Seminarausarbeitung zum Thema

Extrahierung und Kodierung von Gesichtsmodellen

Matthias Weber, Dezember 2002

Seminar 3D-Rekonstruktion, Wintersemester 02/03
Lehrstuhl für Informatik IV
Universität Mannheim

Betreuer: Dirk Farin

0. Überblick

Die Übertragung von Videosequenzen ist eine Aufgabe, die für eine akzeptable Qualität sehr viel Bandbreite zwischen Sender und Empfänger benötigt.

Diese Arbeit stellt Methoden vor, mit denen eine Videosequenz, insbesondere Gesichter, als Modell kodiert wird, um dann nur die Änderungen dieses Modells möglichst effizient zu übertragen.

Das schliesst die Aufgaben mit ein, ein Gesicht zu lokalisieren, ein vordefiniertes Gesichtsmodell anhand von automatisch erkannten Merkmalspunkten daran anzupassen und die Animationsparameter des Modells zu komprimieren.

So kann eine visuelle Kommunikation auch über einen Kanal mit stark begrenzter Bandbreite möglich gemacht werden.

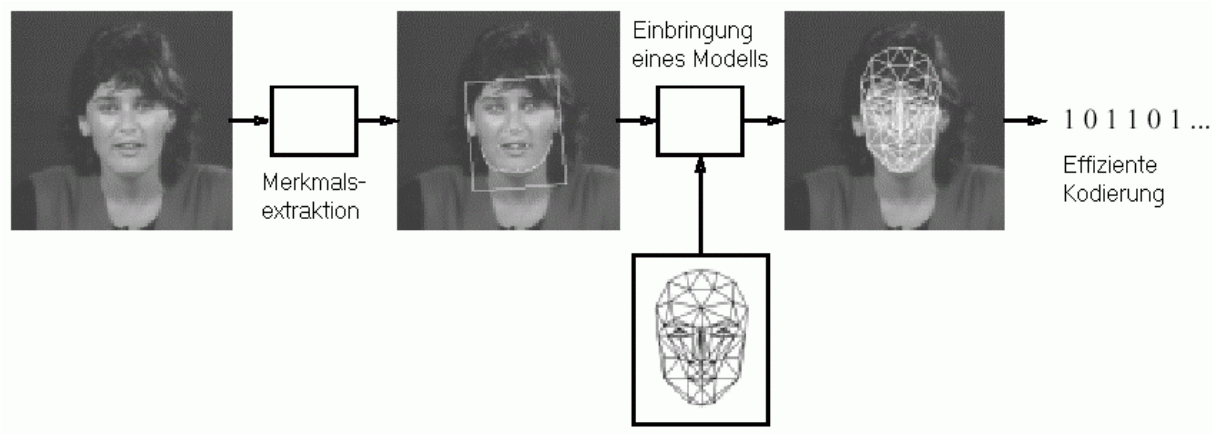

Abb. 1: Überblick

1. Einführung

1.1 Ein visuelles Kommunikationssystem

Der Weg, über die eine Nachricht vom Sender zum Empfänger geleitet wird, wird Kanal genannt. Dabei kann es sich um eine Telefonleitung oder einfach ein Blatt Papier handeln. Handelt es sich bei der Nachricht um ein Bild oder eine Abfolge von Bildern (Video), spricht man von einem visuellen Kommunikationssystem.

Die Kapazität eines Kanals ist natürlich begrenzt, deshalb wird in der Regel Datenkomprimierung eingesetzt. Verlustbehaftete Algorithmen vernachlässigen die Teile des Datenstromes, deren Fehlen am wenigsten auffällt.

Verbreitete digitale Kanäle mit geringer Bandbreite sind GSM, das normale Handynetz mit 8 kbit/s, welches für komprimierte Sprache entwickelt wurde, sowie ISDN mit 64 kbit/s.

Die Modellbasierte Bildkodierung ist für verlustbehaftete Komprimierung auf Kanälen mit sehr geringer Bitrate gedacht. Auf der Senderseite wird ein bekanntes Modell möglichst passend an das zu übertragende Bild angepasst, übertragen, und auf der Empfängerseite wieder zu einem Bild zusammengefügt.

1.2 Modellbasierte Kodierung

Die Grundidee der Modellbasierten Kodierung ist wie folgt: Auf der Senderseite wird das Bild einer Kamera mit Methoden der Computer Vision analysiert und das oder die relevanten Objekte, beispielsweise ein menschliches Gesicht, wird identifiziert. Es wird dann ein Modell davon erstellt, welches in der Regel aus einem Drahtgitter und einer darüberliegenden Textur besteht.

Die Übertragung des Modells wiederum erfolgt prädiktiv: Es wird nicht nicht in jedem Frame das komplette Modell neu übertragen, sondern nur die Veränderungen einzelner Knoten im Vergleich zum vorhergehenden Frame.

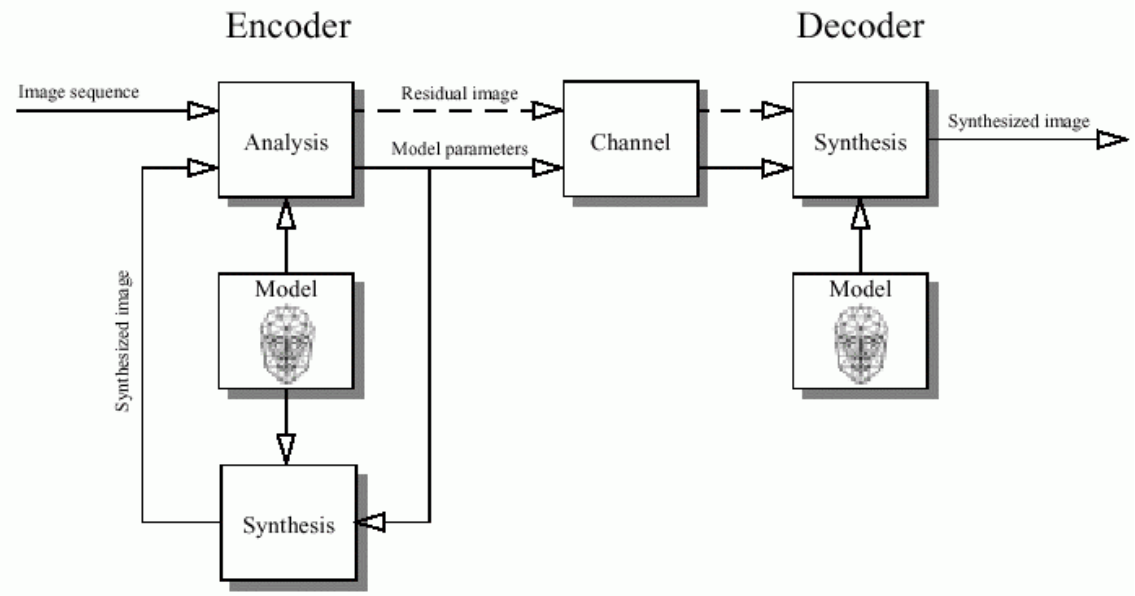

Abb. 2: Selbstkontrolle auf der Senderseite: Der Sender überprüft das ermittelte Modell durch einen Vergleich mit der originalen Bildsequenz

1.3 Gesichtsanimation mit MPEG-4

Der Standard MPEG-4 aus dem Jahre 1999 beschreibt unter anderem, wie Parameter für Gesichtsanimationen zu dekodieren sind.
Es sind 84 Merkmalspunkte des Gesichtes definiert, welche für die Form und Bewegung als entscheidend betrachtet werden. Diese heissen *Facial Definition Parameters* (FDPs).

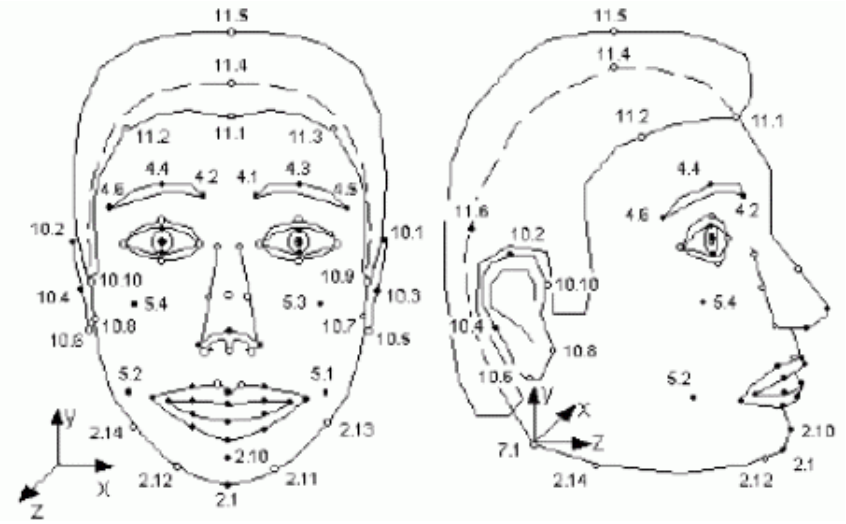

Abb. 3: Definierte Merkmalspunkte

Der Empfänger bekommt die Koordinaten dieser Punkte übergeben und transformiert sein eigenes Modell entsprechend. Das Modell des Empfängers kann unterschiedlich sein.

1.4 Überblick über die Arbeit

Dieses Kapitel lieferte eine Einführung in das Thema.
Kapitel 2 befasst sich mit Methoden, ein Gesicht zu lokalisieren.
Kapitel 3 stellt Methoden vor, Gesichtszüge zu extrahieren: Welche Form soll das angepasste Gesichtsmodell haben? Dazu werden Methoden der Optimierung angewendet.
Kapitel 4, Kodierung der Parameter des Gesichtsmodells, behandelt Ansätze für Algorithmen, die die Parameter des Gesichtsmodelles effizient komprimieren und übertragen.
Kapitel 5 diskutiert die Techniken, die in den anderen Kapiteln beschrieben wurden, und wie sie zu einem einfachen modellbasierten Kodierungssystem zusammengefügt werden können.

Anmerkung: Die Aufgaben der Decoderseite, also die Darstellung des übertragenen Modells, ist nicht Teil dieser Arbeit und wird hier nicht behandelt.

2. Gesichtsfindung

2.1 Einführung

Dieser Bereich behandelt zwei Aufgaben: Ist ein Gesicht im Bild enthalten, und wenn ja, wo und mit welcher Ausrichtung?

Diese Schritte stellen für das menschliche Auge kein Problem dar, haben sich für das Maschinensehen aber als schwierig herausgestellt. Unterschiedliche Beleuchtungen und Winkel können das selbe Gesicht unterschiedlich erscheinen lassen.

Das Ergebnis dieses Schrittes, also den Suchbereich für die Merkmalspunkte möglichst gut einzugrenzen, ist für die Extrahierung der Gesichtsmerkmale (siehe Kapitel 3) wichtig, um Fehler, wie irrtümliche Erkennungen, zu vermeiden.

2.2 Schnelle Algorithmen für Gesichtsentdeckung

Eine einfache und schnelle Methode ist die Suche nach einem elliptischen Objekt im Bild.

Dazu muss das Bild mit den Methoden der Bildverarbeitung erst vorverarbeitet werden, um Kanten als eindeutige Linien zu erhalten. Dies geschieht durch einen Gradientenfilter, der Ecken erkennt; mit einem Threshold-Filter werden starke Kanten erkannt. Dann wird eine Ellipse ermittelt, die möglichst gut auf die ermittelten Kanten passt.

Abb. 4: Ellipse fit

Dieser Algorithmus ist weniger geeignet, wenn das Gesicht teilweise verdeckt ist.

Eine weitere Methode ist die Farbklassifikation, die ermittelt, welche Bereiche des Bildes farblich zu einem Bereich gehören, der vorher als typisch hautfarben ermittelt wurde.

Damit Beleuchtungsunterschiede keinen Einfluss haben, verwendet man hier eine andere Farbskala als RGB. Die Pixelinformation wird in die [Cr, Cb]-Ebene transformiert, dabei bleibt nur die Farbinformation erhalten, während die Helligkeitsinformation verworfen wird. Dies verringert auch die Dimension des Merkmalsraumes, eine Methode, die in der Mustererkennung als Feature Selection bekannt ist.

Median-Filterung als letzter Schritt vermeidet eine eventuelle, unerwünschte Körnigkeit der Bereiche.

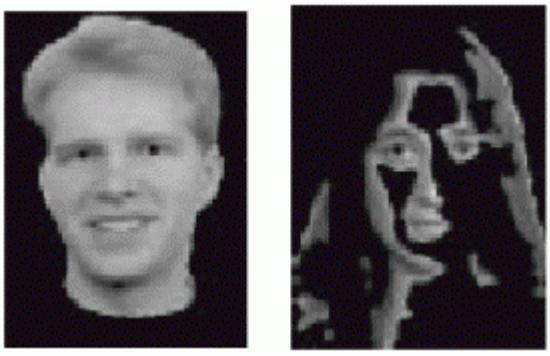

Abb. 5: Ein gutes und ein schlechtes Ergebnis der Farbklassifikation. In dieser Darstellung sind die Bereiche geschwärzt, welche nicht der Haut zugeordnet wurden.

Diese beiden vorgestellten Methoden sind relativ fehleranfällig und setzen voraus, dass überhaupt ein Gesicht im Bild vorhanden ist. Für eine Segmentierung sind sie nicht geeignet, erfüllen aber als Vorverarbeitungsschritte durchaus ihren Zweck.

2.3 Deterministic Template Matching

Template Matching setzt eine gegebene Vorlage voraus und findet eine Position im Bild, wo diese möglichst deckungsgleich passt.

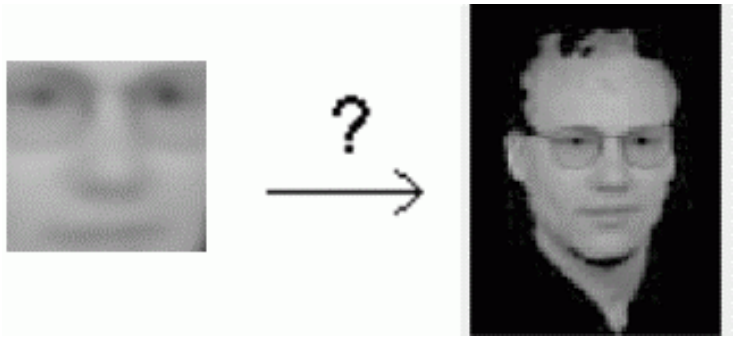

Abb. 5: Template Matching

Die Methode hat folgende Nachteile: Das Aussehen des zu suchenden Objektes muss vorher bekannt sein; einfache Drehungen des Objekts können dazu führen, dass die Schablone nicht mehr anwendbar ist.

2.4 Heuristische Erkennung von transformierten Objekten

Die Komplexität des Template Matchings steigt stark an, wenn man in Betracht zieht, dass es mehrere verschiedene Transformationen gibt, von denen jede eine Dimension zum Suchraum hinzufügt: Die Vorlage kann in zwei Raumrichtungen bewegt sein, sie kann gedreht und in der Größe geändert sein.

Eine *vollständige Durchsuchung* schliesst sich daher aus, und die Suchalgorithmen der Optimierung sind gefragt. Ein *Greedy-Algorithmus*, wie der Gradientenabstieg, würde schnell in lokalen Optima hängen bleiben, also nur die nächstbeste ungefähre Fundstelle liefern. Diese beiden einfachen Methoden sind nicht zufriedenstellend, deshalb findet ein Kompromiss Anwendung, der *Simulated Annealing* genannt wird.

Im Rahmen dieses Algorithmus spielt die Zielfunktion (hier: ein Mass für Gleichheit von Vorlage und Bild) anfangs keine Rolle, die "Fundstelle" springt zufällig hin und her.
Allmählich steigt die Wahrscheinlichkeit, dass die Zielfunktion beachtet wird, bis das Verhalten einem Gradientenabstieg gleicht. So wird die Wahrscheinlichkeit erhöht, dass das gefundene Optimum kein kleines, lokales ist.

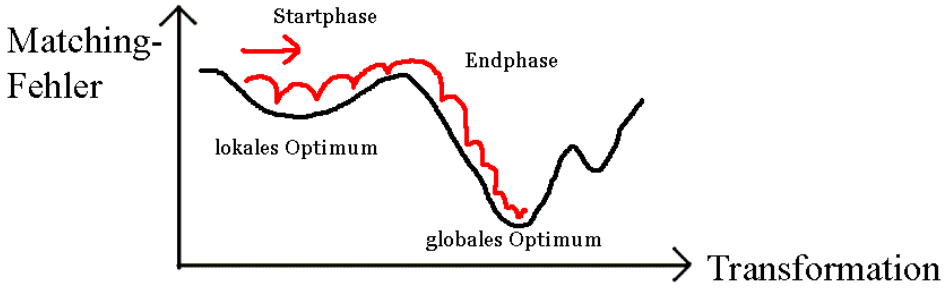

Abb. 6: Veranschaulichung des Simulated Annealing: Die aktuell überprüfte Position (rot) kann sich anfangs auch in eine ungünstige Richtung (hier aufwärts) fortbewegen, gegen Ende ist nur noch eine Abwärtsbewegung möglich.

2.5 Abschluss

Einfache Methoden wie die Unterscheidung aufgrund der Farbwerte können die sichtbaren Hautbereiche sehr schnell erkennen. Sie sind jedoch leicht zu täuschen und nicht robust genug für praktische Anwendungen. Trotzdem können grosse Bereiche des Bildes von der späteren Suche ausgeschlossen werden.
Stochastische Heuristiken, wie Simulated Annealing, können die Suche erheblich beschleunigen.

3. Extrahierung von Gesichtsmerkmalen

3.1 Einführung

Die Extrahierung von Gesichtsmerkmalen befasst sich mit dem Auffinden bestimmter Punkte oder Konturen eines gegebenen Gesichts. Das hängt eng mit der Aufgabe zusammen, ein Modell daran anzupassen.

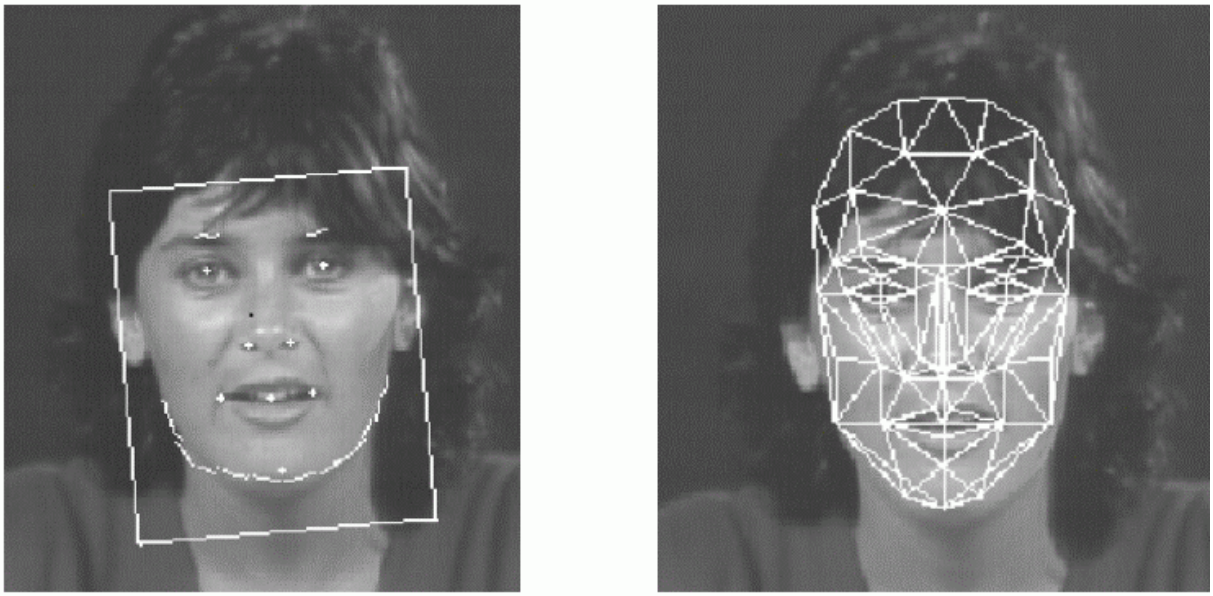

Abb. 7: Erkannte Gesichtsmerkmale (man beachte die Drehungserkennung); ein anhand dieser Zwischenergebnisse angepasstes Drahtgittermodell

3.2 Irisfindung

Hier wird ein schnelles Verfahren vorgestellt, um die Positionen der Augen in einem Gesicht zu finden.

Es werden folgende Eigenschaften der Iris ausgenutzt: Sie ist annähernd kreisförmig, dunkel gegen einen hellen Hintergrund, und der obere und der untere Teil sind meist durch Lider verdeckt. Ausserdem liegen die zwei Augen etwa auf der gleichen Höhe; so können manche fehlerhafte Erkennungen von vornherein ausgeschlossen werden.

Eine Iris wird mittels dreier Parameter beschrieben: Zentrumskoordinaten und Radius (x_0, y_0, r). Wir definieren die Funktion

$$f_{\Theta}(x_0, y_0, r) = \int_{\theta \in \Theta} I(x_0 + r\cos\theta, y_0 + r\sin\theta)d\theta, \quad \Theta =]0, 2\pi]$$

wobei I(x, y) ein Grauwertbild ist. Für einen gegebenen Mittelpunkt (x_0, y_0) wird der wahrscheinlichste Radius das r sein, für das die partielle Ableitung nach r,

$$\frac{\partial}{\partial r} f(x_0, y_0, r)$$

maximal ist. Diese Ableitung ist auch ein Maß für die Wahrscheinlichkeit einer Iris mit dem Zentrum (x_0, y_0) und dem Radius r. Die wahrscheinlichste Fundstelle wird wie folgt ermittelt:

$$(x_0, y_0, r) = \arg \max_{x_0, y_0, r} \left(\frac{\partial}{\partial r} f(x_0, y_0, r)\right)$$

Um die Geschwindigkeit zu erhöhen *und* die Komplexität zu vermindern, kann man das Intervall Theta auf die linken und rechten Viertelkreise beschränken, weil der Irisumriss oft von den beiden Augenlidern verdeckt wird.

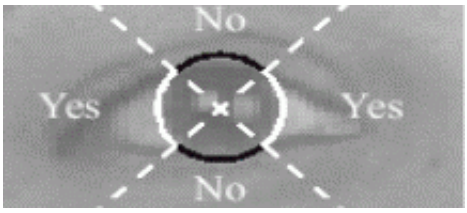

Abb. 8: Detail, mit relevanten Kreissektoren

Eine zusätzliche Faltung der Irisfunktion mit einem Gaußkern kann Sinn machen, um evtl. wegen geringer Auflösung auftretende abrupte Sprünge zu vermeiden. Der gesamte Berechnungsaufwand hierfür scheint auf den ersten Blick gross, benötigt jedoch tatsächlich nur wenige hundert Multiplikationen pro Pixel.

Abb. 9: Ergebnis der Detektierung

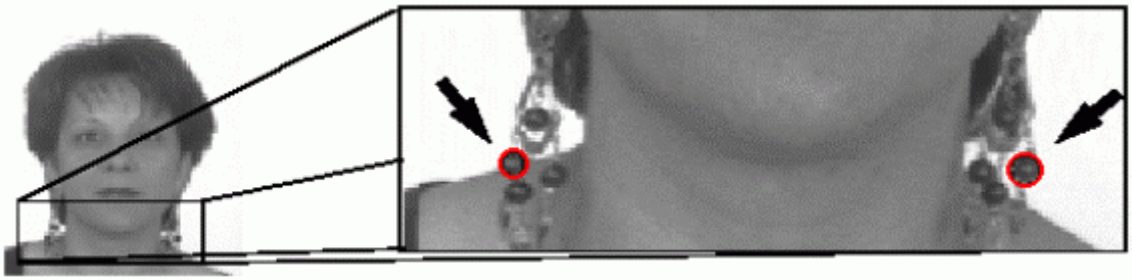

Abb. 10: Der Algorithmus sucht nur nach Kreisen und kann sich auch so täuschen lassen.

3.3 Deformable Line Templates, Dynamic Programming

Nicht nur einzelne Merkmalspunkte, sondern auch Umrisslinien des Gesichts können lokalisiert werden. Zu diesem Zweck benutzt man Kurvenvorlagen (line templates), zu denen passende Kurven im Bild gesucht werden.

Ein line template besteht aus einzelnen Segmenten, welche wiederum aus einzelnen Elementen bestehen. Das Bild entspricht natürlich nie genau der Vorlage, deshalb ist das template anpassungsfähig: Die Segmente haben eine feste Form, sind aber untereinander geringfügig verschiebbar und ermöglichen so eine Verzerrung der gesamten Kurvenvorlage. Diese ist begrenzt, um sich nicht zu sehr von der ursprünglichen Form zu entfernen.

Das Gesamtproblem stellt sich als Optimierungsproblem dar: Finde eine möglichst gute Überdeckung mit einer Kontur im Bild unter Verwendung von erlaubten Verzerrungen des templates.

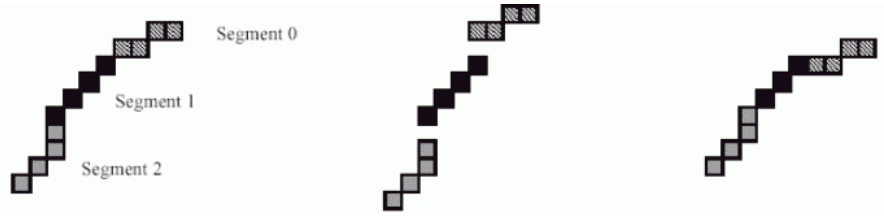

Abb. 11: Ein Teil eines Templates, mit drei Segmenten zu je vier Elementen. Mögliche Verzerrungen.

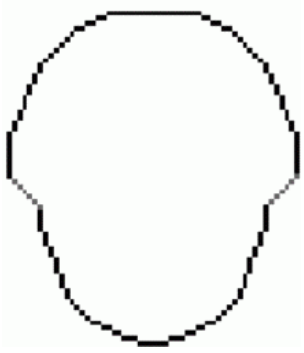

Abb. 12: Eine grobe Kopfkontur, die hier Verwendung findet

Verzerrungen der Vorlage werden erreicht, indem das jeweils nächste Segment an einer leicht anderen Position anknüpft, dabei können auch Lücken und Überlappungen entstehen.

Der Berechnungsaufwand für das Optimierungsproblem, und damit auch die Laufzeit, ist abhängig von der Anzahl der Segmente und den erlaubten Veränderungen. Die Wachstumsklasse des Optimierungsproblems ist unglücklicherweise exponentiell, weil jede Deformation Änderungen für den Rest der Länge des templates nach sich zieht.

Das Problem lässt sich allerdings auf ein shortest-path-Problem auf einem gewichteten Graphen abbilden: Spalten von Knoten entsprechen jeweils einem Segment, die vertikale Position eines Knotens stellt die Richtung der Verzerrung dar. Die Kantengewichte entsprechen der resultierenden Deckungsgleichheit mit den Kanten im Bild.

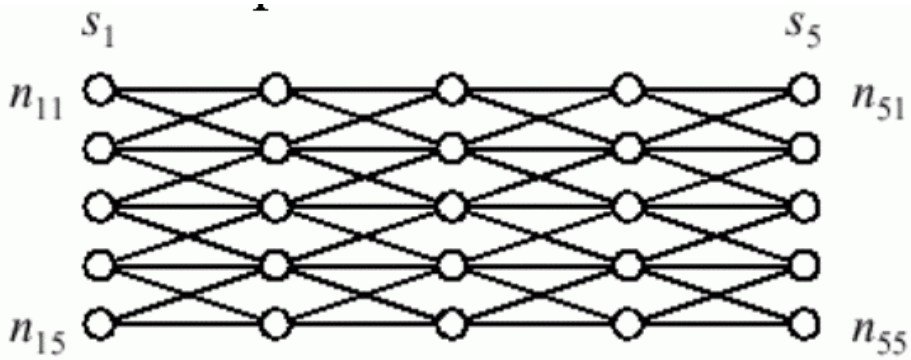

Abb. 13: Visualisierung des shortest-path-Problems für deformable templates. Es sind nicht alle möglichen Kanten eingezeichnet.

Während der Verfolgung einer Kontur kann der kürzeste Weg im Gitter ab einer gewissen Entfernung vom jeweils neu hinzugefügten Segment als fest betrachtet werden. Dies entspricht dem Dynamic Programming, Zwischenergebnisse (d.h. ein bisher optimaler Weg) werden weiterverwendet. So lässt sich die Wachstumsordnung auf linear mit der Anzahl der Segmente reduzieren.

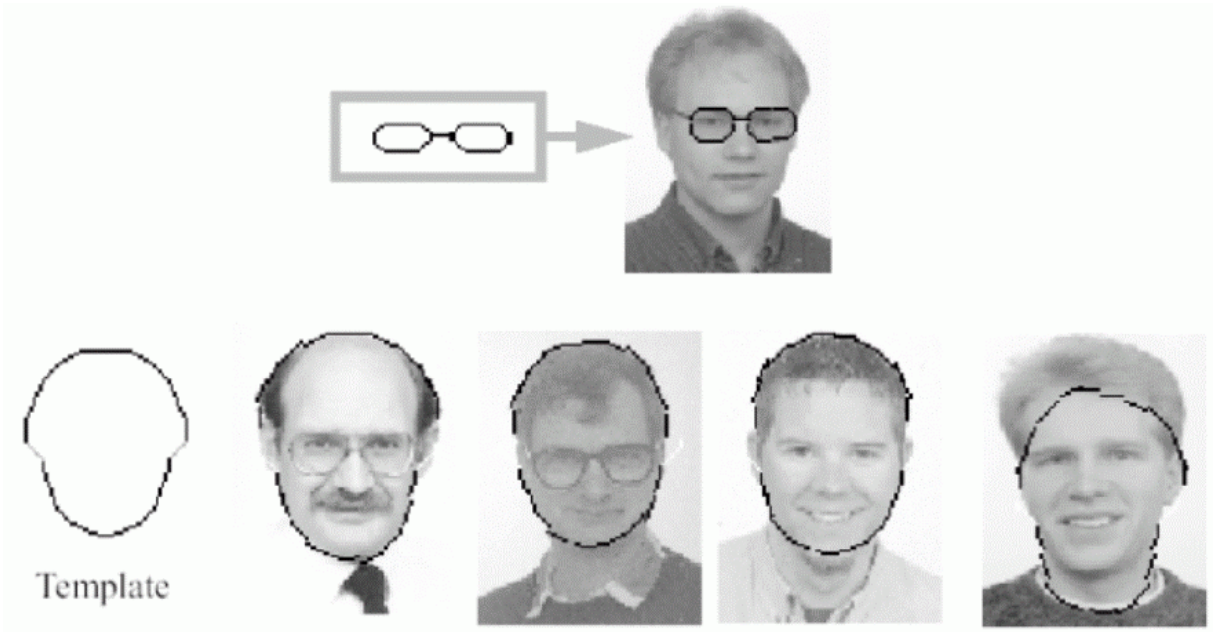

Abb. 14: oben: Lokalisierung einer Brille mit einem Deformable Line Template
unten: 2 gute Erkennungen, im vorletzten Bild wurde das Kinn nicht getroffen, im letzten Bild verfolgte der Algorithmus die falschen Konturen.

3.4 Kombination von Deformable Templates und Statistical Pattern Matching

Hier findet eine Kombination der 2.3 beschriebenen Methode des pattern matchings mit einem deformierbaren Graphen statt. Die Güte der Überdeckung ist bestimmt durch die Gleichheit der templates und die Stärke der Verzerrung des Graphens, der sie verbindet.

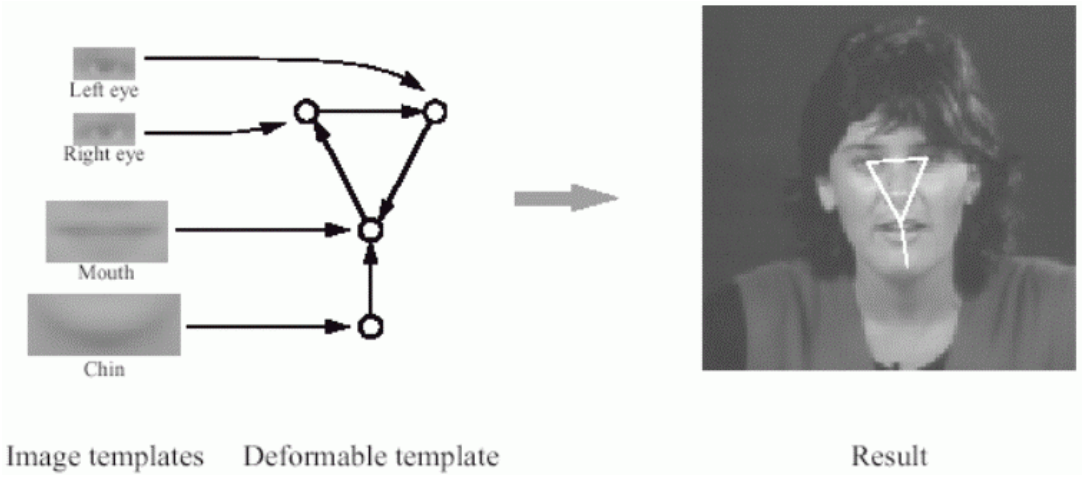

Abb. 15: Kleinere templates finden eindeutige Merkmalspunkte, ihre relative Position untereinander wird durch einen Graphen kontrolliert

Merkmale, an denen die Knoten des Graphen festgemacht werden, werden hier Attraktoren genannt. Welche Attraktoren sind hierfür geeignet? Sie sollten gleichzeitig schnell zu berechnen und robust sein. Ein robuster Attraktor, der das Template zu einer nahezu optimalen Deformation führt, ist einem exakteren vorzuziehen, der gelegentlich versagt. Die Irisfindung aus Kapitel 3.2 bietet sich an, ebenso Mundwinkelerkennung durch pattern matching.

3.5 Deformierbare Graphen

Deformierbare Graphen sind eine Erweiterung des normalen Graphen, welcher aus Knoten und Kanten besteht.

Ein Graph ist hier zusammengesetzt aus sites und gerichteten arcs.
Eine site besteht aus einem array von nodes sowie einem Attraktor, bei dem es sich um einen Attraktor für ein statistical template oder um einen einfachen Kantenattraktor handelt.
Ein arc verbindet jeweils zwei sites, er definiert Standard-Abstände (xy) zwischen den sites sowie die maximalen Abweichungen davon.
Ein node bezieht sich auf eine Position im Bild und besitzt einen Wert v, der durch den Attraktor der site an dieser Position bestimmt wird.
Ein evaluierter Graph ist ein Graph, dessen Attraktoren berechnet wurden und die Werte der nodes entsprechend gesetzt sind.

Die endgültige Deformierung des Graphen baut sich hier von den untersten Söhnen anfangend auf. Innerhalb jeder site wird die wahrscheinlichste Position des zu suchenden Features ermittelt. Wahrscheinlichkeitswerte setzen sich über die Väter der sites nach oben durch, schliesslich wird die Wurzel mit dem höchsten Wert ausgewählt und von ihr aus wieder nach unten mit den wahrscheinlichsten feature points weitergearbeitet. Die Suche nach einem feature point wird durch das Gebiet der site, zu der er gehört, beschränkt.

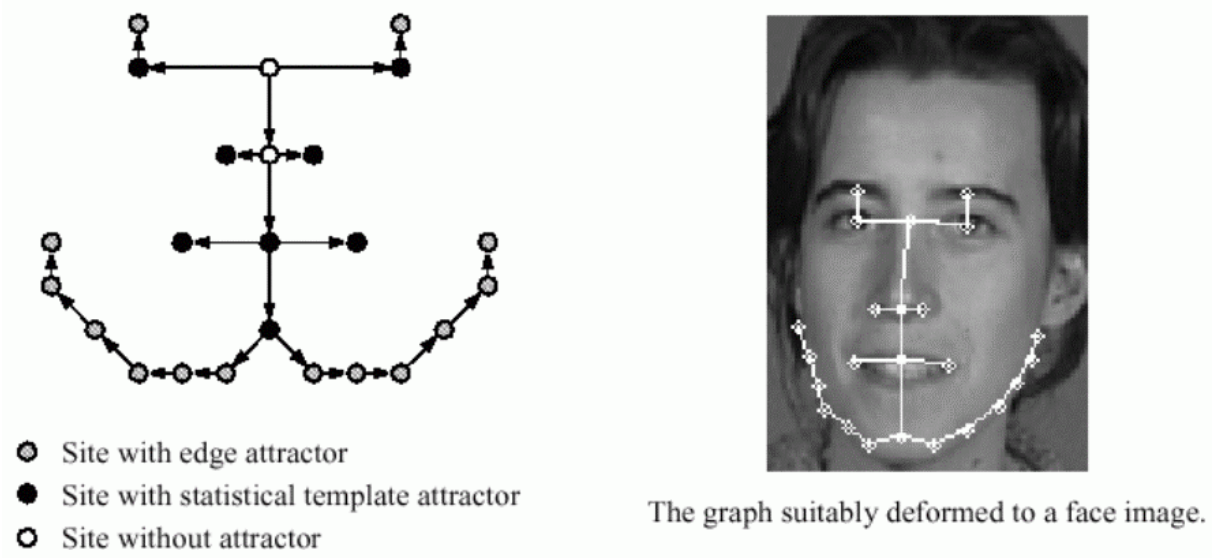

Abb. 16: Aufbau eines deformierbaren Graphen für die Gesichtserkennung, Ergebnis

3.6 Ein Drahtgittermodell an die extrahierten Gesichtsmerkmale anpassen

Für die modellbasierte Kodierung wird nun ein Drahtgittermodell an die extrahierten Gesichtsmerkmale angepasst. Dieses muss auf Sender- und Empfängerseite gleich sein, damit der Sender eine Selbstkontrolle vornehmen kann.

Eine gewisse Fehlerkontrolle erfolgt durch limitierte Deformationen des Modells.

Die Positionen von Knoten, die nicht direkt Feature Points entsprechen, müssen durch Interpolation ermittelt werden.

Das CANDIDE-Modell ist ein relativ einfaches Modell, das bereits 1987 vorgestellt wurde. Es besteht aus etwa 100 Dreiecken, und obwohl bereits wesentlich komplexere Modelle existieren, reicht es aber für viele Zwecke bereits aus.

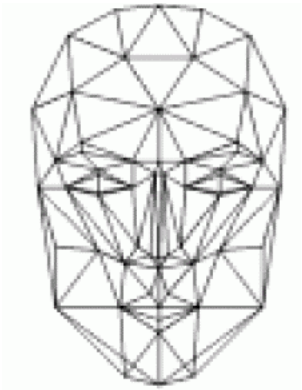

Abb. 17: Das CANDIDE-Modell mit 100 Polygonen

Zu beachten ist, dass hier quasi der Sprung vom flachen Bild zum dreidimensionalen Modell stattfindet, die Tiefeninformation geht ausschliesslich aus der Grundform des Gesichtsmdells hervor. In der Regel wird diese einfach übernommen.

3.7 Abschluss

In diesem Kapitel wurde das Problem der Merkmalsextraktion als mehrdimensionales Optimierungsproblem betrachtet. Die Algorithmen laufen nahezu in Echtzeit ab, so ist es möglich sie in einem modellbasierten Kodierungsschema zu verwenden.

4. Kodierung der Parameter des Gesichtsmodells

4.1 Gesichtsmodellparameter

Die bestimmenden Parameter sind hier die Form, die Textur und die Animation des Modells.

Diese Parameter unterteilt man in zwei Gruppen:
In solche, für die es reicht, sie am Anfang einmalig zu übertragen. Dazu gehört die Art des verwendeten Modells, also seine Grundform, sowie die Textur, die über die Oberfläche gespannt wird.
Die anderen Parameter, die in jedem Frame neu übermittelt werden müssen, sind die Animationsparameter, welche die Unterschiede zum vorherigen Frame kodieren.

4.2 Kodieren von lokalen Bewegungsparametern mittels MPEG-4

MPEG-4 definiert 66 Facial Action Parameters (FAPs). Jeder dieser Parameter beschreibt die Bewegung eines der Gesichtsmerkmalspunkte in einer von drei Dimensionen.

Die erforderliche Bitrate berechnet sich wie folgt:
Für jedes FAP werden durchschnittlich 6 bit benötigt. Um eine flüssige Animation zu erreichen, ist eine zeitliche Auflösung von 15-30 Hz notwendig.
Es lassen sich einige Einsparungen erreichen: Mittels Maskierung wird vorher übermittelt, welche FAPs sich überhaupt geändert haben; dann werden nur diese übertragen.
Ausserdem ist eine begrenzte Bewegungsvorhersage möglich.
Durch eine Kombination der genannten Methoden resultiert eine Bitrate von 2 kbit/s, welche sich durch übliche Komprimierung (Huffman etc.) auf 1 kbit/s reduzieren lässt.

Sprache und Modellparameter können über den gleichen Kanal übertragen werden.

4.3 Gesichtsmimik, Action Units und Basis Functions

Die FAPs sind nicht unabhängig voneinander, sondern hängen in der Art der Bewegung oftmals eng zusammen. Das Ziel ist, die Anzahl der nötigen Parameter auf weniger mit nahezu dem gleichen Inhalt zu reduzieren.
Mit Action Units können mehrere FAPs zusammengefasst werden, es wird die Bewegung von einzelnen willkürlichen Muskeln kodiert, die für die Bewegung ganzer Hautpartien verantwortlich sind.
Mit Facial Basis Functions (FBFs) geht die Merkmalsauswahl noch einen Schritt weiter, sie beschreiben u.a. Aktionen wie "nach links sehen", "Mund öffnen" oder "Kopf nach rechts drehen".

4.4 Abschluss

In diesem Kapitel wurden Methoden beschrieben, um Gesichtsanimationsparameter zu beschreiben und zu komprimieren. Effiziente Kompression und Frames ohne Verzögerung zwischen ihnen machen Echtzeitanwendungern wie Videotelefonie möglich, mit geringen Bitraten von bis zu 1000 Bit/s.

5. Schlussfolgerungen

Gesichtsfindung: Anhand von Form, Farbe und gematchten Templates kann ein Gesicht recht schnell, wenn auch nicht immer zuverlässig grob lokalisert werden.

Merkmalsextraktion: Durch Methoden wie Iris Detection oder Deformable Line Templates findet die Anpassung eines Drahtgittermodells statt. Durch die Anwendung des Modells wird die Menge der zu übertragenden Daten signifikant reduziert.

Kodierung: Das ermittelte Modell wird effizient und in Echtzeit über einen begrenzten Kanal übertragen.

Referenzen

[1] Jörgen Ahlberg. Extraction and Coding of Face Model Parameters. PhD-Thesis, Linköpings universitet, Sweden, 1999. ISBN 91-7219-425-1