Institut für Logik, Komplexität
und Deduktionssysteme (ILKD)
Fakultät für Informatik
Universität Karlsruhe (TH)

ADAPTIVE VERFAHREN ZUR KAMERABASIERTEN ERKENNUNG VON GESICHTSMERKMALEN IM PKW

Diplomarbeit
von

Andy King

Oktober 2003

Verantwortlicher Betreuer: Prof. Dr.rer.nat. Alexander Waibel
Betreuender Mitarbeiter: Dr.-Ing. Rainer Stiefelhagen

Inhaltsverzeichnis

Kapitel 1

Einleitung und Motivation

Moderne Fahrerassistenzsysteme sollen in der Lage sein, den Fahrer im Fall einer plötzlich auftretenden Gefahrensituation zu warnen. Eine Warnung soll allerdings nur dann abgegeben werden, wenn davon ausgegangen werden kann, dass der Fahrer aufgrund seiner aktuellen Blickrichtung die Situation nicht erfasst haben kann (zum Beispiel wenn er sich mit dem Beifahrer unterhält und eine von links auf die Fahrbahn tretende Person übersieht).
Im Rahmen der bisherigen Arbeiten am DaimlerChrysler Forschungszentrum in Ulm wurden bereits eine Kamera, die das Gesicht des Fahrers erfasst, in ein Versuchsfahrzeug integriert, verschiedene Messkampagnen durchgeführt sowie ein Echtzeit-3D-Tracking zum Tracken des Kopfes entwickelt und getestet.

Ziel der vorliegenden Arbeit ist die Verbesserung der Initialisierung des bestehenden Tracking: Um den statistischen Tracker (Kalmanfilter) zu initialisieren, müssen im Kamerabild die Augen, die Nase und der Mund als Gesichtsmerkmale identifiziert werden (Abb. 1.1). Es gibt momentan schon eine Initialisierung, diese ist aber nicht gut genug. In der Literatur sind zwar verschiedene Ansätze zur Lösung dieses Problems beschrieben, allerdings sind sie üblicherweise nicht für die folgenden Randbedingungen geeignet.
Erstens ist die Umgebung nicht kooperativ, das heißt zum einen ist die Kamera in einem Fahrzeug montiert und es gibt keine Möglichkeit, die Beleuchtung zu beeinflussen (Lichtwechsel von etlichen zehn Dezibel innerhalb weniger Bilder inklusive Schlagschatten sind keine Seltenheit) und zum anderen ist der Fahrer nicht kooperativ, das heißt die Algorithmen sollten bei den verschiedensten Personen (helle/dunkle Haut, mit/ohne Brille, mit/ohne Vollbart) ohne vorheriges individuelles Training funktionieren.
Zweitens handelt es sich bei der Bildgröße um eine Auflösung von 640x480 (non-interlaced): Um den relevanten Teil des Innenraums zu erfassen, wird ein Objektiv von ungefähr 8 mm Brennweite verwendet, weshalb die Auflösung des Gesichtes nicht besonders gut ist.
Drittens ist die angestrebte Video-Taktrate beim Tracking 80 Hz. Die gesamte Algorithmik sollte daher maximal zirka 10 ms (1 GHz-Prozessor) pro Bild benötigen und nur die relevanten Bildteile verarbeiten. Bei der Initialisierung kann man sich aber (im Gegensatz zum Tracking der Augenlider und des Mundes) mehr Zeit lassen.

Abbildung 1.1: (links) Fahrzeugcockpit mit installierter Kamera, (Mitte) Aufnahme des Fahrers, (rechts) Finden des Gesichtes und der Gesichtsmerkmale (Augen, Nase, Mund)

Das folgende Kapitel beschäftigt sich mit den Grundlagen der Gesichtsdetektion und Gesichtserkennung mittels Videobildern, um allen Lesern einen breiten Hintergrund zu geben. Es wird der in der Literatur beschriebene momentane Stand der Technik gezeigt.
Im dritten Kapitel wird das bestehende Tracking-System am DaimlerChrysler Forschungszentrum in Ulm dargestellt. Dabei werden auch die Randbedingungen und Herausforderungen erläutert.
Im vierten Kapitel werden die untersuchten Ansätze zur Erkennung von Gesichtsmerkmalen detailliert beschrieben. Dabei wird auf die einzelnen Komponenten des Erkenners eingegangen: die Fenstertechnik, um die Rohdaten „einzusammeln“, mögliche Vorverarbeitungsschritte, die Merkmalsextraktion und schließlich zwei beispielhafte Klassifikatorarten.
Bei diesen Klassifikatoren handelt sich einerseits um Polynomklassifikatoren und andererseits um Entscheidungsbäume.
Das Kapitel ‚Experimente und Ergebnisse‘ erläutert zuerst die Grundbegriffe der Evaluierung des Erkennungssystems und zeigt dann anhand von Experimenten und dazugehörigen MATLAB Diagrammen, wie gut die hier entwickelten Algorithmen funktionieren.
Das letzte Kapitel gibt eine Zusammenfassung der Diplomarbeit und verweist auf Erweiterungsmöglichkeiten, die in der Zukunft erstrebenswert wären.

Anhang A zeigt Beispielframes, die in der verwendeten Datenbank enthalten sind.
Anhang B erläutert den Aufbau der Labeldatei für das überwachte Lernen.
Im Anhang C ist ein exemplarischer Auszug aus dem Auswerteskript dargestellt.
Anhang D beinhaltet eine Fülle von MATLAB-Diagrammen, die die Parametervariationen bei den gemachten Experimenten veranschaulichen.
Schließlich befindet sich im Anhang E das Abkürzungsverzeichnis und im Anhang F sind begleitende Internetseiten.

Die Implementierung der Algorithmen erfolgte unter “MathWorks“ MATLAB 6.1 (Release 12.1).

Diese Diplomarbeit wurde am DaimlerChrysler Forschungszentrum in Ulm durchgeführt. Ich möchte mich an dieser Stelle bei meinen dortigen Betreuern Dr. Tilo Schwarz und Dr. Martin Fritzsche herzlich bedanken.
Ebenso gilt mein Dank Prof. Dr.rer.nat. Alexander Waibel und Dr.-Ing. Rainer Stiefelhagen, ohne sie wäre die Arbeit nicht zustande gekommen.
Weiterer Dank gilt Dr. Otto Löhlein, der bei Fragen zu dieser Diplomarbeit jederzeit zur Verfügung stand.

Kapitel 2

Stand der Technik - Literaturrecherche

Im vorliegenden Kapitel soll dem Leser zuerst ein Überblick über das Gebiet der Gesichtsdetektion und Gesichtserkennung mittels Videobildern gegeben werden. Anschließend wird der psychologische und neurophysiologische Hintergrund beleuchtet, um Parallelen zwischen der Natur und künstlichen Erkennungssystemen zu ziehen. Dann werden verschiedene Methodiken zur Detektion vorgestellt, wobei eine Kategorisierung der Ansätze gemacht wird. Danach wird auf die Evaluierung näher eingegangen. Schließlich erfolgt noch eine kurze Zusammenfassung der angesprochenen Punkte und es werden mögliche Anwendungsgebiete aufgezeigt.
Die Ausführungen entstammen hauptsächlich den Veröffentlichungen [1], [2] und [3], die den Stand der Technik in den letzten Jahren hervorragend zusammenfassen.

2.1 Überblick

Das Problem der maschinellen Gesichtserkennung, im Englischen „face recognition technology“ (**FRT**) genannt, kann laut [1] folgendermaßen allgemein formuliert werden: *Hat man „normale“ (das heißt unbewegte) Bilder oder Bilder aus Videosequenzen einer sogenannten „Szene“, dann besteht die Aufgabe nun darin, eine oder mehrere Personen in dieser Szene mittels Gesichtsaufnahmen, die in einer Datenbank hinterlegt sind, zu identifizieren.*

Natürlich können auch zusätzliche Informationen (falls vorhanden) wie Rasse, Alter und Geschlecht dazu benutzt werden, um die Suche gewissermaßen einzuengen. Die Lösung des obigen Problems beinhaltet dabei die Segmentierung der Gesichter aus willkürlichen Szenen (Gesichtsdetektion), die Extraktion von Merkmalen aus der Gesichtsregion, die Identifikation und den Vergleich.

Was die zeitliche Entwicklung betrifft, gilt folgendes: In den frühen und mittleren 70er Jahren des letzten Jahrhunderts wurden typische Techniken der Musterklassifizierung angewendet, die auf gemessenen Eigenschaften zwischen Merkmalen in Gesichtern oder Gesichtsprofilen beruhen. Während der 80er Jahre des letzten Jahrhunderts ruhte die Forschung auf dem Gebiet der Gesichtserkennung hauptsächlich und erst seit den 90er Jahren wächst das Interesse der Forschung an FRT wieder sehr stark. Dies ist sicherlich auch dadurch motiviert, dass Informationen über die Identität, den Zustand und die Absicht eines Benutzers über Bilder gewonnen werden können, und dass Computer dann beispielsweise auf Gesichtsausdrücke und ähnliches reagieren können.

Bei diesem Thema ist es sinnvoll zu wissen, mit was sich Psychologen und Neurowissenschaftler beschäftigen, um dann möglicherweise neue Ansätze für ein „technisches“ Erkennungssystem zu bekommen. Forschungsschwerpunkte sind dabei: Einzigartigkeit von Gesichtern; ob Gesichtserkennung holistisch (also ganzheitlich) oder durch die Analyse von lokalen Merkmalen stattfindet; Analyse von Gesichtsausdrücken und

deren Nutzen für die Erkennung; wie Kleinkinder Gesichter wahrnehmen; wie Gesichter im Gehirn gespeichert werden; die Unfähigkeit, invertierte Gesichtern genau zu erkennen; die Existenz eines sogenannten „Großmutter-Neurons“ für die Gesichtserkennung; die Rolle der rechten Gehirnhälfte für bei der Gesichtswahrnehmung und Krankheiten wie Prosopagnosia (im Englischen auch „face-blindness“ genannt), bei denen die betroffenen Menschen Gesichter nicht mehr oder nur sehr schwach erkennen können. Im nächsten Unterkapitel 2.2 wird auf diese psychologischen und neuropsychologischen Themen detaillierter eingegangen.

Wichtig zu erwähnen ist auf jeden Fall auch die Einteilung der Bilder in statische und dynamische, die aus einer Echtzeit-Videosequenz stammen.
Die erstere Gruppe hat mehrere Vorteile (zum Beispiel ist das Segmentierungsproblem relativ einfach aufgrund des kontrollierbaren beziehungsweise definierbaren Bildaufnahmeprozesses), aber auch Nachteile (beispielsweise kann die automatische räumliche Bestimmung und Segmentierung eines Gesichtes für jeden beliebigen Segmentierungsalgorithmus eine große Herausforderung sein, wenn nur ein einziges statisches Bild zur Verfügung steht).
Im Gegensatz dazu neigen die Bilder, die man mit einer Videokamera erhält, zu schlechter Qualität und der Hintergrund ist oft sehr unberechenbar; jedoch ist aufgrund der Sequenz-Eigenschaft die Segmentierung einer sich bewegenden Person beziehungsweise deren Gesichtes mit Hilfe der Bewegungsanalyse viel einfacher. Außerdem ist die teilweise Rekonstruktion des Gesichtsbildes mittels bestehender Modellen wie auch das Problem der störenden Verkleidung beziehungsweise allgemein der Gesichtsveränderung einfacher zu lösen als bei statischen Vergleichsaufnahmen.

Es gibt mindestens vier wichtige Gebiete, die für FRT relevant sind und wo die Bildsequenzanalyse sehr hilfreich ist:
- Segmentierung von sich bewegenden Objekten (Menschen) in einer Videosequenz
- Strukturschätzung (im Englischen „structure from motion“ genannt)
- 3-D Modelle für Gesichter
- nichtstarre Bewegungsanalyse (zum Beispiel Detektion von Augenblinzeln)

Die nichttriviale Aufgabe der Segmentierung einer Gesichtsregion (Gesichtsdetektion) ist der erste Schritt bei der Gesichtserkennung. Dafür gibt es im statischen Fall verschiedene Vorgehensweisen:
In einem der ersten Papers auf diesem Gebiet ([9]) wird eine Karte aus den erhaltenen Kanten des Originalbildes mit einer großen ovalen Schablone (im Englischen „Template“ genannt, oft eine Ellipse) verglichen, wobei Variationen in Position und Größe des „Templates“ möglich sind. An Positionen, die dafür sprechen, dass sich dort tatsächlich der Kopf befindet, werden die Kanten an den erwarteten Position von Augen, Mund und anderen Gesichtsmerkmalen näher betrachtet, um eine „Kopfhypothese“ zu bestärken oder zu widerlegen.
In [10] wird auch ein top-down Algorithmus für die Kopfextraktion verwendet: geglättete Versionen des Originalbildes (beispielsweise durch lokale Mittelwertbildung) werden zuerst nach Kanten durchsucht, die den Umriss eines Kopfes definieren könnten. Diese extrahierten Lagen der Kanten werden auf das Originalbild zurückprojiziert und eine lokale Feinsuche nach Kanten, die den Kopfumriss bilden könnten, wird gestartet, wobei mehrere Heuristiken verwendet werden, um die Kanten zu verbinden. Hat man dann einen Kopfumriss erhalten, werden wie beim vorherigen Ansatz die erwarteten Positionen von

Augen, Nase und Mund untersucht.
Ähnliche Ansätze basieren auf hierarchischen Bildskalierungen (sogenanntes „Multiresolution"-Schema): zuerst wird in der geringsten Auflösung (zum Beispiel mittels eines sogenannten „Sobeloperators") ein „Template" an den Kopfumriss angepasst und dann wird bei höheren Auflösungen nach Merkmalen wie Augen, Augenbrauen und Lippen gesucht.
In [11] wird ein verformbares „Template" verwendet, das ebenfalls auf dem Kantenbild arbeitet: es ist skalierbar und in Translation und Rotation veränderbar. Für die Bestimmung von Hypothese-Kandidaten wird eine Kostenfunktion eingeführt.
Leider sind viele der Systeme bedingt durch die iterative Verformbarkeit eines ganzen „Templates" durch (zufällige) Skalierung, Translation und Rotation (wobei diese Transformationen nicht zu Ergebnissen führen sollten, die keine Ähnlichkeit mit einem menschlichen Kopf haben) und die notwendige Optimierung (beispielsweise durch sogenanntes „simulated annealing") sehr rechenintensiv!

Alternativ gibt es auch die Möglichkeit über den Ansatz der sogenannten „Eigengesichter", komplette Gesichter zu detektieren: im Kontext der sogenannten Karhunen-Loève (KL) Methode für die Merkmalsgewinnung wird dieser Ansatz näher beschrieben.

Aufbauend auf die Segmentierung folgt die Merkmalsextraktion.

In [13] werden die Bildmerkmale beziehungsweise Bildfeatures in vier Gruppen eingeteilt:
- visuelle Merkmale
- statistische Pixelmerkmale
- Merkmale, beruhend auf Transformationskoeffizienten
- und algebraische Merkmale

Die letzteren geben die intrinsischen (das heißt innewohnenden) aber nicht unbedingt sichtbaren Merkmale eines Bildes wieder und werden durch sogenannte Singulärwerte als Ergebnis der Singulärwertzerlegung (im Englischen „singular value decomposition" (SVD) genannt) repräsentiert. Dieser Merkmalsvektor aus Singulärwerten ist äußerst stabil und invariant gegenüber Varianz der Bildintensität, Transposition, Rotation, Translation und Spiegelung.

Ein Grund, wieso die „Karhunen-Loève" (KL) Methode (obwohl optimal) bei den Forschern im Bereich der Bildkompression nicht angenommen wurde, ist ihre Berechnungskomplexität (schnelle Transformationen wie die diskrete Sinus und Kosinus Transformation werden daher bevorzugt). In [14] wurde die KL Entwicklung für die Darstellung und Erkennung von Gesichtern wieder aufgenommen: hat man einmal die Eigenvektoren (sogenannte „Eigenbilder" beziehungsweise „Eigengesichter", im Englischen „eigenfaces" genannt) berechnet, kann jedes Bild in der Ausgangsmenge mittels einer gewichteten Kombination von Eigenbildern ungefähr rekonstruiert werden (wobei natürlich die Genauigkeit zum gegebenen Bild mit der Anzahl der verwendeten Eigenbilder steigt). Die Gewichte, die man durch die Projektion des Bildes auf die Eigenbild-Komponenten (mittels einer einfachen Inneren Produkt-Operation) erhält, dienen dann als Merkmale.

In [15] wird die (lineare) KL Entwicklung mit zwei anderen Operationen verknüpft:

- KL-IPAT („Karhunen-Loève transform of intensity pattern in the affine-transformed target image“): die KL Entwicklung wird auf das (in Position und Größe) standardisierte Gesichtsbild angewendet
- KL-FSAT („Karhunen-Loève transform of the Fourier spectrum in the affine-transformed target image“): die KL Entwicklung wird auf das Fourier-Spektrum (anstatt der räumlichen Daten) angewendet

Falls das Zielfenster nicht genau „sitzt“ (oft bedingt durch wechselnde Umstände bei der Bildaufnahme) ist die Merkmalsrobustheit bei KL-FSAT aufgrund der Verschiebungsinvarianz des Fourier-Spektrums gegeben! KL-FSAT ist im Vergleich zu KL-IPAT auch viel robuster was verschiedene Kopforientierungen betrifft.

In [12] wird ebenfalls dieser statistische Ansatz der Eigengesichter für die Gesichtsdetektion und Identifikation verwendet. Hat man einmal die Eigengesichter bestimmt, kann jedes Gesicht der Datenbank wie oben beschrieben als Vektor von Gewichten repräsentiert werden. Ein neues Testbildes wird dann genau dem Bild in der Datenbank zugeordnet, dessen Gewichte den Gewichten des Testbildes (im Euklidischen Abstand) am nächsten sind. Durch die Feststellung, dass die Projektion eines Gesichtsbildes und eines Nichtgesichtsbildes sehr unterschiedlich aussieht, hat man praktischerweise einen Detektionsansatz für Gesichter.
Das Konzept der Eigengesichter kann auch auf sogenannte „Eigenmerkmale“ (im Englischen „eigenfeatures“ genannt) wie „Eigenaugen“ und „Eigenmund“ erweitert werden, um Merkmale wie Augen und Mund zu detektieren. Gerade wenn große Variationen im Eingabebild vorhanden sind, ist dieser merkmalsbasierte Mechanismus sehr nützlich. Beispielsweise erhielt die Gruppe um Professor Pentland vom Massachusetts Institute of Technology (MIT) bei einer großen Datenmenge von 7562 Bilder, die mit ungefähr 3000 Leuten aufgenommen wurden, Detektionsraten von 94% für die Augen, von 80% für die Nase und von 56% für den Mund.
Man könnte den Ansatz mit Eigenfunktionen auch auf mehrere Eigenrepräsentationen erweitern: neben der Unterscheidung verschiedener Aufnahmewinkel könnte man ebenso für verschiedene Rassen, Altersgruppen, Geschlechter eine separate Eigenklasse einführen.
Vom ästhetischen Standpunkt her gesehen ist die Eigenanalyse nicht besonders attraktiv, da die Strukturinformation lediglich mittels einzelner Zahlen ohne größeren Zusammenhang kodiert wird. Aber sie hat den großen Vorteil gegenüber anderen Verfahren, dass sie vom rechentechnischen Standpunkt aus ziemlich überschaubar ist.

Natürlich kann man bei der Extraktion von Gesichtsmerkmalen (besonders für die Augendetektion und -erkennung) ähnlich der Segmentierung von Gesichtsregionen verformbare „Templates“ (mit einer Energieminimierung) verwenden. Nachteil ist aber einerseits wieder der enorme Rechenaufwand (sehr viele Parameter) und andererseits zusätzlich die Nichteindeutigkeit (beispielsweise kann ein „Template“-Algorithmus unter Umständen Auge und Augenbraue verwechseln, je nach Initialisierung). Dieser Ansatz ist daher eher logischer Natur.

[16] verfolgt einen gänzlich anderen Ansatz: um die Augen, die Nase und den Mund im Gesicht zu finden, benutzen sie einen sogenannten generalisierten Symmetrie-Operator (mit einem speziellen Symmetrie-Maß), der in einer abgewandelten Form auch zur Verfolgung eines (symmetrischen) Autohecks eingesetzt wird.

Obwohl dieser Ansatz einige Vorteile hat (beispielsweise Skalierungs- und Orientierungsunabhängigkeit), ist die Suche nach den Symmetriepunkten aufgrund des fehlenden a priori Wissens über den Ort des Gesichtes wie bei den vorherigen „Template"-Ansätzen ziemlich rechenintensiv.

In [17] wird ein wissensbasiertes System für die Gesichtsdetektion bei handgezeichneten Skizzen vorgestellt (welches auch auf das Kantenbild eines „normalen" Intensitätsbildes angewendet werden kann). Das System basiert auf einer sogenannten „blackboard-Architektur" und verwendet WENN-DANN Regeln (zum Beispiel: WENN die obere Mundlinie noch nicht gefunden ist, jedoch die untere Mundlinie bereits detektiert wurde, DANN beschränke die Suche nach der oberen Mundlinie auf die Bildregion, die dem unteren Mundbereich direkt anschließt).
Die Verarbeitung geschieht auf vier verschiedenen Abstraktionsebenen (Liniensegmente, Komponentenstücke, Komponenten und das Gesicht): auf der Gesichtsebene wird dann beispielsweise nach der geometrischen Anordnung von Komponenten gesucht, welches ein Gesicht in einem Bild am besten wiedergibt.
Großer Nachteil dieser Methode ist aber, dass man für sämtliche auftretenden Fälle eine eigene Regel braucht und das Regelwerk damit quasi „explodiert" beziehungsweise nicht mehr überschaubar ist!

In [18] und [19] werden sogenannte Gabor basierte Wavelets als Merkmalsdetektoren verwendet, und deren Charakterisierung erfolgt über die Frequenz, die Position und die Orientierung. Merkmale, die sich aus diesen Wavelets ableiten, halten auch holistische Eigenschaften, da sie das Gesicht als ein Cluster von Punkten repräsentieren: der Ort dieser Punkte innerhalb und um die Augen, des Mundes und so weiter kann ziemlich zufällig sein, jedoch kann man Masken um diese Regionen benutzen, um Punktmerkmale zu betonen.

Wir wollen noch kurz den dynamischen Fall betrachten, das heißt wenn eine Videosequenz vorliegt:
Im Kontext von FRT können die Bewegungsparameter für die Vorhersage der Objektposition in folgenden Frames sehr nützlich sein, da sie die Segmentierungsaufgabe einfacher machen. Man kann außerdem aus der Strukturinformation 3-D Modelle für Gesichter bauen und diese Modellen dann für die Gesichtserkennung benutzen, wenn Verdeckung, Veränderungen und Gesichtsrekonstruktionen auftreten.
Im Gegensatz zu monokularen Bildsequenzen erhält man dabei bei binokularen Bildsequenzen absolute Tiefenwerte in dem man die sogenannte „Stereo Triangulation" anwendet.

In [20] wird eine pixelbasierte, einfache Änderungsdetektion mittels Differenzbildern verwendet. Diese Technik (oft kombiniert mit einem einfachen Schwellwertverfahren) stößt aber an ihre Grenzen, wenn mehrere sich bewegende Objekte, Beleuchtungsänderungen und Verdeckungen vorhanden sind.

[21] benutzt Methoden basierend auf sogenannten Flussfeldern, um sich bewegende Menschen zu segmentieren. Falls es Situation gibt, in denen sich sowohl die Kamera als auch das Objekt bewegen, benötigt man komplizierte Segmentierungsprozeduren wie beispielsweise der sogenannte optische Fluss. Die genaue Berechnung eines solchen optischen Flussfeldes ist aber bis heute ein ungelöstes Problem.

In der Literatur findet man zur Bildsequenzanalyse neben der Analyse von Differenzbildern und Diskontinuitäten in (optischen) Flussfeldern mittels Clustering auch sogenannte Linienprozesse, „template“-basierte Ansätze und „Markov random field“(MRF)-Modelle. Die MRF-Theorie ist besonders beliebt, um kontextuelle Beschränkungen in einem Gesichtsmuster (die auch für die Textursegmentierung benutzt werden) zu modellieren, wobei diese normalerweise durch eine kleine Anzahl an Nachbarschaftspixel und korrelierte Merkmale bestimmt werden.

2.2 Psychologischer und neurophysiologischer Hintergrund

Entwickler von Gesichtserkennungsalgorithmen und -systemen sollten sich mit relevanten psychologischen und neuropsychologischen Studien beschäftigen, aber sie sollten auch so klug sein, nur die zu „verwerten“, die aus einer praktischen Sichtweise umsetzbar und besonders wichtig sind.
Das menschliche Erkennungssystem nutzt sowohl für das Speichern als auch das Wiederabrufen von Gesichtsbildern einzelne oder aber mehrere Reize gleichzeitig, die von vielen manchmal sogar alle Sinnen stammen (wie beispielsweise vom Sehen, Hören, Riechen oder Tasten). Das menschliche Gehirn hat aber seine Schwächen in der Gesamtzahl von Personen, an die es sich genau erinnern kann!

Als Beweis für die Existenz eines dedizierten Gesichtsverarbeitungsprozesses im menschlichen Gehirn wird in der Literatur folgendes angebracht:

- An Gesichter erinnern sich Menschen viel einfacher als an andere Objekte (zum Beispiel 50 verschiedene Steine), wenn sie in einer aufrechten Orientierung präsentiert werden.
- Patienten, die an der Krankheit „Prosopagnosia“ leiden, können Gesichter, die ihnen früher vertraut waren, plötzlich nicht mehr erkennen und zeigen normalerweise keine andere Ausprägung von schwerwiegender Blindheit.
- Und es wird argumentiert, dass Babys so auf die Welt kommen, dass sie von Anfang an von Gesichtern angezogen werden.

Sowohl holistische als auch merkmalsbezogene Informationen sind für die Wahrnehmung und Erkennung von Gesichtern entscheidend: Aufgrund von Studien wird vermutet, dass die globalen Beschreibungen als ein sogenannter „front end“ für eine feinere, merkmalsbasierte Wahrnehmung dient (falls jedoch dominante Merkmale wie große Ohren, eine gekrümmte Nase oder starrende Auge da sind, werden die holistischen Beschreibungen eher nicht gebraucht). Deshalb sollten Entwickler von künstlichen Gesichtserkennungssystemen sowohl globale als auch lokale Merkmale für die Darstellung und Erkennung von Gesichtern miteinbeziehen!

Außerdem ist es wohl so, dass Kinder unter zehn Jahren sich unbekannte Gesichter anhand von isolierten Merkmalen im Gedächtnis halten, die sich beispielsweise von der Kleidung, von einer Brille, vom Haarstil, von einem Hut oder ähnlichem ableiten; jedoch scheinen die Kinder um das zehnte Lebensjahr ihren Erkennungsmechanismus von einem der auf diesen isolierten Merkmalen beruht zu einem mit holistischer Analyse zu verändern.

Desweiteren hat man festgestellt, dass Haare, Gesichtsumrisslinien, Augen und Mund für die Wahrnehmung und Erinnerung von Gesichtern bei frontalen Aufnahmen sehr wichtig sind und die Nase dabei eher eine ungeordnete Rolle spielt (jedoch sitzen bei der Gesichtserkennung mittels Profilen mehrere Bezugspunkte in der Nasenregion). Der obere Teil des Gesichtes ist für die Erkennung nützlicher als der untere Teil und je attraktiver ein Gesicht ist, umso besser ist die Erkennungsrate dieses Gesichtes (danach kommen die am wenigsten attraktiven Gesichtern, gefolgt von den „normalen“ Gesichtern).
Man hat auch festgestellt, dass ausgeprägte Gesichter länger im Gedächtnis bleiben und besser und schneller erkannt werden als typische Gesichter. Jedoch verhält es sich umgekehrt, wenn eine Entscheidung getroffen werden soll, ob irgendein Objekt ein Gesicht

ist oder nicht: die Erkennung dauert dabei bei einem untypischen Gesicht länger als bei einem typischen. Daraus lässt sich der Schluss ziehen, dass es für die Detektion und für die Identifikation beim Menschen verschiedene Mechanismen gibt!

Je nach spezieller Erkennungsaufgabe spielen die niederfrequenten, im Bandpass liegenden und hochfrequenten Komponenten eine unterschiedliche Rolle: für die Bestimmung des Geschlechtes reichen die niederfrequenten Komponenten alleine, während für die Identifikation die hochfrequenten Komponenten gebraucht werden. Daraus folgt, dass die niederfrequenten Komponenten im Bild zu einer globalen Beschreibung beitragen und die hochfrequenten Komponenten, die feineren Details beschreiben wie sie bei der Identifikationsaufgabe vonnöten sind. Diese Analyse der räumlichen Frequenzen bekräftigt den Einsatz von Algorithmen mit mehreren Auflösungen und mehreren Skalierungen für verschiedene Probleme im Bereich der Gesichtswahrnehmung.

Eine Studie hat ergeben, dass die rechte Gehirnhälfte wohl maßgeblich an der Gesichtserkennung beteiligt ist: die überwältigende Mehrheit der untersuchten Leute wählte das Gesicht, welches dem linken Blickfeld (LBF) gezeigt wurde und welches zuerst auf der rechten Gehirnhälfte ankommt. Das LBF war sowohl in der Antwortgeschwindigkeit wie auch in der Genauigkeit und in der Langzeitgedächtnisaufnahmefähigkeit besser.
Es ist wohl so, dass die zwei Gehirnhälften gleichzeitig verschiedene Informationstypen bearbeiten können: die Überlegenheit der rechten Gehirnhälfte in der Gesichtsverarbeitung (und Emotionsinterpretation) dürfte wohl daher kommen, dass die linke Gehirnhälfte in der Sprachverarbeitung überlegen ist.

In Abbildung 2.1 sieht man ein kleines „Puzzle“ (basierend auf einer Installation im „San Francisco Exploratorium“). Was fällt dem Leser beim Betrachten der Gesichtsbilder abgesehen von der Bildspiegelung auf?

Abbildung 2.1: „Puzzle“ als psychologischer Hintergrund

Abbildung 2.2: Auflösung des „Puzzles“ („Thatcher Illusion“)

Es handelt sich dabei um eine optische Illusion, die in der Literatur als „Thatcher Illusion“ bezeichnet wird.

In Abbildung 2.2 werden die ursprünglichen Gesichtsbilder gezeigt, um die optische Illusion aufzulösen. Im rechten Bild sind nämlich die Augen und der Mund separat gespiegelt worden. Aus irgendeinem Grund sind diese „Einzelspiegelungen“ nicht so offensichtlich, wenn der Rest des Gesichtes auch gespiegelt wird (Abb. 2.1)
Dies mag wohl daran liegen, dass Menschen ein Bild dadurch als Gesicht identifizieren indem sie nach einer speziellen Anordnung der Gesichtsmerkmale suchen ohne zu überprüfen, ob die Orientierung dieser einzelnen Bildkomponenten der Orientierung des Gesamtobjektes entspricht.

Abschließend sei noch erwähnt, dass Menschen Leute ihrer eigenen Rasse besser erkennen als Leute einer anderen Rasse, das heißt, die Charakteristiken eines „Durchschnittsgesichtes“ sind für verschiedene Rassen unterschiedlich! Außerdem ist beispielsweise in Japan die Mehrheit der weiblichen Gesichtsmerkmale heterogener als die männlichen. Die schlechtere Erkennung von anderen Rassen ist aber kein psychophysikalisches Problem (zum Beispiel bedingt durch unterschiedliche Reflektionseigenschaften der verschiedenen Hautfarben), sondern eher ein psychosoziales (beispielsweise Vorurteile oder die ungewohnte Klasse von Reizen).

2.3 Kategorisierung der Gesichtsdetektion

Nach dem Exkurs in die Psychologie wollen wir nochmals die im Überblick (2.1) bereits angesprochene Gesichtsdetektion näher betrachten und eine genaue Kategorisierung von Ansätzen vornehmen, die ein einziges Bild für die Detektion heranziehen. Die Ausführungen stammen dabei hauptsächlich aus [2].

Als erster Schritt eines jeden Systems zur Gesichtsverarbeitung werden robuste und effiziente Algorithmen für die Gesichtsdetektion benötigt. Ist ein einziges Bild gegeben, so ist das Ziel, alle Bildregionen zu identifizieren, die ein Gesicht darstellen und das ganze sollte unabhängig von der dreidimensionalen Position, Orientierung und Beleuchtungseigenschaft sein. Dies ist natürlich eine sehr schwierige Aufgabe, da Gesichter nicht starr sind und sowohl eine hohe Variabilität in Größe, Form, Farbe und Textur besitzen als auch eine große Veränderung in Skalierung, Position, Orientierung (aufrecht, rotiert) und Haltung (frontal, seitlich) aufweisen. Dazu kommt, dass Gesichtsausdrücke, Verdeckungen und Beleuchtungsbedingungen ebenfalls die Gesamterscheinung des Gesichtes beeinflussen können.

Die Gesichtsdetektion lässt sich folgendermaßen definieren: *Das Ziel der Gesichtsdetektion ist es, bei einem beliebigen Bild zu entscheiden, ob Gesichter dargestellt sind und wenn ja, wo genau und in welchem Ausmaß sie vorkommen*.
Sie kann also als ein zwei Klassen-Problem angesehen werden: Bildregionen werden als Gesicht oder als „Nichtgesicht“ klassifiziert (jeweils mit einer großen Variabilität innerhalb der Klasse). Die Dimension des Merkmalraums ist dabei extrem groß (zum Beispiel die Anzahl der Pixel in normalisierten Trainingsbildern) und die Klassen für Gesichtsbilder und Nichtgesichtsbilder werden eindeutig durch multimodale Verteilungsfunktionen charakterisiert, wobei sinnvolle Entscheidungsgrenzen meist nichtlinear im Raum der erwarteten Bilder sind.

Die Herausforderungen bei der Gesichtsdetektion sind unter anderem:

- Gesichtshaltung: Dazu gehören Fragen wie Kamera und Gesicht zu einander ausgerichtet sind (frontal, 45 Grad, seitliches Profil, auf dem Kopf stehend) und dadurch resultierend ob Gesichtsmerkmale (wie ein Auge oder die Nase) teilweise oder ganz verdeckt sind.
- Existenz von strukturellen Komponenten: Gesichtsmerkmale (wie (Oberlippen-)Bärte und Brillen) können da sein oder auch nicht und es gibt eine große Variabilität unter diesen Komponenten in Bezug auf Form, Farbe und Größe.
- Mienenspiel
- (teilweise) Verdeckung
- Bildorientierung: Gesichtsbilder unterscheiden sich je nach Rotation um die optische Achse der Kamera.
- Aufnahmebedingungen: Faktoren wie die Beleuchtungsmethode (zum Beispiel der Grad der Lichtintensität) und die Kameracharakteristiken (zum Beispiel die verwendete Linse) beeinflussen die Darstellung des Gesichtes im Bild.

Eng verknüpft mit der Gesichtsdetektion sind die folgenden Themen:

- *Gesichtslokalisierung*: Hier wird die Position eines einzigen Gesichtes bestimmt (es ist also sozusagen eine vereinfachte Version der Gesichtsdetektion).
- *Detektion von Gesichtsmerkmalen*: Das untersuchte Thema dieser Diplomarbeit stellt die Existenz und den Ort von bestimmten Merkmalen im Bild fest (dazu zählen beispielsweise Augen, Augenbrauen, Nase, Nasenlöcher, Mund, Lippen, Ohren); wieder mit der Voraussetzung, dass es nur ein einziges Gesicht im Bild gibt.
- *Gesichtserkennung oder Gesichtsidentifikation*: Wie im Unterkapitel 2.1 beschrieben wird ein Eingabebild mit einer Datenbank verglichen und ein „Treffer" beziehungsweise ein „Nichttreffer" gemeldet.
- *Gesichtsauthentifizierung*: Dabei wird die Identität einer Person mittels eines Eingabebildes untersucht.
- *Gesichts-Tracking*: Wie in den Vorarbeiten dieser Diplomarbeit werden Position und möglicherweise auch Orientierung eines Gesichtes in einer Bildsequenz kontinuierlich (in Echtzeit) geschätzt.
- *Erkennung von Gesichtsausdrücken*: Hier beschäftigt man sich mit der Bestimmung von Gemütszuständen bei Menschen (zum Beispiel glücklich, traurig, empört).

Die Ansätze zur Gesichtsdetektion in einem einzigen Bild lassen sich nach [2] in vier Kategorien einteilen:

- *Wissensbasierte Methoden (top-down)*: Regelbasierte Methoden enthalten Wissen über den Aufbau eines „typischen" Gesichtes (oft beinhalten die Regeln Beziehungen zwischen Gesichtsmerkmalen) und dienen hauptsächlich der Gesichtslokalisierung.
- *Merkmalsbasierte Methoden (bottom-up)*: Sie versuchen strukturelle Merkmale zu finden (die selbst bei variierenden Bedingungen existieren) und dienen ebenfalls vorwiegend der Gesichtslokalisierung.
- *„Template-Matching" Methoden*: Mehrere „Standardpatterns" (also Schablonen) für Gesichter beziehungsweise Gesichtsmerkmale werden gespeichert und mit Eingabebildern korreliert; dies kann sowohl für die Gesichtslokalisierung als auch für die Gesichtsdetektion verwendet werden.
- *„Appearance-based" Methoden*: Hier werden die Modelle/„Templates" im Gegensatz zur vorangehenden Kategorie über Trainingsbilder gelernt, wodurch man eine repräsentative Variabilität der Gesichter abdeckt; diese Methoden werden meist für die Gesichtsdetektion eingesetzt.

In den folgenden Abschnitten werden die einzelnen Kategorien detailliert beschrieben und veranschaulicht.

2.3.1 Wissensbasierte Methoden (top-down)

Hier werden zuerst Gesichtsmerkmale extrahiert und „Gesichtskandidaten" dann aufgrund von Regeln (über die relative Distanz und Position der Gesichtsfeatures zueinander) identifiziert. Die Regeln werden anhand menschlichen Wissens über die Charakteristika der Gesichtsregionen (zum Beispiel über die Intensitätsverteilung und -differenz) aufgestellt. Dabei hat man bei der Umsetzung dieses Wissens in Regeln zum einen das Problem der zu strikten Regeln (eigentliche Gesichter werden nicht erkannt) und zum anderen das Problem der zu allgemeinen Regeln (irgendwelche Bildbereiche werden fälschlicherweise als Gesicht detektiert).

Am wenigsten Probleme hat man bei Heuristiken, die frontale Gesichter in einer „geordneten“ Szene detektieren sollen.

In [22] wird ein hierarchisches wissensbasiertes System mit drei Regelebenen verwendet (dabei wird eine sogenannte Multiauflösungs-Hierarchie von Bildern durch Mittelung und Unterabtastung erzeugt (siehe auch [100] zur Erklärung dieser Begriffe)):

- Ebene 3: Bei geringster Bildauflösung wird einfach nach „Gesichtskandidaten“ gesucht, indem man ein Fenster über das Eingabebild wandern lässt und an jeder Position bestimmte Regeln anwendet.
- Ebene 2: Die auf Ebene 3 gefundenen Kandidaten werden zuerst (lokal) histogrammnormalisiert und dann nach Kanten abgesucht.
- Ebene 1: Die übrig gebliebenen Kandidaten werden schließlich noch mit anderen Regeln untersucht, die sich an den Merkmalen (wie Augen und Mund) orientieren.

Dabei sind die Regeln auf höherer Ebene allgemeine Beschreibungen wie ein Gesicht aussieht, während die Regeln auf niedrigerer Ebene Details der Gesichtmerkmale heranziehen. Diese Vorgehensweise, um den Rechenaufwand zu reduzieren, wird im Englischen „coarse-to-fine“- beziehungsweise „focus-of-attention“-Strategie genannt.

In [23] werden Gesichtsmerkmale mittels einer Projektionsmethode lokalisiert, die Professor Kanade schon Anfang der siebziger Jahre in seiner Doktorarbeit für die Bestimmung der Gesichtskontur verwendet hat ([24]): Das horizontale Profil (das heißt die horizontale Projektion der Intensitätswerte) eines Eingabebildes wird zuerst bestimmt und von den zwei lokalen Minima, die man durch die Erkennung von abrupten Änderungen erhält, wird angenommen, dass sie der linken und rechten Kopfseite entsprechen. Ähnlich verhält es sich mit dem vertikalen Profil (das heißt der vertikalen Projektion der Intensitätswerte); dabei entsprechen die lokalen Minima beziehungsweise Scheitelpunkte idealerweise den Mundlippen, der Nasenspitze und den Augen.
Darauf aufbauend werden Regeln für die Detektion von Augenbraue/Augen, Nasenlöcher/Nase und Mund verwendet, um diese Kandidaten zu überprüfen.
Diese Methode hat aber Schwierigkeiten, wenn Gesichter auf komplexem Hintergrund oder mehrere Gesichter auftreten. Außerdem funktioniert sie nur gut, wenn das Fenster, auf dem die Profile berechnet werden, passend „sitzt“, um irreführende Interferenzen zu vermeiden.

2.3.2 Merkmalsbasierte Methoden (bottom-up)

Da Menschen selbst bei verschiedenen Perspektiven und Beleuchtungsbedingungen normalerweise keine Probleme haben, Objekte beziehungsweise Gesichter zu erkennen, muss es Eigenschaften oder Merkmale geben, die invariant bei solchen Veränderungen sind! Gesichtsmerkmale (wie Augenbrauen, Augen, Nase, Mund und Haarkontur) werden normalerweise mit Hilfe eines Kantendetektors (wie der Canny-Operator) extrahiert. Basiert auf den extrahierten Merkmalen, wird dann ein *statistisches Modell* aufgebaut, um die Beziehungen zwischen diesen Merkmalen zu beschreiben und um Gesichtshypothesen zu bestärken beziehungsweise abzuschwächen. Ein große Problem dieser merkmalsbasierten Algorithmen ist aber, dass die Bildmerkmale aufgrund von Beleuchtung, Rauschen und Verdeckung sehr verändert beziehungsweise überhaupt nicht da sein können.

Man gliedert bei den Merkmalen nochmals in vier Fälle:
- Gesichtsmerkmale
- Textur
- Hautfarbe
- Kombination mehrerer Merkmale

Anstatt nur Kanten zu verwenden, wird in [25] eine einfache Gesichtsdetektion beschrieben, die auf sogenannten „*Blobs*“ und linearen Sequenzen von ähnlich orientierten Kanten (sogenannte „Streaks“) beruht: Ihr Gesichtsmodell besteht aus zwei dunklen „Blobs“ und drei hellen „Blobs“, um einerseits die Augen und andererseits die Wangenknochen und die Nase zu repräsentieren (die Detektion der „Blobs“ geschieht dabei auf einem mit dem Laplaceoperator gefilterten Bild niedriger Auflösung). Die „Streaks“ werden vom Modell für die Repräsentation der Kontur des Gesichtes, der Augenbrauen und der Lippen benutzt.
Das Eingabebild wird zuerst mittels zweier Dreieckskonfigurationen, die die räumliche Anordnung der „Blobs“ wiedergeben, nach Kandidaten „abgescannt“ und ein Gesicht ist dann detektiert, wenn die „Streaks“ innerhalb eines Kandidaten identifiziert wurden.

In [26] wurde eine Methode für das Finden von Gesichtsmerkmalen und Gesichtern in Graustufenbildern entwickelt: Nach einer Bandpass-Filterung werden sogenannte „*morphologische Operatoren*“ (siehe [100]) angewendet, um Regionen mit einer hohen Intensität, die eine ganze bestimmte Form haben (zum Beispiel Augen), hervorzuheben. Das Histogramm des vorverarbeiteten Bildes zeigt typischerweise einen ausgeprägten „Peak“ und anhand des Wertes dieses „Peaks“ und seiner Breite werden adaptive Schwellwerte bestimmt, um zwei Binärbilder zu erzeugen. In diesen beiden Binärbilder werden mit Hilfe von zusammenhängenden Komponenten die Gebiete der möglichen Gesichtmerkmale identifiziert. Schließlich werden Kombinationen solcher Gebiete mit Klassifikatoren ausgewertet, um zu entscheiden, ob ein Gesicht da ist oder nicht.

Eine probabilistische Methode, basierend auf lokalen Merkmalsdetektoren und zufälligem Graphen-Matching wird in [27] beschrieben, um ein Gesicht in einer komplexen Szene zu finden. Die Motivation bestand darin, das Problem der Gesichtslokalisierung als ein Suchproblem zu formulieren! Ziel ist es, die Anordnung bestimmter Gesichtsmerkmale zu finden, die einem Gesichtsmuster am wahrscheinlichsten entspricht; fünf Merkmale (zwei Augen, zwei Nasenlöcher und die Nasen-Lippen Partie) werden dabei für die Beschreibung eines typischen Gesichtes verwendet.
Für jedes Paar von Gesichtsmerkmalen des gleichen „Typs“ (zum Beispiel linkes/rechtes Augen-Paar) wird ihre relative Distanz zueinander berechnet und anhand vieler Bilder werden diese Distanzen dann durch eine Gauß-Verteilung modelliert. Auf ähnliche Weise wird ein komplettes „Template“ für die einzelnen Gesichtsmerkmale erzeugt, in dem die Ergebnisse mehrerer multiorientierter und multiskalierter Gauß-Filter erster Ordnung (auf die Pixel innerhalb der Gesichtsmerkmale) über viele Gesichter in einer Datenbank gemittelt werden.
Die erwartete Position der anderen Merkmale wird mit einem statistischen Modell über die gegenseitigen Distanzen geschätzt und die Konstellation, die einem Gesicht am nächsten kommt, wird bestimmt: Dabei wird das Auffinden der besten Konstellation als zufälliges Graphen-Matching formuliert, wobei die Knoten des Graphen den Merkmalen in einem Gesicht entsprechen und die Kanten die Abstände zwischen den verschiedenen Merkmalen widerspiegeln.

In [28] und [29] wird zuerst ein Gauß-Filter zweiter Ordnung auf ein Rohdatenbild angewendet und lokale Maxima in der Filterantwort deuten auf mögliche Positionen von Gesichtsmerkmalen hin. Danach werden die Kanten um diese interessanten Punkte näher untersucht und in Regionen gruppiert. Die Gruppierung erfolgt dabei nach Nachbarschaft und Ähnlichkeiten in Orientierung und Stärke der Kanten. Die messbaren Charakteristika einer Region (wie Kantenlänge, Kantendicke und Intensitätsverteilung) werden berechnet und in einem Merkmalsvektor gespeichert. Hat man eine Trainingsmenge an Gesichtsmerkmalen, so kann der Mittelwert und die Kovarianz-Matrix jedes Gesichtsmerkmalvektors berechnet werden. Eine Bildregion wird genau dann ein Kandidat für ein Gesichtsmerkmal, wenn die sogenannte „*Mahalanobis Distanz*“ zwischen den korrespondierenden Merkmalsvektoren unterhalb eines Schwellwertes liegt. Die markierten Merkmale werden dann anhand von Modellwissen über die Anordnung zueinander weiter gruppiert: jedes Gesichtsmerkmal und jede Gruppierung wird schließlich mit einem *Bayes'schen Netzwerk* ausgewertet. Durch diesen Ansatz können Gesichter unter verschiedenen Orientierungen und Haltungen detektiert werden!

Benutzt man als Merkmal die Textur eines Gesichtes, die gewöhnlich über statistische Merkmale zweiter Ordnung (SGLD) berechnet wird, dann hat man den großen Vorteil, dass man auch Gesichter findet, die nicht aufrecht sind oder bei denen Bärte beziehungsweise Brillen auftreten. Manchmal wird auch Farbinformation (wie orange-ähnliche Bildteile) in das Gesichtstextur-Modell mitaufgenommen.

Die Hautfarbe alleine bietet sich natürlich auch zur Differenzierung der Gesichter von anderen Objekten im Bild an. Obwohl verschiedene Menschen unterschiedliche Hautfarbe besitzen, haben mehrere Studien gezeigt, dass der Hauptunterschied zum großen Teil in der Intensität statt in der Chrominanz-Information liegt. Um deshalb Pixel als Hautpixel zu „labeln“, das heißt zuzuordnen, wurden in der Vergangenheit mehrere Farbräume untersucht; darunter der RGB, der normalisierte RGB, der HSV (oder HSI), der YcrCb, der YIQ, der YES, der CIE XYZ, und der CIE LUV (zu einer näheren Erläuterung sei auf meine Studienarbeit [101] verwiesen).
Statt nichtparametrische Methoden (wie Histogramme, Parzen-Fenster und k-nächster-Nachbar) zu verwenden, kommen für die Modellierung der Hautfarbe oft (unimodale) Gauß-/Normalverteilungen (parametrische Methoden) und Gauß-Mixturen (semiparametrische Methoden) zum Einsatz. Die Motivation für das Verwenden einer Gauß-Mixtur liegt in der Feststellung, dass das Farbhistogramm für die Haut, das man bei Leute unterschiedlicher ethnischer Herkunft erhält, keine unimodale sonder eher eine multimodale Verteilung aufweist.
Es sei noch erwähnt, dass Histogramm-Modelle gegenüber Mixtur-Modellen für die Hautdetektion zwar genauer aber auch rechenaufwendiger sind! Außerdem reicht die Hautfarbe alleine nicht, um Gesichter zu detektieren oder zu tracken (die Farberscheinung allgemein hängt nämlich sehr vom Blickwinkel und von Beleuchtungsbedingungen ab); deshalb werden modulare Systeme, die eine Kombination von Formanalyse, Farbsegmentierung und Bewegungsinformation für die Bestimmung und das Tracking des Kopfes und des Gesichtes in einer Bildsequenz anwenden, immer beliebter.

Schließlich gehören in diese Kategorie der merkmalsbasierten Methoden noch die Ansätze, die mehrere (Gesichts-)Merkmale kombinieren. Dabei werden oft globale Merkmale (wie Hautfarbe, Größe und Form) dazu benutzt, um „Gesichtskandidaten“ zu finden, und dann werden diese Kandidaten mit lokalen (detaillierten) Merkmalen (wie Augenbrauen, Nase und Haar) nochmals überprüft.
Ein typischer Ansatz fängt mit der Detektion beziehungsweise Segmentierung von hautähnlichen Regionen an (zum Beispiel im HSV-Raum); danach, werden diese hautähnlichen Pixel mittels einer Analyse von Zusammenhangskomponenten oder eines Clustering-Algorithmus gruppiert (oft durch Regionenwachstum bei grober Auflösung).

Falls die Form einer solchen zusammenhängenden Region über geometrische Momente einer Ellipse oder einem Oval entspricht, dann wird sie ein „Gesichtskandidat“. Zum Schluss werden lokale Merkmale für die Verifikation benutzt: beispielsweise erhält man Auge-Augenbraue- und Nase-Mund-Merkmale über horizontale Kanten oder Augen und Mund alleine über die Feststellung, dass sie dunkler sind als der Rest des Gesichtes.

Detektionsmethoden, die im Gegensatz zu pixelbasierten Methoden auf der Struktur, der Farbe und der Geometrie aufsetzen, können Gesichter selbst bei verschiedenen Orientierungen und bei Anwesenheit von Bärten oder Brillen finden.

In [30] und [31] werden sowohl die (lokale) Symmetrie von Gesichtern als auch die Hautfarbenklassifizierung (zum Beispiel im YES Farbraum) angewendet: Eine ellipsenförmige „Gesichtsschablone“ wird dabei zuerst benutzt, um die Ähnlichkeit der Gesichtsfarbenregionen basierend auf der sogenannten Hausdorff Distanz zu bestimmen; dann werden die Augenmittelpunkte durch mehrere Kostenfunktionen, die die inhärente Symmetrie des Gesichtes und der Augen ausnutzen, lokalisiert; schließlich werden noch die Nasenspitze und der Mundmittelpunkt anhand der Augendistanz räumlich festgelegt. Dieser Ansatz funktioniert aber nur gut bei einzelnen frontalen Gesichtern und wenn beide Augen sichtbar sind!

2.3.3 „Template-Matching“ Methoden

Hier werden die Korrelationswerte zwischen dem Eingabebild und (normalerweise frontalen) Standard-Muster für die Gesichtskontur, die Augen, die Nase und den Mund unabhängig voneinander berechnet, um „Kandidaten“ nach dem Prinzip des „focus of attention“ beziehungsweise „region of interest“ (ROI) zu erhalten. Zum Einsatz kommen gewöhnlich Kantendetektoren (wie beispielsweise der sogenannte „Marr-Hildreth Operator“ oder der „Laplace-Operator“); es gibt aber auch noch andere Ansätze, um allgemein Liniensegmente im Bild zu finden: über die „Principal Component Analysis (**PCA**)“, angewendet auf Beispielgesichter, erhält man sogenannte „Eigensilhouetten“, also eine Menge an Basiskonturen von typischen Gesichtern, die dann zusammen mit der allgemeinen Hough-Transformation für die Gesichtslokalisierung benutzt werden.

Man unterscheidet in dieser Kategorie im Prinzip zwei Fälle:
- vordefinierte Schablonen (im Englischen „predefined templates“ genannt)
- und verformbare Schablonen (im Englischen „deformable templates“ genannt)

Obwohl die „manuell“ oder über eine Funktion bestimmten vordefinierten Schablonen leicht zu implementieren sind, haben sie den großen Nachteil, dass sie (selbst bei hierarchischem „Template-Matching“) nicht mit Variationen in Skalierung, Haltung und Form umgehen können und dadurch für die Gesichtsdetektion eher nicht geeignet sind.

Als Lösung für dieses Problem bieten sich verformbare (parametrisierbare) Schablonen und „Unterschablonen“ an.
In [32] werden Gesichtsmerkmale durch parametrisierte „Templates“ beschrieben und die beste Anpassung des (a priori) elastischen Modells erfolgt über eine Minimierung der Energiefunktion für die Parameter. Nachteil an dieser Methode ist wie schon im Überblicksunterkapitel erwähnt, dass die verformbaren Schablonen in der Nähe des interessierenden Objektes initialisiert werden müssen; ansonsten kann es sein, dass sie sich auf anderen Objekten „festsetzen“.
Weiterhin werden auch noch:

- „*active contour models/active shape models*“ (*ASM*) wie die bekannten “*snakes*”
- und “*point distribution models*“ (*PDM*)

für die Detektion eingesetzt.

2.3.4 „Appearance-based“ Methoden

Im Gegensatz zu den vorangehenden „Template-Matching” Methoden, bei denen die Schablonen von Experten mehr oder weniger vorgegeben werden, werden die „Templates“ in dieser Kategorie direkt aus Beispielen in Bildern gelernt! Dabei basieren die nun betrachteten Methoden auf Techniken der *statistischen Analyse* und des *maschinellen Lernens*: es sollen dadurch relevante Charakteristika eines Gesichts- und „Nichtgesichts“-Bildes beziehungsweise Teilbildes in der Form von Verteilungsmodellen oder Unterscheidungsfunktionen gefunden werden. Dabei findet aus Gründen der Berechenbarkeit und der Detektionsleistung oft eine Dimensionsreduktion statt.
Die Verteilungsmodelle sind in ein wahrscheinlichkeitstheoretisches „Framework“ eingebunden: ein Bildvektor oder ein (daraus abgeleiteter) Merkmalsvektor wird als Zufallsvariable x gesehen, die für Gesichter und „Nichtgesichter“ durch die klassenbedingten Dichtefunktionen p(x|Gesicht) und p(x|„Nichtgesicht“) charakterisiert wird. Eine Klassifizierung nach Bayes beziehungsweise eine sogenannte „Maximum Likelihood“-Methode (ML-Methode) wird dann für die Einteilung von Bildregionen in Gesicht- oder „Nichtgesicht“-Region verwendet. Da eine direkte Implementierung des Bayes’schen Ansatzes hier nicht möglich ist (x ist hochdimensional, die klassenbedingten Dichtefunktionen sind multimodal und bis heute ist unklar, ob es „natürliche“ parametrisierte Formen dieser Funktionen gibt), wird viel Arbeit in die Bestimmung von empirisch validierten parametrischen und nichtparametrischen Approximationen von p(x|Gesicht) und p(x|„Nichtgesicht“) gesteckt.
Ein alternativer Ansatz, um Gesichter und „Nichtgesichter“ zu trennen, sind die Diskriminanz- beziehungsweise Unterscheidungsfunktionen (wie eine Trenn(hyper-)ebene oder eine Schwellwertfunktion). Dabei werden normalerweise die Bildmuster in einen Vektorraum mit niedrigerer Dimension projiziert und dann wird eine Diskriminanzfunktion für die Klassifizierung bestimmt (gewöhnlich über Abstandsmetriken) oder mittels mehrschichtiger Neuronaler Netze wird eine nichtlineare Trennhyperebene definiert (man verwendet auch sogenannte „Support Vector Machines“ (SVMs) und andere Methoden mit

„Kernelfunktionen“, wobei diese die Muster implizit in einen höherdimensionalen Raum projizieren).

Die Ansätze in dieser Kategorie der Gesichtsdetektion lassen sich laut [2] in neun Untergruppen gliedern, auf die im folgenden näher eingegangen wird:

- „Eigengesichter“
- verteilungsbasierte Methoden
- Neuronale Netze (NNe)
- „Support Vector Machines“ (SVMs)
- Sparse Network of Winnows (SnoW)
- Naiver Bayes Klassifikator
- “Hidden Markov”-Modelle (HMMe)
- informationstheoretischer Ansatz
- induktives Lernen

Der Ansatz mit „Eigengesichter“ beruht auf der „Karhunen-Loève Transformation“ (in der Literatur auch „Principal Component Analysis“ (PCA) und „Hotelling Transformation“ genannt) und wurde bereits im Überblick (2.1) beschrieben. In [33] wird gezeigt, dass sich Bilder von Gesichtern auf lineare Weise durch eine relativ geringe Anzahl an Basisbildern zusammensetzen lassen. Hat man eine Menge von Trainingsbildern (m auf n Pixel) oder auch Trainingsvektoren (der Größe m x n) gegeben, dann werden die Basisvektoren, die einen optimalen Unterraum (hier den „Gesichtsraum“) aufspannen, dadurch bestimmt, dass der mittlere quadratische Fehler zwischen der Projektion dieser Trainingsbilder auf den Unterraum und den ursprünglichen Trainingsbildern minimiert wird. Die Menge der optimalen Basisvektoren sind dann die „Eigenbilder“ (beziehungsweise hier „Eigengesichter“), da sie einfach die Eigenvektoren der Kovarianz-Matrix sind, die aus der vektorisierten Form der Gesichtsbilder in der Trainingsmenge berechnet wird. Um ein Gesicht in einer Szene zu detektieren, wird für alle Bildpositionen der Abstand zwischen der jeweiligen Bildregion und des „Gesichtsraumes“ berechnet, wodurch man schließlich eine Art von „Gesichtskarte“ bekommt, die einem Aufschluss über die wahre Position des Gesichtes gibt. Viele Arbeiten auf dem Gebiet der Gesichtsdetektion, Gesichtserkennung und Merkmalsextraktion (wie zum Beispiel [12]) haben diesen Ansatz der Zerlegung in Eigenvektoren und des Clusterns von Bildprojektionen übernommen.

In [34] wurde ein verteilungsbasiertes System entwickelt, dass aus zwei Komponenten besteht: einerseits verteilungsbasierte Modelle für Gesichter und „Nichtgesichter“, die aus positiven und negativen Beispielbildern gelernt werden, und andererseits ein mehrschichtiges Perzeptron (MLP = multilayer perceptron) für die Klassifizierung. Mittels eines modifizierten „K-Means-Algorithmus“ werden sechs Gesichts- und sechs „Nichtgesichts“-Cluster bestimmt, die dann jeweils durch eine multidimensionale Gauß-Funktion repräsentiert und approximiert werden. Auf ein Eingabebild und die „prototypischen“ Cluster werden zwei Abstandsmetriken angewendet:

- einmal im Unterraum mit niedrigerer Dimension, der durch die größten Eigenvektoren bestimmt wird, die normalisierte Mahalanobis Distanz zwischen der Projektion des Testmusters und des Clusterschwerpunktes, der dem Mittelwert der Gauß-Verteilung entspricht
- und zum anderen der Euklidsche Abstand zwischen dem (originalen) Testmuster und seiner Projektion in den genannten Unterraum

Bei der Clusterbestimmung ist die Auswahl von „Nichtgesichts“-Beispielen im Vergleich zu repräsentativen Beispielen für (frontale) Gesichtsmuster deutlich schwieriger, da die Grenzen des Vektorunterraumes für typische Gesichter dadurch definiert und idealerweise verbessert angepasst werden. Eine Möglichkeit ist beispielsweise die Verwendung einer sogenannten „bootstrap“-Methode: „False Positives“, das heißt bisher noch nie gesehene Eingabemuster, die fälschlicherweise als Gesicht klassifiziert wurden, werden der Trainingsmenge als neue „Nichtgesichts“-Beispiele hinzugefügt!

In [35] werden zwei interessante Methoden zur Detektion vorgestellt:
- eine Mixtur an „Faktoranalyzern“ (MFA)
- und „Fisher’s lineare Diskriminante“ (FLD)

Faktoranalyse (FA) ist eine statistische Methode, um die Kovarianz-Struktur hochdimensionaler Daten mit Hilfe einer kleinen Anzahl innewohnender Variablen zu modellieren und ist in vielerlei Hinsicht der PCA sehr ähnlich.
Aber im Gegensatz zur FA ist die PCA aus Sicht der Klassifikation suboptimal, wenn die Beispiele eine bestimmte Struktur haben: die mittels PCA projizierten Samples aus verschiedenen Klassen können nämlich oft „verschmiert“ werden. Die PCA definiert kein geeignetes Dichtemodell für die Daten, da die Kosten, um einen Datenpunkt zu kodieren, überall entlang der Achsen des Hauptkomponentenraumes gleich sind. Außerdem ist die PCA gegenüber unabhängigem Rauschen in den Merkmalen der Daten nicht robust, da die Hauptkomponenten die Varianz der Eingabedaten maximieren und dadurch ungewollte Variationen verstärken.
Auch die zweite Methode, also FLD, um Samples von einem hochdimensionalen Raum an Bildern in einen Merkmalsraum mit niedrigerer Dimension zu projizieren, ist der PCA in der Musterklassifikation überlegen, da FLD versucht, die Projektionsrichtung zu finden, die die Daten am besten trennt.

Natürlich kommen bei Ansätzen zur Gesichtsdetektion wie so oft bei der Mustererkennung auch Neuronale Netze zum Einsatz. Der Vorteil ist dabei, dass man ein System so trainieren kann, dass es die komplexe klassenbedingte Dichtefunktion der Gesichtsmuster im Prinzip vollständig lernen kann. Nachteil ist aber, dass die Netzwerk-Architektur sehr stark angepasst werden muss (zum Beispiel die Anzahl der Schichten und Neuronen, die Lernrate und so weiter), um eine außergewöhnliche Leistung zu bekommen.
Beliebt ist die Anwendung mehrerer Neuronaler Netze, beispielsweise eines für jede Skalierung eines Eingabebildes. Es folgt dann die Bildung einer Gesamtentscheidung (zum Beispiel über logische Operatoren wie „und“/„oder“) aus den einzelnen Detektionsergebnissen, um die Erkennungsleistung zu verbessern.

„Support Vector Machines“ werden für die Detektion von Gesichtern ebenfalls verwendet. Es handelt sich dabei um ein neues Paradigma: Während die meisten Methoden für das Trainieren eines Klassifikators (wie beispielsweise Bayes oder Neuronale Netze) darauf aufgebaut sind, den Trainingsfehler (sogenanntes „empirisches Risiko“) zu minimieren, operieren SVMs auf einem anderen Induktionsprinzip, nämlich auf der Minimierung des „strukturellen Risikos“; dabei wird versucht, die obere Grenze des zu erwartenden Generalisierungsfehlers zu minimieren. Eine SVM ist also ein linearer Klassifikator, bei dem die trennende Hyperebene so gewählt wird, dass der erwartete Klassifizierungsfehler

der bisher noch nie gesehenen Testmuster minimiert wird! Die Berechnung dieser optimalen Hyperebene ist aber sowohl speicher- als auch zeitaufwendig, weshalb man effiziente Methoden für das Trainieren einer SVM bei großen Problemen entwickelt hat.

Im Gegensatz zu Methoden, die die globale Erscheinung eines Gesichtes modellieren, beschreiben Schneiderman und Kanade in [36] einen naiven Bayes Klassifikator, um die gemeinsame Wahrscheinlichkeit der lokalen Erscheinung und Position der Gesichtsmuster beziehungsweise Unterregionen des Gesichtes bei verschiedenen Auflösungen zu schätzen.

Dabei wird bei jeder Skalierung ein Gesichtsbild in vier rechteckige Unterregionen aufgeteilt; diese Unterregionen werden dann mittels PCA in einen Vektorraum mit niedrigerer Dimension projiziert. Zwischen den Unterregionen gibt es keine statistische Abhängigkeit. Es gibt zwei Gründe, einen naiven Bayes Klassifikator zu verwenden:

- einerseits erhält man eine bessere Schätzung der bedingten Dichtefunktionen dieser Unterregionen
- und andererseits hat man so eine funktionale Form der posteriori Wahrscheinlichkeit, um die gemeinsamen Statistiken von lokaler Erscheinung und Position des Objektes abzudecken

Im Gegensatz zu „Template-Matching“ Methoden, bei denen man auf die genaue Ausrichtung der Augen, der Nase und so weiter zu einem Referenzpunkt achten muss, wird bei „Hidden Markov Modellen“ versucht, Gesichtsregionen mit Zuständen zu verknüpfen. Ein Gesicht wird dann durch einen scannenden Prozess (mit Überlappung) erkannt, bei dem die Regionen in einer geeigneten Reihenfolge durchlaufen werden (zum Beispiel von oben nach unten und von links nach rechts). Das heißt man erhält Korrespondenzen: ein Zustand entspricht beispielsweise der Stirn und ein anderer Zustand den Augen. In dem man die (räumliche) Struktur eines Gesichtes berücksichtigt (wie „Haare-Stirn-Augen-Mund“) hat man natürlich auch Randbedingungen an die Zustandsübergänge. Die Parameteranpassung beim Training der HMMs (also das Bestimmen der Zustandsübergangswahrscheinlichkeiten und daraus resultierend der optimalen Zustandssequenz) erfolgt normalerweise mit der Standard Viterbi-Segmentierungsmethode und dem Baum-Welch Algorithmus.

In der Gruppe der informationstheoretischen Ansätze wird in [37] die Kullback relative Information angewendet, um den Markov Prozess zu finden, der die informationsbasierte Unterscheidung zwischen der Gesichts- und „Nichtgesichts“-Klasse maximiert. Dabei werden aus einer Trainingsmenge die sogenannten „most informative pixels“ (MIP) gewählt: das sind Pixel, die die beste Klassentrennung anhand von Wahrscheinlichkeitsfunktionen liefern. Es hat sich herausgestellt, dass sich diese MIP-Verteilungen vor allem auf die Augen- und Mundregionen fokusieren und die Nasenregion irrelevant ist. Um Gesichter zu detektieren, wird schließlich ein Fenster über das Eingabebild geführt: jedes Mal wird dabei die sogenannte „distance from face space“ (DFFS) berechnet, damit eine Einteilung des Fensterinhaltes in Gesicht oder „Nichtgesicht“ erfolgen kann.

Zuletzt wollen wir noch auf die Gruppe des induktiven Lernens näher eingehen.
In [38] wird der sogenannte „C4.5-Algorithmus“ von Professor Quinlan angewendet, um einen Entscheidungsbaum als Klassifikator für Gesicht und „Nichtgesicht“ anhand von positiven und negativen Beispielen zu lernen. Die Blätter geben dabei die Klassenzugehörigkeit an und die einzelnen Knoten entsprechen einem Test, der auf einem einzigen Attribut ausgeführt wird.
In [39] wird eine Methode vorgestellt, um Gesichter mit Hilfe des sogenannten „Find-S-Algorithmus“ ([40]) von Professor Mitchell zu lernen. Dabei wird angenommen, dass die Verteilung von Gesichtsmustern (p(x|face)) durch eine Menge von Gauß’schen Clustern approximiert werden kann und dass der Abstand einer „Gesichtsinstanz“ zu einem der Clusterschwerpunkte kleiner sein sollte als ein Bruchteil der maximalen Distanz der Punkte in diesem Cluster zum dazugehörigen Schwerpunkt. Der „Find-S-Algorithmus“ wird dann dazu benutzt, um den Schwellwert für den Abstand zu lernen, damit Gesichter und „Nichtgesichter“ unterschieden werden können. Im Gegensatz zu anderen Methoden, die sowohl positive als auch negative Beispiele verwenden, kommen hier keine negativen Beispiele, also „Nichtgesichter“ zum Einsatz!

2.4 Evaluierungsprinzipien

Eine der wichtigsten Erkenntnisse, die man auch aus ähnlichen Problemen (zum Beispiel bei OCR (Optical Character Recognition) das heißt optische Zeichenerkennung und bei Fingerabdruck-Klassifikation) ziehen kann, ist sicherlich, dass große Mengen an Testbildern für eine vollwertige Evaluierung des Systems zur Gesichtserkennung unverzichtbar sind.

Außerdem ist es extrem wichtig, dass die Beispiele (im Englischen „samples“ genannt) statistisch gesehen so ähnlich wie möglich zu den Bildern sind, die nachher in der Anwendung tatsächlich auftreten. Die Bewertung sollte differenziert nach Kosten für Erkennungsfehler erfolgen und neben der reinen Erkennung sollte ebenfalls der Rückweisungsfehler des Systems untersucht werden.

Man sollte sich im klaren sein, dass die Mustererkennung nicht von einer formellen Theorie getragen wird (wie beispielsweise die Physik oder die Mathematik), die dann klar definierbare Grundprinzipien hätte, in wie weit Ergebnisse einer Anwendung auf eine andere übertragbar sind. Stattdessen sind diese Systeme statistischer Natur, mit messbaren Verteilungen für Erfolg und Misserfolg. Da die speziellen Werte dieser Verteilungen sehr anwendungsabhängig sind und keine existierende Theorie die Vorhersage dieser Werte für neue Anwendungen möglich macht, sollte die sinnvollste Evaluierung die sein, die der speziellen Anwendung am nächsten ist.

Obwohl es zahlreiche Algorithmen für die Gesichtsdetektion gibt, wurden die meisten nicht auf Datenmengen mit einer großen Anzahl an Bildern getestet und die meisten experimentellen Ergebnisse beruhen auf unterschiedlichen Testmengen! Damit man Methoden anständig vergleichen kann, ist es sicherlich sinnvoll eine Art von „Benchmark“-Datenmenge zusammenzustellen, die die Probleme in der realen Welt repräsentieren sollte und die jedem Entwickler als Referenzmenge zur Verfügung stehen sollte. Die am meisten benutzte Datenbank für die Evaluierung im Bereich der Gesichtsdetektion ist momentan die von Rowley an der CMU (neben der von Sung und Poggio am MIT und der von Eastman Kodak als kommerzieller Vertreter).
Doch selbst wenn gleiche Testmengen verwendet werden, besteht immer noch Uneinigkeit unter den Forschern, was eigentlich eine erfolgreiche Detektion bedeutet.
Außerdem gibt es noch eine Vielzahl an Faktoren, die eine Bewertung der Algorithmen (vor allem für die „Appearance-based“ Ansätze) verkompliziert:

- Es kommen häufig unterschiedliche Trainingsmengen und „Tuningparameter“ zum Einsatz. Die Anzahl und die Mannigfaltigkeit der Trainingsbeispiele haben dabei einen direkten Einfluss auf die Klassifikationsleistung.

- Die Trainings- und Ausführungszeit ist bei verschiedenen Methoden auch sehr unterschiedlich. Obwohl die Trainingszeit bei vielen Systemen ignoriert wird, ist sie bei echtzeitfähigen Anwendungen, die ein „online“ Training auf verschiedenen Datenmengen benötigen, natürlich sehr relevant.

- Die Anzahl der Fenster bei einer abscannenden Methode variiert auch sehr stark. Dabei gilt, dass je mehr Fenster ausgewertet werden, desto höher ist die Anzahl der Falschdetektionen.

- In den meisten Systemen sind die Entscheidungskriterien, die für die Detektionsraten verwendet werden, nicht klar definiert. So können Kriterien angewendet werden, die fast alle Teilbilder eines Eingabebildes als erfolgreiche Detektionen für Gesichtsmuster klassifizieren, während bei „strengeren“ Kriterien (beispielsweise müssen bei jeder erfolgreichen Detektion alle sichtbaren Augen und der Mund da sein) diese Teilbilder fast immer als „Falschalarme“ bewertet werden.

- Die Evaluierungskriterien sollten vom Zweck des Detektors abhängen. Soll der Detektor beispielsweise dazu benutzt werden, Leute zu zählen, dann ist die Summe der „False Positives“ und der „False Negatives“ das passende Kriterium. Die Kosten für einen Fehlertyp sollten richtig gewichtet werden, damit man einen optimalen(!) Klassifikator mit Hilfe der Bayes’schen Entscheidungsregel bauen kann. Dies wird auch durch eine Studie ([41]) belegt, die darauf hinweist, dass die Genauigkeit des Klassifikators (also die Detektionsrate bei der Gesichtsdetektion) alleine kein passendes Kriterium für viele Aufgaben in der realen Welt ist. Bei der Klassifikationsgenauigkeit wird nämlich davon ausgegangen, dass die Kosten für die Falschklassifizierung gleich sind, doch dies ist sehr problematisch, da bei vielen Problemen in der realen Welt ein Typ von Klassifizierungsfehler viel „teurer“ sein kann als ein anderer. Außerdem wird bei der Genauigkeitsmaximierung implizit davon ausgegangen, dass die Klassenverteilung für die „Zielumgebung“ bekannt ist. Man nimmt also an, dass die Testdatenmenge die wahre Arbeitsumgebung des Gesichtsdetektors darstellt; doch diese Annahme ist selten gerechtfertigt.

Bei realen Systemen ist es wichtig zu wissen, welche Ressourcen zur Berechnung benötigt werden, insbesondere wie zeit- und speicheraufwendig die benutzten Algorithmen sind. Dabei kann die Geschwindigkeit auf Kosten der Genauigkeit höhere Priorität haben.

Es besteht ein potentielles Risiko, eine allgemeine mittelgroße Standard-Testmenge zu benutzen, da sich Forscher neue Methoden ausdenken oder bestehende Methoden so anpassen, um auf dieser Testmenge dann bessere Ergebnisse zu bekommen. Man lässt sich also mehr oder weniger auf das sogenannte „*Testen auf der Trainingsmenge*“ ein, was natürlich nicht akzeptabel ist. Im Vergleich zu einer hypothetischen Testmenge schneiden die Methoden hier natürlich besser ab, aber trotzdem ist die eigentliche Leistung in der Praxis noch nicht bekannt. Diesen unerwünschten Effekt kann man vermeiden, wenn man eine genügend große und repräsentative allgemeine Testmenge verwendet beziehungsweise wenn die Methoden auf einer kleineren Testmenge evaluiert werden, die zufällig generiert wurde.
Entweder ist es die Aufgabe einer Arbeitsgemeinschaft von Forschern im Bereich der Gesichtsdetektion oder aber eines Dritten, eine angemessene und wirkungsvolle Leistungsbewertung vorzugeben.

2.5 Zusammenfassung und Anwendungen

Die maschinelle Gesichtserkennung (kurz FRT) hat mehrere Anwendungen, angefangen vom statischen Abgleich „kontrollierter“ Fotos wie in Reisepässen, Kreditkarten, Fotoausweisen, Führerscheinen und Verbrecherfotos bis hin zur Auswertung von Videobildern einer Überwachungskamera in Echtzeit, um beispielsweise Drogenhandel oder terroristische Aktivitäten zu unterbinden. Dabei besteht die größte Herausforderung wohl bei willkürlichen Szenen, wie sie zum Beispiel auf einem Flughafen gegeben sind.
FRT ist also ein zentraler Punkt bei Problemen wie elektronisches Abgleichen und Durchstöbern einer „Gesichtsdatenbank“.

Es gibt im Prinzip zwei große Anwendungsgebiete für FRT:

- Kommerzielle Anwendungen:
  Dazu zählen beispielsweise die Benutzeridentifikation für Bankkarten (Kreditkarten). Der Vorteil des Gesichtsidentifikation gegenüber alternativen Methoden (zum Beispiel: Identifikation über Fingerabdruck) liegt zum einen sowohl in der Bequemlichkeit (sie gehört zur Klasse der sogenannten nichtintrusiven Verfahren) als auch im Kostenaufwand und zum anderen kann sie bei unsicheren Fällen ohne großes Training von Menschen korrigiert werden.
- Strafverfolgung:
  Das Grundprinzip beim „Verbrecherfoto“-Problem ist so, dass das System Merkmale des Eingabebildes (welches möglicherweise zuerst gedreht werden muss) mit den Merkmalen in der Datenbank vergleichen muss. Oft werden die Gesichtsbilder mit Hilfe der Pupillen-Distanz (zum Beispiel 30 Pixel) normalisiert.
  Experimente haben gezeigt, dass es innerhalb der „inneren“ Merkmale (Auge, Nase, Mund) eine Bedeutungshierarchie gibt und dass die Kopfkontur eine sehr wichtige Rolle für die Erkennung spielt.

Weitere Anwendungen sind beispielsweise:

- Erkennung von Gesichtsausdrücken (glücklich, traurig, empört,...)
- Videokodierung

Zwischen psychologischen Studien und Literatur in der Ingenieurwelt gab beziehungsweise gibt es kaum einen „Austausch“. Über 30 Jahre Forschung der Psychologie und Neurowissenschaft auf dem Gebiet der menschlichen Erkennung von Gesichtern wird in der Literatur dokumentiert. Obwohl natürlich die maschinelle Erkennung von Gesichtern nicht 100% den Prinzipien der menschlichen Erkennung folgen sollte, ist es für Ingenieure, die Systeme für die Gesichtserkennung entwickeln, doch sehr hilfreich, sich der relevanten Untersuchungen und deren Ergebnisse bewusst zu sein.
Am wichtigsten zu erwähnen ist aber, dass es überhaupt keine Bewertungs- oder „Benchmark“-Studien gibt, die große Datenbanken benutzen, deren Bildqualität repräsentativ für Anwendungen im kommerziellen Bereich und in der Strafverfolgung ist. Stattdessen, wurden viele der bestehenden Ansätze auf relativ kleinen Datenbanken getestet (typischerweise mit weniger als 100 Bildern!).

Sowohl holistische beziehungsweise globale als auch einzelne, unabhängige (also lokale) Merkmale tragen laut psychologischer Forschung zur Gesichtserkennung bei: genau dies wird in einem künstlichen Erkennungssystem beispielsweise durch die Begriffe

„Eigengesichter“ (KL Entwicklung) und „Eigenmerkmale“ (wie „Eigenmund“, „Eigenaugen“) wiedergegeben. Die lokalen Beschreibungen (wie Auge, Nase, Mund) können natürlich auch durch Ansätze wie verformbare „Templates“ gewonnen werden.

Methoden die Neuronale Netze für die Gesichtserkennung benutzen, können sowohl numerische als auch strukturelle Informationen, die für die Erkennung relevant sind, verarbeiten und haben die Fähigkeit bei unvollständiger Information zu generalisieren und damit die Erkennungsaufgabe zu lösen.
Methoden, die auf verformbaren „Templates“ beruhen, beispielsweise für die Augen- oder Mundextraktion haben zwei große Nachteile: zum einen muss die initiale Plazierung des „Templates“ wirklich gut sein und zum anderen ist die Anpassung der vielen Parameter sehr rechenaufwendig.
„Datengetriebene“ Methoden (basierend auf Lernalgorithmen) sind sehr von der Trainingsmenge abhängig; deshalb sollte man eine geeignete Datenbank für diese Methoden finden.

Die Einteilung der Methoden zur Gesichtsdetektion in die oben beschriebenen Kategorien ist natürlich nur ein Vorschlag. Manche Methoden können auch in mehr als eine der vier Hauptkategorien klassifiziert werden. Beispielsweise benutzen „Template-Matching“ Methoden normalerweise einerseits ein Gesichtsmodell und „Subtemplates“, um Gesichtsmerkmale zu extrahieren und anderseits werden dann diese Merkmale für die Lokalisierung und Detektion von Gesichtern verwendet.
Außerdem ist die Grenze zwischen wissensbasierten Ansätzen und einigen „Template-Matching“ Methoden sehr verschwommen, da für die Definition der Gesichtsschablonen implizit menschliches Wissen über den Aufbau eines Gesichtes verwendet wird.
Alternativ könnte man die Ansätze für die Gesichtsdetektion auch nach folgendem Aspekt kategorisieren: beziehen sich die Ansätze auf lokale Merkmale oder behandeln sie ein Gesichtsmuster als ganzes, also liegt eine holistische Vorgehensweise vor?

Der wichtigste Teil in der Gesichtserkennung ist die Fähigkeit, bestehende Methoden zu evaluieren und neue Richtungen basierend auf diesen Bewertungen zu definieren. Dabei sollten die in der Evaluierung verwendeten Bilder so gewählt werden, dass sie den Bildern, die später dem Erkennungssystem vorgeführt werden, recht ähnlich sind.

Ein System zur Detektion von Gesichtern sollte auch bei voller Variation der folgenden Punkte robust funktionieren:
- Beleuchtungsbedingungen
- Orientierung, Haltung und (teilweise) Verdeckung
- Gesichtsausdrücke
- Existenz von Brillen, Haare im Gesicht und verschiedene Haarstile

Man hat im Prinzip die volle Breite an Herausforderungen wie auch bei der allgemeinen Objektklassenerkennung (als eines der größten Aufgabengebiete des Maschinensehens). Nichtsdestotrotz hat man bei dieser speziellen Objektgruppe den Vorteil der sehr offensichtlichen Regularitäten, die von vielen heuristischen oder modellbasierten Methoden ausgenutzt werden oder die mittels „datengetriebener“ Methoden einfach gelernt werden. Obwohl Gesichter eine riesige Variabilität innerhalb der Klasse aufweisen, handelt es sich bei der Gesichtsdetektion nach wie vor um ein klassisches zwei-Klassen-Erkennungsproblem (Gesicht versus „Nichtgesicht“).

Kapitel 3

Systembeschreibung

Im folgenden wird beschrieben wie das bereits bestehende Tracking-System am DaimlerChrysler Forschungzentrum in Ulm aufgebaut ist; also in welche Umgebung beziehungsweise in welches „Tracking-Konzept“ die in dieser Diplomarbeit entwickelten Algorithmen zur Leistungsverbesserung eingebettet werden. Außerdem wird in diesem Kapitel auf die Herausforderungen beim Tracking im PKW näher eingegangen und zum Abschluss dieser Systembeschreibung wird die Zielsetzung der Diplomarbeit genau aufgezeigt.

3.1 Aufbau des Kopf-Tracking Systems

Die Aufgabe des Kopf-Tracking Systems ist es, den Kopf einer Person im Fahrzeug personenunabhängig zu verfolgen. Dabei sind die 6 Freiheitsgrade des Kopfes (3 Raum- und 3 Winkelkoordinaten) die Zustandsgrößen, die man gerne wissen will.

3.1.1 Hardware

Das Kopf-Tracking System besteht aus:

- Einem Versuchsfahrzeug: hier eine E-Klasse als T-Modell (Abb. 3.1, links).
- Einer digitalen Mono-Kamera (Abb. 3.1, Mitte). Dies ist eine Industriekamera mit (PAL) 640x480 Pixel Auflösung (Vollbild), 8 oder 12 Bit Grauwertauflösung (progressive-scan, non-interlaced) und bis zu 40 Hz Bildwiederholfrequenz.
  Das Objektiv hat 8 mm Brennweite. Die Brennweite ist so gewählt, dass die auf dem Armaturenbrett montierte Kamera einerseits den Kopf des Fahrers und seine Umgebung gut erfasst, andererseits noch genügend Auflösung für die Gesichtsmerkmale bietet.
- Einem PCI-Framegrabber, der die Verbindung zur Kamera herstellt und sowohl die Bilder aus der Kamera ausliest als auch die Kamera ansteuert (Belichtungszeit, Verstärkung, ...)
- Einem Fahrzeugrechner (Abb. 3.1, rechts), hier ein Linux-PC, auf dem die Verarbeitungssoftware läuft. Aktuell benötigt das Kopf-Tracking im Tracking-Mode zirka 1 ms Rechenzeit pro Frame auf einem Pentium III mit 1 GHz.

Abbildung 3.1: (links) Fahrzeug, (Mitte) montierte Kamera, (rechts) Fahrzeugrechner

3.1.2 Algorithmik und Kalmanfilter-Framework

Das Verfahren zum Verfolgen des Kopfes ist ein featurebasiertes Tracking-Verfahren. Die Messung der Gesichtsfeature-Positionen im Bild werden mit Hilfe eines Kalmanfilters (Abb. 3.2) über die Bildsequenz integriert. Es beruht auf folgenden Grundlagen:

- Da der Kopf in allen sechs Freiheitsgraden verfolgt werden soll, liegt dem Tracking ein Kopfmodell zu Grunde, das die Positionen der Gesichtsmerkmale in 3-D modelliert.
- Da die Prädiktion eines Kalmanfilters um so besser ist, je kleiner die Zeitabstände zwischen den Messungen sind, sollte die Detektion auf dem Einzelbild möglichst schnell gehen. In unserem Fall (40 Hz Bildwiederholfrequenz) heißt das, die Verarbeitung sollte pro Bild weniger Zeit benötigen als 1/40 Sekunden, also 25 ms.

Die beiden Hauptkomponenten des Kalmanfilters sind das Bewegungsmodell und die Messgleichung.

3.1.2.1 Bewegungsmodell

Das Bewegungsmodell beschreibt die Dynamik des Modells in der Zeit. In unserem Fall wird der Kopf durch ein einfaches Modell beschrieben, das aus sechs Punkten besteht, die sich in einem mit dem Kopf fest verbundenen Koordinatensystem befinden. Der Ursprung dieses Koordinatensystems liegt im Schwerpunkt des Kopfes und die y-Achse zeigt nach vorne (in Richtung Nase). Die sechs Punkte repräsentieren die Gesichtsfeatures (Augen, Nasenlöcher und Mundwinkel) und lassen sich in gewissen Grenzen innerhalb des kopffesten Koordinatensystems verschieben, um verschiedene Gesichtsgeometrien (zum Beispiel langsame Bewegung des Mundes) repräsentieren zu können. Das Kopfmodell wiederum ist in den dreidimensionalen Raum eingebettet, dessen Ursprung fest mit dem Fahrzeug verbunden ist.

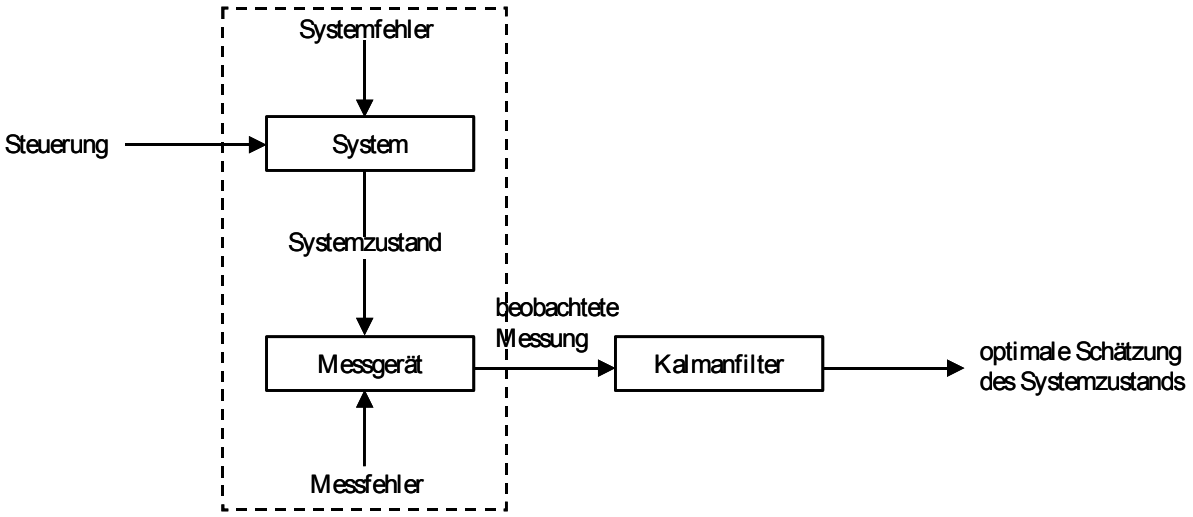


Abbildung 3.2: Funktion des Kalmanfilters

Es wird nun angenommen, dass sich der Kopf in erster Näherung gleichförmig, also mit konstanter Geschwindigkeit und Winkelgeschwindigkeit bewegt. Das Bewegungsmodell nimmt also den Kopf als trägen kräftefreien Körper an, der sich mit konstanten Geschwindigkeiten bewegt. Die freien Parameter der sechs Featurepunkte im Kopfkoordinatensystem werden als konstant modelliert, das heißt man nimmt an, dass sie sich bis auf eine Drift während des Trackings nicht ändern. Es werden also sowohl die sechs Freiheitsgrade des Kopfes im Raum, ihre Geschwindigkeiten als auch die inneren freien Parameter des Kopfmodells getrackt. Diese zu bestimmenden Parameter entsprechen dem Zustandsvektor des Kalmanfilters.

3.1.2.2 Messgleichung

Die Messgleichung beschreibt, wie die sechs Featurepunkte des Modells im 3-D Koordinatensystem in das Kamerabild projiziert werden. Ist also die Kamerageometrie bekannt, lassen sich über die Messgleichung die sechs Positionen im Bild bestimmen, an denen die Gesichtsfeatures bei gegebenen 3-D Koordinaten des Kopfmodells sind. In unserem Fall wird ein Lochkameramodell angenommen.

3.1.2.3 Tracking-Zyklus

Nehmen wir nun an, der Tracker würde bereits laufen und zum Bild n wäre die Position n des Kopfes bekannt. Dann läuft der Tracking-Zyklus folgendermaßen ab:

1. Prädiziere mit der Modellgleichung die 3-D Position n+1 für den nächsten Zeitschritt.
2. Berechne mit der Messgleichung die Positionen der Features im Bild n+1 für den nächsten Zeitschritt.
3. Suche im Bild n+1 in der Nähe der prädizierten Featurepositionen nach den tatsächlichen Positionen.
4. Korrigiere die prädizierten 3-D Positionen n+1 anhand der gemessenen Positionen.
5. Ist der Track noch „gut genug“ (ist der Kopf zum Beispiel noch innerhalb des Fahrzeuginnenraums), geht es weiter mit 1., sonst wird der Track abgebrochen.

3.1.2.4 Initialisierung mit dem „Matched Filter“

Nun muss der Tracking-Zyklus irgendwann einmal anfangen oder bei Abbruch wieder neu gestartet werden, das heißt in beiden Fällen dass das Tracking initialisiert wird. Dieser Initialisierungsschritt läuft folgendermaßen ab:

1. Im Bild wird nach möglichen Kandidaten für Gesichtsmerkmale (Augen, Nase, Mundwinkel) gesucht. Dabei kommt eine Art von „*Matched Filter*“ zum Einsatz, der die Punkte herausfiltert, die relativ zu ihrer lokalen Umgebung statistisch signifikant dunkler sind (wie zum Beispiel Pupille, Nasenloch).
2. Es wird ein 3-D Position des Kopfmodells gesucht, die möglichst gut zu den gefundenen Gesichtsmerkmal-Kandidaten passt.
3. Passt die 3-D Position gut genug (das heißt ist der mittlere quadratische Abstand der Featurepositionen zu den Kandidaten klein genug), wird der Zyklus mit der gefundenen 3-D Position gestartet. Sonst wird das nächste Bild geholt und es geht weiter mit 1.

Die Detektion der Gesichtsmerkmale ist momentan sehr einfach gehalten. Der Hintergrund ist, dass man sich auf dem Einzelbild Fehler erlauben darf, wenn die Bildfrequenz hoch genug ist und die Messungen mit einem statistischen Filter integriert werden. Die Features werden immer in einer kleinen Umgebung um die prädizierten Featurepositionen gesucht. Die Größe der Umgebung ist von den Varianzen der 3-D Koordinanten (die ja der Kalmanfilter kennt) und vom Kopfabstand zur Kamera abhängig.

Der (bisherige) „Matched Filter"-Ansatz funktioniert nun so:
Der Punkt P=(x,y) im Bild soll untersucht werden. R sei ein Rechteck der Breite b und Höhe h mittig um den Punkt P. Px sind alle Pixel, die auf R liegen (nicht im Rechteck, nur auf dem Rand).
Sei g(p) der Grauwert von Pixeln, dann ist m=Mittelwert(g(Px)) und v=Varianz(g(Px)).

Wenn die Bedingung g(P)<m-a*v erfüllt ist, dann wird das korrespondierende Pixel im Detektionsbild auf schwarz gesetzt ansonsten auf weiß. Der Punkt P muss also signifikant dunkler als seine Umgebung sein.
Zum Schluss werden noch alle einzelnen schwarzen Pixel im Detektionsbild gelöscht. Der Parameter „a" ist ein Skalierungsfaktor, mit dem man einstellen kann, wie viel ein Punkt dunkler als seine Umgebung sein muss, damit er im Detektionsbild schwarz gesetzt wird.

Die Detektoren für die einzelnen Gesichtsmerkmale sind folgendermaßen realisiert:

- Pupille, Nasenlöcher: Hier wird wie eben beschrieben nach dem dunkelsten Punkt in der Umgebung gesucht. Dieser Ansatz ist sehr einfach, er findet aber auch bei spiegelnden Brillen oft noch die Pupille.
- Mundwinkel: Hier wird nach einem horizontalen Kantenstück gesucht.

Der Vorteil dieser Methode zur Featuredetektion ist die einfache und simple Implementierung. Aber es hat sich gezeigt, dass die Vorgehensweise mit Hilfe des „Machted Filter"-Ansatzes einfach nicht gut genug ist und außerdem wird überhaupt keine Information über die einzelnen Gesichtmerkmal-Klassen geliefert.

In Abbildung 3.3 sieht man das am DaimlerChrysler Forschungszentrum implementierte Kopf-Tracking System. Neben den reinen Zustandsgrößen, also den 6 Freiheitsgraden (x,y,z,phi,theta,psi) des Kopfes werden auch noch deren Standardabweichungen beziehungsweise allgemeine Veränderungen im Laufe des „Trackings" ausgegeben und grafisch dargestellt.

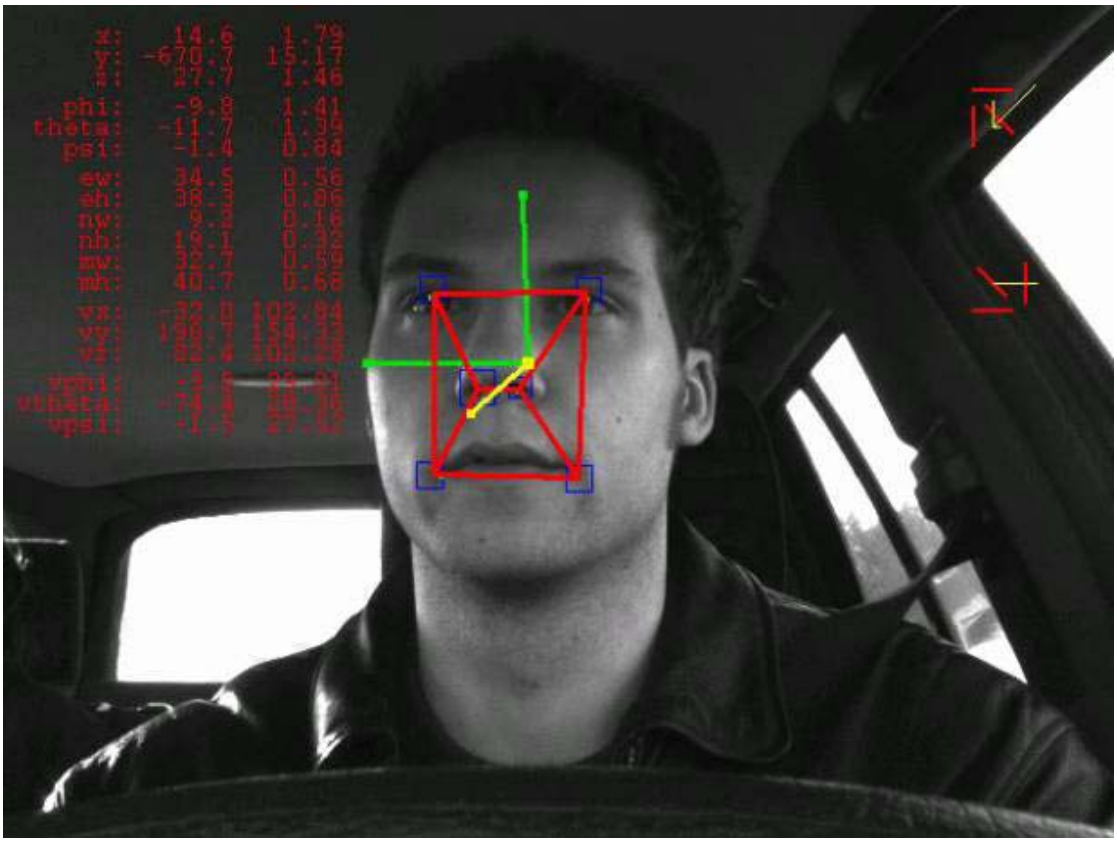


Abbildung 3.3: Kopf-Tracking System mit zusätzlichen Visualisierungen der Zustandsgrößen

3.2 Probleme beim Tracking im PKW

Im folgenden wird anhand von Beispielbildern erläutert, worin die Probleme beim Tracking im PKW bestehen und wieso ein sich anbietendes „Template-Matching“ für die Detektion von Gesichtsmerkmalen in diesem Umfeld eher ungeeignet ist.

3.2.1 Allgemeine Herausforderungen

Zu den Hauptproblemen beim Tracking im PKW zählen die folgenden Aspekte:

- Es handelt sich um einen nicht kooperativen Fahrer: der Fahrer macht was er will (Abb. 3.4, links oben), das heißt das Gesicht ist nicht zwangsläufig frontal auf die Kamera gerichtet, und er erwartet, dass das System sofort automatisch fehlerfrei funktioniert. Die Algorithmen sollten also bei den verschiedensten Personen ohne vorheriges individuelles Training die gewünschte Leistung erbringen.

- Bärte, Brillen, Haarstil, Schmuck, helle/dunkle Haut und allgemein Hintergrund-Rauschen erschweren die Detektion der Gesichtsmerkmale.
  (Abb. 3.4, rechts oben, links Mitte, rechts unten)

- Bei der Bildgröße handelt es sich wie bereits erwähnt um eine Auflösung von 640x480 (non-interlaced). Um den relevanten Teil des Innenraumes zu erfassen, wird ein Objektiv von ungefähr 8 mm Brennweite verwendet, weshalb die Auflösung des Gesichtes nicht besonders gut ist. Außerdem hat man nur Graustufenbilder zur Verfügung, also fallen Ansätze, die sich auf die (RGB-)Farbverteilung im Eingabebild stützen, komplett weg.

- Reflexionen, (Schlag-)Schatten und eine sehr hohe/schnelle Dynamik im Fahrzeug erschweren das Tracking enorm im Vergleich zu anderen (definierteren) Umgebungen. (Abb. 3.4, rechts Mitte, links unten)
  Die Umgebung ist also auch nicht kooperativ, das heißt zum einen ist die Kamera fest im Fahrzeug montiert und zum anderen gibt es keine Möglichkeit, die Beleuchtung zu beeinflussen: Lichtwechsel von etlichen zehn Dezibel innerhalb weniger Bilder inklusive Schlagschatten sind keine Seltenheit.

- Eine weitere große Herausforderung ist die benötigte Rechenzeit: irgendwann soll das System nicht auf einem 1 GHz-Prozessor sondern auf einem Mikrokontroller laufen! Die angestrebte Video-Taktrate beim Tracking ist momentan 80 Hz, bei der Initialisierung kann man sich aber mehr Zeit lassen.

Abbildung 3.4: Darstellung beispielhafter Herausforderungen beim Tracking im PKW

3.2.2 Probleme beim „Template-Matching“ mittels Kantenbilder

Um die Probleme beim „Template-Matching“ zu veranschaulichen, dienen die folgenden Bildanreihungen (siehe Abb. 3.5). Untersucht wurden ein Bewerbungsbild (mit definierter Umgebung) und drei weitere Bilder aus einer realen Sequenz beim Tracking im Fahrzeug: dabei handelt es sich um ein Nachtsichtbild, ein Tagbild mit Reflexionen und schließlich noch ein Tagbild mit Schatteneffekten. Zum Einsatz kamen ein Canny-, ein Laplace of Gaussian(LoG)- (auch Marr-Hildreth- genannt) und ein Sobel-Kantenoperator (für eine detaillierte Erklärung dieser Operatoren beziehungsweise Filter sei auf [100] verwiesen).

Beim Bewerbungsbild lieferten alle drei Operatoren brauchbare Ergebnisse, wobei der Sobel-Operator für die Detektion der Augen, der Nase und des Mundes am besten geeignet war, da am wenigsten Rauscheffekte aufgetreten sind.
Beim Nachtsichtbild sieht es ähnlich aus, wobei der Canny-Operator viel zu viel Rauschen liefert, um beim LoG- und beim Sobel-Operator die störenden runden Lichtsignale fälschlicherweise als Augen detektiert werden könnten.

Beim Reflexionsbild funktioniert der Sobel-Operator jedoch gar nicht gut: er findet keine Augen, keine Nase und kein Mund, sondern nur noch den Körperumriss. Beim LoG werden die Augen und die Nasenlöcher schwach erkannt und beim Canny ist die Detektion der Gesichtsmerkmale trotz vieler störender Kanten im Fahrzeug am besten.
Beim Schattenbild versagen im Prinzip alle drei Operatoren: Man kann kein „Template“ definieren, dass dann in den Kantenbildern die entsprechenden Gesichtmerkmale finden würde. Es entstehen abstruse, willkürliche Formen, je nach Schatteneinwirkung.

Abbildung 3.5: Anwendung verschiedener Kantenoperatoren auf unterschiedliche Eingabebilder

3.3 Zielsetzung dieser Diplomarbeit

In dieser Diplomarbeit soll die Initialisierung des bestehenden Kopf-Tracking Systems mit Hilfe von adaptiven Verfahren verbessert werden. Es gibt zwar schon wie in Abschnitt 3.1.2.4 beschrieben eine Initialisierung mit dem „Matched Filter“, aber diese ist einfach nicht gut genug.

Man wünscht sich robuste Algorithmen, die auch bei den vorher beschriebenen Randbedingungen beziehungsweise Problemen (speziell im Fahrzeug) funktionieren.
Dabei sollen die implementieren Verfahren mittels eines Trainings angepasst werden, aber dann selbständig auf unbekannten Testbildern sofort die gewünschte Erkennungsleistung bei vertretbarem Rechenaufwand erbringen.

Natürlich will man nicht nur eine Einteilung in Vorder- und Hintergrund-Klasse wie es der „Machted Filter“ momentan liefert, sondern gewünscht ist eine genaue Zuweisung der Bildregionen zu den folgenden sieben Klassen:

- *rechtes Auge*
  (in der Labeldatei ist die Position des zugehörigen Zentralpixels abgekürzt mit „erx“ = eye right, x-position und „ery“ = eye right, y-position)
- *linkes Auge*
  (in der Labeldatei ist die Position des zugehörigen Zentralpixels abgekürzt mit „elx“ = eye left, x-position und „ely“ = eye left, y-position)
- *rechtes Nasenloch*
  (in der Labeldatei ist die Position des zugehörigen Zentralpixels abgekürzt mit „nrx“ = nose right, x-position und „nry“ = nose right, y-position)
- *linkes Nasenloch*
  (in der Labeldatei ist die Position des zugehörigen Zentralpixels abgekürzt mit „nlx“ = nose left, x-position und „nly“ = nose left, y-position)
- *rechter Mundwinkel*
  (in der Labeldatei ist die Position des zugehörigen Zentralpixels abgekürzt mit „mrx“ = mouth right, x-position und „mry“ = mouth right, y-position)
- *linker Mundwinkel*
  (in der Labeldatei ist die Position des zugehörigen Zentralpixels abgekürzt mit „mlx“ = mouth left, x-position und „mly“ = mouth left, y-position)
- und *Hintergrund* (hier auch *„Garbage“* genannt)
  (beim „Labeln“ werden alle Pixel im Bild, die nicht in die definierte Region um die Zentralpixel der zuvor erwähnten Klassen fallen, automatisch auf Hintergrund gesetzt)

Die ersten sechs Klassen sind genau die Gesichtsmerkmale, wie sie in dieser Diplomarbeit verwendet werden. Natürlich muss man solche Merkmale von „berechneten“ Bildmerkmalen unterscheiden!
Durch diese Klassenzuweisung hat der Kalman-Filter einen qualitativ höherwertigen „Input“, wodurch die Gesamterkennungsleistung der Gesichtsregion und der Gesichtsmerkmale implizit besser wird.

Kapitel 4

Untersuchte Ansätze zur Erkennung von Gesichtsmerkmalen

Wie man im „Stand der Technik“-Kapitel gesehen hat, gibt es eine Vielzahl von Möglichkeiten, Gesichter beziehungsweise Gesichtsmerkmale in einem Frame einer Videosequenz zu erkennen. Deshalb muss man sich aus Zeitgründen auf einige wenige Ansätze konzentrieren, von denen man sich gute Erkennungsleistungen verspricht und die im Fahrzeugumfeld auch robust funktionieren. Im folgenden wird nun beschrieben, welche Ansätze genauer untersucht beziehungsweise implementiert wurden und wie sich das hier verwendete Erkennungssystem allgemein zusammensetzt.

4.1 Übersicht des prinzipiellen Aufbaus – Komponenten des Erkenners

Der Erkenner für die Gesichtsmerkmale (rechtes/linkes Auge, rechtes/linkes Nasenloch, rechter/linker Mundwinkel) setzt sich quasi aus vier Komponenten zusammen (Abb. 4.1):

- *Rohdatengenerierung*: hier wird ein Fenster (einer bestimmten Größe) verwendet, das über den gesamten Eingabeframe gleitet
- *Vorverarbeitung* beziehungsweise *Normalisierung* der Fensterinhalte oder keines von beidem
- *Merkmalsextraktion* aus den (vorverarbeiteten/normalisierten) Fenstern und möglicherweise Dimensionsreduktion
- *Klassenzuweisung* der Merkmalsvektoren über einen Klassifikator, der zuvor anhand von „Klassenlabels“ in anderen Eingabebildern trainiert worden sein muss (siehe Anhang B)

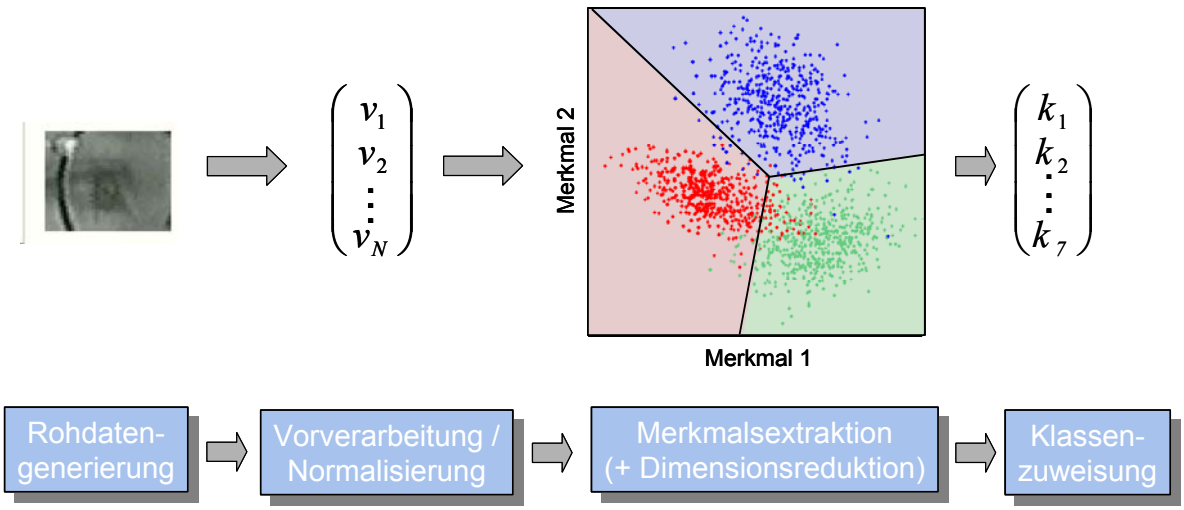


Abbildung 4.1: Komponenten des Erkennungssystems für die sechs Gesichtsmerkmale (die siebte Klasse ist Hintergrund beziehungsweise „Garbage“)

4.2 Fenstertechnik – Abscannen der Eingabeframes

In Anlehnung an das menschliche Sehen, das sich neben der holistischen Verarbeitung auch auf lokale Regionen durch ein „Abscannen“ eines Bildes konzentriert (beispielsweise beim Lesen einer Zeitschrift), wird zur Rohdatengenerierung eine Fenstertechnik eingesetzt.
Dabei wird ein Fenster von links nach rechts (x-Richtung) und von oben nach unten (y-Richtung) über das Eingabeframe geschoben (Abb. 4.2). Das Fenster kann dabei rechteckig oder quadratisch sein. Neben dieser Breiten- und Höheneinstellung kann man auch festlegen, ob das Fenster unterabgetastet wird, um das Verfahren zu beschleunigen. Dabei kann man in x- und y-Richtung jeweils die Sprungweite innerhalb eines Fensters angeben. Idealerweise sollten dabei die Bedingungen des Abtasttheorems (siehe [100]) erfüllt bleiben. Außerdem ist über eine Schrittweite in x- und y-Richtung die Überlappung beziehungsweise der Versatz der Fenster untereinander einstellbar, also wie viele Pixel weiter das nächste Fenster die Daten „abscannt“.
Diese Fensterparameter (Größe, Unterabtastung und Schrittweite/Versatz) werden einmal festgelegt und ändern sich dann nicht mehr für eine untersuchte Methode.
Damit man keine Daten „verliert“ beziehungsweise um irgendwelche Sprungeffekte zu vermeiden, sollte der Versatz am besten ein Pixel groß sein und es sollte keine Unterabtastung stattfinden; einzig die Größe der Fenster kann dann variabel sein.

Bei dieser Technik hat man im Vergleich zu anderen Methoden (wie dem „Matched Filter“) Vor- und Nachteile.
Zu den Vorteilen zählen vor allem:

- Man verarbeitet im Idealfall das ganze Eingabebild, wodurch keine Regionen „vergessen“ werden.
- Man hat durch die Fensterform sofort eine Matrixgestalt und kann für die Weiterverarbeitung der Rohdaten übliche Matrixoperationen darauf anwenden.
- Die Technik ist relativ leicht implementierbar (zumindest in MATLAB).

Nachteile sind unter anderem:

- Wie evaluiert man die ganzen Fensterregionen? Was macht man mit einem „halben Auge“ beziehungsweise einem „Drittel Mundwinkel“?
- Alle Pixel im Eingabebild werden unter Umständen „angefasst“, was sehr rechenintensiv sein kann.

Abbildung 4.2: Fenstertechnik für die Rohdatengenerierung

Für die Weiterverarbeitung und Klassifikation werden die Rohdaten im gesamten Bild durch die Blockoperation „aufgesammelt“ und dann anschließend gemeinsam klassifiziert. Es wird also ein einzelnes Fenster nicht sofort einer Klasse zugewiesen. Dies liegt daran, dass die Implementierung in MATLAB erfolgte. MATLAB ist matrizenorientiert und man sollte wenn möglich keine unnötigen Schleifen/Rekursionen verwenden.

Wie bei den Evaluierungsprinzipien (2.4) schon erwähnt, sollte es klar sein, dass je mehr Fenster ausgewertet werden, desto höher ist die Anzahl der Falschklassifikationen.
Außerdem sollte man folgendes beachten: Je größer der Kontext eines Fensters ist, umso mehr Information über die Bildregion steht zur Verfügung, aber gleichzeitig man hat auch mehr Stör-/Rauschsignale und der Rechenaufwand für die Vorverarbeitung, Merkmalsextraktion und Klassifizierung steigt!

Für das Trainieren des Erkennungssystems beziehungsweise genauer gesagt des Klassifikators braucht man natürlich beispielhafte Samples, die als Repräsentanten der sieben Klassen fungieren. In Abbildung 4.3 sieht man exemplarische Gesichtsmerkmale und in Abbildung 4.4 werden Hintergrund- beziehungsweise „Garbage“-Fenster dargestellt. Diese Fensterbeispiele sind 30x30 Pixel groß und entstammen dem Frame in Abbildung 4.2.

Abbildung 4.3: Gesichtsmerkmale

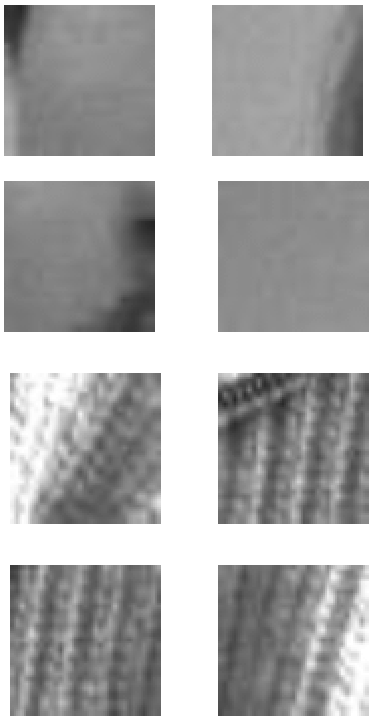


Abbildung 4.4: „Garbage“

4.3 Mögliche Vorverarbeitungsschritte/ Normalisierungen

Um störende Effekte wie Spiegelungen, Reflexionen, Schattenbildung, Lichtsignale oder ähnliches bedingt durch die Fahrzeugumgebung (vor allem Glasfenster) und die Sonneneinstrahlung (oder allgemein Witterungsverhältnisse) auszugleichen, sollte man auf die Rohdaten eine Vorverarbeitung (wie die Mittelung) beziehungsweise eine Normalisierung anwenden.
Zur Veranschaulichung dieser Störsignale im Eingabeframe sei auf Abbildung 3.4 des Kapitel 3 verwiesen.

Natürlich kann es auch Rauschen aufgrund von Fehlern im Bildaufnahmeprozess geben (beispielsweise durch die verwendete Kameratechnik wie CCD, die Signalübertragung oder durch Überbelichtung, vor allem bei Nachtsichtaufnahmen). Dabei entsprechen die Pixelwerte nicht den wahren Intensitäten der realen Szene. Im Ortsraum kommen dann lineare Filter zum Einsatz und im Frequenzraum werden dafür sogenannte zweidimensionale „finite impulse response"(FIR)-Filter durch eine Faltung implementiert. Normalerweise sollte das Kamerasystem diese Unregelmäßigkeiten schon a priori durch entsprechende Operationen „abfangen".

Zur Rauschunterdrückung und Mittelung verwendet man oft die folgenden drei Filterarten:
- Lineare Filter
- Medianfilter
- Adaptive Filter

Für eine genaue Erläuterung der Mittelung und der Filter sei auf [100] verwiesen.
Die Wirkungsweise von Glättungsfiltern sieht man in Abbildung 4.5. Dabei wurde zuerst das Ausgangsbild mit binärem Rauschen verfälscht (hier: „Salt&Pepper"-Rauschen, das heißt zufällige Pixel werden auf schwarz oder weiß gesetzt).
Bei einem linearen Filter (wie beispielsweise das Binomialfilter) wird Gaußsches Rauschen zwar wirksam unterdrückt, jedoch binäres Rauschen nur sehr schlecht. Wie man sieht, werden Kanten dann verwischt und das Bild wirkt unscharf (Abb. 4.5, Mitte). Schlimmer noch: wenn die Maske eines Glättungsoperators über ein Objektkante läuft, enthält sie Bildpunkte von Objekt und Hintergrund. Das Filter liefert an dieser Stelle ein unsinniges Ergebnis. Das gleiche gilt, wenn eine bestimmte Anzahl von Bildpunkten zum Beispiel aufgrund von Übertragungsfehlern fehlerhafte Werte aufweist.

Abbildung 4.5: (links) Ausgangsbild mit zusätzlichem „Salt&Pepper"-Rauschen, (Mitte) Ergebnis des linearen Glättungsfilters, (rechts) Ergebnis des Medianfilters

Anders sieht es da bei einem Medianfilter aus, ein sogenanntes Rangordnungsfilter das zur Klasse der nichtlinearen Filter gehört. Rangordnungsfilter detektieren und eliminieren die Störungen durch fehlerhafte Bildpunkte. Die Grauwerte werden dabei innerhalb der Maske ihrer Größe nach sortiert, und ein Pixel wird selektiert. Das Medianfilter selektiert den mittleren Wert. Da binäres Rauschen den Grauwert völlig ändert, ist es sehr unwahrscheinlich, dass der fehlerhafte Grauwert gleich dem mittleren Grauwert in der direkten Umgebung ist. Auf diese Weise wird der mittlere Grauwert der Umgebung verwendet, um den Grauwert des gestörten Bildpunktes wiederherzustellen. Wie man in Abbildung 4.5 (rechts) sieht, werden dadurch Ausreißer eliminiert und das Bild bleibt scharf.

Ein adaptives Filter ist beispielsweise ein sogenanntes Wiener Filter (eine Art von linearem Filter), das adaptiv auf ein Bild angewendet wird, das heißt es passt sich selbständig an die lokale Bildvarianz an. Wenn die Varianz groß ist, leistet das Wiener Filter wenig Glättung, doch da wo die Varianz klein ist, glättet das Filter mehr.
Dieser Ansatz funktioniert oft besser als ein lineares Filter. Das adaptive Filter ist selektiver als ein vergleichbares lineares Filter, wodurch Kanten und andere hochfrequente Bildteile erhalten bleiben. Jedoch ist es im Vergleich zu einem linearen Filter rechenaufwendiger.
Die beste Leistung zeigt ein solches Wiener Filter bei Anwesenheit von sogenanntem „weißen Rauschen“ wie das Gaußsche Rauschen. In Abbildung 4.6 sieht man genau diese Wirkungsweise: das Ursprungsbild des Saturns wurde dabei mit Gaußschem Rauschen überlagert und der adaptive Filter liefert ein sehr gutes Ergebnis.

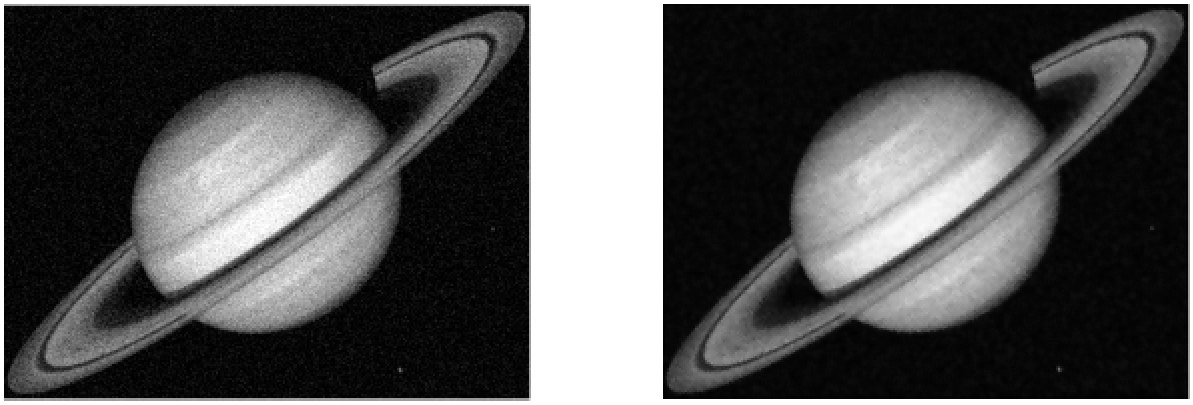


Abbildung 4.6: (links) verrauschte Version eines Saturnbildes, (rechts) Ergebnis des adaptiven Filters

Bei den Bildverbesserungstechniken sollte man unterscheiden zwischen einer objektiven Sichtweise (zum Beispiel der Verbesserung des Signal-Rausch-Verhältnisses) und einer subjektiven Sichtweise (zum Beispiel werden durch die Veränderung von Farben oder Intensitäten bestimmte Merkmale prägnanter).

Zu den gängigsten Normalisierungen beziehungsweise Intensitätsanpassungen im Bereich der digitalen Bildverarbeitung zählen:
- Mittelwertfreie Daten
- Varianznormalisierte Daten
- Histogrammnormalisierte Daten
- Gammawertangepasste Daten

Die Intensitätsanpassung ist eine Technik, um die Graustufenwerte eines Bildes auf einen neuen Bereich zu „mappen".
Man verwendet dazu normalerweise ein Histogramm: das ist in der Bildanalyse ein Diagramm für die Darstellung der Intensitätsverteilung in einem Bild. Es werden dabei n gleich verteilte sogenannte „bins" auf der x-Achse aufgetragen, die jeweils einen Graustufenbereich repräsentieren und auf der y-Achse wird die dazugehörige Anzahl der Pixel vermerkt, deren Graustufenwerte genau in diesen Bereich fallen. Die Information in einem Histogramm kann für eine geeignete Verbesserungsoperation verwendet werden. Zum Beispiel können in einem Bildhistogramm, das eine kleine Spanne an Intensitäten aufweist durch eine Anpassung die Werte auseinander gezogen werden, wodurch der Kontrast erhöht wird. Natürlich kann man auch eine Kontrasterniedrigung erzwingen.
Abbildung 4.7 zeigt ein Bild mit Reiskörnern und das dazugehörige Histogramm mit 64 „bins", das aufgrund des dunkelgrauen Hintergrundes im Bild einen „Peak" bei zirka 100 aufweist.
In Abbildung 4.8 wurde eine Intensitätsanpassung des Reiskörner-Bildes vorgenommen, indem das Histogramm nun den gesamten Bereich der 64 „bins" ausfüllt. Dadurch wurde der Kontrast erhöht.

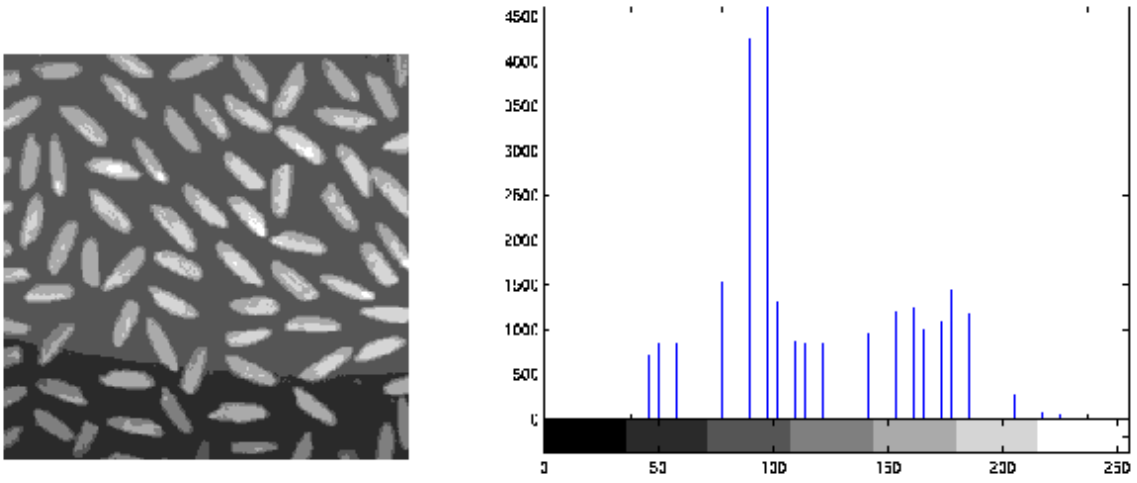


Abbildung 4.7: (links) Graustufenbild, (rechts) dazugehöriges Histogramm

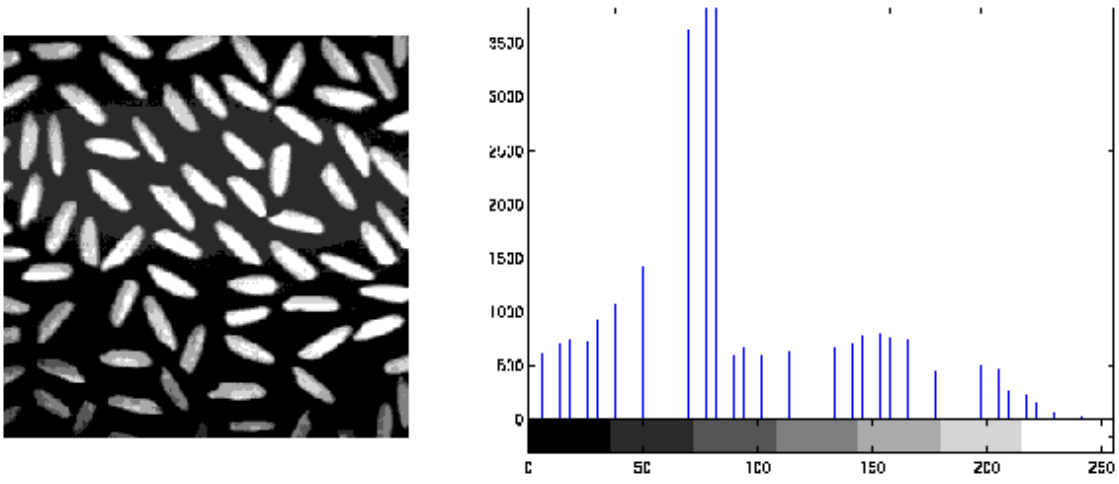


Abbildung 4.8: (links) kontrastverstärktes Graustufenbild, (rechts) dazugehöriges Histogramm

In Abbildung 4.9 wird die (automatische) Histogrammnormalisierung verdeutlicht, wodurch das Histogramm des Ausgabebildes einem bestimmten (vorgegebenen) häufig flachen Histogramm angenähert wird. Das Originalbild zeigt hier einen schwachen Kontrast, wobei die meisten Werte in der Mitte des Intensitätsbereichs liegen. Durch die Histogrammnormalisierung wird ein kontrastverstärktes Ausgabebild erzeugt, dessen Grauwerte gleichmäßig entlang des Intensitätsspektrums von 0 bis 255 verteilt sind.

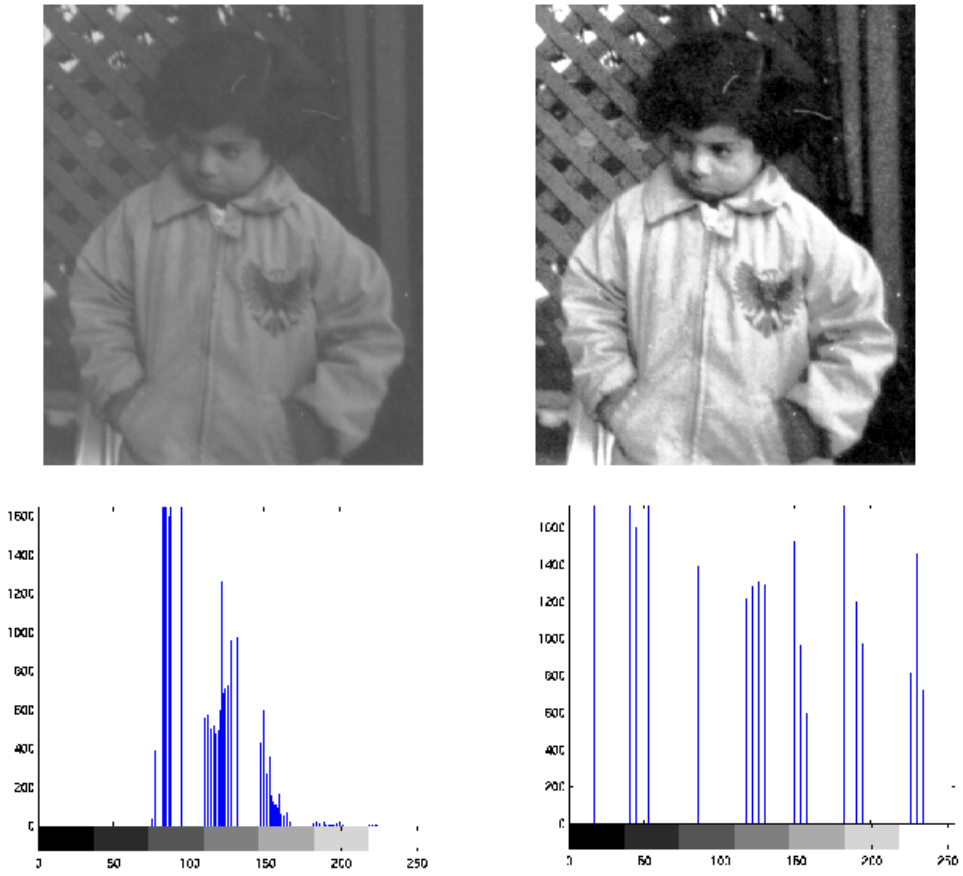


Abbildung 4.9: (links) Ausgangsbild und korrespondierendes Histogramm, (rechts) histogrammnormalisiertes Ergebnisbild

Normalerweise werden die Werte zwischen dem kleinsten und größten Grauwert des Ausgangsbildes linear auf die entsprechenden Werte des Ergebnisbildes „gemappt“.
Durch einen zusätzlichen Faktor, Gammawert genannt, kann dieses „Mapping” auch nichtlinear sein, das heißt der Mittelwert des Ausgangsbildes wird dann nicht zwangsläufig dem Mittelwert des Ergebnisbildes zugeordnet.
Gamma kann jeden Wert zwischen Null und Unendlich annehmen. Dabei gilt: Ist Gamma 1 (Standard), dann erfolgt ein lineares „Mapping“. Falls Gamma kleiner als 1 ist, dann wird das „Mapping“ in Richtung höherer (also heller) Ausgabewerte gewichtet und falls Gamma größer als 1 ist, dann gibt es eine Verzerrung in Richtung niedrigerer (also dunkler) Ausgabewerte.

Abbildung 4.10 zeigt genau diese Beziehungen: in den drei Transformationskurven beziehungsweise –graphen stellt die x-Achse die Intensitätswerte des Eingabebildes dar und die y-Achse gibt dann die Intensitätswerte des Ausgabebildes wieder.

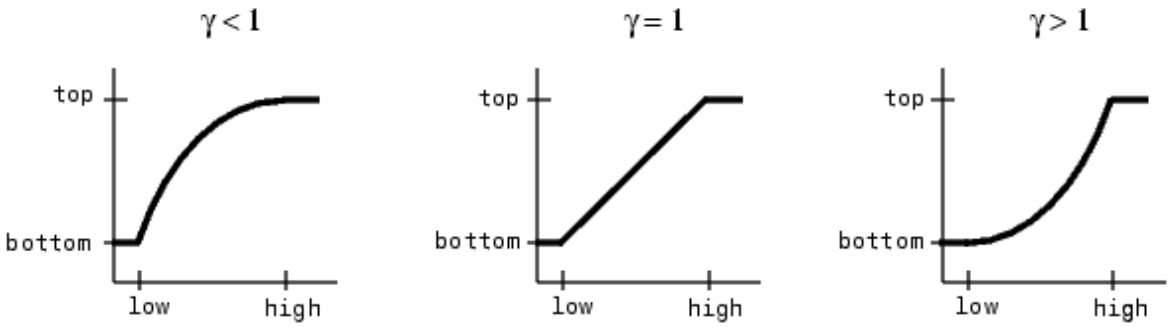


Abbildung 4.10: Drei verschiedene Graphen, die den Zusammenhang der Grauwerte zwischen Eingabe- und Ausgabebild bei verschiednen Gammawerten wiedergeben

In Abbildung 4.11 sieht man das Ergebnis bei Anwendung eines Gammawertes von 0.5.

Abbildung 4.11: (links) Ausgangsbild, (rechts) gammakorrigiertes Ergebnisbild

Man kann nun die beschriebenen Normalisierungen sofort auf den gesamten Frame anwenden oder man macht eine lokale Normalisierung über die einzelnen Fenster beziehungsweise „Patches“, die man beim Abscannen des Eingabebildes erhält. Im Kontext der Gesichtsmerkmal-Erkennung und der verwendeten Fenstertechnik bietet sich eher die lokale Normalisierung an. Man denke beispielsweise an den Fall, dass nur das rechte Auge durch die Sonneneinstrahlung geblendet wird und der Rest des Gesichtes „normalen“ Lichtbedingungen ausgesetzt ist.

4.4 Varianten der Merkmalsextraktion - Featureberechnung

Nachdem die Rohdaten über die Fenstertechnik generiert wurden und gegebenenfalls eine Vorverarbeitung beziehungsweise eine Normalisierung dieser Daten stattgefunden hat, folgt nun die Merkmalsextraktion, die neben den später beschriebenen Klassifikationsmethoden sicherlich eine zentrale Rolle im Erkennungssystem einnimmt.

In dieser Diplomarbeit wurden die folgenden Featureberechnungen implementiert, auf die dann nachher detaillierter eingegangen wird:

- Verarbeitung der *Rohdaten an sich* (ohne weitere Berechnungen) und gegebenenfalls zur Dimensionsreduktion in Kombination mit der *PCA*
- *Zentrierte stochastische Momente*
- *Projektion* beziehungsweise *Summenbildung* der Intensitätswerte in x-, y- und Diagonalenrichtung
- *Singulärwertzerlegung (SVD)* der über die Fenstertechnik erhaltenen Matrizen

Weitere mögliche Merkmale, die zwar untersucht aber nicht direkt implementiert wurden, weil sie zu viele Nachteile bieten oder zu aufwendig sind, wären:

- *FFT-Berechnungen* auf den einzelnen Fenstern beziehungsweise „Patches“
- *Gradienten-/Kantenbilder* in Kombination mit der sogenannten *„Chamferdistanz“*, um mit der Korrespondenz von erwarteten und erhaltenen Kanten umgehen zu können
- *Positionsangaben (x,y)* in Kombination mit einer *Gaußmodellierung*

Bei der FFT-Variante hat sich herausgestellt, dass die hier verwendeten Fenstergrößen einfach zu klein sind. Obwohl die FFT einige Vorteile bietet (vor allem die Verschiebungsinvarianz) sollte man sich doch bewusst sein, dass es sich hierbei um eine „teuere“, das heißt rechenaufwendige Operation zur Merkmalsextraktion handelt.

Schon eher bietet sich da die Berechnung des Gradientenbildes an. Jedoch hat man dabei das große Problem, dass Information aufgrund von Beleuchtungsbedingungen ziemlich schnell verloren geht beziehungsweise dass die Ergebnisbilder sehr verrauscht und unbrauchbar werden (vergleiche dazu die Überlegungen im Abschnitt 3.2.2 des Kapitel 3 und Abb. 3.5). Will man diesen Kantenansatz trotzdem verfolgen, müsste man sich eine Bibliothek von prototypischen Augen, Nasenlöcher und Mundwinkel anlegen und dann über die Berechnung der „Chamferdistanz“ auf jedes Fenster des Eingabeframes eine Entscheidung über das Vorhandensein der entsprechenden Gesichtsmerkmale treffen.

Schließlich könnte man auch einfach die Position der Gesichtsmerkmale als Feature benutzen, zumal man von frontalen Gesichtsaufnahmen ausgeht. Über eine Gaußmodellierung hätte man zwar kleine Streuungen von der angenommenen Position im Griff, aber bei starken und raschen Kopfbewegungen oder einer (komplett) anderen Kameraausrichtung dürfte dieser Ansatz ziemlich schnell fehlschlagen.

4.4.1 Rohdaten an sich

Verwendet man ein Fenster der Breite b und der Höhe h zum „Abscannen“, so erhält man die (vorverarbeiteten/normalisierten) Rohdaten entweder in Matrixform

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1b-1} & x_{1b} \\ x_{21} & x_{22} & \cdots & x_{2b-1} & x_{2b} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{h-11} & x_{h-12} & \cdots & x_{h-1b-1} & x_{h-1b} \\ x_{h1} & x_{h2} & \cdots & x_{hb-1} & x_{hb} \end{pmatrix} \tag{4.1}$$

wobei die x_{ij} Graustufen- beziehungsweise Intensitätswerten zwischen 0 und 255 entsprechen oder auf ein Intervall zwischen 0 und 1 abgebildet werden.

Alternativ kann man auch die Spalteneinträge dieser Matrix von Spalte 1 angefangen bis Spalte b aneinander hängen und bekommt nun einen großen Spaltenvektor der Länge h*b

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{h \cdot b-1} \\ x_{h \cdot b} \end{pmatrix} \tag{4.2}$$

Die Dimension dieses Rohdaten-Vektors wächst mit steigender Fenstergröße ziemlich schnell an: bei einem 16x16 Fenster hat man bereits 256 Dimensionen und bei einem 20x20 großen Fenster erhält man sogar schon 400 Dimensionen! Da der Rechenaufwand mit der Merkmalsdimension beträchtlich zunimmt und außerdem die Klassifikatoren oft mit vielen Merkmalen zur Überadaption neigen, vor allem wenn die Menge der Lernbeispiele zu klein ist, werden Bewertungsverfahren zur Auswahl von Merkmalen aus einem Satz der zur Verfügung stehenden Merkmale oder Verfahren zur Dimensionsreduktion (Abschnitt 4.5.1.4) durch Projektion benötigt. Zur Lösung dieses Problems kann man beispielsweise (wie hier implementiert) die PCA auf die (zu großen) Merkmalsvektoren anwenden.

4.4.2 Zentrierte stochastische Momente

Um weitestgehend unabhängig von der Fenstergröße und den Beleuchtungsbedingungen zu sein, bietet sich die Merkmalsextraktion über die zentrierten stochastischen Momente an.
Das k-te Moment der zur Stichprobe (4.2) (beziehungsweise (4.1)) gehörenden empirischen Verteilung ist laut Wahrscheinlichkeitstheorie $\frac{1}{h \cdot b} \cdot \sum_{i=1}^{h \cdot b} x_i^{\,k}$.

Der Mittelwert μ von (4.2) (beziehungsweise (4.1)) ist also das erste Moment:

$$\mu = \frac{1}{h \cdot b} \cdot \sum_{i=1}^{h \cdot b} x_i \tag{4.3}$$

Das k-te zentrierte stochastische Moment m_k von (4.2) (beziehungsweise (4.1)) ist dann für $k \geq 2$ folgendermaßen definiert:

$$m_k = \frac{1}{h \cdot b} \cdot \sum_{i=1}^{h \cdot b} (x_i - \mu)^k \tag{4.4}$$

Die Varianz oder Streuung von (4.2) (beziehungsweise (4.1)) ist gleichbedeutend mit dem zweiten zentrierten Moment.

Es wurden zwei Varianten implementiert, einmal mit (I) und einmal ohne (II) Mittelwert als erste Merkmalskomponente.

$$\text{(I) } \vec{m} = \begin{pmatrix} \mu \\ m_2 \\ \vdots \\ m_k \end{pmatrix} \qquad \text{und} \qquad \text{(II) } \vec{m} = \begin{pmatrix} m_2 \\ \vdots \\ m_k \end{pmatrix} \tag{4.5}$$

Es lassen sich aus den zentrierten Momenten (Gleichung (4.4)) auch noch andere statistische Merkmale berechnen, wie zum Beispiel:

- Schiefe: $s = \dfrac{m_3}{\sqrt{(m_2)^3}}$
- Kurtosis: $k = \dfrac{m_4}{(m_2)^2}$
- Energie: $e = \sum_{i=1}^{h \cdot b} x_i^{\,2}$
- Entropie: $p = -\sum_{i=1}^{h \cdot b} |x_i|^2 \cdot \log |x_i|^2$
- Normierte Entropie: $n = -\sum_{i=1}^{h \cdot b} |\overline{x_i}|^2 \cdot \log |\overline{x_i}|^2$ mit $\overline{x_i} = \dfrac{x_i}{\sqrt{\sum_{i=1}^{h \cdot b} x_i^{\,2}}}$

Zusätzlich können auch noch die folgenden aus dem Mittelwert beziehungsweise Wertebereich abgeleiteten Merkmale gebildet werden:

- Mittelwert ohne die 5% niedrigsten und 5% höchsten Werte (90%-Mittelwert)
- Maximale Abweichung der Daten vom Mittelwert
- Harmonischer Mittelwert
- und die Wertespanne

4.4.3 Projektion / Summenbildung

Die nächste Merkmalsmethode lehnt sich an den im Stand der Technik bereits beschriebenen Ansatz [23] und an die von Professor Kanade in seiner Doktorarbeit verwendete Projektionsmethode zur Gesichtskonturbestimmung ([24]) an.

Das horizontale Profil wird durch die Projektion beziehungsweise Summenbildung der Intensitätswerte in y-Richtung berechnet („Spaltensumme“)

$$Spaltensumme\ X = \left(\sum_{i=1}^{h} x_{i1},\quad \sum_{i=1}^{h} x_{i2},\quad \cdots,\quad \sum_{i=1}^{h} x_{ib} \right) \tag{4.6}$$

und das vertikale Profil wird durch die Projektion beziehungsweise Summenbildung der Intensitätswerte in x-Richtung bestimmt („Zeilensumme“)

$$Zeilensumme\ X = \begin{pmatrix} \sum_{j=1}^{b} x_{1j} \\ \sum_{j=1}^{b} x_{2j} \\ \vdots \\ \sum_{j=1}^{b} x_{hj} \end{pmatrix} \tag{4.7}$$

Zusätzlich wird noch die Spur (im Englischen „trace“ genannt), also die Summation der Hauptdiagonalelemente der quadratischen Matrix X (Gleichung (4.1)) beziehungsweise des quadratischen Fensters (der Breite b und Höhe b) sowie die Summe der „Gegendiagonalelemente“ (hier „SpurII“ genannt) berechnet.

$$Spur\ X = \sum_{j=1}^{b} x_{jj} \qquad \text{und} \qquad SpurII\ X = \sum_{j=1}^{b} x_{(b+1-j)j} \tag{4.8}$$

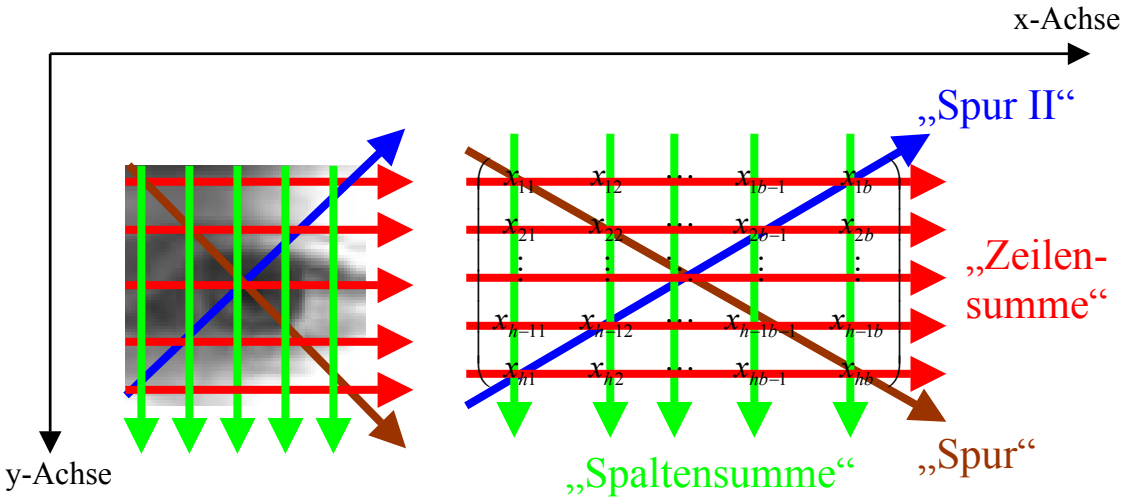


Abbildung 4.12: Darstellung der Spalten-, Zeilensumme, der Spur und der SpurII (links) im Originalfensterbild, (rechts) in der Matrixform

Natürlich kann man die einzelnen Summenbildungen auf verschiedene Weise kombinieren. Tabelle 4.1 zeigt alle möglichen Kombinationen auf (mehr oder weniger nach Dimension geordnet), die im Erkennungssystem implementiert wurden.

Summenbildungs-/Projektionsart	Bedeutung
1	Spur
2	SpurII
3	Spur & SpurII
4	Spaltensumme
5	Zeilensumme
6	Spaltensumme & Spur
7	Spaltensumme & SpurII
8	Spaltensumme & Spur & SpurII
9	Zeilensumme & Spur
10	Zeilensumme & SpurII
11	Zeilensumme & Spur & SpurII
12	Spaltensumme & Zeilensumme
13	Spaltensumme & Zeilensumme & Spur
14	Spaltensumme & Zeilensumme & SpurII
15	Spaltensumme & Zeilensumme & Spur & SpurII

Tabelle 4.1: Überblick über alle implementierten Summenkombinationen/Projektionsarten

4.4.4 Singulärwertzerlegung (SVD)

In [13] werden wie bereits in Kapitel 2 erwähnt die Bildmerkmale beziehungsweise Bildfeatures in vier Gruppen eingeteilt:

- visuelle Merkmale
- statistische Pixelmerkmale
- Merkmale, beruhend auf Transformationskoeffizienten
- und algebraische Merkmale

Die letzteren geben die intrinsischen (das heißt innewohnenden) aber nicht unbedingt sichtbaren Merkmale eines Bildes wieder und werden durch sogenannte Singulärwerte als Ergebnis der Singulärwertzerlegung (im Englischen „singular value decomposition“ (SVD) genannt) einer Matrix repräsentiert. Dieser Merkmalsvektor aus Singulärwerten ist äußerst stabil und invariant gegenüber Varianz der Bildintensität, Transposition, Rotation, Translation und Spiegelung.

Für eine genaue mathematische Beschreibung der Singulärwertzerlegung sei auf ein Mathematik-Lehrbuch wie den „Bronstein“ ([102]) verwiesen.

4.5 Beispielhafte Klassifikatoren

Für die Klassifikation in der Diplomarbeit wurden ausschließlich Polynomklassifikatoren und Entscheidungsbäume eingesetzt. Deswegen wird in den nachfolgenden Abschnitten eine Einführung in die Theorie und Funktion einerseits des Polynomklassifikators und andererseits des Entscheidungsbaums gegeben. Weiterführende Literatur, in der auch andere Klassifikatoren (beispielsweise der Bayes Klassifikator und das Multilayer-Perzeptron) beschrieben werden, findet sich in [5] und [6].

Zunächst wollen wir aber auf die Grundbegriffe der Klassifikation näher eingehen.
Die Aufgabe der Klassifikation ist es, Gruppen von Messwerten symbolisch zu kategorisieren. Pro Messung werden *N* Messwerte erhalten. Die Messungen werden einer fest vorgegebenen Zahl *K* von Gruppen zugeordnet.

Die Messdaten werden auch als Merkmale bezeichnet (siehe Teilkapitel 4.4). Alle Messdaten, die bei einer Messung ermittelt werden, können als *N*-dimensionaler Merkmalsvektor *v* geschrieben werden.

$$v = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_N \end{pmatrix} \tag{4.9}$$

Der Merkmalsvektor v repräsentiert meistens einen Punkt im $\Re^N$, welcher auch als Merkmalsraum bezeichnet wird. Die Aufgabe der Klassifikation ist es, diese Messungen oder Merkmale bestimmten Gruppen zuzuordnen. Die einzelnen Gruppen werden in der Mustererkennung als Klassen bezeichnet. Diese Klassen werden in der Menge Ω zusammengefasst, welche die einzelnen Klassen ω_1 bis ω_K enthält:

$$\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\} \tag{4.10}$$

In der Menge der zu unterscheidenden Klassen Ω besteht zunächst kein zahlenmäßiger Zusammenhang zwischen den Elementen. Um dies zu erreichen, wird der Klassenindex k eingeführt, der die Nummern der Klassen widerspiegelt. Der Klassenindex k ist eine natürliche Zahl aus dem Intervall 1,2,...,*K*.

$$\begin{array}{rcccccc} \Omega & = & \{\omega_1, & \omega_2, & \ldots, & \omega_K\} \\ & & \Updownarrow & \Updownarrow & & \Updownarrow \\ k & \in & \{1, & 2, & \ldots, & K\} \end{array} \tag{4.11}$$

Auf der Zahlengeraden existieren Nachbarschaftsbeziehungen, z.B. |3-1| > |3-2|, die es in der Menge Ω nicht gibt. Um diese Beziehungen zu unterbinden, werden Zielvektoren y eingeführt. Die Zielvektoren y sind Einheitsvektoren die im *K*-dimensionalen Raum den Klassen entsprechen. Werden die Zielvektoren zu einer Matrix zusammengefasst, so ergibt sich die Einheitsmatrix

$$\mathrm{I}^K = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix} = \left(y_1, y_2, \dots, y_K\right) \tag{4.12}$$

Die Aufgabe des Erkennungssystems besteht darin, den vorliegenden Merkmalsvektor *v* über eine für diesen Zweck geeignete Funktion auf den zugehörigen Zielvektor *y* abzubilden. Diese Funktion gehört zu einer Familie von Funktionen, die über eine niedrige Anzahl von Parametern parametrisiert wird. Mit verschiedenen Grundstrukturen gelangt man zu verschiedenen Familien der Abbildungsfunktionen, innerhalb derer die Funktionen durch Parameterveränderung geformt und gestaltet werden können. Dadurch wird die Aufgabe, ein Erkennungssystem für ein bestimmtes Einsatzgebiet zu konstruieren, zur Aufgabe der Parameteroptimierung.

Das Optimierungsziel lautet in diesem Fall: $f(v) = y_k$ mit $v \in \omega_k$. Diese Vorgabe führt auf eine Menge von Wertepaaren an Merkmalsvektoren *v* und erwünschtem Ergebnis *y*, die in einer Tabelle $\begin{pmatrix} v_1, v_2, \dots, v_K \\ y_1, y_2, \dots, y_K \end{pmatrix}^T$ zusammengestellt sind. Das Optimierungsziel besteht nun darin, die tatsächlich erzeugten Funktionswerte *d* den Sollvorgaben *y* möglichst anzunähern.

Die in der Tabelle zusammengestellten Wertepaare $[v, y]$ sind Beispiele, an denen die Abbildungsfunktion angepasst wird. Das Erkennungssystem lernt also anhand der Beispiele, wie es sich verhalten soll. Deswegen wird das „Belehrungsmaterial“ auch die Lernstichprobe oder das Trainingsset genannt.

In Abbildung 4.13 wird das Trainieren des Erkennungssystems schematisch dargestellt.
Die Lernstichprobe ist nur eine Teilmenge aller möglichen $[v, y]$-Paare. Somit kann das Erkennungssystem nur anhand dieser Stichprobe lernen. Die Frage ist, wie gut das Erkennungssystem auf weiteren unabhängig gebildeten Stichproben funktioniert. Diese Stichproben, welche zum Testen des Erkennungssystems verwendet werden, heißen Teststichproben.

Das Erkennungssystem wird im Allgemeinen eine bessere Leistung bei der Lernstichprobe als bei der Teststichprobe zeigen. Der Test an der Lernstichprobe wird mit dem Begriff Reklassifizierung bezeichnet. Ausschlaggebend für die Leistung des Erkennungssystems in der realen Umgebung ist die Generalisierungsfähigkeit, das heißt wie gut der Klassifikator auf Teststichproben ist.

Ein entscheidender Parameter ist der Stichprobenumfang. Mit wachsendem Stichprobenumfang für die Lern- beziehungsweise Teststichprobe sollten die Reklassifizierungs- und Generalisierungswerte gegen die realen Werte gehen. Theoretisch ([7]) ist es möglich, den Stichprobenumfang so zu bestimmen, dass eine vernünftige Aussage über die Performance des Klassifikators möglich wird. Im Allgemeinen gilt aber, dass Erkennungssysteme mit mehr Anpassungsfähigkeit und größerer Anzahl an freien Parametern auch umfangreichere Lern- und Teststichproben benötigen.

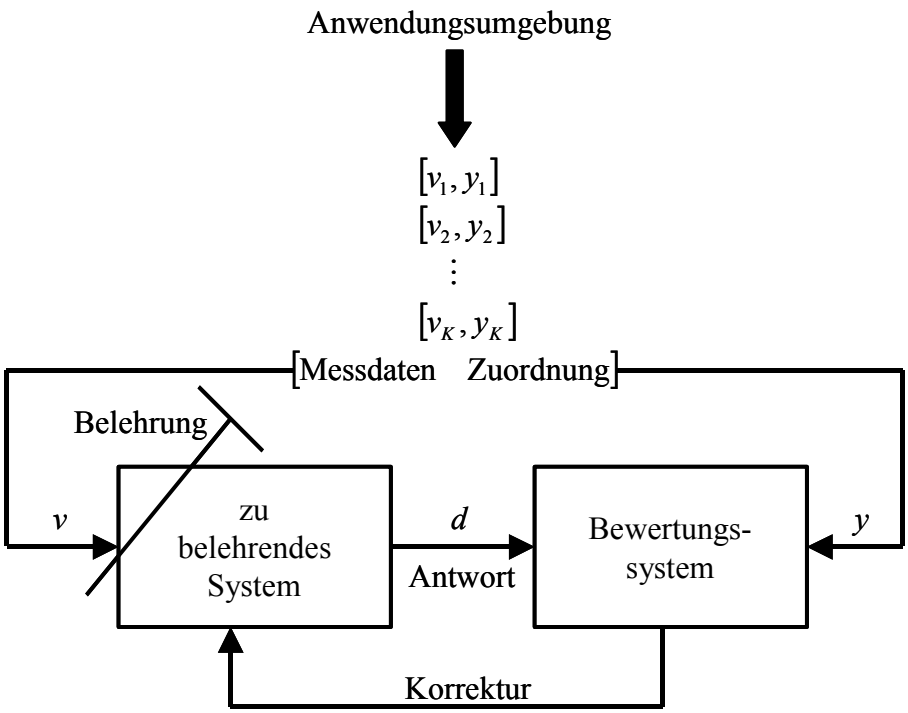


Abbildung 4.13: Lernen aus Beispielen: Die Anwendungsumgebung erzeugt Wertepaare von Messdaten (Merkmalsvektoren *v*) und Sollzuordnungen (erwünschtes Ergebnis *y*). Die Werte werden in einer Tabelle gespeichert und belehren das Erkennungssystem

4.5.1 Polynomklassifikator

Zuerst betrachten wir den Polynomklassifikator und dessen Grundlagen.

4.5.1.1 Grundbegriffe und Theorie

Beim Polynomklassifikator werden die Entscheidungsfunktionen durch Polynome, also eine wohldefinierte Grundstruktur, gebildet.
Nach dem Approximationssatz von Weierstraß kann jede stetige Funktion beliebig genau durch Polynome angenähert werden, sofern der Grad des Polynoms groß genug gewählt wird. Die Entscheidungsfunktion $d_k(v)$ hat mit dem Polynomansatz mit dem Grad G folgende Struktur:

$$\begin{aligned} d_k(v) = &\quad a_{0,k} + a_{1,k}v_1 + a_{2,k}v_2 + \cdots + a_{N,k}v_N + a_{N+1,k}v_1^2 + \cdots + a_{M,k}v_K^G \\ = &\, a_{0,k}x_0 + a_{1,k}x_1 + a_{2,k}x_2 + \cdots + a_{N,k}x_N + a_{N+1,k}x_{N+1} + \cdots + a_{M,k}x_M \\ = &\quad a_k^{\,T} \cdot x(v) \end{aligned} \tag{4.13}$$

Die Unterscheidungsfunktion d_k besteht aus einer Linearkombination von Produkttermen in v_i mit den Koeffizienten $a_{j,i}$. $x(v)$ enthält sämtliche Kombinationen aller Produktterme und wird als Polynomstrukturvektor bezeichnet. Werden die *K* Entscheidungsfunktionen

$a_{j,i}$ zum Schätzvektor d zusammengefasst, ergibt sich die folgende kompakte Matrizenschreibweise:

$$d(v) = A^T \cdot x(v) \tag{4.14}$$

Der Polynomstrukturvektor $x(v)$ wird durch den Entwurf vorgegeben. Die Koeffizientenmatrix A wird mit Hilfe der Quadratmitteloptimierung (siehe Abschnitt 4.5.1.2) bestimmt. Werden alle Terme des Polynoms bis zu einem gewissen Grad G verwendet, so wird dies vollständiger Polynomansatz des Grades G genannt. Für die Länge M des Polynoms gilt:

$$M = \binom{N+G}{G} \tag{4.15}$$

Zum Beispiel erhält man für G = 1 eine lineare Entscheidungsfunktion und somit einen linearen Klassifikator. Damit hat der Polynomstrukturvektor $x(v)$ die Form:

$$x(v) = \begin{pmatrix} 1 \\ v_1 \\ v_2 \\ \vdots \\ v_N \end{pmatrix} = \begin{pmatrix} 1 \\ v \end{pmatrix} \tag{4.16}$$

Für einen quadratischen Klassifikator beziehungsweise eine quadratische Entscheidungsfunktion mit G = 2, sieht $x(v)$ folgendermaßen aus:

$$x(v) = \begin{pmatrix} 1 & v_1 & v_2 & \dots & v_N & v_1^2 & v_1 v_2 & v_1 v_3 & \dots & v_N^2 \end{pmatrix}^T \tag{4.17}$$

Analoges Vorgehen gilt für die Bildung des Polynomstrukturvektors höherer Polynomgrade.

In der Tabelle 4.2 sind Beispiele für die Länge des Schätzvektors bei verschiedenen Polynomgraden und bei verschiedener Merkmalsanzahl. Die Länge des Polynoms wächst beim vollständigen Ansatz sehr schnell mit dem Polynomgrad. Bei den meisten realen Anwendungen ist es daher notwendig, durch Dimensionsreduktion (siehe Abschnitt 4.5.1.4) kürzere Merkmalsvektoren zu bekommen, um dadurch den Berechnungsaufwand kleiner zu halten.

	Länge des Merkmalvektors					
Grad G	N = 10	N = 20	N = 30	N = 40	N = 50	N = 60
1	11	21	31	41	51	61
2	66	231	496	861	1326	1891
3	286	1881	5456	12341	23426	39711

Tabelle 4.2: Schätzgleichungslänge für den vollständigen Polynomansatz mit Grad G für *N* Merkmale

4.5.1.2 Quadratmitteloptimierung

Zum Einstellen der Koeffizientenmatrix *A* wird die sogenannte Quadratmitteloptimierung angewendet. Durch die Quadratmitteloptimierung wird der mittlere Fehler der Sollgröße und der einzustellenden Größe über alle Beispiele minimiert. Bei der Klassifikation sind dabei die verwendeten Zielvektoren diese Sollgrößen.
Salopp formuliert liegt in der Koeffizientenmatrix *A* das gesamte Lernpotential!

Es wird nach der Abbildung gesucht, die die Merkmalsvektoren *v* auf die Zielvektoren *y* abbildet. Demzufolge muss $y - A^T x$ für alle Merkmale und deren zugehörigen Zielvektoren *y* minimiert werden.
Somit ergibt sich der Quadratmittelansatz für den Polynomklassifikator:

$$S^2(A_{opt}) = E\left\{\| y - d(v) |^2\right\} = E\left\{\| y - A^T x(v) |^2\right\} \tag{4.18}$$

$S^2(A)$ wird der mittlere quadratische Schätzfehler oder auch die Reststreuung genannt. Die Gleichung 4.18 kann mit Hilfe der Eigenschaft von zwei Vektoren *a* und *b*

$$a^T b = Spur\left[ba^T\right] \tag{4.19}$$

und allgemeiner mit zwei Hilfsmatrizen P und Q mit übereinstimmender Dimension

$$Spur[QP] = Spur[PQ] \tag{4.20}$$

umgeformt werden. Somit ergibt sich

$$\begin{aligned}
S^2(A) &= E\left\{ \left| y - A^T x \right|^2 \right\} \\
&= E\left\{ \left| y - A^T x \right|^T \cdot \left| y - A^T x \right| \right\} \\
&= E\left\{ Spur\left[\left| y - A^T x \right| \cdot \left| y - A^T x \right|^T \right] \right\} \\
&= Spur\left[E\left\{yy^T\right\} - E\left\{yx^T\right\} \cdot A - A^T \cdot E\left\{xy^T\right\} + A^T \cdot E\left\{xx^T\right\} \cdot A \right] \\
&= Spur\left[E\left\{yy^T\right\}\right] + Spur\left[A^T \cdot E\left\{xx^T\right\} \cdot A\right] - 2Spur\left[A^T \cdot E\left\{xy^T\right\}\right] \\
&= E\left\{|y|^2\right\} + Spur\left[A^T \cdot E\left\{xx^T\right\} \cdot A\right] - 2Spur\left[A^T \cdot E\left\{xy^T\right\}\right]
\end{aligned} \tag{4.21}$$

Das Ziel ist es $S^2(A)$ zu minimieren. Die Lösung wird durch einen Variationsansatz bestimmt: *A* sei die gesuchte Lösung, das heißt jede Abweichung von *A* um δA führt zu einer Verschlechterung des Ergebnisses von *A*

$$S^2(A + \delta A) \geq S^2(A) \qquad \forall \delta A \neq 0 \tag{4.22}$$

Die rechte Seite der Ungleichung ist durch die Gleichung 4.21 gegeben. Für die linke Seite ergibt sich:

$$\begin{aligned}
S^2(A + \delta A) = {} & E\left\{|y|^2\right\} + Spur\left[A^T \cdot E\left\{xx^T\right\} \cdot A\right] - 2Spur\left[A^T \cdot E\left\{xy^T\right\}\right] + \\
& 2Spur\left[\delta A^T E\left\{xx^T\right\} A\right] + Spur\left[\delta A^T E\left\{xx^T\right\} \delta A\right] - \\
& 2Spur\left[\delta A^T E\left\{xy^T\right\}\right]
\end{aligned} \tag{4.23}$$

Werden die Gleichungen 4.21 und 4.23 in die Ungleichung 4.22 eingesetzt, ergibt sich die neue Ungleichung

$$Spur\left[\delta A^T E\left\{xx^T\right\}\delta A\right] - 2Spur\left[\delta A^T \left(E\left\{xy^T\right\} - E\left\{xx^T\right\}A\right)\right] \geq 0 \tag{4.24}$$

Der erste Term $Spur\left[\delta A^T E\left\{xx^T\right\}\delta A\right] = E\left\{\left|\delta A^T x\right|^2\right\} \geq 0$ kann für beliebiges δA nicht negativ werden. Die Ungleichung 4.24 ist notwendigerweise erfüllt, wenn der zweite Term verschwindet. Durch das Nullsetzen des Ausdrucks in den runden Klammern ergibt sich die Bestimmungsgleichung für die optimale Koeffizientenmatrix A.

$$\begin{aligned} E\left\{xx^T\right\} \cdot A &= E\left\{xy^T\right\} \\ A_{opt} &= E\left\{xx^T\right\}^{-1} \cdot E\left\{xy^T\right\} \end{aligned} \tag{4.25}$$

Die Gleichung 4.25 kann nur gelöst werden, wenn die Momentenmatrix $E\left\{xx^T\right\}$ invertierbar ist, das heißt die Matrix darf nicht singulär sein ($\det(E\left\{xx^T\right\}) \neq 0$). Im Allgemeinen kann diese Voraussetzung nicht angenommen werden, da damit zu rechnen ist, dass mindestens zwei Zeilen oder Spalten linear abhängig sind. Damit gibt es für die Koeffizientenmatrix A keine eindeutige Lösung, sondern mehrere Lösungen. Aus allen möglichen Lösungen kann eine besonders geeignete Lösung erzwungen werden. Dies ist mit einem modifizierten Gauß-Jordan-Algorithmus und einer geeigneten Pivotstrategie möglich. Pivotstrategien werden hier nicht näher erklärt, sondern es wird auf die Literatur verwiesen ([5], [6])

4.5.1.3 ONEvsALL (OA)- und PAIRWISE (PW)-Struktur

Bei dem Polynomklassifikator können verschiedene Klassifikatorstrukturen gewählt werden. Bei Mehrklassen-Problemen wird üblicherweise das K-Klassen-Problem in K einzelne Probleme aufgeteilt, wobei jeder Klassifikator genau eine Klasse von allen anderen separieren soll. Dieses Verhalten der Klassifikatoren wird mit *ONEvsALL* (one class versus all others, hier abgekürzt mit **OA**) bezeichnet. Ein Beispiel ist in der Abbildung 4.14 auf der linken Seite gegeben. Es ist ein zweidimensionales Problem mit drei Klassen (+,#,*). Jeder der drei Linien teilt eine Klasse von den beiden anderen ab. Die Entscheidungen der einzelnen Klassifikatoren müssen noch zu einer Gesamtentscheidung zusammengefasst werden. Gewöhnlich wird die sogenannte „*winner-takes-all*“-Strategie eingesetzt ([8]).

Eine andere Möglichkeit besteht darin, die Klassengrenzen jeweils paarweise zu bestimmen (*PAIRWISE*, hier abgekürzt mit **PW**). Dazu werden je zwei Klassen miteinander verglichen und die anderen K-2 Klassen werden momentan nicht berücksichtigt. Dadurch ergeben sich insgesamt $\frac{K \cdot (K-1)}{2}$ Entscheidungsfunktionen. In Abbildung 4.14 auf der rechten Seite sind die Klassengrenzen für ein zweidimensionales drei Klassenproblem abgebildet. Es werden die Klassen ’#’ gegen ’+’, ’#’ gegen ’*’ und ’*’ gegen ’+’ miteinander verglichen. Ein min/max-Entscheider liefert die endgültigen Klassengrenzen. Weitere Informationen sind in [5] und [8] aufgeführt.

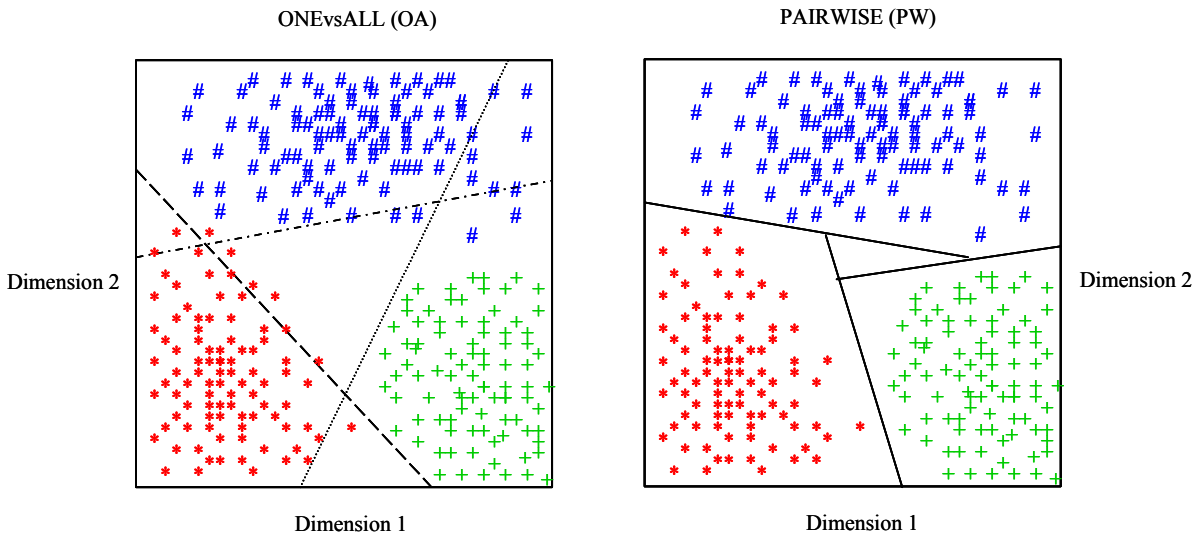


Abbildung 4.14: Klassengrenzen für ein K = 3 Klassenproblem mit den Klassifikatorstrukturen ONEvsALL (OA) und PAIRWISE (PW)

4.5.1.4 Dimensionsreduktion

Aus der Tabelle 4.2 wird ersichtlich, dass die Länge des Polynoms mit steigendem Grad und zunehmender Dimension N des Merkmalvektors sehr stark zunimmt. Der Stichprobenumfang muss bei einer größeren Länge des Polynoms ebenfalls vergrößert werden.

Mit Hilfe der Dimensionsreduktion ist es möglich, von einer großen Anzahl N an Merkmalen auf eine handhabbare Anzahl $M < N$ herunterzukommen. Die Schwierigkeit bei der Dimensionsreduktion besteht darin, die richtigen M Merkmale zu finden, die die Leistung des Klassifikators in der Erkennungsrate möglichst wenig beeinflussen. Die N Merkmale spannen mathematisch betrachtet einen N-dimensionalen Merkmalsraum auf. Durch die Auswahl der M Merkmale spannen diese im N-dimensionalen Raum einen M-dimensionalen Unterraum auf. Um diese M Merkmale auszuwählen, wird üblicherweise die sogenannte *Hauptachsentransformation* (abgekürzt mit **HAT**) eingesetzt. Alle Merkmale im N-dimensionalen Raum bilden eine Art Massenverteilung, die durch ihre Hauptachsen definiert ist und einen Schwerpunkt besitzt. Die HAT legt den Schwerpunkt der Merkmalsverteilung in den Ursprung eines neuen Koordinatensystems und dreht das alte Koordinatensystem entsprechend der Massenverteilung. In Abbildung 4.15 ist dies beispielhaft für eine zweidimensionale Verteilung dargestellt. Die Eigenvektoren b_m der HAT stellen die Achsen des neuen Koordinatensystems dar. Die zugehörigen Eigenwerte λ_m sind eine Rangliste der erzeugten Eigenvektoren. Es werden diejenigen Eigenvektoren ausgewählt, die die M größten Eigenwerte aufweisen, das heißt es werden die Eigenvektoren ausgewählt, die den größten Einfluss auf das Gesamtergebnis haben. Die restlichen M-N Eigenvektoren werden nicht weiter berücksichtigt. Mit dieser Operation wird der ursprüngliche Merkmalsvektor v in einen transformierten und gekürzten Merkmalsvektor $\hat{w}$ abgebildet.

Die Transformation wird über die ersten beiden statistischen Momente aller Merkmalsvektoren *v* berechnet. Das ist der Erwartungswert μ

$$\mu = E\{v\} \tag{4.26}$$

und die Kovarianzmatrix *K*

$$K = E\{(v-\mu)(v-\mu)^T\}. \tag{4.27}$$

Nach Umformungen ergibt sich folgende Transformationsvorschrift für die HAT:

$$w = B^T \cdot (v-\mu) \tag{4.28}$$

Die Matrix *B* ist die spaltenweise Anreihung der M Eigenvektoren von *K*, sortiert nach abfallenden Eigenwerten λ_m. Die Eigenvektoren b_m sind untereinander orthogonal. Die Rücktransformation für die transformierten und gekürzten Merkmalsvektoren $\hat{w}$ wird mit

$$\hat{v} = B \cdot \hat{w} + \mu \tag{4.29}$$

berechnet.
Der mit der Dimensionsreduktion verbundene Rekonstruktionsfehler R^2 kann aus den Eigenwerten λ_m berechnet werden.

$$R^2 = E\{|\hat{v}-v|^2\} = \sum_{m=M+1}^{N} \lambda_m \tag{4.30}$$

Unter dem Begriff Rekonstruktionsfehler wird verstanden, dass der transformierte und gekürzte Merkmalsvektor $\hat{w}$ benutzt wird, um über die Rücktransformation $\hat{w} \rightarrow \hat{v}$ der HAT den ursprünglichen Merkmalsvektor *v* zu rekonstruieren. Der Erwartungswert für den entstehenden Fehler $|\hat{v}-v|^2$ wird minimiert.

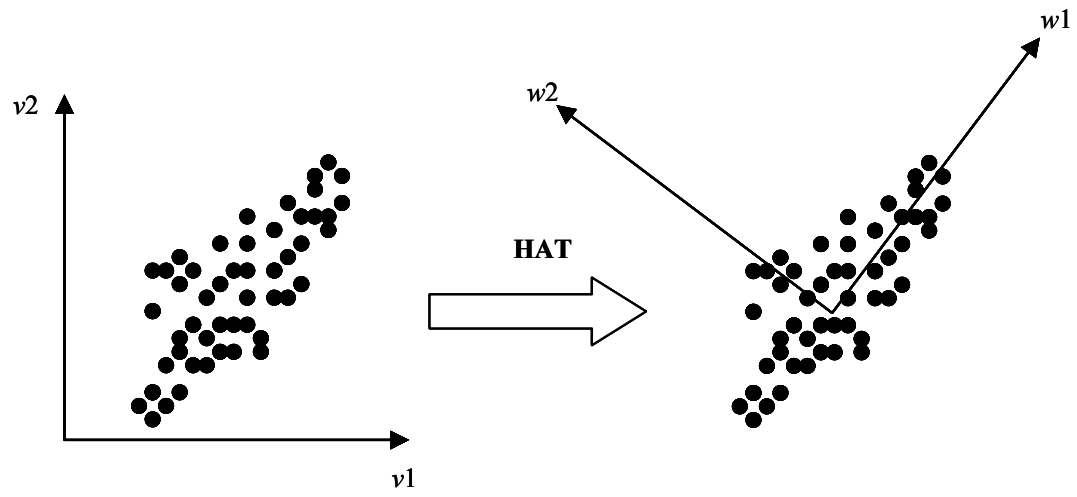


Abbildung 4.15: Hauptachsentransformation (HAT) anhand eines zweidimensionalen Beispiels

4.5.2 Entscheidungsbäume

Als typischer Vertreter des Induktiven Lernens (siehe Abschnitt 2.3.4 „Appearance-based“ Methoden im Unterkapitel zur Kategorisierung der Gesichtsdetektion) können Entscheidungsbäume (im Englischen „decision trees“ genannt) angesehen werden.
Wie bereits im „Stand der Technik“-Kapitel erwähnt, wird in [38] ein Entscheidungsbaum als Klassifikator für Gesicht und „Nichtgesicht“ eingesetzt. Natürlich kann man diesen Ansatz auf die Klassifizierung von Gesichtsmerkmalen erweitern: man hat nun statt zwei Klassen sieben Klassen, jeweils eine für die einzelnen Gesichtsmerkmale und den Hintergrund.

Die Motivation Entscheidungsbäume einzusetzen liegt in den folgenden Punkten:
- Bäume sind wegen ihrer Geschwindigkeit interessant: irgendwann sollen die Algorithmen in einem Mikrokontroller (und nicht auf 1 GHz-Prozessor) laufen!
- Man erhält möglicherweise eine gute Partionierung.
- Manchmal kommt man schon mit wenigen Merkmalen zur Lösung.
- In einer Entscheidungsstufe müssen nicht alle Pixel „angefasst“ werden.

Neben der Klassifizierung von Objekten (wie hier in der Bildverarbeitung), sind Bäume vor allem in der Medizin bei der Einschätzung von Krankheitsbildern mittels (messbarer) Symptome sehr beliebt.

4.5.2.1 Grundbegriffe und Theorie

Ein Entscheidungsbaum besteht im wesentlichen aus Knoten (sogenannte Testattribute), die bestimmte Werte (numerisch (diskret oder kontinuierlich) oder auch nominal) annehmen können und aus Blättern, die dann das Ergebnisse (hier die Klassenzugehörigkeit) liefern. Jeder Ast beschreibt schließlich, welche Attributwert-Kombination welche Klasse liefert. Abbildung 4.16 dient als Anschauung: Anhand der Testattribute ‚Größe’, ‚Oberfläche’ und ‚Form’ werden die zu klassifizierenden Objekte einer bestimmten Objektgruppe zugeordnet (‚Positiv’) oder eben nicht (‚Negativ’) (sogenanntes „Zwei-Klassen-Problem“).

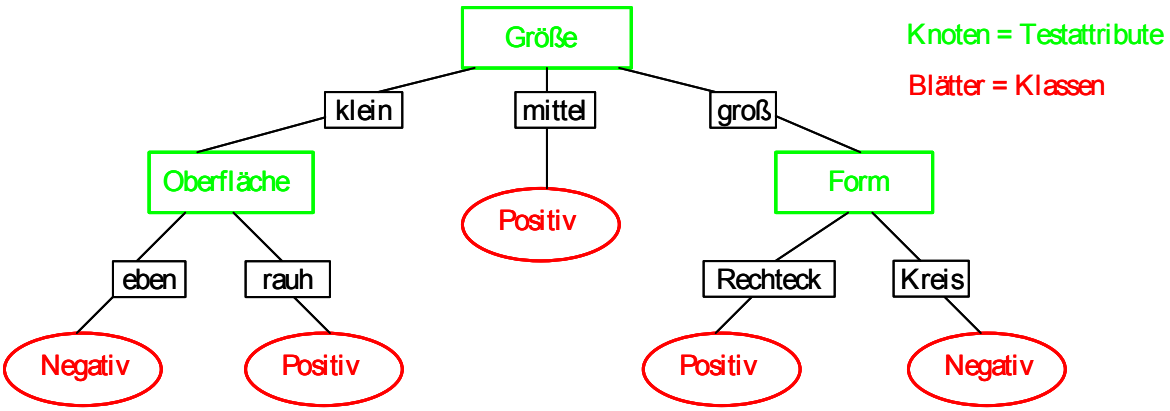


Abbildung 4.16: Exemplarischer Entscheidungsbaum

Zentrale Fragen bei Entscheidungsbäumen sind unter anderem:

- Auswahl der geeignetsten Testattribute? (eventuell auch Kostenberücksichtigung)
- Wie soll der Baum initial aufgebaut werden (nichtinkrementeller Fall)?
- Wie wird der Baum bei Auftreten neuer Ereignisse verändert (inkrementeller Fall)?
- Wie groß ist die maximale Klassifikationsgeschwindigkeit und die Baumkomplexität?

Für den Aufbau beziehungsweise das Training eines Entscheidungsbaumes gibt es mehrere sogenannte „Top Down Induction of Decision Trees"(TDIDT)-Verfahren:

- *ID3* (Quinlan, 1986): Ist ein nichtinkrementelles Training. Die Basisidee besteht darin, das Testattribut aufgrund der Klassifikationsgüte auszuwählen. Dabei soll das Testattribut die Entropie minimieren beziehungsweise den Informationsgewinn maximieren.
- *C4.5* (Quinlan, 1992) und die Nachfolger *C5.0* beziehungsweise *See5*: Es handelt sich dabei um eine Verbesserung von ID3 durch generalisierte Regeln, die man durch nachträgliches Pruning des Ausgangsbaumes erhält.
- *ID5R* (Utgoff, 1989): Ist ein inkrementelles Training. Der Baum ist nach Berücksichtigung aller Beispiele quasi äquivalent zu ID3.

Für weitergehende Erläuterungen wie beispielsweise zu „Splittingregeln", „Pruningverfahren" und Erweiterungen allgemein sei auf die angegebene Literatur ([40], [42], [43]) verwiesen.

Kapitel 5

Experimente und Ergebnisse

5.1 Evaluierung des Erkennungssystems

Viele Veröffentlichungen im Bereich der Gesichtsdetektion vergleichen die Leistung mehrerer Methoden untereinander gewöhnlich mit den Begriffen Detektionsrate und Fehlerkennungshäufigkeit. Natürlich werden auch noch andere Metriken angewendet, um Algorithmen zu evaluieren. Dazu zählen unter anderem die Lernzeit, die Ausführungszeit, die Anzahl der Trainingsbeispiele und das Verhältnis zwischen Detektionsraten und „Fehlalarmen".

5.1.1 Datenbank und „Labeling"

Für das Trainieren und Testen des Erkennungssystems wurde eine SQL-Datenbank (Postgresql) aufgebaut: zur Zeit enthält sie 10 verschiedene Personen mit insgesamt zirka 60000 Frames; es gibt sowohl Tag- als auch Nachtaufnahmen (siehe Anhang A zur Illustration).
Es handelt sich bei den verwendeten Videosequenzen um reale Fahrtaufnahmen, somit wird die Evaluierung im „richtigen", das heißt praxisbezogenen Szenarium gemacht.
Außerdem befinden sich in der Datenbank auch die sogenannten „Metadaten": darunter versteht man Kalmanfilter-Parameter und Klassenlabels. Das „Labeling", also die Zuordnung von Bildregionen zu den sieben Klassen, erfolgt dabei händisch beziehungsweise (mittels eines eigenen Trackers) pseudoautomatisch, um den Zeitaufwand zu reduzieren. Dabei ist das „Labeling" +/- 1 oder 2 Pixel genau. Anhang B zeigt exemplarisch den Aufbau der Labeldatei für das überwachte Lernen und die Auswertung: in jeder Zeile wird dem entsprechenden Frame die Position der einzelnen Gesichtsmerkmale in x- und y-Richtung zugeordnet. Es erfolgt also genaue Zuweisung der einzelnen Pixel zu den folgenden sieben Klassen (in Klammern angegeben ist jeweils die in der Labeldatei angegebe Positionsbeschreibung des zur untersuchten Bildregion gehörenden Zentralpixels):

- *rechtes Auge* („erx" / „ery" = eye right, x-position / y-position)
- *linkes Auge* („elx" / „ely" = eye left, x-position / y-position)
- *rechtes Nasenloch* („nrx" / „nry" = nose right, x-position / y-position)
- *linkes Nasenloch* („nlx" / „nly" = nose left, x-position / y-position)
- *rechter Mundwinkel* („mrx" / „mry" = mouth right, x-position / y-position)
- *linker Mundwinkel* („mlx" / „mly" = mouth left, x-position / y-position)
- und *Hintergrund/„Garbage"*
  (beim „Labeln" werden alle Pixel im Bild, die nicht in die definierte Region um die Zentralpixel der zuvor erwähnten Klassen fallen, automatisch auf Hintergrund gesetzt)

Die ersten sechs Klassen sind genau die Gesichtsmerkmale, wie sie in dieser Diplomarbeit verwendet werden. Natürlich muss man solche Merkmale von „berechneten" Bildmerkmalen unterscheiden!

Es kann auch der Fall auftreten, dass bestimmte Gesichtsmerkmale in einem einzelnen Frame gar nicht auftreten beziehungsweise gefunden werden (zum Beispiel, wenn sich der Fahrer zur Seite dreht): dann steht in der entsprechenden Zeile und Spalte für das nicht vorhandene Gesichtsmerkmal „1“ für die x-Koordinate beziehungsweise „–1“ für die y-Koordinate (siehe rot markiertes Rechteck in der exemplarischen Labeldatei).

5.1.2 Grundbegriffe der Auswertung

Für die nachfolgenden Ausführungen sind die folgenden Begriffe beziehungsweise Definitionen essenziell:

- *Detektionsrate*: darunter versteht man allgemein das Verhältnis zwischen der Anzahl an Gesichtsmerkmalen, die vom Algorithmus korrekt detektiert wurden und der Anzahl an Gesichtsmerkmalen wie sie von einem Menschen beziehungsweise dem „Lehrer“ bestimmt wurden

- *Vordergrund-Detektionsrate* (im Englischen „*foreground detection rate*“ genannt): ist im Prinzip eine Vereinfachung der detaillierten Detektionsrate insofern das nur noch zwischen Vorder- und Hintergrund unterschieden wird (zwei Klassen-Problem) und die einzelnen Klassen für die Gesichtsmerkmale nicht genauer bewertet werden; dies impliziert natürlich bessere Ergebnisse als bei der detaillierten Untersuchung

Die Gesamtfehlerzahl beziehungsweise genauer gesagt die *Gesamtzahl an Falschklassifikationen* (im Englischen „*total amount of misclassification*“), die als Ausdruck *falsch klassifizierte Pixel* in den nachfolgenden Diagrammen auftaucht, setzt sich aus drei Fehlertypen zusammen:

- *Verwechslungen innerhalb der Gesichtsmerkmale* (im Englischen „*misclassification across facial features*“) oder auch *Vertauschungen Gesichtsmerkmalpixel untereinander* in den jeweiligen Diagrammen

- *„False Negatives“*: (eigentliche) Gesichtsmerkmale werden nicht als solche erkannt, sondern in die Hintergrundklasse eingeordnet, wodurch schlechtere Detektionsraten resultieren

- und *„False Positives“*: eine Bildregion wird fälschlicherweise als Gesichtsmerkmal deklariert, obwohl sie zur Hintergrundklasse gehört

Da die Verwechslung von korrespondierenden Gesichtsmerkmalen (das heißt beispielsweise linkes Auge wird als rechtes Auge erkannt oder umgekehrt) aufgrund der Symmetrieeigenschaft des menschlichen Gesichtes „tolerierbar“ ist, gibt es für diese Fehlerbetrachtung noch einen gesonderten Fall bei der Auswertung (im Englischen „*misclassification across facial features, swapping allowed*“ genannt). Dieser Fall wird in den nachfolgenden MATLAB-Diagrammen als *Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)* bezeichnet.

Bei einer „fairen“ Bewertung sollte man all diese Begriffe immer im Hinterkopf haben und kritisch damit umgehen: man kann nämlich oft die Parameter eines Algorithmus so anpassen, dass zwar die Detektionsrate steigt, aber das ganze auf Kosten einer erhöhten Anzahl an Falschklassifikationen!

5.1.3 Konfusionsmatrix

Bei Mehr-Klassen-Problemen bietet sich für die Darstellung der Erkennungsleistung die sogenannte Konfusion- beziehungsweise Verwechslungsmatrix an (siehe auch Anhang C).
Dabei stehen die Zeilen für die Soll-/Zielklassen und die Spalten für die Istklassen, wodurch man eine sehr kompakte Repräsentation der Erkennungsergebnisse hat. Gewünscht sind natürlich möglichst viele Einträge auf der Diagonalen dieser Matrix.

5.1.4 Abstandsabhängige Auswertung der Labelbilder

Um sowohl das bei der Fenstertechnik angesprochene Problem der „halben Augen“ beziehungsweise der „Drittel Mundwinkel“, also der teilweisen Erkennung von Gesichtsmerkmalen, als auch das Problem des etwas ungenauen „Labels“ in den Griff zu bekommen, definiert man um die in der Labeldatei (Anhang B) exakt vorgegebenen Labelpositionen der Gesichtsmerkmale eine tolerierbare Region, so dass Fenster, die sonst „Garbage“ wären und deren Zentralpixel aber innerhalb dieser Region fallen, dann ebenfalls als das entsprechende Gesichtsmerkmal definiert werden.

Die zulässige Ungenauigkeit wurde dabei zuerst fix auf 5 Pixel in x- und y-Richtung festgelegt, das heißt die tolerierbare Region um die vorgegebene Labelposition war 5x5 Pixel groß.

Später beim Vergleich mit dem „Matched Filter“-Ansatz (siehe Auswertung der ROC-Diagramme) konnte man diese Region zur besseren Evaluierung dann variabel über einen Radius einstellen.
Dabei entspricht beispielsweise Radius 2 einer Kreisfläche, deren Pixel einen Abstand <= 2 zur vorgegebenen Labelposition durch die Labeldatei haben (sie ist also sozusagen das Äquivalent zur „ungeraden“ 5x5 Region).

5.2 Experimente in MATLAB

Natürlich will man wissen, wie die Algorithmen in der Praxis funktionieren. Anhand von Experimenten mit einer Trainingsmenge und einer Testmenge, die sich nicht überlappen sollten, folgt nun die praktische Bewertung der untersuchten Ansätze, wobei das Hauptaugenmerk auf dem Polynomklassifikator liegt.

5.2.1 Trainings-/Testsequenzen

Um alle implementierten Algorithmen beziehungsweise deren Variationen vergleichen zu können, wurde eine feste Trainingsmenge und eine feste Testmenge bestimmt, die aber disjunkt zueinander sind. Andere Mengenkonstellationen wurden auch betrachtet, aber nicht im großen Umfang für eine reale Evaluierung.
Für die Trainingsmenge wurde immer jeweils der 20.te Frame einer Sequenz benutzt. Grundlage waren dabei die Sequenzen ‚dariu1', ‚christian1', ‚tilo1' und ‚udo1'.
Natürlich braucht man neben Beispielen für die sechs Gesichtsmerkmale auch welche für „Garbage". Pro Frame werden dabei 128 Hintergrund-Samples so gewürfelt, dass sie nicht zu nah an den (wahren) Labelpositionen der Gesichtsmerkmale sind. Es wird ein minimaler Abstand zu diesen Labelpositionen so definiert, dass er das Maximum von (16, „aktuelle Fensterbreite", „aktuelle Fensterhöhe"), also mindestens 16 Pixel, annimmt.
Bei der Testmenge wurde mit einer Schrittweite von 120 gearbeitet, das heißt jeder 120.te Frame in einer sequentiellen Bildfolge wurde Teil der Testmenge, um den Variationen innerhalb einer Sequenz gerecht zu werden. Die Testsequenzen setzen sich dabei aus den Personen ‚andy1', ‚nikki1' und ‚urban1' zusammen, um ein repräsentatives Ergebnis zu erhalten.
Die Größe der Fenster um die Gesichtsmerkmale beziehungsweise der abgescannten Bildregionen wurde auf 8x8, 12x12, 16x16 und 20x20 festgelegt und näher untersucht. Dabei gilt wie bereits im Unterkapitel 4.2 erwähnt: Je größer der Kontext eines solchen quadratischen Fensters ist, umso mehr Information über die Bildregion steht zur Verfügung, aber gleichzeitig man hat auch mehr Stör-/Rauschsignale und der Rechenaufwand für die Vorverarbeitung, Merkmalsextraktion und Klassifizierung steigt!

5.2.2 Polynomklassifikator

Beim Polynomklassifikator interessiert einerseits in weit sich der Aufbau des Polynomklassifikators an sich auf die Erkennungsleistung auswirkt und andererseits welche Merkmalsextraktion beziehungsweise welche Variante davon am besten funktioniert.

Die Varianten des Polynomklassifikators werden hauptsächlich definiert durch den Grad (in der Praxis sind aus Effizienzgründen eigentlich nur Grad 1 und Grad 2 sinnvoll, siehe Tabelle 4.2) und durch die Klassifikatorstruktur (hier: ONEvsALL (OA) versus PAIRWISE (PW), siehe Abschnitt 4.5.1.3).

Bei den Experimenten hat sich herausgestellt, dass für die Erkennung der sechs Gesichtsmerkmale die PW-Struktur der OA-Struktur „überlegen" ist und der quadratische Polynomklassifikator im Vergleich zum linearen Polynomklassifikator stabilere und robustere Ergebnisse liefert.

Natürlich interessiert man sich auch für die beste Merkmalsmethode. In Kombination mit dem Polynomklassifikator hat insgesamt gesehen die Projektion beziehungsweise die Summenbildung in x-, y- und Diagonalen-Richtung am besten funktioniert. Danach folgt die Singulärwertzerlegung (SVD), dann die Verwendung der „reinen“ Rohdaten (gegebenenfalls in Kombination mit der PCA) und schließlich die Anwendung der zentrierten stochastischen Momente, die immer noch gute Ergebnisse liefert.
Die normalisierte Variante der Merkmalsmethoden führte durchweg zu weniger Falschklassifikationen.

Innerhalb der einzelnen Methoden zur Merkmalsextraktion gibt es häufig Parametervariationen, die selbstverständlich zu unterschiedlichen Performanzen führen und die auch genauer untersucht wurden. Anhand von Diagrammen, die in MATLAB erzeugt wurden, seien diese Variationen und deren Auswirkungen exemplarisch veranschaulicht.

Bei der Merkmalsextraktion durch die (zentrierten) stochastischen Momente zeigt sich, dass die ersten vier beziehungsweise fünf Momente vollkommen ausreichend sind (Abb. 5.1). Höhere Momente bringen keine Leistungsveränderung.
Wie man in Abbildung 5.1 auch sieht, verhalten sich die einzelnen Evaluierungsaspekte teilweise gegenläufig zu einander: während sowohl die falsch klassifizierten Pixel insgesamt als auch die „False Positives“ bei Hinzunahme höherer Momente stetig fallen, steigt die Anzahl der „False Negatives“ durch das dritte zentrierte Moment. Die Vordergrund-Detektionsrate steigt bei Hinzunahme der Varianz (zweites zentriertes Moment) zum Mittelwert (erstes Moment), fällt aber durch das dritte zentrierte Moment wieder ab, während dieses Moment für die Vertauschungen der Gesichtsmerkmalpixel untereinander und den Korrespondenzfall eine deutliche Verbesserung bringt.

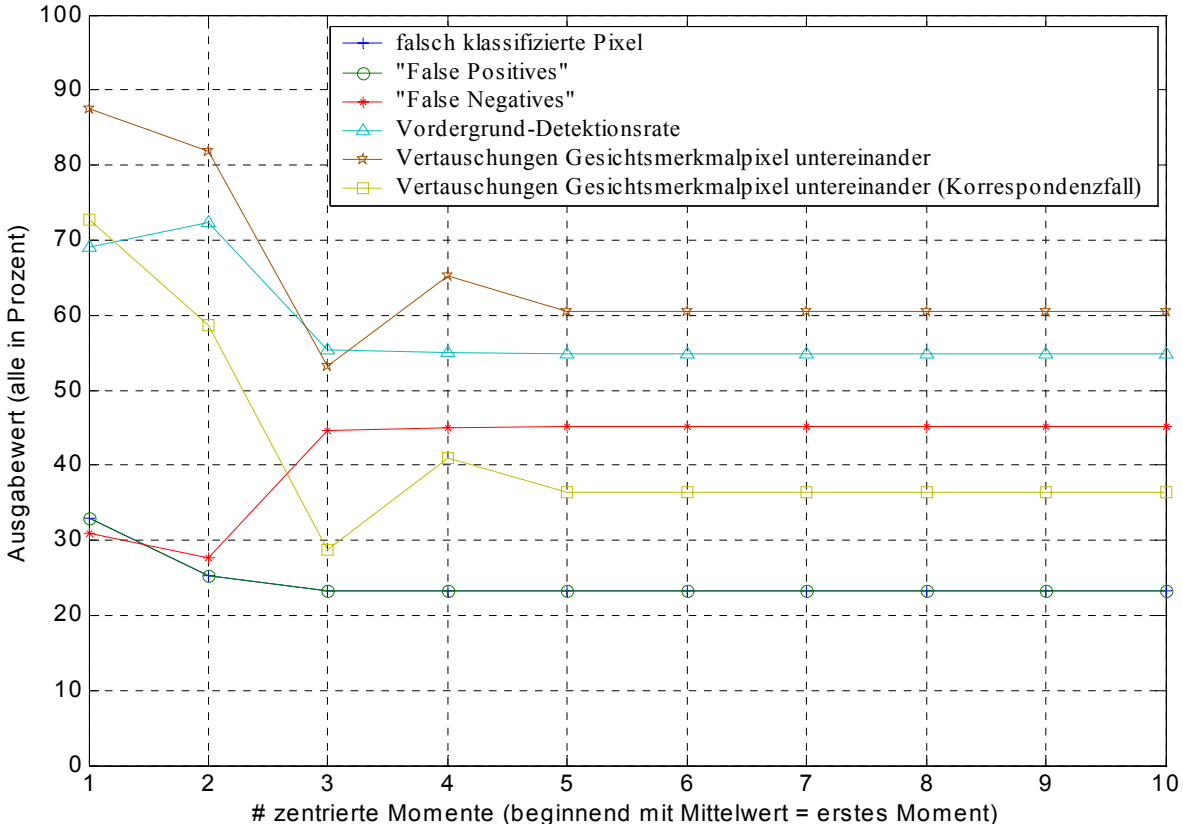


Abbildung 5.1: Quadrat. Polynomklassifikator (PW), Fenster: 12x12, Merkmal: (zentrierte) Momente

Bei der Summenbildungs- beziehungsweise Projektionsmethode schwanken die Ergebnisse noch stärker (Abb. 5.2).
Man ist zwar mit der „vollen“ Projektionsart (Summenbildungsart 15) eigentlich immer auf der sicheren Seite, doch sind bei „teilweiser“ Summenbildung je nach dem welcher Evaluierungsaspekt einem wichtig ist unter Umständen bessere Ergebnisse erzielbar.

Auffällig ist in Abbildung 5.2 vor allem der Übergang von Projektionsart 4 auf 5 (wie auch von 8 auf 9). Projektionsart 4 verwendet nur die Summenbildung in y-Richtung und Projektionsart 5 nur die Summenbildung in x-Richtung. Wie man sieht, fallen bei diesem Übergang sowohl die falsch klassifizierten Pixel insgesamt, die „False Positives“ als auch die „False Negatives“ und die Vordergrund-Detektion steigt wie gewünscht, jedoch das ganze auf Kosten einer höheren Anzahl an Vertauschungen der Gesichtsmerkmalpixel untereinander (auch für den Korrespondenzfall).
Die Projektion in x-Richtung, also die Zeilensumme, bringt somit insgesamt gesehen deutlich mehr als die Projektion in y-Richtung, die Spaltensumme. Im Vergleich zur „vollen“ Summenbildungsart 15 ist bei Summenbildungsart 5 natürlich auch die Merkmalsdimension deutlich geringer (verwendet man beispielsweise ein 20x20 Fenster, braucht man statt einem 42-dimensionalen Vektor nur noch einen 20-dimensionalen Merkmalsvektor bei vergleichbarer Performanz).

Die Hinzunahme der Spurberechungen bringt hier (Abb. 5.2) keine oder nur marginale Verbesserungen.

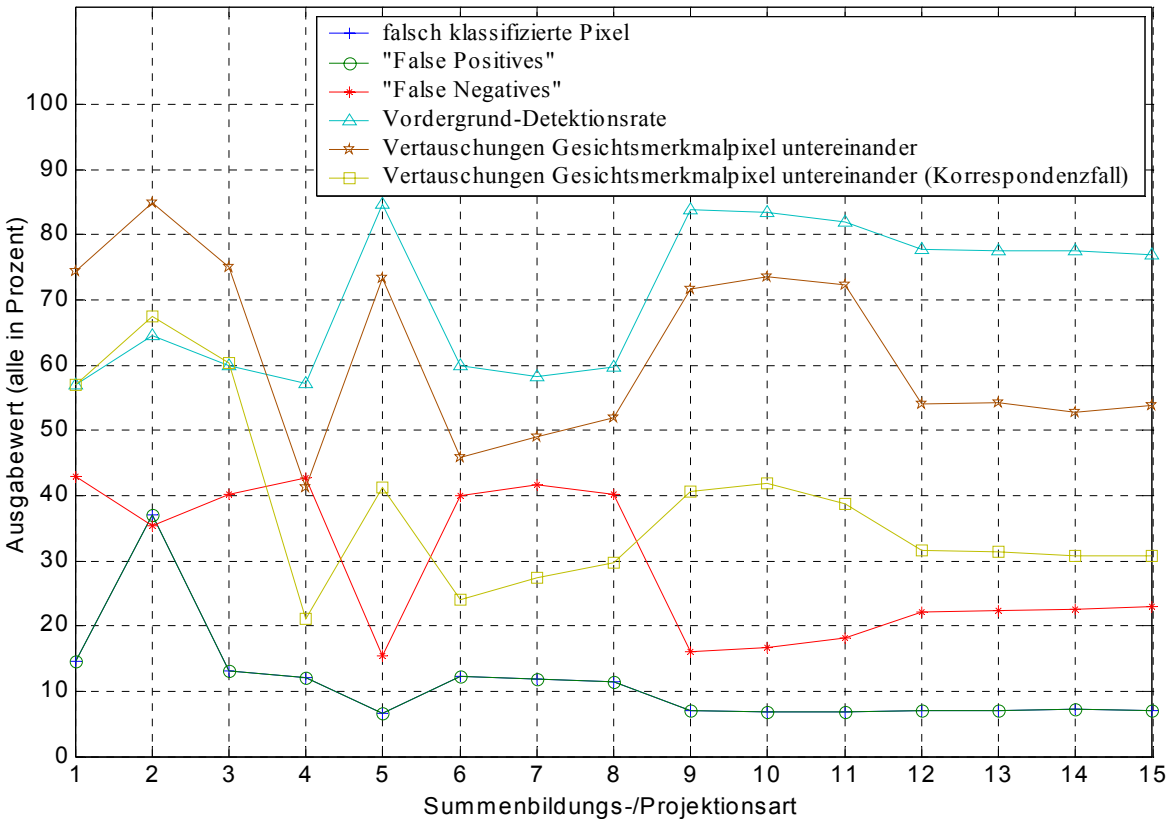


Abbildung 5.2: Quadrat. Polynomklassifikator (PW), Fenster: 12x12 (normalisiert), Merkmal: Summenbildung/Projektion

Zuletzt wollen wir noch auf die Principal Component Analysis (PCA) zur Dimensionsreduktion genauer eingehen. Es werden 5 Fälle definiert, je nach Dimensionsreduktion der Rohdaten auf 5, 10, 15, 20 oder 25 Dimensionen. Betrachtet man die Abbildung 5.3 von rechts nach links, dann fallen die „False Negatives“ kontinuierlich bei einer Reduktion von 25 Dimensionen auf 5 Dimension und die Vordergrund-Detektionsrate steigt ebenfalls stetig für eine PCA auf kleinere Dimensionen. Jedoch steigt dabei die Anzahl der falsch klassifizierten Pixel und der „False Positives“ leicht und beim Übergang von 10 auf 5 Dimensionen dann stärker. Wenn man schließlich noch den Verlauf der Vertauschungen der Gesichtsmerkmalpixel untereinander wie auch des Korrespondenzfalles verfolgt, zeigt sich, dass die PCA-Reduktion auf 10 Dimensionen hier insgesamt am besten funktioniert.

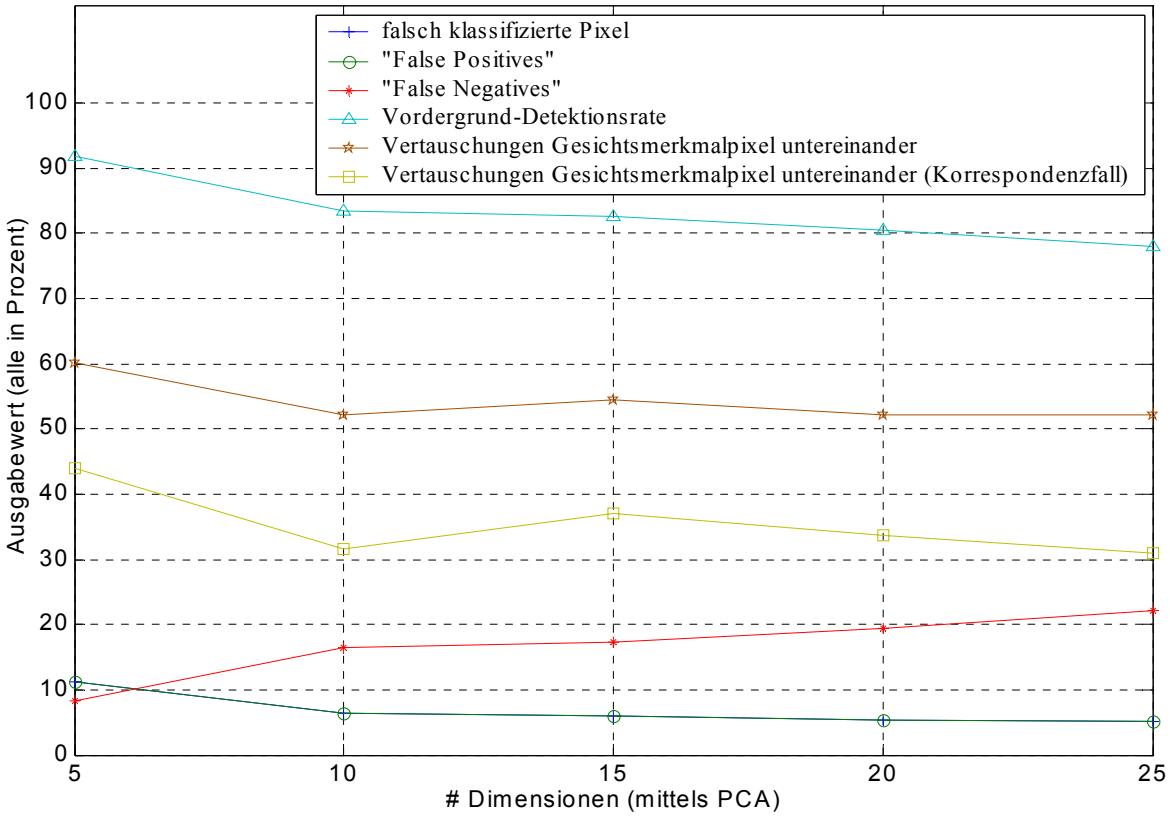


Abbildung 5.3: Quadrat. Polynomklassifikator (PW), Fenster: 16x16, Merkmal: PCA auf Rohdaten

5.2.3 Baumklassifikator

Nachdem wir das Verhalten des Polynomklassifikators zur Erkennung der Gesichtsmerkmale genauer untersucht haben, wollen wir nun auf die Entscheidungsbäume zur Klassifikation der Daten näher eingehen.

Aus Zeitgründen wurden bei der Merkmalsextraktion und der anschließenden Klassifizierung durch die Bäume nicht alle möglichen Parametervariationen der Merkmale wie zuvor beim Polynomklassifikator durchgespielt. Es wurden stattdessen immer die „vollen“ Merkmale verwendet: das heißt, alle verfügbaren Rohdaten (ohne Dimensionsreduktion) der Fenstertechnik, die „volle“ Summenbildung (Summenbildungsart 15) und hier sogar die 40(!) ersten zentrierten stochastischen Momente.
Genau wie bei den Polynomklassifikatoren erhält man bei den Bäumen die gleiche Einteilung der Merkmale nach der Performanz: insgesamt gesehen hat die Projektion beziehungsweise die Summenbildung in x-, y- und Diagonalen-Richtung am besten funktioniert. Danach folgt die Singulärwertzerlegung (SVD), dann die Verwendung der „reinen“ Rohdaten und schließlich die Berechnung der zentrierten stochastischen Momente.
Die normalisierte Variante der Merkmalsmethoden führte analog zum Polynomklassifikator durchweg zu weniger Falschklassifikationen.

Untersucht wurde natürlich auch das Verhalten des „normalen“ und des „geprunten“ Baumes bei der Klassifikation. Dabei hat sich herausgestellt, dass die „geprunte“ Baumversion trotz längeren Trainings eigentlich immer vorzuziehen ist, wie man an den exemplarischen Labelbildern in Abbildung 5.4 auch ganz direkt sehen kann. Durch das „Pruning“ wird das Hintergrundrauschen (wie die störenden Kanten entlang der Fahrzeugscheiben und der Kopfumriss) deutlich reduziert und die Gesichtsmerkmale werden wie gewünscht viel besser identifiziert.

Abbildung 5.4: vorhergesagte Labelbilder bei Klassifikation durch (links) „normalen“ Baum, (rechts) „geprunte“ Baumversion

In Abbildung 5.5 sieht man den ursprünglichen Baum bei Verwendung der Singulärwertzerlegung (SVD) und in Abbildung 5.6 ist die Version dargestellt, die man nach dem optimalen „Pruning“ erhält. Die einzelnen Merkmalskomponenten sind dabei mit svd1, svd2, svd3, ..., svd16 markiert und wie man erkennt, kann eine einzelne Merkmalskomponente in einem Ast aus Gründen des optimalen „Splittings“ durchaus mehrmals vorkommen. Die Klassen sind hier numerisch angegeben: das heißt, die sechs Gesichtsmerkmale werden durch die Zahlen „1“ bis „6“ wiedergegeben und der Hintergrund beziehungsweise „Garbage“ wird durch die Zahl „7“ repräsentiert.
Wie man deutlich erkennt, nimmt die Anzahl der Baumebenen durch das optimale „Pruning“ von 111 auf 49 ab, wodurch die Abbildung auch deutlich lesbarer wird. Trotz des längeren Trainings der „geprunten“ Baumversion hat man durch die verkürzten Äste nun eine viel schnellere Klassenzuweisung, was natürlich in der Praxis willkommen ist.

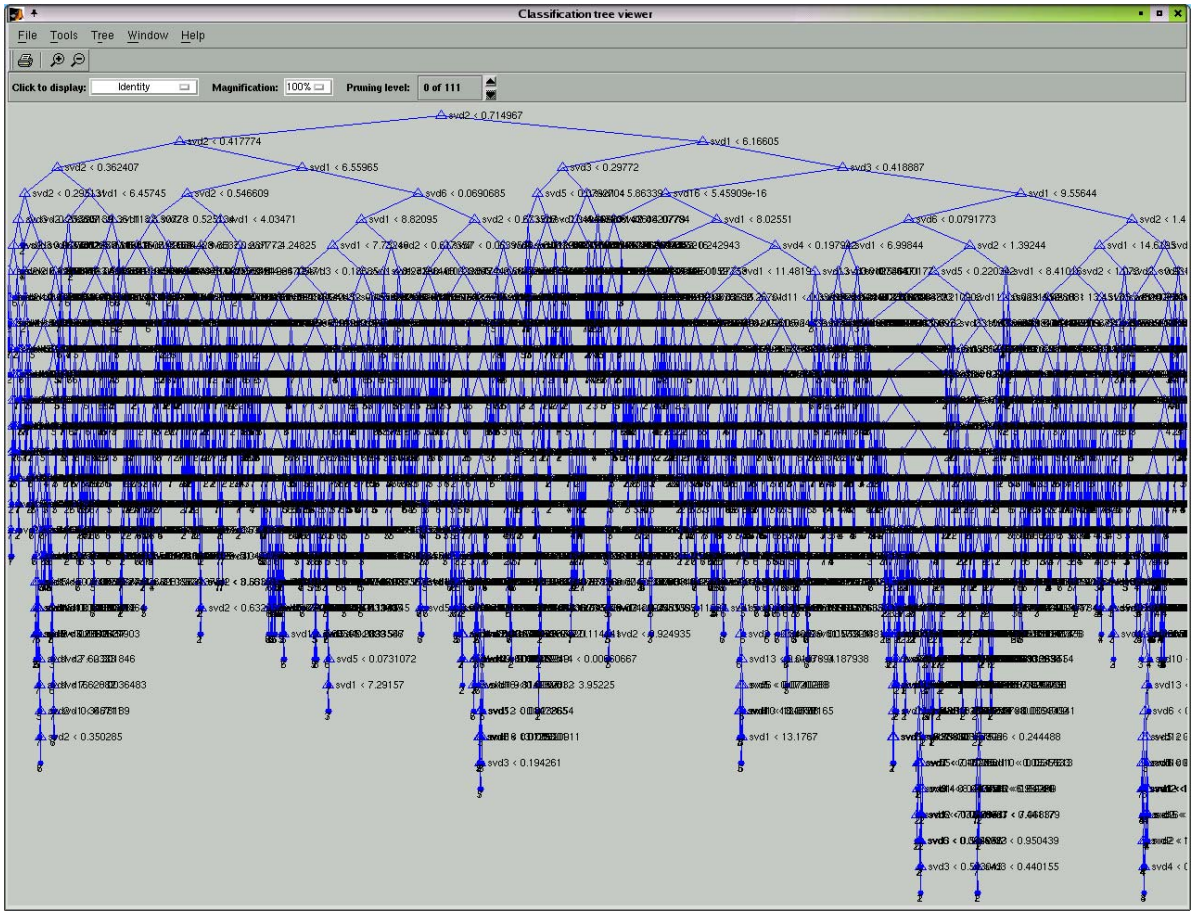


Abbildung 5.5: ursprünglicher Baumklassifikator, Fenster: 16x16, Merkmal: SVD

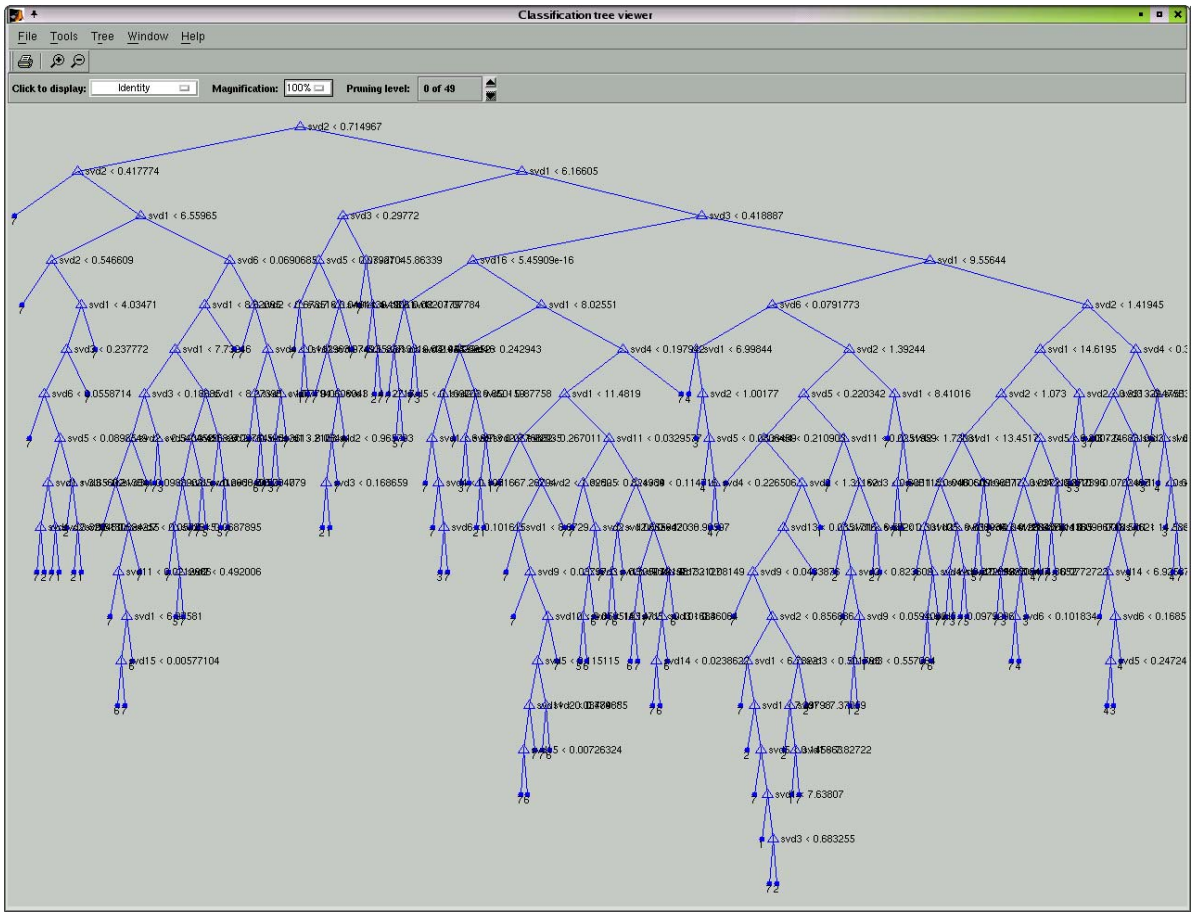


Abbildung 5.6: Baumklassifikator (nach optimalem Pruning), Fenster: 16x16, Merkmal: SVD

Analog zu den Abbildungen 5.5 und 5.6 sei hier noch ein „echter“ erzeugter Baum (Abb. 5.7) und seine „geprunte“ Version (Abb. 5.8) bei Verwendung der reinen Rohdaten dargestellt.
Die einzelnen Merkmalskomponenten sind dabei mit x1, x2, x3, ..., x64 markiert und wie man wieder erkennt, kann eine einzelne Komponente in einem Ast aus Gründen des optimalen „Splittings“ mehrmals vorkommen.
Die Anzahl der Baumebenen nimmt in diesem Fall durch das optimale „Pruning“ von 91 auf 63 ab, wodurch auch die Abbildung wieder übersichtlicher wird.

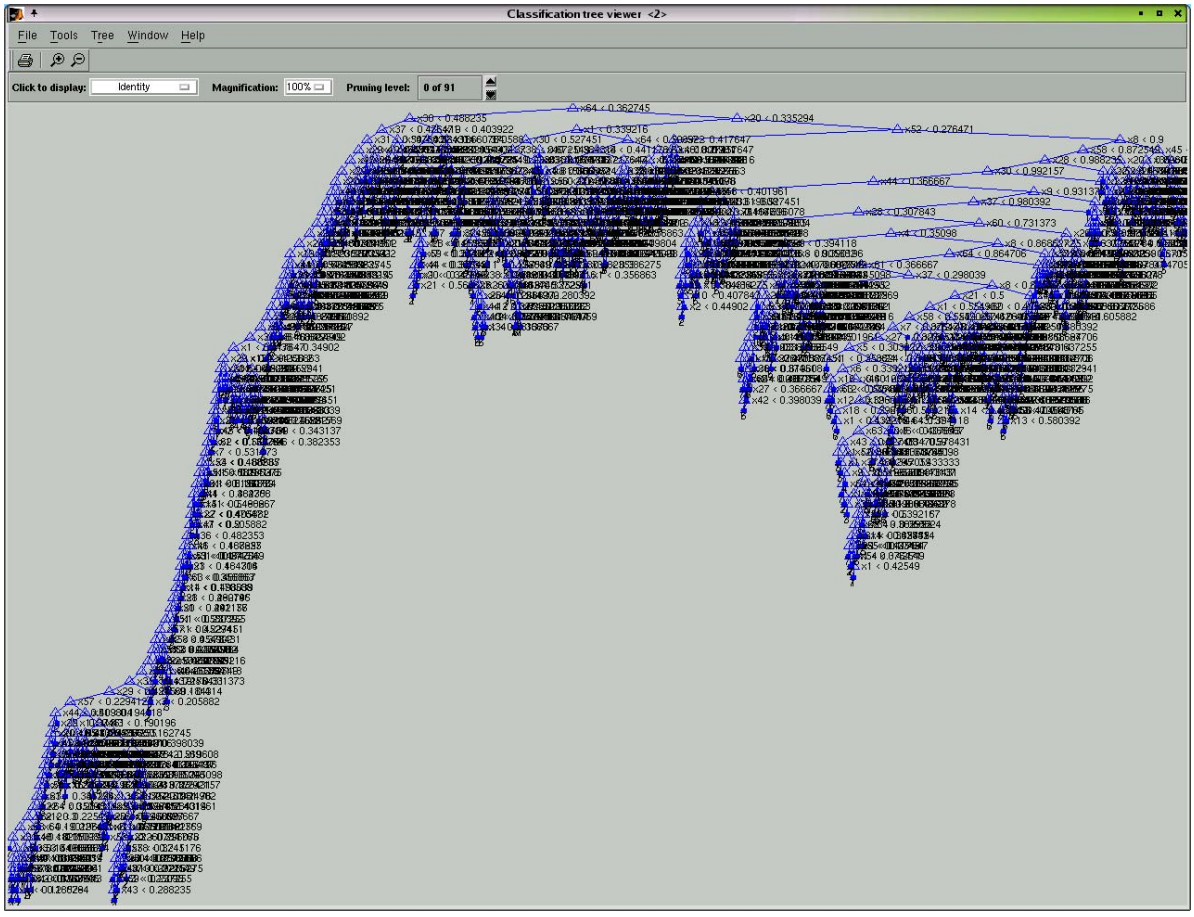


Abbildung 5.7: ursprünglicher Baumklassifikator, Fenster: 8x8, Merkmal: Rohdaten (Dim. 64)

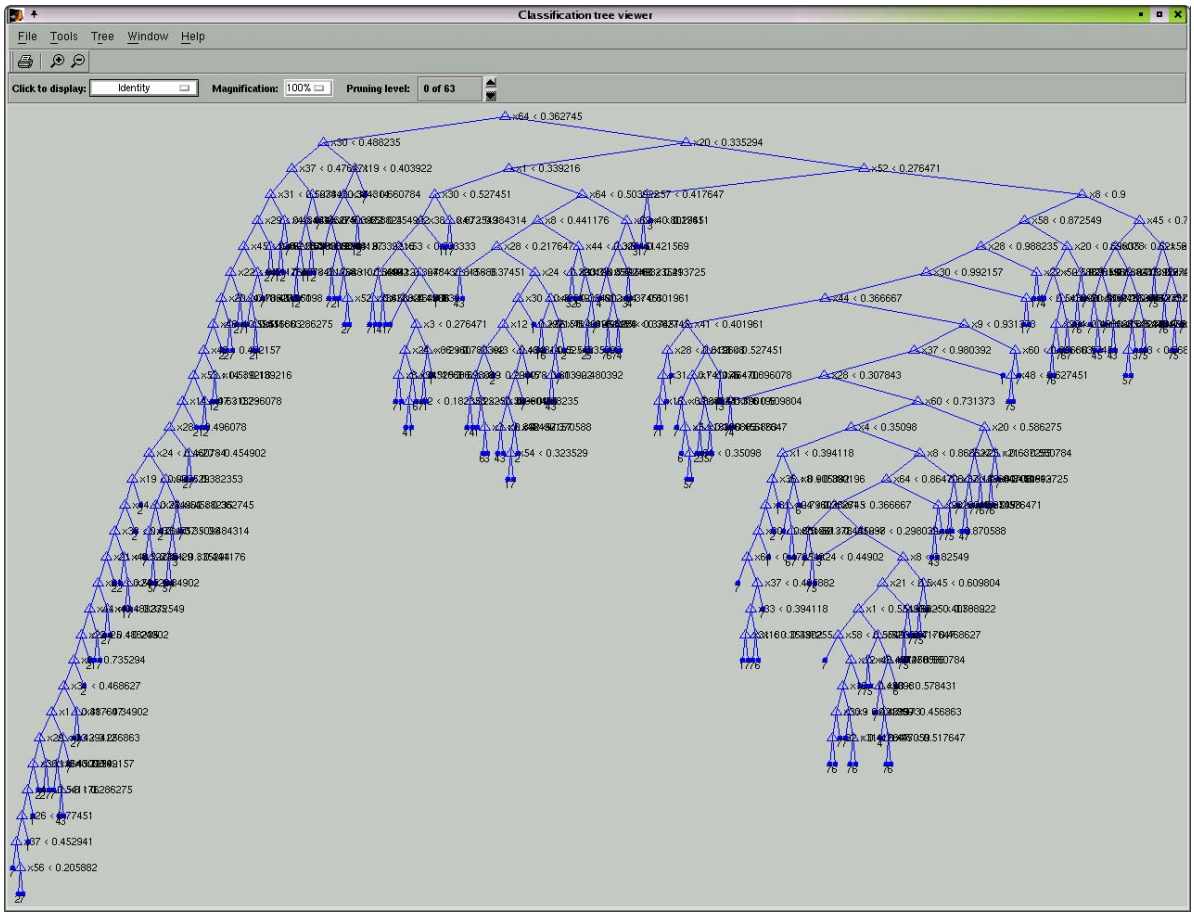


Abbildung 5.8: Baumklassifikator (nach optimalem Pruning), Fenster: 8x8, Merkmal: Rohdaten (Dim. 64)

5.2.4 Vergleich mit „Matched Filter“ – ROC-Diagramme

Im nachfolgenden Diagramm (Abb. 5.9) für den Radius 2 sieht man den typischen Verlauf einer ROC(=Receiver Operator Characteristic)-Kurve (graue Linie). Darin abgebildet sind alle 772 gemachten Experimente und als Referenzpunkt (rot markiertes Pluszeichen) der „Matched Filter“.

Die gemachten Experimente sind je nach Klassifikator unterschiedlich markiert:
- Ansätze mit dem linearen Polynomklassifikator sind als cyan-farbene Kreuze dargestellt
- Ansätze mit dem quadratischen Polynomklassifikator werden durch blaue Sterne wiedergegeben
- und Baumansätze sind als grüne Punkte markiert.

Außerdem sind jeweils zwei Ansätze mit dem linearen Polynomklassifikator, mit dem quadratischen Polynomklassifikator und mit dem Entscheidungsbaum hervorgehoben, die besonders gute beziehungsweise interessante Ergebnisse lieferten.

Gewünscht sind Punkte in der linken oberen Ecke des Diagramms: das heißt, diese Ansätze haben dann sowohl eine niedrige Anzahl an „False Positives“ als auch eine hohe Anzahl an korrekt detektierten Vordergrundpixel (sogenannte *„True Positives“*). Das wäre dann hier der (durch einen blauen Pfeil markierte) Ansatz mit dem quadratischen Polynomklassifikator (PW) in Kombination mit der PCA-Dimensionsreduktion der Rohdaten auf 10 Dimensionen bei einer Fenstergröße von 16x16.

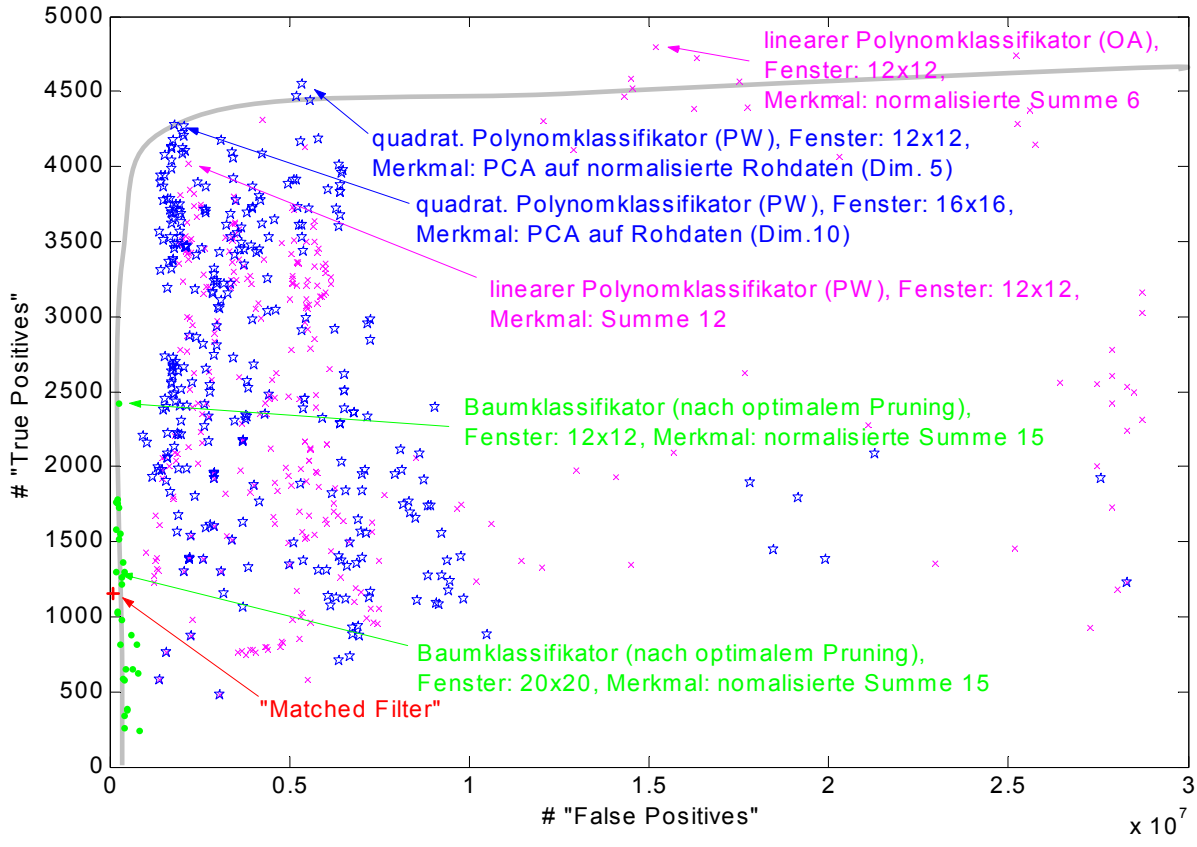


Abbildung 5.9: ROC-Diagramm für Radius 2, mit markierten interessanten Experimenten

Wie man sieht, produzieren die Baum-Ansätze relativ wenige „False Positives“ und detektieren die (wahren) Vordergrundpixel (im Vergleich zum „Matched Filter) je nach verwendeter Merkmalsmethode schon recht gut. Hier hat die Kombination geprunter Baum mit Merkmal „volle Summenbildung“ (Summenart 15) der normalisierten Rohdaten bei Fenstergröße 12x12 am besten funktioniert.
Bei den Polynomklassifikatoren sieht es im Prinzip gerade umgekehrt aus: sie detektieren sehr viele „True Positives“ (deutlich mehr als der „Matched Filter“ und die Baum-Ansätze), produzieren aber oft auch (gleichzeitig) sehr viele „False Positives“.

Sicherlich fällt dem Leser die hohe Skalierung (Faktor 10^7) der x-Achse des Diagramms auf: Die implementierten Ansätze mit dem Polynomklassifikator machen wie eben beschrieben (im Vergleich zum „Matched Filter“ und den Baumansätzen) sehr viele „False Positives“. Es werden also Hintergrundpixel fälschlicherweise als Vordergrund detektiert. Dies ist ein Problem des vollständigen Hintergrundlernens.
Beim implementierten Training werden wie bereits erwähnt „Garbage“-Beispiele immer nur „gewürfelt“ und man kann mit 128 erzeugten Samples pro Frame bei der verwendeten Auflösung von 640x480 Pixel natürlich nicht den kompletten aktuellen Hintergrund lernen, geschweige denn alle möglichen Hintergrund-Samples, die irgendwann mal auftreten könnten, abdecken.
Nichtsdestotrotz ist diese Art von Fehler nicht so gravierend wie fehlende Vordergrundpixel. Werden nur wenige Vordergrundpixel erkannt ist die Information (komplett) weg, während die „False Positives“ durch eine geeignete Nachbearbeitung der gefundenen „Vordergrund-Blobs“ nochmals deutlich gesenkt werden können.

In Abbildung 5.10 sieht man für die Testperson ‚urban1’ die Ergebnisbilder des „Matched Filters“, des (in Bezug auf das ROC-Diagramm, Abb. 5.9) besten Polynomklassifikator-Ansatzes und der besten „Baummethode“.

Abbildung 5.10: vorhergesagte Labelbilder der Testperson ‚urban1’ beim (links) „Matched Filter“, (Mitte) quadrat. Polynomklassifikator, Fenster: 16x16, Merkmal: PCA auf Rohdaten (10), (rechts) geprunten Baumklassifikator, Fenster: 12x12, Merkmal: normalisierte Summe 15

Bisher wurde nur Radius 2 näher betrachtet: bei den Radien 3 und 4 (Abb. 5.11) sieht es ähnlich aus, wobei die Unterschiede im Hinblick auf die „True Positives“ zwischen dem „Matched Filter“ und den in dieser Diplomarbeit implementierten Ansätzen nochmals deutlich zunimmt.

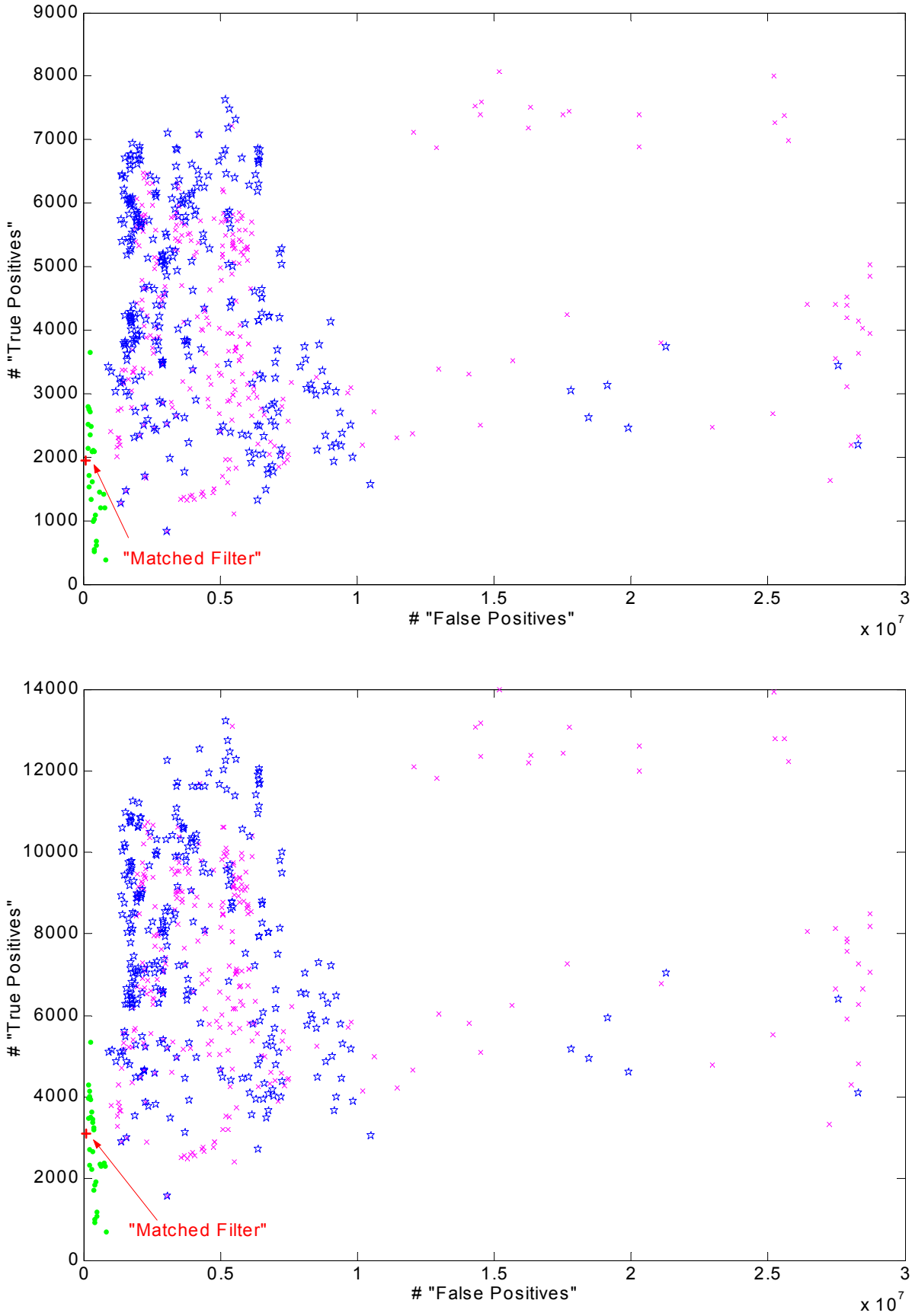


Abbildung 5.11: ROC-Diagramme für Radius 3 (oben) und Radius 4 (unten)

Kapitel 6

Zusammenfassung und Ausblick

In dieser Diplomarbeit wurde die Initialisierung eines bestehenden Tracking-Systems für Gesichtsmerkmale eines PKW-Fahrers mit Hilfe von adaptiven Verfahren verbessert.
Das zugrundeliegende Tracking-Verfahren ist featurebasiert: die Messungen der Gesichtsfeature-Positionen im Bild werden mit Hilfe eines Kalmanfilters über die Bildsequenz integriert. Zwar gab es im bereits bestehenden Tracking-System schon eine Initialisierung der Gesichtsmerkmale durch einen sehr einfachen „Matched-Filter"-Ansatz, dieser ist aber für realistische Anforderungen nicht gut genug. Wünschenswert wäre ein Erkennungssystem, das auch bei den sehr schwierigen Randbedingungen speziell im Fahrzeugumfeld (wie Schlagschatten und Reflexionen), Gesichtsmerkmale im aktuellen Eingabeframe robust findet.

Die in dieser Arbeit implementieren Verfahren werden mittels eines Trainings angepasst und erbringen dann selbständig auf unbekannten Testbildern sofort die gewünschte Erkennungsleistung bei vertretbarem Rechenaufwand.
Im Gegensatz zur ursprünglichen „Matched Filter"-Methode, die nur eine Einteilung in Vorder- und Hintergrund-Klasse liefert, ermöglichen die hier implementierten Algorithmen eine genaue Zuweisung der Bildregionen zu den folgenden sieben Klassen:

- rechtes Auge
- linkes Auge
- rechtes Nasenloch
- linkes Nasenloch
- rechter Mundwinkel
- linker Mundwinkel
- und Hintergrund

Für die Klassifikation wurden sowohl Polynomklassifikatoren als auch Entscheidungsbäume zur besseren Vergleichbarkeit eingesetzt.
Bei den durchgeführten 772 Experimenten wurden einerseits die Art des Klassifikators beziehungsweise dessen Strukturen und andererseits auch die verschiedenen Methoden zur Merkmalsextraktion genauer untersucht.

Dabei hat sich beim Polynomklassifikator gezeigt, dass die sogenannte PAIRWISE-Struktur der sogenannten ONEvsALL-Struktur in diesem Kontext „überlegen" ist. Der quadratische Polynomklassifikator lieferte im Vergleich zum linearen Polynomklassifikator bessere beziehungsweise robustere Ergebnisse.

Bei den Entscheidungsbäumen ist die „geprunte" Baumversion eigentlich immer vorzuziehen.

Außerdem hat sich herausgestellt, dass bei allen Klassifikatoren die Projektions-/Summenbildungsmethode am besten funktioniert hat, gefolgt von der Singulärwertzerlegung (SVD), dann die Verwendung der reinen Rohdaten und schließlich die Berechnung der zentrierten stochastischen Momente.

Eine Normalisierung/Vorverarbeitung der Rohdaten hat dabei durchweg zu weniger Falschklassifikationen geführt und sollte deshalb trotz erhöhten Berechnungsaufwandes durchgeführt werden.

Bei den Parametervariationen innerhalb der einzelnen Methoden zur Merkmalsgewinnung hat sich folgendes ergeben: die ersten vier beziehungsweise fünf zentrierten Momente sind vollkommen ausreichend; höhere Momente bringen keine Leistungsveränderung. Die „volle“ Summenbildungsart ist oft am besten, trotzdem erhält man auch schon bei „teilweiser“ Summenbildung (vor allem bei der Projektion in x-Richtung) vergleichbar gute Ergebnisse.

Bei der Dimensionsreduktion durch die PCA erzielte man bei einer Reduktion auf 10 Dimension oft die beste Erkennungsleistung.

In Bezug auf die „False Positives“ alleine und die „False Positives“ pro „True Positive“ sind unter den 30 beziehungsweise 20 besten hier implementierten Ansätzen nur Baummethoden vertreten; vor allem in Kombination mit der „vollen“ Summenbildung der normalisierten Pixelwerte. Dabei gab es eine Tendenz zu größeren Fenstern (16x16 und 20x20). Die besten Baumergebnisse lieferten eine „False Positives“-Rate von 0,57% bei einer „True Positives“-Rate von 37,89%. Die einzelnen Gesichtsmerkmale (Auge, Nasenloch, Mundwinkel) wurden dabei mit einer Rate von 66,86% (Korrespondenzfall) richtig klassifiziert. Die „False Negatives“-Rate betrug jedoch 62.11%.

Betrachtet man nun die besten „True Positives“-Ergebnisse und „False Negatives“-Ergebnisse, dann sieht es gerade umgekehrt aus: unter den 200 besten untersuchten Ansätzen finden sich nur welche, die den Polynomklassifikator verwenden (vor allem in Kombination mit der Summenbildung der normalisierten Rohdaten und bei Verwendung der reinen Rohdaten (mit der PCA zur Dimensionsreduktion)). Die Fenstergröße war dabei bis auf zu kleine Fenster (8x8) relativ egal. Der beste Polynomklassifikator-Ansatz lieferte eine „True Positives“-Rate von 78,12% (linear) beziehungsweise 84,12% (quadratisch) bei einer „False Negatives“-Rate von 22,13% beziehungsweise 15,66% und einer „False Positives“-Rate von 8,17% beziehungsweise 8,70%. Hier wurden die einzelnen Gesichtsmerkmale mit einer Rate von 60,8% beziehungsweise 59% (Korrespondenzfall) korrekt erkannt.

Je nachdem, was dem Anwender der implementierten Methoden wichtig ist, sind also bei ähnlicher Klassifikationsgeschwindigkeit einerseits die Entscheidungsbäume oder die Polynomklassifikatoren vorzuziehen, wobei die Trainingszeit für die Bäume (inklusive „Pruning“) um einen Faktor 8 länger geht.

Im direkten Vergleich mit dem ursprünglichen „Matched Filter“-Ansatz konnten mit den neu entwickelten Verfahren deutliche Verbesserungen erzielt werden: zum einen erhält man nun eine drastisch höhere „True Positives“-Rate (bis zu Faktor 5), das heißt der Vordergrund wird viel besser erkannt (bei einem Auswerteradius von 4 wurden beispielsweise bei der „Matched Filter“-Methode nur 3111 Vordergrundpixel richtig erkannt, während die neuen Ansätze bis zu 13997 Pixel richtig als Vordergrund zugeordnet haben). Zum anderen können wie bereits zu Beginn dieser Zusammenfassung erwähnt alle Bildregionen mit den neuen Ansätzen den einzelnen Gesichtsmerkmalen und dem Hintergrund direkt zugeordnet werden.

Um noch bessere und robustere Erkennungsergebnisse zu erhalten, wären in der Zukunft die folgenden zwei Punkte auf jeden Fall wünschenswert.
Zum einen sicherlich eine größere und bessere Trainings- und Testmenge mit vielen verschiedenen Fahrern und zum anderen eine Nachbearbeitung der erhaltenen „Blobregionen“ im vom Klassifikator vorhergesagten Labelbild. Hier könnte man beispielsweise durch sogenannte „morphologische Operatoren“ (wie Dilatation und Erosion, Opening und Closing), momentenbasierte Formmerkmale, Fourierdeskriptoren, Objektsymmetrien, Formparameter (wie Fläche, Umfang, Rundheit, umgebendes Rechteck (bounding box)) und Nachbarschaftsrelationen (wie 4er-/8er-Zusammenhangskomponenten) Verbesserungen erzielen. Denkbar wäre natürlich auch, noch mehr a priori Wissen über das Gesicht beziehungsweise die Anordnung der Gesichtsmerkmale in das Erkennungssystem beziehungsweise in das bestehende Kopfmodell einzubauen.

Anhang A

Datenbank: Beispielframes/-sequenzen

Für das Trainieren und Testen des Erkennungssystems wurde eine SQL-Datenbank (Postgresql) aufgebaut: zur Zeit enthält sie 10 verschiedene Personen mit insgesamt zirka 60000 Frames; es gibt sowohl Tag- als auch Nachtaufnahmen, wie man in der Abbildung A.1 gut erkennt.
Es handelt sich bei den verwendeten Videosequenzen um reale Fahrtaufnahmen, somit wird die Evaluation im „richtigen“, das heißt praxisbezogenen Szenarium gemacht.

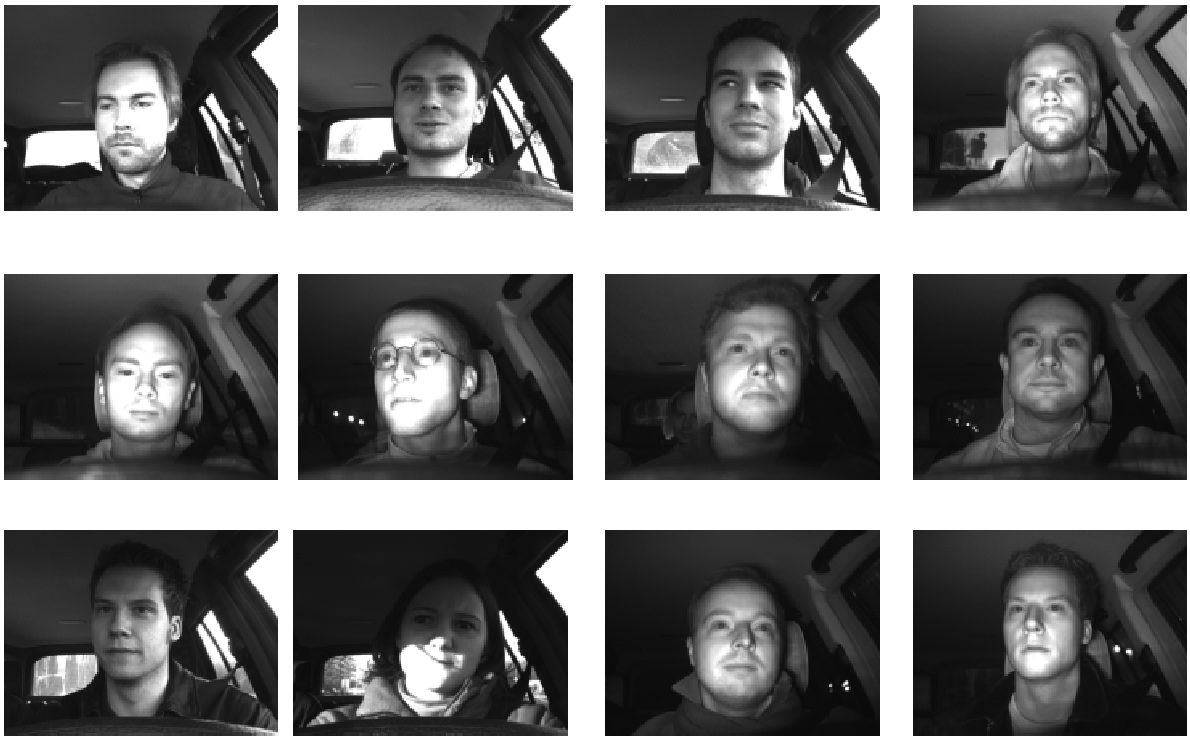


Abbildung A.1: Beispielbilder zum Trainieren und Testen des Erkennungssystems

Anhang B

Aufbau der Labeldatei

In der Labeldatei wird für jeden Frame einer Sequenz die Position der sechs Gesichtsmerkmale (rechtes/linkes Auge, rechtes/linkes Nasenloch und rechter/linker Mundwinkel) in x- und y-Koordinaten angegeben. Sie ist nach folgendem Schema aufgebaut:

filename|sequence_id|image_id|num|erx|ery|elx|ely|nrx|nry|nlx|nly|mrx|mry|mlx|mly
gaze0000433470/gaze0000433471.pgm|433470|433471|1|310|179|400|194|339|233|366|240|303|278|384|297
gaze0000433470/gaze0000433472.pgm|433470|433472|2|309|177|401|193|339|233|366|240|303|277|384|296
gaze0000433470/gaze0000433473.pgm|433470|433473|3|309|176|401|191|339|232|366|239|303|277|385|296
gaze0000433470/gaze0000433474.pgm|433470|433474|4|309|176|401|191|339|232|366|239|303|276|385|295
gaze0000433470/gaze0000433475.pgm|433470|433475|5|309|175|401|190|339|232|366|239|303|276|385|295
gaze0000433470/gaze0000433476.pgm|433470|433476|6|309|175|401|190|339|232|366|239|303|276|385|294
gaze0000433470/gaze0000433477.pgm|433470|433477|7|309|175|401|190|339|232|366|238|303|275|385|294
gaze0000433470/gaze0000433478.pgm|433470|433478|8|309|178|400|193|339|231|366|238|303|275|386|294
gaze0000433470/gaze0000433479.pgm|433470|433479|9|308|180|398|194|338|231|365|238|302|275|385|293
gaze0000433470/gaze0000433480.pgm|433470|433480|10|308|181|397|194|338|231|365|238|302|275|385|293
gaze0000433470/gaze0000433481.pgm|433470|433481|11|307|180|398|194|337|231|365|238|302|275|386|293
gaze0000433470/gaze0000433482.pgm|433470|433482|12|307|179|398|192|336|231|364|238|301|275|386|293
gaze0000433470/gaze0000433483.pgm|433470|433483|13|305|177|397|191|336|231|363|237|300|275|385|292
gaze0000433470/gaze0000433484.pgm|433470|433484|14|304|176|397|188|335|231|362|237|299|275|385|292
gaze0000433470/gaze0000433485.pgm|433470|433485|15|303|175|396|188|334|231|362|237|299|276|384|291
gaze0000433470/gaze0000433486.pgm|433470|433486|16|302|175|395|187|333|231|361|237|298|276|384|291
gaze0000433470/gaze0000433487.pgm|433470|433487|17|301|175|393|187|332|230|360|236|298|276|384|291
gaze0000433470/gaze0000433488.pgm|433470|433488|18|300|175|392|187|331|230|359|236|297|276|383|290
gaze0000433470/gaze0000433489.pgm|433470|433489|19|299|175|391|186|330|230|359|236|297|276|383|290
gaze0000433470/gaze0000433490.pgm|433470|433490|20|298|176|390|186|330|231|358|236|297|276|383|290
gaze0000433470/gaze0000433491.pgm|433470|433491|21|297|178|389|186|329|231|357|236|296|277|381|289
gaze0000433470/gaze0000433492.pgm|433470|433492|22|296|182|387|190|328|231|356|236|296|277|381|289
gaze0000433470/gaze0000433493.pgm|433470|433493|23|296|183|385|190|327|232|355|236|295|278|381|289
gaze0000433470/gaze0000433494.pgm|433470|433494|24|291|183|385|189|326|232|355|237|295|279|381|290
gaze0000433470/gaze0000433495.pgm|433470|433495|25|292|181|384|188|325|233|353|237|294|280|381|290
gaze0000433470/gaze0000433496.pgm|433470|433496|26|290|181|382|186|324|234|353|238|293|281|380|290
gaze0000433470/gaze0000433497.pgm|433470|433497|27|288|183|381|186|322|235|351|238|293|282|379|290
gaze0000433470/gaze0000433498.pgm|433470|433498|28|286|184|379|185|321|236|350|239|292|283|379|291
gaze0000433470/gaze0000433499.pgm|433470|433499|29|285|184|377|185|320|236|349|239|291|283|378|291
gaze0000433470/gaze0000433500.pgm|433470|433500|30|283|185|376|185|318|237|348|240|290|284|377|291
gaze0000433470/gaze0000433501.pgm|433470|433501|31|281|186|374|185|317|238|346|240|289|285|375|291
gaze0000433470/gaze0000433502.pgm|433470|433502|32|279|187|373|184|315|238|345|241|288|285|374|291
gaze0000433470/gaze0000433503.pgm|433470|433503|33|278|187|371|185|313|239|343|241|287|286|374|292
gaze0000433470/gaze0000433504.pgm|433470|433504|34|276|188|369|185|311|239|342|242|285|287|372|292
gaze0000433470/gaze0000433505.pgm|433470|433505|35|275|189|368|185|310|240|340|242|284|287|371|292
gaze0000433470/gaze0000433506.pgm|433470|433506|36|274|189|367|185|308|241|339|243|283|288|370|293
gaze0000433470/gaze0000433507.pgm|433470|433507|37|271|191|364|186|306|242|336|244|281|290|369|294
gaze0000433470/gaze0000433508.pgm|433470|433508|38|271|191|364|186|305|243|336|245|280|290|367|294
gaze0000433470/gaze0000433509.pgm|433470|433509|39|270|192|363|188|304|244|335|245|280|291|367|295
gaze0000433470/gaze0000433510.pgm|433470|433510|40|269|192|362|188|303|244|335|246|280|291|367|295
gaze0000433470/gaze0000433511.pgm|433470|433511|41|269|193|362|188|303|244|334|246|279|292|366|295
gaze0000433470/gaze0000433512.pgm|433470|433512|42|268|193|361|188|302|244|332|245|279|292|365|295
gaze0000433470/gaze0000433513.pgm|433470|433513|43|266|192|359|188|301|244|332|245|278|291|365|295
gaze0000433470/gaze0000433514.pgm|433470|433514|44|264|192|355|187|299|243|330|244|277|291|363|294

gaze0000433470/gaze0000433515.pgm|433470|433515|45|262|191|352|186|297|242|328|243|275|290|361|293
gaze0000433470/gaze0000433516.pgm|433470|433516|46|260|190|350|185|294|241|325|242|272|289|358|292
gaze0000433470/gaze0000433517.pgm|433470|433517|47|258|190|347|184|290|241|321|241|270|289|355|291
gaze0000433470/gaze0000433518.pgm|433470|433518|48|256|190|345|183|286|240|316|240|267|289|350|289
gaze0000433470/gaze0000433519.pgm|433470|433519|49|253|190|342|183|282|240|313|240|264|289|347|289
gaze0000433470/gaze0000433520.pgm|433470|433520|50|251|191|339|183|278|240|309|240|261|289|343|288
gaze0000433470/gaze0000433521.pgm|433470|433521|51|249|191|336|182|274|240|305|240|258|289|339|288
gaze0000433470/gaze0000433522.pgm|433470|433522|52|246|192|333|182|271|241|301|240|256|289|336|288
gaze0000433470/gaze0000433523.pgm|433470|433523|53|244|192|330|182|267|241|297|240|253|289|333|288
gaze0000433470/gaze0000433524.pgm|433470|433524|54|241|193|327|181|263|241|293|240|250|290|330|288
gaze0000433470/gaze0000433525.pgm|433470|433525|55|239|193|324|181|260|241|289|239|247|290|326|287
gaze0000433470/gaze0000433526.pgm|433470|433526|56|237|193|321|180|256|241|286|239|245|290|323|287
gaze0000433470/gaze0000433527.pgm|433470|433527|57|234|194|318|180|253|241|282|239|242|290|320|286
gaze0000433470/gaze0000433528.pgm|433470|433528|58|232|195|315|180|250|241|279|239|240|290|317|286
gaze0000433470/gaze0000433529.pgm|433470|433529|59|230|195|312|179|247|241|276|239|1|-1|315|286
gaze0000433470/gaze0000433530.pgm|433470|433530|60|228|195|310|179|244|241|273|238|1|-1|312|285
gaze0000433470/gaze0000433531.pgm|433470|433531|61|226|196|307|179|242|241|271|239|1|-1|310|285
gaze0000433470/gaze0000433532.pgm|433470|433532|62|225|196|305|179|240|241|269|239|1|-1|308|285
gaze0000433470/gaze0000433533.pgm|433470|433533|63|223|197|303|179|238|241|266|238|1|-1|307|285
gaze0000433470/gaze0000433534.pgm|433470|433534|64|222|197|302|179|237|241|265|238|1|-1|305|285
gaze0000433470/gaze0000433535.pgm|433470|433535|65|221|197|301|179|236|242|263|238|1|-1|304|285
gaze0000433470/gaze0000433536.pgm|433470|433536|66|220|198|300|179|235|242|262|238|1|-1|303|285
gaze0000433470/gaze0000433537.pgm|433470|433537|67|220|198|299|179|234|242|262|238|1|-1|303|285
gaze0000433470/gaze0000433538.pgm|433470|433538|68|219|198|299|179|234|242|262|239|1|-1|302|285
gaze0000433470/gaze0000433539.pgm|433470|433539|69|221|198|300|179|234|243|262|239|1|-1|302|285
gaze0000433470/gaze0000433540.pgm|433470|433540|70|222|198|305|180|234|243|261|239|1|-1|302|285
gaze0000433470/gaze0000433541.pgm|433470|433541|71|224|199|306|180|234|243|262|239|1|-1|303|286
gaze0000433470/gaze0000433542.pgm|433470|433542|72|224|199|306|180|235|243|263|240|1|-1|303|286
gaze0000433470/gaze0000433543.pgm|433470|433543|73|225|199|307|180|235|243|264|240|1|-1|304|286
gaze0000433470/gaze0000433544.pgm|433470|433544|74|225|199|307|180|236|243|264|240|1|-1|305|286
gaze0000433470/gaze0000433545.pgm|433470|433545|75|225|199|308|181|237|244|265|240|1|-1|306|287
gaze0000433470/gaze0000433546.pgm|433470|433546|76|227|199|308|182|238|244|267|241|1|-1|306|287
gaze0000433470/gaze0000433547.pgm|433470|433547|77|228|201|310|185|238|243|267|241|1|-1|307|287
gaze0000433470/gaze0000433548.pgm|433470|433548|78|229|201|312|185|239|244|268|241|1|-1|308|287
gaze0000433470/gaze0000433549.pgm|433470|433549|79|230|199|313|184|240|244|269|241|1|-1|308|287
gaze0000433470/gaze0000433550.pgm|433470|433550|80|230|198|311|183|241|244|270|241|1|-1|309|287
gaze0000433470/gaze0000433551.pgm|433470|433551|81|229|198|312|182|242|243|271|241|236|295|310|287
gaze0000433470/gaze0000433552.pgm|433470|433552|82|230|197|313|181|243|243|273|241|237|295|311|287
gaze0000433470/gaze0000433553.pgm|433470|433553|83|230|197|314|181|244|243|273|240|238|294|312|287
gaze0000433470/gaze0000433554.pgm|433470|433554|84|231|197|315|182|245|242|274|240|238|294|313|287
gaze0000433470/gaze0000433555.pgm|433470|433555|85|231|197|316|181|246|242|275|240|239|294|314|287
gaze0000433470/gaze0000433556.pgm|433470|433556|86|232|196|317|181|247|242|276|239|240|294|315|286
gaze0000433470/gaze0000433557.pgm|433470|433557|87|233|196|318|181|248|242|278|240|241|294|316|286
gaze0000433470/gaze0000433558.pgm|433470|433558|88|233|197|318|181|249|242|279|240|241|294|317|286
gaze0000433470/gaze0000433559.pgm|433470|433559|89|233|199|321|184|250|242|280|240|242|294|318|287
gaze0000433470/gaze0000433560.pgm|433470|433560|90|234|200|321|186|251|242|281|240|243|294|319|287
gaze0000433470/gaze0000433561.pgm|433470|433561|91|234|200|324|185|252|242|281|240|245|294|320|287
gaze0000433470/gaze0000433562.pgm|433470|433562|92|236|198|322|184|253|242|283|240|246|294|322|287
gaze0000433470/gaze0000433563.pgm|433470|433563|93|236|198|322|182|254|242|284|240|247|294|323|287
gaze0000433470/gaze0000433564.pgm|433470|433564|94|236|197|322|182|255|242|285|240|248|294|324|287
gaze0000433470/gaze0000433565.pgm|433470|433565|95|237|197|323|181|256|242|285|240|249|294|324|287
gaze0000433470/gaze0000433566.pgm|433470|433566|96|237|197|324|182|257|242|286|240|249|294|325|287
gaze0000433470/gaze0000433567.pgm|433470|433567|97|236|197|324|182|257|242|287|240|250|294|326|287
gaze0000433470/gaze0000433568.pgm|433470|433568|98|236|197|324|182|257|242|287|240|250|294|326|287
gaze0000433470/gaze0000433569.pgm|433470|433569|99|236|197|324|182|258|242|288|240|250|295|327|287
gaze0000433470/gaze0000433570.pgm|433470|433570|100|237|197|325|181|258|242|288|240|251|294|327|287
gaze0000433470/gaze0000433571.pgm|433470|433571|101|237|197|325|181|259|243|288|240|251|295|327|287
gaze0000433470/gaze0000433572.pgm|433470|433572|102|237|197|325|181|259|243|289|241|251|295|327|287
gaze0000433470/gaze0000433573.pgm|433470|433573|103|237|198|325|182|260|243|290|241|251|295|328|287

gaze0000433470/gaze0000433574.pgm|433470|433574|104|239|198|325|183|260|243|290|241|252|295|328|287
gaze0000433470/gaze0000433575.pgm|433470|433575|105|238|201|329|186|260|243|290|241|252|295|329|288
gaze0000433470/gaze0000433576.pgm|433470|433576|106|239|201|329|187|261|244|291|242|253|295|329|288
gaze0000433470/gaze0000433577.pgm|433470|433577|107|239|200|328|186|262|244|292|242|253|296|330|288
gaze0000433470/gaze0000433578.pgm|433470|433578|108|241|198|326|184|262|244|292|242|254|296|331|289
gaze0000433470/gaze0000433579.pgm|433470|433579|109|240|198|327|184|263|244|292|242|254|296|331|289
gaze0000433470/gaze0000433580.pgm|433470|433580|110|241|198|328|183|263|244|294|243|255|296|332|289
gaze0000433470/gaze0000433581.pgm|433470|433581|111|241|198|329|184|264|244|295|243|255|296|332|289
gaze0000433470/gaze0000433582.pgm|433470|433582|112|241|198|329|184|265|244|296|243|256|296|333|289
gaze0000433470/gaze0000433583.pgm|433470|433583|113|241|198|330|184|266|244|297|243|257|296|334|289
gaze0000433470/gaze0000433584.pgm|433470|433584|114|242|198|331|184|267|244|298|243|257|296|335|289
gaze0000433470/gaze0000433585.pgm|433470|433585|115|243|197|332|184|268|244|298|243|258|296|335|289
gaze0000433470/gaze0000433586.pgm|433470|433586|116|244|197|332|184|269|244|299|243|259|296|336|289
gaze0000433470/gaze0000433587.pgm|433470|433587|117|245|198|333|185|269|244|300|243|259|296|337|290
gaze0000433470/gaze0000433588.pgm|433470|433588|118|245|201|335|188|271|244|301|243|260|296|338|290
gaze0000433470/gaze0000433589.pgm|433470|433589|119|249|201|340|189|272|245|302|244|261|297|338|290
gaze0000433470/gaze0000433590.pgm|433470|433590|120|249|201|342|189|273|245|303|244|262|297|340|291
gaze0000433470/gaze0000433591.pgm|433470|433591|121|248|199|337|187|274|245|304|244|262|297|340|291
gaze0000433470/gaze0000433592.pgm|433470|433592|122|249|198|338|186|275|245|305|244|263|297|341|291
gaze0000433470/gaze0000433593.pgm|433470|433593|123|249|197|338|185|276|245|306|244|264|297|342|291
...

Anhang C

Exemplarischer Auszug aus dem Auswerteskript

Die Auswertung wurde mittels eines Skripts automatisiert, das beispielsweise folgende Ausgabe lieferte:

```
****************************************
****scanning window width & height: 12****
****************************************
Trainpersons: dariu1 christian1 tilo1 udo1
--------------------------------------------------------------------------------------------
Number of moments: 1
--------------------------------------------------------------------------------------------
Train classifier.
Adapting coefficient matrix of the polynomial classifier.
...
...
...
********************************************************************************************
Number of moments: 10
--------------------------------------------------------------------------------------------
Train classifier.
Adapting coefficient matrix of the polynomial classifier.

PC_ADAPT: vvvvvvvvvvvvvvvvvvvvvvvvv INFO vvvvvvvvvvvvvvvvvvvvvvvvv
PC_C_PSL: dimension of original features                    =   10
PC_C_PSL: degree of polynomial classifier                   =    2
PC_C_PSL: length of polynomial structure list               =   66
PC_EH_FE: dimension of enhanced features                    =   66
PC_MOM:   < class   1 > number of used observation vectors   = 1357
PC_MOM:   < class   2 > number of used observation vectors   = 1355
PC_MOM:   < class   3 > number of used observation vectors   = 1354
PC_MOM:   < class   4 > number of used observation vectors   = 1353
PC_MOM:   < class   5 > number of used observation vectors   = 1352
PC_MOM:   < class   6 > number of used observation vectors   = 1354
PC_MOM:   < class   7 > number of used observation vectors   =174080
PC_MOM:   < class sum > number of used observation vectors   =182205
PC_PAIR:  the classifier structure used is PAIRWISE
PC_PAIR:  all CLASSES are treated equally
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  1 versus class  2
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   60
PC_PAIR:  residual variance of adaption                     =  0.495
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  1 versus class  3
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   60
PC_PAIR:  residual variance of adaption                     = 0.0902
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  1 versus class  4
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   60
PC_PAIR:  residual variance of adaption                     =  0.105
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  1 versus class  5
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     = 0.0747
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  1 versus class  6
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     = 0.0826
```

```
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  1 versus class  7
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.227
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  2 versus class  3
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   60
PC_PAIR:  residual variance of adaption                     = 0.0613
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  2 versus class  4
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   60
PC_PAIR:  residual variance of adaption                     =  0.102
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  2 versus class  5
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   60
PC_PAIR:  residual variance of adaption                     = 0.0503
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  2 versus class  6
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     = 0.0711
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  2 versus class  7
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.246
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  3 versus class  4
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.378
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  3 versus class  5
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.278
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  3 versus class  6
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.254
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  3 versus class  7
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.152
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  4 versus class  5
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.103
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  4 versus class  6
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.119
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  4 versus class  7
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   60
PC_PAIR:  residual variance of adaption                     =  0.115
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  5 versus class  6
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   62
PC_PAIR:  residual variance of adaption                     =  0.431
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  5 versus class  7
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   60
PC_PAIR:  residual variance of adaption                     =  0.163
```

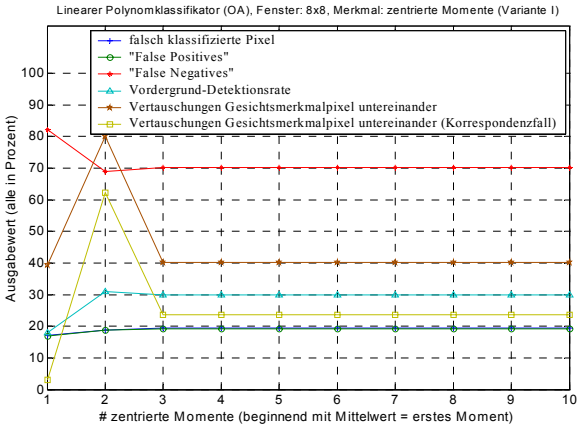
```
PC_REGRE: <minimum residual variance> pivoting (default)
PC_PAIR:  adapting class  6 versus class  7
PC_PAIR:  dimension of enhanced features                    =   66
PC_PAIR:  unused features for adaption                      =   61
PC_PAIR:  residual variance of adaption                     =  0.178
elapsed_time =
   10.1167
-------------------------------------------------------------------------------------------
Testperson_1: andy1
-------------------------------------------------------------------------------------------
Number of testsamples / testframes / foreground pixel / background pixel
11210038 / 38 / 5700 / 11204338
-------------------------------------------------------------------------------------------
Total amount of misclassification (in percent): 2419182 (21.58%)
False positives (in percent): 2413791 (21.54%)
False negatives (in percent): 4947 (86.79%)
Correctly detected foreground pixel (foreground detection rate): 753 (13.21%)
Misclassification across facial features (in percent): 444 (58.96%)
Misclassification across facial features, swapping allowed (in percent): 439 (58.30%)
-------------------------------------------------------------------------------------------
Confusionmatrix for all testsamples of this testperson:
           0           0           0           0           0           0         950
           1           6           0           0           0           0         943
           0           5           0           4           0           0         941
          47         150           0         303           0           0         450
           0           0           0           0           0           0         950
           0          10          58         169           0           0         713
      224023      331729      247129      269352      629089      712469     8790547
-------------------------------------------------------------------------------------------
Testperson_2: nikki1
-------------------------------------------------------------------------------------------
Number of testsamples / testframes / foreground pixel / background pixel
8555029 / 29 / 4175 / 8550854
-------------------------------------------------------------------------------------------
Total amount of misclassification (in percent): 2984953 (34.89%)
False positives (in percent): 2981566 (34.87%)
False negatives (in percent): 1624 (38.90%)
Correctly detected foreground pixel (foreground detection rate): 2551 (61.10%)
Misclassification across facial features (in percent): 1763 (69.11%)
Misclassification across facial features, swapping allowed (in percent): 1053 (41.28%)
-------------------------------------------------------------------------------------------
Confusionmatrix for all testsamples of this testperson:
           9         104           0           0           0           0         587
           9          59           0           3           0           0         654
          16          17          53         276           0           0         338
           0          16          14         645           0           0           0
           0           0         268          80           0         307          45
           0           0         173         480           0          22           0
      203687      346724      298169      291193      870884      970909     5569288
-------------------------------------------------------------------------------------------
Testperson_3: urban1
-------------------------------------------------------------------------------------------
Number of testsamples / testframes / foreground pixel / background pixel
8555029 / 29 / 4350 / 8550679
-------------------------------------------------------------------------------------------
Total amount of misclassification (in percent): 1146506 (13.40%)
False positives (in percent): 1143981 (13.38%)
False negatives (in percent): 438 (10.07%)
Correctly detected foreground pixel (foreground detection rate): 3912 (89.93%)
Misclassification across facial features (in percent): 2087 (53.35%)
Misclassification across facial features, swapping allowed (in percent): 386 (9.87%)
-------------------------------------------------------------------------------------------
Confusionmatrix for all testsamples of this testperson:
           0         361           1          89           0           0         274
           0         430           0         137           0           0         158
           0           0         107         528           0          90           0
           0           0          95         601           0          29           0
           0           0           0          10           8         707           0
           0           0           3          27          10         679           6
       11879        6721       26369       24718      148079      926215     7406698
-------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
```

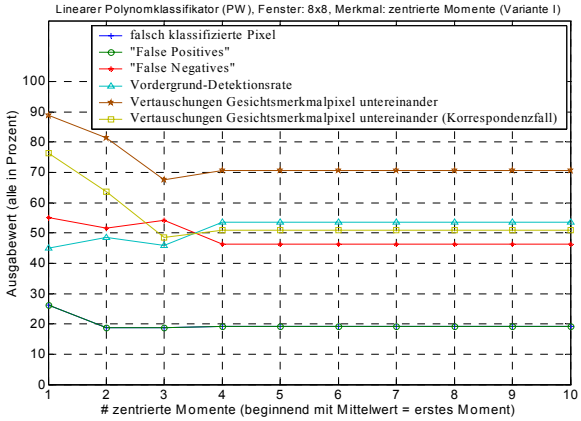
```
All testpersons:
-------------------------------------------------------------------------------------------
Number of testsamples / testframes / foreground pixel / background pixel
28320096 / 96 / 14225 / 28305871
-------------------------------------------------------------------------------------------
Total amount of misclassification (in percent): 6550641 (23.29%)
False positives (in percent): 6539338 (23.26%)
False negatives (in percent): 7009 (45.25%)
Correctly detected foreground pixel (foreground detection rate): 7216 (54.75%)
Misclassification across facial features (in percent): 4294 (60.47%)
Misclassification across facial features, swapping allowed (in percent): 1878 (36.48%)
-------------------------------------------------------------------------------------------
Confusionmatrix for all testsamples of all testpersons:
           9         465           1          89           0           0        1811
          10         495           0         140           0           0        1755
          16          22         160         808           0          90        1279
          47         166         109        1549           0          29         450
           0           0         268          90           8        1014         995
           0          10         234         676          10         701         719
      439589      685174      571667      585263     1648052     2609593    21766533
*******************************************************************************************
```

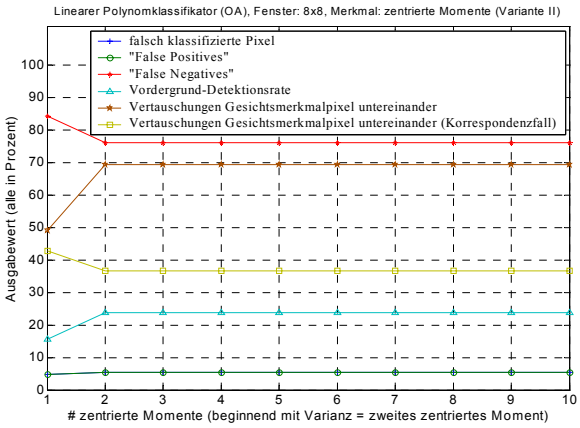
Anhang D

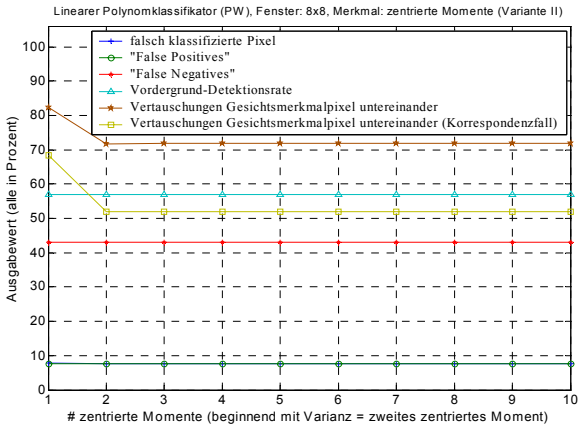
MATLAB Diagramme

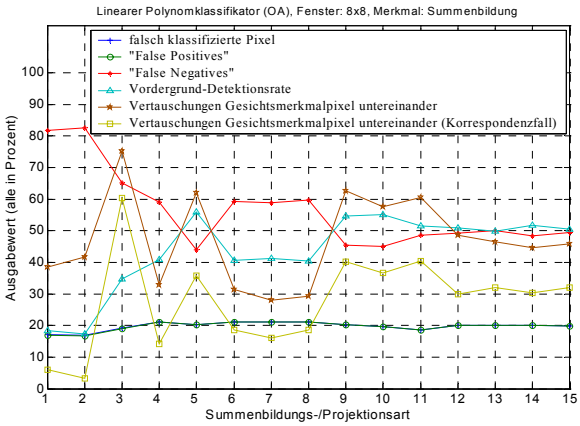
Für detaillierte Untersuchungen des Polynomklassifikators dienen diese Diagramme.

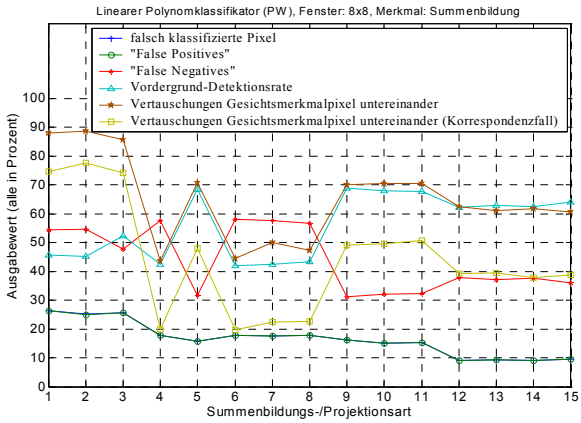


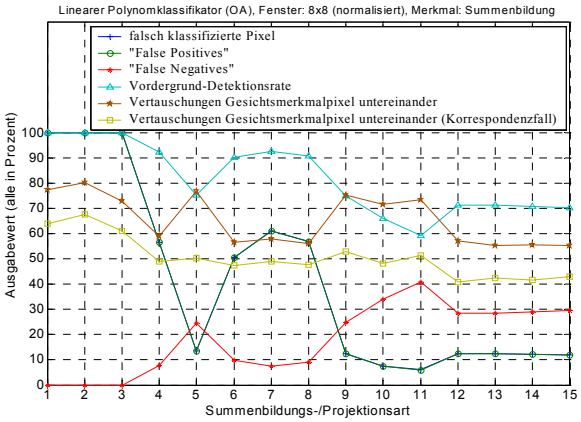


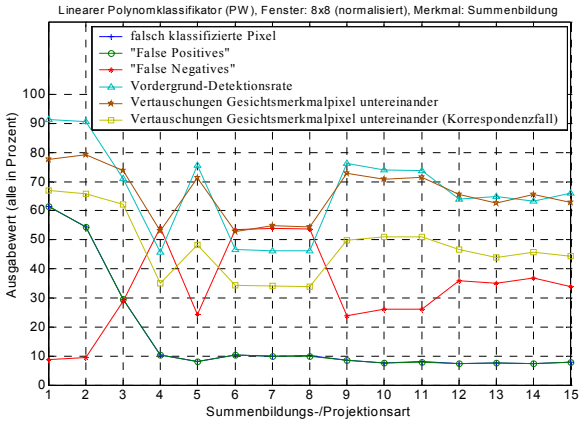


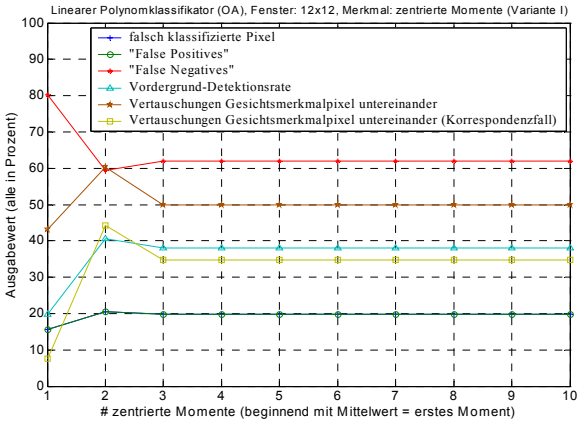


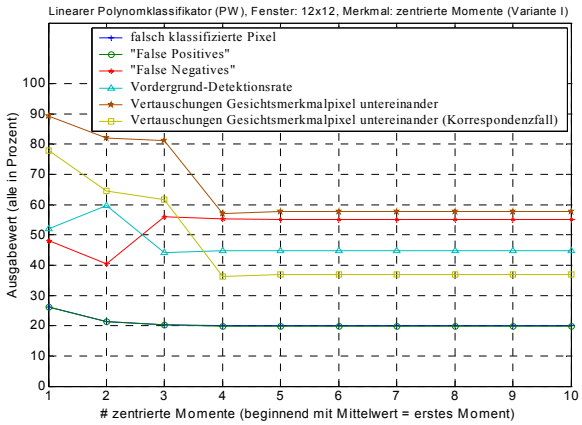


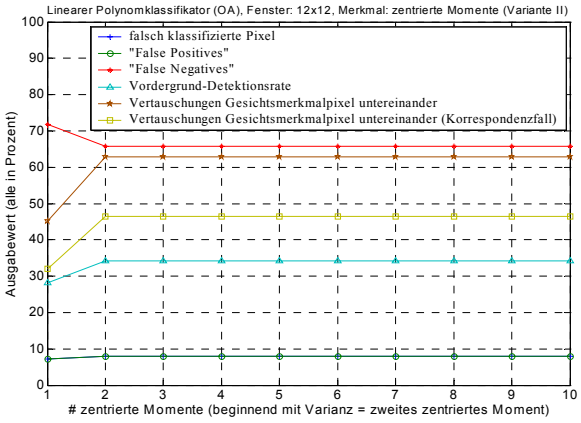


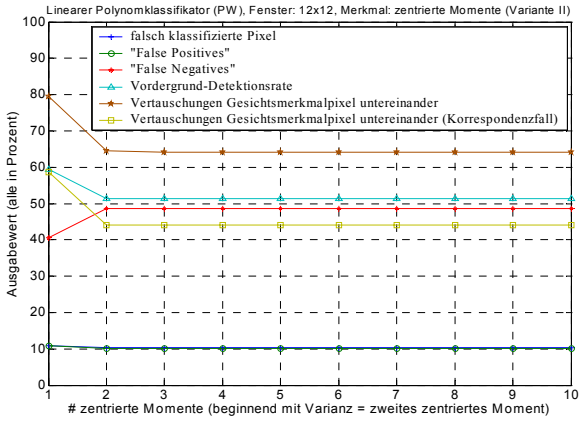

Linearer Polynomklassifikator (OA), Fenster: 8x8 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

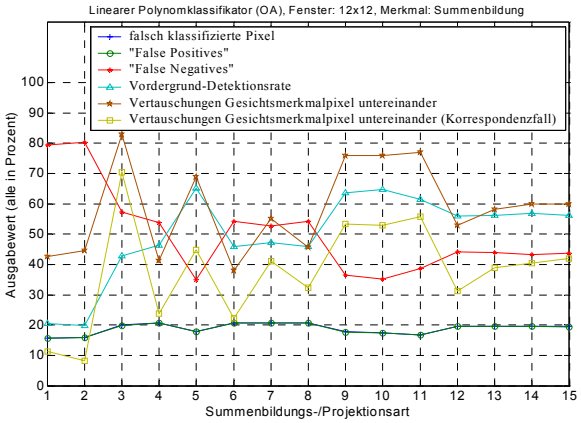

Linearer Polynomklassifikator (PW), Fenster: 8x8 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

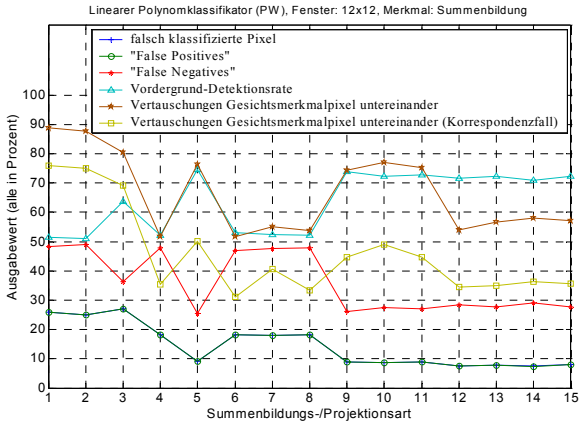

Linearer Polynomklassifikator (OA), Fenster: 12x12, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

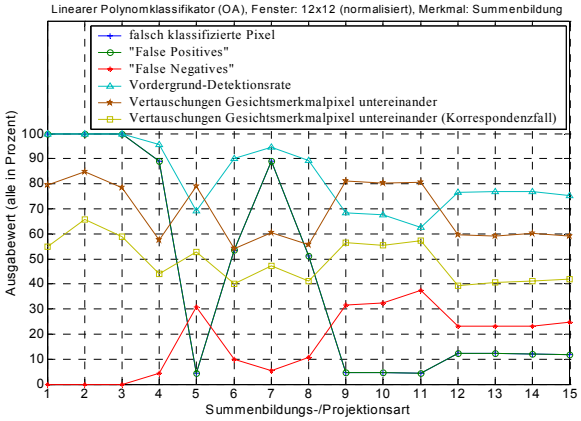

Linearer Polynomklassifikator (PW), Fenster: 12x12, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

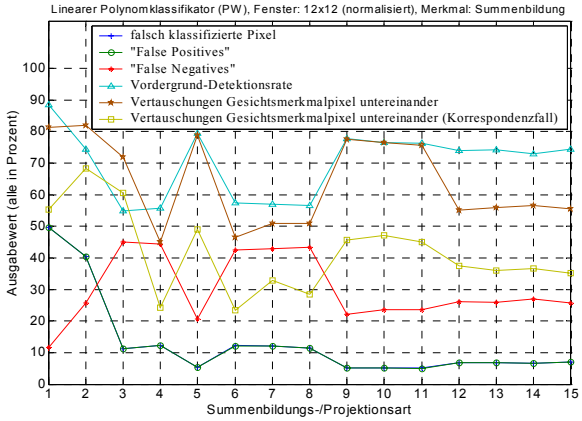

Linearer Polynomklassifikator (OA), Fenster: 12x12, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

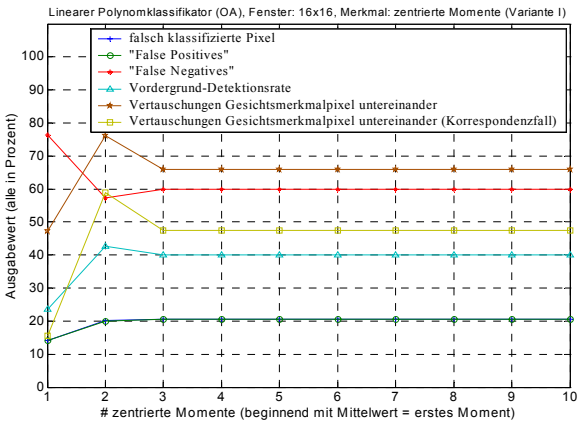

Linearer Polynomklassifikator (PW), Fenster: 12x12, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

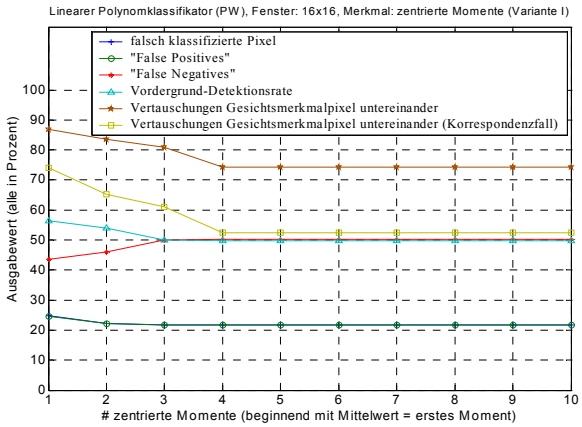

Linearer Polynomklassifikator (OA), Fenster: 12x12, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

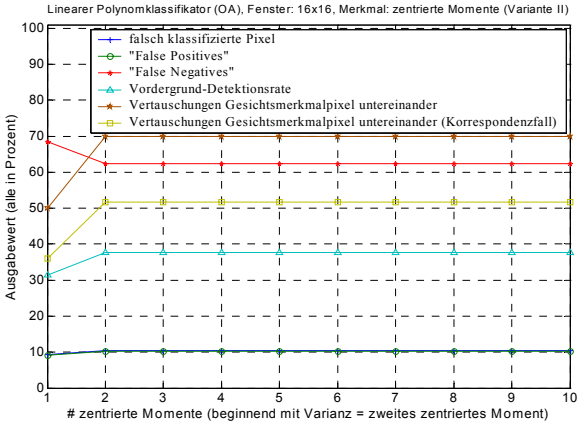

Linearer Polynomklassifikator (PW), Fenster: 12x12, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

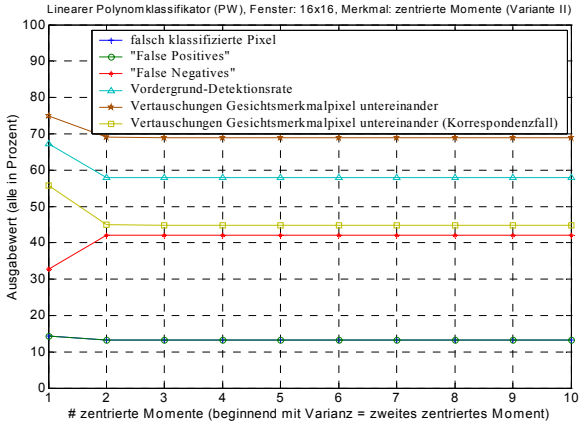

Linearer Polynomklassifikator (OA), Fenster: 12x12 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

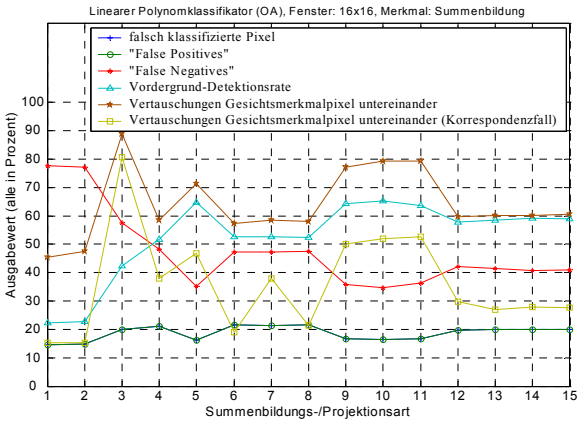

Linearer Polynomklassifikator (PW), Fenster: 12x12 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

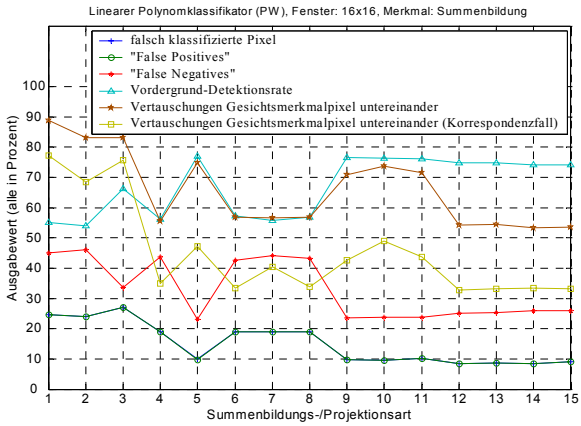

Linearer Polynomklassifikator (OA), Fenster: 16x16, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

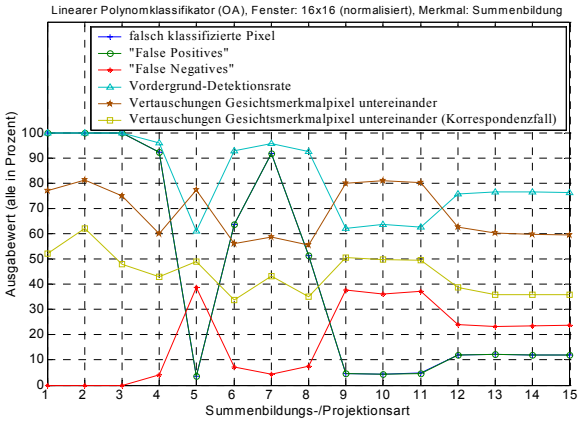

Linearer Polynomklassifikator (PW), Fenster: 16x16, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

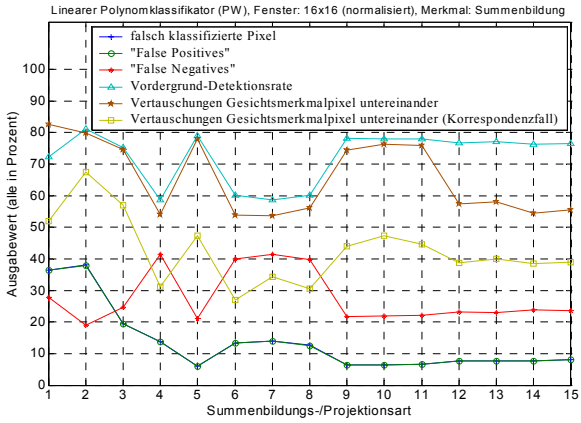

Linearer Polynomklassifikator (OA), Fenster: 16x16, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

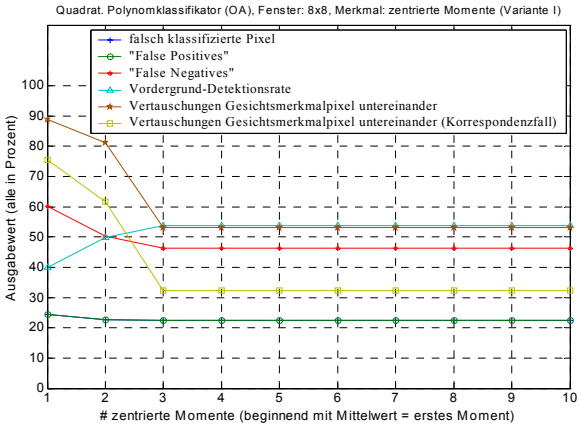

Linearer Polynomklassifikator (PW), Fenster: 16x16, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

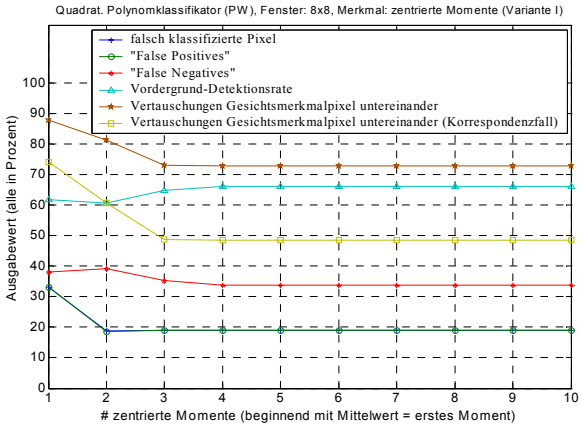

Linearer Polynomklassifikator (OA), Fenster: 16x16, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

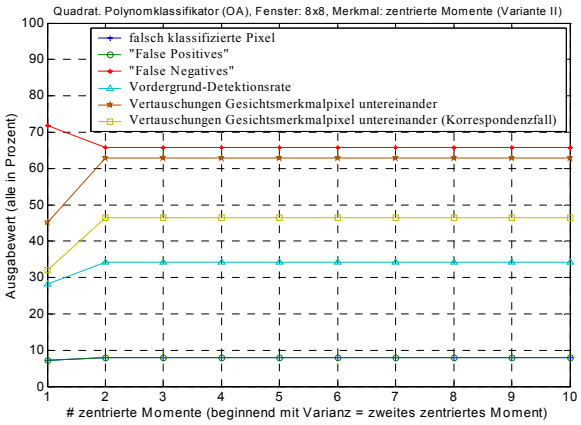

Linearer Polynomklassifikator (PW), Fenster: 16x16, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

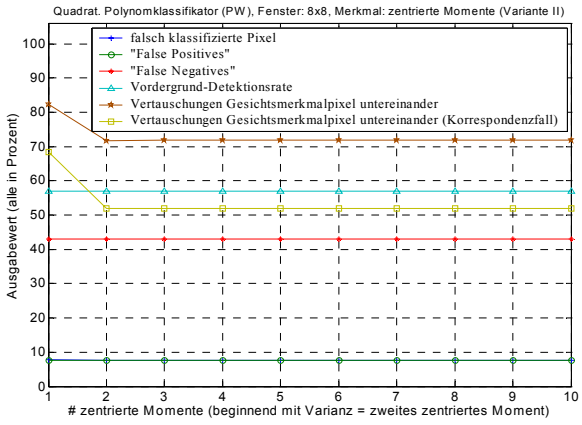

Linearer Polynomklassifikator (OA), Fenster: 16x16 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

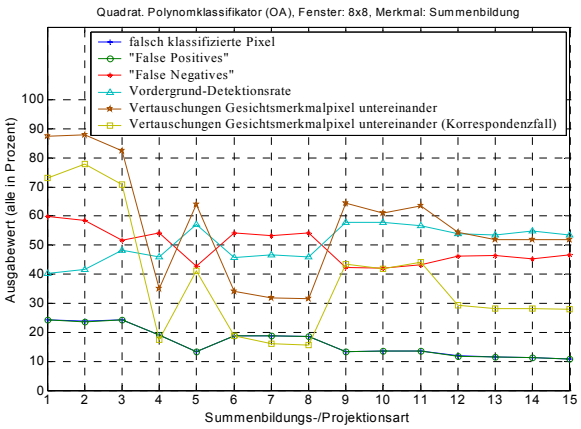

Linearer Polynomklassifikator (PW), Fenster: 16x16 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

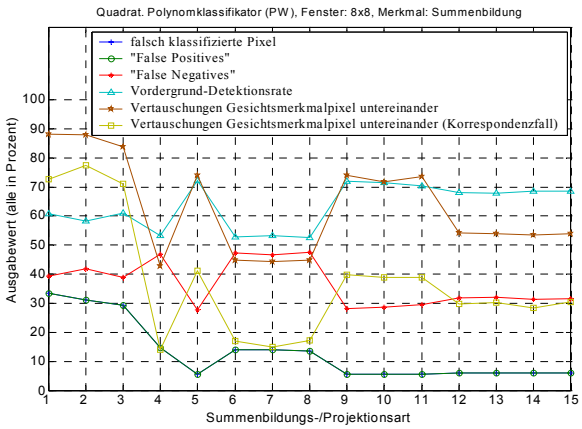

Quadrat. Polynomklassifikator (OA), Fenster: 8x8, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

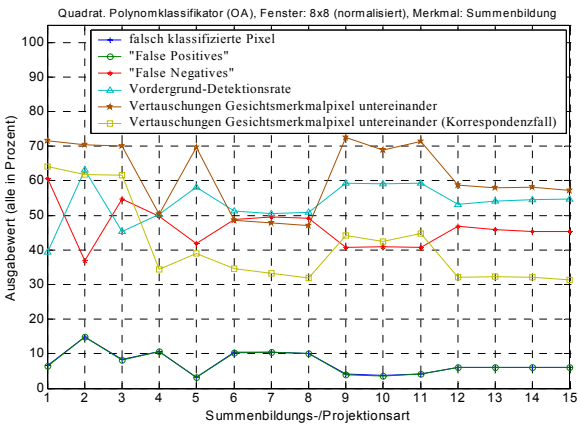

Quadrat. Polynomklassifikator (PW), Fenster: 8x8, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

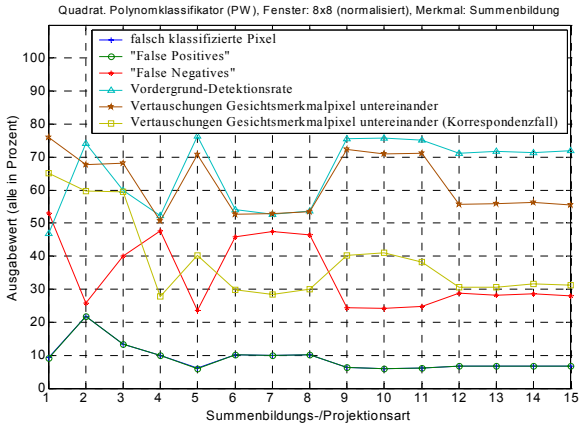

Quadrat. Polynomklassifikator (OA), Fenster: 8x8, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

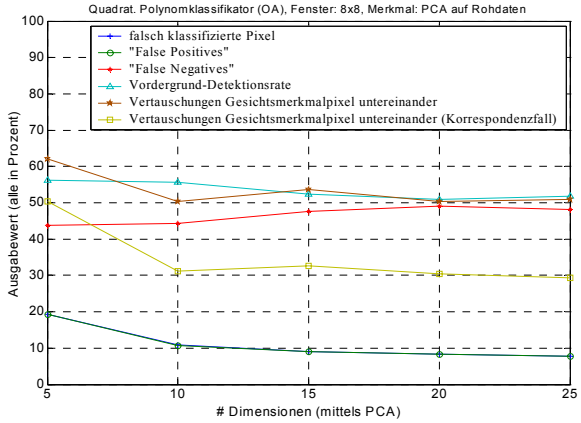

Quadrat. Polynomklassifikator (PW), Fenster: 8x8, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

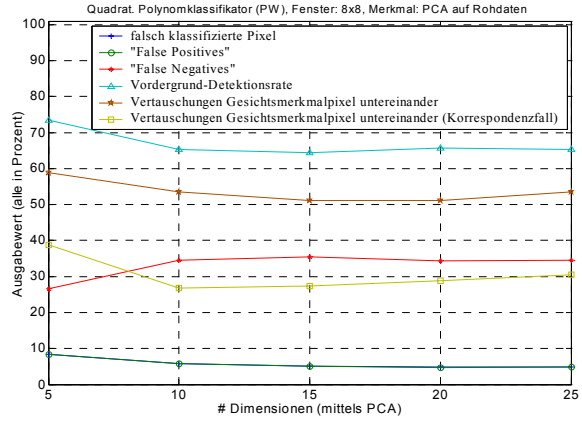

Quadrat. Polynomklassifikator (OA), Fenster: 8x8, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

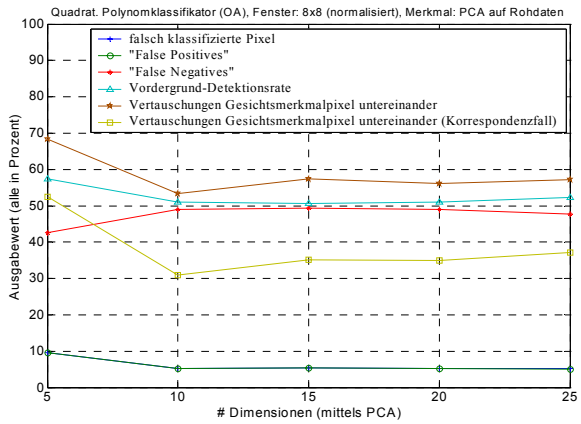

Quadrat. Polynomklassifikator (PW), Fenster: 8x8, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

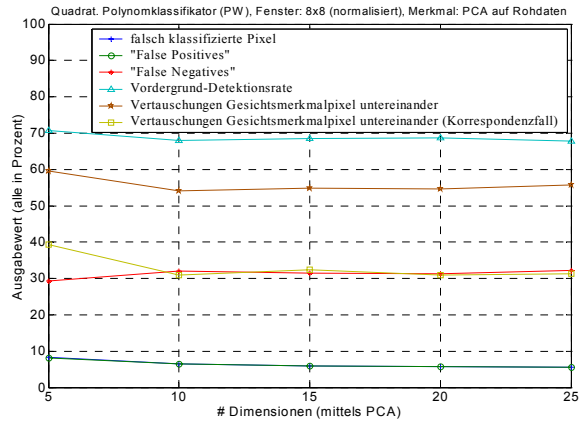

Quadrat. Polynomklassifikator (OA), Fenster: 8x8 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

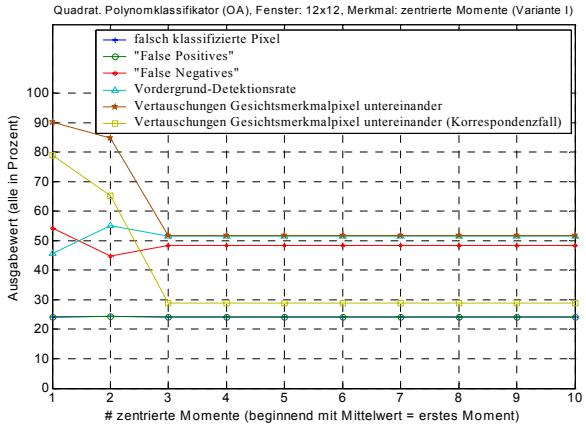

Quadrat. Polynomklassifikator (PW), Fenster: 8x8 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

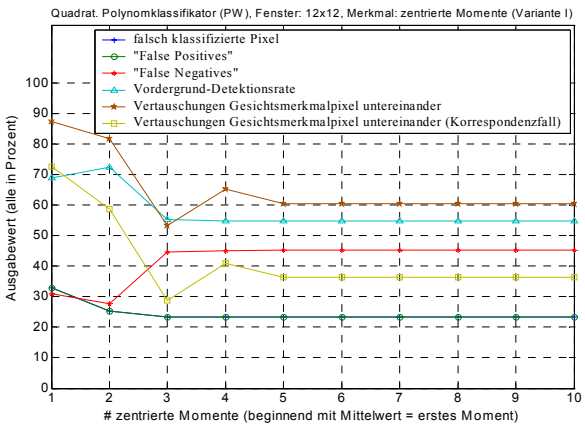

Quadrat. Polynomklassifikator (OA), Fenster: 8x8, Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

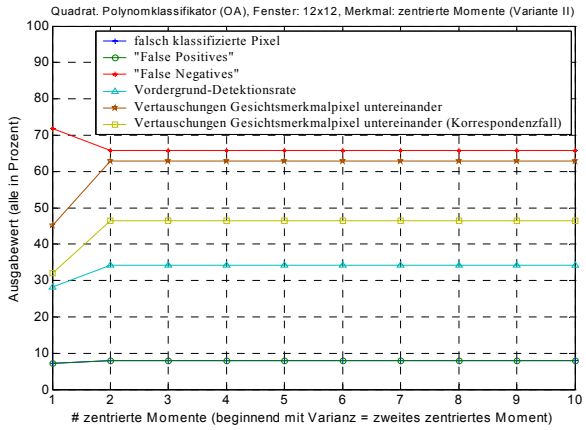

Quadrat. Polynomklassifikator (PW), Fenster: 8x8, Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

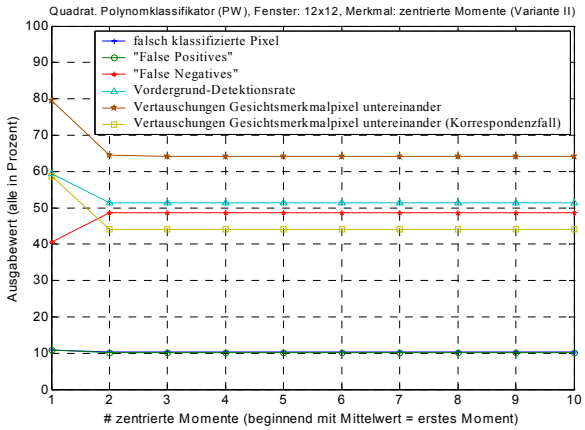

Quadrat. Polynomklassifikator (OA), Fenster: 8x8 (normalisiert), Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

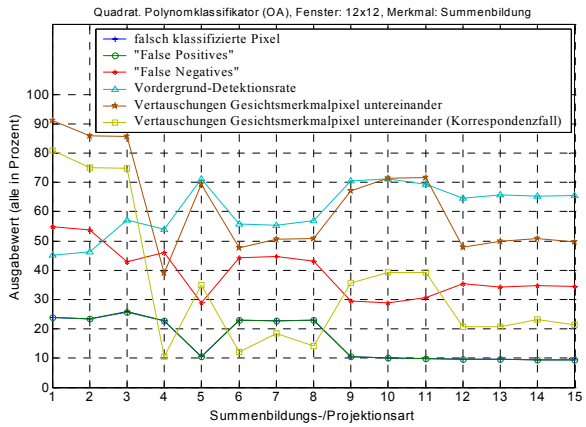

Quadrat. Polynomklassifikator (PW), Fenster: 8x8 (normalisiert), Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

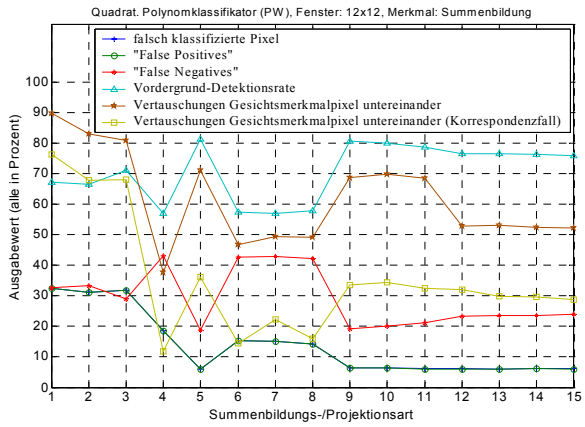

Quadrat. Polynomklassifikator (OA), Fenster: 12x12, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

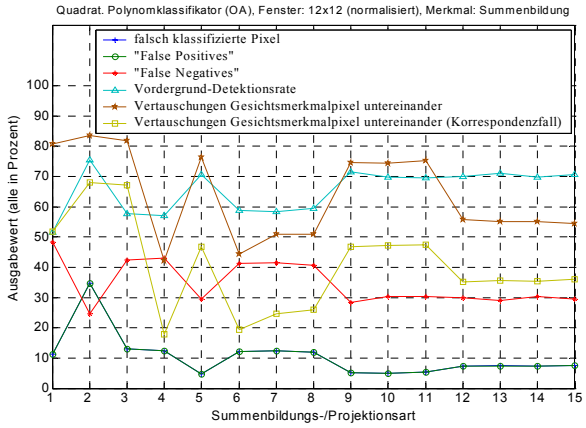

Quadrat. Polynomklassifikator (PW), Fenster: 12x12, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

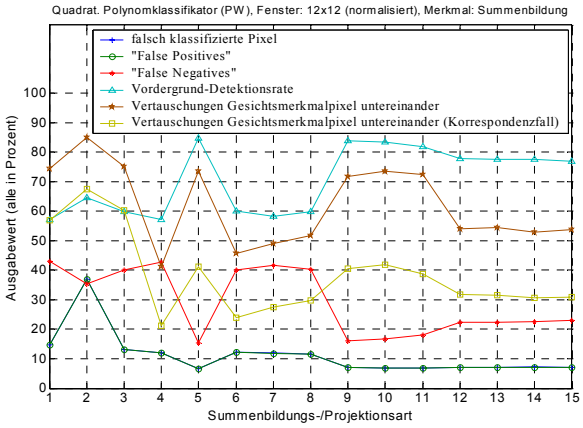

Quadrat. Polynomklassifikator (OA), Fenster: 12x12, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

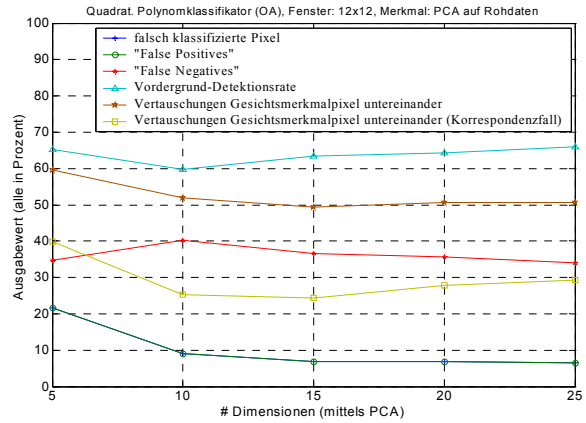

Quadrat. Polynomklassifikator (PW), Fenster: 12x12, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

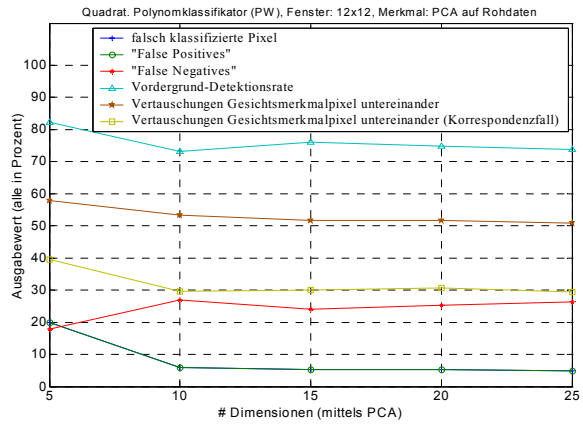

Quadrat. Polynomklassifikator (OA), Fenster: 12x12, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

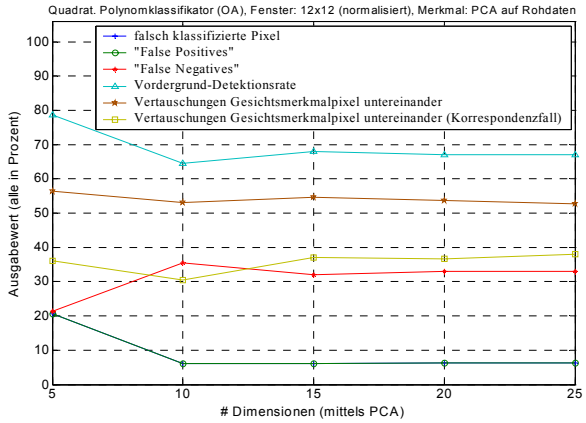

Quadrat. Polynomklassifikator (PW), Fenster: 12x12, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

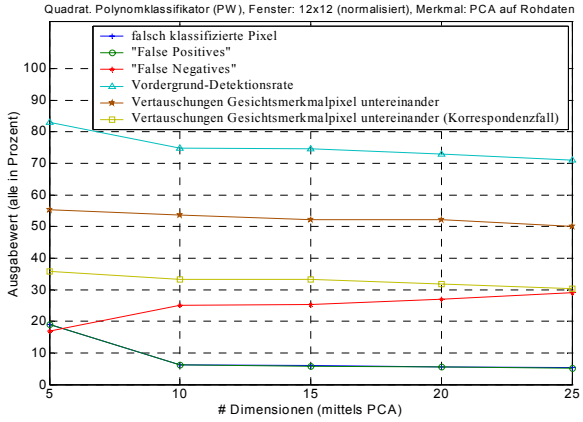

Quadrat. Polynomklassifikator (OA), Fenster: 12x12 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

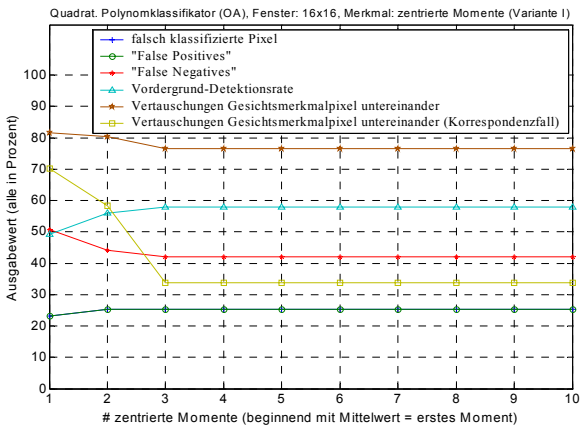

Quadrat. Polynomklassifikator (PW), Fenster: 12x12 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart

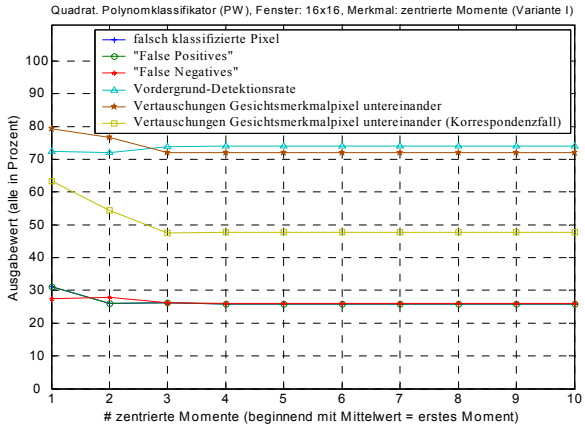

Quadrat. Polynomklassifikator (OA), Fenster: 12x12, Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

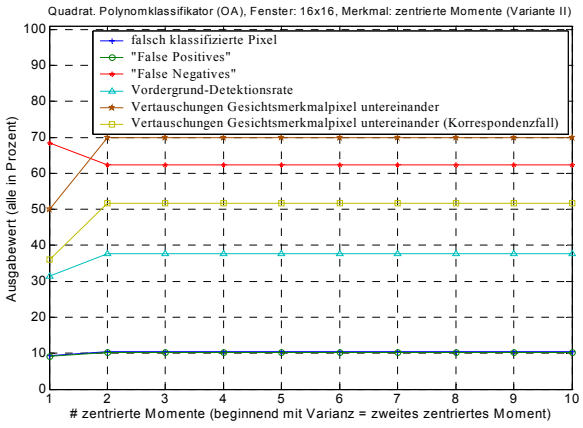

Quadrat. Polynomklassifikator (PW), Fenster: 12x12, Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

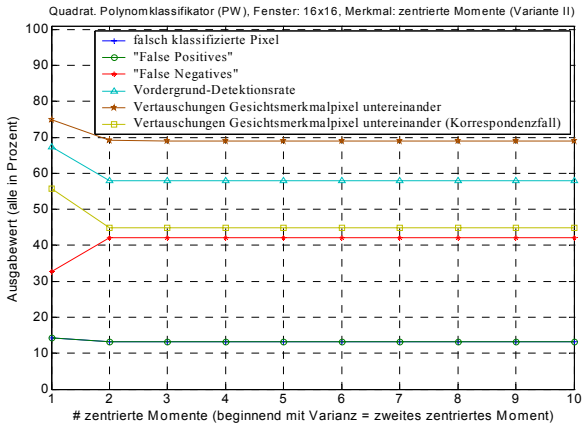

Quadrat. Polynomklassifikator (OA), Fenster: 12x12 (normalisiert), Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

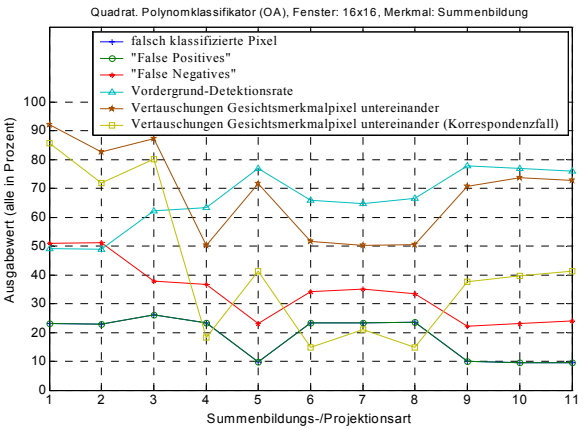

Quadrat. Polynomklassifikator (PW), Fenster: 12x12 (normalisiert), Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

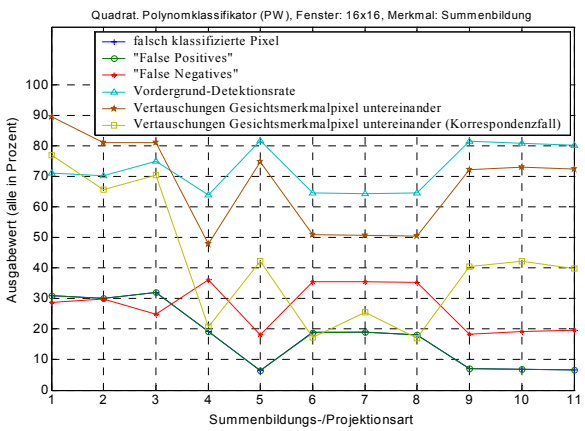

Quadrat. Polynomklassifikator (OA), Fenster: 16x16, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

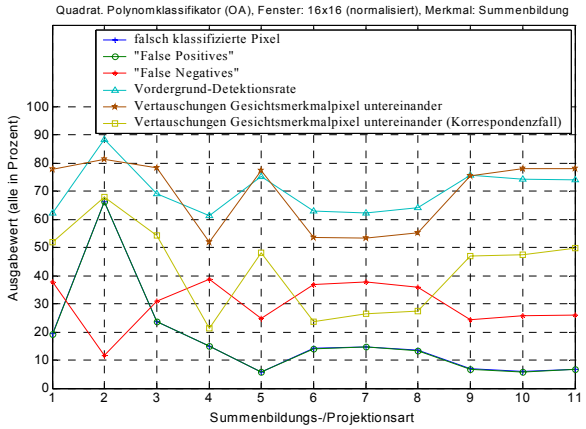

Quadrat. Polynomklassifikator (PW), Fenster: 16x16, Merkmal: zentrierte Momente (Variante I)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Mittelwert = erstes Moment)

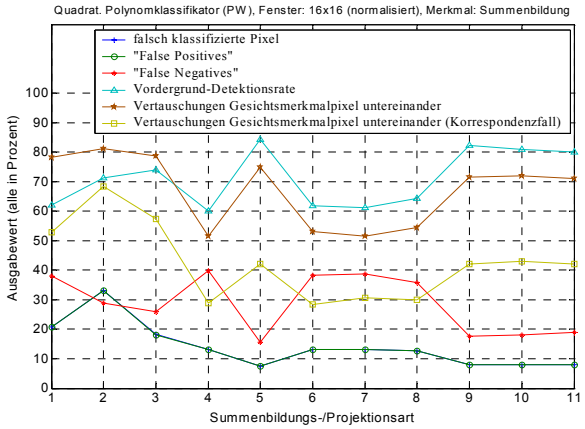

Quadrat. Polynomklassifikator (OA), Fenster: 16x16, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

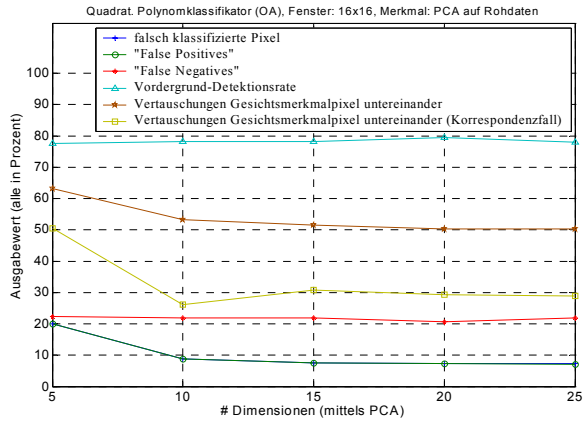

Quadrat. Polynomklassifikator (PW), Fenster: 16x16, Merkmal: zentrierte Momente (Variante II)
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
zentrierte Momente (beginnend mit Varianz = zweites zentriertes Moment)

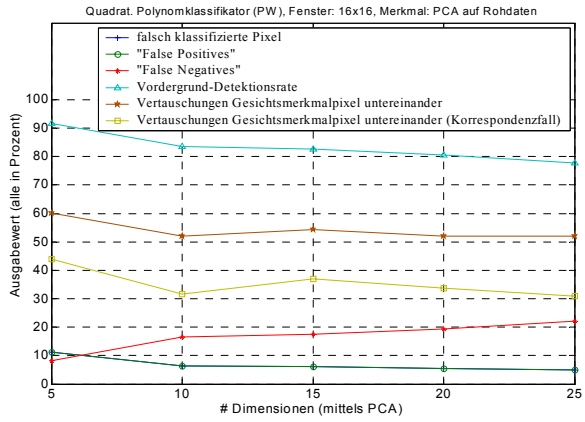

Quadrat. Polynomklassifikator (OA), Fenster: 16x16, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart


Quadrat. Polynomklassifikator (PW), Fenster: 16x16, Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart


Quadrat. Polynomklassifikator (OA), Fenster: 16x16 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart


Quadrat. Polynomklassifikator (PW), Fenster: 16x16 (normalisiert), Merkmal: Summenbildung
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Summenbildungs-/Projektionsart


Quadrat. Polynomklassifikator (OA), Fenster: 16x16, Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)


Quadrat. Polynomklassifikator (PW), Fenster: 16x16, Merkmal: PCA auf Rohdaten
falsch klassifizierte Pixel
"False Positives"
"False Negatives"
Vordergrund-Detektionsrate
Vertauschungen Gesichtsmerkmalpixel untereinander
Vertauschungen Gesichtsmerkmalpixel untereinander (Korrespondenzfall)
Ausgabewert (alle in Prozent)
Dimensionen (mittels PCA)

Anhang E

Verwendete Abkürzungen

ASM	**A**ctive **S**hape **M**odel
CCD	**C**harge **C**oupled **D**evice (deutsch: amplitudenkontinuierliches Schieberegister)
CMU	**C**arnegie **M**ellon **U**niversity
DFFS	**D**istance **f**rom **F**ace **S**pace
FA	**F**actor **A**nalysis
FFT	**f**ast **F**ourier **T**ransformation (deutsch: schnelle Fouriertransformation)
FIR	**f**inite **I**mpulse **R**esponse (Filter)
FRT	**F**ace **R**ecognition **T**echnology (deutsch: Gesichtserkennungstechnologie)
HAT	**H**aupt**a**chsen**t**ransformation
HMM	**H**idden **M**arkov **M**odel
KL	**K**arhunen-**L**oève (Transformation/Entwicklung/Methode)
LBF	**l**inkes **B**lick**f**eld
LoG	**L**aplace **o**f **G**aussian
MAP	**M**aximum **a** **p**osteriori
MIP	**m**ost **i**nformative **P**ixels
MIT	**M**assachusetts **I**nstitute of **T**echnology
ML	**M**aximum **L**ikelihood
MLP	**M**ulti**l**ayer **P**erceptron (deutsch: mehrschichtiges Perzeptron)
MRF	**M**arkov **R**andom **F**ield
NN	**n**euronal **N**etwork (deutsch: neuronales Netz)
OA	**O**NEvs**A**LL/**o**ne versus **a**ll others (deutsch: eine gegen alle anderen)
OCR	**O**ptical **C**haracter **R**ecognition (deutsch: optische Zeichenerkennung)
PAL	**P**hase **A**lternation **L**ine
PCA	**P**rincipal **C**omponent **A**nalysis
PDM	**P**oint **D**istribution **M**odel
PW	**P**AIR**W**ISE (deutsch: eine gegen jeweils eine andere)
RGB	**r**ed, **g**reen, **b**lue (deutsch: rot, grün, blau)
ROC	**R**eceiver **O**perator **C**haracteristic
ROI	**R**egion **o**f **I**nterest
SGLD	**s**patial **G**rey **L**evel **D**ependency (Matrix)
SVD	**S**ingular **V**alue **D**ecomposition (deutsch: Singulärwertzerlegung)
SVM	**S**upport **V**ector **M**achine
TDIDT	**T**op **D**own **I**nduction of **D**ecision **T**rees

Anhang F

Verweise auf Internetseiten

Die folgenden Webseiten geben begleitende Informationen und veranschaulichen die in dieser Diplomarbeit behandelten Themen. Natürlich kann in der schnelllebigen Internetwelt nicht gewährleistet werden, dass die Links dauerhaft gültig sind.

1. Links zur Gesichtserkennung („Face Recognition“) und -detektion („Face Detection“):
   - **http://www-2.cs.cmu.edu/~cil/vision.html**
     ist die „Computer Vision“-Homepage der Carnegie Mellon University (CMU). Seit 1994 hat man dadurch einen zentralen Zugriff auf relevante Verweise im Internet im Bereich der Forschung des maschinellen Sehens, wobei dann nach spezifischen Unterthemen verzweigt wird (’Vision Groups’, ’Hardware’, ’Software’, ’Demos’, ’Test Images’, ’Conferences’, ’Publications’, ’General Info’ und ’Related Links’).
   - **http://www.ri.cmu.edu/labs/lab_51.html**
     ist die Seite der „Face Group“ (Prof. Kanade) am „Robotics Institute“ der CMU.
   - **http://www-white.media.mit.edu/vismod/demos/facerec/**
     ist die „Face Recognition“-Demoseite der „Vision and Modeling“-Gruppe am MIT (Massachusetts Institute of Technology) Media Laboratory. Hier finden sich unter anderem Erläuterungen zu dem Eigengesichter-Ansatz.
   - **http://www.dai.ed.ac.uk/CVonline/**
     „CVonline“ (CV = Computer Vision) ist ein „on-line“ Kompendium über das maschinelle Sehen für Forschung und Anwendung der Informatik-Fakultät an der Universität Edinburgh.
   - **http://vision.ai.uiuc.edu/mhyang/face-detection-survey.html**
     Diese Seite enthält nützliche Informationen (wie Beispielcode und –skripte), um Methoden für die Gesichtsdetektion zu entwickeln. Außerdem wird auf bekannten (Benchmark-)Gesichtsdatenbanken für eine Evaluierung verwiesen.
   - **http://home.t-online.de/home/Robert.Frischholz/face.htm**
     ist die „Face Detection“-Homepage von Dr. Robert Frischholz. Hier liegt der Schwerpunkt innerhalb der Gesichtserkennung auf der genauen Detektion von Gesichtern in willkürlichen Szenen. Die Homepage bietet unter anderem Informationen beziehungsweise Links zu Software, „on-line“ Demos, Gesichtsdatenbanken, Veröffentlichungen, Techniken und Patenten.
   - **http://www.research.att.com/resources/trs/TRs/96/96.5/96.5.1/96.5.1.abs.html**
     und
     http://www.research.att.com/resources/trs/TRs/96/96.4/96.4.1/96.4.1.body.htm
     zeigen ein multimodales System der Forschung von AT&T Bell (Lucent Technology) für die Bestimmung von Köpfen/Gesichtern und Gesichtsteilen.
   - **http://research.microsoft.com/~szli/FaceGroup/default.asp**
     ist die Seite der Gruppe bei der Microsoft eigenen Forschung, die sich mit Gesichtern beschäftigt; unter anderem mit Demos und Veröffentlichungen.

- **http://troi.ifp.uiuc.edu/~antonio/index.html**
  ist die Homepage von Antonio Colmenarez, der sich vor allem für die Gesichtsanalyse innerhalb des HCII (Human-Computer Intelligent Interface) interessiert.
- **http://www.wi.leidenuniv.nl/home/lim/face.detection.html**
  Hier wird eine Methode zur Gesichtsdetektion und –erkennung beschrieben und mit Demos veranschaulicht.
- **http://mi.eng.cam.ac.uk/~kcy/Research.html**
  beschreibt den Forschungsschwerpunkt von Kin Choong Yow („Speech, Vision and Robotics"-Gruppe, Universität Cambridge) im Bereich der automatischen Gesichtsdetektion und –lokalisierung.
- **http://web.media.mit.edu/~jebara/uthesis/thesis.html**
  ist die „on-line" Version der Thesis "3D Pose Estimation and Normalization for Face Recognition"von Tony S. Jebara (Centre for Intelligent Machines, McGill University Montréal).
- **http://www.markus-hofmann.de/**
  Homepage von Markus Hofmann zu seiner Diplomarbeit „Grundsätzliche Untersuchung von Bildverarbeitungsalgorithmen zur Gesichtererkennung".

2. Spezielle Links zu Gesichtsmerkmalen („Facial Features"):
   - **http://image.pirl.umd.edu/knkim/**
     Kyungnam Kim (Universität Maryland) beschäftigt sich unter anderem mit „Facial Feature Tracking" und hier speziell mit „Eye Tracking".
   - **http://www.syseng.anu.edu.au/~gareth/homepage/Research.htm#feature_detection**
     zeigt Ansätze zum Tracken des Mundes und anderer Gesichtsmerkmale.
   - **http://w3.sys.es.osaka-u.ac.jp/~yokoyama/ronbun/roman96/sum-roman96.html**
     illustriert Methoden („Snakes") für die automatische Detektion von Kontur- und Merkmalspunkten im Gesicht.

3. Links zu Entscheidungsbäumen:
   - **http://www2.cs.uregina.ca/~hamilton/courses/831/notes/ml/dtrees/4_dtrees1.html**
     ist eine online Foliensammlung über Entscheidungsbäume.
   - **http://www.cs.mdx.ac.uk/staffpages/serengul/Concept.Learning.and.Decision.Trees.htm**
     bietet ebenfalls online Informationen zu Entscheidungsbäumen.
   - **http://www.cs.mdx.ac.uk/staffpages/serengul/ML/supervised.html**
     zeigt Ansätze (hier speziell Entscheidungsbäume und deren Trainingsalgorithmen ID3, ID4, ID5 und ID5R) für überwachtes Lernen.
   - **http://www.ai.univie.ac.at/~johann/c45ofai.html**
     stellt eine Variante des C4.5-Algorithmus („c.45-ofai") vor, die man auch herunterladen kann.
   - **http://www.cs.uvm.edu:9080/kdd/c4.5.htm**
     ist ein C4.5 online interface, bei dem man Daten direkt verarbeiten kann.
   - **http://www.cs.umbc.edu/471/lectures/18/sld033.htm**
     ist eine online Foliensammlung über C4.5.

- **http://www.cse.unsw.edu.au/~quinlan/**
  Diese ist die Homepage von Prof. Ross Quinlan mit ausgewählten Veröffentlichungen und Software (wie C4.5 Release 8) zum Herunterladen.
- **http://www.rulequest.com**
  Hier kann man eine (kostenlose) Versuchsversion von See5/C5.0 „downloaden".

4. MATLAB-Links:
   - **http://www.mathworks.com/**
     Offizielle Homepage von „MathWorks" MATLAB.
   - **http://www-edlab.cs.umass.edu/cs570/Documents/matlab.pdf**
     Eine Einführung in MATLAB (im PDF-Format) von Carlo Tomasi, mit dem Schwerpunkt auf digitale Bildverarbeitung.
   - **http://tiger.technion.ac.il/~eladyt/classification/index.htm**
     Gibt einen Überblick über die „Classification Toolbox" von Elad Yom-Tov für MATLAB, die man sich hier für wissenschaftliche Zwecke komplett kostenlos herunterladen kann. Die Toolbox stellt Algorithmen für überwachtes, unüberwachtes Lernen und für die Merkmalsauswahl zur Verfügung (darunter auch die „Baumalgorithmen" ID3, CART und C4.5).
   - **http://www.cs.wisc.edu/~olvi/uwmp/msmt.html**
     enthält MATLAB-Code für den sogenannten „MSM-T" (Multisurface Method Tree)-Algorithmus

5. Sonstiges:
   - **http://citeseer.nj.nec.com/**
     ist sehr hilfreich, wenn man „digitale" Literatur/Papers sucht, vor allem wird gezeigt, welche anderen Papers auf das jeweilige Paper referenzieren.
   - **http://www.prosopagnosia.com/**
     Diese Seite wird von Cecilia Burman gepflegt und zeigt anhand von Beispielbildern, was man sich unter der Krankheit „Prosopagnosia" beziehungsweise „face-blindness" vorstellen sollte.
   - **http://www.mlnet.org**
     MLnet ist das sogenannte „Machine Learning network" und enthält Informationen und Verweise im Bereich des maschinellen Lernens.
   - **http://www.gavrila.net**
     Homepage von Dr. Dariu M. Gavrila, Mitarbeiter am DaimlerChrysler Forschungszentrum in Ulm. Hier werden viele interessante Themen, vor allem in den Bereichen „Smart Vehicles" und „Looking at People" dargestellt.

Abbildungsverzeichnis

Tabellenverzeichnis

Literatur- und Quellenverzeichnis

[1] Rama Chellappa, Charles L. Wilson, and Saad A. Sirohey:
Human and Machine Recognition of Faces: A Survey
Department of Electrical Engineering and Center for Automation Research, University of Maryland, College Park, MD 20742, USA
and National Institute of Standards and Technology, Gaithersburg, MD 20899, USA
Proceedings of the IEEE, vol. 83, no. 5, pp. 705-740 (May 1995)

[2] Ming-Hsuan Yang, David J. Kriegman, and Narendra Ahuja:
Detecting Faces in Images: A Survey
Honda Fundamental Research Labs, Mountain View, CA 94041, USA
and Department of Electrical and Computer Engineering and Beckman Institute, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 1, pp. 34 - 58 (January 2002)

[3] Erik Hjelmås and Boon Kee Low:
Face Detection: A Survey
Department of Informatics, University of Oslo, N-0316 Oslo, Norway
and Department of Meteorology, University of Edinburgh, Edinburgh EH9 3JZ, Scotland, United Kingdom
published in Computer Vision and Image Understanding, vol. 83, pp. 236-274, Academic Press (2001)

[4] Christopher M. Bishop:
Neural Networks for Pattern Recognition
Oxford University Press (1995)
ISBN 0-19-853864-2

[5] Jürgen Schürmann:
Pattern Classification: A Unified View of Statistical and Neural Approaches
Wiley-Interscience, John Wiley & Sons, Inc. (1996)
ISBN 0-471-13534-8

[6] Jürgen Schürmann, Ulrich Kreßel:
Mustererkennung mit statistischen Methoden
Vorlesungsskript, Universität Ulm (2001)

[7] V. N. Vapnik:
The Nature of Statistical Learning Theory
Springer Verlag (1995)

[8] Ulrich Kreßel, I. Graf:
Paarweise Klassifikation mit Support-Vektoren, Mustererkennung 1997
19. DAGM-Symposium, Braunschweig, Seiten 295-304, Springer Verlag (1997)

[9] T. Sakai, M. Nagao, and S. Fujibayashi:
Line extraction and pattern recognition in a photograph
Pattern Recognition, vol. 1, pp 233-248 (1969)

[10] M. D. Kelly:
Visual identification of people by computer
Techn. Rep. AI-130, Stanford AI Proj., Stanford, CA (1970)

[11] V. Govindaraju, S. N. Srihari, and D. B. Sher:
A computational model for face location
In Proc. 3rd IEEE Int'l Conf. on Computer Vision, pp. 718-721 (1990)

[12] M. A. Turk and A. P. Pentland:
Face recognition using eigenfaces
In Proc. Int. Conf. On Patt. Recog., pp. 586-591 (1991)

[13] Z. Hong:
Algebraic feature extraction of image for recognition
Patt. Recog., vol. 24, pp. 211-219 (1991)

[14] L. Sirovich and M. Kirby:
Low-dimensional procedure for the characterization of human face
J. Opt. Soc. Amer., vol. 4, pp. 519-524 (1987)

[15] S. Akamatsu, T. Sasaki, H. Fukamachi, and Y. Suenaga:
A robust face identification scheme–KL expansion of an invariant feature space
In SPIE Proc.: Intell. Robots and Computer Vision X: Algorithms and Techn., vol. 1607, pp. 71-84 (1991)

[16] D. Reisfeld and Y. Yeshuran:
Robust detection of facial features by generalized symmetry
In Proc. 11th Int. Conf. On. Patt. Recog., pp. 117-120 (1992)

[17] M. J. Conlin:
A rule based high level vision system
In SPIE Proc.: Intell. Robots and Compu. Vision, vol. 726, pp. 314-320 (1986)

[18] J. Buhmann, M. Lades, and C. v. d. Malsburg:
Size and distortion invariant object recognition by hierarchical graph matching
In Proc., Int. Joint Conf. on Neural Networks, pp. 411-416 (1990)

[19] M. Lades, J. Vorbruggen, J. Buhmann, J. Lange, C. v. d. Malsburg, and R. Wurtz:
Distortion invariant object recognition in the dynamic link architecture
IEEE Trans. Computers, vol. 42, pp. 300-311 (1993)

[20] N. N.:
Combining evidence from multiple views of 3-D objects
In SPIE Proc.: Sensor Fusion IV: Control Paradigms and Data Structures, vol. 1611 (1991)

[21] A. Shio and J. Sklansky:
Segmentation of people in motion
In Proc. IEEE Workshop on Visual Motion, pp. 325-332 (1991)

[22] G. Yang and T.S. Huang:
Human Face Detection in Complex Background
Pattern Recognition, vol. 27, no. 1, pp. 53-63 (1994)

[23] C. Kotropoulos and I. Pitas
Rule-Based Face Detection in Frontal Views
Proc. Int'l Conf. Acoustics, Speech and Signal Processing, vol. 4, pp. 2537-2540 (1997)

[24] T. Kanade:
Picture Processing by Computer Complex and Recognition of Human Faces
PhD thesis, Kyoto Univ. (1973)

[25] D. Chetverikov and A. Lerch:
Multiresolution Face Detection
Theoretical Foundations of Computer Vision, vol. 69, pp. 131-140 (1993)

[26] H. P. Graf, T. Chen, E. Petajan, and E. Cosatto:
Locating Faces and Facial Parts
Proc. First Int'l Workshop Automatic Face and Gesture Recognition, pp. 41-46 (1995)

[27] T. K. Leung, M. C. Burl, and P. Perona:
Finding Faces in Cluttered Scenes Using Random Labeled Graph Matching
Proc. Fifth IEEE Int'l Conf. Computer Vision, pp. 637-644 (1995)

[28] K. C. Yow and R. Cipolla:
A Probabilistic Framework for Perceptual Grouping of Features for Human Face Detection
Proc. Second Int'l Conf. Automatic Face and Gesture Recognition, pp. 16-21 (1996)

[29] K. C. Yow and R. Cipolla:
Feature-Based Human Face Detection
Image and Vision Computing, vol. 15, no. 9, pp. 713-735 (1997)

[30] E. Saber and A. M. Tekalp:
Frontal-View Face Detection and Facial Feature Extraction Using Color, Shape and Symmetry Based Cost Functions
Pattern Recognition Letters, vol. 17, no. 8, pp. 669-680 (1998)

[31] Q. B. Sun, W. M. Huang, and J. K. Wu:
Face Detection Based on Color and Local Symmetry Information
Proc. Third Int'l Conf. Automatic Face and Gesture Recognition, pp. 130-135 (1998)

[32] A. Yuille, P. Hallinan, and D. Cohen:
Feature Extraction from Faces Using Deformable Templates
Int'l Journal Computer Vision, vol. 8, no. 2, pp.99-111 (1992)

[33] M. Kirby and L. Sirovich:
Application of the Karhunen-Loève Procedure for the Characterization of Human Faces
IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 12, no. 1, pp. 103-108 (Jan. 1990)

[34] K.-K. Sung and T. Poggio:
Example-Based Learning for View-Based Human Face Detection
IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pp. 39-51 (Jan. 1998)

[35] M.-H. Yang, N. Ahuja, and D. Kriegman:
Mixtures of Linear Subspaces for Face Detection
Proc. Fourth Int'l Conf. Automatic Face and Gesture Recognition, pp. 70-76 (2000)

[36] H. Schneiderman and T. Kanade:
Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition
Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 45-51 (1998)

[37] M. S. Lew:
Information Theoretic View-Based and Modular Face Detection
Proc. Second Int'l Conf. Automatic Face and Gesture Recognition, pp. 198-203 (1996)

[38] J. Huang, S. Gutta, and H. Wechsler:
Detection of Human Faces Using Decision Trees
Proc. Second Int'l Conf. Automatic Face and Gesture Recognition, pp. 248-252 (1996)

[39] N. Duta and A. K. Jain:
Learning the Human Face Concept from Black and White Pictures
Proc. Int'l Conf. Pattern Recognition, pp. 1365-1367 (1998)

[40] Tom M. Mitchell:
Machine Learning
McGraw Hill (1997)
ISBN 0-07-115467-1

[41] F. Provost and T. Fawcett:
Robust Classification for Imprecise Environments
Machine Learning, vol. 42, no. 3, pp. 203-231 (2001)

[42] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone:
Classification and Regression Trees (CART)
Chapman & Hall (1993)

[43] C. E. Brodley and P. E. Utgoff:
Multivariate Decision Trees
Machine Learning, 19(1) (1995)

[100] Bernd Jähne:
Digitale Bildverarbeitung
5., überarbeitete und erweiterte Auflage, Springer-Verlag (2002)
ISBN 3-540-41260-3

[101] Andy King:
Farbbasierte Segmentierung von Körperregionen
Studienarbeit, ILKD Prof. Dr. Waibel, Fakultät für Informatik, Universität Karlsruhe

[102] **Teubner-Taschenbuch der Mathematik (Teil 1)**
begr. von I. N. Bronstein und K. A. Semendjajew, weitergeführt von G. Grosche und hrsg. von E. Zeidler
ISBN 3-8154-2001-6