche Merkmale enthaltende Bildspalten auf gleiche Spaltenmodelle abgebildet werden und somit dafür zu sorgen, dass nicht wie beim klassischen P2D-HMM verschiedene Merkmalsspalten parallel in ein und dasselbe Spaltenmodell trainiert werden, sondern tatsächlich die relative Merkmalsabfolge modelliert wird. Dazu wird dieses Modell zunächst ohne die zusätzlichen Transitionen, also als klassisches P2D-HMM, mit synthetischen Rundumansichten trainiert (siehe Abbildung 4.18), die immer in gleichartigen Merkmalsspalten beginnen und enden. Dieser Trainingsschritt kann personenunabhängig vorgenommen werden und so zunächst ein allgemeines Modell für Rundumansichten von Köpfen generieren.

Diese synthetischen Rundumansichten werden erzeugt, indem immer die gleiche Bildspalte aus den Trainingsvideos aneinandergehängt wird. Anschließend wird manuell der Start und das Ende dieser Spaltensequenz festgelegt, in dem ein immer ähnlicher Bereich ausgeschnitten wird (hier von Hinterkopfansicht zu Hinterkopfansicht, siehe Abbildung 4.19). Für die erste Initialisierung des zyklischen Modells wird also die Merkmalsabfolge durch manuelle Auswahl vorgegeben. Die hier dargestellten synthetischen Textur-Rundumansichten (Abbildung 4.19) wurden nur zum Verständnis erzeugt. Natürlich werden für das tatsächliche Training nicht Texturelemente aneinandergehängt, sondern die jeweilig an dieser (blockweisen) Bildspalte entstehenden Merkmale, genauer gesagt, die quantisierten DCTmod2 Mermalsvektoren einer Spalte.

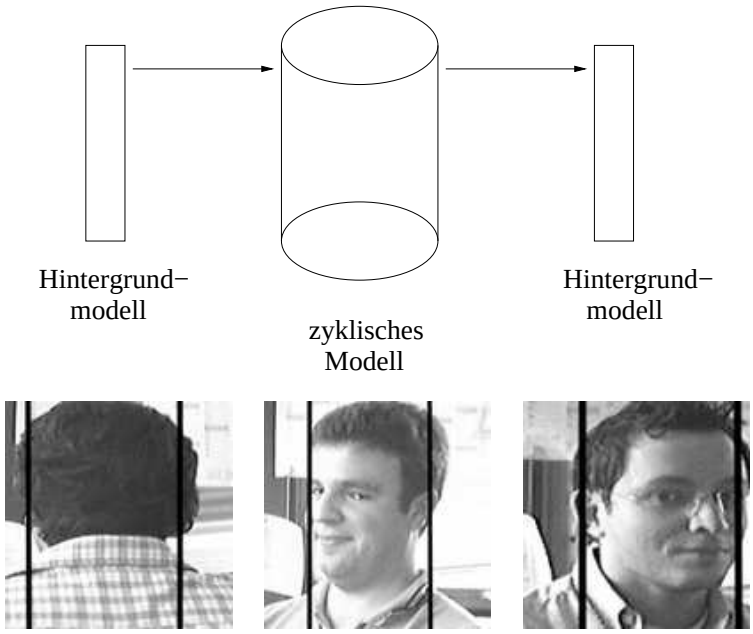


Abbildung 4.20: Anhängen von Hintergrundmodellen links und rechts an das zyklische Modell. Spaltenmodelle sind durch Rechtecke abstrahiert, das zyklische Modell ist durch den Zylinder abstrahiert. Der erwünschte Effekt ist, dass das Modell selbständig zwischen Hintergrund und Personenansicht segmentieren kann, wie es durch die schwarze Linie in den jeweiligen Ausschnitten angedeutet ist.

Durch diese Initialisierung soll zunächst ein grobes Abbilden gleichartiger Merkmalsspalten auf immer die gleichen Spaltenmodelle erreicht werden. Wenn dieses erreicht ist, werden die Transitionen, die aus dem klassischen Modell ein Zyklisches erzeugen, hinzugefügt und ein weiteres Training mit den tatsächlichen Beobachtungen, also den jeweiligen Bildausschnitten mit Kopfansichten in unterschiedlicher Rotation, vorgenommen. Dieser Schritt trainiert nun personenspezifisch, es wird also ein Modell pro Person erzeugt.
Da allerdings die vorliegenden Kopfansichten an den linken und rechten Rändern immer einige Bildspalten enthalten, die den Hintergrund zeigen, wird zusätzlich vor und hinter dem eigentlichen zyklischen P2D-HMM jeweils ein Spaltenmodell eingefügt, das dieses kompensieren soll, siehe Abbildung 4.20. Dieses Spaltenmodell wird mit allen möglichen Hintergrundelementen trainiert. Durch diese Maßnahme soll das dadurch resultierende Gesamtmodell selbständig segmentieren, wo die Bildinhalte von Hintergrund zu Kopfansichten wechselt. Nachdem die Segmentierung eines Trainingsbeispiels in Hintergrundelemente und Personenansicht vorliegt, kann der segmentierte Personenansichtsteil zum personenspezifischen Training des zyklischen P2D-HMM verwendet werden.

Zustände	Kodebucheintr.	Erkennungsrate klassisches P2D-HMM	Erkennungsrate zyklisches P2D-HMM
15×7	500	-	0.864
$\mathbf{10 \times 5}$	500	0.827	**0.889**
5×5	500	**0.837**	0.854
3×3	500	0.803	0.813
1×1	500	0.695	0.674
1×3	500	0.821	0.796

Tabelle 4.5: Erkennungsrate der klassischen und zyklischen P2D-HMM unter Verwendung von nur 8 Blickrichtungen zum Training.

4.5 Ergebnisse

4.5.1 Experimente unter Laborbedingungen

Als Erstes wird hier evaluiert, ob diese neuartige Struktur tatsächlich in der Lage ist, unter gleichen Trainingsvoraussetzungen besser zu generalisieren. Dafür werden zunächst die Experimente auf den einfachen, unter Laborbedingungen aufgenommenen Daten unter Verwendung der zyklischen Struktur wiederholt (siehe Tabelle 4.5). Diese Ergebnisse sind direkt mit den bereits erwähnten Ergebnissen, die mit der klassischen, nicht zyklischen P2D-HMM Struktur erzeugt wurden, zu vergleichen, auch hier wurden nur 8 Trainingsbeispiele pro Klasse verwendet. An den Ergebnissen in Tabelle 4.5 sieht man, dass die zyklischen Modelle leicht besser sind als die klassischen Modelle. Offensichtlich gelingt es also durch diese Struktur zumindest einige horizontale Aspekte in den Merkmalsabfolgen zu modellieren. Besonders dann, wenn viele Spaltenmodelle verwendet werden (z.B. 10×5 Modell) ergibt sich eine deutliche Verbesserung gegenüber der klassischen links-rechts Struktur.
Der Effekt, der durch das zusätzliche Hintergrundmodell (hier Spaltenmodell mit 5 Zuständen) vor und hinter dem eigentlichen personenspezifischen zyklischen Modell erzielt wird, ist schwer zu beurteilen. Zumindest bei den direkt vergleichbaren Fällen mit 1×1 und 1×3 Zuständen wirkt es sich leicht negativ aus. Da dort ja nur eine Spalte oder sogar nur ein Zustand vorkommt, kann sich die zyklische Struktur nicht entfalten, da sich eine Selbsttransition ergeben würde, die in dem einen Spaltenmodell bereits vorkommt. Für diesen Fall ist das zyklische Modell also identisch mit dem klassischen Modell, welches allerdings ohne Hintergrundmodell verwendet wurde. Für die anderen Fälle läßt sich nur schwer eine Aussage treffen, ob das Hintergrundmodell vorteilhaft ist oder nicht, es bleibt lediglich die Aussage, dass Hintergrundmodell und zyklisches Modell, wenn sie zusammen verwendet werden, besser sind als klassische P2D-HMM.
An dieser Stelle sei noch mal darauf hingewiesen, dass sich diese verbesserten Ergeb-

nisse nur erzielen lassen, wenn die zyklischen Modelle mit den synthetischen Rundumansichten initialisiert werden. Nur so kann realisiert werden, dass beim Training tatsächlich gleichartige Merkmale auf gleiche Spaltenmodelle abgebildet werden, ohne diese Maßnahme sind die Erkennungsraten von zyklischem und nicht zyklischem Modell auf vergleichbarem Niveau. Um das Argument zu entkräftigen, dass ja dann mehr Trainingsmaterial zur Verfügung stand, wird noch mal darauf hingewiesen, dass diese Initialisierung personenunabhängig stattfand also ohne klassenspezifisches Trainingsmaterial.

4.5.2 Experimente unter Realweltbedingungen

Als Nächstes wird der hier vorgestellte Ansatz auf der zweiten, anspruchsvolleren Datenbank mit Aufnahmen aus einer Büroumgebung evaluiert. Dabei wird für jede der 25 Personen ein klassenspezifisches Modell anhand der Trainingsdaten erstellt, anschließend werden in der Erkennungsphase alle Bilder der Testdaten klassifiziert, indem das Modell ermittelt wird, für das die Produktionswahrscheinlichkeit maximal ist. Für alle folgenden Auswertungen werden DCTmod2 Merkmale blockweise extrahiert, benachbarte Blöcke überlappen sich dabei um 50%, also um 4 Pixel bei 8×8 Pixel großen Blöcken. Die resultierende Merkmalsvektoren werden mittels k-means Clustering [Mac67] auf 500 Quantisierungsstufen quantisiert, dabei werden zur Erzeugung des Kodebuchs alle Merkmalsvektoren der Trainingsdaten verwendet. Die hier eingesetzten Trainingsdaten sind jeweils 8 Ansichten der Person in den 8 verschiedenen Blickrichtungen (frontaler Blick, linkes und rechtes Profil, Hinterkopf, Halbprofile hinten und vorne), dabei werden jeweils die durch den ASM Detektor gefundenen Ausschnitte verwendet. In Abbildung 4.21 sind einige Beispiele für Trainings- und Testausschnitte zu sehen. Während bei den Trainingsdaten gezielt darauf geachtet wird, nur "gute" Ausschnitte zu verwenden kommen im realen Testfall natürlich auch weniger gute Detektionsergebnisse vor, da diese ja automatisch durch den beschriebenen ASM Ansatz ermittelt werden.

Zum Vergleich werden zunächst Erkennungsergebnisse mittels eines klassischen, nicht zyklischen P2D-HMM generiert. Die Erkennungsraten in Abhängigkeit der Anzahl der verwendeten Zustände sind in Tabelle 4.6 dargestellt. Wie bereits anhand der auf dem einfacheren Daten gewonnenen Erkenntnisse vermutet, schneidet das klassische P2D-HMM bei dieser Aufgabenstellung eher schlecht ab, da es nicht in der Lage ist, aus einzelnen Blickrichtungen die Merkmalsfolgen anderer Blickrichtungen zu generalisieren. Insgesamt konnte keine der getesteten Konfigurationen eine Erkennungsleistung $> 40\%$ erreichen. Dieses unterstreicht auch nochmal die Schwierigkeit der hier gegebenen Erkennungsaufgabe, im Vergleich zu den unter Laborbedingungen erreichten Ergebnissen. Interessant sind hier wieder die Erkennungsleistungen, wenn man mit einem einzelnen Spaltenmodell, also $1 \times N$ Zustände, modelliert, siehe Tabelle 4.6. Da ja bereits festgestellt wurde, dass die vertikalen Beziehungen, also welches Merkmal über welchen anderen Merkmalen liegt, bei der Rotation um die

Abbildung 4.21: Trainingsbeispiele (obere Reihe) und Testbeispiele (untere Reihe) für die gegebene Erkennungsaufgabe.

z-Achse stabil sind, kann dieses Spaltenmodell zumindest die relative vertikale Lage von Merkmalen modellieren und schneidet dadurch verhältnismäßig besser ab, als komplexere Modellierungen mit mehreren Spalten. Da aufgrund der vorgegebenen Modellstruktur mit mehrereren Spalten zwar probiert wird, die relativen horizontalen Lagen der Merkmale zu modellieren, dieses aber nicht gelingen kann, da ja die Merkmalsfolgen immer von links nach rechts abgearbeitet werden, die zyklische Natur der Auftretens dieser Merkmale je nach Blickwinkel aber nicht erfasst werden kann, verschlechtert sich die Erkennungsleistung bei Verwendung mehrerer Spaltenmodelle. Dies lässt sich einfach am folgenden Beispiel verdeutlichen: Die Merkmalsfolge, die nur die erste (blockweise) Bildspalte beschreibt wird immer auf das erste Spaltenmodell abgebildet, egal um welche Ansicht es sich handelt, also welches Gesichtsmerkmal in dem gegebenen Blickwinkel in der ersten Spalte zu sehen ist. Die Gegenprobe, also die Verwendung mehrerer Spaltenmodelle mit jeweils 1 Zustand (z.B. ein 3×1 Modell), zeigt, dass die Erkennungsleistung sogar schlechter ist, als wenn man direkt auf nur einen Zustand abbildet. Dies belegt die Erkenntnis, dass ein klassisches P2D-HMM nicht in der Lage ist, Rotation um die z-Achse zu modellieren.
Der beschriebene Ansatz mit zyklischen P2D-HMM wurde nun ebenfalls auf der gleichen Datenbank validiert. In Tabelle 4.7 sind die Ergebnisse unter Verwendung der gleichen Merkmalsvektoren wie bei dem Experiment zuvor aufgelistet. Im direkten Vergleich der Ergebnisse des klassischen und des zyklischen P2D-HMM sieht man, dass offensichtlich die Modifikationen der Struktur eine Verbesserung herbeiführen. Das zyklische Modell kann deutlich besser die verschiedenen Blickwinkel der Personen modellieren und erzielt bessere Erkennungsraten. Allerdings muss man zu-

Zustände	Kodebuchgröße	Erkennungsrate
9×9	500	0.327
7×7	500	**0.395**
5×5	500	0.311
1×5	500	0.304
3×3	500	0.235
1×3	500	0.256
3×1	500	0.218
1×1	500	0.225

Tabelle 4.6: Ergebnisse mittels klassischem P2D-HMM Ansatz auf den unter Realweltbedingungen aufgenommenen Daten.

Zustände	Kodebuchgröße	Erkennungsrate
3×5	500	0.356
5×5	500	0.392
7×7	500	0.422
7×3	500	0.385
7×5	500	0.411
10×5	500	**0.446**

Tabelle 4.7: Ergebnisse mittels zyklischem P2D-HMM Ansatz auf den unter Realweltbedingungen aufgenommenen Daten mit Vortraining durch künstliche Rundumansichten.

gestehen, dass der betriebene Aufwand beim Training deutlich höher ist, da ja das zyklische Modell mit synthetischen Rundumansichten personenunabhängig vortrainiert wurde und zusätzlich ein Hintergrundmodell verwendet wurde.

Eine abschließende Auswertung findet nun im Hinblick auf die Blickwinkel des Kopfes statt, die durch manuelle Annotation als Informationen vorliegen. Für diese Auswertungen wurde ein zyklisches P2D-HMM verwendet mit 6×6 Zuständen und 500 Kodebucheinträgen. Diesmal wurde aber für die finale Auswertung eine blockweise Überlappung von 75% in der Merkmalsextraktion verwendet, da aus der frontalen Gesichtserkennung bekannt ist, dass diese Maßnahme die Ergebnisse positiv beeinflussen kann.
Die Trainingsschritte umfassten die Initialisierung anhand der synthetischen Rundumansichten, die Erstellung eines Hintergrundmodells, die Segmentierung der Trainingsdaten mittels Hintergrundmodell und allgemeinem personenunabhängigen Kopfmodell und abschließend das personenspezifische Training der Kopfmodelle mit den

segmentierten Daten.

Zunächst wurde die Reklassifikation der Trainingsdaten überprüft, um festzustellen, ob es überhaupt möglich ist, die verschiedenen Ansichten eines Kopfes in ein Modell zu trainieren. Dies ist in Abbildung 4.22 durch die gepunktete Linie dargestellt. Wie man sieht, ist die Reklassifikationsrate nahe 100%, lediglich für einige extreme Ansichten im Hinterkopfbereich sinkt diese leicht ab. Als nächstes wurde die Klassifikation der Testdaten auf manuell ausgeschnittenen Bildern überprüft, es gilt hier die Leistung des Erkenners ohne eventuelle Einflüsse des Detektionsansatzes zu beurteilen. Dies ist durch die hellgraue Kurve dargestellt. Als letztes wurde die Erkennungrate auf den vollautomatisch durch den ASM Ansatz ausgeschnittenen Bildern erzeugt, dieses ist durch die dunkelgraue Kurve dargestellt.
Für die manuell ausgeschnittenen Bilder ergeben sich folgenden Beobachtungen: Wie man erkennen kann, ist im frontalen Bereich von etwa $-30°$ bis $30°$ die Erkennungsrate deutlich höher als für den Rest, in diesem Bereich ist die durchschnittliche Erkennungsrate $66,1\%$. Dies ist ein sehr gutes Ergebnis, wenn man die schlechte Bildqualität und die extremen Beleuchtungsbedingungen bedenkt, also die insgesamt sehr anspruchsvolle Aufgabe. Ebenfalls erkennbar ist, dass offensichtlich frontale Bereiche deutlich robuster erkannt werden als Hinterkopfbereiche, in denen die Erkennungsrate deutlich abfällt. Dies war zu erwarten und entspricht dem gesunden Menschenverstand, da ja in der Hinterkopfansicht deutlich weniger personenspezifische Merkmale zu erkennen sind. Über alle Blickrichtungen gemittelt wurde eine Erkennungsrate von 47.7% erreicht, bei einer Ratewahrscheinlichkeit von 4%.
Unter Anwendung der vollautomatischen Detektion mittels ASM fällt die durchschnittliche Erkennungsrate noch einmal leicht ab, sie erreicht im Mittel 41.4%. Im Vergleich dazu konnte ein klassisches, nicht zyklisches 6×6 P2D-HMM unter Verwendung der gleichen Merkmalsvektoren zum Training und Testen nur eine Erkennungsrate von 37.6% erreichen.

4.6 Zusammenfassung und Diskussion

In diesem Kapitel konnten zwei verschiedene neuartige Ansätze zur blickrichtungsunabhängigen Personenerkennung vorgestellt werden. Der erste Ansatz unter Verwendung eines Active Appearance Modells ist zwar nur in der Lage, Ansichten die in etwa zwischen linker und rechter Halbprofil-Ansicht liegen, zu klassifizieren, kann aber selbständig bestimmen, ob eine gegebene Ansicht diese Anforderung erfüllt oder nicht. Daraus ergeben sich durchaus sinnvolle Einsatzmöglichkeiten, vor allem in Verbindung mit Trackingalgorithmen. Wenn ein Kopfausschnitt durch einen Trackingalgorithmus im Bild verfolgt wird, versucht der Erkennungsansatz solange die Identität festzustellen, bis dieses mit hoher Konfidenz erreicht wurde. Dann bedarf es nur noch Überprüfungen von Zeit zu Zeit, oder bei Neuinitialisierung des

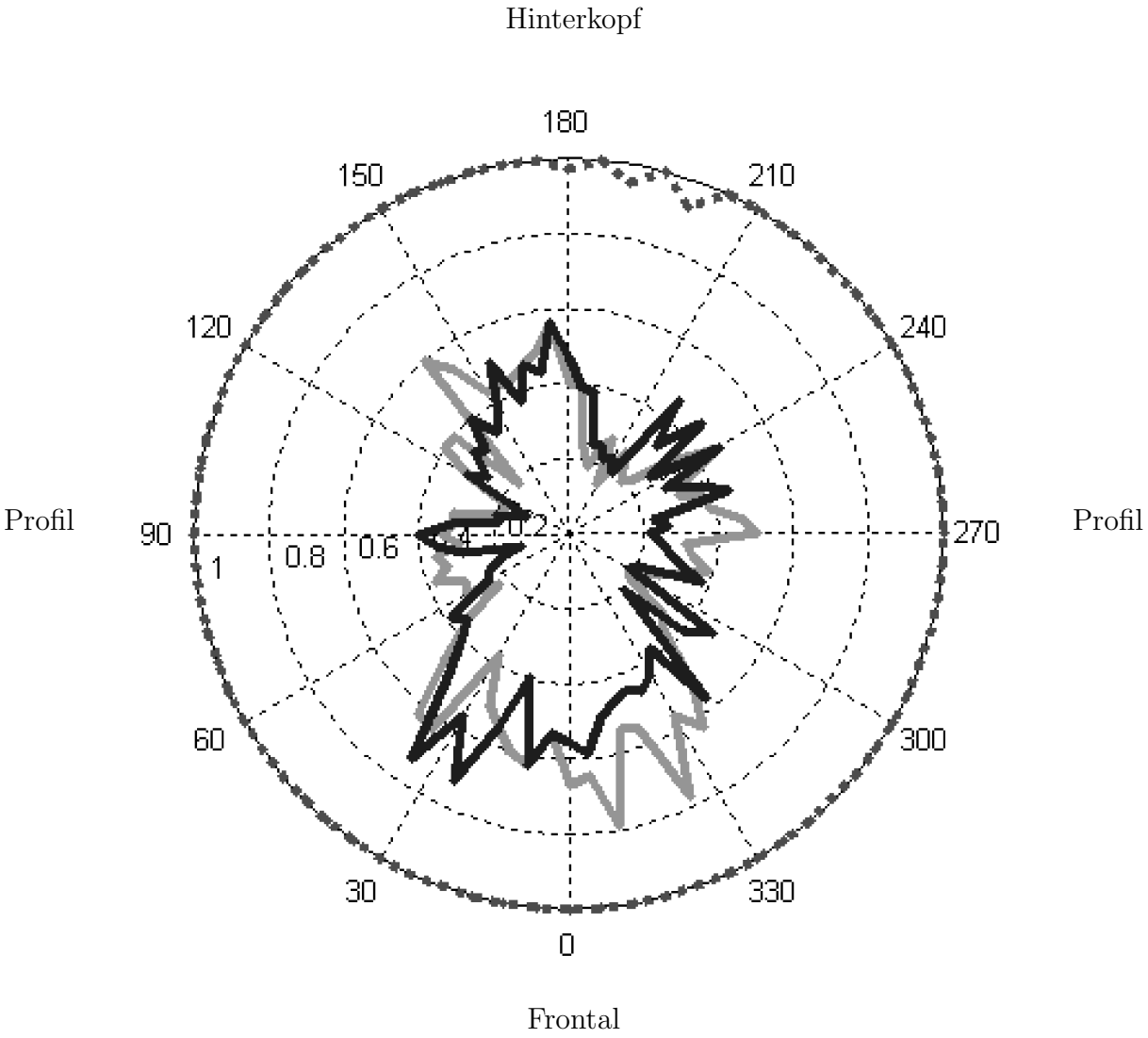


Abbildung 4.22: Erkennungsraten der Reklassifikation (gepunktete Linie) und Klassifikation auf manuell ausgeschnitten Testdaten (hellgraue Kurve) und auf den durch das ASM getrackten Testdaten (dunkelgraue Kurve)

Trackers. Wenn man sich die Erkennungsleistung des beschriebenen Ansatzes anschaut, so ist die Klassifizierungsaussage für hohe Konfidenzwerte mit 92.5% sehr zuverlässig, so dass eine Umsetzung dieses Ansatzes in einer realen Anwendung denkbar wäre.

Der zweite Ansatz ist in der Lage, die Ansichten aus allen Blickrichtungen zu klassifizieren, wobei es natürlich blickrichtungsabhängige Leistungsunterschiede gibt. Insgesamt ist die Erkennungsaussage auf einem niedrigerem Niveau. Aber auch hier ist es denkbar, die rohen Erkennungsergebnisse durch einfache Maßnahmen zu verbessern. Eine denkbare, leicht zu realisierende Möglichkeit wäre es, nicht nur ein einzelnes Bild zu klassifizieren, sondern eine Sequenz von mehreren Bildern, um dann durch Mehrheitsentscheidungen oder andere Fusionsstrategien zu robusteren Ergebnissen zu kommen. Insgesamt konnte in den Untersuchungen gezeigt werden, dass eine Struktur, wie sie durch zyklische P2D-HMM erzeugt werden kann, besser in der Lage ist, verschiedene Blickrichtungen in ein Modell aufzunehmen, als die durch die frontale Gesichtserkennung bekannte klassische P2D-HMM Struktur. Auf den Daten, die unter Laborbedingungen aufgenommen wurden, konnten sehr hohe Erkennungsraten von ca. 89% bei Verwendung von nur 8 Trainingsansichten erreicht werden. Für die Versuche auf den Daten in einer Büroumgebung, mit schlechter Bildqualität und starken Beleuchtungsvariationen, konnte im Mittel immerhin noch eine Erkennungsrate von etwa 40% bei einer Ratewahrscheinlichkeit von 4% erreicht werden. Diese Erkennungsraten schließen Hinterkopf- und andere Extremansichten mit ein, ebenso durch die automatische Detektion auftretende Fehldetektionen.

Die zugrunde liegende Modellannahme der Projektion der Gesichtstextur auf eine Zylinderoberfläche ist nur näherungsweise richtig, besonders in den Ansichten nahe der Profilansicht weichen die tatsächlichen Oberflächennormalen des Gesichtes stark von denen eines Zylinders ab. Zukünftige Verbesserungen dieses Ansatzes könnten an dieser Stelle ansetzen. Sinnvoll wäre es, gezielt nach Merkmalen zu suchen, die eine starke Tiefenrotationsinvarianz besitzen, oder aber die Veränderung der Merkmale in Abhängigkeit des Blickwinkels explizit zu modellieren.

Neben der Ausrichtung, die bereits durch die Detektion mittels Active Shape Modell stattfindet, gilt in beiden Erkennungsansätzen, dass eine weitere implizite Ausrichtung der Daten durchgeführt wird. Im Active Appearance Modell ist die Parameterschätzung des Formmodells ein Ausrichtungsschritt, der optimiert, welche Bildbereiche überhaupt von Belang sind. Im Falle des zyklischen P2D-HMM ist zum einen die Segmentierung des Hintergrundes eine Selektion des zu verwendenden Ausschnittes, zum anderen ist das Abbilden von Merkmalen auf Zuständen auch eine Art implizite Ausrichtung.

5

Paarweiser Vergleich von Gesichtsbildern

5.1 Einleitung

Der paarweise Vergleich (engl. “pair matching”) ist ein erst in den letzten Jahren aufgekommenes Anwendungsgebiet der Gesichtserkennung. Es wird davon ausgegangen, dass zwei Bilder vorliegen, in denen Gesichter vorkommen. Durch einen geeigneten Algorithmus soll nun entschieden werden, ob in den beiden Bildern die gleiche Person vorkommt, oder nicht. Anders als bei der Authentifikation liegt aber kein klassenspezifisches Trainingsmaterial vor, sondern nur Domänenspezifisches, also in diesem Fall Gesichtsbilder, allerdings ohne dass die Personen des Testfalls enthalten sind. Typischerweise wird der Vergleich auf Daten vorgenommen, die eine hohe Variabilität in Bezug auf Beleuchtung, Hintergrund, Pose, Geschlecht und Mimik aufweisen. Insbesondere die in den jeweiligen Bildpaaren vorkommenden unterschiedlichen Blickwinkel haben auch hier einen erheblichen Einfluss auf die Erkennungsaufgabe. Die hohe Variabilität der Daten in dieser Aufgabe wird auch durch den engl. Fachbegriff “unconstrained recognition” beschrieben, frei übersetzt etwa “Erkennung ohne einschränkende Bedingungen”. Das Ziel ist es, möglichst viele Anforderungen von Anwendungen der Realwelt abzubilden. Dadurch ist die Aufgabenstellung, auch wenn es sich nur um ein Zwei-Klassen-Problem handelt, durchaus sehr anspruchsvoll.

Geeignete Daten zum Vergleich verschiedener Verfahren liegen mit dem für die Forschung frei erhältlichen “Labeled Faces in the Wild”-Datensatz [HRBLM07] vor. Dieser Datensatz wurde erzeugt, indem alle möglichen Bildquellen im Internet mit einem Viola und Jones Gesichtsdetektor (siehe 2.1.2) abgesucht wurden. Alle Bilder, in denen man Gesichter fand, wurden so abgespeichert, dass das detektierte Gesicht in der Mitte des Bildes ist und ein relativ zur Gesichtsgröße definierter Ausschnitt ausgeschnitten wurde. Anschließend wurden alle falsch positiven Detektionen aussortiert und die Bilder so umbenannt, dass sie den Namen der darin enthaltenen Person

Abbildung 5.1: Beispiele aus der "Labeled Faces in the Wild"-Datenbank". Bildpaare, die gleiche Personen enthalten, sind in der oberen Reihe dargestellt, Bildpaare mit unterschiedlichen Personen in der unteren Reihen. Eine hohe Variabilität in Bezug auf Blickrichtung, Beleuchtung, Mimik, Bekleidung, Styling (Kosmetik, Frisur) und Hintergrund erschwert die Erkennungsaufgabe.

zuzüglich einer Nummerierung erhielten. Aus den somit erhaltenen 13233 Bildern von 5741 verschiedenen Personen wurden nun 10 Validierungssets mit jeweils 600 Bildpaaren gebildet. Dabei enthalten jeweils 300 Paare Bilder der gleichen Person und 300 Paare Bilder unterschiedlicher Personen. Beim Erstellen der Sets wurde darauf geachtet, dass sie personendisjunkt sind. Jede Person kommt nur in einem Validierungsset, zusätzlich nur entweder im Trainings- oder im Testteil dieses Sets, vor, darin aber durchaus in unterschiedlichen Bildern und Bildpaaren. Einige Beispielbildpaare sind in Abbildung 5.1 dargestellt. Wenn man sich die gestellte Aufgabe hinsichtlich der Verarbeitungskette Detektion, Ausrichtung, Erkennung anschaut, so stellt man fest, dass durch die Art der Erstellung der Datenbank, nämlich durch Anwenden eines Gesichtsdetektors auf unzählige Bilder, der erste Schritt dieser Kette bereits unwiderruflich abgearbeitet wurde. Wie man anhand der Beispieldaten (Abbildung 5.1) erkennen kann, ist die gestellte Erkennungsaufgabe sehr anspruchsvoll, und beinhaltet neben der Variation der Blickrichtung auch weitere Einflüsse.

5.2 Vorverarbeitung

Da die Bilder unterschiedlichste Beleuchtungscharakteristika haben, bietet es sich an, zunächst eine Beleuchtungskorrektur auf die Bilder anzuwenden, bevor diese weiterverarbeitet werden. Im vorliegenden Fall wurde die bereits im Kapitel 3 vorgestellte Beleuchtungskorrektur durch Histogrammausgleich verwendet. Im Hinblick auf die hohe Variabilität der Beleuchtung innerhalb der Daten, könnten hier in Zukunft weitere Verbesserungen erzielt werden, indem gezielt andere Verfahren ausgetestet werden.

5.3 Ausrichtung

Da die Daten dadurch erstellt wurden, dass ein Gesichtsdetektor auf Bilder aus dem Internet angewandt wurde, ist der Detektionsschritt in der Datenbank mit enthalten. Ein Ausrichtungsschritt, der die doch relativ groben Ausschnitte die ein Detektor liefert, aufeinander ausrichtet, somit einen normierten Gesichtsausschnitt liefert, wäre vorteilhaft. Im Zusammenhang mit der Datenbank "Labeled Faces in the Wild" werden bereits Daten, die mit der Methode des "Funneling" ausgerichtet wurden, mitgeliefert. Es bietet sich also an, bei der Auswertung Ergebnisse auf ausgerichteten und nicht ausgerichteten Daten zu vergleichen, dieses geschieht in der Sektion 5.5.

Hier wird nun zunächst die Methode des "Funneling" [HJLM07] kurz vorgestellt, welches eine Weiterentwicklung des "concealing" [LM06] ist. Der Name dieser Methode ergibt sich aus dem englischen Begriff "funnel", welches übersetzt "Trichter" bedeutet und beschreibt das schrittweise Ausrichten, welches zunächst grob beginnt und immer präziser wird, ähnlich einem immer enger werdenden Trichter.
Das Basiskonzept dieser Methode ist das Verteilungsfeld. Wenn $\chi = 1, 2, .., M$ der Satz aller möglichen Merkmalswerte an einer konkreten Pixelposition ist, ist das Verteilungsfeld die Verteilung von χ an jeder Pixelposition im Bild. Eine einfache Möglichkeit, diese zu schätzen, ist die Auftretenshäufigkeit der einzelnen Merkmalswerte anhand eines Satzes von Beispielbildern zu berechnen. Dieses Verteilungsfeld kann als unabhängiges generatives Pixelmodell für Bilder interpretiert werden, bei der man an jede Pixelposition i eine Zufallsvariable χ_i positioniert. Ein Bild wird dann dadurch generiert, dass aus dem Satz der möglichen Merkmalswerte χ für jedes χ_i ein zufälliger Merkmalswert basierend auf der gegebenen Verteilung gezogen wird.
Der eigentliche Ausrichtungsprozess geschieht nun dadurch, dass jedes einzelne Bild mittels einer affinen Abbildung so transformiert wird, dass die Entropie des resultierenden gesamten Verteilungsfeldes aller Bilder minimiert wird. Bei Annahme voneinander unabhängiger Pixelmodelle und Gleichverteilung der auftretenden Transformationen ist dieses gleichwertig zu einer Maximierung der Wahrscheinlichkeit jedes einzelnen Bildes bezogen auf das Verteilungsfeld anhand der Wahl der Transformationsparameter. In einfachen Worten bedeutet dieses, jedes Bild wird auf das Verteilungsfeld, welches aus allen Bildern erzeugt wurde, ausgerichtet. Danach kann aus den resultierenden Bildern wieder ein neues Verteilungsfeld berechnet werden, welches eine geringere Entropie hat, als das der vorhergehenden Stufe. Dieser Prozess des abwechselnden Berechnen des Verteilungsfeldes und des Ausrichten der einzelnen Bilder des Trainingsbildsatzes wird solange wiederholt, bis Konvergenz erreicht ist. Die dabei entstandenen Verteilungsfelder der einzelnen Iterationsstufen bezeichnet man nun als "image funnel" oder "Bildtrichter". Dieser besteht aus "weiten" Verteilungsfeldern, also welche mit hoher Entropie, die schrittweise immer fokussierter

Abbildung 5.2: Ergebnisse des “Funneling” (aus [HJLM07]).

werden, also immer geringere Entropie enthalten. Dieser “Bildtrichter” ermöglicht es, Bilder, die nicht in den Trainingsdaten enthalten sind, auszurichten, indem sie in der gleichen Reihenfolge schrittweise auf die einzelnen Verteilungsfelder ausgerichtet werden. Einige Ergebnisse des mit der Datenbank mitgelieferten Ansatzes sind in Abbildung 5.2 abgebildet. Die in die ursprünglichen Bilder eingezeichneten roten Rechtecke stellen den Bereich dar, den man auf die gesamte Bildfläche abbilden müsste, um den ausgerichteten Bildausschnitt zu erhalten. Für Interessierte sei an dieser Stelle auf eine ausführliche Erläuterung dieses Ansatzes in [HJLM07] hingewiesen.

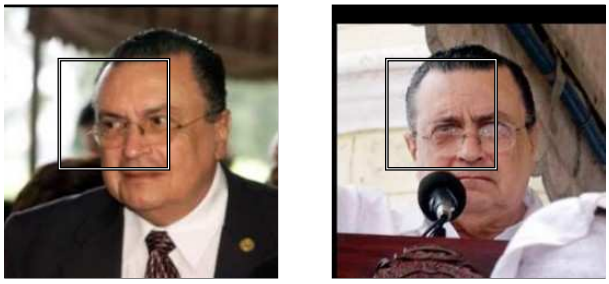


Abbildung 5.3: Vergleich zweier Bilder durch Vergleich von einzelnen Bildregionen: Örtlich korrespondierende Bildregionen werden verglichen um einen Ähnlichkeitswert zu erzeugen. Aus mehreren solchen Vergleichen von Bildregionen ensteht dann die Merkmalssequenz bestehend aus Ähnlichkeitswerten.

5.4 Erkennung durch eine Summe gewichtete Ähnlichkeitsmaße

Der dem hier entwickelten neuartigen Verfahren zugrunde liegende Gedanke ist Folgender: Die zu vergleichenden Bilder werden in mehrere, gleichmäßig im Bild verteilte, sich überlappende Ausschnitte zerteilt, genau wie mit einem "sliding window" fest definierter Größe und Schrittweite. Mittels verschiedenster Ähnlichkeits- und Distanzmaße, basierend auf unterschiedlichen Merkmalen, werden nur die Ähnlichkeiten jeweils örtlich korrespondierender Bildausschnitte der zwei Bilder berechnet (siehe Abbildung 5.3) und in einem Merkmalsvektor hinterlegt. Eine gewichtete Summe dieser einzelnen Ähnlichkeiten soll eine Gesamtähnlichkeit beschreiben. Dabei werden die einzelnen Gewichte dieser Summe durch ein Lernverfahren bestimmt. Es wird also trainiert, wie stark die Aussagekraft jedes einzelnen Ausschnittes des Bildes, unter Verwendung der unterschiedlichen Ähnlichkeitsmaße ist.

5.4.1 Ähnlichkeitsmaße für Bildausschnitte

Da in der vorliegenden Aufgabenstellung davon ausgegangen werden muss, dass durch mögliche Variationen der Pose die Inhalte der zu vergleichenden Ausschnitte nicht exakt örtlich korrespondieren, macht es wenig Sinn, Distanz- oder Ähnlichkeitsmaße zu verwenden, die statisch die örtliche Grauwertverteilung beurteilen. Für diese Aufgabenstellung sinnvolle Maße sollten zumindest innerhalb einer begrenzten Umgebung ortsinvariant sein, da eine je nach Ausschnittsgröße mehr oder weniger grobe örtliche Information allein schon durch die Zerlegung des Bildes in einzelne Ausschnitte bewahrt wird. Ein geeigneter Ansatz, um dieses zu Erreichen, ist die Verwendung von Histogrammen von Merkmalshäufigkeiten, da bei der Erstellung von Histogrammen jede örtliche Information vernachlässigt wird und nur noch die Häufigkeiten des Auftretens bestimmter Merkmalswerte eine Rolle spielen.

Eine weitere Möglichkeit besteht darin, die Ähnlichkeit basierend auf statistischen Kenngrößen bestimmter Charakertistika wie Luminanz, Kontrast und Struktur zu bewerten, oder aber die Ähnlichkeit im Ortsfrequenzbereich zu ermitteln, die ebenfalls eine relativ ortsinvariante Bewertung ermöglicht.
Die in diesem Kontext verwendeten Merkmale werden im Folgenden vorgestellt. Es handelt sich hierbei sowohl um Merkmalshistogramm-basierte Ähnlichkeitsmaße (Local Binary Pattern, Three Patch Local Binary Pattern, Histogram of Gradients) als auch um direkte Ähnlichkeitsmaße (Structural Similarity, Ortsfrequenz-basierte Ähnlichkeit).

Histogramme lokaler Binärmuster (Local Binary Pattern (LBP))

Ein Textur Deskriptor, der im Englischen "local binary pattern" (LBP) genannt wird, hat sich in der Objekt- und Gesichtserkennung bereits bewährt (siehe [OPM02], [MRH06] und [AHP06]). LBP werden dadurch erzeugt, dass der Mittelwert eines zentralen Bildbereiches mit den Mittelwerten umherliegender Bildbereiche verglichen wird. Wenn der Wert des zentralen Bereiches niedriger ist als der eines umherliegenden Bildbereiches gleicher Grösse, wird der resultierende Wert auf "1" gesetzt, andernfalls auf "0". Ein Binärmuster wird nun dadurch erzeugt, dass der zentrale Bildbereich im Uhrzeigersinn mit allen darumliegenden Bildbereichen verglichen wird, dieses Muster wird als LBP bezeichnet und als Deskriptor verwendet (siehe Abbildung 5.4).

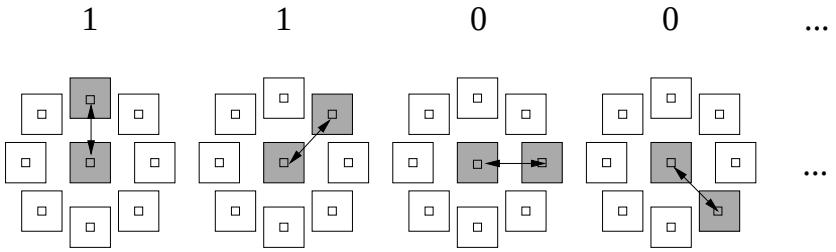


Abbildung 5.4: Local binary pattern: Im Uhrzeigersinn wird verglichen, ob der Mittelwert des zentralen Bildbereiches kleiner ist als der des benachbarten Bildbereiches. Die binären Ergebnisse ("kleiner" oder "nicht kleiner") werden in einer Bitsequenz gespeichert.

Im Kontext dieser Arbeit, werden acht Nachbarregionen, die auf einem Umkreis mit einem Radius von 3 oder 5 Pixeln bezogen auf das Zentralpixel des zentralen Ausschnittes liegen, verwendet. Die Größe jeder Region ist dabei entweder 3×3 oder 5×5 Pixel. Nachdem die resultieren Bitsequenzen ermittelt wurden, werden sie dahingehend überprüft, ob sie gleichförmige Muster (engl. "uniform patterns") sind oder nicht. Dabei wird die Anzahl der Transitionen von "0" auf "1" oder von

"1" auf "0" gezählt. Ist diese größer zwei, spricht man von einem nicht gleichförmigen Muster. Beispiel: 00111110 ist ein gleichförmiges Muster, 01100101 nicht. Als Hintergrund dieser Unterscheidung sei an die kreisförmige Anordnung der Nachbarregionen erinnert, bei gleichförmigen Mustern ist genau ein Kreissegment anders als der Rest. Das dadurch entstehende Bitmuster gilt nun als charakteristisches texturbeschreibendes Muster für das Zentralpixel des untersuchten Bildbereichs. In diesem Merkmal werden also im Grunde genommen Richtungsinformationen in Form eines Kreissegmentes hinterlegt.
Dieser Texturdeskriptor LBP wird nun für jedes Pixel innerhalb eines Bildausschnitts berechnet. Anschließend wird, um eine gewisse Ortsinvarianz zu gewährleisten, ein Histogramm über die Auftretenshäufigkeiten der Binärmuster erstellt. Hierbei werden alle "non-uniform patterns" also alle nicht einheitlichen Muster in einem Histogrammabschnitt (engl. "bin") vereinigt, alle gleichförmigen Muster erhalten jeweils einen eigenen Histogrammabschnitt.

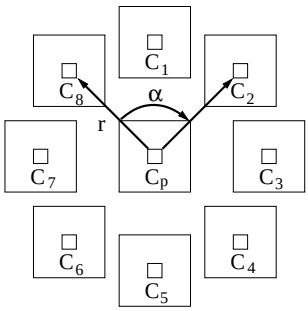
Eine Erweiterung der "Local Binary Patterns" sind die sogenannten "Three Patch Local Binary Patterns"(TPLBP). Wie der Name schon sagt, werden hier drei Bildbereiche miteinander verglichen. Hier werden allerdings nicht mehr Mittelwerte von Bildregionen verglichen, sondern es wird ein Distanzmaß verwendet, z.B. die euklidische Distanz zwischen jeweils zwei Bildausschnitten. Ähnlich wie bei den normalen Local Binary Patterns geht man hier von einer zentralen Bildregion P_c und darumliegenden Bildregionen P_i aus. Für jeweils einen festen Abstand α zwischen Bildausschnitten, die auf dem Umkreis liegen, wird untersucht, welcher von den zwei auf dem Umkreis liegenden Bildausschnitten ähnlicher zum zentralen Bildausschnitt ist (siehe Abbildung 5.5). Man geht davon aus, dass die auf dem Umkreis liegenden Ausschnitte im Uhrzeigersinn durchnummeriert werden. α ist dann ein Abstand im Index, oder anders interpretiert ein diskreter Winkelabstand zwischen den zur Verfügung stehenden Nachbarrichtungen. Im konkreten Fall werden acht auf dem Umkreis liegende Ausschnitte $P_{i=1..8}$ angenommen, der Radius des Umkreises sei r. Eine achtstellige Bitsequenz $\sum_{i=1}^{8} f(x) \cdot 2^{(i-1)}$ (als Binärzahl interpretiert) wird nun dadurch erzeugt, dass der Vergleich für alle Paare mit dem Indexabstand α im Uhrzeigersinn durchgeführt wird.

$$\mathrm{TPLBP}_{r,\alpha} = \sum_{i=1}^{8} f(d(P_i, P_c) - d(P_{((i-1+\alpha) \bmod 8)+1}, P_c)) \cdot 2^{(i-1)} \tag{5.1}$$

mit

$$f(x) = \begin{cases} 1 & \text{falls} \quad x > 0 \\ 0 & \text{sonst} \end{cases} \tag{5.2}$$

Auch das TPLBP wird für jeden Pixel innerhalb einer Bildregion berechnet. Anschließend wird, wie bei den LBP, zwischen gleichförmigen und nicht gleichförmigen Bitmustern unterschieden und ein Histogramm über die Auftretenshäufigkeit



$$
\begin{aligned}
&\mathrm{TPLBP}_{r,2}(p) = \\
&f(d(P_1,P_c) - d(P_3,P_c))2^0 + \\
&f(d(P_2,P_c) - d(P_4,P_c))2^1 + \\
&f(d(P_3,P_c) - d(P_5,P_c))2^2 + \\
&f(d(P_4,P_c) - d(P_6,P_c))2^3 + \\
&f(d(P_5,P_c) - d(P_7,P_c))2^4 + \\
&f(d(P_6,P_c) - d(P_8,P_c))2^5 + \\
&f(d(P_7,P_c) - d(P_1,P_c))2^6 + \\
&f(d(P_8,P_c) - d(P_2,P_c))2^7
\end{aligned}
$$

Abbildung 5.5: Berechnung des "Three Patch Local Binary Pattern", hier mit dem festen Indexabstand $\alpha = 2$.

der einzelnen gleichförmigen Bitmuster erstellt. Nicht gleichförmige Muster erhalten ebenfalls wieder nur einen gemeinsamen Histogrammabschnitt.

Im Rahmen dieser Arbeit wurden acht verschiedene TPLBP-Histogramme mit $\alpha = 2..4$ verwendet, für jeden möglichen Richtungsvergleich wurde ein eigenes Histogramm erstellt. Die Stärke dieser Binärmerkmale ergibt sich aus mehreren Teilaspekten. Der erste unmittelbar einleuchtenden Aspekt ist, dass durch die relativen Vergleiche benachbarter Ausschnitte die lokale Gesamthelligkeit nur eine geringe Rolle spielt, diese Merkmale sind also zu einem gewissen Grad beleuchtungsinvariant. Ein weiterer Aspekt ist, dass sie Richtungsinformationen zu lokalen Mustern extrahieren, also bewerten in welchen Richtungen sich lokale Strukturen wiederholen und somit ähnlich sind. In Kombination mit einem regionalen Histogramm, abstrahieren sie diese lokale Information zu einer zusammengefassten Beschreibung der Bildregion. Dieses wirkt sich besonders in diesem Anwendungsfall mit unterschiedlichen Gesichtsposen und Helligkeiten positiv aus.

Histogramm von Gradienten (HoG)

Das Histogramm von Gradienten (engl. "Histogram of oriented Gradients (HoG)") ist ein weit verbreiteter Deskriptor und hat sich als erfolgreiches Merkmal im Bereich der Objekt- und Personendetektion bewährt (siehe [SRBB06],[DT05] und [AMM+08]). Besonders wegen der hohen Variabilität der Daten in Bezug auf Beleuchtung und Pose ist dieses Merkmal für die gegebene Aufgabenstellung geeignet.

Um dieses Merkmal für einen gegebenen Bildausschnitt I zu berechnen, werden zunächst die Gradienten durch die partiellen Ableitungen in x- und y Richtung berechnet. In der Regel werden diese durch Sobel-Operatoren S_x und S_y genähert.

$$\nabla(I(x,y)) = \begin{pmatrix} \mathrm{S}_x(I(x,y)) \\ \mathrm{S}_y(I(x,y)) \end{pmatrix} \tag{5.3}$$

Der Betrag des Gradienten $|\nabla(I(x,y))|$ kann dann folgendermaßen bestimmt werden:

$$|\nabla(I(x,y))| = \sqrt{(\mathrm{S}_x(I(x,y)))^2 + (\mathrm{S}_y(I(x,y)))^2} \tag{5.4}$$

Die Orientierung des Gradienten, also die Richtung in die er zeigt, kann durch

$$\Theta(x,y) = \arctan(\mathrm{S}_x(I(x,y)), \mathrm{S}_y(I(x,y))) \tag{5.5}$$

bestimmt werden. Nachdem diese Werte für jeden Pixel bestimmt wurden, werden typischerweise noch zusätzliche weitere Berechnungschritte eingeführt, um das Gradientenfeld auszudünnen. Gängige Methoden sind hier entweder das Anlegen eines Schwellwertes auf dem Gradientenbetrag, oder eine sogenannte "Non-Maximum-Supression" also eine Unterdrückung aller Gradienten, deren Betrag kein lokales Maximum bezogen auf die Nachbarn in Gradientenrichtung darstellt. Dafür wird typischerweise die Anzahl der möglichen Richtungen diskretisiert. Durch das vorgegebene Pixelraster bietet sich hier die Verwendung der gängigen acht Richtungen an, die zwischen einem Pixel und den acht direkten Nachbarpixeln entstehen. Für die "Non-Maximum-Supression" wird nun überprüft, ob der Betrag des Gradienten an der gerade betrachteten Stelle höher ist, als der Betrag der Gradienten aller Nachbarpixel in Gradientenrichtung.

Ein Histogramm wird nun, nachdem das Gradientenfeld berechnet und ausgedünnt wurde, erstellt, indem die Auftretenshäufigkeit der möglichen diskretisierten Gradientenrichtungen ermittelt wird, jede mögliche Richtung erhält einen Histogrammabschnitt. Als Variante des HoG Merkmals besteht weiterhin die Möglichkeit, zusätzlich den Betrag des Gradienten zu diskretisieren oder Gradientenoperatoren verschiedener Skalierung zu verwenden, um ein feiner gegliedertes Histogramm zu erstellen. Im Rahmen dieser Arbeit wurde die Basisvariante mit Diskretisierung der Richtungsinformation auf acht Orientierungen und Verwendung der "Non-Maximum-Supression" verwendet, ohne weitere Unterscheidungen im Hinblick auf den absoluten Betrag durchzuführen.

Erzeugung eines Ähnlichkeitsmaßes aus Histogrammen

Um aus zwei Histogrammen eine Ähnlichkeitsaussage zu erzeugen, werden diese mit Hilfe eines geeigneten Ähnlichkeitsmaßes kombiniert. Für den hier beschriebenen Ansatz werden die Histogramme gleicher Bildausschnitte der unterschiedlichen Bilder I_a und I_b zu jeweils einem ortsbezogenem Ähnlichkeitsmaß zusammengefasst, welches schließlich nur noch durch einen skalaren Wert repräsentiert wird. Dadurch lässt sich eine erhebliche Dimensionsreduzierung für weitere, danach folgende Mustererkennungschritte bewerkstelligen, da zwei je nach Quantisierung doch eher höherdimensionale Histogramme auf ein Skalar abgebildet werden.

Im Rahmen der hier durchgeführten Experimente wurde als Ähnlichkeitsmaß ein symmetrisches Maß der χ^2- Familie verwendet, welches auch als "probabilistic symmetric χ^2" [Cha08] bezeichnet wird:

$$s_{p\chi^2}(H_1, H_2) = 2 \sum_{i=1}^{N} \frac{(H_{1,i} - H_{2,i})^2}{(H_{1,i} + H_{2,i})} \tag{5.6}$$

Hier sind H_1 und H_2 jeweils die zu vergleichenden Histogramme mit jeweils N Histogrammabschnitten (engl. "bins").

Ähnlichkeit der Bildstruktur, Structural Similarity (SSIM)

Die Ähnlichkeit der Bildstruktur, engl. "Structural Similarity (SSIM)" [WBSS04], ist ein Ähnlichkeitsmaß, das auf der Annahme basierend vorgeschlagen wurde, dass das menschliche visuelle System stark an das Extrahieren struktureller Informationen aus dem Blickfeld angepasst ist. SSIM vergleicht lokal die Aspekte Luminanz $l(I_a, I_b)$, Kontrast $c(I_a, I_b)$ und Struktur $s(I_a, I_b)$ zweier Bildausschnitte I_a und I_b und führt diese in ein gemeinsames Maß über:

$$\mathrm{SSIM}(I_a, I_b) = f(l(I_a, I_b), c(I_a, I_b), s(I_a, I_b)) \tag{5.7}$$

Die einzelnen Aspekte werden beschrieben durch statistische Maßeinheiten wie Mittelwert (μ_a und μ_b), Standardabweichung (σ_a und σ_b) sowie Kovarianz σ_{ab} der Intensitäten in den Bildbereichen I_a und I_b. Zunächst werden die Mittelwerte der Bildausschnitte berechnet, angenommen die Bildausschnitte I_a und I_b seien $N \times M$ Pixel groß:

$$\mu_a = \frac{1}{NM} \sum_{x=1}^{N} \sum_{y=1}^{M} I_a(x, y) \tag{5.8}$$

Anschließend werden die Standardabweichungen bestimmt:

$$\sigma_a = \sqrt{\frac{1}{NM - 1} \sum_{x=1}^{N} \sum_{y=1}^{N} (I_a(x, y) - \mu_a)^2} \tag{5.9}$$

Die Berechnungen für μ_b und σ_b sind entsprechend bezogen auf I_b.

Ein Vergleich der Luminanz $l(I_a, I_b)$ wird nun erreicht, indem der Mittelwert als charakteristische Beschreibung der Luminanz interpretiert wird, also werden die Luminanzwerte der beiden Bildausschnitte verglichen durch

$$l(I_a, I_b) = \frac{2\mu_a\mu_b + C_1}{\mu_a^2 + \mu_b^2 + C_1}. \tag{5.10}$$

Hierbei ist die Konstante C_1 dafür da, eine Instabilität dieses Ausdrucks zu vermeiden, falls $\mu_a^2 + \mu_b^2$ sehr nahe an Null ist. Ähnlich dazu, wird der Kontrastvergleich

zweier Bildausschnitte formuliert, als beschreibendes Maß für den Kontrast wird die Standardabweichung verwendet:

$$c(I_a, I_b) = \frac{2\sigma_a\sigma_b + C_2}{\sigma_a^2 + \sigma_b^2 + C_2} \tag{5.11}$$

Der Vergleich der Bildstruktur wird nach Subtraktion der Luminanz und Normierung der Varianz durchgeführt. Die dadurch resultierenden Bildausschnitte $(I_a - \mu_a)/\sigma_a$ und $(I_b - \mu_b)/\sigma_b$ werden durch ihr inneres Produkt miteinander verglichen. Mathematisch entspricht dies der Berechnung des Korrelationskoeffizienten zwischen I_a und I_b. Zunächst wird die Kovarianz σ_{ab} berechnet:

$$\sigma_{ab} = \frac{1}{NM-1}\sum_{x=1}^{N}\sum_{y=1}^{M}(I_a(x,y) - \mu_a)(I_b(x,y) - \mu_b) \tag{5.12}$$

Die Kovarianz σ_{ab} dividiert durch das Produkt der Standardabweichungen $\sigma_a\sigma_b$ entspricht genau dem Korrelationskoeffizienten. Dieser wird als Strukturmaß $s(I_a, I_b)$ hergenommen, allerdings wird auch hier wieder eine Konstante C_3 eingeführt um Instabilitäten bei sehr kleinen Standardabweichungen zu vermeiden.

$$s(I_a, I_b) = \frac{\sigma_{ab} + C_3}{\sigma_a\sigma_b + C_3} \tag{5.13}$$

Abschließend werden die drei Ausdrücke für l,c und s zu einem gemeinsamen Maß zusammengeführt:

$$SSIM(I_a, I_b) = [l(I_a, I_b)]^\alpha [c(I_a, I_b)]^\beta [s(I_a, I_b)]^\gamma \tag{5.14}$$

hierbei sind α, β, und γ Parameter um die relative Wichtigkeit der drei einzelnen Komponenten einzustellen. Um den Ausdruck weiter zu vereinfachen, setzt man $\alpha = \beta = \gamma = 1$ und $C_3 = C_2/2$. Dadurch erhält man eine spezielle Form des SSIM-Index

$$\mathrm{SSIM}(I_a, I_b) = \frac{(2\mu_a\mu_b + C_1)(2\sigma_{ab} + C_2)}{(\mu_a^2 + \mu_b^2 + C_1)(\sigma_a^2 + \sigma_b^2 + C_2)} \tag{5.15}$$

Diese letzte Form wurde hier in Kontext des "pair matching" dazu verwendet, um zwei Bildausschnitte unterschiedlicher Bilder miteinander zu vergleichen.

Ortsfrequenz basierte Ähnlichkeit

Im Gegensatz zu einem Ähnlichkeitsmaß, das im Ortsbereich arbeitet, wurde im Rahmen der hier durchgeführten Untersuchungen auch ein Maß verwendet, welches Ortfrequenzeigenschaften der jeweiligen Bildausschnitte berücksichtigt. Der Grundgedanke hierbei ist, dass zwei ähnliche Bilder oder Bildausschnitte ein sehr ähnliches

Betragsspektrum haben müssten, selbst wenn sie zueinander verschoben sind, welches sich ja in der Phasenlage niederschlagen würde. Der Vorteil eines geeigneten Maßes basierend auf dem Betragsspektrum wäre also eine Translationsinvarianz.
Als einfaches Ähnlichkeitsmaß zweier Betragsspektren, wurde hier der Korrelationskoeffizient verwendet. Zunächst wird jeder der miteinander zu vergleichenden Bildausschnitte in den Ortsfrequenzbereich transformiert, hierzu verwendet man die Diskrete Fourier Transformation (DFT).

$$\hat{I}(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x,y) e^{-2\pi i \frac{xu}{M}} e^{-2\pi i \frac{yv}{N}} \tag{5.16}$$

in vorliegenden Fall wird also ein Bildausschnitt $I(x,y)$ in den Ortsfrequenzbereich $\hat{I}(u,v)$ transformiert. $i = \sqrt{-1}$ ist hier die imaginäre Zahl, e die Eulersche Zahl.

Der Korrelationskoeffizient der Betragsspektren $|\hat{I}_a|$ und $|\hat{I}_b|$ der Bildausschnitte I_a und I_b wird dann als Ähnlichkeitsmaß $FTSIM$ verwendet:

$$\text{FTSIM}(I_a, I_b) = \frac{\sum_{u=0}^{K-1} \sum_{v=0}^{L-1} |\hat{I}_a(u,v)||\hat{I}_b(u,v)|}{\sqrt{\sum_{u=0}^{K-1} \sum_{v=0}^{L-1} \hat{I}_a(u,v)^2} \cdot \sqrt{\sum_{u=0}^{K-1} \sum_{v=0}^{L-1} \hat{I}_b(u,v)^2}} \tag{5.17}$$

5.4.2 Zusammenführung der einzelnen Maße zu einer gewichteten Summe

Nachdem nun einige Ähnlichkeitmaße eingeführt worden sind, wird nun eine geeignete Methode vorgestellt, diese zusammenzuführen. Dabei werden zwei Aspekte berücksichtigt: Zum einen die örtliche Zusammenführung der Ähnlichkeiten, die auf den unterschiedlichen Bildausschnitten berechnet wurden, zum anderen die Zusammenführung der verschiedenartigen Maße zu einem Gesamtmaß.

Für die vorliegende Anwendung des Vergleichs von Bildpaaren ist es logisch, dass nicht jeder Bildausschnitt die gleiche Aussagekraft hat. Viele Ausschnitte zeigen den Hintergrund, weitere beziehen sich eher auf homogene Bildstrukturen (zum Beispiel eine Hautfläche im Gesicht). Andere hingegen zeigen charakteristische Elemente des zu vergleichenden Gesichts wie zum Beispiel Augen, Augenbrauen, Nase oder Mund. Um dieses zu modellieren, werden die einzelnen Bildregionen durch eine gewichtete Summe zusammengeführt, das Gewicht einer jeden Region beschreibt die Bedeutung des jeweiligen Ausschnitts für eine Gesamtaussage.

Wenn davon ausgegangen wird, dass die vorliegenden Bilder I_a und I_b in $K \times L$ Ausschnitte zerlegt werden, und für jeden Ausschnitt $I_{a,k,l}$ im Bild I_a J verschiedene Ähnlichkeitmaße $s_j(I_{a,k,l}, I_{b,k,l})$ zum korrespondierenden Ausschnitt $I_{b,k,l}$ im Bild I_b berechnet wurden, kann eine gewichtete Summe $S(I_a, I_b)$ wie folgt berechnet werden:

$$S(I_a, I_b) = \sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{l=1}^{L} w_{j,k,l} s_j(I_{a,k,l}, I_{b,k,l}) \tag{5.18}$$

Die eigentliche Aufgabe in der Zusammenführung der verschiedenen Bildregionen und Ähnlichkeitsmaße liegt jetzt darin, die verschiedenen Gewichte $w_{j,k,l}$ zu ermitteln. Dieses soll anhand der vorhandenen Trainingsbeispiele der Bildpaare gleichen oder ungleichen Inhaltes geschehen. Wenn man alle Bildpaare der Klasse "gleiche Person in beiden Bildern" mit dem Referenzwert "1" versieht (1 entspricht der maximalen Ähnlichkeit) und alle Bildpaare der Klasse "unterschiedliche Personen in den Bildern" mit dem Referenzwert "-1" hat man alle nötigen Daten, um die Gewichte zu lernen. In einer einfachen Linearkombination, wie Gleichung 5.18 eine ist, bietet sich zunächst eine multivariate lineare Regression an, um die Gewichte zu bestimmen.
Eine weitere Möglichkeit besteht darin, die Zusammenführung der einzelnen Ähnlichkeitsfunktionen mittels eines neuronalen Netzes zu modellieren. Wenn man von einem neuronalen Netz mit $J \times K \times L$ Input-Neuronen und 1 Output-Neuron ausgeht, hat man eine Formulierung, die der gewichteten Summe sehr nahe kommt, der einzige Unterschied besteht in der Aktivierungsfunktion $\phi(x)$, typischerweise eine Sigmoid-Funktion. Eine Modellierung mit einem neuronalen Netz, hier das einschichtige Perzeptron, würde also wie folgt lauten, wenn die Aktivierungsfunktionen des Output-Neurons ϕ ist:

$$S(I_a, I_b) = \phi\left(\sum_{j=1}^{J} \sum_{k=1}^{K} \sum_{l=1}^{L} w_{j,k,l} s_j(I_{a,k,l}, I_{b,k,l})\right) \tag{5.19}$$

Da die Formulierung mit einem Neuronalen Netz (Gleichung 5.19) sehr ähnlich zu der ursprünglichen Idee (Gleichung 5.18) ist, gewichtete Summen zu bilden, bietet es sich an, auch andere Netzarchitekturen auszutesten, insbesondere, ob das Hinzufügen von Hidden-Schichten einen Performanzgewinn erzeugt oder nicht.

5.5 Ergebnisse

5.5.1 Kenngrößen der Evaluierung

Alle Ergebnisse werden nach dem vorgegebenen Benchmarking Schema erzeugt, das mit der Datenbank verknüpft ist [HRBLM07]. Die 10 Validierungssets mit jeweils 300 Bildpaaren als positive und 300 Bildpaaren als negative Trainingsbeispiele werden in 10 Evaluierungsrunden so aufgeteilt, dass jeweils 9 zum Training verwendet werden, das übrig bleibende Datenset wird zum Testen verwendet. Es ergeben sich also insgesamt 10 Validierungsrunden, basierend auf jeweils 5400 Trainingsbildpaaren und 600 Testdaten, die unabhängig voneinander trainiert und ausgewertet wurden. Die hier aufgeführten Ergebnisse beziehen sich auf die Klassifikation auf den bereits durch

"Funneling" ausgerichteten Daten, wenn es nicht explizit anders angegeben ist. Alle zu vergleichenden Bilder wurden auf 128×128 Pixel skaliert, und in 7×7 sich um 16 Pixel überlappende Bildausschnitte der Größe 32×32 Pixel zerlegt. Für jeden Bildausschnitt wurde auf die beschriebene Art und Weise eine Ähnlichkeitsaussage in Form eines skalaren Wertes pro zugrunde liegendem Ähnlichkeitsmaß erzeugt ("probabilistic symmetric χ^2" für die Histogramm-basierten Ähnlichkeitsaussagen, SSIM oder FTSIM).

Als Kenngrößen der Gesamtevaluierung wird die mittlere Klassifikationsleistung μ (Anzahl richtig Klassifizierter durch Gesamtanzahl, engl. "mean accuracy"), sowie die Standardfehler des Mittelwertes S_E der Ergebnisse aus den 10 Validierungsrunden angegeben. Dabei ist die mittlere Klassifikationsleistung

$$\mu = \frac{1}{10} \sum_{i=1}^{10} p_i \tag{5.20}$$

wobei p_i der Prozentsatz der richtig klassifizierten Proben des i-ten Validierungsset ist. Dann ist der Standardfehler des Mittelwertes

$$S_E = \frac{\sigma}{\sqrt{10}} \tag{5.21}$$

mit der Standardabweichung

$$\sigma = \sqrt{\frac{1}{9} \sum_{i=1}^{10} (p_i - \mu)^2} \tag{5.22}$$

5.5.2 Vergleich der einzelnen Ähnlichkeitsmaße

Um aussagekräftige Ergebnisse zu produzieren, werden zunächst alle Ähnlichkeitsmaße, je nach urspünglichem Merkmal, aus dem sie abgeleitet wurden, einzeln betrachtet. Dieses wird so umgesetzt, indem in den Formeln 5.18 und 5.19 die Anzahl der verschiedenen Ähnlichkeitsmaße auf $J = 1$ gesetzt wird. Es wird also für jedes einzelne Ähnlichkeitsmaß eine örtliche Zusammenführung über alle Bildausschnitte erzeugt, anschließend kann die Aussagekraft der verschiedenen Maße miteinander verglichen werden.
Hierbei wurden zunächst alle Gewichte durch das Lernen mit einem einfachen einschichtigen Perzeptron ermittelt. In Tabelle 5.1 zeigt sich, dass die lokalen Binärmerkmale die besten Ergebnisse erzeugen, hier also die höchste Aussagekraft enthalten. Die mittleren Klassifikationsleistungen, basierend auf den anderen Merkmalen, sind ebenfalls deutlich über der Ratewahrscheinlichkeit (50%) und könnten somit bei einer kombinierten Lösung einen Beitrag leisten.

Feature	μ	S_E
LBP3	0.7108	± 0.0048
LBP5	0.**7131**	± 0.0038
HoG	0.6701	± 0.0057
$\text{TPLBG}_{\alpha=2}$	0.6980	± 0.0066
$\text{TPLBG}_{\alpha=3}$	0.6924	± 0.0046
$\text{TPLBG}_{\alpha=4}$	0.6550	± 0.0061
SSIM	0.6703	± 0.0046
FTSIM	0.6510	± 0.0078

Tabelle 5.1: Mittlere Klassifikationsleistung μ und Standardfehler S_E für die örtliche Zusammenführung der Ähnlichkeitsmaße der verschiedenen Bildausschnitte, je nach zugrunde liegendem Merkmal.

5.5.3 Zusammenführung der verschiedenen Maße

Nachdem die Merkmale einzeln betrachtet wurden, werden nun geeignete Fusionsstrategien, die die verschiedenen Maße miteinander verknüpfen, untersucht. Wie bereits erläutert, wird zunächst eine Zusammenführung über eine gewichtete Summe verwendet, diesmal allerdings nicht nur die örtliche Zusammenführung, sondern auch die Zusammenführung der verschiedenartigen Ähnlichkeitsmaße, basierend auf den unterschiedlichen Merkmalen. Die Bestimmung der Gewichte geschieht zum einen durch multivariate lineare Regression, zum anderen durch ein- oder mehrschichtige Neuronale Netze. Die Ergebnisse zeigen, dass das einschichtige Neuronale Netz im Vergleich am besten abschneidet (siehe Tabelle 5.2).

Methode	μ	S_E
Lineare Regression	0.7213	0.0059
Neuronales Netz (ohne Hidden Schicht)	**0.7434**	0.0052
Neuronales Netz (3 Neuronen in Hidden Schicht)	0.7339	0.0050
Neuronales Netz (5 Neuronen in Hidden Schicht)	0.7404	0.0037
Neuronales Netz (6 Neuronen in Hidden Schicht)	0.7357	0.0039
Neuronales Netz (7 Neuronen in Hidden Schicht)	0.7319	0.0052
Neuronales Netz (10 Neuronen in Hidden Schicht)	0.7221	0.0045
Neuronales Netz (15 Neuronen in Hidden Schicht)	0.7267	0.0034

Tabelle 5.2: Mittlere Klassifikationsleistung bei Verwendung von linearer Regression und verschiedenen Architekturen Neuronaler Netze, um eine gewichtete Zusammenführung der unterschiedlichen Ähnlichkeitsmaße in den verschiedenen Bildausschnitten zu erzielen.

Beste Ergebnisse konnten hier mit einem Neuronalen Netz ohne Hidden Schicht erzielt werden, das die Eingabewerte direkt auf das Output-Neuron abbildet (einschichtiges Perzeptron). Es zeigt sich zwar, dass mehrschichtige Netze deutlich höhere Reklassifikationsraten erreichen, allerdings sinkt dafür die Generalisierungfähigkeit, also die Leistung bei der Klassifikation.

5.5.4 Einfluss des Ausrichtungsschrittes

Abschließend wird nun der Einfluss des Ausrichtungsschrittes untersucht. Hierfür werden die Ergebnisse mit und ohne Verwendung des Ausrichtungsschrittes "Funneling" verglichen. Hier wird jeweils nur noch die Gesamterkennungsrate nach der gewichteten Zusammenführung angegeben (siehe Tabelle 5.3).

Methode	μ	S_E
mit Funneling	**0.7434**	0.0052
ohne Funneling	0.7241	0.0058

Tabelle 5.3: Mittlere Klassifikationsleistung in Abhängigkeit des Ausrichtungsschrittes "Funneling".

Es zeigt sich, dass der Ausrichtungsschritt einen spürbaren Effekt auf die Gesamtklassifikationsleistung hat. Da ja im vorliegenden Ansatz davon ausgegangen wird, dass die durch Ähnlichkeitsmaße bewerteten Regionen der verschiedenen Bilder zumindest grob örtlich miteinander korrespondieren, ist es klar, dass sich eine Ausrichtung positiv auswirken muss.

5.6 Vergleich zu anderen Ansätzen

"Labeled Faces in the Wild" ist ein Datenset, welches für Benchmarkingzwecke konzipiert wurde, es ist also ein Datensatz zum Vergleich der Leistungsfähigkeit verschiedener Verfahren. Es wurden bereits die Ergebnisse mehrerer verschiedener Ansätze auf dieser Datenbank publiziert. Somit ist es möglich einen direkten Vergleich zwischen den verschiedenen "state of the art" Methoden durchzuführen, siehe Tabelle 5.4. Der im Rahmen dieser Arbeit entwickelte Algorithmus ist fettgedruckt.

Ein guter Anhaltspunkt zur Bewertung der Schwierigkeit der Klassifikationsaufgabe ist es, sich die Klassifikationsleistung eines Standardverfahrens anzuschauen, hier der "Eigenfaces" Methode [TP91a], bei der die Trainingsdaten jeweils zur Berechnung der Hauptachsen verwendet wurden. Anschließend wurde für jedes einzelne

Methode	μ	S_E
Gewichtete Summen von Ähnlichkeitsmaßen	**0.7434**	± 0.0052
3x3 Multi-Region Histograms[SL09]	0.7295	± 0.0055
Eigenfaces [TP91a]	0.6002	± 0.0079
joint alignment [HJLM07]	0.7393	± 0.0049
MERL [NJ07]	0.7052	± 0.0060
MERL+joint alignment [NJ07]	**0.7618**	± 0.0058
Hybrid descriptor-based [WHT08]	**0.7847***	± 0.0051

Tabelle 5.4: Vergleich zu anderen Ansätzen: mittlere Klassifikationsleistung μ und Standardabweichung S_E der verschiedenen Verfahren. Im mit * gekennzeichneten Ansatz wurden zusätzlich künstliche Trainingsbeispiele erzeugt, das schränkt die Vergleichbarkeit ein.

Bild eines Bildpaars die Projektion auf diese Hauptachsen berechnet und als parametrische Beschreibung verwendet. Mittels eines Abstandsklassifikators (hier Mahalanobisdistanz), wurde dann bewertet, ob es sich um Bildpaare mit gleichen oder unterschiedliches Personen handelt. Die mittlere Klassifikationsleistung von ca. 60% für diesen Ansatz zeigt, dass die hier zu lösende Aufgabe durchaus anspruchsvoll ist. Bei genauerer Betrachtung der Daten kann erklärt werden, warum der "Eigenfaces"-Ansatz nicht besonders gut funktioniert. Die zu vergleichenden Bilder weisen eine hohe Varianz bei den Aspekten Pose, Verschiebung, Skalierung, Hintergrund und Beleuchtung auf. Es ist bekannt dass der Eigenface Ansatz ungeeignet ist, solch variante Daten zu bearbeiten, da diese Varianzen sich zwar auf die Berechnung der Hauptkomponenten auswirken, allerdings nicht charakteristisch zur Unterscheidung von Individuen sind.

Es gilt also Verfahren zu finden, die robuster im Umgang mit diesen varianten Einflüssen sind. Eine Methode, die in den Mitsubishi Electric Research Laboratories entwickelt wurde (MERL, siehe [NJ07]), basiert darauf, mit vielen Detektoren einzelne Gesichtsmerkmale zu finden. Diese gefundenen Merkmale dienen dazu, den Gesichtsausschnitt festzulegen, der anschließend zu Klassifikation verwendet werden soll. Es handelt sich also um einen Ausrichtungsschritt mit dem Ziel Rotation, Translation und Skalierung in der Bildebene zu kompensieren. Nachdem dieser Ausschnitt festgelegt wurde, werden nun einzelne Subregionen des Gesichts durch Extraktion von Haar ähnlichen Merkmalen beschrieben, der daraus entstehende Merkmalsvektor wird zur Klassifikation herangezogen. Mit diesem Verfahren konnte eine Klassifikationsleistung von ca. 70% erreicht werden, also eine deutliche Steigerung gegenüber den Eigenfaces.

Ein weiterer interessanter Ansatz sind die "Multi-Region Histograms“ [SL09], mit denen eine Erkennungsleistung von ca. 73% erreicht werden konnte. Dabei werden die Häufigkeiten quantisierter, blockweise extrahierter DCT-Merkmalsvektoren verwendet, um die mittleren Histogramme der Quantisierungsstufen auf fest definierten Regionen zu berechnen. Diese mittleren Histogramme werden dann mit denen des zweiten Bildes verglichen. Dieser Ansatz ist ähnlich dem hier vorgestellten Algorithmus, der entscheidende Unterschied, neben den anderen Merkmalen, ist der, dass Bildregionen nicht durch einen Lernprozess unterschiedlich gewichtet werden, und dass keine Abbildung auf skalare Werte für die einzelnen Regionen stattfindet. Der Ansatz der Multi-Region-Histograms ist verwandt mit HMM-Ansätzen die auf blockweise extrahierten Merkmalen arbeiten, der Unterschied ist der, dass kein automatisches Alignment wie beim HMM vorgenommen wird, also welcher Block auf welchen Zustand abgebildet wird, sondern, dass ein fest vorgegebenes Schema der regionalen Zuordnung verwendet wird.

Die bisher besten und erfolgreichsten Ergebnisse konnten durch Kombination verschiedener Ansätze erreicht werden (siehe [NJ07] und [WHT08]). Typischerweise wird hierbei ein Ausrichtungssansatz mit einem oder mehreren Merkmalsextraktions- und Klassifikationsansätzen verknüpft.
Einer dieser Ansätze [WHT08] verwendet ebenfalls diverse lokale Binärmerkmale (LBP, TPLBP), treibt die Merkmalsextraktion allerdings noch weiter und verwendet auch Binärmerkmale unter der Verwendung von vier Bildausschnitten (Four Patch Local Binary Patterns). Allerdings werden die Merkmalsvektoren, die auf den zu vergleichenden Bildern extrahiert worden sind, nicht etwa durch regionale Histogrammbildung oder Berechnung von Ähnlichkeitsmaßen komprimiert, sondern in voller Dimensionalität für jedes Merkmal einzeln zur Klassifikation verwendet. Nun wird für jedes Bild eines Bildpaares ein eigener Klassifikator erstellt, indem das Problem als "Lernen anhand eines Beispiels“ interpretiert wird. Für jedes Bild aus einem zu vergleichenden Bildpaar wird dafür ein solcher Klassifikator erstellt, indem die Bilder eines der 9 Trainingssets nicht als Bildpaare interpretiert werden, sondern das Wissen, dass die Sets personendisjunkt sind, ausgenutzt wird, alle 1200 in dem Set vorkommende Einzelbilder als Negativbeispiele zu verwenden. Für ein gegebenes Bildpaar mit einem Bild I_a und I_b bedeutet dies, dass ein Klassifikator durch Diskriminanzanalyse für Bild I_a mit 1200 Negativbeispielen und einem Positivbeispiel (I_a selbst) trainiert wird und dann der Wert dieses Klassifikators für I_b ermittelt wird. Anschließend wird ein zweiter Klassifikator für I_b trainiert und der Wert von I_a ermittelt. Diese beiden Werte werden gemittelt.
Dieser Trainingsprozess wird für jeden Merkmalstyp einzeln wiederholt, somit entsteht für jeden Merkmalstyp ein mittlerer Wert pro Bildpaar. Über alle Merkmalstypen ergibt sich somit ein Vektor solcher mittleren Werte. Dieser Vektor wird nun noch zusätzlich mit diversen direkten Distanzmaßen zwischen den Merkmalsvektoren

der Bilder I_a und I_b angereichert. Die Unterscheidung, ob es nun ein gleiches Gesicht ist oder nicht, wird durch einen weiteren Klassifikator (hier eine Support Vektor Machine) erzeugt, der anhand dieses Vektors entscheidet. Dieser Entscheidungsklassifikator wird mit den resultierenden Vektoren trainiert, die durch Ausführung genau des gleichen Prozesses auf den Bildpaaren der verbleibenden 8 Validierungssets ermittelt werden.
Wie leicht zu erkennen ist, wurde hier ein erheblicher Aufwand in den Trainingsprozess gesteckt, um gute Erkennungsleistungen zu erzeugen. Die getroffenen Maßnahmen weichen zwar formal nicht vom Benchmarkingschema ab, dennoch wurde das Wissen ausgenutzt, dass die Trainingssets personendisjunkt sind, um künstlich zusätzliche Negativbeispiele zu erzeugen, dieses beschränkt leider die Vergleichbarkeit der Ergebnisse zu den anderen Verfahren.

Insgesamt schneidet der im Rahmen dieser Arbeit entwickelte Ansatz recht gut ab, siehe Tabelle 5.4, und liegt mit 74.34% Erkennungsrate, trotz seines einfachen Grundprinzips und einer dadurch effektiven Berechnungsmöglichkeit, im oberen Mittelfeld der bisher veröffentlichten Ergebnisse auf dem "Labeled Faces in the Wild" Datensatz. Da dieser Datensatz noch recht jung ist, er wurde erstmals 2007 der Öffentlichkeit zur Verfügung gestellt, ist zu erwarten, dass in Zukunft noch viele weitere Ansätze auf diesem Datenset weiterentwickelt und ausgetestet werden und somit die Erkenntnisse im Bereich des "unconstrained pair matching" weiter wachsen werden.

5.7 Zusammenfassung und Diskussion

Durch einen einfachen, aber doch effektiven Ansatz konnte hier eine konkurrenzfähige Erkennungsleistung von 74.34% erreicht werden. Zugrunde liegen hier diverse Merkmalsextraktionsverfahren, die extrahierten Merkmale werden durch ihre Häufigkeiten in den Bildregionen in Histogrammen hinterlegt. Die jeweiligen Histogramme der korrespondierenden Bildregionen zweier Bilder werden durch ein Ähnlichkeitsmaß bewertet, somit auf ein Skalar abgebildet. Damit wird eine enorme Dimensionsreduktion erreicht. Eine Klassifikation, ob es sich um zwei Bilder gleichen oder unterschiedlichen Inhalts handelt, wird anhand einer gewichteten Summe der Ähnlichkeiten der einzelnen Bildregionen vorgenommen.

Dieser Ansatz ist sehr einfach zu erweitern, oder auf andere Merkmalstypen umzustellen. Das Erlernen der Gewichte der einzelnen Bildregionen kann durchaus mit anderen Verfahren maschinellen Lernens durchgeführt werden, hier ist sicherlich noch Optimierungsspielraum. Interessant wäre es auch, ähnlich einer Auflösungspyramide, verschieden große Blöcke zur Merkmalsextraktion zu verwenden oder wie

beim Ansatz der "Multi-Region Histograms" mehrere Blöcke durch Mittelung zusammenzufassen, um einfache abgeleitete Merkmalshistogramme zu erzeugen, die als zusätzliche Klassifikationsgrundlage dienen können. Weitere Verbesserungsmöglichkeiten bestehen darin, diese Methode mit einer der anderen erfolgreichen Methoden zu fusionieren, um eine insgesamt noch höhere Klassifikationsleistung zu erzeugen. Dieses Vorgehen hat sich bewährt, so sind die bis dato besten Ergebnisse auf diesem Datenset durch Fusionen mehrerer Ansätze zustande gekommen.

Zusätzlich hat es sich gezeigt, dass geschickte Strategien zur künstlichen Vermehrung der Trainingsdaten in einer gesteigerten Erkennungsleistung resultieren können. Allerdings erschweren solche Maßnahmen immer die Vergleichbarkeit zu anderen Ansätzen und sollten deshalb immer nur als zusätzliche Evaluationen durchgeführt werden.

6

Erkennung von Gesichtern in 3D- und Tiefendaten

In diesem Kapitel wird ein neuartiges Verfahren zur Erkennung von Gesichtern in 3D- und Tiefendaten vorgestellt. Zunächst wird hier aber erst mal eine grundsätzliche Betrachtung der Vorteile einer 3D- oder Tiefendatenerfassung im Hinblick auf die gegeben Aufgabenstellung vorgenommen. Dabei sind 3D-Daten solche, die für Punkte im 3D ein- oder mehrkanälige Meßgrößen liefern (Texturkanäle), während Tiefendaten nur aus den räumlichen Informationen selbst bestehen.

Zwei der immer wieder auftretenden Problematiken bei der Gesichtserkennung sind variante Beleuchtung und die Kopfpose oder Blickrichtung der zu bewertenden Person. Diese beiden Probleme können durch Verwendung von Tiefeninformationen, also zusätzlichem geometrischen Wissen im Vergleich zur 2D-Bildverarbeitung, kompensiert werden. Tiefeninformation ist beleuchtungsinvariant, zusätzlich können dadurch, dass explizite Informationen zur Geometrie der beobachteten Objekte vorliegen, physikalische Beleuchtungsmodelle herangezogen werden, um die Helligkeiten in eventuell vorhandenen Texturkanälen zu normieren. In manchen Anwendungsfällen auftretende Verdeckung kann theoretisch einfacher bearbeitet werden, da Tiefeninformation eine einfache Segmentierungsgrundlage bietet. Durch geometrische 3D-Transformationen können in einem Ausrichtungsschritt variable Kopfposen auf eine vorgegebene Blickrichtung normiert werden. Die Blickrichtung oder Kopfpose kann also umgerechnet werden, um eine blickrichtungsnormierte Ansicht zu generieren. Dieses vereinfacht die anschließende Erkennung, die dann mit den klassischen Mustererkennungsverfahren durchgeführt werden kann.

Im Vergleich zu klassischen Methoden auf 2D-Texturbildern stellen sich allerdings für viele Standardverarbeitungsschritte, wie zum Beispiel die Gesichtsdetektion, neue Herausforderungen, insbesondere, wenn nichtfrontale Posen zugelassen werden. Da durch die höhere Dimensionalität mehrere Freiheitsgrade in Bezug auf

Rotation und Translation hinzugekommen sind, ist eine Detektionsaufgabe theoretisch deutlich schwieriger. In der Praxis ist es allerdings so, dass die heute zur Verfügung stehenden Aufnahmesysteme, die in diesem Kontext verwendet werden, oft nur eine Nahaufnahme der zu untersuchenden 3D Szene ermöglichen, die aufgrund messtechnischer Schwierigkeiten auch nur einen limitierten Tiefenbereich hat. Dadurch entfällt meist das Problem der Vordergrundsegmentierung, welches die Detektionsaufgabe auch schon wieder vereinfacht. Im Zusammenhang mit heutiger Messtechnik wird übrigens oft auch von 2.5D-Daten gesprochen, da es sich nicht um wirkliche 3D-Informationen handelt, sondern pro Bildpunkt nur ein Tiefenwert ermittelt werden kann. Es wird also stets nur die nächste dem Aufnahmesensor zugewandte Objektoberfläche erfasst.

Im Folgenden wird eine neuartige Verarbeitungskette zum Erkennen von Personen in 3D- und Tiefendaten vorgestellt (siehe [SR08]). Der Gesamtablauf besteht dabei aus mehreren Stufen die im Einzelnen vorgestellt werden (siehe Abbildung 6.1).

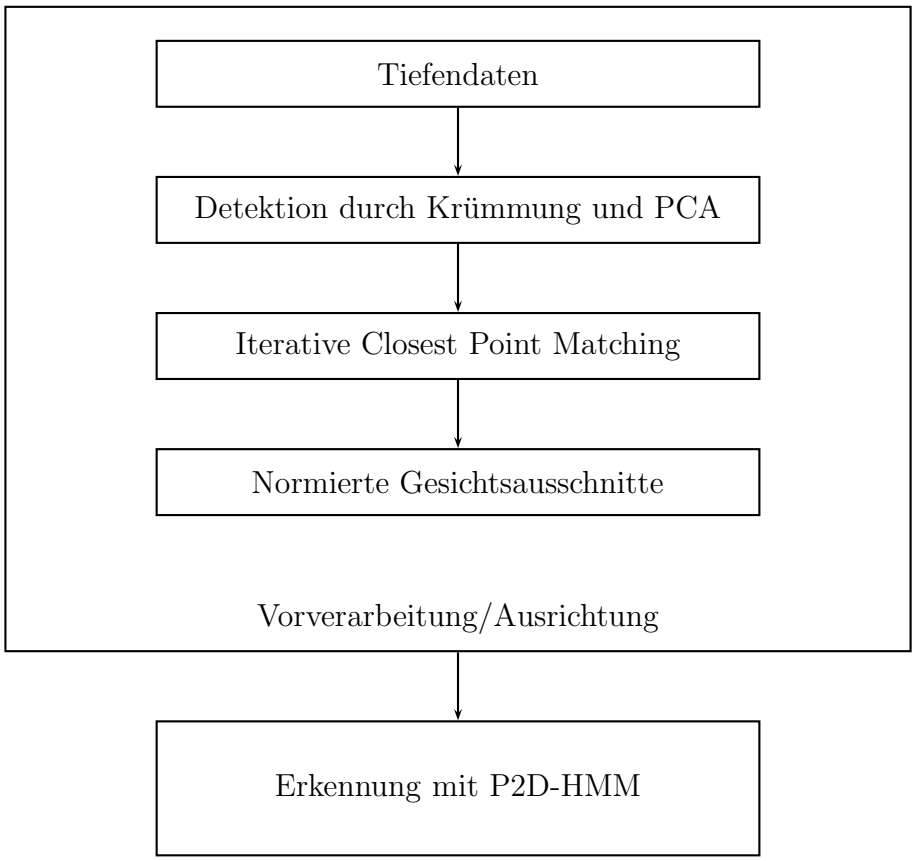


Abbildung 6.1: Verarbeitungskette zur Gesichtserkennung in Tiefendaten.

6.1 Verwendete Datenbasis

Für die im Folgenden aufgezeigten Untersuchungen wurde die GAVAB Datenbank (siehe [MS04]) verwendet. Diese für Forschungszwecke frei erhältliche Datenbank enthält ausschließlich Tiefendaten, also keine Texturinformationen, von Gesichtern, die mit verschiedenen Kopfposen und Gesichtsausdrücken aufgenommen wurden. In dieser Datenbank sind 61 verschiedene Individuen aufgenommen, für jede Person sind dabei sieben verschiedene Datensets vorhanden. In Abbildung 6.2 werden die sieben verschiedenen Datensets für eine Person dargestellt, dabei wird die Tiefeninformation als Intensität dargestellt.

Abbildung 6.2: Die gleiche Person als Tiefenbild (interpoliert und geglättet) in den sieben verschiedenen Datensets, von links nach rechts: *neutral1*, *neutral2*, *smile*, *laugh*, *random*, *look up*, *look down*.

Als geeignete Vorverarbeitung der Tiefendaten, die räumlich diskret vorliegen, werden die Daten zunächst interpoliert, so dass für jeden Pixel innerhalb der äußeren Grenzen des Gesichtes ein Tiefenwert vorliegt. Im vorliegenden Fall wird dieses so realisiert, dass ein Filter für alle Pixel an denen kein Messwert vorliegt, den Mittelwert aus einer Umgebung mit fest definierter Größe berechnet. Vorher wird der Wertebereich der Tiefendaten durch lineare Skalierung auf [0..255] normiert.

- Für jeden Pixel (x_i,y_j) in $I(x, y)$:
  - Wenn $I(x_i, y_j) = 0$, also wenn kein Meßwert an der Stelle (x_i,y_j), dann

$$I(x_i, y_j) \leftarrow \frac{\sum_{k=-d}^{d} \sum_{l=-d}^{d} I(x_i + k, y_j + l)}{\sum_{k=-d}^{d} \sum_{l=-d}^{d} \mathbf{sgn}(I(x_i + k, y_j + l))} \tag{6.1}$$

  - Wenn $I(x_i, y_j) \neq 0$, bleibt dieser Wert erhalten.

Die Summe über die Signumfunktion $\mathbf{sgn}(I(x_i + k, y_j + l))$ im Nenner des obigen Ausdrucks ist dabei dafür zuständig, die Anzahl der Messwerte > 0 zu zählen. Durch diese selektive Filterung werden kleinere Löcher im Tiefenscan geschlossen, und ein kontinuierlicher Verlauf des gesamten Tiefenbildes gewährleistet. Hier ist anzumerken, dass dieser Vorverarbeitungsschritt nur bedingt geeignet ist Tiefendaten im allgemeinen aufzubereiten. Da allerdings im Anwendungsfall "Gesicht" keine sprunghaften Änderungen in der Tiefeninformation vorkommen, ist er hier sinnvoll.

6.2 Gesichtsdetektion in Tiefendaten

Eine Gesichtsdetektion auf 3D-Daten kann unter gewissen Voraussetzungen genau so vorgenommen werden, wie eine auf 2D-Texturbildern. Dieses bietet sich dann an, wenn frontale Gesichtposen aufgenommen werden, und wenn ein Texturkanal zur Verfügung steht. Wenn allerdings nur Tiefendaten, und diese auch noch in verschiedene Kopfposen, vorliegen, muss ein anderer geeigneter Ansatz gefunden werden.

Ein interessantes Merkmal für die Untersuchung von Tiefendaten ist die Krümmung der beobachteten Oberfläche, siehe [BJ86][CCS06]. Diese ist rotationsinvariant, und zwar nicht nur, was Rotationen in der Bildebene betrifft, sondern auch bezogen auf Rotationen in der Tiefe. Dieses Merkmal ist zwar nicht im eigentlichen Sinne skalierungsinvariant, doch zumindest was das Vorzeichen betrifft. Im Folgenden werden nun zwei Krümmungsmerkmale vorgestellt, die für Oberflächen im 3D berechnet werden können. Die beiden Krümmungsmaße sind die mittlere Krümmung H und die Gaußsche Krümmung K.

$$H = \frac{(1 + I_x^2)I_{yy} - 2I_x I_y I_{xy} - (1 + I_y^2)I_{xx}}{2(1 + I_x^2 + I_y^2)^{\frac{3}{2}}} \tag{6.2}$$

$$K = \frac{I_{xx}I_{yy} - I_{xy}^2}{(1 + I_x^2 + I_y^2)^2} \tag{6.3}$$

Um diese bestimmen zu können, werden die Richtungsableitungen des Tiefenwertes I_x und I_y durch Verwendung eines Gradientenoperators bestimmt. Im vorliegenden Fall wurden die Sobel-Operatoren S_x und S_y für die numerische Berechnung der horizontalen und vertikalen Richtungsableitung herangezogen:

$$I_x = S_x * I = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} * I \tag{6.4}$$

$$I_x = S_y * I = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & 2 & -1 \end{pmatrix} * I \tag{6.5}$$

Hier ist $*$ der Faltungsoperator. Für die Richtungsableitungen zweiter Ordnung werden die Gradientenoperatoren mehrfach hintereinander angewendet, so dass

$$I_{xx} = (I * S_x) * S_x, \tag{6.6}$$

$$I_{xy} = (I * S_x) * S_y \tag{6.7}$$

und

$$I_{yy} = (I * S_y) * S_y. \tag{6.8}$$

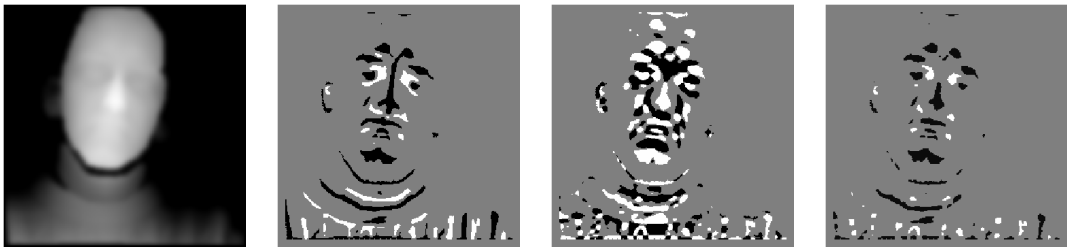


Abbildung 6.3: Tiefpassgefiltertes Tiefenbild (erstes Bild), mittlere Krümmung (zweites Bild) und Gaußsche Krümmung (drittes Bild). Negative Krümmungswerte sind schwarz dargestellt, positive weiss. Durch Kombination können konkave Regionen (schwarz) und konvexe Regionen (weiss) als Hypothesen für die Nasenspitze und Augenecken ermittelt werden (viertes Bild).

Durch Schwellwertbildung wird für jeden Bildpunkt bestimmt, ob die dort berechneten Krümmungen H und K nahe Null, positiv oder negativ sind (verwendete Schwellwerte: mittlere Krümmung: ± 0.05, Gaußsche Krümmung: ± 0.0005). Anhand des Vorzeichens der beiden Krümmungsmaße kann bestimmt werden, ob die Oberflächenkrümmung konkav oder konvex ist.
Die einzelnen Bildpunkte werden durch einen Segmentierungsalgorithmus (connected component analysis, z.B. [DST92]) zu Regionen gleicher Krümmungscharakteristik zusammengefasst, anschließend dienen die örtlichen Mittelwerte zusammenhängender Krümmungsregionen als beschreibende Koordinaten für die jeweilige Region. Da es bekannt ist, dass die Nasenspitze eine elliptisch konkave Region ($H < 0$, $K > 0$) und die Augenecken (Region zwischen Nasenbein und Augen) elliptisch konvexe Regionen ($H > 0$, $K > 0$) sind, können nun Hypothesen für diese Gesichtsmerkmale gefunden werden, siehe Abbildung 6.3. Um möglichst keine kleinen unbedeutenden Krümmungsregionen zu finden, kann das Tiefenbild zur Rauschunterdrückung vorher stark tiefpassgefiltert werden, ohne dass die allgemeine Krümmungscharakteristik verloren geht.

Nachdem die Koordinaten für konvexe und konkave Regionen gefunden wurden, werden nun alle möglichen Tupel bestehend aus einer Nasenspitzenhypothese (konkave Region) und zwei Augeneckenhypothesen (konvexe Regionen) gebildet. Ein solches Tupel wird im Folgenden als Gesichtshypothese bezeichnet. Die Anzahl der Gesichtshypothesen wird nun in einem ersten Schritt durch Expertenwissen über relative Distanzen und Positionierung von Augenecken und Nasenspitze ausgedünnt. Im Rahmen dieser Arbeit wird nur ein sehr kleiner Satz an Regeln verwendet, um eine Gesichtshypothese im ersten Schritt als gültig zu erklären. Dabei werden die Positionen der jeweiligen beiden Augeneckenhypothesen als $\vec{p}_{e,1/2} = (p_{e,1/2,x}, p_{e,1/2,y})^T$ und die Nasenhypothesen als $\vec{p}_n = (p_{n,x}, p_{n,y})^T$ bezeichnet. Diese Punktkoordinaten

beschreiben jeweils den örtlichen Mittelwert der jeweiligen Krümmungsregion.

Ein Tupel gilt als gültige Hypothese, wenn

- die Abstände der Hypothesen folgende relative Beziehungen erfüllen:

$$\begin{aligned} 2 \cdot d_{eukl}(\vec{p}_{e,1}, \vec{p}_{e,2}) &> \max(d_{eukl}(\vec{p}_{e,1}, \vec{p}_n), d_{eukl}(\vec{p}_{e,2}, \vec{p}_n)) \\ d_{eukl}(\vec{p}_{e,1}, \vec{p}_{e,2}) &< 2 \cdot \max(d_{eukl}(\vec{p}_{e,1}, \vec{p}_n), d_{eukl}(\vec{p}_{e,2}, \vec{p}_n)) \end{aligned} \tag{6.9}$$

- die Nase unter den Augen ist: $p_{n,y} < \min(p_{e,1,y}, p_{e,2,y})$
- das Tupel einzigartig ist (Kommutativität der Augenecken)
- Die Krümmungsregionen eine Mindestgröße haben.

Hier werden bewusst nur sehr grobe Regeln verwendet, so dass durchaus viele Hypothesen nach Anwendung dieser Regeln übrig bleiben. In einem zweiten Schritt wird nun jede einzelne Hypothese auf ihre Gültigkeit hin klassifiziert. Dafür werden Ausschnitte der jeweiligen Hypothesen aus den Tiefenbildern erzeugt. Jeder Ausschnitt wird auf eine feste Größe skaliert, hier auf 96×64 Pixel, und mit einem Klassifikator wird nun entschieden, ob es sich um eine gültige Gesichtshypothese handelt oder nicht.

Der hier verwendete Klassifikator basiert auf der Hauptachsentransformation und wird durch Beispiele erzeugt. Auf den Tiefenbildern des Trainingsdatensatzes werden Gesichtshypothesen, wie beschrieben, gebildet. Eine manuelle Auswahl selektiert die gültigen Hypothesen, die Ausschnitte, die diese beschreiben, dienen zum Training des Klassifikators. Einige Beispiele solcher Ausschnitte sind in Abbildung 6.4 zu sehen.

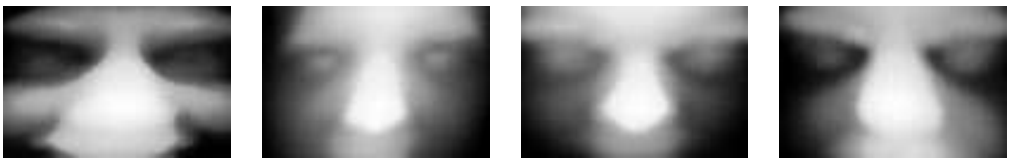


Abbildung 6.4: Beispiele für gültige Gesichtshypothesen aus den Trainingsdaten. Diese werden verwendet um den PCA basierten Klassifikator zu erzeugen.

Um die Hauptachsen der vorliegenden Daten zu ermitteln, wird nun eine PCA auf den Trainingsbildern ausgeführt. Bei N Trainingsbildern $\vec{x}_i$ mit $i = 1..N$, ist der erste Schritt die Berechnung des Mittelwertes $\overline{\vec{x}}$.

$$\overline{\vec{x}} = \frac{\sum_{i=1}^{N} \vec{x}_i}{N} \tag{6.10}$$

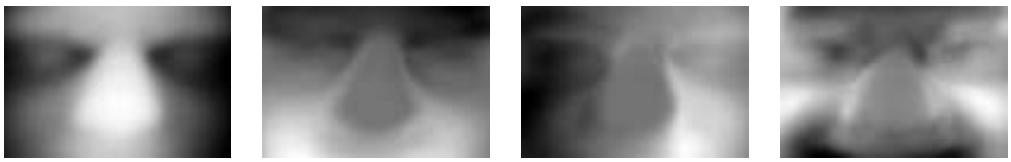


Abbildung 6.5: Mittelwert (links) und die ersten drei Hauptachsen, nach Rückprojektion in den Bildraum.

Anschließend wird die $N \times (96 \cdot 64)$ Matrix A der mittelwertfreien Daten generiert.

$$A = (\vec{a}_1 = \vec{x}_1 - \overline{\vec{x}}, .., \vec{a}_N = \vec{x}_N - \overline{\vec{x}}) \tag{6.11}$$

Die Kovarianzmatrix C ergibt sich dann zu

$$C = \frac{1}{N-1} A A^T \tag{6.12}$$

Die Matrix der Eigenvektoren $E = (\vec{e}_1, .., \vec{e}_k)$ der Kovarianzmatrix C wird berechnet und die Eigenvektoren werden absteigend nach Größe ihrer Eigenwerte λ_i sortiert. Die l Eigenvektoren mit den l größten Eigenwerten werden so gewählt, dass das Verhältnis der damit darstellbaren Varianz zur Gesamtvarianz 98% entspricht (siehe Gleichung. 6.13).

$$l : \frac{\sum_{i=1}^{l} \lambda_i}{\sum_{i=1}^{k} \lambda_i} \geq 0.98 \tag{6.13}$$

Für die gegebene Anwendung führt das zu $l = 50$ Eigenvektoren. Sie spannen einen linearen Unterraum auf, der gut geeignet ist, gültige Gesichtshypothesen zu beschreiben. Der Mittelwert und die drei bedeutendsten Eigenvektoren sind in Abbildung 6.5 aufgezeigt.

Jeder Ausschnitt $\vec{x}$ der gleichen Größe (96×64 Pixel) kann nun in diesen linearen Unterraum projiziert werden, dabei werden die Gewichte $\vec{w}$ berechnet.

$$\vec{w} = E_l^T (\vec{x} - \overline{\vec{x}}) \tag{6.14}$$

Die Matrix E_l enthält dabei nur die l Eigenvektoren $E = (\vec{e}_1, .., \vec{e}_l)$ mit den l größten Eigenwerten.

Für jeden Bildausschnitt einer Gesichtshypothese $\vec{x}_h$ wird eine Rückprojektion $\vec{x}_r$ erzeugt, indem sie durch Berechnung des Gewichtsvektors $\vec{w}$ auf die l Eigenvektoren projiziert wird, und anschließend zurücktransformiert wird in den originalen Tiefenbildraum (siehe Gleichung 6.15).

$$\vec{x}_r = \overline{\vec{x}} + \sum_{i=1}^{l} w_i \vec{e}_i = \overline{\vec{x}} + E_l \vec{w} \tag{6.15}$$

Wenn der Kreuzkorrelationskoeffizient cc der Hypothese $\vec{x}_h$ und der Rückprojektion $\vec{x}_r$ größer ist als ein Schwellwert T_{cc}, wird die Hypothese als gültige Gesichtsregion klassifiziert. Es hat sich gezeigt, dass ein Schwellwert von $T_{cc} = 0.98$ ein geeigneter Wert ist.

$$cc = \frac{\vec{x}_r \vec{x}_h}{|\vec{x}_r||\vec{x}_h|}. \tag{6.16}$$

Wenn mehr als eine Hypothese als gültig gewertet wird, wird diejenige mit dem höchsten Kreuzkorrelationkoeffizienten cc gewählt. Da die Gesichtshypothese ja durch die Nasenspitze und die Augenecken definiert ist, wird die Position dieser Gesichtmerkmale automatisch mit extrahiert. Einige Beispiele für so detektierte Gesichtsausschnitte sind in Abbildung 6.6 dargestellt.

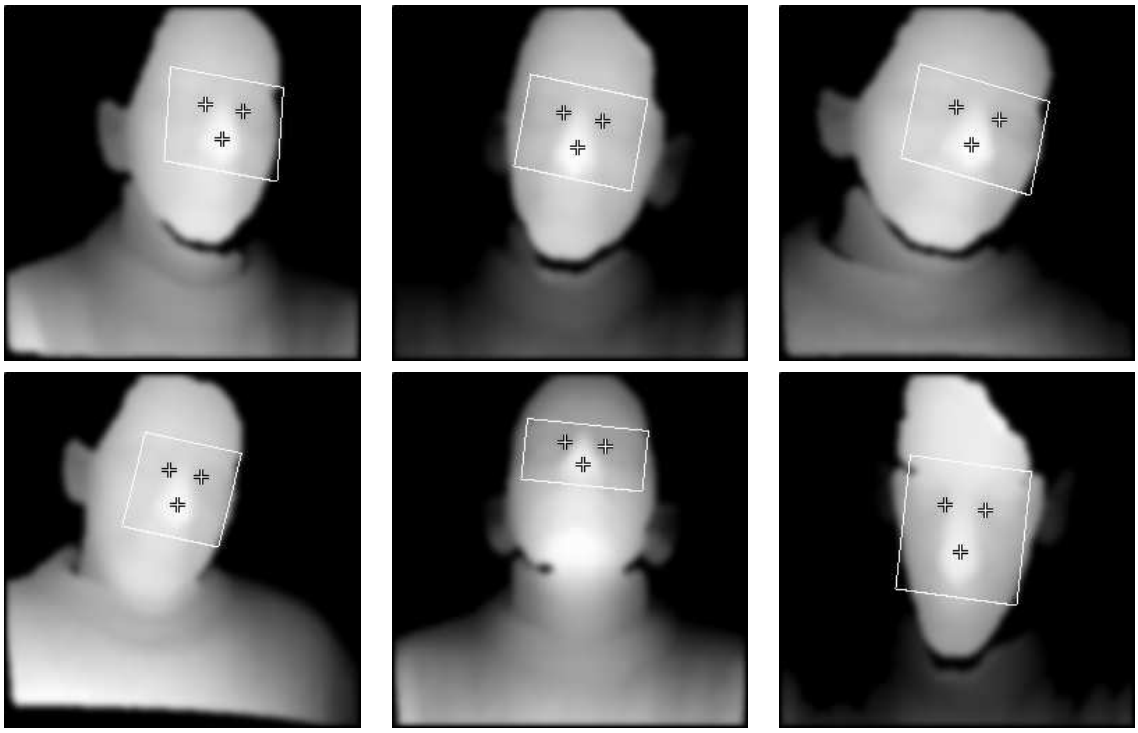


Abbildung 6.6: Ergebnisse der Detektion, Kreuze symbolisieren die Augenecken und Nasenspitze, die jeweiligen zur Klassifikation verwendeten Ausschnitte sind durch die eingezeichneten Bereiche dargestellt. Die folgenden Bilder zeigen die Ergebnisse auf den Datensets *neutral2*, *smile*, *laugh*, *random*, *look up* und *look down* (von oben links nach unten rechts).

Diese Detektion funktioniert sehr robust, ist allerdings noch sehr grob. Um nun eine erste grobe Ausrichtung des Gesichtes auf eine normierte Blickrichtung zu erlangen, wird nun eine 3D-Rotation ausgeführt, so dass die Verbindungslinie zwischen den Augenecken horizontal in der Bildebene liegt und die Nase nach vorne zeigt. Eine einfache Methode, dieses zu erreichen, ist, einmal eine manuelle 3D-Ausrichtung eines Beispielgesichtes vorzunehmen, und anschließend die Referenzkoordinaten für Augenecken und Nasenspitze dieses Beispiels zu ermitteln. Dadurch werden die Koordinaten $(\vec{p}_{e,1,ref}, \vec{p}_{e,2,ref}, \vec{p}_{n,ref})$ für die Augenecken und Nasenspitze der Wunschansicht ermittelt. Eine automatische Ausrichtung anhand der detektierten Koordinanten $(\vec{p}_{e,1}, \vec{p}_{e,2}, \vec{p}_n)$ kann dann dadurch vorgenommen werden, dass das Gleichungsystem

$$(\vec{p}_{e,1,ref}, \vec{p}_{e,2,ref}, \vec{p}_{n,ref}) = S \cdot R \cdot (\vec{p}_{e,1}, \vec{p}_{e,2}, \vec{p}_n) + \vec{t} \tag{6.17}$$

gelöst wird. Dabei ist R eine Rotationsmatrix mit drei unbekannten Rotationwinkeln und $S = s \cdot E$ eine Skalierungsmatrix mit dem Skalierungsfaktor s, E ist hier die Einheitsmatrix, $\vec{t}$ ist ein Verschiebungsvektor mit x, y und z-Komponente. Da der Wert des Bildes I an der Stelle $\vec{p}$ ja dem Tiefenwert entspricht, sind also für jeden Punkt $\vec{p}$ der x-,y- und z-Koordinatenwert vorhanden, somit besteht dieses Gleichungssystem aus 9 Gleichungen und ist damit überbestimmt. Eine Lösung über die Methode des kleinsten quadratischen Fehlers ist damit möglich, die Lösung dieses Gleichungssystems wird in Anhang B beschrieben. Da es sich bei den detektierten Augen- und Nasenhypothesen allerdings um die Mittelwerte von Regionen mit bestimmten Krümmungseigenschaften handelt, und nicht um exakte Punkte, ist diese Aussrichtung noch recht grob.

6.3 Ausrichtung auf normierte Blickrichtungen

Nachdem nun eine erste, grobe Aussrichtung des Gesichtausschnittes vorliegt, wird nun ein zweiter präziser Ausrichtungsschritt vorgenommen, der allerdings alle zur Verfügung stehenden Punktkoordinaten, somit die geometrischen Informationen des gesamten Tiefenscans, verwendet. Dieses wird dadurch realisiert, dass eine Variante des Iterative Closest Point (ICP) Algorithmus verwendet wird, der bereits in 2.2.2 eingeführt wurde. Dabei wird die Punktwolke des gerade zu untersuchenden Gesichtes auf die eines Referenzgesichtes abgebildet. In dieser Anwendung wird, genau wie bei dem ersten groben Ausrichtungsschritt, einfach ein Gesicht aus der Datenbank manuell auf eine frontale Blickrichtung ausgerichtet und für alle anderen Gesichter als Referenz verwendet. Anschließend wird jeder mit dem beschriebenen Detektionsansatz gefundener Gesichtssausschnitt mittels ICP auf diese Referenz ausgerichtet.

Da der ICP Algorithmus in diesem Fall Bilder, die verschiedene Personen mit unterschiedlichen Blickwinkeln und Gesichtsausdrücken zeigen, aufeinander ausrichten soll, bietet sich die Variante des Trimmed ICP an (siehe [CSSK02]). Dabei werden nicht alle zur Verfügung stehenden Punkte zum Ausrichten verwendet, sondern nur die n Punktpaare mit den n kleinsten Abständen. Dadurch wird verhindert, dass einzelne Regionen, die nur in einem Ausschnitt vorhanden sind, oder deren lokale Punktverteilungen total unähnlich den korrespondierenden lokalen Verteilungen der Referenz sind, das Ergebnis ungünstig beeinflussen. Wenn also in einem der beiden Kopfscans z.B. der Hals zu sehen ist, und im anderen nicht, würde sich unter Verwendung aller Punkte kein gutes Ergebnis erzielen lassen, da ja auch die Punkte, die nur in einer Punktwolke vorkommen, mit in die zu minimierende Kostenfunktion einfließen würden.
Der ICP Algorithmus konvergiert immer zum nächsten lokalen Minimum. Da aber durch Grobausrichtung auf die durch den Detektor gefundenen Augenecken und Nasenspitzenkoordinaten eine gute Initialisierung vorgenommen werden kann, können hier gute Ergebnisse erzielt werden.

Wenn man sich noch einmal die Grundschritte des ICP für das Ausrichten einer Punktwolke A auf die Punktwolke B anschaut, nämlich

- Finde für jeden Punkt in B den nächten Punkt in A. Sortiere die Punktpaare nach Abstand.
- Erzeuge die Punktmengen A^* und B^* die aus den n Punktpaaren mit den niedrigsten Abständen bestehen.
- Löse das Gleichungssystem

$$B^* = s \cdot R \cdot A^* + t \tag{6.18}$$

  um Schätzungen für Skalierung s, Rotation R und Translation $\vec{t}$ zu erhalten.
- Wende die geschätzte Transformation an, und erzeuge den Datensatz für die nächste Iteration

$$A \leftarrow s \cdot R \cdot A + \vec{t} \tag{6.19}$$

- Wiederhole bis Konvergenz.

so sieht man, dass die Rechenzeit des Gesamtalgorithmus im erheblichen Maße von der Anzahl der Punkte und vom Finden der Punktkorrespondenzen abhängig ist. Dieses findet in der Regel über eine vollständige Suche statt. Um diesen Verarbeitungsschritt zu beschleunigen wird der 3D-Raum in Würfel zerlegt, im Englischen bezeichnet man dieses als “boxed space”. Wenn nun für einen gegebenen Punkt der nächste Nachbar gesucht wird, wird nur innerhalb des gleichen Würfels und der benachbarten Würfel gesucht. Im vorliegenden Fall wurde der Raum in $7 \times 7 \times 7$ Teile

Abbildung 6.7: Beispiele für ausgerichtete und normalisierte Gesichter.

zerlegt, also in 343 Würfel. Bei der Suche eines nächsten Punktes werden nur noch die $3 \times 3 \times 3$ räumlich benachbarten Würfel abgesucht also nur noch 27. Bei jeder einzelnen Suche wird sich also erspart, $\frac{343-27}{343} \approx 92.1\%$ der Daten zu durchsuchen (bei angenommener räumlicher Gleichverteilung der Punkte). Eine weitere Maßnahme, den Suchaufwand zu verringern, ist es, die Anzahl der Meßpunkte in den ersten Iterationsschritten niedrig zu halten, und erst später, wenn nur noch feine Veränderungen auftreten, mit der Gesamtheit aller zur Verfügung stehender Meßpunkte zu arbeiten. Im vorliegenden Fall mit einer Meßpunktanzahl pro Tiefenscan von ca. 10000 bis 20000 Punkten wurde in den ersten Iterationsschritten gleichabständig bis auf 1000 Punkte unterabgetastet. Erst nach einer festen Anzahl von Iterationen, oder bei Erreichen von Konvergenz auf der jeweiligen Auflösungstufe, wird zu den nächsten Auflösungstufen von 2000, 5000 und als letztes von allen Meßpunkten umgeschaltet.
Für die Lösung des Gleichungssystems $B^* = s \cdot R \cdot A^* + \vec{t}$ wurde die Quaternionenmethode verwendet, die im Anhang C ausführlich erläutert wird.

Insgesamt konnte eine ICP Variante realisiert werden, die recheneffizient arbeitet und gute "Feinanpassungen" an das Referenzgesicht lieferte. Nachdem die Gesichter alle ausgerichtet wurden, werden sie ähnlich wie in der 2D-Gesichtserkennung ausgeschnitten. Da ja eine Ausrichtung auf eine Referenz stattgefunden hat, kann der zu verwendende Ausschnitt relativ zu dieser Referenz definiert werden. Auch hier wurde wieder ein Ausschnitt verwendet, der relativ zum Augenabstand des Refenzgesichtes definiert wurde. Ein fester, in der Bildebene rechteckiger Ausschnitt mit einer Breite von zwei und einer Höhe von drei Augenabständen wurde verwendet. Die Ausschnitte werden alle auf eine feste Größe, hier 128×128 Pixel, normiert und der Wertebereich wird linear auf [0..255] skaliert.

In Abbildung 6.8 ist dargestellt, wie sich beim Durchlauf des Trimmed ICP eine Punktwolke nach und nach auf die Referenz zieht. Einige Beispiele extrahierter Ausschnitte nach der Ausrichtung mittels ICP werden als Tiefenbild in Abbildung 6.7 dargestellt.

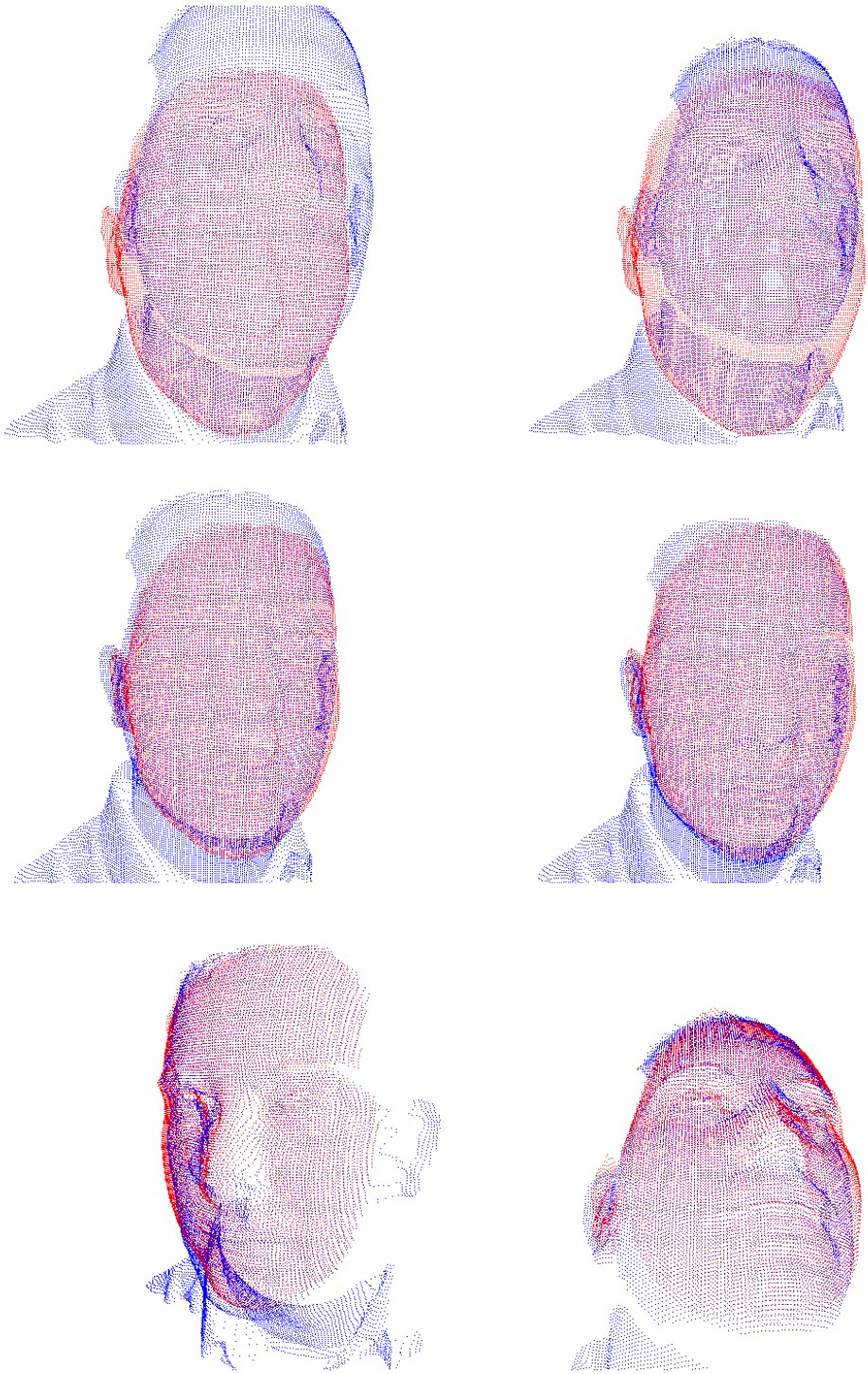


Abbildung 6.8: Ausrichtung mit Trimmed ICP, die blaue Punktwolke wird auf die rote Referenzpunktwolke ausgerichtet, links oben: Initialisierung, dann schrittweise Ergebnisse nach 1, 50 und 143 Iterationen. Die letze Zeile zeigt das Resultat nach 143 Iterationen (Konvergenz) in verschiedenen 3D-Ansichten.

6.4 Erkennung

Die extrahierten Ausschnitte als Tiefenbilder können jetzt mit klassischen Erkennungsmethoden, wie bei frontalen Bildausschnitten, bearbeitet werden. Die eigentliche Leistung des hier beschriebenen Ansatzes ist also die Erzeugung blickrichtungsnormierter Ansichten und nicht etwa die Neuentwicklung des eigentlichen Klassifikators. Trotzdem gilt es die erreichten Ergebnisse mit anderen zu vergleichen. Im Folgenden wird der bereits im Fall der 2D-Gesichtserkennung auf frontalen Gesichtsausschnitten verwendete P2D-HMM Ansatz angewendet, um mit der Literatur vergleichbare Ergebnisse zu erzeugen.
Im konkreten Fall wurde ein diskretes P2D-HMM mit 5×5 Zuständen und einer Kodebuchgröße von 1024 verwendet. Als Merkmale wurden wieder die 18-dimensionalen DCTmod2-Merkmale mit einem Überlappungsgrad von 50% blockweise extrahiert, wie es ebenfalls bereits bei der klassischen texturbasierten 2D-Gesichtserkennung durchgeführt wurde. Der einzige Unterschied besteht hier darin, dass diesmal die Merkmale auf Tiefendaten berechnet wurden, nicht auf Texturdaten.
Es sei noch darauf hingewiesen, dass dieser Ansatz sehr leicht auf eine Erkennung unter Verwendung von 3D-Daten, also Tiefen- und Texturinformationen, zu erweitern ist. Die Grundstruktur des gesamten Algorithmus ist im einfachsten Fall die gleiche, es werden lediglich die Merkmale der Farbkanäle zu den jeweilige Merkmalsvektoren hinzugefügt. Im Fall einer Erkennung auf RGB- und Tiefendaten bedeutet dieses, dass die DCTmod2 Merkmale auf jedem der 4-Känale (Rot, Grün, Blau und Tiefe) einzeln berechnet werden und anschließend ein kombinierter Merkmalsvektor durch Aneinanderhängen erzeugt wird. Dann wird noch die Kodebuchgröße unter Berücksichtigung des nun wesentlich hochdimensionaleren Merkmalsraum angepasst, der restliche Ablauf würde gleich bleiben.

6.5 Ergebnisse

6.5.1 Evaluierung der Ausrichtung

Als Erstes wird eine Evaluierung der Ausrichtungsschritte vorgenommen. Für den Detektionsschritt, der ja gleichzeitig die erste grobe Ausrichtung vornimmt, wurde hier manuell überprüft, ob das durch den Detektionsalgorithmus gefundene Augenecken-Nasenspitzen Tupel tatsächlich die Koordinaten der Augenecken und der Nasenspitze zeigt. Für den zweiten Ausrichtungsschritt durch den Trimmed Iterative Closest Point Algorithmus, wurde die Korrektheit anhand des resultierenden finalen Bildausschnitts bewertet. Hier wurde überprüft, ob der Gesichtausschnitt Stirn, Augen, Nase und Mund enthält, die Augen horizontal ausgerichtet wurden, und keine weiteren Elemente, die nicht zum Gesicht gehören, wie zum Beispiel der Hals, enthalten sind (siehe Abbildung 6.9). Es ist klar, dass diese Bewertungsmethoden nur

Abbildung 6.9: Beispiele für korrekt ausgerichtete Ausschnitte (links) und falsch ausgerichtete Ausschnitte (rechts).

	Krümmungsmerkmale + PCA	Trimmed ICP
neutral1	1.6	0.0
neutral2	0.0	0.0
random	4.9	1.6
laugh	8.1	1.6
smile	6.5	0.0
look up	14.7	13.1
look down	4.9	0.0

Tabelle 6.1: Anteil der falsch ausgerichteten Daten in % je nach Datenset nach dem Detektionsschritt und dem Ausrichtungsschritt mittels ICP.

qualitativ bewerten, ob ein Ausschnitt korrekt ist oder nicht, aber nicht quantifizieren, wie groß ein Fehler ist. Hier wären in Zukunft weitere Untersuchungen sinnvoll.

Es zeigt sich, dass die Ausrichtung insgesamt gut funktioniert. Interessant ist, dass der Trimmed ICP Algorithmus offensichtlich in der Lage ist, sogar einige der Fehllokalisationen des beschriebenen Detektionsansatzes zu kompensieren. Das lässt sich dadurch erklären, dass die Detektionsstufe ja nur die Initialisierung des ICP vorgibt, dieser dennoch zum nächsten lokalen Minimum hin konvergiert. Wenn also die Fehldetektion nur einen relativ kleinen räumlichen Transformationsunterschied darstellt, kann dieses kompensiert werden.

Auffällig ist, dass innerhalb des Datensets *look up* die meisten Fehler auftreten. Dadurch, dass die Personen innerhalb dieses Datensets nach oben schauen, verschieben sich die Abstände von Augenecken und Nasenspitze in der Bildebene, somit werden recht schnell die doch recht einfachen Regeln zum Aussortieren der Hypothesen verletzt, außerdem wird der Inhalt des Ausschnittes, der durch den PCA-Klassifikator bewertet wird, nur aus einem sehr schmalen Höhenbereich extrahiert. Eine Anpassung der Regeln und eine 3D-Rotation der Hypothese in die Bildebene vor dem Ausschneiden müsste diesen Effekt reduzieren können.

Ein ähnlicher Effekt tritt auch bei gesenkten Blicken auf, wie sie im Datenset *look down* enthalten sind. Allerdings werden hier durch die natürliche Geometrie des Gesichtes weitaus extremere Auslenkungen der Blickrichtung nach unten benötigt, um die gegebene Regel zu verletzen oder den Ausschnitt zu schmal werden zu lassen.

Abbildung 6.10: Die Positionen der detektierten Augenecken werden verwendet, um das Bild in der Bildebene auszuschneiden, ohne dass eine Ausrichtung stattfindet. Die Bilder zeigen eine Person in den 7 verschiedenen Datensätzen.

Als zweites Experiment, um die Ausrichtung auszuwerten, werden nun die Erkennungsraten in Abhängigkeit der verwendeten Ausrichtungsschritte aufgelistet, siehe Tabelle 6.2. Als Erkenner wurde ein P2D-HMM mit 5×5 Zuständen eingesetzt. Die DCTmod2 Merkmale wurden aus 8×8 Pixel großen Blöcken extrahiert, benachbarte Blöcke wurden dabei um 4 Pixel überlappt. Die Merkmale wurden mittels k-means auf 1024 Quantisierungsstufen diskretisiert. Als Trainingsdatensatz wurde *neutral1* verwendet, zusätzlich aus diesen Daten abgeleitete zufällige Variationen durch Einsatz von Rotation, Translation und Skalierung.
Alle Ergebnisse werden bezogen auf einen Ausschnitt angegeben, der nur den Kopf enthält, um zu vermeiden, dass das Mustererkennungsverfahren Merkmale lernt, die nur innerhalb dieser Datenbank charakteristisch für einzelne Personen sind, wie zum Beispiel die während der Datenbankerstellung getragenen Kleidung, die Körperhaltung und Weiteres.

Um die Erkennung mit und ohne Ausrichtungsschritte zu bewerten, wurden ausgehend von den durch den Detektor gefundenen Nasen- und Augenpositionen verschiedene Kopfausschnitte erzeugt: Der erste Fall schneidet das Tiefenbild aufgrund der 2D-Position der Augen- und Nasenhypothesen in der Bildebene aus (siehe Abbildung 6.10), der Zweite nutzt die 3D-Koordinaten der Augen- und Nasenhypothesen, um diese mittels einer 3D Transformation auf eine frontale Referenz abzubilden und schneidet erst dann in der Bildebene aus (siehe Abbildung 6.11).

Abbildung 6.11: Die Positionen der detektierten Augenecken und Nasenspitze werden verwendet, um den 3D Datensatz grob auf Referenzpositionen auszurichten. Das wirkt sich bei frontalen Ansichten kaum aus, einen kleinen positiven Effekt hat dies vor allen für extreme Blickwinkel. (Bild 6, *look down* und Bild 7, *look up*), vergl. Abbildung 6.10.

Abbildung 6.12: Die Kopfausschnitte nach Durchführung des abschließenden Aussrichtungsschrittes durch Trimmed ICP (eine Person in den 7 verschiedenen Datensets).

	Gesamter 3D-Scan	Kopf-Ausschnitt in Bildebene	Ausschnitt nach 3D-Ausrichtung durch Detektor	Ausschnitt nach Trimmed ICP
neutral2	*0.901*	0.721	0.786	0.951
random	*0.770*	0.417	0.428	0.738
laugh	*0.820*	0.525	0.541	0.721
smile	*0.868*	0.639	0.672	0.902
look up	*0.229*	0.164	0.428	0.803
look down	*0.344*	0.197	0.521	0.885

Tabelle 6.2: Einfluss der Ausrichtung auf die Erkennungrate: Experimente mit und ohne Einsatz der verschiedenen Ausrichtungsschritte (auf dem Kopfausschnitt). Für weitere Vergleiche werden noch die Erkennungsergebnisse unter Verwendung des gesamten Tiefenscans, ohne dass eine Detektion oder Ausrichtung stattgefunden hat, angegeben (kursiv).

Der dritte Fall nutzt die komplette Ausrichtungskette, es wird also zusätzlich eine Feinausrichtung mittels Trimmed ICP durchgeführt, bevor in der Bildebene ausgeschnitten wird, siehe Abbildung 6.12.

Die erste Ergebnisspalte in Tabelle 6.2 zeigt die Ergebnisse auf den gesamten Tiefenscans ohne Detektion und Ausrichtung, also einschließlich Oberkörper. Hier kann der Klassifikator, zumindest bei recht ähnlichen Körperhaltungen (*neutral2* bis *smile*), zusätzliche Charakteristika verwenden, die innerhalb dieser Datenbank personenspezifisch sind (Kleidung, Körperhaltung) und somit die Erkennungsrate positiv beeinflussen. Allerdings ist ein Einsatz dieser Ausschnitte in der Praxis nicht zu empfehlen, da dort, im Gegensatz zu einer Momentaufnahme in einer Datenbank, natürlich mittel- und langfristig keine personenspezifischen Körperhaltungen oder Kleidungen vorkommen.
Auf den tatsächlich interessierenden Ausschnitten, die nur den Kopf zeigen, und somit stabile personenspezifische Informationen liefern, konnte durch die Ausrichtungsschritte erreicht werden, dass ein normierter, für den Klassifikator gut geeig-

neter Ausschnitt zur Verfügung steht. Die erste grobe 3D-Ausrichtung anhand der nur 3 Koordinatenangaben für die Augenecken und Nasenspitze alleine genommen erzielt schon eine kleine Verbesserung gegenüber dem unausgerichtetem Kopfausschnitt. Wenn anschließend noch ein Trimmed ICP zur genaueren 3D-Ausrichtung ausgeführt wird, werden sehr gute Ergebnisse erzielt. Dabei ist der Trimmed ICP abhängig von einer groben Initialisierung, wie sie durch den ersten Ausrichtungsschritt vorgenommen wurde, er könnte also nicht alleine, ohne diese Initialisierung, gute Ausrichtungsergebnisse erzielen.

6.5.2 Vergleich zur anderen Ansätzen

Zum Vergleich werden nun die hier erreichten Ergebnisse und die Ergebnisse von drei weiteren Algorithmen, deren Ergebnisse auf dieser Datenbank publiziert wurden, aufgeführt, siehe Tabelle 6.3.

Der als "salient wrinkles"[ABDP06] beschriebene Ansatz untersucht dabei ebenfalls Punkte mit auffälligen Krümmungsinformationen. Er erzeugt daraus einen hochdimensionalen Merkmalsvektor und verwendet diesen direkt für die Klassifikation.

Der Ansatz der "iso-geodesic stripes" [BDP06], extrahiert den geometrischen Verlauf von Streifen, die den gleichen Abstand zur Nasenspitze haben. Dabei wird dieser Abstand allerdings nicht räumlich, sondern entlang der Oberfläche des Gesichtes gemessen. Dieses kann dadurch erreicht werden, dass die Punktwolken trianguliert werden, und als Abstandsmaß der kürzeste Pfad entlang der Triangulationskanten ermittelt wird.

Ein dritter Ansatz, die sogenannten "ridge images" [MAM09], erzeugt 3D-Binärbilder aus den Tiefendaten, indem Krümmungskanten detektiert werden. Eine Ausrichtung mittels ICP verwendet dann nur die 3D-Koordinaten dieser Kanten, eine Bewertung der Ähnlichkeit findet dann direkt über den Wert der durch den ICP minimierten Kostenfunktion, der Summe der Abstände der Punkte auf diesen Kanten, statt.

Für diese Ansätze werden die jeweils besten Erkennungsergebnisse unter Verwendung verschiedener Parametereinstellungen aufgelistet. Der Ansatz, der in dieser Arbeit beschrieben wird, ist in der Tabelle 6.3 fettgedruckt, in den Zahlen werden die Gesamterkennungsergebnisse angegeben, einschließlich der in Tabelle 6.1 aufgeführten Fehldetektionen. Dabei wurde ausschließlich der Datensatz *neutral1* zum Training verwendet. Für alle hier aufgelisteten Erkennungsergebnisse wurden dabei die gleichen Parametereinstellungen verwendet (5×5 Zustände, Kodebuchgröße 1024, 50% Überlappung der Merkmalsblöcke).

	Ausrichtung + P2D-HMM	salient wrinkles [ABDP06]	iso-geodesic stripes [BDP06]	ridge images [MAM09]
neutral2	**0.951**	0.91	0.945	0.950
random	0.738	0.77	**0.805**	0.634
laugh	0.721	0.80	**0.811**	0.689
smile	**0.902**	0.83	0.844	0.836
look up	0.803	0.77	**0.928**	0.886
look down	0.885	0.78	**0.933**	0.853

Tabelle 6.3: Erkennungsraten auf den verschiedenen Datensets.

Es bleibt anzumerken, dass der hier beschriebene Erkennungsansatz unter Verwendung des gesamten 3D-Datensatzes, und nicht nur des Kopfausschnittes, mit 77% und 82% vergleichbare Erkennungraten auch für die Datensets *random* und *laugh* liefert (siehe Tabelle 6.2), allerdings wurde bereits diskutiert, dass ein Einsatz des gesamten Ausschnittes in der Praxis sinnlos wäre.

Einige weitere publizierte Ergebnisse beziehen sich ausschließlich auf die 5 näherungsweise frontalen Ausschnitte (*neutral1, neutral2, random, laugh, smile*). Dabei wird eine Erkennungsrate, die durch eine "Leave-One-Out"-Strategie ermittelt wurde, angegeben. Das bedeutet, es werden jeweils 4 Tiefenscans zum Training verwendet, auf dem Verbleibenden wird klassifiziert. So wurde eine Erkennungrate von 91% durch eine Projektion der Daten mittels einer 2D-PCA und anschließende Klassifikation mittels Support Vector Machines [MFA08] erreicht. Eine Erkennungsrate von 93.7% konnte durch eine andere Form der Anwendung des ICP erreicht werden [NC08]. Dabei wurden manuell definierte Subregionen der 3D-Scans aufeinander abgebildet, dann findet eine Bewertung durch die Abstände der jeweiligen Subregionen zweier Bilder statt.
Zum Vergleich: Der im Rahmen dieser Arbeit vorgestellte Algorithmus erreicht bei Anwendung dieser "Leave-One-Out" Strategie auf den resultierenden 5 Validierungssets eine mittlere Erkennungsrate von 93.1% auf den automatisch ausgerichteten Daten, dabei schwanken die Erkennungsraten der einzelnen Sets zwischen 85.2% (Testset: *random*) und 98.4% (Testset: *neutral1*).

Wie sich zeigt, kann der hier beschriebene Algorithmus mit dem aktuellen Stand der Technik mithalten. Die erzielten Erkennungsergebnisse auf den Datensets *neutral2* und *smile* sind besser, als die aller anderen bisher veröffentlichten Algorithmen, dafür sind die auf den anderen Datensets leicht schlechter. Das schlechtere Erkennungsergebnis auf dem *look up* Datenset lässt sich durch das teilweise Ver-

sagen des Detektors erklären, welches bereits diskutiert wurde. Ansonsten ist es auffällig, dass für emotionslose (*neutral2*) oder das Gesicht nur leicht verformende Gesichtsausdrücke (*smile*) sehr gute Erkennungsergebnisse erzeugt werden. Für Gesichtsausdrücke, die sich allerdings stark in der Gesichtsoberfläche auswirken (*laugh*, starkes Lachen, oft mit offenem Mund, *random* zufällige Emotion, oft sehr stark ausgeprägt, teilweise mit Verdeckungen durch Hände im Gesicht), schlechtere Ergebnisse erzielt werden. Da hier nur das Datenset *neutral1*, ein emotionsloser, frontaler Gesichtsausdruck, zum Training verwendet wurde, ist es plausibel, dass die Erkennung auf Datensets, die davon stark abweichen, schlechter wird. An dieser Stelle sei noch einmal betont, dass hier eine vollautomatische Verarbeitungskette, einschließlich automatischer Detektion und Ausrichtung ausgewertet wurde, nicht nur der Erkennungsschritt.

6.6 Zusammenfassung und Ausblick

Ein neuartiger Ansatz zur Erkennung von Gesichtern in Tiefendaten, die aus texturlosen räumlichen Informationen bestehen, wurde vorgestellt. Dazu wurde ein Detektionsansatz, eine Kombination von Hypothesenbildung durch Krümmungsmerkmale und anschließender Klassifikation durch einen PCA basierten Klassifikator, entwickelt. Das Ergebnis dieser Detektion wird mit einem Trimmed ICP auf eine Referenz präzise ausgerichtet.
Durch diese neuartige Ausrichtungskette, die vollautomatisch auf den gegebenen Tiefendaten arbeitet, können blickrichtungsnormierte Ausschnitte erzeugt werden, die dann mit herkömmlichen Klassifkationsansätzen ausgewertet werden können. Eine abschließende Erkennung auf dem normierten Gesichtsausschnitt wurde hier durch das bereits bekannte und mehrfach eingesetzte P2D-HMM durchgeführt. Insgesamt werden durch das Zusammenspiel dieser verschiedenen Algorithmen Erkennungsergebnisse erzeugt, die mit dem aktuellen Stand der Technik konkurrieren können und diesen teilweise überbieten.

Für zukünftige Untersuchungen bietet das Themenfeld der 3D-Gesichtserkennung vielfältige Möglichkeiten.
So kann der hier beschriebene Ansatz sehr leicht auf eine zusätzliche Verwendung von Texturinformationen erweitert werden, indem im einfachsten Fall zunächst einmal die Texturmerkmale dem Klassifikator zusätzlich zur Verfügung gestellt werden. Wenn man sich die erzielten Ergebnisse aus der klassischen 2D-Gesichtserkennung anschaut, die ja rein auf Texturinformationen basieren, und sich vorstellt, dass diese mit den hier erzielten Erkennungsergebnissen, die ausschließlich auf Tiefeninformationen basieren, fusioniert werden, so wäre es denkbar, dass aus der Kombination durchaus sehr zuverlässige Erkennungsalgorithmen erzeugt werden könnten.
Weitere Untersuchungen könnten zusätzlich aufzeigen, wie die Texturdaten nicht

nur in der Erkennungsstufe, sondern auch in der Detektions- und Ausrichtungsstufe sinnvoll genutzt werden könnten. Der hier vorstellte Detektionsansatz könnte mit den bekannten texturbasierten Detektionsansätzen kombiniert werden um zusätzliche Robustheit zu garantieren. Weiterhin wäre es denkbar, die im ICP geschätzten Punktkorrespondenzen, die ja momentan ausschließlich aufgrund räumlicher Distanz ermittelt werden, durch eine Methode zu ermitteln, die ebenfalls die Texturinformationen berücksichtigten würde. Dieses könnte im einfachsten Fall durch Anpassung des Abstandsmaßes geschehen, welches verwendet wird, um für jeden Punkt der Referenz den nächsten Nachbar im auszurichtenden Datensatz zu ermitteln. Durch diese oder ähnliche Maßnahmen müsste es möglich sein, eine noch robustere Ausrichtung zu erzielen.

Auch der Aspekt, dass es bei vorhandenen geometrischen Informationen möglich ist, physikalische Beleuchtungsmodelle einzusetzen, um schwierige Beleuchtungsbedingungen in Texturkanälen zu kompensieren, ist ein starkes Argument für die zukünftige Erforschung der 3D-Gesichtserkennung.

Letztendlich wird die ständige Weiterentwicklung der Meßtechnik und 3D-Datenerfassung automatisch dafür sorgen, dass 3D-Daten einem breiten Feld von Anwendungen zur Verfügung stehen. Dadurch werden auch die Algorithmen, die sich mit der Bearbeitung dieser Daten befassen, an Bedeutung gewinnen und noch stärker ins Blickfeld rücken.

7

Zusammenfassung

Im Rahmen dieser Arbeit wurden mehrere grundverschiedene Ansätze und Anwendungsgebiete der Gesichtserkennung untersucht. Diese Anwendungsgebiete reichten von der klassischen Zutrittskontrolle mittels Erkennung auf frontalen Gesichtsbildern über die Klassifikation von Bildern unterschiedlicher Blickwinkel bis hin zur Erkennung auf 3D- und Tiefendaten.

7.1 Authentifikation auf Frontalansichten

Als Erstes wurde anhand ausführlicher Experimente untersucht, wie sich zwei Standardalgorithmen der Gesichtserkennung, nämlich Eigenfaces und Pseudo-Zweidimensionale Hidden Markov Modelle, verhalten, wenn man eine Zusammenführung zur Lösung einer Authentifikationsaufgabe erreichen möchte. Die dabei zu erzielende Lösung war stark an anwendungsnahe einschränkende Bedingungen hinsichtlich Rechenzeit und Speicherbedarf gebunden. Dennoch konnte durch geschickte Kombination der beiden Ansätze ein Algorithmus entwickelt werden, der trotz Einschränkungen bezüglich der verwendeten Ressourcen besser war, als jeder der beiden Algorithmen für sich. Für die gegebene Aufgabe der Authentifikation konnte unter Verwendung begrenzter Ressourcen eine Equal Error Rate von 3.4% erreicht werden, 1.9% ohne diese Begrenzung, welches durch einen Test auf einer repräsentativen Datenbank (FERET) ermittelt wurde. In einer Online-Demonstration der entwickelten Anwendung konnte sogar eine Equal Error Rate von 0% erreicht werden, allerdings wurde nur auf einer kleinen Stichprobe von 32 Personen getestet. Insgesamt konnten alle Anforderungen einer realen Anwendung bezüglich Rechenzeit, Speicherplatzbedarf und Erkennungsleistung dieser Authentifikationaufgabe erfüllt werden.

7.2 Blickrichtungsunabhängige Erkennung in 2D-Bildern

Als Nächstes wurde sich der Aufgabenstellung zugewandt, Personen beliebiger Blickrichtung hinsichtlich ihrer Identität zu klassifizieren. Als Ausgangspunkt für zu entwickelnde Algorithmen standen jeweils die mehr oder weniger präzise detektierten Bildausschnitte zur Verfügung, die den Kopf enthielten. Um diese Aufgabenstellung zu lösen, wurden zwei unterschiedliche Ansätze untersucht.
Der erste Ansatz ermittelte durch Anwendung von Active Appearance Modellen eine parametrische Beschreibung der in dem untersuchten Bildausschnitt jeweilig vorhandenen Gesichter und zusätzlich ein Konfidenzmaß, welches aufgrund der Ähnlichkeit zwischen einem durch das Modell synthetisiertem Gesicht und dem originalen Bildausschnitt bewertete, wie gut diese Beschreibung ist. Der entstehende Parametervektor wurde dann durch ein neuronales Netz hinsichtlich der Identität der Person klassifiziert. Als Ergebnis lagen hier sehr gute Klassifikationsraten von 92.5% vor, unter der Annahme, dass nur solche Bildausschnitte klassifiziert werden, die durch das Konfidenzmaß als gut bewertet wurden. Im vorliegenden Fall waren dies etwa 15% der Bilder, oder bei angenommener Gleichverteilung der Bilder über die Rotationswinkel, ein Blickwinkelbereich von ca. $\pm 27°$.
Ein zweiter Ansatz sollte unabhängig davon, welcher Blickwinkel gerade zu sehen ist, eine Klassifikationsaussage liefern. Als Modellvorstellung wurde angenommen, dass die Textur einer Person in verschiedenen Blickwinkeln näherungsweise der Projektion der dreidimensionalen Person auf eine umschließende Zylinderoberfläche entspricht. Aufgrund dieser Modellvorstellung wurde ein zyklisches P2D-HMM entwickelt, welches aus einem geschlossenenen Ring von Spaltenmodellen besteht. Durch geschickte Initialisierungen beim Training durch künstliche Rundumansichten konnte gewährleistet werden, dass die horizontale Abfolge von Merkmalen, die ja je nach Rotation des Blickwinkels eine unterschiedliche Reihefolge hat, im Modell festgehalten werden kann. Dieser Ansatz wurde auf zwei verschiedenen Datensätzen getestet und mit klassischen, nicht zyklischen P2D-HMM verglichen. Unter Laborbedingungen, bei Verwendung von 8 verschiedenen Blickrichtungen zum Training, konnte das zyklische Modell eine Identifikationsrate von ca. 89% erreichen, ein klassisches P2D-HMM erreichte unter den gleichen Bedingungen nur ca. 84%. Bei Versuchen auf einem sehr anspruchsvollen, unter Realweltbedingungen aufgenommenen Datenset mit extremen Beleuchtungsbedingungen und schlechter Bildqualität konnte immerhin noch eine Klassifikationsrate von ca. 45% durch das zyklische Modell im Vergleich zu ca. 40% bei Verwendung der klassischen P2D-HMM Struktur erreicht werden, dabei lag die Ratewahrscheinlichkeit bei 4%. Die klassifizierten Bildausschnitte schließen dabei alle Blickrichtungswinkel mit ein, auch Profil- und Hinterkopfansichten.

7.3 Vergleich von Bildpaaren

Weiterhin wurde in dieser Arbeit der Anwendungsfall des Vergleichens von Bildpaaren untersucht. Hier standen unzählige, unter unbekannten Umgebungsbedingungen und Blickwinkeln entstandene Bilder von Personen aus der "Labeled Faces in the Wild"-Datenbank zur Verfügung. Nun sollten Bildpaare dahingehend klassifiziert werden, ob in beiden Bildern jeweils die gleiche Person zu sehen ist, oder nicht. Ein neuer Ansatz, der eine gewichtete Summe von Ähnlichkeitsmaßen bildet und aufgrund dieser entscheidet, wurde vorgestellt. Dabei wurden die zu vergleichenden Bilder in sich überlappende Bildausschnitte zerteilt. Die korrespondierenden Regionen beider Bilder wurden jeweils aufgrund von direkten Ähnlichkeitsmaßen oder durch Merkmalshistogramm-basierte Ähnlichkeitsmaße verglichen. Eine gewichtete Summe über die verschiedenen Arten von Ähnlichkeitsmaßen und die verschiedenen Bildregionen führte zu einem Wert, auf dem mittels eines Schwellwertklassifikators entschieden wurde. Dabei wurden die Gewichte der einzelnen Maße und Bildregionen durch ein Lernverfahren, nämlich die Anwendung eines neuronalen Netzes, ermittelt. Insgesamt konnte mit ca. 74% eine gute Erkennungsleistung erreicht werden, die mit den Leistungen der aus der Literatur bekannten Algorithmen konkurrieren kann.

7.4 Gesichtserkennung auf Tiefendaten

Als Letztes wurde das Anwendungsgebiet der Gesichtserkennung anhand Tiefendaten behandelt. Tiefendaten sind dadurch interessant, dass geometrisches Wissen über das zu klassifizierende Gesicht vorhanden ist. Anhand dieses Wissens kann eine Ausrichtung auf einen normierten Gesichtsausschnitt mit normiertem Blickwinkel vorgenommen werden, auf dem dann anschließend klassifiziert werden kann. Um das zu erreichen, wurde zunächst ein Detektionsansatz vorgestellt, der anhand der Oberflächenkrümmung des Gesichtes Hypothesen für Augenecken und Nasenspitze ermittelte. Anschließend wurden die Hypothesen durch einen PCA-basierten Klassifikator als wahr oder falsch bewertet. Die so gefundenen Nasen- und Augenkoordinaten dienten zu einer ersten, groben Ausrichtung auf eine frontale Gesichtspose. Ein zweiter Auschrittungsschritt mittels Iterative Closest Point Algorithmus führte daraufhin eine Feinausrichtung auf ein Referenzgesicht aus, wodurch ein blickrichtungsnormierter Gesichtsausschnitt erzeugt wurde. Eine Klassifikation dieses Ausschnittes hinsichtlich der Identität der Person konnte für neutrale Gesichtsausdrücke mit 95% Erkennungsleistung und für lächelnde Gesichter mit ca. 90% erreicht werden, welches die Leistung aller bis jetzt veröffentlichten anderen Ansätze überbietet. Für andere Gesichtsausdrücke konnten immerhin noch Erkennungsraten $> 72\%$ erreicht werden, welches etwa dem aktuellen Stand der Technik entspricht. Insgesamt konnte also durch das gezielte Ausrichten der Tiefendaten eine robuste Erkennungsstrategie entwickelt werden.

A

Maße zur Bewertung der Leistungsfähigkeit

In diesem Anhang werden gängige Qualitätsmaße zur Beurteilung der einzelnen Verarbeitungsschritte Detektion, Ausrichtung und Erkennung beschrieben. Je nach Evaluationsstrategie werden die hier aufgeführten Masse noch mit weiteren statistischen Maßen wie Mittelwert, Standardabweichung und Varianz kombiniert, insbesondere dann, wenn mehrere Evaluationsrunden durchgeführt werden, wie bei der Kreuzvalidierung.

A.1 Detektion

Zur Beurteilung der Detektionsleistung spielen zwei Kriterien eine Rolle [MMP+02]: Erstens, ob ein Gesicht detektiert wurde, und zweitens, wo das Gesicht detektiert wurde.

Der erste Aspekt wird typischerweise durch die Maße "Falsch-Positiv"-, "Wahr-Positiv"-, "Falsch-Negativ"- und "Wahr-Negativ"-Rate beschrieben, und ist gleichwertig mit der Bewertung eines Zwei-Klassen-Klassifikators. In diesen einzelnen Maßen wird jeweils die Aussage des Detektors ("Positiv" oder "Negativ") und die Richtigkeit dieser Aussage ("Wahr" oder "Falsch") berücksichtigt. Gängigerweise werden die Abkürzungen FP, TP, FN und TP verwendet, hier steht "T" für "True", "F" für "False", "P" für "Positive" und "N" für "Negative". Für alle Vergleiche, die auf einem Testset durchgeführt werden, werden die absoluten Häufigkeiten dieser vier möglichen Fälle gezählt und in einer Tabelle aufgelistet (siehe Tabelle A.1).

Aus diesen Häufigkeiten ergeben sich die gängigen Maßeinheiten zur Beschreibung der Leistung eines Detektors.

	Gesicht vorhanden	Gesicht nicht vorhanden
Gesicht detektiert	TP	FP
Gesicht nicht detektiert	FN	TN

Tabelle A.1: Definition der Begriffe TP, TN, FP und FN. Die absoluten Häufigkeiten werden in solch eine Tabelle eingetragen.

- Falsch-Positiv Rate:
  $R_{FP} = \frac{FP}{FP+TN}$, ist der Anteil der tatsächlich negativen Testbeispiele, die fälschlicherweise als positiv klassifiziert wurden.

- Falsch-Negativ Rate:
  $R_{FN} = \frac{FN}{FN+TP}$, ist der Anteil der tatsächlich positiven Testbeispiele, die fälschlicherweise als negativ klassifiziert wurden.

- Spezifität(Wahr-Negativ Rate):
  $R_{TN} = \frac{TN}{FP+TN}$, ist der Anteil der tatsächlich negativen Testbeispiele, die richtig als negativ klassifiziert wurden.

- Sensitivität(Wahr-Positiv Rate):
  $R_{TP} = \frac{TP}{FN+TP}$, ist der Anteil der tatsächlich positiven Testbeispiele, die richtig als positiv klassifiziert wurden.

- Detektionsrate (Korrektklassifikationsrate):
  $R_T = \frac{TP+TN}{TN+FN+TP+FP}$, ist der Anteil der korrekt klassifizierten / detektierten Testdaten. Dieses wird auch oft als Klassifikationsleistung oder im englischen als “accuracy” beschrieben.

- Fehldetektionsrate (Falschklassifikationsrate):
  $R_F = \frac{FP+FN}{TN+FN+TP+FP}$, ist der Anteil der falsch klassifizierten / detektierten Testdaten.

Eine gängige Beschreibung eines Detektors findet sich in der ROC Kurve, der Receiver Operator Characteristics Kurve, in der die Wahr-Positiv Rate im Verhältnis zur Falsch-Positiv Rate in Anhängigkeit eines Parameters aufgetragen wird. Dies wird in der Regel dann getan, wenn ein einstellbarer Schwellwert T im Detektor die

Unterscheidung in "detektiert" oder "nicht detektiert" vornimmt und die optimale Einstellung gesucht wird, für die der Quotient $\frac{R_{TP}(T)}{R_{FP}(T)}$ maximal wird.

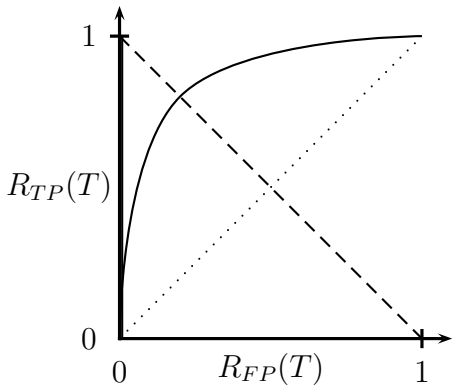


Abbildung A.1: ROC Kurve: Je weiter die Kurve links oben ist, desto besser ist der Detektor (TP-Rate hoch, FP-Rate niedrig). Die gepunktete Linie stellt die Ratewahrscheinlichkeit in einem Zweiklassenproblem dar. Der Schnitt der Kurve mit der gestrichelten Linie stellt den besten Arbeitspunkt dar, durch den der zu verwendende Schwellwert T ermittelt werden kann.

Das zweite Kriterium, dass bei der Auswertung von Detektionen eine Rolle spielt, ist die örtliche Lage des Detektionsergebnis in Bezug auf die tätsachliche Lage des zu detektierenden Objektes. Um dieses zu Beurteilen, werden die Maße Genauigkeit (engl. "precision") und Trefferquote (engl. "recall") verwendet. Die Genauigkeit ist dabei der Anteil der Pixel, die als Detektionsergebis vorliegen und auch tatsächlich Teil des zu detektierenden Objektes sind. Die Trefferquote ist der Anteil der tatsächlichen Objektpixel, die auch als Objektpixel detektiert werden. Im Idealfall sind beide Maße eins.

$$\textbf{Recall} = \frac{|\textbf{detektierte Pixel} \cap \textbf{Objektpixel}|}{|\textbf{Objektpixel}|} \tag{A.1}$$

$$\textbf{Precision} = \frac{|\textbf{detektierte Pixel} \cap \textbf{Objektpixel}|}{|\textbf{detektierte Pixel}|} \tag{A.2}$$

Als kombiniertes aussagekräftiges Maß werden Precision und Recall im harmonischen Mittel als F_1-Maß(engl. "f_1-measure") angegeben:

$$F_1 = \frac{2 \cdot (\textbf{Precision} \cdot \textbf{Recall})}{(\textbf{Precision} + \textbf{Recall})} \tag{A.3}$$

Je höher dieses F_1-Maß ist, desto besser ist der Detektor. Ein Wert von $F_1 = 1$ ist optimal.

A.2 Ausrichtung

Die Ausrichtung stellt in der Regel eine Optimierung des durch den Detektionsschritt gefundenen Ausschnitts dar, und wird deshalb oft mit den gleichen Qualitätmaßen beschrieben. Zusätzlich wird oft allerdings die Lage einzelner charakteristischer Merkmale in Beziehung gesetzt. Beim Anwendungsfall "Gesicht" ist dies typischerweise die Position von charakteristischen Punkten, wie Augen- oder Mundmittelpunkt. Als Fehlermaß wird dann oft der geometrische Punkt zu Punkt Abstand, also die euklidische Distanz, zwischen ausgerichtetem Gesicht und dem Referenzgesicht, auf das ausgerichtet wurde, verwendet. Wenn mehrere Punkte als Referenz verwendet werden, wird der mittlere und der maximale Abstand zwischen allen N Punkten im ausgerichteten Gesicht $\vec{p}_a$ und im Referenzgesicht $\vec{p}_r$ angegeben.

$$\overline{d}(\vec{p}_a, \vec{p}_r) = \frac{\sqrt{(\vec{p}_a - \vec{p}_r)^2}}{N} \tag{A.4}$$

ist dann der mittlere Punktabstand und

$$d_{max}(\vec{p}_a, \vec{p}_r) = \max_{i=1..N} \sqrt{(\vec{p}_{a,i} - \vec{p}_{r,i})^2} \tag{A.5}$$

der maximale Punktabstand.

A.3 Identifikation

Für die Identifikationsaufgabe wird gängigerweise die Erkennungsrate oder die Cumulative Matching Curve (CMC) angegeben [PMRR00]. Die Erkennungsrate RR ist dabei der Anteil der korrekt klassifizierten Testdaten zur Gesamtanzahl der Testdaten.

$$RR = \frac{\textbf{Anzahl korrekt klassifizierte Testdaten}}{\textbf{Gesamtanzahl Testdaten}} \tag{A.6}$$

Die CMC ist eine Kurve, die davon ausgeht, dass eine n-besten Liste als Ergebnis vorliegt. Diese n-besten Liste beschreibt dabei die n Klassen, die die n besten Klassifikationsergebnisse, also je nach Klassifikator die n kleisten Abstandswerte oder die n höchsten Ähnlichkeits- oder Wahrscheinlichkeitswerte liefern. Nun wird in der Cumulative Matching Curve angegeben, ob die richtige Klasse in den ersten n Klassen dieser Liste enthalten ist.

$$CMC(n) = \frac{\textbf{Anzahl "richtige Klasse in n - besten Liste enthalten"}}{\textbf{Gesamtanzahl Testdaten}} \tag{A.7}$$

Die CMC ist eine stetig steigende Kurve, da ja die Klassifikationsaufgabe für höhere n zunehmend einfacher wird. Wenn n der Gesamtanzahl der Klassen entspricht, ist die CMC immer eins. Für den Fall das $n = 1$ ist die CMC gleich der Erkennungsrate, die auch als Identifikationsrate bezeichnet wird.

	Klassifikation als gleiche Person	Klassifikation als ungleiche Person
tatsächl. gleiche Person	TP	FP
tatsächl. ungleiche Person	FN	TN

Tabelle A.2: Definition der Begriffe TP, TN, FP und FN für die Authentifikation und den Paarvergleich. Die absoluten Häufigkeiten werden in diese Tabelle eingetragen.

A.4 Authentifikation und Paarvergleich

Der Fall der Authentifikation, bei der es sich ja um ein Zweiklassenproblem handelt, wird ähnlich wie die Detektion mit FP,TP,FN und TN bewertet [RPM98]. Eine zu prüfende Person gibt sich als jemand aus, es wird überprüft, ob es die gleiche Person ist, wie die, für die sie sich ausgibt, oder nicht. Ähnlich verhält es sich beim Paarvergleich ("pair matching") bei dem ebenfalls festgestellt werden soll, ob es sich um die gleiche Person handelt oder nicht. Daraus ergibt sich die leicht veränderte Definition der jeweiligen Maße im Vergleich zu einer Detektionsaufgabe (siehe Tabelle A.2).

Die Berechnung der daraus abgeleiteten Maße und Raten verhält sich wieder genau so, wie bereits im Abschnitt A.1 erläutert.

Da viele Authentifikationsalgorithmen mit einem Schwellwert T arbeiten, anhand dessen entschieden wird, ob ein Gesicht als "gleich" oder "ungleich" gewertet wird (ähnlich dem Schwellwert bei der Erstellung einer ROC-Kurve beim Detektor), hat es sich bewährt, eine weiteres Maß einzuführen, die sogenannte "Equal-Error-Rate" (EER), also der Wert, an dem Falsch-Positiv- und Falsch-Negativ-Rate in Abhängigkeit eines Schwellwertes T gleich sind. Da je nach Wahl dieses Schwellwertes, die einzelnen FP- und FN-Raten beliebig skaliert werden können, dieses aber zueinander gegenläufig ist, ist die EER ein gutes objektives Beschreibungmittel zur Leistungsfähigkeit eines Authentifikationsalgorithmus. Wenn der Zusammenhang gilt, dass eine Verringerung des Schwellwertes T dazu führt, dass Testdaten eher der Klasse "gleiches Gesicht" zugeordnet werden, dann die sinkt die FN-Rate für kleinere T, gleichzeitig steigt aber die FP-Rate. Dieses wird in einem Diagramm dargestellt, in dem das Verhalten von FP-Rate $R_{FP}(T)$ und $R_{FN}(T)$ in Abhängigkeit des Schwellwertes T dargestellt wird. Anhand des Schnittpunktes von FP-Rate und FN-Rate kann der Schwellwert T ermittelt werden, der zur Bestimmung der EER verwendet werden muss. Je niedriger die EER, desto besser ist das System. Typischerweise wird der Schwellwert auf den Wertebereich $T \in [0, 1]$ normiert, welches die relative Positi-

on zwischen maximal und minimal vorkommenden absoluten Schwellwert beschreibt.

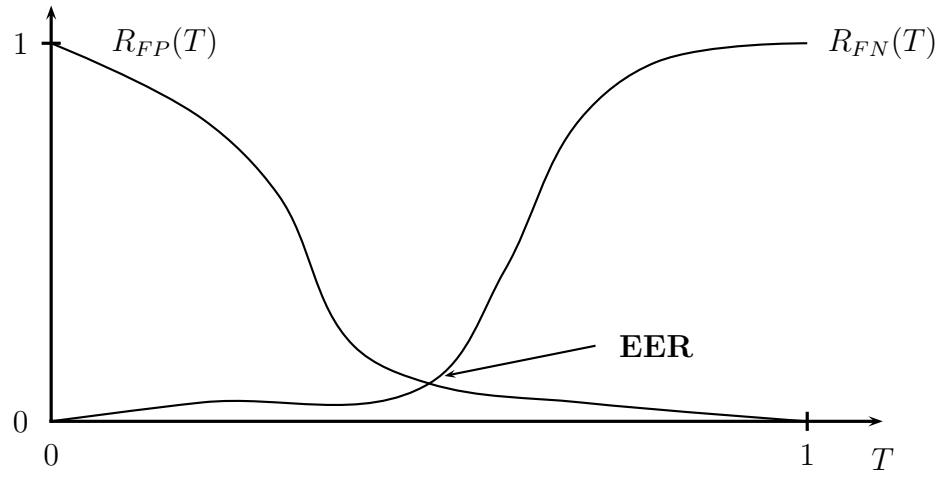


Abbildung A.2: Bestimmung der EER durch Veränderung des Schwellwertes T. Die sich zueinander gegenläufig verhaltenden FP-Rate R_{FP} und FN-Rate R_{FN} schneiden sich bei der EER.

B

Bestimmung von geometrischen Transformationen

Das Lösen überbestimmter linearer Gleichungssysteme zur Bestimmung von geometrischen Transformationen spielt eine erhebliche Rolle bei vielen Ausrichtungsschritten. Im Rahmen dieser Arbeit ist der 2D-Ausrichtungsfall Teil der Formmodelle der Active Shape und Active Appearance Modelle, der 3D-Ausrichtungsfall ist das entscheidende Element des Iterative Closest Point Algorithmus. In beiden Fällen wird gefordert, dass die Transformation eine formerhaltende Transformation (engl. "rigid body transformation") ist.

Eine allgemeine Formulierung des zu lösenden Problems lautet:

$$A = s \cdot R \cdot B + \vec{t} \tag{B.1}$$

wobei A und B jeweils die Koordinaten bekannter Punktmengen sind, bei ASM und AAM sind dies die Koordinaten der Labelpunkte, beim ICP sind dies die in der jeweiligen Iteration durch das "Nächster Nachbar"-Kriterium ermittelten Punktpaare. s ist ein Skalierungsfaktor, R eine Rotationsmatrix und $\vec{t}$ ein Verschiebungsvektor.

$$A = (\vec{a}_1, .., \vec{a}_N) \tag{B.2}$$

$$B = (\vec{b}_1, .., \vec{b}_N) \tag{B.3}$$

Durch die zu schätzenden Transformation wird also $\vec{a}_1$ möglichst nahe zu $\vec{b}_1$, $\vec{a}_2$ möglichst nahe zu $\vec{b}_2$ usw., abgebildet. Gesucht ist die Lösung, die den mitteleren quadratischen Abstand zwischen A und $sRB + \vec{t}$ minimiert.

B.1 Transformation in der Bildebene

Wenn man die Transformation für den 2-Dimensionalen Fall ausschreibt, so erhält man als vereinfachtes parametrisches Gleichungssystem:

$$\begin{pmatrix} a_{x,1} & \dots & a_{x,N} \\ a_{y,1} & \cdots & a_{y,N} \end{pmatrix} = \begin{pmatrix} c & -d \\ d & c \end{pmatrix} \cdot \begin{pmatrix} b_{x,1} & \dots & b_{x,N} \\ b_{y,1} & \cdots & b_{y,N} \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{B.4}$$

Durch t_x, t_y werden dabei die translatorischen Anteile beschrieben, c beschreibt einen einheitlichen Skalierungsfaktor und d den rotatorischen Anteil der Transformation. Durch die gegebene parametrische Form werden allerdings auch Scherungen zugelassen, dieses wird im Folgenden zur Vereinfachung vernachlässigt.

Es gilt die Transformationsparameter zu finden, die die mittlere quadratische Abweichung minimieren.

$$E(c,d,t_x,t_y) = \frac{1}{N}\sum_{i=1}^{N}(cb_{x,i} - db_{y,i} + t_x - a_{x,i})^2 + (db_{x,i} + cb_{y,i} + t_y - a_{y,i})^2 \tag{B.5}$$

Durch Bilden der partiellen Ableitungen und Gleichsetzen mit Null kommt man zu:

$$\begin{array}{rcl} c(\frac{1}{N}\sum b_{x,i}^2 + \frac{1}{N}\sum b_{y,i}^2) + t_x\frac{1}{N}\sum b_{x,i} + t_y\frac{1}{N}\sum b_{y,i} & = & \frac{1}{N}\sum b_{x,i}a_{x,i} + \frac{1}{N}\sum b_{y,i}a_{y,i} \\ d(\frac{1}{N}\sum b_{x,i}^2 + \frac{1}{N}\sum b_{y,i}^2) + t_y\frac{1}{N}\sum b_{x,i} - t_x\frac{1}{N}\sum b_{y,i} & = & \frac{1}{N}\sum b_{x,i}a_{y,i} - \frac{1}{N}\sum b_{y,i}a_{x,i} \\ c\frac{1}{N}\sum b_{x,i} - d\frac{1}{N}\sum b_{y,i} + t_x & = & \frac{1}{N}\sum a_{x,i} \\ d\frac{1}{N}\sum b_{x,i} + c\frac{1}{N}\sum b_{y,i} + t_y & = & \frac{1}{N}\sum a_{y,i} \end{array} \tag{B.6}$$

Um die Berechnung zu vereinfachen, wird davon ausgegangen, dass die Punktwolke B zuerst so verschoben wurde, dass der Schwerpunkt im Koordinatenursprung liegt, sie also mittelwertfrei ist, somit $\frac{1}{N}\sum b_{x,i} = 0$ und $\frac{1}{N}\sum b_{y,i} = 0$. Dadurch ergibt sich:

$$\begin{array}{rcl} t_x & = & \frac{1}{N}\sum a_{x,i} \\ t_y & = & \frac{1}{N}\sum a_{y,i} \end{array} \tag{B.7}$$

und

$$\begin{array}{rcl} c & = & (\frac{1}{N}\sum a_{x,i}\sum b_{x,i} + \frac{1}{N}\sum a_{y,i}b_{y,i})/(\frac{1}{N}\sum a_{x,i}^2 + \frac{1}{N}\sum a_{y,i}^2) \\ d & = & (\frac{1}{N}\sum a_{x,i}\sum b_{x,i} - \frac{1}{N}\sum a_{y,i}b_{y,i})/(\frac{1}{N}\sum a_{x,i}^2 + \frac{1}{N}\sum a_{y,i}^2) \end{array} \tag{B.8}$$

Falls die ursprüngliche Punktwolke B nicht mittelwertfrei ist, gilt es die Verschiebung, die dazu nötig wäre, in der finalen Lösung zu berücksichtigen, dass bedeutet, die Verschiebung in den Ursprung vor Durchführung dieser Rechnung durchzuführen und das Ergebnis der Rechnung zurückzuverschieben.

B.2 Bestimmung der 3D Transformation durch Einheitsquaternionen

Ausgehend von der Transformation

$$A = s \cdot R(B) + \vec{t} \tag{B.9}$$

sollen nun für eine Transformation im 3D die Rotation R, der Skalierungsfaktor s und die Translation $\vec{t}$ gefunden werden. Im folgenden wird die Berechnungsmethode von Horn unter Verwendung von Einheitsquaternionen erläutert, wie sie in der Literatur beschrieben wurde (siehe [Hor87]).

Ohne eine bestimmte Formulierung der Rotation R zu verwenden, gilt im allgemeinen, dass eine Rotation eine lineare Operation ist, und Längen erhält, so dass

$$||R(\vec{b})||^2 = ||\vec{b}||^2 \tag{B.10}$$

dabei ist $||\vec{b}||^2 = \vec{b}\vec{b}$ das Quadrat der Länge von $\vec{b}$ oder anders formuliert das Skalarprodukt mit sich selbst. Da es sich bei den aufeinander abzubildenden Daten nicht um perfekt gleiche Daten handelt, wird es nicht möglich sein eine Transformation zu finden, die diese Gleichung exakt für jeden Punkt löst. Es wird also wiederum, ähnlich wie im 2D-Fall, mit einer Minimierung des quadratischen Fehlers E gearbeitet.

$$E = \sum_{i=1}^{N} ||\vec{a}_i - (s \cdot R(\vec{b}_i) + \vec{t})||^2 \tag{B.11}$$

Im Folgenden werden Überlegungen angestellt, wie zunächst Translation und Skalierung ermittelt werden können, anschließend wird die Rotation betrachtet.

B.2.1 Schwerpunkte der Punktwolken

Es hat sich herausgestellt, dass es günstig ist alle Berechnungen auf die Schwerpunkte $\overline{\vec{a}}$ und $\overline{\vec{b}}$ der Punktwolken A und B zu beziehen, die definiert sind, durch

$$\overline{\vec{a}} = \frac{1}{N} \sum_{i=1}^{N} \vec{a}_i \tag{B.12}$$

$$\overline{\vec{b}} = \frac{1}{N} \sum_{i=1}^{N} \vec{b}_i \tag{B.13}$$

Anschließend werden alle Daten als mittelwertfreie Daten notiert:

$$A' = (\vec{a}_1 - \overline{\vec{a}}, \cdots, \vec{a}_N - \overline{\vec{a}}) \tag{B.14}$$

$$B' = (\vec{b}_1 - \overline{\vec{b}}, \cdots, \vec{b}_N - \overline{\vec{b}}) \tag{B.15}$$

Dadurch kann die Kostenfunktion umgeschrieben werden zu:

$$E = \sum_{i=1}^{N} ||\vec{a'}_i - (s \cdot R(\vec{b'}_i) + \vec{t'})||^2 \tag{B.16}$$

mit

$$\vec{t'} = \vec{t} - \overline{\vec{a}} + sR(\overline{\vec{b}}). \tag{B.17}$$

Dieses ergibt sich zu:

$$\begin{aligned} E &= \sum_{i=1}^{N} ||\vec{a'}_i - (s \cdot R(\vec{b'}_i) + \vec{t'})||^2 \\ &= \sum_{i=1}^{n} ||\vec{a'}_i - sR(\vec{b'}_i)||^2 - 2\vec{t'} \sum_{i=1}^{N} (\vec{a'}_i - R(\vec{b'})) + N||\vec{t'}||^2. \end{aligned} \tag{B.18}$$

Der mittlere Term ergibt Null, da die Daten mittelwertfrei sind, und die Rotation ja längenerhaltend ist. Also bleiben nur der erste und der dritte Term übrig. Der Erste ist unabhängig von $\vec{t'}$ und der letzte Term kann nicht negativ werden. Diese Kostenfunktion kann also offensichtlich bezüglich der Translation mit $\vec{t'} = \vec{0}$ oder unter Berücksichtigung von Gleichung B.17 mit

$$\vec{t} = \overline{\vec{a}} - sR(\overline{\vec{b}}) \tag{B.19}$$

minimiert werden. Daraus ergibt sich, das die Translation die Differenz zwischen Schwerpunkt von A und skaliertem und rotiertem Schwerpunkt von B ist. Nachdem später die Skalierung und Rotation bestimmt worden ist, wird zu dieser Gleichung B.19 zurückgekehrt.

B.2.2 Berechnung des Skalierungsfaktors

Wenn also weiterhin von mittelwertfreien Daten ausgegangen wird, ist der verbliebene zu minimierende Fehlerterm (siehe Gleichung B.18)

$$\sum_{i=1}^{N} ||\vec{a'} - sR(\vec{b'})||^2. \tag{B.20}$$

Durch Umformen unter der Berücksichtigung, dass $||R(\vec{b'})||^2 = ||\vec{b'}||^2$ erhält man

$$\sum_{i=1}^{n} ||\vec{a'}_i||^2 - 2s \sum_{i=1}^{N} \vec{a'} \cdot R(\vec{b'}) + s^2 \sum_{i=1}^{N} ||\vec{b'}||^2 \tag{B.21}$$

oder in anderer Schreibweise

$$S_A - 2sD + s^2 S_B \tag{B.22}$$

wobei S_A und S_B die Summen der Quadrate der Koordinatenvektoren (relativ zum Schwerpunkt) sind, und D das Skalarprodukt zwischen den jeweiligen Koordinatenvektoren von A und den rotierten Koordinatenvektoren von B. Durch Berechnung der quadratischen Ergänzung und Umformen kommt man zu

$$\begin{aligned}(s\sqrt{S_B})^2 - 2s\sqrt{S_B}\frac{D}{\sqrt{S_b}} + (\frac{D}{\sqrt{S_B}})^2 - (\frac{D}{\sqrt{S_B}})^2 + S_A \\ = (s\sqrt{S_B} - \frac{D}{\sqrt{S_B}})^2 + \frac{(S_A S_B - D^2)}{\sqrt{S_B}}\end{aligned} \tag{B.23}$$

Dieser Ausdruck wird bzgl. s dadurch minimiert, dass der erste Term Null wird, also $s = \frac{D}{\sqrt{S_B}}$, somit ergibt sich für die Skalierung s

$$s = \frac{\sum_{i=1}^{N} \vec{a'}_i \cdot R(\vec{b'}_i)}{\sum_{i=1}^{N} ||\vec{b'}_i||^2} \tag{B.24}$$

Symmetrie der Skalierung

Wenn man, anstelle vom Finden der besten Parameter für die Transformation

$$A = s \cdot R(B) + \vec{t} \tag{B.25}$$

von der inversen Transformation

$$B = \underline{s} \cdot \underline{R}(A) + \underline{\vec{t}} \tag{B.26}$$

ausgeht, erhofft man sich eigentlich auch die inversen Transformationsparameter zu finden.

$$\underline{s} = 1/s, \underline{\vec{t}} = -\frac{1}{s} R^{-1}(\vec{t}), \underline{R} = R^{-1} \tag{B.27}$$

Dies wird allerdings nach den bisherigen Erkenntnissen nicht erreicht, da ja dann $\underline{s} = \underline{D}/S_A$ oder

$$s = \frac{\sum_{i=1}^{N} \vec{b'}_i \cdot \underline{R}(\vec{a'}_i)}{\sum_{i=1}^{N} ||\vec{a'}_i||^2} \tag{B.28}$$

wäre, welches ungleich $1/s$ ist. Wenn man davon ausgeht, dass die Meßgenauigkeit in beiden Punktwolken etwa gleich ist, wäre es sinnvoll eine symmetrische Formulierung für die Fehlerfunktion zu wählen:

$$E = \sum_{i=1}^{N} ||\frac{1}{\sqrt{s}} \vec{a'}_i - \sqrt{s} R(\vec{b'}_i)||^2 \tag{B.29}$$

dann ergibt sich daraus

$$\frac{1}{s}\sum_{i=1}^{N}||\vec{a'}_i||^2 - 2\sum_{i=1}^{N}\vec{a'}_i \cdot R(\vec{b'}_i) + s\sum_{i=1}^{N}||\vec{b'}_i||^2 \tag{B.30}$$

oder anders ausgedrückt

$$\frac{1}{s}S_A - 2D + sS_B \tag{B.31}$$

Durch Umformen mittels quadratischer Ergänzung würde man nun

$$(\sqrt{s}\sqrt{S_B} - \frac{1}{\sqrt{s}}\sqrt{S_A})^2 + 2(\sqrt{S_A S_B} - D) \tag{B.32}$$

erhalten. Dies wird in Hinblick auf den Skalierungsfaktor s minimiert, wenn der erste Term Null wird, oder $s = \sqrt{S_A/S_B}$, welches sich zu

$$s = \sqrt{\frac{\sum_{i=1}^{N}||\vec{a'}_i||^2}{\sum_{i=1}^{N}||\vec{b'}_i||^2}} \tag{B.33}$$

ergibt. Ein großer Vorteil dieser Herangehensweise ist es, dass sie erlaubt, den Skalierungsfaktor ohne Wissen über die Rotation zu ermitteln. Weiterhin zeigt sich die wichtige Eigenschaft, dass die Bestimmung der Rotation nicht durch die Wahl des Skalierungsfaktors beeinflusst wird. In beiden Formulierungen B.23 und B.32 wird die Kostenfunktion dadurch minimiert, dass D so groß wie möglich wird. Es gilt also die Rotation R zu finden, so dass die Gleichung

$$E = \sum_{i=1}^{N}\vec{a'}_i \cdot R(\vec{b'}_i) \tag{B.34}$$

so groß wie möglich wird.

B.2.3 Repräsentation der Rotation

Es gibt viele mögliche Repräsentationen für Rotation, z.B. Euler Winkel [BL91], Gibbs Vektor [Gib91], Cayley-Klein Parameter [Cay59][Kle71], Pauli Spin Matrizen [Pau50], Achsen und Drehwinkel, orthonormale Matrizen und Hamilton's Quaternionen [Ham53]. Von all diesen Repräsentationen werden orthonormale Matrizen am Häufigsten eingesetzt, vor allem in der Photogrammetrie und Robotik. Allerdings gibt es viele Vorteile, die die Notation mittels Einheitsquaternionen bietet. Ein Vorteil ist, dass es wesentlich einfacher ist, einzufordern, dass ein Quaternion einen Betrag von eins hat, als Umzusetzen, dass eine Matrix orthonormal ist. Weiterhin sind Einheitsquaternionen nah an der geometrisch intuitiven Notation durch

Achsen und Drehwinkel.
Im Folgenden wird also die Aufgabe, die Rotation zu bestimmen, die

$$E = \sum_{i=1}^{N} \vec{a'}_i \cdot R(\vec{b'}_i) \tag{B.35}$$

maximiert, durch Verwendung von Einheitsquaternionen gelöst. Falls notwendig, kann aus den ermittelten Quaternionen dann eine orthonormale Rotationsmatrix konstruiert werden. Als erstes wird im Folgenden der Begriff der Einheitsquaternionen eingeführt und erläutert.

Quaternionen

Ein Quaternion kann man sich entweder als einen Vektor mit 4 Dimensionen vorstellen, als eine Komposition aus einem Skalar und einem Raumvektor (3-dimensional) oder als eine komplexe Zahl mit 3 verschiedenen Imaginärteilen. Quaternionen werden im Folgenden durch Symbole mit einem hochgestellten Kreis über ihnen dargestellt. Wenn man der Notation als komplexe Zahl folgt, ist

$$\mathring{q} = q_0 + iq_x + jq_y + kq_z \tag{B.36}$$

ein Quaternion mit Realteil q_0 und drei Imaginärteilen q_x, q_y und q_z. Die Multiplikation von Quaternionen kann definiert werden, als das Produkt seiner Komponenten. Die Imaginärteile und ihre Beziehungen untereinander seien definiert durch

$$\begin{aligned} i^2 = -1, j^2 = -1, k^2 = -1, \\ ij = k, jk = i, ki = j \end{aligned} \tag{B.37}$$

und

$$ji = -k, kj = -i, ik = -j. \tag{B.38}$$

Wenn nun ein zweites Quaternion $\mathring{r}$ vorliegt,

$$\mathring{r} = r_0 + ir_x + jr_y + kr_z \tag{B.39}$$

ist das Produkt

$$\begin{aligned} \mathring{r}\mathring{q} = (r_0q_0 - r_xq_x - r_yq_y - r_zq_z) \\ +i(r_0q_x + r_xq_0 + r_yq_z + r_zq_y) \\ +j(r_0q_y - r_xq_z + r_yq_0 + r_zq_x) \\ +k(r_0q_z + r_xq_y - r_yq_x + r_zq_0). \end{aligned} \tag{B.40}$$

Das Produkt $\mathring{q}\mathring{r}$ hat eine ähnliche Form, aber an sechs Stellen dieses Terms andere Vorzeichen. Somit gilt $\mathring{r}\mathring{q} \neq \mathring{q}\mathring{r}$. Wenn man sich nun die Gleichung B.40 in einer Matrix-Vektor Schreibweise notiert, erhält man

$$\mathring{r}\mathring{q} = \begin{pmatrix} r_0 & -r_x & -r_y & -r_z \\ r_x & r_0 & -r_z & r_y \\ r_y & r_z & r_0 & -r_x \\ r_z & -r_y & r_x & r_0 \end{pmatrix} \mathring{q} = \mathbb{R}\mathring{q} \tag{B.41}$$

und für $\mathring{q}\mathring{r}$ erhält man

$$\mathring{q}\mathring{r} = \begin{pmatrix} r_0 & -r_x & -r_y & -r_z \\ r_x & r_0 & r_z & -r_y \\ r_y & -r_z & r_0 & r_x \\ r_z & r_y & -r_x & r_0 \end{pmatrix} \mathring{q} = \underline{\mathbb{R}}\mathring{q}. \tag{B.42}$$

Der Unterschied zwischen $\mathbb{R}$ und $\underline{\mathbb{R}}$ liegt in einer Transponierung der rechten unteren 3×3 Submatrix. Die Summe der Quadrate der Elemente jeder Spalte ist immer

$$r_0^2 + r_x^2 + r_y^2 + r_z^2, \tag{B.43}$$

also das Skalarprodukt $\mathring{r} \cdot \mathring{r}$.

Skalarprodukt von Quaternionen

Das Skalarprodukt zweier Quaternionen ist die Summe der Produkte der korrespondierenden Komponenten:

$$\mathring{p} \cdot \mathring{q} = p_0 q_0 + p_x q_x + p_y q_y + p_z q_z. \tag{B.44}$$

Das Betragsquadrat eines Quaternions ist das Skalarprodukt mit sich selbst:

$$||\mathring{q}||^2 = \mathring{q} \cdot \mathring{q}. \tag{B.45}$$

Ein Einheitsquaternion ist ein Quaternion, dessen Betrag eins ist. Das konjugierte Quaternion negiert die Imäginarteile, ist also

$$\mathring{q}^* = q_0 - iq_x - jq_y - kq_z. \tag{B.46}$$

Die Produktschreibweise als 4×4 Matrix (siehe Gleichungen B.41 und B.42) des konjugierten Quaternions $\mathring{q}^*$ sind die Transponierten der Produktmatrizen des Quaternions $\mathring{q}$ selbst. Da die Matrizen orthogonal sind, ergeben sich die Produkt mit den Transponierten zu einer Diagonalmatrix, $QQ^T = \mathring{q} \cdot \mathring{q}I$ mit I als 4×4 Einheitsmatrix. Daraus ergibt sich, dass das Produkt $\mathring{q}\mathring{q}^*$ real ist und dem Skalarprodukt eines Quaternions mit sich selbst, also dem Betragsquadrat, entspricht :

$$\mathring{q}\mathring{q}^* = (q_0^2 + q_x^2 + q_y^2 + q_z^2) = \mathring{q}\mathring{q}^* = \mathring{q} \cdot \mathring{q}. \tag{B.47}$$

Daraus ergibt sich, dass für jedes Quaternion, das einen Betrag ungleich Null hat, eine Inverse existiert:

$$\mathring{q}^{-1} = \frac{1}{\mathring{q} \cdot \mathring{q}} \mathring{q}^*. \tag{B.48}$$

Im Falle des Einheitsquaternions, wenn der Betrag also eins ist, ist die Inverse die Konjugierte.

Eigenschaften von Quaternionenprodukten

Skalarprodukte bleiben erhalten, da die Produktmatrizen orthogonal sind, also ist

$$(\mathring{q}\mathring{p}) \cdot (\mathring{q}\mathring{r}) = (Q\mathring{p}) \cdot (Q\mathring{r}) = (Q\mathring{p})^T (Q\mathring{r}) \tag{B.49}$$

und

$$(Q\mathring{p})^T (Q\mathring{r}) = \mathring{p}^T Q^T Q\mathring{r} = \mathring{p}^T (\mathring{q} \cdot \mathring{q}) I\mathring{r}. \tag{B.50}$$

Daraus ergibt sich, dass

$$(\mathring{q}\mathring{p}) \cdot (\mathring{q}\mathring{r}) = (\mathring{q} \cdot \mathring{q})(\mathring{p} \cdot \mathring{r}). \tag{B.51}$$

Im dem Fall, dass $\mathring{q}$ ein Einheitsquaternion ist, vereinfacht sich dies weiter zu $(\mathring{p} \cdot \mathring{r})$. Ein weiterer Spezialfall ist

$$(\mathring{p}\mathring{q}) \cdot (\mathring{p}\mathring{q}) = (\mathring{p} \cdot \mathring{p})(\mathring{q} \cdot \mathring{q}), \tag{B.52}$$

welches bedeutet, dass der Betrag eines Produktes genau das Produkt der Beträge ist. Weiterhin ergibt sich

$$(\mathring{p}\mathring{q}) \cdot \mathring{r} = \mathring{p} \cdot (\mathring{r}\mathring{q}^*), \tag{B.53}$$

dieses wird sich später als nützlich erweisen.

3D-Orts- und Richtungsvektoren können durch rein imaginäre Quaternionen dargestellt werden. Wenn $\vec{r} = (x, y, z)^T$, dann kann das passenden Quaternion als

$$r = 0 + ix + jy + kz \tag{B.54}$$

dargestellt werden. Im Falle eines rein Imaginären Quaternions sind die Produktmatrizen schiefsymmetrisch. Das heist in diesem Spezialfall sind

$$\mathbb{R}^T = -\mathbb{R}, \underline{\mathbb{R}}^T = -\underline{\mathbb{R}}. \tag{B.55}$$

Einheitsquaternionen und Rotation

Die Länge eines Vektors ändert sich nicht durch Rotation, genauso bleibt der Winkel zwischen zwei Vektoren konstant. Das bedeutet, Rotation erhält die Ergebnisse von Skalarprodukten. Dieses bedeutet, dass Rotation durch Quaternionen repräsentiert werden kann, wenn ein Weg gefunden wird, von rein imaginären Quaternionen auf rein imaginäre Quaternionen abzubilden, unter der Vorraussetzung, dass Skalarprodukte erhalten bleiben.
Es wurde bereits festgestellt, dass die Multiplikation mit einem Einheitsquaternion das Skalarprodukt erhält, siehe Gleichung B.51, also dass

$$(\mathring{q}\mathring{p}) \cdot (\mathring{q}\mathring{r}) = (\mathring{p} \cdot \mathring{r}) \tag{B.56}$$

unter der Vorraussetzung, dass $\mathring{q} \cdot \mathring{q} = 1$. Diese einfache Multiplikation selbst kann aber nicht als Repräsentation für die Rotation dienen, da das Produkt zwischen

Einheitsquaternion und rein imaginärem Quaternion nicht unbeding wieder ein rein imaginäres Quaternion als Ergbnis hervorruft. Deshalb wird das zusammengesetzte Produkt

$$\mathring{r}' = \mathring{q}\mathring{r}\mathring{q}^* \tag{B.57}$$

benutzt, dessen Ergebnis sicher rein imaginär ist. Das kann durch Erweiterung dieses Ausdrucks gezeigt werden:

$$\mathring{q}\mathring{r}\mathring{q}^* = (Q\mathring{r})\mathring{q}^* = \underline{Q}^T(Q\mathring{r}) = (\underline{Q}^T Q)\mathring{r}, \tag{B.58}$$

hier sind Q und $\underline{Q}$ die 4×4 Matrizen zur Darstellung des Produktes mit $\mathring{r}$ (siehe B.41 und B.42). Durch Ausmultiplizieren von $\underline{Q}^T Q$ erhält man

$$\underline{Q}^T Q = \begin{pmatrix} \mathring{q} \cdot \mathring{q} & 0 & 0 & 0 \\ 0 & (q_0^2 + q_x^2 - q_y^2 - q_z^2) & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 0 & 2(q_y q_x + q_0 q_z) & (q_0^2 - q_x^2 + q_y^2 - q_z^2) & 2(q_y q_z - q_0 q_x) \\ 0 & 2(q_z q_x + q_0 q_y) & 2(q_z q_y + q_0 q_x) & (q_0^2 - q_x^2 - q_y^2 - q_z^2) \end{pmatrix}. \tag{B.59}$$

Daraus ergibt sich, dass $\mathring{r}'$ rein imaginär ist, wenn $\mathring{r}$ rein imaginär ist. Q und $\underline{Q}$ sind orthonormal, wenn $\mathring{q}$ ein Einheitsquaternion ist. Dann ergibt sich $\mathring{q} \cdot \mathring{q} = 1$ und die rechte untere 3×3 Submatrix von $\underline{Q}^T Q$ ist ebenfalls orthonormal. Bei genauerer Betrachtung sieht man, dass dies eine Rotationsmatrix R ist, die den Ortsvektor $\vec{r}$ auf den Ortsvektor $\vec{r}'$ abbildet.

$$\vec{r}' = R\vec{r} \tag{B.60}$$

Weiterhin ist interessant, dass

$$(-\mathring{q})\mathring{r}(-\mathring{q}^*) = \mathring{q}\mathring{r}\mathring{q}^* \tag{B.61}$$

also repräsentiert $-\mathring{q}$ die gleiche Rotation wie $\mathring{q}$.

Finden der besten Rotation

Es wird nun zu dem ursprünglichen Problem zurückzugekehrt. Es gilt das Quaternion $\mathring{q}$ zu finden, dass

$$\sum_{i=1}^{N} (\mathring{q}\mathring{b'}_i\mathring{q}^*) \cdot \mathring{a'}_i \tag{B.62}$$

minimiert, in diesem Fall sind die oben verwendeten mittelwertfreien Ortsvektoren $\vec{a'}_i$ und $\vec{b'}_i$ in Quaternionenrepräsentation $\mathring{a'}_i$ und $\mathring{b'}_i$ notiert worden. Mit den hergeleiteten Erkenntnissen kann dieses zu

$$\sum_{i=1}^{N} (\mathring{q}\mathring{b'}_i) \cdot (\mathring{a'}_i\mathring{q}) \tag{B.63}$$

umgeformt werden. In Matrizenschreibweise ist

$$\mathring{q}\mathring{b'}_i = \begin{pmatrix} 0 & -b'_{x,i} & -b'_{y,i} & -b'_{z,i} \\ b'_{x,i} & 0 & b'_{z,i} & -b'_{y,i} \\ b'_{y,i} & -b'_{z,i} & 0 & b'_{x,i} \\ b'_{z,i} & b'_{y,i} & -b'_{x,i} & 0 \end{pmatrix} \mathring{q} = \underline{B}_i \mathrm{q} \tag{B.64}$$

und

$$\mathring{a'}_i\mathring{q} = \begin{pmatrix} 0 & -a'_{x,i} & -a'_{y,i} & -a'_{z,i} \\ a'_{x,i} & 0 & -a'_{z,i} & a'_{y,i} \\ a'_{y,i} & a'_{z,i} & 0 & -a'_{x,i} \\ a'_{z,i} & -a'_{y,i} & a'_{x,i} & 0 \end{pmatrix} . \mathring{q} = A_i \mathrm{q} \tag{B.65}$$

Hier sind A_i und $\underline{B}_i$ schiefsymmetrisch und orthogonal. Die Summe, die zu maximieren ist ergibt sich zu

$$\begin{aligned} &\sum_{i=1}^{N} (\underline{B}_i \mathring{q}) \cdot (A_i \mathring{q}) \\ &= \sum_{i=1}^{N} \mathring{q}^T B_i^T A_i \mathring{q} \\ &= \mathring{q}^T (\sum_{i=1}^{N} B_i^T A_i) \mathring{q} \\ &= \mathring{q}^T (\sum_{i=1}^{N} N_i) \mathring{q} = \mathring{q}^T N \mathring{q} \end{aligned} \tag{B.66}$$

mit $N_i = \underline{B}_i^T A_i$ und $N = \sum_{i=1}^{N} N_i$. Jedes N_i ist symmetrisch, damit ist N auch symmetrisch.

Wenn wir nun also die Matrix M einführen, mit

$$M = \sum_{i=1}^{N} \vec{b'} \vec{a'}^T, \tag{B.67}$$

dann sind dessen Elemente die Summen der Produkte der Koordinaten in der Punktwolke B mit den Koordinaten in der Punktwolke A. Diese Matrix enthält alle notwendigen Informationen die benötigt werden, um die Lösung der kleinsten Quadrate zu finden. Wenn die einzelnen Elemente von M wie folgt notiert werden

$$M = \begin{pmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{pmatrix}, \tag{B.68}$$

mit

$$S_{xx} = \sum_{i=1}^{N} b'_{x,i} a'_{x,i}, S_{xy} = \sum_{i=1}^{N} b'_{x,i} a'_{y,i}, .. \tag{B.69}$$

und so weiter, dann ergibt sich N zu

$$N = \begin{pmatrix} S_{xx}+S_{yy}+S_{zz} & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx} \\ S_{yz}-S_{zy} & S_{xx}-S_{yy}-S_{zz} & S_{xy}+S_{yx} & S_{zx}+S_{xz} \\ S_{zx}-S_{xz} & S_{xy}-S_{yx} & -S_{xx}+S_{yy}-S_{zz} & S_{yz}+S_{zy} \\ S_{xy}-S_{yx} & S_{zx}+S_{xz} & S_{yz}-S_{zy} & -S_{xx}-S_{yy}+S_{zz} \end{pmatrix} \tag{B.70}$$

Die 10 von einander unabhängigen Elemente von N lassen sich also aus Summen und Differenzen der Elemente von M darstellen. Die Spur der Matrix N, also die Summe der Diagonalelemente ist Null.

Maximierung des Matrixproduktes durch Finden der Eigenvektoren

Im Folgenden wird gezeigt, dass das Quaternion $\mathring{q}$, welches $\mathring{q}^T N \mathring{q}$ maximiert, der Eigenvektor von N ist, der den größten positiven Eigenwert hat. Daher wird die Gleichung $\det(N - \lambda I) = 0$ gelöst, um die Eigenwerte zu bestimmen. Anschließend wird der zum größten Eigenwert λ_m gehörige Eigenvektor $\mathring{e}_m$ durch Lösen der homogenen Gleichung

$$(N - \lambda_m I)\mathring{e}_m = 0 \tag{B.71}$$

bestimmt. Da die symmetrische 4×4 Matrix N vier reelle Eigenwerte, $\lambda_1, .., \lambda_4$ hat, können damit vier zueinander orthogonale Eigenvektoren $\mathring{e}_1, .., \mathring{e}_4$ bestimmt werden, so dass

$$N\mathring{e}_i = \lambda_i \mathring{e}_i \tag{B.72}$$

Diese Eigenvektoren spannen einen vierdimensionalen Raum auf, in dem jedes beliebige Quaternion $\mathring{q}$ als Linearkombination der Form

$$\mathring{q} = \alpha_1 \mathring{e}_1 + \alpha_2 \mathring{e}_2 + \alpha_1 \mathring{e}_3 + \alpha_4 \mathring{e}_4 \tag{B.73}$$

dargestellt werden kann. Da alle Eigenvektoren zueinander othogonal sind, ergibt sich, dass

$$\mathring{q} \cdot \mathring{q} = \alpha_1^2 + \alpha_2^2 + \alpha_3^2 + \alpha_4^2 \tag{B.74}$$

Da $\mathring{q}$ ein Einheitsquaternion sein soll, muss dies eins sein. Weiterhin ergibt sich, dass

$$N\mathring{q} = \alpha_1 \lambda_1 \mathring{e}_1 + \alpha_2 \lambda_2 \mathring{e}_2 + \alpha_3 \lambda_3 \mathring{e}_3 + \alpha_4 \lambda_4 \mathring{e}_4 \tag{B.75}$$

da es sich bei den Vektoren $\mathring{e}$ um Eigenvektoren von N handelt. Daraus folgt, dass

$$\mathring{q}^T N \mathring{q} = \mathring{q} \cdot (N\mathring{q}) = \alpha_1^2 \lambda_1 + \alpha_2^2 \lambda_2 + \alpha_3^2 \lambda_3 + \alpha_4^2 \lambda_4 \tag{B.76}$$

Wenn wir annehmen, dass die Eigenwerte absteigend nach ihrer Größe sortiert wurden, $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4$ dann sieht man, dass

$$\mathring{q}^T N \mathring{q} \leq \alpha_1^2\lambda_1 + \alpha_2^2\lambda_2 + \alpha_3^2\lambda_3 + \alpha_4^2\lambda_4 = \lambda_1. \tag{B.77}$$

Da ja beim Einheitsquaternion die Summe aller α^2 eins ergeben muss, ist dieser Ausdruck maximal, wenn $\alpha_1 = 1$ und $\alpha_2 = \alpha_3 = \alpha_4 = 0$. Also ist diejenige Rotation optimal, die durch das Einheitsquaternion beschrieben wird, welches der Eigenvektor mit dem größten Eigenwert der Matrix N ist. Wenn dieser bestimmt wurde, kann die Rotationsmatrix R durch Einsetzen in Gleichtung B.59 und Ausschneiden der linken unteren 3×3 Submatrix ermittelt werden:

$$R = \begin{pmatrix} (q_0^2 + q_x^2 - q_y^2 - q_z^2) & 2(q_x q_y - q_0 q_z) & 2(q_x q_z + q_0 q_y) \\ 2(q_y q_x + q_0 q_z) & (q_0^2 - q_x^2 + q_y^2 - q_z^2) & 2(q_y q_z - q_0 q_x) \\ 2(q_z q_x + q_0 q_y) & 2(q_z q_y + q_0 q_x) & (q_0^2 - q_x^2 - q_y^2 - q_z^2) \end{pmatrix} \tag{B.78}$$

Da also nun Skalierung (Gleichung B.33) und Rotation ermittelt werden können, kann abschliessend durch Einsetzen in Gleichung B.19 die Translation ermittelt werden, damit ist die Bestimmung der geometrischen Transfomation vollständig.

C

Grundlagen neuronaler Netze

Die Grundlagen der im Rahmen dieser Arbeit verwendeten Neuronalen Netze werden im Folgenden vorgestellt. Ausgegangen wird dabei zunächst von einem einzelnen Neuron, das einen Eingabevektor in einen Ausgabewert umwandelt. Durch paralleles Aneinanderreihen solcher Neuronen entsteht eine Schicht von künstlichen Neuronen. Diese wandelt einen Eingabevektor in einen Ausgabevektor um. Jedes Neuron wird dabei durch eine Neuronenfunktion, auch Aktivierungsfunktion genannt, definiert, die den gesamten Eingabevektor als Parameter erhält. Durch Reihenschaltung mehrerer solcher Neuronenschichten, der Ausgangsvektor einer Schicht dient dabei als Eingangsvektor der nächsten Schicht, erhält man das sogenannte Multi-Layer-Perzeptron (MLP).

C.1 Lernregel für das einschichtige Perzeptron

Das einlagige oder einschichtige Perzeptron (engl. "single-layer perceptron")[Ros58] ist ein Neuronales Netz, welches eine Eingabeschicht direkt auf eine Ausgabeschicht abbildet, es besteht also nur aus Input-Neuronen der Eingabeschicht und Output-Neuronen der Ausgabeschicht. Als Output-Neuronen dienen hier sogenannte Skalarprodukt-Neuronen, also solche, die den Eingangsvektor durch Skalarprodukt mit einem Gewichtsvektor auf einen skalaren Ausgabewert abbilden. Beim Lernen für das einlagige Perzeptron gilt es, die Gewichtungen der Eingangswerte der Output-Neuronen so anzupassen, dass für gegebene Eingangsvektoren $\vec{x}$ die dazugehörigen Ausgangsvektoren $\vec{t}$ erreicht werden. Die dabei verwendeten Ein- und Ausgangsvektoren bezeichnet man dabei als Trainingsdaten. Es gilt alle Gewichte w_{ij}, also die Gewichte zwischen i-ten Input-Neuron x_i und j-ten Output-Neuron, in Bezug auf den tatsächlichen Ausgangswert o_j und gewünschten Ausgangswert t_j anzupassen. Dieses geschieht durch

$$w_{ij}^{neu} = w_{ij}^{alt} + \Delta w_{ij} \tag{C.1}$$

mit

$$\Delta w_{ij} = \alpha(t_j - o_j)x_i. \tag{C.2}$$

Dabei ist $\alpha \in [0, 1]$ die sogenannte Lernrate, die bestimmt, wie intensiv die Änderung in jeder Trainingsiteration ist. Der Trainingsprozess läuft dabei folgendermaßen ab: Zunächst werden alle Gewichte w_{ij} mit zufälligen Werten initialisiert. Danach werden schrittweise, pro Trainingsbeispiel die Werte nach der oben angegeben Formel verändert. Dieses wird mehrfach wiederholt, bis sich die Werte der Gewichte nur noch wenig ändern, welches zum Beispiel durch eine abklingende Lernrate erreicht werden kann.

C.2 Lernen für mehrschichtige Netze

Ein mehrschichtiges neuronales Netz (engl. "multi-layer perceptron") besteht, zusätzlich zu der bereits für das einschichtige Perzeptron eingeführten Input- und Outputschicht, aus einer oder mehreren Hidden-Schichten. Eine Hidden Schicht ist dabei eine Schicht, die zwischen zwei Schichten eingefügt wird, als Eingabewerte erhält sie den Ausgangsvektor der vorherliegenden Schicht, ihr Ausgangsvektor dient als Eingabevektor für die nächste Schicht.
Das Lernen in mehrschichtigen Neuronalen Netzen findet oft durch den Backpropagation Algorithmus [RHW88] statt. Er gehört zu den Methoden des überwachten Lernens. Es wird also wie beim Lernen für das einlagige Perzeptron davon ausgegangen, dass Beispieldaten von Ein- und Ausgangsvektoren zur Verfügung stehen. Hierbei werden Neuronen der ersten Schicht (Input-Layer) als Input-Neuronen bezeichnet, Neuronen der mittleren Schichten (Hidden Layer) als Hidden-Neuronen und Neuronen der Ausgabeschicht (Output Layer) als Output-Neuronen.

Ziel ist es, den mittleren quadratischen Fehler über alle n Trainingsbeispiele zwischen gewünschten Ausgangsvektor $\vec{t}$ und durch das Netz erzeugtem Ausgangsvektor $\vec{o}$ zu minimieren.

$$E = \frac{1}{2}\sum_{i=1}^{n}(\vec{t_i} - \vec{o_i})^2 \tag{C.3}$$

Wenn nur davon ausgegangen wird, dass jedes Neuron j seine Ausgabe o_j dadurch berechnet, dass alle Eingabewerte x_i mit den Gewichten w_{ij} gewichtet und aufsummiert werden, und dieses als Eingangswert für eine differenzierbare Aktivierungsfunktion ϕ verwendet wird, die auf o_j abbildet, kann dieses für einen k-dimensionalen Eingangsvektor wie folgt formuliert werden:

$$o_j = \phi(v_j) \tag{C.4}$$

mit

$$v_j = \sum_{i=1}^{k} w_{ij} x_i. \tag{C.5}$$

Hierbei spielt es zunächst keine Rolle, ob die Eingabewerte x_i eventuelle Ausgaben vorherliegender Schichten sind, oder ob die Ausgaben o_j als Eingabewerte für die nächste Schicht dienen.

Da die Kostenfunktion E im Hinblick auf die Einstellung der Gewichte w_{ij} minimiert werden soll, wird die partielle Ableitung $\frac{\partial E}{\partial w_{ij}}$ gebildet, die sich durch Anwendung der Kettenregel zu

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial o_j} \frac{\partial o_j}{\partial v_j} \frac{\partial v_j}{\partial w_{ij}} \tag{C.6}$$

ergibt. Diese soll in ein Gradientenabstiegsverfahren eingebettet werden, welches die Gewichte w_{ij} jeweils in entgegengesetzer Richtung des steilsten Anstiegs modifiziert.

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} = -\eta \delta_j x_i \tag{C.7}$$

Hierbei ist η eine Lernrate, mit der die Stärke der Gewichtsveränderungen eingestellt werden kann (ähnlich der Schrittweite beim Gradientenabstieg). Hier müssen nun zwei Fälle unterschieden werden:

$$\delta_j = \begin{cases} \phi'(v_j)(t_j - o_j) & \text{falls j ein Output} - \text{Neuron} \\ \phi'(v_j) \sum_k \delta_k w_{jk} & \text{falls j ein Hidden} - \text{Neuron} \end{cases} \tag{C.8}$$

Damit können nun die Fehlersignale δ_k beginnend mit der Ausgangsschicht bestimmt werden, und diese auf die jeweils davorliegende Schicht übertragen werden, um deren Fehlersignale zu bestimmen. Dieses propagiert sich von der Ausgangsschicht des Netzes schrittweise bis zur ersten Schicht des Netzes, daher der Name Backpropagation. Nachdem die Änderung Δw_{ij} aller Gewichte berechnet wurde, kann diese angewendet werden:

$$w_{ij}^{neu} = w_{ij}^{alt} + \Delta w_{ij} \tag{C.9}$$

Die Wahl der Lernrate η ist wichtig für das Verfahren, da ein zu hoher Wert dafür sorgen kann, dass ein Minimum verfehlt wird, ein zu kleiner Wert das Lernen unnötig verlangsamt. Hier gibt es unterschiedliche Ansätze, mit variabler Lernrate zu arbeiten, um die Fehlerminimierung zu beschleunigen. Eine im Rahmen dieser Arbeit eingesetze Modifikation des Standardverfahrens Backpropagation, ist Backpropagation mit Trägheitsterm ("backpropagation with momentum", [HKP91]). Dabei wird ein Trägheitsparameter $a \in [0, 1]$, auch Momentum genannt, verwendet, um in der Gewichtsveränderung des aktuellen Schrittes $t+1$ die des vorhergehenden Iterationsschrittes t mit zu berücksichtigen.

$$\Delta w_{ij}^{t+1} = -(1-a)\eta\delta_j x_i + a\Delta w_{ij}^{t+1} \tag{C.10}$$

Durch den Trägheitsterm werden Probleme der Backpropagation-Regel vermieden. Wenn die Gewichtsveränderungen sehr klein oder sehr groß werden, sorgt der Trägheitsterm dafür, dass das Lernen nicht zu stark abgebremst oder beschleunigt wird. Dies kann dafür sorgen, dass flache Plateaus in der Kostenfunktion schneller überwunden werden können, oder dass steile Schluchten in der Kostenfunktion, und damit große Gradienten, nicht zum Verfehlen des Minimums führen.

C.3 Einsatz von neuronalen Netzen zur Klassifikation

Um ein neuronales Netz zur Klassifikation einzusetzen, definiert man den Ausgangsvektor so, dass für jede Klasse ein Ausgangswert erzeugt wird, der die Klassenzugehörigkeit angibt. In der Praxis wird diese Klassenzugehörigkeit gerne mit dem Wertepaar 0 und 1 oder dem Wertepaar -1 und 1 realisiert. Für ein Klassifikationsproblem mit n Klassen erhält man somit einen n dimensionalen Ausgangsvektor. Trainingsdaten werden nun erzeugt, in dem für die bekannten Beispiele (Eingangsvektoren) der Ausgangswert der zugehörigen Klasse maximal ist (1), der aller anderen Klassen minimal (0 oder -1). Das neuronale Netz wird nun trainiert und in der Anwendungsphase verhält es sich so, dass die Eingangsvektoren auf Ausgangsvektoren abgebildet werden. Die Klassenzugehörigkeit kann nun abgelesen werden, im einfachsten Fall, wird derjenigen Klasse zugeordnet, für die die Klassenzugehörigkeit am größten ist.

Abkürzungsverzeichnis

DAR-pipeline	Detection-Alignment-Recognition-pipeline
AdaBoost	Adaptive Boosting
ASM	Active Shape Modell
ICP	Iterative Closest Point (-Algorithmus)
AAM	Active Appearance Modell
HMM	Hidden Markov Modell
P2D-HMM	Pseudo-Zweidimensionales Hidden Markov Modell
PCA	Principal Component Analysis, Hauptkomponentenanalyse
FN	False Negative, Falsch Negativ
FP	False Positive, Falsch Positiv
TN	True Negative, Wahr Negativ
TP	True Positive, Wahr Positiv
EER	Equal Error Rate
SAFEE	Security of Aircraft in the Future European Environment
DCT	Discrete Cosine Tranform, Diskrete Kosinus Transformation
DCTmod2	aus DCT-Koeffizienten abgeleitete Merkmale
LBP	Local Binary Pattern, Merkmalsextraktion
TPLBP	Three Patch Local Binary Pattern, Merkmalsextraktion
HoG	Histogram of oriented Gradients, Merkmalsextraktion
SSIM	Structural Similarity, Ähnlichkeitsmaß für Bilder
CMC	Cumulative Matching Curve
ROC	Receiver Operator Characteristic
RR	Recognition Rate, Erkennungsrate
MLP	Multi Layer Perceptron, mehrschichtiges Neuronales Netz
PAL	Phase Alternation Line, europäischer Fernsehstandard
VGA	Video Graphics Array, Grafikstandard
fps	frames per second, Bildrate
MERL	Mitsubishi Electric Research Laboratories

Symbolverzeichnis

Allgemeine Symbole

I(x,y)	Ein Bild I (“image”) als zweidimensionales Feld
x	horizontale Koordinatenangabe
y	vertikale Koordinatenangabe
z	Tiefen-Koordinatenangabe
s	Skalierungsfaktor
T	Schwellwert
R_{FP}	Falsch Positiv Rate
R_{FN}	Falsch Negativ Rate
cc	Kreuzkorrelationskoeffizient
S_x, S_y	Sobel-Operatoren in x- und y-Richtung
d	Abstandsmaß
d_{eukl}	Euklidische Distanz
d_{mah}	Mahalanobis-Distanz
$s(I_a, I_b)$	Ähnlichkeitsmaß der Bilder I_a und I_b
$S(I_a, I_b)$	Gewichtete Summe von Ähnlichkeitsmaßen zweier Bilder
$s_{p\chi^2}$	Ähnlichkeitsmaß, ”probabilistic symmetric χ^2“
H	Histogramm
μ	Mittelwert
σ	Standardabweichung
$\hat{I}$	Ein Bild in den Ortsfrequenzbereich transformiert
χ	Verteilung von Merkmalen
ϕ	Aktivierungsfunktion eines Neuronalen Netzes
w	Gewichtungsfaktor
C	Kodebuchgröße, Anzahl von Quantisierungsstufen
α	Indexabstand
∇	Gradient, Vektor der partiellen Ableitungen

Gesichtdetektion mit Haar ähnlichen Merkmalen

II	"integral image", Integralbild
ZS	Zeilensumme
RS	Summe aller Pixel in einem Rechteck
l	Schwellwert
h	schwacher Klassifikator basierend auf einem Haar ähnlichen Merkmal
$\vec{x}$	Bildausschnitt als Vektor der Pixelwerte
p	Vorzeichenfaktor $p \in \{-1, 1\}$
$f(\vec{x})$	Wert des Bildausschnittes $\vec{x}$ bzgl. eines Haar ähnlichen Merkmals
α	Gewichtungsfaktor
H	starker Klassifikator
y	Klassenzugehörigkeit $y \in \{1, 0\}$
m	Anzahl positiver Trainingsbeispiele
k	Anzahl negativer Trainingsbeispiele
w	Gewichtung der Trainingsbeispiele
ϵ	Klassifikationsfehler
e	Fehler einer Entscheidung, $e = 0$ wenn richtig, $e = 1$ wenn falsch klassifiziert wurde
fp	maximal akzeptierbare Falsch-Positiv Rate pro Kaskadenstufe
FP_{ziel}	ingesamt zu erreichende Falsch-Positiv Rate
FP	aktuelle Gesamt-Falsch-Positiv Rate
d	minimale Detektionsrate pro Kaskadenstufe
D	aktuelle Gesamtdetektionsrate
P	Menge der positiven Trainingsbeispiele
N	Menge der negative Trainingsbeispiele
$\emptyset$	leere Menge von Trainingsbeispielen

Active Shape und Active Appearance Modelle

$\vec{s}$	formbeschreibender Vektor bestehend aus Punktkoordinaten $\vec{s} = (x_1, y_1, .., x_l, y_l)$
$\overline{\vec{s}}$	mittlere Form
R_s	Rotation und Skalierung, skalierte Rotationsmatrix
$\vec{t}$	Translation, Verschiebungsvektor
A_s	Matrix der mittelwertfreien Formvektoren
C_s	Kovarianzmatrix der mittelwertfreien Formvektoren

Active Shape und Active Appearance Modelle (Fortsetzung)

$\vec{e}_{s,i}$	Formeigenvektor
$\lambda_{s,i}$	Formeigenwert
$w_{s,i}$	Gewichtung für den Formeigenvektor
$\vec{w}_s$	Vektor der Formparameter
$E_{s,k}$	Matrix mit den k bedeutensten Formeigenvektoren
$\vec{g}$	Textur- bzw. Grauwertvektor
$\overline{\vec{g}}$	mittlerer Texturvektor
$\vec{e}_{g,i}$	Textureigenvektor
$E_{g,k}$	Matrix, mit den k bedeutensten Textureigenvektoren
$\lambda_{g,i}$	Textureigenwert
$w_{g,i}$	Gewichtung für den Textureigenvektor
$\vec{w}_g$	Vektor der Texturparameter
$\vec{w}$	Zusammengesetzter Vektor aus Form- und Texturparametern
$\overline{\vec{w}}$	mittlerer zusammengesetzter Parametervektor
$\vec{e}_{a,i}$	Appearance-Eigenvektor
$w_{a,i}$	Gewichtung des Appearance-Eigenvektors
$\vec{w}_a$	Appearance-Parametervektor bestehend aus allen $w_{a,i}$
$\vec{p}$	Vektor bestehend aus $\vec{w}_a$ und Parametern von R_s und $\vec{t}$
$\vec{r}$	Residuum, Differenz zweier Texturvektoren
P	Prediktormatrix

Iterative Closest Point, geometrische Transformationen

A	auszurichtender Datensatz, Punktwolke $A = (\vec{a}_1, \vec{a}_2, ..., \vec{a}_k)$
B	Referenzdatensatz, Punktwolke $B = (\vec{b}_1, \vec{b}_2, ..., \vec{b}_k)$
$\vec{a}_i$	ein Punkt aus Datensatz A
$\overline{\vec{a}}$	Schwerpunkt von A
$\vec{b}_i$	ein Punkt aus Datensatz B
$\overline{\vec{b}}$	Schwerpunkt von B
s	Skalierungsfaktor
R	Rotationsmatrix
E	Einheitsmatrix
$\vec{t}$	Verschiebungsvektor

Eigenfaces, Hautkomponentenanalyse

$\vec{x}$	Ein Gesichtsbild als Spaltenvektor, Gesichtsvektor
$\overline{\vec{x}}$	das mittlere Gesicht als Spaltenvektor
A	Matrix der mittelwertfreien Gesichtsvektoren
C	Kovarianzamtrix der mittelwertfreien Gesichtsvektoren
$\vec{e}$	Eigenvektor
λ	Eigenwert
$\vec{w}$	Projektion, Gewichtsvektor
q	Qualitätsmaß
$\vec{x}_r$	Rückprojektion

Hidden Markov Modelle

λ	Ein Hidden Markov Modell
S	Zustand
Q	Menge aller Zustände
A	Matrix der Übergangswahrscheinlichkeiten
a_{ij}	Übergangswahrscheinlichkeit von Zustand S_i zu Zustand S_j
B	Matrix der Beobachtungswahrscheinlichkeiten
o	eine Beobachtung
O	Eine Beobachtungssequenz
q_t	Zustand zum Zeitpunkt t
q	Zustandssequenz $q = q_1, \ldots, q_T; q_t \in Q$
π	Vektor der Einsprungwahrscheinlichkeiten
π_j	Wahrscheinlichkeit, dass die Zustandssequenz im Zustand S_j beginnt
$b_j(o)$	Wahrscheinlichkeitverteilung der Beobachtungungen o im Zustand S_j

Definition des Gesichtsausschnitts

$\vec{p}_{la}$	Ortsvektor, Koordinaten des Mittelpunktes des linken Auges
$\vec{p}_{ra}$	Ortsvektor, Koordinaten des Mittelpunktes des rechten Auges
d_a	Augenabstand
$\vec{v}_a$	Richtungsvektor der vom linken zum rechten Auge zeigt
$\vec{v}_b$	Ein zu $\vec{v}_a$ senkrechter Vektor
$\vec{m}_a$	Mittelpunkt zwischen den Augen
b	Breitenfaktor
h	Höhenfaktor

Helligkeitskorrekturen, Histogrammausgleich

g	Grauwert, typischerweise $g \in [0, 255]$
a(g)	absolute Häufigkeit eines Grauwertes g
$r(g)$	relative Häufigkeit eines Grauwertes g
$s_a(g)$	absolute Summenhäufigkeit aller Grauwerte $\leq g$
$s_r(g)$	relative Summenhäufigkeit aller Grauwerte $\leq g$
$\vec{p}_z$	Zeilenprojektion
$\vec{p}_s$	Spaltenprojektion
$P(x, y)$	Projektion entstanden aus dem Produkt von Zeilen- und Spaltenprojektion

Merkmalsextraktion

d_o	Überlappung benachbarter Blöcke in Pixeln
s_b	Seitenlänge des Blocks in Pixel
N_b	Anzahl von Blöcken
H	Höhe in Pixel
B	Breite in Pixel
N_{bz}	Anzahl Blöcke pro Zeile
N_{bs}	Anzahl Blöcke pro Spalte
$C_{i,j}$	der DCT-Koeffizient für die i-te horizontale und j-te vertikale Kosinusfrequenz

Fusion Eigenfaces und HMM

$p_i(I)$	Normierte Produktionswahrscheinlichkeit eines Bildes für ein gegebenes HMM der Klasse i
$d_i(I)$	Normierte und invertierte Mahalanobistanz eines Bildes zu dem Datenbankeintrag der Klasse i

Structural Similarity, Bewertung von Bildähnlichkeiten

l	Luminanz
c	Kontrast
s	Struktur

Gesichtsdetektion in Tiefendaten

H	mittlere Krümmung
K	Gaußsche Krümmung
$\vec{p}$	Ein Ortsvektor mit den Koordinaten x,y und z
$\vec{p}_e$	Koordinaten der Augenhypothese
$\vec{p}_n$	Koordinaten der Nasenhypothese
$\vec{x}_h$	Gesichtshypothese

Literaturverzeichnis

[ABDP06] Antini, G. ; Berretti, S. ; Del Bimbo, A. ; Pala, P.: 3D face identification based on arrangement of salient wrinkles. In: *Proceedings of the IEEE International Conference on Multimedia & Expo (ICME)*. Los Alamitos, CA, USA : IEEE Computer Society, 2006. – ISBN 1–4244–0366–7, S. 85–88

[AHP06] Ahonen, T. ; Hadid, A. ; Pietikäinen, M.: Face Description with Local Binary Patterns: Application to Face Recognition. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (2006), Nr. 12, S. 2037–2041. – ISSN 0162–8828

[AMM+08] Albiol, A. ; Monzo, D. ; Martin, A. ; Sastre, J. ; Albiol, A.: Face recognition using HOG-EBGM. In: *Pattern Recognition Letters* 29 (2008), Nr. 10, S. 1537–1543. – ISSN 0167–8655

[ANR74] Ahmed, N. ; Natarajan, T. ; Rao, K.R.: Discrete Cosine Transform. In: *IEEE Transactions on Computers* 23 (1974), Nr. 1, S. 90–93. – ISSN 0018–9340

[AR05] Acharya, T. ; Ray, Ajoy K.: *Image Processing - Principles and Applications*. Wiley-Interscience, 2005. – ISBN 0–4717–1998–6

[ASR07] Arsić, D. ; Schuller, B. ; Rigoll, G.: Suspicious Behavior Detection in Public Transport by Fusion of Low-Level Video Descriptors. In: *Proceedings 8th International Conference on Multimedia and Expo ICME 2007, Beijing, China*, 2007. – ISBN 1–4244–1016–9, S. 2018–2021

[BCF06] Bowyer, K. W. ; Chang, K. I. ; Flynn, P. J.: A survey of approaches and challenges in 3D and multi-modal 3D + 2D face recognition. In: *Computer Vision and Image Understanding* 101 (2006), Nr. 1, S. 1–15. – ISSN 1077–3142

[BDP06] Berretti, S. ; Del Bimbo, A. ; Pala, P.: Description and retrieval of 3D face models using iso-geodesic stripes. In: *Proceedings of the 8th ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR)*, ACM, 2006. – ISBN 978–1–60558–312–9, S. 13–22

[Bil98] Bilmes, J.: A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models / Computer Science Institute, Univ. of California, Berkeley. 1998 (Technical Report 97-021). – Forschungsbericht

[BJ86] Besl, P. J. ; Jain, R. C.: Invariant surface characteristics for 3D object recognition in range images. In: *Computer Vision, Graphics and Image Processing* 33 (1986), Nr. 1, S. 33–80. – ISSN 0734–189X

[BL90] Besslich, P. W. ; Lu, T.: *Diskrete Orthogonaltransformationen*. Springer Verlag, 1990. – ISBN ISBN 3–540–52151–8

[BL91] Bojanczyk, A. W. ; Lutoborski, A.: Computation of the Euler angles of a symmetric 3X3 matrix. In: *SIAM Journal on Matrix Analysis and Applications* 12 (1991), Nr. 1, S. 41–48. – ISSN 0895–4798

[BM92] Besl, P. J. ; McKay, N. D.: A Method for Registration of 3-D Shapes. In: *IEEE Transactions on Pattern Analysis Machine Intelligence* 14 (1992), Nr. 2, S. 239–256. – ISSN 0162–8828

[BR02] Blanz, V. ; Romdhani, S.: Face Identification across Different Poses and Illuminations with a 3D Morphable Model. In: *FGR '02: Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*. Washington, DC, USA : IEEE Computer Society, 2002. – ISBN 0–7695–1602–5, S. 202

[BV99] Blanz, V. ; Vetter, T.: A morphable model for the synthesis of 3D faces. In: *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. New York, NY, USA : ACM Press/Addison-Wesley Publishing Co., 1999. – ISBN 0–201–48560–5, S. 187–194

[Cay59] Cayley, A.: A Sixth Memoir upon Quantics. In: *Royal Society of London Philosophical Transactions Series I* 149 (1859), S. 61–90

[CCS06] Colombo, A. ; Cusano, C. ; Schettini, R.: 3D face detection using curvature analysis. In: *Pattern Recognition* 39 (2006), Nr. 3, S. 444–455. – ISSN 0031–3203

[CET01] Cootes, T. F. ; Edwards, G. J. ; Taylor, C. J.: Active Appearance Models. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23 (2001), Nr. 6, S. 681–685. – ISSN 0162–8828

[Cha08] Cha, S.-H.: Taxonomy of nominal type histogram distance measures. In: *MATH'08: Proceedings of the American Conference on Applied Mathematics*. Stevens Point, Wisconsin, USA : World Scientific and Engineering Academy and Society (WSEAS), 2008. – ISBN 978–960–6766–47–3, S. 325–330

[CKn02] Cootes, T. F. ; Kittipanya-ngam, P.: Comparing Variations on the Active Appearance Model Algorithm. In: *Proceedings of the British Machine Vision Conference* 2 (2002), S. 837–846

[CSSK02] Chetverikov, D. ; Svirko, D. ; Stepanov, D. ; Krsek, Pavel: The Trimmed Iterative Closest Point Algorithm. In: *Proceedings of the 16th International Conference on Pattern Recognition (ICPR)* Bd. 3. Washington, DC, USA : IEEE Computer Society, 2002. – ISBN 0–7695–1695–X, S. 545–548

[CT93] Cootes, T. F. ; Taylor, C. J.: Active Shape Model Search using Local Grey-Level Models: A Quantitative Evaluation. In: *Proceedings of the 4th British Machine Vision Conference*, BMVA Press, 1993, S. 639–648

[CT04] Cootes, T.F. ; Taylor, C.J.: Statistical Models of Appearance for Computer Vision / University of Manchester. 2004. – Forschungsbericht

[CTCG95] Cootes, T. F. ; Taylor, C. J. ; Cooper, D. H. ; Graham, J.: Active shape models—their training and application. In: *Computer Vision and Image Understanding* 61 (1995), Nr. 1, S. 38–59. – ISSN 1077–3142

[CV03] Ciocoiu, I.B. ; Valmar, B.: A comparison between two preprocessing techniques in PCA-based face recognition. In: *International Symposium on Signals, Circuits and Systems (SCS)* Bd. 1, 2003. – ISBN 0–7803–7979–9, S. 285–288

[CWT00] Cootes, T. F. ; Walker, K. ; Taylor, C. J.: View-Based Active Appearance Models. In: *FG '00: Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000*. Washington, DC, USA : IEEE Computer Society, 2000. – ISBN 0–7695–0580–5, S. 227

[CZZ04] Chen, S. ; Zhang, D. ; Zhou, Z.-H.: Enhanced (PC)2 A for face recognition with one training image per person. In: *Pattern Recognition Letters* 25 (2004), Nr. 10, S. 1173–1181. – ISSN 0167–8655

[DST92] Dillencourt, M. B. ; Samet, H. ; Tamminen, M.: A general approach to connected-component labeling for arbitrary image representations. In: *Journal of the ACM* 39 (1992), Nr. 2, S. 253–280. – ISSN 0004–5411

[DT05] Dalal, N. ; Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* Bd. 1. Washington, DC, USA : IEEE Computer Society, 2005. – ISBN 0–7695–2372–2, S. 886–893

[ELF97] Eggert, D. W. ; Lorusso, A. ; Fisher, R. B.: Estimating 3-D rigid body transformations: a comparison of four major algorithms. In: *Machine Vision and Applications* 9 (1997), Nr. 5-6, S. 272–290. – ISSN 0932–8092

[EMR00] Eickeler, S. ; Mueller, S. ; Rigoll, G.: Recognition of JPEG compressed face images based on statistical methods. In: *Image and Vision Computing* 18 (2000), Nr. 4, S. 279–287

[ESE08] Ezra, E. ; Sharir, M. ; Efrat, A.: On the performance of the ICP algorithm. In: *Computational Geometry: Theory and Applications* 41 (2008), Nr. 1-2, S. 77–93. – ISSN 0925–7721

[ETC98] Edwards, G. J. ; Taylor, C. J. ; Cootes, T. F.: Interpreting Face Images Using Active Appearance Models. In: *FG '98: Proceedings of the 3rd. International Conference on Face & Gesture Recognition*. Washington, DC, USA : IEEE Computer Society, 1998. – ISBN 0–8186–8344–9, S. 300

[Fit01] Fitzgibbon, A. W.: Robust Registration of 2D and 3D Point Sets. In: *Proceedings of the British Machine Vision Conference (BMVC)*, 2001. – ISBN 1–901725–16–2, S. 411–420

[For97] Fortune, S.: Voronoi diagrams and Delaunay triangulations. In: *Handbook of discrete and computational geometry* (1997), S. 377–388. ISBN 0–8493–8524–5

[FS95] Freund, Y. ; Schapire, R. E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: *EuroCOLT '95: Proceedings of the Second European Conference on Computational*

Learning Theory. London, UK : Springer-Verlag, 1995. – ISBN 3–540–59119–2, S. 23–37

[FSB97] Fromherz, T. ; Stucki, P. ; Bichsel, M.: A Survey of Face Recognition / University of Zurich, Department of Computer Science. 1997 (97.01). – Forschungsbericht

[Gau05] Gaultier, D.: *Security of Aircraft in the Future European Environment (SAFEE)*. `http://europa.eu.int/comm/research/aeronautics/info/news/article_681_en.html`. Version: September 2005

[GB03] Gross, R. ; Brajovic, V.: An Image Preprocessing Algorithm for Illumination Invariant Face Recognition. In: *Audio- and Video-Based Biometric Person Authentication*, Springer Verlag, 2003. – ISBN 978–3–540–40302–9, S. 10–18

[Gib91] Gibbs, J. W.: On the Rôle of Quaternions in the Algebra of Vectors. In: *Nature* 43 (1891), April, S. 511–513

[Haa10] Haar, A.: Zur Theorie der orthogonalen Funktionensysteme. In: *Mathematische Annalen* 69 (1910), Nr. 3, S. 331–371. – ISSN 0025–5831 (Print) 1432–1807 (Online)

[Ham53] Hamilton, W. R.: *Lectures on Quaternions: Containing a Systematic Statement of a New Mathematical Method*. Hodges and Smith, Dublin, 1853

[HJLM07] Huang, G. B. ; Jain, V. ; Learned-Miller, E. G.: Unsupervised Joint Alignment of Complex Images. In: *Proceedings of the 11th IEEE International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA : IEEE Computer Society, 2007. – ISBN 978–1–4244–1630–1, S. 1–8

[HKP91] Hertz, J. ; Krogh, A. ; Palmer, R. G.: *Introduction to the theory of neural computation*. Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 1991. – ISBN 0–201–50395–6

[Hor87] Horn, B. K. P.: Closed-form solution of absolute orientation using unit quaternions. In: *Journal of the Optical Society of America. A* 4 (1987), Apr, Nr. 4, S. 629–642

[HPA02] Heseltine, T. ; Pears, N. ; Austin, J.: Evaluation of image preprocessing techniques for eigenface-based face recognition. In: *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* Bd. 4875, 2002, S. 677–685

[HRBLM07] Huang, G. B. ; Ramesh, M. ; Berg, T. ; Learned-Miller, E.: Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments / University of Massachusetts. 2007 (Technical Report 07-49). – Forschungsbericht

[IB98a] Isard, M. ; Blake, A.: Condensation - conditional density propagation for visual tracking. In: *International Journal of Computer Vision* 29 (1998), Nr. 1, S. 5–28. – ISSN 0920–5691

[IB98b] Isard, M. ; Blake, A.: ICONDENSATION: Unifying Low-Level and High-Level Tracking in a Stochastic Framework. In: *Proceedings of the 5th European Conference on Computer Vision (ECCV)* Bd. 1. London, UK : Springer-Verlag, 1998. – ISBN 3–540–64569–1, S. 893–908

[Jäh01] Jähne, B.: *Digitale Bildverarbeitung*. 5., überarbeitete und erweiterte Auflage. Springer, 2001. – ISBN 3–540–41260–3

[JL05] Jain, A. K. ; Li, S. Z.: *Handbook of Face Recognition*. Secaucus, NJ, USA : Springer-Verlag New York, Inc., 2005. – ISBN 0–3874–0595–X

[Kle71] Klein, F.: Ueber die sogenannte Nicht-Euklidische Geometrie. In: *Mathematische Annalen* 4 (1871), Dezember, Nr. 4, S. 573–625. – ISSN 0025–5831

[LM06] Learned-Miller, E. G.: Data Driven Image Models through Continuous Joint Alignment. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28 (2006), Nr. 2, S. 236–250. – ISSN 0162–8828

[Mac67] MacQueen, J. B.: Some methods of classification and analysis of multivariate observations. In: *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability* Bd. 1, University of California Press, 1967, S. 281–297

[MAM09] Mahoor, M. H. ; Abdel-Mottaleb, M.: Face recognition based on 3D ridge images obtained from range data. In: *Pattern Recognition* 42 (2009), Nr. 3, S. 445–451. – ISSN 0031–3203

[MB98] Martinez, A. M. ; Benavente, R.: The AR Face Database / Computer Vision Center, Universitat Autònoma de Barcelona. 1998 (24). – Forschungsbericht

[MB04] Matthews, I. ; Baker, S.: Active Appearance Models Revisited. In: *International Journal of Computer Vision* 60 (2004), Nr. 2, S. 135–164. – ISSN 0920–5691

[MFA08] Mousavi, M. H. ; Faez, K. ; Asghari, A.: Three Dimensional Face Recognition Using SVM Classifier. In: *Proceedings of the Seventh IEEE/ACIS International Conference on Computer and Information Science (ICIS)*. Washington, DC, USA : IEEE Computer Society, 2008. – ISBN 978–0–7695–3131–1, S. 208–213

[MMP+02] Mariano, V. Y. ; Min, J. ; Park, J.-H. ; Kasturi, R. ; Mihalcik, D. ; Li, H. ; Doermann, D. ; Drayer, T.: Performance Evaluation of Object Detection Algorithms. In: *Proceedings of the 16th International Conference on Pattern Recognition (ICPR)* Bd. 3. Washington, DC, USA : IEEE Computer Society, 2002. – ISBN 0–7695–1695–X, S. 965–969

[MRH06] Marcel, S. ; Rodriguez, Y. ; Heusch, G.: On the Recent Use of Local Binary Patterns for Face Authentication / IDIAP Research Institute. 2006 (Idiap-RR-34-2006). – Forschungsbericht

[MS04] Moreno, A. B. ; Sanchez, A.: GavabDB, a 3D Face Database. In: *2nd COST Workshop on Biometrics on the Internet: Fundamentals, Advances and Applications* (2004), S. 77–82

[NC08] Nair, P. ; Cavallaro, A.: Matching 3D Faces with Partial Data. In: *Electronic Proceedings of the 19th British Machine Vision Conference*, British Machine Vision Association, 2008, S. http://www.comp.leeds.ac.uk/bmvc2008/proceedings/papers/286.pdf

[Nef99] Nefian, A. V.: An embedded HMM-based approach for face detection and recognition. In: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* Bd. 6. Los Alamitos, CA, USA : IEEE Computer Society, 1999. – ISBN 0–7803–5041–3, S. 3553–3556

[NH98] Nefian, A. V. ; Hayes III, M. H.: Hidden Markov Models For Face Recognition. In: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* Bd. 5, 1998. – ISBN 0–7803–4428–6, S. 2721–2724

[NJ07] Nowak, E. ; Jurie, F.: Learning Visual Similarity Measures for Comparing Never Seen Objects. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007. – ISBN 1–4244–1180–7, S. 1–8

[OPM02] Ojala, T. ; Pietikäinen, M. ; Mäenpää, T.: Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary

Patterns. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), Nr. 7, S. 971–987. – ISSN 0162–8828

[Pat97] Patterson, D. W.: *Künstliche neuronale Netze. Das Lehrbuch*. Pearson Education Deutsch, 1997. – ISBN 978–3827295316

[Pau50] Pauli, W.: On the Connection between Spin and Statistics. In: *Progress of Theoretical Physics* 5 (1950), Juli, S. 526–543

[PM08] Papandreou, G. ; Maragos, P.: Adaptive and constrained algorithms for inverse compositional Active Appearance Model fitting. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008. – ISBN 978–1–4244–2242–5, S. 1–8

[PMRR00] Phillips, P. J. ; Moon, H. ; Rizvi, S. A. ; Rauss, P. J.: The FERET Evaluation Methodology for Face-Recognition Algorithms. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22 (2000), Nr. 10, S. 1090–1104. – ISSN 0162–8828

[PMS94] Pentland, A. ; Moghaddam, B. ; Starner, T.: View-based and modular eigenspaces for face recognition. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994. – ISBN 0–8186–5825–8, S. 84–91

[QGSR01] Quintiliano, P. ; Guadagnin, R. ; Santa-Rosa, A.: Practical Procedures to Improve Face Recognition Based on Eigenfaces and Principal Component Analysis. In: *Pattern Recognition and Image Analysis* 11 (2001), Nr. 2, S. 372–375

[Rab90] Rabiner, L. R.: A tutorial on hidden Markov models and selected applications in speech recognition. (1990), S. 267–296. ISBN 1–55860–124–4

[RBK98] Rowley, H. A. ; Baluja, S. ; Kanade, T.: Neural Network-Based Face Detection. In: *IEEE Transactions On Pattern Analysis and Machine Intelligence* 20 (1998), S. 23–38. – ISSN 0162–8828

[RHW88] Rumelhart, D. E. ; Hinton, G. E. ; Williams, R. J.: Learning representations by back-propagating errors. (1988), S. 696–699. ISBN 0–262–01097–6

[RL01] Rusinkiewicz, S. ; Levoy, M.: Efficient Variants of the ICP Algorithm. In: *Proceedings on the International Conference on 3D Digital Imaging and Modeling*. Los Alamitos, CA, USA : IEEE Computer Society, 2001. – ISBN 0–7695–0984–3, S. 145

[Ros58] Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. In: *Psychological review* 65 (1958), Nr. 6, S. 386–408

[Row99] Rowley, H. A.: *Neural Network-Based Face Detection*, Carnigie Mellon University, School of Computer Science, Diss., 1999

[RPM98] Rizvi, S. A. ; Phillips, P. J. ; Moon, H.: A Verification Protocol and Statistical Performance Analysis for Face Recognition Algorithms. In: *CVPR '98: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Washington, DC, USA : IEEE Computer Society, 1998. – ISBN 0–8186–8497–6, S. 833

[Rus02] Russ, J. C.: *Image Processing Handbook, Fourth Edition*. Boca Raton, FL, USA : CRC Press, Inc., 2002. – ISBN 0–8493–1142–X

[SBB03] Sim, T. ; Baker, S. ; Bsat, M.: The CMU Pose, Illumination, and Expression Database. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25 (2003), Nr. 12, S. 1615–1618. – ISSN 0162–8828

[Sch09] Schreiber, S.: *Personenverfolgung und Gestenerkennung in Videodaten*, Technische Universität München, Diss., 2009

[SK87] Sirovich, L. ; Kirby, M.: Low-dimensional procedure for the characterization of human faces. In: *Journal of the Optical Society of America A* 4 (1987), März, S. 519–524

[SL09] Sanderson, C. ; Lovell, B.C.: Multi-Region Probabilistic Histograms for Robust and Scalable Identity Inference. In: *The 3rd IAPR/IEEE International Conference on Biometrics (ICB)*, Springer, 2009. – ISBN 978–3–642–01792–6, S. 199–208

[SP03] Sanderson, C. ; Paliwal, K. K.: Fast features for face authentication under illumination direction changes. In: *Pattern Recognition Letters* 24 (2003), Nr. 14, S. 2409–2419. – ISSN 0167–8655

[SR08] Störmer, A. ; Rigoll, G.: A multi-step alignment scheme for face recognition in range images. In: *Proceedings of the 15th IEEE International Conference on Image Processing (ICIP)*, 2008. – ISBN 978–1–4244–1764–3, S. 2748–2751

[SRBB06] Suard, F. ; Rakotomamonjy, A. ; Bensrhair, A. ; Broggi, A.: Pedestrian Detection using Infrared images and Histograms of Oriented Gradients. In: *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2006. – ISBN 4–901122–86–X, S. 206–212

[SS04] Störmer, A. ; Stadermann, J.: Constructing Faces using Active Appearance Models and Evaluating the Similarity to the Original Image Data. In: *Proceedings of the 3rd Workshop on SelfOrganisation of AdaptiVE Behavior (SOAVE 2004)* Bd. Fortschritt-Berichte VDI Reihe 10, VDI-Verlag, 2004, S. 47–56

[SSR06] Schreiber, S. ; Störmer, A. ; Rigoll, G.: A Hierarchical ASM/AAM Approach in a Stochastic Framework for Fully Automatic Tracking and Recognition. In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2006. – ISBN 1–4244–0481–9, S. 1773–1776

[SSR08] Schreiber, S. ; Störmer, A. ; Rigoll, G.: Omnidirectional tracking and recognition of persons in planar views. In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2008. – ISBN 978–1–4244–1764–3, S. 1476–1479

[Stö04] Störmer, A.: *Untersuchung der Möglichkeiten zur Beschreibung von Gesichtern mit einer minimalen Anzahl von Appearance-Parametern*, Technische Universität Ilmenau, Diplomarbeit, 2004

[SY94] Samaria, F.S. ; Young, S.: HMM-Based Architecture for Face Identification. In: *Image and Vision Computing* 12 (1994), October, Nr. 8, S. 537–543. – ISSN 0262–8856

[TP91a] Turk, M. ; Pentland, A.: Eigenfaces for recognition. In: *Journal of Cognitive Neuroscience* 3 (1991), Nr. 1, S. 71–86. – ISSN 0898–929X

[TP91b] Turk, M.A. ; Pentland, A.P.: Face recognition using eigenfaces. In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 1991. – ISBN 0–8186–2148–6, S. 586–591

[Vit67] Viterbi, A.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. In: *IEEE Transactions on Information Theory* 13 (1967), April, Nr. 2, S. 260–269. – ISSN 0018–9448

[VJ01] Viola, P. ; Jones, M.: Rapid object detection using a boosted cascade of simple features. In: *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1 (2001), S. 511–518. ISBN 0–7695–1272–0

[VJ04] Viola, P. ; Jones, M.: Robust Real-time Object Detection. In: *International Journal of Computer Vision* Bd. 57, Springer Netherlands, 2004. – ISSN 0920–5691 (Print) 1573–1405 (Online), S. 137–154

[Wal06] Wallhoff, F.: *Entwicklung und Evaluierung neuartiger Verfahren zur automatischen Gesichtsdetektion, Identifikation und Emotionserkennung*, Technische Universität München, Diss., 2006

[WBSS04] Wang, Z. ; Bovik, A. C. ; Sheikh, H. R. ; Simoncelli, E. P.: Image quality assessment: from error visibility to structural similarity. In: *Image Processing, IEEE Transactions on* 13 (2004), Nr. 4, S. 600–612. – ISSN 1057–7149

[WER01] Wallhoff, F. ; Eickeler, S. ; Rigoll, G.: A Comparison of Discrete and Continuous Output Modeling Techniques for a Pseudo-2d Hidden Markov Model Face Recognition System. In: *Proceedings of the International Conference on Image Processing (ICIP)* Bd. 2, 2001, S. 685–688

[WFKv97] Wiskott, L. ; Fellous, J.-M. ; Krüger, N. ; von der Malsburg, C.: Face Recognition by Elastic Bunch Graph Matching. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19 (1997), Nr. 7, S. 775–779. – ISSN 0162–8828

[WHT08] Wolf, L. ; Hassner, T. ; Taigman, Y.: Descriptor Based Methods in the Wild. In: *Real-Life Images workshop at the European Conference on Computer Vision (ECCV)*, 2008

[WZ02] Wu, J. ; Zhou, Z.-H.: Face recognition with one training image per person. In: *Pattern Recognition Letters* 23 (2002), Nr. 14, S. 1711–1719. – ISSN 0167–8655

[YB07] Yan, P. ; Bowyer, K. W.: A fast algorithm for ICP-based 3D shape biometrics. In: *Computer Vision and Image Understanding* 107 (2007), Nr. 3, S. 195–202. – ISSN 1077–3142

[YKA02] Yang, M.-H. ; Kriegman, D. J. ; Ahuja, N.: Detecting Faces in Images: A Survey. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24 (2002), Nr. 1, S. 34–58. – ISSN 0162–8828

[ZCRP03] Zhao, W. ; Chellappa, R. ; Rosenfeld, A. ; Phillips, P. J.: Face recognition: A literature survey. In: *ACM Computing Surveys* 35 (2003), Nr. 4, S. 399–458. – ISSN 0360–0300