Gesichtserkennung mit Eigenfaces

Filip Martinovský
FB Informatik
FH Zittau/Görlitz
safimart@stud.hs-zigr.de

Philipp Wagner
FB Informatik
FH Zittau/Görlitz
siphwagn@stud.hs-zigr.de

Abstract

Die maschinelle Gesichtserkennung als biometrische Identifikationsmethode stellt hohe Anforderungen an ihre Algorithmen. Gute Gesichtserkennungsverfahren sollten eine niedrige Falschklassifikationsrate aufweisen, Robustheit gegenüber Störeinflüssen besitzen und dabei möglichst performant arbeiten.

Dieser Beleg stellt Eigenfaces vor, ein auf der Hauptkomponentenanalyse basierendes, statistisches Verfahren zur Gesichtserkennung.[TP91] Die Idee besteht darin aus einer Menge an normalisierten Beispielbildern die Linearkombinationen zu finden, die ein Gesichtsbild am besten approximieren.

Eine prototypische Implementierung erfolgte in C++ und der Bibliothek OpenCV.

1 Einführung

Durch die allverfügbare Rechenleistung hält die Analyse biometrischer Merkmale mit statistischen Verfahren Einzug in unserer heutigen Gesellschaft. Galt die Gesichtserkennung noch vor 20 Jahren als eines der komplexesten Probleme der Bildverarbeitung findet heute selbst die Digitalkamera Gesichter von Personen im Bildbereich und adjustiert dementsprechend den Fokus. Der Reisepass muss ein Biometrisch korrektes Photo haben[1] und Googles Bildersuche filtert bei Bedarf bekannte Köpfe.

Die Eigenfaces-Methode zur Gesichtserkennung, der Name entstammt der Hauptkomponentenanalyse und der Lösung des Eigenwertproblems, wurde von Pentland und Turk am MIT Media Labs entwickelt. Während Anfangs die Präsenz von Personen in einem Bild sichergestellt werden sollte (um Geräte zur Analyse des Zuschauerverhaltens zu vereinfachen[2]), fand die Technik schnell eine breite Anwendung im Biometriesektor. Mit dem 1993 gestarteten FERET-Programm (FacE REcognition Technology) des U.S. Army Research Laboratory wurden Gesichterkennungsverfahren mit einer Datenbank von 14,126 Bildern und 1,199 Personen erstmals einem Praxistest unterzogen und Forschungsergebnisse veröffentlicht.[3] In Deutschland wurden Gesichtserkennungssysteme durch das Bundesamt für Sicherheit in der Informationstechnik untersucht. [BSI03]

Pentland und Turk führten die Arbeiten von Sirovich und Kirby [SK87] fort, indem sie das komplexe Problem der Gesichterkennung auf ein zweidimensionales Problem abbildeten. Denn es ist leicht einzusehen: Gesichter unterliegen einer ähnlichen Konfiguration und werden von Gott nicht völlig zufällig verteilt. Folglich liegen Gesichter in einem niedrigdimensionalem Raum, dem *Face-space*[4] und werden durch einen kleinen Satz an charakteristischen Merkmalen, den Eigenvektoren, beschrieben.

Dieser Beleg beschreibt die Motivation hinter Eigenfaces und erklärt den Kern des Algorithmus: die Hauptkomponentenanalyse. Die Umsetzung eines Prototypen erfolgte auf Basis von OpenCV und die Leistungsfähigkeit wurde mit der AT&T Gesichtsdatenbank ermittelt. Ein abschliessendes Fazit beschreibt die Probleme der Eigenfaces und gibt einen Einblick in den heutigen Stand der Forschung.

2 Definitionen

Dieser Abschnitt schafft die mathematischen Grundlagen für ein Verständnis der Hauptkomponentenanalyse und folglich auch der Eigenfaces.

2.1 Durchschnitt, Standardabweichung und Varianz

Eine der wichtigsten statistischen Kenngrößen ist der Durchschnitt einer Stichprobe:

$$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}$$

jedoch ist der Durchschnitt alleine kein gutes Maß um einen gegebenen Datensatz zu charakterisieren. Man betrachte die folgenden Datensätze:

$$A = [\,0,\ 0,\ 10,\ 10\,],\ B = [\,5,\ 5,\ 5,\ 5\,]$$

Zwar ist $\overline{A} = \overline{B}$, daß aber beide Datensätze gleich wären würde niemand behaupten. Worin sich beide Datensätze unterscheiden ist die Streuung der Daten. Die durchschnittliche Distanz jedes Wertes zum Durchschnitt der Stichprobe ist die Standardabweichung s:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \overline{X})^2}{(n-1)}}$$

Auf das gegebene Beispiel angewendet würde die Standardabweichung:

$$s(A) = 7.071,\ s(B) = 0$$

ergeben. Eng verknüpft mit der Standardabweichung ist die Varianz. Die Varianz ist die quadrierte Standardabweichung s^2 und wird beispielsweise im t-Test oder den Kovarianzen verwendet:

$$s^2 = \frac{\sum_{i=1}^{n}(X_i - \overline{X})^2}{(n-1)}$$

Um das Beispiel zu vervollständigen ergibt sich:

$$s^2(A) = 33{,}3,\ s^2(B) = 0$$

[1] http://www.bundesdruckerei.de/de/service/service_buerger/buerger_persdok/persdok_epassMstr.html

[2] Patent #5,164,992: http://www.freepatentsonline.com/5164992.html

[3] http://www.frvt.org/feret/default.htm

[4] Gesichtsraum

Standardabweichung und Varianz stellen ein Maß dar, um eine Aussage über eine *eindimensionale* Stichprobe zu treffen.

2.2 Kovarianz und Kovarianzmatrix

Viele statistische Zusammenhänge drücken sich erst *zwischen* mehreren Dimensionen aus, die Varianz muss also erweitert werden. Die Kovarianz ist eine Maßzahl für den linearen Zusammenhang zweier statistischer Zufallsvariablen:

$$Cov(X,Y) = \frac{\sum_{i=0}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{(n-1)}$$

und ist eine Verallgemeinerung der Varianz, denn es gilt

$$Var(X,X) = Cov(X,X)$$

Die Kovarianz macht eine Aussage über die *Richtung des Zusammenhangs*, nicht über die Stärke des Zusammenhangs (zur Normierung müsste der Korrelationskoeffizient gebildet werden):

- **Positive Werte** bedeuten, daß beide Dimensionen einen gleichsinnigen linearen Zusammenhang aufweisen, d.h. beide Dimensionen wachsen gleichermassen. Man spricht von einer positiven Korrelation.
- **Negative Werte** treten bei einem gegensinnig linearen Zusammenhang auf, d.h. eine Dimension wächst beispielsweise während die andere fällt. Man spricht von einer negativen Korrelation.
- **0**,bedeutet es besteht zwischen X und Y kein linearer Zusammenhang. Man sagt beide sind nicht miteinander korreliert.

Für n dimensionale Daten lassen sich insgesamt $\frac{n!}{(n-2)!*2}$ Kovarianzen unter den Dimensionen berechnen, ein praktischer Weg um alle Kovarianzen zu bestimmen ist die Kovarianzmatrix aufzustellen, als:

$$C^{N\times N} = (c_{i,j} = Cov(Dim_i, Dim_j))$$

In einem Beispiel für 3 Dimensionen x,y,z ergibt sich so die Kovarianzmatrix C:

$$C = \begin{pmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{pmatrix}$$

Diese Matrix ist zwangsläufig *symmetrisch*, da $Cov(X,Y) = Cov(Y,X)$ durch die Kommutativität der Multiplikation in der Formel der Kovarianz gegeben ist (in der Diagonalen stehen, wie im Beispiel zu sehen ist, die Varianzen der Dimensionen). Durch die Symmetrie der Kovarianzmatrix ist:

$$c_{i,j} = c_{j,i}$$

was gleichbedeutend ist mit der Aussage, daß die Kovarianzmatrix C gleich der Transponierten Kovarianzmatrix C^T ist. Es hält die Aussage:

$$C = C^T$$

2.3 Eigenwerte und Eigenvektoren

Ein Eigenvektor bezeichnet in der linearen Algebra einen vom Nullvektor verschiedenen Vektor, dessen Richtung durch die Abbildung nicht geändert wird. Ein Eigenvektor unterliegt somit nur einer gewissen Streckung der als Eigenwert bezeichnet wird. Das *spezielle Eigenwertproblem*, welches auch in diesem Beleg Anwendung findet (durch die thematisierte *Symmetrie der Kovarianzmatrix*), bezieht sich auf lineare Abbildungen im endlichdimensionalen Vektorraums von symmetrischen Matrizen.

Abb. 1. Durchschnittsgesicht

2.4 Hauptkomponentenanalyse

Ein häufig auftretendes Problem in der statistischen Mustererkennung ist die Auswahl relevanter Merkmale aus einem Datensatz. Man sagt, diese *Merkmalsselektion* transformiert die Daten vom Datenbereich in den Merkmalsbereich und kann dabei die gleiche oder eine geringere Dimensionalität als die Originaldaten aufweisen. Es ist jedoch im Allgemeinen wünschenswert, die Dimensionalität der Daten zu reduzieren. Konkret auf die vorliegende Problemstellung kann ein Bild der Größe $N \times N$, durch einen Vektor im N^2-dimensionalen Raum beschrieben werden. Ein Bild mit 128×128 Pixel würde bereits einen Punkt im $16,384$-dimensionalen Raum darstellen.

Bei Dimensionsreduktion allgemein, ist es notwendig Merkmale zu erhalten, die den Datensatz am besten beschreiben. Wird der Vektor an einer beliebigen Position beschnitten, so ist der durchschnittliche Fehler durch diese Kompression definiert als die Summe der Varianzen der von $\vec{x}$ eliminierten Elemente. Die Frage ist, ob eine Transformationsmatrix T existiert, so daß der durchschnittliche Fehler durch die Transformation $T\vec{x}$ minimal ist, diese Matrix ist optimal.

Mathematisch ausgedrückt suchen wir die optimale Transformationsmatrix T, die einen m-dimensionalen Vektor $\vec{x}$ in einen l-dimensionalen Vektorraum mit $l < m$ transformiert.[Hay08]

Es sei:

$\vec{x}$ ...eine m-dimensionale Stichprobe

Gesucht ist:

$$T\vec{x} = \vec{x}_{neu}$$

T ...die optimale $m \times m$ Transformationsmatrix, so daß die Varianz an den Hauptachsen maximiert ist und der Mean-Square-Error (MSE)[5] minimert ist.

Wichtig ist die Normalisierung des Eingabevektors $\vec{x} = \vec{x} - mean(\vec{x})$, d.h.:

$$E[\vec{x}] = 0$$

E ...ist die Funktion für den Erwartungswert

Sei $\vec{q}$ ein m-dimensionaler Einheitsvektor, so gilt:

$$\|\vec{q}\| = 1, \text{ daraus folgt unmittelbar } \|\vec{q}\| = (\vec{q}^T\vec{q})$$

(Denn bei $1 = \sqrt{1}$ ist 1 im reellwertigen Bereich die einzige Lösung die diese Beschränkung erfüllt.) Die Projektion A von $\vec{x}$ auf $\vec{q}$ ist nötig um Vektoren im Vektorraum aufeinander abzubilden, es gilt:

$$A = \vec{x}^T\vec{q} = \vec{q}^T\vec{x}$$

Erwatungswert und Varianz im Vektorraum, sind gegeben

[5] Durchschnittlicher Fehler

durch:

$$\begin{aligned} E[A] &= \vec{q}^T E[\vec{x}] = 0 \\ \sigma^2 &= E[A^2] = E[(\vec{q}^T\vec{x})(\vec{x}^T\vec{q})] = \vec{q}^T E[\vec{x}\,\vec{x}^T]\vec{q} = \vec{q}^T C\vec{q} \\ C &= E[\vec{x}\,\vec{x}^T] \end{aligned} \quad (1)$$

C ist die Korrelationsmatrix, wir wissen über C:

- Symmetrisch
- $C^T = C \Rightarrow \vec{a}^T C\vec{b} = \vec{b}^T C\vec{a}$

Aus der Gleichung 1 erkennen wir, dass die Varianz σ^2 der Projektion A eine Funktion des Vektors $\vec{q}$ ist. Man kann also folgende Funktion $\Phi(\vec{q})$ definieren:

$$\Phi(\vec{q}) = \sigma^2 = \vec{q}^T C\vec{q} \quad (2)$$

Es muss so ein Vektor $\vec{q}$ gefunden werden, dass die Funktion $\Phi(\vec{q})$ ein Exrema erreicht. Gesucht ist ein globales Minimum. Das bedeutet jede kleine Änderung von $\vec{q}$ um $\delta\vec{q}$ führt zu einem Anstieg es Funktionswertes $\Phi(\vec{q})$.

$$\Phi(\vec{q}+\delta\vec{q}) = \Phi(\vec{q})$$

Aus der Gleichung 2 von $\Phi(\vec{q})$ ergibt sich für $\Phi(\vec{q}+\delta\vec{q})$:

$$\begin{aligned} \Phi(\vec{q}+\delta\vec{q}) &= (\vec{q}+\delta\vec{q})^T C(\vec{q}+\delta\vec{q}) \\ &= \underbrace{\vec{q}^T C\vec{q}}_{=\Phi(\vec{q})} + 2(\delta\vec{q})^T C\vec{q} + \underbrace{(\delta\vec{q})^T C\vec{q}}_{\approx 0} \\ &= \Phi(\vec{q}) + 2(\delta\vec{q})^T C\vec{q} \end{aligned}$$

weil $(\delta\vec{q})^T C\vec{q} \approx 0$ und $\vec{q}^T C\vec{q} = \Phi(\vec{q})$ kann die Gleichung weiter vereinfacht werden:

$$(\delta\vec{q})^T C\vec{q} = 0 \quad (3)$$

Folgende Bedingung muss eingehalten werden damit $\vec{q}+\delta\vec{q}$ ein Einheitsvektor ist:

$$\|\vec{q}+\delta\vec{q}\| = 1$$

oder äquivalent:

$$\begin{aligned} (\vec{q}+\delta\vec{q})^T(\vec{q}+\delta\vec{q}) &= 1 \\ \underbrace{\vec{q}^T\vec{q}}_{\approx 1} + 2(\delta\vec{q})^T\vec{q} + \underbrace{(\delta\vec{q})^T\delta\vec{q}}_{\approx 0} &= 1 \end{aligned}$$

daraus folgt:

$$(\delta\vec{q})^T\vec{q} = 0 \quad (4)$$

Um die Gleichungen 3 und 4 gleichzusetzen wird ein Skalierungsfaktor λ eingeführt:

$$\begin{aligned} (\delta\vec{q})^T C\vec{q} - \lambda(\delta\vec{q})^T\vec{q} &= 0 \\ (\delta\vec{q})^T(C\vec{q} - \lambda\vec{q}) &= 0 \end{aligned} \quad (5)$$

weil $(\delta\vec{q})^T \neq 0$ folgt aus Gleichung 5:

$$\begin{aligned} C\vec{q} - \lambda\vec{q} &= 0 \\ C\vec{q} &= \lambda\vec{q} \end{aligned} \quad (6)$$

Die Gleichung 6 ist in der linearen Algebra als Eigenwertproblem bekannt und wird weiter unten beschrieben.

2.4.1 *Eigenwertproblem*

Eine Matrix heisst orthogonal, wenn die Transponierte Matrix gleich der Inversen ist,

$$A^T A = AA^T = I \quad (7)$$

Eine Matrix heisst normal, wenn gilt:

$$AA^T = A^T A \quad (8)$$

Eine reelle symmetrische Matrix ist als normale Matrix diagonalisierbar mit einem vollen Satz Eigenvektoren, wobei die Eigenvektoren zu den Eigenwerten orthogonal sind. Eine normale Matrix ist über einem reellen Vektorraum selbstadjungiert und kann aus diesem Grund ausschliesslich reelle Eigenwerte enthalten.

Zur Lösung des Eigenwertproblems muss folgende Gleichung aufgelöst werden:

$$Ax = \lambda x \quad (9)$$

Dabei hat ein homogenes lineares Gleichungssystem für x genau dann eine nichttriviale Lösung wenn $A - \lambda I$ singulär ist, also:

$$det(A - \lambda I) = 0 \quad (10)$$

dabei bezeichnet $det(M)$ die Determinante der Matrix M. Durch die Entwicklung von $det(A - \lambda I) = 0$:

$$det(A - \lambda I) = a_0 + a_1\lambda + \ldots + a_{n-1}\lambda^{n-1} + (-\lambda)^n = p_n\lambda \quad (11)$$

ist leicht einzusehen, daß die Eigenwertbedinung eine Polynomgleichung vom Grade n ist. Jede Matrix A hat somit genau n Eigenwerte $\lambda_1, \ldots, \lambda_n$. Für die Lösung des speziellen Eigenwertproblems gibt es zwei unterschiedliche Verfahrensklassen:

1. Transformationsverfahren
   - Jacobi-Verfahren
   - Householder-Tridiagonalisierung
   - QR-Algorithmus
2. Iterationsverfahren
   - Vektoriteration
   - Railey-Ritz-Algorithmus

Durch die Formulierung des kleinsten Fehler ist einzusehen, daß die Achsen die einen Datensatz am besten beschreiben durch die Eigenvektoren mit den größen Eigenwerten beschrieben werden.

2.5 Beispiel: Hauptkomponentenanalyse

2.5.1 *In Octave*

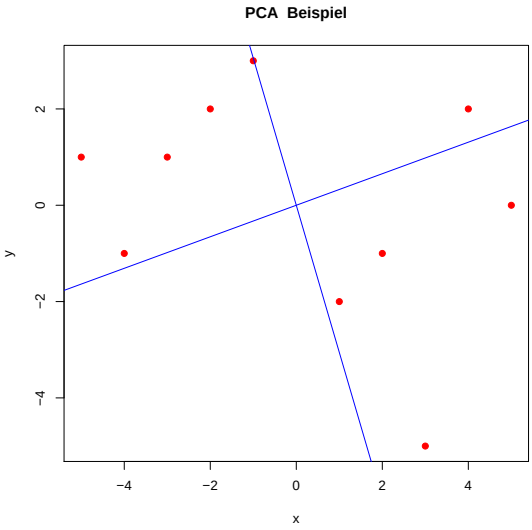
GNU Octave[6], die freie und quelloffene Alternative zu Matlab, bringt Solver zum Lösen der Eigenwertgleichung mit. Beispielsweise mit der Funktion `eig(R)`, wobei `R` eine Matrix sein muss.

```
octave:1> x = [-5 -4 -3 -2 -1 1 2 3 4 5];
octave:2> y = [ 7  5  7  8  9 4 5 1 8 6];
octave:3> x1 = x - mean(x);
octave:4> y1 = y - mean(y);
octave:5> R = [ sum(x1.*x1)/9 sum(x1.*y1)/9; sum(y1.*x1)
      /9 sum(y1.*y1)/9];
octave:6> [V,D] = eig(R);
octave:7> C = (-6:1:6)'
octave:8> z = [x1; y1]'* V(:,2);
octave:9> V,D,z
V =

   -0.31112   -0.95037
   -0.95037    0.31112
```

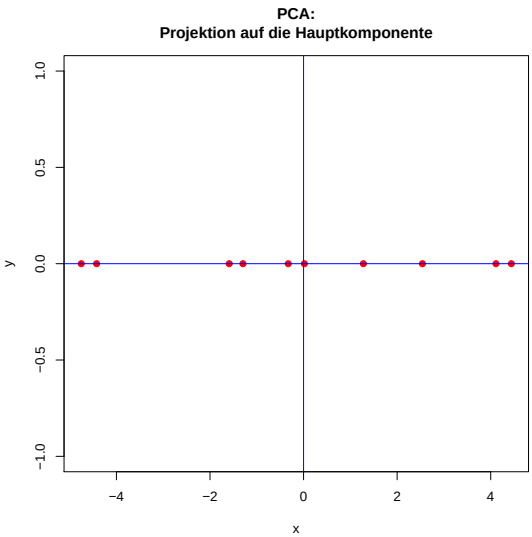
[6] http://www.gnu.org/software/octave/

```
D =

Diagonal Matrix

    4.7553         0
         0    13.0225

z =

    5.0630
    3.4904
    3.1622
    2.5230
    1.8837
   -1.5726
   -2.2119
   -4.4067
   -3.1792
   -4.7518
```

Es ergeben sich also zwei Eigenvektoren, die das neue Koordinatensystem aufspannen:



Die Projektion auf die Hauptkomponente ergibt eine eindimensionale Abbildung:



3 Eigenfaces

Da Gesichter einer ähnlichen Konfiguration entsprechen sind sie keinesfalls zufällig im (hochdimensionalen) Bildraum verteilt, sondern liegen in einem niedrigdimensionaleren Unterraum. Die Idee der Hauptkomponentenanalyse, in der Bildverarbeitung oft auch Karhunen-Loeve-Transformation genannt, ist es genau die Vektoren zu finden, welche die Verteilung der Bilder am besten beschreiben. Diese Vektoren spannen den Unterraum der Bilder auf, welchen wir im folgenden *Gesichtsraum* nennen. Da diese Vektoren die Eigenvektoren der originalen Gesichter sind, werden sie *Eigenfaces* genannt.

3.1 Berechnung der Eigenfaces

1. **Normalisierung des Bildes**: Die Hauptkomponentenanalyse erzwingt eine Normalisierung der Eingabedaten. Die Summe aller Pixel eines Bildes Γ_i wird durch die Gesamtanzahl der Pixel geteilt und anschliessend von dessen originalen Bildvektor abgezogen.

$$norm(\Gamma_i) = \Gamma_i - \left[\frac{1}{N*N}\sum_{p=1}^{N*N}\Gamma_{ip}\right] \tag{12}$$

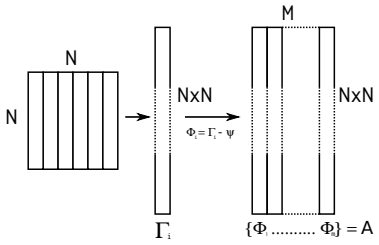
2. **Berechnung des Durchschnittsgesichts**: Aus allen M normalisierten Gesichtern in der Gesichtsdatenbank wird das Durchschnittsgesicht gebildet. Ein Beispiel für ein Durchschnittsgesicht zeigt ein 2.4. Das Durchschnittsgesicht ist definiert als:

$$\Psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n \tag{13}$$

3. **Normierung der Gesichter** um einen Datensatz mit einem Durchschnitt von 0 zu erzeugen. Es wird die Abweichung jedes Gesichtes vom Durchschnittsgesicht berechnet als:

$$\Phi_i = \Gamma_i - \Psi \tag{14}$$

4. **Aufstellen der Gesichtsmatrix aller Bilder Φ_i**:



5. **Kovarianzmatrix aufstellen** aus den normierten Gesichtern kann die Kovarianzmatrix aufgestellt werden als:

$$C = \frac{1}{M}\Sigma_{n=1}^{M}\Phi\Phi^T = AA^T = A^TA \tag{15}$$

6. **Eigenwerte und Eigenvektoren von C lösen**: die Hauptkomponenten sind die Eigenvektoren mit den größten Eigenwerten, wie dem Beweis ab Formel 3 zu entnehmen ist.

7. **Abspeichern der Eigenvektoren**: jeder Eigenvektor ist ein Bildvektor der Länge $N \times N$. Dieser kann somit ebenfalls als Bild dargestellt werden, und es ergeben sich die Eigenfaces. Die ersten 3 Eigenfaces der beiden Autoren zeigt Abbildung 2. Meist werden nur 10-15 Eigenfaces abgespeichert, da diese

Abb. 2. Die ersten 3 Eigenfaces der Autoren (Der Eigenautor).

durch die größte Varianz in den Trainingsdaten charakterisiert sind.

3.2 Klassifikation neuer Bilder

Die Klassifikation unbekannter Bilder erfolgt in den Schritten:

1. **Normalisierung des Testbildes** mit Formel 12.
2. **Projektion des Testbildes** in den Gesichtsraum:

$$\omega_k = v_k^T(\Gamma - \Psi), k \in \{1 \dots M'\}$$

, wobei v der k-te Eigenvektor (Eigenface) und M' die Anzahl der Eigenvektoren (Eigenfaces) ist.

Dies ergibt den M'-dimensionalen Gewichtsvektor:

$$\Omega^T = [\omega_1, \omega_2, \dots, \omega_{M'}]$$

, wobei jedes ω den Anteil des Eigenface v_k an Γ darstellt.

3. **Nächsten Nachbar Suche**: dabei wird im Prototypen die Euklidische Distanz verwendet und Ω wird mit allen, in den Gesichtsraum projezierten, Gesichtern Ω_k der Trainingdatenbank verglichen.

$$\varepsilon_k = ||\Omega - \Omega_k||$$

Um nicht alles als Person zu identifizieren ist die Angabe einer iterativ bestimmten Schwelle θ_ε notwendig. Das kleinste ε_k das die folgende Formel erfüllt, ist schliesslich das gesuchte Gesicht:

$$\varepsilon_k < \theta_\varepsilon$$

4 Implementierung

Ein Prototyp wurden in C++ entwickelt und verwendet die Bildverarbeitungsbibliothek OpenCV. Abbildung 3 zeigt den Prototypen, welcher sowohl einen Live-Modus zur Gesichtserkennung in Echtzeit, als auch die Klassifikation von gegebenen Datensätzen erlaubt.

Trainings- und Testdatensätze die evaluiert werden sollen können über XML (eXtensible Markup Language) eingepflegt werden, dabei dürfen sich beide Mengen nicht überschneiden. Direkt im Framework wurde keine Kreuzvalidierung implementiert, sondern mit Hilfe von Skripten realisiert.

4.1 OpenCV

OpenCV (*Open Computer Vision*) ist eine quelloffene, unter BSD-Lizenz stehende, Programmierbibliothek für maschinelles Sehen, wobei der Fokus auf Echtzeit-Bildverarbeitung liegt. 1999 von Intel Research initiiert, wurde die erste Alphaversion der Bibliothek im Jahre 2000 veröffentlicht und erreichte 2006 die Versionsnummer 1.0. Heute wird die Bibliothek aktiv von Willow Garage[7], einem in Menlo Park (California) ansässigen Unternehmen

[7] http://www.willowgarage.com

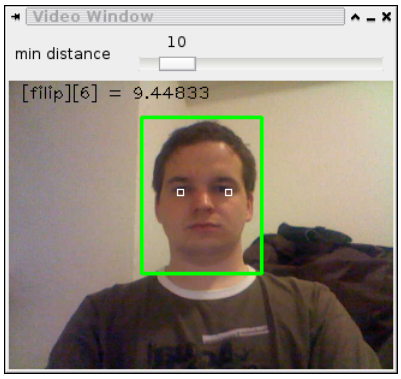


Abb. 3. Live Demo des Prototypen.

im Bereich der Robotik, gepflegt und hat die Versionsnummer 2.0 erreicht.

Die Bibliothek, optimiert für Intelprozessoren, wurde in C geschrieben und wird in C/C++ programmiert. Es existieren jedoch Bindings für Sprachen wie C# oder Python.

Die Machine Learning Library von OpenCV enthält:

- Boosting
- Decision tree learning
- Expectation-maximization algorithm
- k-nearest neighbor algorithm
- Naive Bayes classifier
- Artificial neural networks
- Random forest
- Support vector machine (SVM)

Durch die in OpenCV 2.0 beta eingeführte Verwendung[8] von LAPACK/BLAS (Linear Algebra PACKage/Basic Linear Algebra Subprograms) für Berechnungen der linearen Algebra, stehen der Bibliothek erprobte Solver zur Verfügung, wie auch der in diesem Beleg verwendete Eigenwertsolver.

5 Experiment

5.1 Vorhandene Datensätze

Im folgenden werden 3 häufig verwendete Datasets im Bereich der Gesichtserkennung diskutiert.

Yale Facedatabase A besteht aus 165 Bildern von 15 Personen. Es gibt 11 Bilder von jeder Person, wobei jedes Bild sich unterscheidet hinsichtlich Lichtquelle (links, zentriert, rechts), Brille (mit, ohne), Gesichtsausdruck (freudig, traurig, normal, verschlafen, überrascht, zwinkernd). Die Graustufenbilder, 1997 aufgenommen, haben die Größe 320×243 und haben eine große Varianz, wie in Abbildung 5 zu sehen ist.

Yale Facedatabase B besteht aus 10 Personen unter 576 verschiedenen Perspektiven. Die 5760 Graustufenbilder der Größe 640×480 Pixel messen komprimiert rund 1 Giagbyte.

AT&T Facedatabase besteht aus je 10 Bildern von 40 verschiedenen Personen. Die von 1992 bis 1994 für ein Forschungsprojekt entstandenen Aufnahmen erfolgten hinter einem homogenen Hintergrund und sind zum größten Teil frontale Gesichtsaufnahmen mit, im Vergleich zu der Yale Facedatabase, weniger perspektivischen Verzerrungen und Gesichtsemotionen.

[8] http://opencv.willowgarage.com/wiki/OpenCV%20Change%20Logs

Abb. 4. AT&T Gesichtsdatenbank.

Abb. 5. Person in der Yale Facedatabase A.

Die Größe der 400 Graustufenbilder beträgt 92×112 Pixel, Abbildung 4 zeigt zwei Personen aus dem Datensatz.

5.2 Auswahl des Datensatzes

Aktuelle Veröffentlichungen im Bereich der Gesichtserkennung legen den Fokus auf die maschinelle Erkennung von Emotion und suchen robuste, statistische Verfahren für variierende Lichtbedingungen und wechselnde Perspektiven.[9]

Da für diesen Beleg die Klassifikationsfähigkeit des entstandenen Prototypen (und in direkter Konsequenz natürlich der Eigenfaces Methode) untersucht werden soll, werden möglichst frontale Aufnahmen eines Gesichtes bei homogener Beleuchtung und homogenem Hintergrund benötigt. Die von AT&T bereitgestellte Datenbank erfüllt diese Charakteristika voll und wurde aus diesem Grund als Datensatz ausgewählt.

Bei der Yale Facedatabase A ist die Anzahl von 15 Individuen zwar gering, jedoch haben die Bilder eine hohe Varianz hinsichtlich der Lichtbedingungen. Die Perspektive ist eine frontale Ansicht, womit sich diese Bilder zum Test des entstandenen Prototypen eignen.

Die Yale Facedatabase B hat den Fokus auf verschiedene Lichtbedingungen und Perspektiven gelegt. Mit einer komprimierten Größe von 1GB, wechselnden Hintergründen und nicht zurechtgeschnitten Aufnahmen, würde es einen hohen Arbeitsaufwand bedeuten die Bilder für den Prototypen entsprechend vorzubereiten. Hinzukommen die wechselnden Lichtbedingungen und teilweise sehr dunklen Aufnahmen und da Eigenfaces sensibel auf variierende Lichteinflüsse reagieren [Bel97] wurde gegen die Yale Facedatabase B entschieden.

5.3 Ergebnisse

Die Klassifikationsfähigkeit hängt von 2 Faktoren ab:

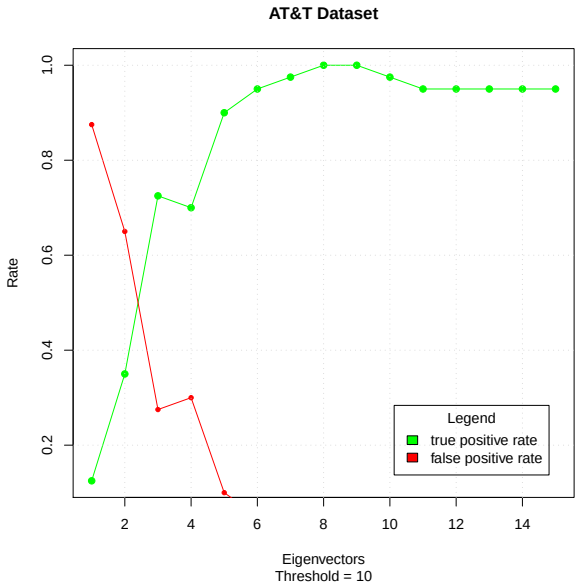
1. **Eigenvektoren**: Je mehr Eigenvektoren verwendet werden, desto besser lässt sich ein Bild rekonstruieren. Ab einer gewissen Schwelle, die auch in den Validationsergebnissen ersichtlich ist, verliert man einen gewissen Grad an Abstraktion und erhält schlechtere Validationsergebnisse. In der Literatur werden meist 10-13 Eigenvektoren als *magische* Zahl angenommen, die folgenden Eigenvektoren enthalten nur noch wenig Varianz und sind in der Entscheidung nicht hilfreich.
2. **Schwellwert**: Der Schwellwert bestimmt darüber ob ein Bild ein Gesicht ist oder nicht. Ein zu hoher Schwellwert führt zu einer höheren Anzahl an Falschklassifikationen, ein zu geringer Wert ist dafür verantwortlich, dass keine Entscheidung hinsichtlich eines Individuums getroffen wird.

Zwischen diesen beiden Faktoren ist auszuwählen. Während 10-13 Eigenvektoren meist zu guten Ergebnissen führen, ist die Schwelle iterativ zu bestimmen. Eine iterative Optimierung des Schwellwertes ist nicht in den Prototypen integriert, sondern durch einen Parameter realisiert.

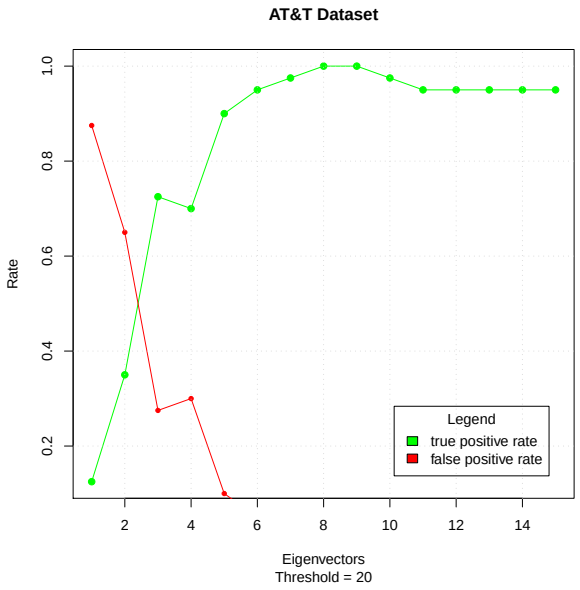
5.3.1 AT&T Database

Ergebnis Die PCA erreicht auf der AT&T Database sehr gute Ergebnisse. Verwendet der Prototyp das gesamte Dataset, klassifiziert der Prototyp in unseren Tests über 90% korrekt der Gesichter korrekt. Dies deckt sich mit anderen veröffentlichten Ergebnissen [KJK02]. Wird die Trainingsmenge beschnitten, so sinkt die Klassifikationsfähigkeit der Eigenfaces.

Klassifikationsergebnis bei Threshold 10



Klassifikationsergebnis bei Threshold 20



5.3.2 Yale Facedatabase A

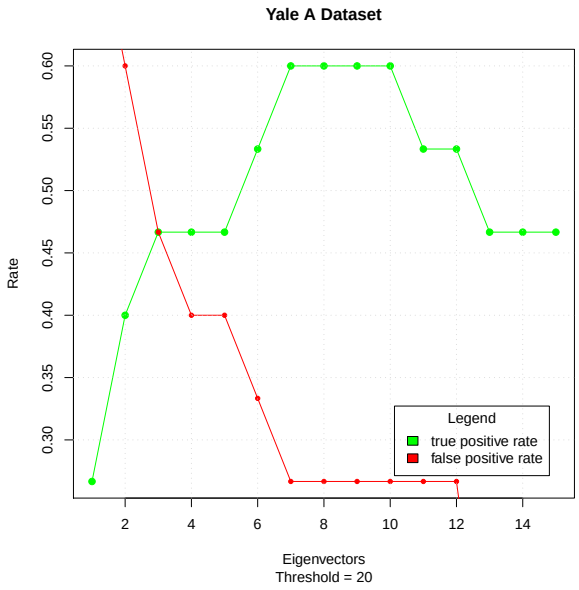
Ergebnis Auf der Yale Face Database A liegt die Klassifikationsrate mit 15 Eigenvektoren bei rund 60% und Veränderungen am Schwellwert führen zu keiner Verbesserung des Ergebnis. Dieses Resultat deckt sich mit Ergebnissen anderer

[9] Ohne konkret zu zitieren. Verweis auf aktuelle Forschung: http://www.face-rec.org/new-papers

Veröffentlichungen [GA04] und zeigt den Einfluss von Lichtverhältnissen und Perspektive auf die Klassifikationsfähigkeit der linearen Hauptkomponentenanalyse.

Um die Leistungsfähigkeit des Klassifikators zu steigern ist eine Vorverarbeitung der Datenbank notwendig um Gesichter zu skalieren, einen homogenen Hintergrund zu schaffen und Einflüsse durch Licht und Perspektive zu minimieren.

Klassifikationsergebnis bei Threshold 20



Klassifikationsergebnisse

Eigenfaces

Im folgenden sind die Eigenfaces einer Person mit einer unterschiedlichen Anzahl an Eigenvektoren dargestellt. Dies verleiht einen visuellen Eindruck über mögliche Fehlklassifikation, im folgenden kann das Bild aber bereits ab 8 Eigenvektoren hinreichend gut rekonstruiert werden.

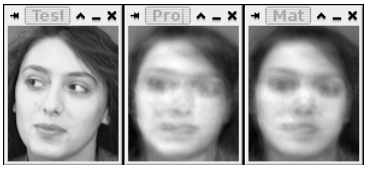
Jedes Bild zeigt: Eingabebild, Projektion des Eingabebilds in den Gesichtsraum, Projeziertes Gesicht der Trainingsdatenbank mit geringstem Abstand zum Eingangsbild.

- **1 Eigenvektor:** Einzige Falschklassifikation.

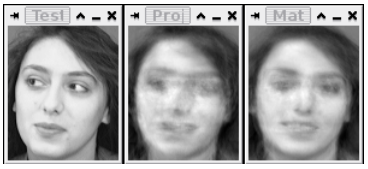
- **8 Eigenvektoren:**

- **15 Eigenvektoren:**

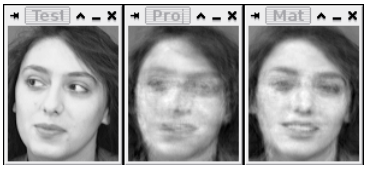
- **20 Eigenvektoren:**



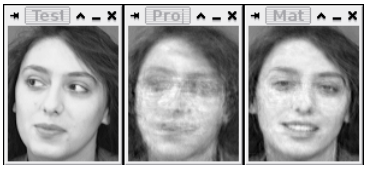
- **50 Eigenvektoren:**



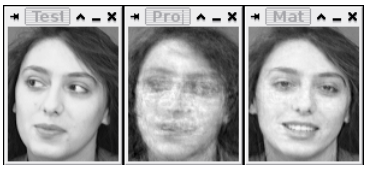
- **100 Eigenvektoren:**



- **200 Eigenvektoren:**



- **300 Eigenvektoren:**



Facespace

Mit Hilfe der ersten 3 Eigenvektoren können die Trainingsbilder im dreidimensionalen Raum als Streudiagramm dargestellt werden. Dabei werden die verschiedenen Personen durch unterschiedliche Farben dargestellt. In Abbildung 6 sind der Übersichtlichkeit halber nur 5 Personen aus der AT&T Database in den Facespace projeziert.

Es zeigt sich, daß sich die 5 Klassen trennen und eine Klassifikation mit Euklidischer Distanz möglich wird.[10]

6 Zusammenfassung

6.1 Fazit

In diesem Beleg wurde die Hauptkomponentenanalyse untersucht und in einem Prototypen für Gesichtserkennung implementiert. Die Gesichtserkennung erfolgte auf Basis des in [TP91] vorgeschlagenen Eigenfaces-Algorithmus, die Klassifikation ist eine Nearest Neighbor Klassifikation.

[10] 1-Nearest-Neighbor.

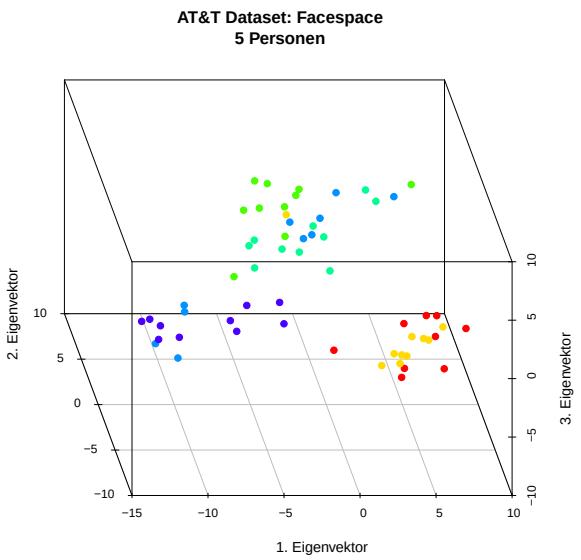


Abb. 6. Facespace des AT&T Dataset mit 5 Personen.

Mit Hilfe der zwei öffentlichen Datensätze *AT&T Database* und *Yale Face Database A* konnte der Prototyp erfolgreich getestet werden und das Verhalten der Eigenfaces nachvollzogen werden. Diese Ergebnisse decken sich mit denen anderer Veröffentlichungen.

Zur Visualisierung wurde eine Teilmenge des Facespace der *AT&T Database* dargestellt und GNU Octave zur Ermittlung der Hauptkomponenten demonstriert.

6.2 Ausblick

Der Eigenfaces Algorithmus verwendet die Hauptkomponentenanalyse und ist kein Klassifikationsverfahren, denn die Hauptkomponentenanalyse kennt keine Klassenzugehörigkeit sondern versucht die Varianz über allen Daten zu maximieren. Dies führt wie in Veröffentlichungen beschrieben [Bel97], [KJK02] zu schlechten Klassifikationsergebnissen bei Störeinflüssen im Bildbereich (Perspektive, Lichtverhältnisse).

Eine Möglichkeit die Klassenzugehörigkeit zu berücksichtigen ist die Lineare Diskriminanten Analyse (LDA) mit Fisher's Linearer Diskriminante (FDA). Dabei wird versucht die Intraklassenvarianz zu minimieren und Interklassenvarianz zu maximieren. In Anlehnung an die Eigenfaces wird diese Methodik in der Gesichtserkennung als Fisherfaces bezeichnet. [Bel97]

Neuere Veröffentlichungen zeigen, daß mit einer Erweiterung der PCA um nicht-lineare Methoden wie Kernel die Fehlerrate der PCA deutlich gesenkt werden kann. Eine Anwendung auf Gesichtserkennung erfolgte in [KJK02] mit Hilfe eines polynomiellen Kernels. Auf dem in diesem Beleg ebenfalls verwendeten AT&T Dataset[11] konnte eine Senkung der Falschklassifikationsrate von 10% (Lineare PCA) auf 2.5% (Kernel PCA) erreicht werden.

A Literatur

[Bel97] BELHUMERU, Hespanha João. Kriegmann D. Peter.: Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19 (1997), Nr. 7, S. 711–720

[BSI03] BSI, Bundesamt für Sicherheit in der I.: *BioFace: Vergleichende Untersuchung von Gesichtserkennungssystemen*. 2003. – Öffentlicher Abschlussbericht BioFace I und II

[GA04] GOTTUMUKKAL, Rajkiran ; ASARI, Vijayan K.: An improved face recognition technique based on modular PCA approach. In: *Pattern Recognition Letters* 25 (2004), Nr. 4, S. 429 – 436. http://dx.doi.org/DOI:10.1016/j.patrec.2003.11.005. – DOI DOI: 10.1016/j.patrec.2003.11.005. – ISSN 0167–8655

[Hay08] HAYKIN, Simon: *Neural Networks and Learning Machines (3rd Edition)*. 3. Prentice Hall, 2008 http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0131471392. – ISBN 0131471392

[KJK02] KIM, K.I. ; JUNG, K.C. ; KIM, H.J.: Face recognition using kernel principal component analysis. 9 (2002), February, Nr. 2, S. 40–42

[SK87] SIROVICH, L. ; KIRBY, M.: Low-dimensional procedure for the characterization of human faces. In: *J. Opt. Soc. Am. A* 4 (1987), Nr. 3, 519–524. http://josaa.osa.org/abstract.cfm?URI=josaa-4-3-519

[TP91] TURK, M. ; PENTLAND, A.: Eigenfaces for recognition. In: *Journal of Cognitive Neuroscience* 3 (1991), S. 71–86

[11] In Veröffentlichungen auch als ORL (Olivetti Research Library) Dataset bezeichnet, http://www.face-rec.org/databases/ nennt es jedoch: *AT&T "The Database of Faces" (formerly "The ORL Database of Faces")*