

Masterarbeit

Fachgebiet der Masterarbeit:
Signalverarbeitung und Bilderkennung

Thema der Masterarbeit:

Echtzeitauswertung minimaler Augenbewegungen (EMA) bei schwerstbehinderten
Personen

Kandidat:	Alexander Kühn (Matr.Nr.:7122993)
Prüfer:	Prof. Dr. Horst Wupper
Zweitprüfer:	Dr.-Ing. Fritz Heinrichmeyer
externer Betreuer:	Gerhard Hornicek
Datum der Ausgabe der Arbeit:	01.06.2008
Datum der Abgabe der Arbeit:	01.06.2009

Wintersemester 2008/2009
Sommersemester 2009

Fernuniversität Hagen
Bereich Mathematik und Informatik
Fachbereich Elektrotechnik
Fachrichtung Master Elektro- und Informationstechnik

Zusammenfassung

Stichworte

Digitale Signalverarbeitung; Blinzelerkennung; Behindertentechnik; Videoanalyse; Echtzeit; Gesichtserkennung; Korrelationsverfahren; Analyse im Farbraum

Zusammenfassung

In dieser Masterarbeit wird die Entwicklung eines Systems zur Erkennung und anschließender Auswertung des Lidschlags von schwerbehinderten Personen beschrieben. Die schwerbehinderten Personen sollen dadurch in die Lage versetzt werden, Schaltvorgänge alleine durch ihren Lidschlag auslösen zu können. Hierdurch soll es ihnen ermöglicht werden, einfache technische Geräte wie Kassettenrekorder oder Haushaltsgeräte zu benutzen. Für die Realisierung eines solchen Systems sollen nur handelsübliche PC-Komponenten zum Einsatz kommen. Der Lidschlag soll dabei über eine Webcam aufgenommen werden. Zur Erkennung der Lidschläge werden zum Auffinden der Augenpositionen Verfahren der Gesichtserkennung angewandt. Nachdem die initiale Augenposition ermittelt wurde, werden die Augen mit einem Kreuzkorrelationsverfahren verfolgt und der Lidzustand mit Hilfe einer Farbraumanalyse bestimmt. Die erreichbare Erkennungsrate liegt bei dem Verfahren bei 75%.

Keywords

Digital signal processing; blink detection; technology designed to assist handicapped people; video analysis; real time; face recognition; cross correlation; color space

Abstract

The following thesis describes the development of an eyelid movement detector designed to give severely disabled persons an opportunity to communicate with the outside world. The user shall be able to accomplish a switching operation solely by moving his eyelid. Thus he would be empowered to use simple technical appliances including cassette recorders or certain household appliances. The choice of components is restricted to the commercially available ones; the eyelid movement shall be recorded by an ordinary webcam. The process of blink detection is done by using the means of face recognition; after detecting the initial eye position, the eyes are being followed by a cross correlation process. The analysis of the color space then helps detecting the specific eyelid state. The method used obtains a recognition rate of 75%.

Inhaltsverzeichnis

1. Einleitung	1
1.1. Motivation	1
1.2. Anforderungsanalyse	2
1.2.1. Funktionale Anforderungen	2
1.2.2. Nichtfunktionale Anforderungen	4
1.3. Das Konzept	4
1.4. Systemarchitektur	5
2. Grundlagen	7
2.1. Bildverarbeitungssysteme	7
2.1.1. Überblick über die Erkennungsverfahren	8
2.1.2. Gesichtserkennungsverfahren	9
2.2. Evaluation der Verfahren	11
3. Methodenentwicklung	13
3.1. Algorithmenzerlegung	13
3.2. Lokalisation der Augen	15
3.2.1. Differenzbildung	15
3.2.2. Die Binarisierung	16
3.2.3. Morphologische Filterung	17
3.2.4. Ermittlung der Augenregion	20
3.2.5. Filterung der Augenregion	24
3.3. Augenverfolgung und Zustandsbestimmung	28
3.3.1. Initialisierung der Verfolgung	29
3.3.2. Verfolgung der Augenposition	35
3.3.3. Bestimmung des Öffnungszustands	37
4. Entwicklung der Software <i>sensbli</i>	41
4.1. Softwarerarchitektur	41
4.1.1. Komponentensichtweise	43
4.1.2. Physikalische Sichtweise	45
4.2. Prototyp	48
4.2.1. Modulkonzept	48
4.2.2. Implementierung	49
4.3. Echtzeitsoftware	51
4.3.1. Programmierumgebung und Bibliotheken	51
4.3.2. Implementierung	53
4.3.3. Programmlauf	54
4.4. Ausgabemodul	56

4.4.1. UDP-Nachrichten	56
4.4.2. Implementierung des Ausgabemoduls	57
5. Untersuchung der Leistungsfähigkeit	62
5.1. Erkennungsleistung	62
5.1.1. Erkennungsleistung der Augenlokalisierung	63
5.1.2. Erkennungsleistung der Augenverfolgung	65
5.2. Laufzeitanalyse	67
6. Zusammenfassung und Ausblick	68
6.1. Erfüllungsgrad der Anforderungen	68
6.1.1. Erfüllungsgrad der funktionalen Anforderungen	68
6.1.2. Erfüllungsgrad der nichtfunktionalen Anforderungen	69
6.2. Potentielle Verfahrensverbesserungen	69
6.3. Fazit	70
A. Literaturverzeichnis	71
B. Experimentelle Auswertungen	73
B.1. Augenlokalisierung	73
B.1.1. Ermittlung der Binarisierungsschwelle	73
B.1.2. Ermittlung des Strukturelements	74
B.1.3. Strukturelemente	74
B.1.4. Parametrisierung des anthropologischen Modells	76
C. Beiliegende CD-ROM	78

Abbildungsverzeichnis

1-1.	Systemkomponenten	5
2-2.	Allgemeiner Aufbau eines Bilderkennungssystems	7
3-3.	Verfahrenszustände	13
3-4.	Daten- und Kontrollfluss	14
3-5.	Unwillkürlicher Lidschlag	16
3-6.	Differenzbild eines unwillkürlichen Lidschlages	17
3-7.	Binarisiertes Differenzbild eines unwillkürlichen Lidschlages	18
3-8.	Binäres Strukturelement	19
3-9.	Anwendung der morphologischen Operatoren	20
3-10.	Morphologisch bearbeitetes Binärbild	20
3-11.	Nachbarschaftsarten	21
3-12.	Beispiel des Labeling-Verfahrens	22
3-13.	Ein mit Labels versehenes Binärbild	22
3-14.	Eigenschaften der Regionen	25
3-15.	Paarbildung der Regionen	25
3-16.	Filterkette der Regionen	26
3-17.	Berechnung des Bewertungsexponenten	28
3-18.	Skalierung der normierten Breite	29
3-19.	Ablauf der Lokalisation	30
3-20.	Maskenbildung des aktuellen Bildes P_n	31
3-21.	Maskenbildung des vorhergehenden Bildes P_{n-2}	31
3-22.	RGB Farbwürfel	32
3-23.	HSV-Farbraum	33
3-24.	Maske mit dazugehöriger Sättigung	34
3-25.	Sättigungshistogramme	34
3-26.	Binarisierte Sättigungen	35
3-27.	Template Matching	36
3-28.	Nachführen des Suchfensters	37
3-29.	Korrelationswert in den Lidzuständen	38
3-30.	Fehlerhafte Zustandserkennung	39
3-31.	Entscheidungsbaum zum Augenzustand	39
3-32.	Ablauf der Augenverfolgung	40
4-33.	Struktur eines signalverarbeitenden Systems	42
4-34.	Prozesskomponente	43
4-35.	Komponentenansicht der Prozesskette	44
4-36.	Zustandsautomat Prozessmanager	44
4-37.	Sequenzdiagramm Zustandswechsel	46
4-38.	Physikalische Schichten	47
4-39.	Prototypen-GUI	49

4-40. Prototypen Visualisierung des Trackings	50
4-41. Klassendiagramm sensbli	54
4-42. Bildschirmaufnahme während des Kalibrierens	55
4-43. Zustandsautomat in dem While-Modus	58
4-44. Zustandsautomat in dem Toggle-Modus	59
4-45. Zustandsautomat in dem Timed-Modus	60
4-46. Relaiskarte	60
5-47. Umgebungssuche	64
B-48. Regioneneigenschaften	77

Tabellenverzeichnis

1-1. Funktionale Anforderungen	3
1-2. Nichtfunktionale Anforderungen	4
2-3. Vergleich der Verfahren	12
4-4. Module des Prototypen	52
4-5. Module der Echtzeitanwendung	53
4-6. Konfigurationsparameter	55
4-7. Übersicht der UDP-Kommandos	57
4-8. Konfiguration des Ausgabemoduls	61
5-9. Übersicht über die Testvideosequenzen	63
5-10. Erkennungsleistung der Augenlokalisierung	65
5-11. Erkennungsleistung der Augenverfolgung	66
5-12. Laufzeitverhalten der Software	67
6-13. Erfüllungsgrad der funktionalen Anforderungen	68
6-14. Erfüllungsgrad der nichtfunktionalen Anforderungen	69

1. Einleitung

Es gibt weder große
Entdeckungen noch wahren
Fortschritt, solange noch ein
unglückliches Kind auf der
Welt ist.

(Albert Einstein)

1.1. Motivation

In der vorliegenden Masterarbeit wird die Entwicklung einer Kommunikationshilfe für schwerbehinderte Personen vorgestellt, welche nur noch mittels kontrollierter Bewegung ihrer Augenlider mit der Außenwelt interagieren können. Der Anstoss für dieses Thema kam dabei durch die bereits bestehenden Kontakte des Autors zu der Arbeitsgruppe ELECOK.

Die Arbeitsgruppe ELECOK wurde ursprünglich von dem bayerischen Staatsministerium für Unterricht, Kultus, Wissenschaft und Kunst im Jahre 1985 ins Leben gerufen, um zu untersuchen, inwieweit durch den Einsatz elektronischer Systeme individuelle Erschwernisse von schwer körperbehinderten Schülern gemildert oder ausgeglichen werden können und somit deren selbständiges Handeln gefördert werden kann (vgl. [ELE]). Inzwischen hat sich die Arbeitsgruppe zu einem Netzwerk aus derzeit zehn Einrichtungen in ganz Bayern entwickelt; sie hat nun auch die Zielsetzung, körperbehinderte Schüler bei der Versorgung mit elektronischen Hilfen zu beraten und zu unterstützen.

Somit erstreckt sich der Aufgabenbereich der ELECOK inzwischen von der Beratung von Schülern, Lehrern und Eltern auch auf die Erstellung neuer Konzepte und auf die Zusammenarbeit mit Technologielieferanten.

Die technischen Lösungen dienen dabei in erster Linie als Hilfsmittel für die Ein- und Ausgabe von Daten, umfassen aber auch die Verarbeitung dieser Daten mit speziell zugeschnittener Lernsoftware. Bei den angepassten Eingabemethoden handelt es sich um Bildschirmtastaturen in Matrixansteuerweise, Mundschalter oder auch um Spracheingabe, während für die Ausgabe angepasste Bildschirmseiten bis zu Symbolmatrizen oder auch komplexe Aktuatoriken (wie Spielzeug oder Fahrzeuge) zum Einsatz kommen. Neben der Auswahl der richtigen Hilfsmittel bietet die ELECOK an, die Schüler auch an den Geräten zu trainieren.

Eine besondere Herausforderung ergab sich für die ELECOK durch die Schülerin Daniela Funke. Daniela ist in ihren Interaktionsmöglichkeiten so stark eingeschränkt, dass nur die Auswertung ihrer kontrollierten Augenlidbewegung für eine Kommunikation sinnvoll erscheint. Bisher gibt es jedoch noch keine auf dem Markt erhältliche

Lösung, die den Anforderungen der Schülerin gerecht wird. Daniela arbeitet derzeit mit einem optisch-mechanischen Lidschlagsensor, dessen Erkennungsqualität jedoch nicht für sie ausreicht, was ihr Kommunikationstraining erschwert.

Aus diesem Grund trat der Leiter der ELECOK-Beratungsstelle Altdorf, Gerhard Hor nicek, an den Autor heran, um eine Ersatzlösung für den optisch-mechanischen Schalter zu finden. Diese Ersatzlösung soll der Schülerin Daniela und anderen Personen, die ähnlich eingeschränkt sind, Fortschritte in ihren Kommunikationsmöglichkeiten eröffnen.

Da eine volle Bezugssumme der benötigten Gerätschaften durch die Krankenkassen nicht zu erwarten ist, soll zudem neben der Funktionalität ein Hauptaugenmerk auch auf einer preisgünstigen Lösung liegen.

Im folgenden werden die Anforderungen an das zu entwickelnde System daher unter diesen Gesichtspunkten zunächst näher spezifiziert und dann einer Betrachtung unterzogen.

1.2. Anforderungsanalyse

Die Anforderungen an das zu entwickelnde Lidschlagerkennungssystem ergeben sich einerseits aus den Funktionalitäten, die der bereits vorhandene Lidschlagsensor bietet, sowie andererseits aus den neuen Wünschen, die aus den Erfahrungen mit dem alten Sensor resultieren. Zudem sollen auch die Unzulänglichkeiten des aktuellen Sensors umgangen werden. Eine grob umrissene Anforderungsliste kann dem Entwicklungsauftrag [Hor08] entnommen werden.

Die Anforderungen an das System lassen sich in

- funktionale und
- nichtfunktionale Anforderungen

unterteilen.

Die funktionalen Anforderungen beschreiben dabei, was das zu entwickelnde Produkt leisten soll; die nichtfunktionalen Anforderungen legen hingegen fest, welche Eigenschaften das Produkt neben der reinen Funktionalität besitzen soll, also beispielsweise die zu nutzende Systemplattform, den maximalen Speicherverbrauch oder auch den Preis.

1.2.1. Funktionale Anforderungen

Die funktionalen Anforderungen sind in Tabelle 1-1 zusammengefasst.

Die Anforderung 7f zur Reaktionszeit des Systems von 400ms ergibt sich aus der psy-

Index	Anforderung
1f	Es soll der kontrollierte Lidschlag einer Person ausgewertet werden.
2f	Die Informationsgewinnung soll berührungslos stattfinden.
3f	Wenn das Lid über eine definierte Zeit geschlossen ist, soll eine Reaktion erfolgen.
4f	Die Reaktion soll ein Schaltsignal darstellen.
5f	Das Schaltsignal soll über eine zugelieferte Relaisplatine ausgegeben werden.
6f	Die Reaktion soll eine konfigurierbare Schaltstrategie sein.
6.1f	Die Reaktion (der aktive Zustand) soll konfigurierbar auf das geschlossene oder geöffnete Lid erfolgen.
6.2f	Die Dauer des aktiven Zustands, bis eine Schaltung erfolgt, soll in 100ms-Schritten konfigurierbar sein.
6.3f	Die Schaltstrategie soll aus mehreren Varianten wählbar sein: <ol style="list-style-type: none"> 1. Solange der aktive Zustand anliegt, soll das Schaltsignal anliegen. 2. Bei jedem Wechsel in den aktiven Zustand soll das Schaltsignal umschalten (Toggle). 3. Beim Wechsel in den aktiven Zustand soll das Schaltsignal eine in 100ms-Schritten wählbare Zeit anliegen.
7f	Die Reaktionszeit des Systems soll 400ms nicht überschreiten.

Tabelle 1-1: Funktionale Anforderungen

chologischen Schwelle, an der eine Person noch ein Ereignis zu einer Schaltreaktion zuordnen kann.

1.2.2. Nichtfunktionale Anforderungen

Die Tabelle 1-2 führt die nichtfunktionalen Anforderungen auf.

Index	Anforderung
1n	Die Adaptierung an unterschiedliche Personen soll leicht und schnell durchführbar sein.
2n	Die Adaptierung an unterschiedliche Arbeitspositionen soll leicht und schnell durchführbar sein.
3n	Das System soll kostengünstig realisiert werden, hierzu soll die Verwendung von in Standardhaushalten vorhandenen EDV-Systemen möglich sein.
4n	Das System soll intuitiv bedienbar sein bei vorhandenen Grundkenntnissen in der Arbeit mit Personal Computern.
5n	Aus Kosten- und Sicherheitsgründen ist das Betriebssystem Linux als Plattform vorzuziehen.
6n	Das Schaltsignal soll über USB zur Steuerung eines potentialfreien Ausgangs zur Verfügung gestellt werden.

Tabelle 1-2: Nichtfunktionale Anforderungen

Ein großer Teil der aufgeführten nichtfunktionalen Anforderungen lässt einen gewissen Interpretationsspielraum zu. Daher musste während der Entwicklungsphase eine weitere Abstimmung mit den Erwartungen der Auftraggeber erfolgen.

1.3. Das Konzept

Nachfolgend soll ein kurzer Überblick über das Konzept und den Anwendungsfall für das Lidschlag-Erkennungssystem gegeben werden.

Der schwerbehinderte Nutzer muss fähig sein, seine Augenlider gezielt für eine bestimmte Zeit zu schließen. Diese Zeit soll dabei deutlich über der Schließzeit während des natürlichen Reflexes zur Befeuchtung der Augen liegen; Letzterer soll im folgenden — zur Abgrenzung von dem auszuwertenden gewollten Lidschlag —, als unwillkürlicher Lidschlag bezeichnet werden.

Die Länge des unwillkürlichen Lidschlags liegt bei einer Zeitspanne von 200-300ms.¹ Daraus lässt sich ableiten, dass mit einer Sicherheitsreserve von 30% ein gewollter

¹Siehe hierzu [Kle], Kap. 5.2.2.

Lidschlag eine Zeitspanne von 400ms nicht unterschreiten sollte.

Die Augenbewegungen des Nutzers sollen von einer Kamererasensorik aufgenommen werden, deren Bilddaten wiederum ein Computerprogramm auswertet. Dieses Computerprogramm soll auf Grund des Bilddatenstroms eine Entscheidung darüber treffen, in welchem Zustand (Lid offen/Lid geschlossen) sich die Augen befinden. Anhand des Zustandsübergangs und der Verweildauer in einem Zustand soll dann eine Aktion ausgelöst werden. Die optische Abtastung durch eine Kamera ermöglicht dabei eine berührungslose Arbeitsweise.

Die Aktion des Systems soll darin bestehen, einen elektrischen Verbraucher über eine Netzschatzbox nach unterschiedlich definierten Strategien ein- und auszuschalten. Die anzuwendende Strategie soll von einem Systembetreuer einstellbar sein; die Strategien sind in der funktionalen Anforderung 6.3f in der Tabelle 1-1 näher aufgeführt.

1.4. Systemarchitektur

Die oben aufgeführten Anforderungen sowie das Konzept führen zu der in Abbildung 1-1 dargestellten Systemarchitektur².

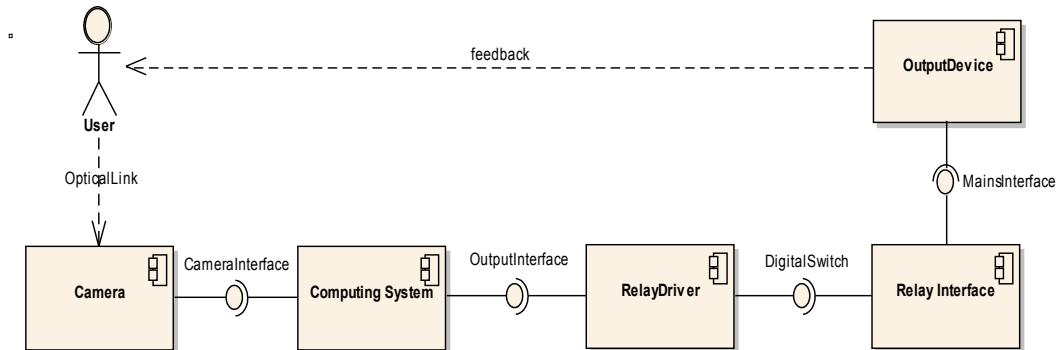


Abbildung 1-1: Systemkomponenten

Die Architektur stellt eine Verarbeitungskette dar, die bei dem Benutzer „User“ beginnt. Der Benutzer interagiert dabei über die optische Verbindung „OpticalLink“ mit der Kamera. Die optische Verbindung repräsentiert die relevanten Komponenten zwischen dem Benutzer und dem Bildaufnehmer der Kamera. Sie wird hier explizit aufgeführt, da diese Komponente in einem Bildverarbeitungssystem durchaus relevant für die Systemleistung ist; so bestimmt sie beispielsweise den Abstand zwischen Nutzer und Kamera und beeinflusst auch die Abbildungsqualität maßgeblich.

Die Kamera „Camera“ wandelt das einfallende Bild in ein elektrisches Signal und

²Die Architektur wird als UML-Komponentendiagramm (Vgl. [RHQ⁺05], S.213ff.) dargestellt

stellt es der verarbeitenden Einheit „Computing System“ über die „CameraInterface“-Schnittstelle bereit. Bei der Kamera und der Schnittstelle muss es sich um (er-schwingliche) Komponenten handelsüblicher Konsumgeräte handeln, so dass die Auswahl eingeschränkt ist. In Betracht kommen somit nur Webcams oder digitale Camcorder, die über die Schnittstellen Firewire und USB mit der Verarbeitungseinheit kommunizieren.

Die Verarbeitungseinheit besteht aus einem Personal Computer nach dem Stand der Technik im Jahr 2008; dabei ist für die Dimensionierungsgröße eines PC die Takt-frequenz seines Prozessors maßgeblich. Da die Bildverarbeitungssoftware eine große Menge an Rechenschritten durchführen muss, ist ein Doppelkernprozessor mit einer Frequenz von 2,2GHz ausreichend; andere Systemparameter wie Festplattengröße, Arbeitsspeicher oder Grafikkartenleistung sind eher zweitrangig. Die Software nutzt die Ein- und Ausgabedienste des Betriebssystems, für das im konkreten Fall ein Linuxderivat zum Einsatz kommen soll. Linux hat hierbei die beiden ausschlagge-benden Vorteile, dass es einerseits frei und ohne Zusatzkosten erhältlich ist und andererseits stark automatisierbar ist, so dass die Blinzelerkennung dann auch ohne weitere Betreuerinteraktion eingesetzt werden kann.

Die Verarbeitungseinheit errechnet den Augenlidzustand des Benutzers und leitet daraus eine Schaltinformation für die Ausgabeeinheit ab. Diese Schaltinformation wird über das „OutputInterface“ an die Treiberkomponente „RelayDriver“ vermit-telt. Die Aufgabe des Relais-Treibers ist es dann, die in der Behindertentechnik üblichen Relais-Schaltboxen an die Verarbeitungseinheit zu adaptieren. Die Schaltbox wird wiederum durch die Komponente „Relais-Interface“ repräsentiert. Die Schalt-box dient dabei zum Ein- und Ausschalten von 230V-Verbrauchern mit Hilfe von behindertenspezifischen Tastern; die Verbraucher können dann in verschiedenen Modi wie „Pulsbetrieb“ oder „Halten für eine gewisse Zeit“ betrieben werden. Die Verbrau-cher werden in der Abbildung als „OutputDevice“ bezeichnet. Der Relais-Treiber stellt das Taster-Schaltsignal über die Schnittstelle „DigitalSwitch“ zur Verfügung. Zur Realisierung des Relaistreibers wurde eine externe USB-Karte zugekauft, die mehrere digitale Schaltausgänge galvanisch getrennt bereitstellt. Der Ausgang schal-tet die Kontakte des Tastereingangs der Relais-Schaltbox direkt durch.

Als Ausgabegerät können unterschiedliche Geräte genutzt werden; hier ergeben sich die Anwendungsfälle vor allem aus pädagogischen Gesichtspunkten. Für die ersten Untersuchungen und Trainingsschritte während der Entwicklung fand ein einfacher Kassettenrecorder Verwendung, aber auch Haushaltsgeräte können angesteuert wer-den. So ist etwa vorstellbar, dass der Nutzer durch seinen Lidschlag ein Rührgerät schalten und somit beim Kochen und Backen unmittelbar mithelfen kann.

2. Grundlagen

Den Schwerpunkt dieser Arbeit bildet die Erstellung eines geeigneten Algorithmus zur Erkennung des Lidschlags einer Person. Dieser Algorithmus wird auf einem Digitalrechner ausgeführt, der die Bildinformation als einen zeitlich äquidistanten Datenstrom von Bilddaten von einer Kamera erhält. Die Aufgabenstellung stammt damit aus dem Gebiet der Bildverarbeitung.

2.1. Bildverarbeitungssysteme

Bei der Bildverarbeitung handelt es sich wiederum um eine Disziplin der Signalverarbeitung mit dem Ziel, Informationen aus Bilddaten zu gewinnen. Die Bildverarbeitung beinhaltet insbesondere die Unterdisziplinen

- Bildverbesserung und
- Bilderkennung.

Die Bildverbesserung dient der Qualitätsverbesserung von in Bildmaterial enthaltener Information mit Hilfe der Signalverarbeitung; die Bilderkennung beschreibt hingegen Methoden zur Entscheidungsfindung aus vorliegenden Bilddaten beziehungsweise zur Klassifikation von Objekten. Diese Methoden werden auch unter dem Begriff „maschinelles Sehen“ eingeordnet.

Aufgrund der Reproduzierbarkeit der Ergebnisse, des geringeren Elektronikaufwands und der hohen Flexibilität bei der Verarbeitung der Daten findet die Verarbeitung heute fast durchgängig im digitalen Bereich statt.

Die Erkennung des Lidschlags ist ein Verfahren aus der Bilderkennung, da eine Information aus dem abstrakten Bilddatenstrom gewonnen werden soll: Die binäre Aussage, ob die Augenlider offen oder geschlossen sind. Bilderkennungssysteme nutzen jedoch auch die Methoden der Bildverbesserung, um beispielsweise eine Informationsreduktion vor der eigentlichen Erkennung vorzunehmen. Der allgemeine Aufbau eines Bilderkennungssystems³ zeigt Abbildung 2-2.

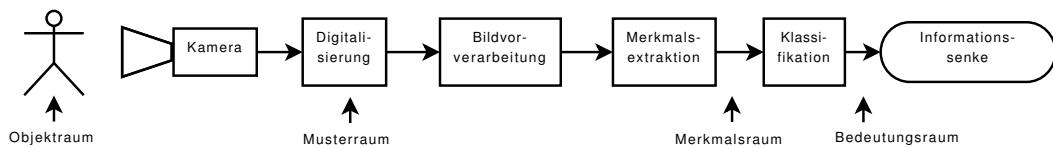


Abbildung 2-2: Allgemeiner Aufbau eines Bilderkennungssystems

³Vgl. [Wah89], S. 2.

Das Bild wird von der Kamera aufgenommen und durch die Digitalisierung in einen Datenstrom gewandelt. Danach werden die Bilddaten für die weiteren Prozessschritte mit Verbesserungsverfahren vorbereitet, um eine Reduktion der Information vorzunehmen (etwa eine Wandlung in Graustufen) oder die Qualität zu verbessern (z.B. Rauschfilterung, Deinterlacing). Darauffolgend findet eine Merkmalsextraktion statt, um aus den Daten Merkmale zu gewinnen. Die so extrahierten Merkmale werden dann mit Hilfe eines Klassifikators eingeteilt um schließlich die gewünschte Information zu generieren, die zur weiteren Verarbeitung einer Informationssenke bereitgestellt wird.

Der Informationsfluss in einem Bilderkennungssystem kann somit in verschiedene Phasen eingeteilt werden. Das zu erkennende Objekt, das in der Kamera abgebildet wird, ist der Objektraum. Die digitale Repräsentation dieses Objekts bezeichnet man als Musterraum, während die Information nach der Merkmalsextraktion in dem sogenannten Merkmalsraum angesiedelt ist. Die gewünschte Information, die das Erkennungssystem gewinnen soll, wird als Bedeutungsraum bezeichnet.

Die Methoden der Bildverbesserung sind mathematische Verfahren, die mit der Theorie der Signalverarbeitung im mehrdimensionalen Bereich erfassbar sind. Die Erkennungsmethoden zur Merkmalsextraktion und Klassifikation sind hingegen zumeist nichtlineare Verfahren, die nicht zwingend deterministisch arbeiten müssen. Hier findet ein fließender Übergang in die Forschungsbereiche der künstlichen Intelligenz und des maschinellen Lernens statt. Die hierzu notwendigen Methoden orientieren sich an der mathematischen Disziplin der Stochastik⁴.

Die gleichen Ansätze werden auch für Videodaten verwendet; hier enthalten die zu verarbeitenden Daten zusätzlich eine zeitliche Komponente.

2.1.1. Überblick über die Erkennungsverfahren

Der zu entwickelnde Algorithmus muss folgende Aufgabeschritte auf die bereitgestellten Videodaten durchführen:

- Die Position der Augen muss festgestellt werden (Augenlokalisierung) und
- der aktuelle Lidzustand muss erkannt werden.

Die Reihenfolge der Verarbeitungsschritte erfordert eine sequentielle Untergliederung: Es müssen zunächst die Augen lokalisiert werden, bevor der Lidzustand erkannt werden kann.

Zur Lokalisation der Augen müssen die Videodaten dergestalt abstrahiert werden, dass die Information, in welchen Bildregionen Augen sein könnten, für eine Entscheidung herangezogen werden kann. Dieser Fragestellung widmet sich der Forschungs-

⁴Vgl. [SBB02].

bereich der Gesichtserkennung. Im folgenden werden daher mehrere Konzepte zur Gesichtserkennung vorgestellt und auf ihre Anwendbarkeit auf das vorliegende Problem untersucht.

2.1.2. Gesichtserkennungsverfahren

Gesichtserkennungsverfahren lassen sich in mehrere Problembereiche unterteilen, je nachdem welches Ziel mit der Auswertung verfolgt werden soll:

Gesichtsidentifikation: Es soll aus der Bildinformation abgeleitet werden, ob und wenn ja welche Personen aus einer Datenbank abgebildet sind.

Gesichtsmerkmalsextraktion: Hier werden enthaltene Merkmale aus einem Gesicht geprüft, wie etwa die Position der Nase, des Mundes und der Augen.

Gesichtsverfolgung: Die einmal lokalisierte Position einer Person soll über die Zeit nachgeführt werden.

Bei der Lokalisation der Augen werden Merkmale, in diesem Fall die Position der Augen, in einem Gesicht ermittelt, so dass es sich hierbei um eine Gesichtsmerkmalsextraktion handelt.

Allen genannten Problembereichen ist gemein, dass die Abbildung eines Gesichts von stark veränderlichen Bedingungen abhängt. Folgende Faktoren sind daher bei der Erkennung zu berücksichtigen:⁵

Beleuchtung: Eine gute Ausleuchtung ist essentiell für die Bildaufnahme; sie kann jedoch von Bildaufnahme zu Bildaufnahme in der Intensität, räumlich, zeitlich (über den Tag hinweg) und auch farblich (Weißabgleich) variieren.

Optische Strecke: Der optische Aufbau der Aufnahmeeinheit, wie bspw. die Verzerrung der Linse, hat einen Einfluss auf die Abbildung.

Pose: Das Gesicht ist bezüglich der Aufnahmeperspektive dauernden Veränderungen unterworfen; selbst eine ruhig sitzende Person bewegt sich minimal.

Strukturelle Komponenten: Gesichtseigenschaften wie Brillen oder Bärte können hinzukommen oder verschwinden.

Gesichtsausdruck: Je nach Gemütslage der Person verändern sich der Gesichtsausdruck und somit die geometrischen Eigenschaften.

Sichtbarkeit: Eine Person kann durch Gegenstände oder andere Personen teilweise verdeckt sein.

⁵Vgl. [YKA], S. 34.

Ein Algorithmus ist dann von hoher Qualität, wenn seine Erkennungsleistung möglichst wenig von diesen Einflüssen abhängt.⁶ Neben der Effizienz ist diese Leistung eines der Hauptoptimierungsprobleme in der Forschung zu Gesichtserkennungsalgorithmen. Es existieren hierzu unterschiedliche Ansätze, die nach folgendem Schema⁷ eingeordnet werden können:

Wissensbasierte Methoden: Diese Methoden wenden ein auf menschliches Wissen und Beobachtung gestütztes Regelwerk zur Gesichtserkennung an. Zumeist enthalten die Regelwerke Aussagen über die Verhältnisse von Gesichtseigenschaften wie des Augen- und Nasenabstands zueinander.

Merkmalsunabhängige Ansätze: Algorithmen dieser Kategorie verfolgen das Ziel, Eigenschaften des Gesichts zu erkennen, die völlig unabhängig von äußeren Einflüssen sind, wie etwa den Hautton.

Schablonenvergleich: Hier werden standardisierte Schablonen von Gesichtern oder Gesichtsmerkmalen auf ein Bild angewendet und die Ähnlichkeit des Bildes mit dieser Schablone bewertet.

Erscheinensbasierte Methoden: Erscheinensbasierte Methoden nutzen Methoden der künstlichen Intelligenz wie neuronale Netze oder Hidden-Markov-Modelle, um aus einem Trainingssatz das Erscheinen von Gesichtern zu erlernen. Diese erlernten Modelle werden dann zur Erkennung von Gesichtern herangezogen.

Während der Recherche nach geeigneten Algorithmen wurden Ansätze aus den unterschiedlichen Kategorien betrachtet und auf ihre Eignung für die Aufgabenstellung untersucht. Im folgenden werden die betrachteten Algorithmen kurz zusammengefasst:

Verfahren nach Betke et al.: Das Verfahren nach Betke et al. ([BMM00]) arbeitet zweistufig und basiert auf der Erkennung von Farbtönen. Zunächst wird die Position des Kopfes bestimmt, indem im Bild nach Ähnlichkeiten mit Hauttönen gesucht wird. Innerhalb des so bestimmten Kopfbereichs werden dann die Augenpositionen durch das Suchen von Bereichen die dem weißen Farbton des Augapfels ähneln, bestimmt. Bei der Erkennung anhand der Ähnlichkeit von Bildbereichen mit Farbtönen handelt es sich um einen merkmalsunabhängigen Ansatz.

Verfahren nach Zhu: Das Verfahren nach Zhu ([ZQ]) basiert auf einem Schablonenvergleich; es ist allerdings ein aktives Verfahren. Durch eine gepulste Infrarotbeleuchtung wird eine Reflexion in der Pupille des Auges gesucht, deren Auftreten mit der Ansteuerung der Beleuchtung korreliert. Zur Stützung der Suche wird zusätzlich eine erscheinensbasierte Methode mit Hilfe von „Sup-

⁶Vgl. [YKA], S. 50.

⁷Vgl. [YKA], S. 35.

portVectorMachines“, die auf die Form des Auges abgestimmt sind, angewendet.

Verfahren nach Venkatesh: Das Verfahren nach Venkatesh ([VPY02]) macht sich die Tatsache zu Nutze, dass sich eine Person niemals ruhig verhält, sondern ihren Kopf immer leicht bewegt. Durch eine Differenzbildung aufeinanderfolgender Videobilder wird eine Kantenerkennung durchgeführt, so dass die Kanten die groben Umrisse des Kopfes repräsentieren. Der Inhalt dieses Bereichs wird dann mit unterschiedlichen Masken von Gesichtern auf eine Ähnlichkeit untersucht.

Verfahren nach Reinders: Reinders verfolgt in seiner Arbeit ([RKG97]) den Ansatz der Erkennung von Gesichtsmerkmalen über neuronale Netze. Das neuronale Netz ist dabei auf die Erscheinungsform der Augen traininiert und sucht den gesamten Bildbereich nach Ähnlichkeiten mit Augen ab. Das Verfahren ist ein Vertreter der erscheinensbasierten Methoden.

Verfahren nach Grauman: Das Verfahren nach Grauman ([GBGB]) arbeitet mit wissensbasierten Methoden. Durch eine Differenzbildung aufeinanderfolgender Bilder in einem Video werden alle Bewegungen herausgestellt. Alle Bewegungen werden mit anthroposophisch-geometrischen Merkmalen der Augenregionen verglichen, um die Augenpositionen herauszufiltern.

2.2. Evaluation der Verfahren

Nicht jedes der aufgeführten Verfahren ist für alle Problemstellungen der Bilderkennung gleichermaßen gut geeignet. Die Kriterien für eine gute Lösung von Bilderkennungsproblemen sind neben der Erkennungsleistung bei optimalem Training auch die Entscheidungsgeschwindigkeit, der Trainingsaufwand für den Algorithmus sowie die Belastung des Rechensystems. Dabei handelt es sich bei der Entscheidungsgeschwindigkeit um ein Zeitmaß, das bestimmt, wie lange ein Algorithmus durchschnittlich benötigt, um aus den Eingangsdaten eine Entscheidung zu treffen. Der Trainingsaufwand für einen Algorithmus kann dagegen von dem Aufstellen von Regeln bis zu dem umfangreichen Zusammentragen und Einlesen von Testdaten reichen. Schließlich unterscheiden sich die Algorithmen hinsichtlich ihrer Berechnungsaufwände, die mit der Komplexität dahinterliegender mathematischer Verfahren einhergehen und das Rechensystem unterschiedlich stark belasten.

Im vorliegenden Fall kommt als zusätzliches Kriterium hinzu, dass kostengünstige Standardhardware genutzt werden soll; somit kommen keine Methoden in Betracht, die spezielle Elektronik – wie etwa gepulste Infrarotbeleuchtung –, voraussetzen.

Eine weitere Besonderheit bei der Lidschlagerkennung ist die Reduktion der Auswertung auf die Zustände, ob das Lid geöffnet oder geschlossen ist. Hieraus ergibt

sich die Konsequenz, dass nur die Augenregion für die Auswertung notwendig ist, während andere Gesichtsmerkmale für die Aufgabenstellung nicht relevant sind.

Die oben vorgestellten Verfahren werden in Tabelle 2-3 anhand der aufgeführten Kriterien bewertet. Je nach Erfüllungsgrad eines Kriteriums wird einem Verfahren ein Wert von 1, 0 und -1 zugewiesen; hierbei steht 1 für eine positive Gewichtung und -1 für eine schlecht erfüllte Eigenschaft. Die Werte der Kriterien werden dann aufsummiert, um schließlich eine Entscheidung für ein Verfahren zu treffen.

Verfahren /Kriterien	Standard-hardware	Geschwindigkeit	Trainingsaufwand	Komplexität	Beschränkung auf Augen	Σ
Betke et. al.	1	1	1	0	-1	2
Zhu	-1	1	1	1	1	3
Venkatesh	1	0	0	-1	-1	-1
Reinders	1	-1	-1	0	1	0
Grauman	1	-1	1	1	1	3

Tabelle 2-3: Vergleich der Verfahren

Die Kriterien sprechen für die Verfahren nach Zhu und nach Grauman. Das Verfahren nach Zhu hat jedoch den gravierenden Nachteil, dass spezialisierte Infrarothardware angewendet wird, so dass auch dieses Verfahren nicht mehr weiter verfolgt wird. Somit kommt für die Aufgabenstellung nur noch das Verfahren nach Grauman in Betracht. Bei diesem Verfahren beschränkt sich die Lokalisation der Gesichtsmerkmale auf die Position der Augen..Diese Lokalisation wird über einfache zeitliche Zusammenhänge gekoppelt mit geometrischen wissensbasierten Ansätzen realisiert. Auch wenn die geometrischen Regeln auf bekannte Abbildungsgrößen ausgelegt werden und dadurch der Benutzer eingeschränkt wird, haben diese Ansätze mehrere Vorteile: So entsteht nur ein geringer Trainingsaufwand und eine hohe Erkennungsgeschwindigkeit ist möglich, da nur verhältnismäßig einfache Regelsätze berechnet werden müssen.

Die Zustandserkennung der Lider erfolgt im Verfahren nach Grauman über ein Kreuzkorrelationsverfahren mit Masken, was zu einer hohen Rechenlast führt. Die Rechenlast bei der Berechnung der Korrelation ist jedoch deterministisch, was eine Voraussetzung für die Bestimmung der Reaktionszeit eines Systems ist.

3. Methodenentwicklung

Das im folgenden beschriebene Verfahren basiert auf den theoretischen Arbeiten von Grauman et al. ([[GBGB](#)]). und wurde im Rahmen der Entwicklung zunächst prototypisch in einer MATLAB-Umgebung modelliert.

Die Aufgabenstellung ermöglicht eine Gliederung in die voneinander unabhängigen Teilaufgaben der Augenlokalisierung und der Augenverfolgung. Durch die Umsetzung und praktische Anwendung des Verfahrens auf Testdaten konnten dessen Schwächen ermittelt werden und entsprechende Verfahrensverbesserungen einfließen.

3.1. Algorithmenzerlegung

Das Verfahren teilt sich in die beiden Schwerpunkte des Lokalisierens der Augen und der anschließenden Bewertung des Öffnungszustands auf. Daher ist auch der Ablauf des Algorithmus entsprechend in mehrere Phasen aufgeteilt, die in dem Zustandsdiagramm [3-3](#) skizziert sind.

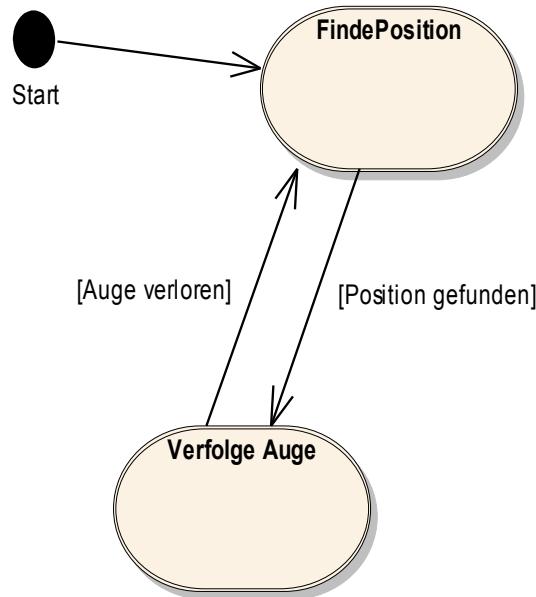


Abbildung 3-3: Verfahrenszustände

Die Teilaufgaben werden durch Systemzustände in einen Zustandsautomaten zusammengefasst, so dass sich eine sequentielle Abarbeitung ergibt. Nach dem Start befindet sich der Algorithmus im Zustand „Finde Augen“. Sobald die Position der Augen lokalisiert wurde, wechselt er in den Zustand „Verfolge Augen“. In diesem Zustand werden die Position der Augen verfolgt und der Öffnungszustand des Lids bestimmt. Wurde die Position der Augen verloren, so wechselt der Algorithmus

wieder zurück in den Zustand „Finde Augen“.

Der Daten- und Kontrollfluss für diesen Ansatz ist in dem Aktivitätsdiagramm 3-4 dargestellt.

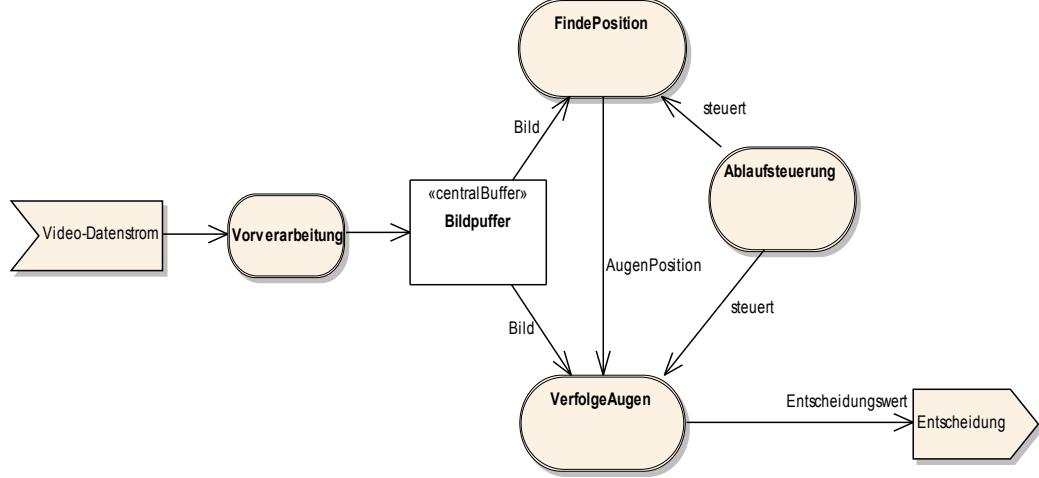


Abbildung 3-4: Daten- und Kontrollfluss

Der Datenfluss beginnt mit der Übergabe des Videodatenstroms. Dabei ist der Videodatenstrom eine Menge

$$\mathcal{V} := \{P_0(t_0), P_1(t_0 + T), \dots, P_{n-1}(t_0 + (n-1) \cdot T), P_n(t_0 + n \cdot T)\} \quad (3.1)$$

von Bildern, die in äquidistanten Zeitabständen T vorliegen. Die Anzahl n der Bilder in dem Datenstrom ist bei einem aufgezeichneten Video begrenzt, bei der Echtzeitverarbeitung jedoch undefiniert. Die einzelnen Videobilder p repräsentieren ein digitales Farbbild und sind definiert als ein 3-Tupel

$$P(t) := (R, G, B) \quad (3.2)$$

Jedes Element des Tupels aus Gl. 3.2 stellt eine Matrix der Größe $M \times N$ dar, die die digitalen Bilddaten für die einzelnen Grundfarben Rot, Grün und Blau enthalten.

Der Datenstrom wird zunächst vorverarbeitet. Liegen die Videodaten im Zeilensprungverfahren vor, so werden sie mit Hilfe eines Deinterlacing-Verfahrens zu Vollbildern zusammengesetzt. Danach werden die Videobilder, die als Farbbilder im RGB-Farbraum vorliegen, in Graustufenbilder gewandelt. Diese Reduktion von Bildinformation ist zweckmäßig, da ein Teil des Algorithmus nicht mit der Farbinformation arbeitet; somit können die Komponenten je nach Anforderung auf die Farb- oder auf die Graustufenbilder zugreifen. Die Umrechnung von Farb- nach Graustufenwerten berücksichtigt dabei bereits den Umstand, dass das menschliche Auge den Grünanteil für die Helligkeitswirkung stärker gewichtet. Demzufolge lautet die

Formel zur Konvertierung von dem RGB-Farbraum in Graustufen für einzelne Bildpunkte:

$$l = 0.299 \times R + 0.587 \times G + 0.114 \times B \quad (3.3)$$

Die RGB- und Grauwertbilder werden in einem Bildpuffer gespeichert. Dieser Bildpuffer hält sowohl das aktuelle Bild wie auch eine bestimmte Anzahl von Bildern aus der Vergangenheit –jeweils als Farb- und Graustufenbild–, vor. Das Vorhalten von älteren Bildern ist deshalb notwendig, weil die Lokalisation der Augen auf Informationen zugreift, die in der zeitlichen Abfolge der Bilder codiert sind. Der Bildpuffer wird als Ringpuffer realisiert; wenn er gefüllt ist, werden die ältesten Daten von vorne neu beschrieben.

Der Bildpuffer wird den beiden Erkennungsmethoden „FindePosition“ und „VerfolgeAugen“ bereitgestellt. Das Modul „FindePosition“ stellt die ermittelten Positionen der Augen für die Zustandserkennung bereit, während das Modul „VerfolgeAugen“ für die weitere Positionsverfolgung der Augen und das Versenden der Ergebnisse verantwortlich ist. Der sequentielle Ablauf beider Methoden wird durch eine Ablaufsteuerung synchronisiert.

3.2. Lokalisation der Augen

Die Lokalisation der Augen wird hier nach dem Verfahren der Detektion des unwillkürlichen Lidschlags nach Grauman durchgeführt. Die Arbeit von Grauman⁸ beschreibt eine Methode, die die Bewegung der Augenlider nutzt, um deren Position in einem Videobild zu erfassen.

Dieses Prinzip macht sich die Tatsache zu Nutze, dass der Mensch periodisch einen unwillkürlichen Lidschlag durchführt. Der Lidschlag wird ca. 20mal pro Minute durchgeführt und hat eine Gesamtlänge von unter 400ms; durch die unwillkürliche Handlung erfolgt der Wechsel von dem offenen in den geschlossenen Zustand unter 100ms. Abbildung 3-5 zeigt den Schließprozess in einer Abfolge von Bildern, die aus einer Videoaufnahme extrahiert wurden.

3.2.1. Differenzbildung

Man kann erkennen, dass innerhalb einer Sequenz von drei Bildern das Auge den Zustand von „offen“ zu „geschlossen“ wechselt. Die Aufnahme erfolgte mit 20 Bildern/s, was zu einer Bildzykluszeit von

$$T = \frac{1}{20 \frac{\text{Bilder}}{\text{s}}} = 50\text{ms} \quad (3.4)$$

⁸[GBGB], S. 34 – 35.



Abbildung 3-5: Unwillkürlicher Lidschlag

führt. Der Zustandswechsel wird also im vorliegenden Beispiel in $150ms$ vollzogen. Wird nun eine Differenzoperation auf die Graustufenbilder angewendet, so entsteht ein neues Bild, dessen Helligkeitswerte ein Maß für die Veränderung zwischen den Bildern darstellen. Diese Differenzoperation bildet dabei die absolute Differenz zwischen jedem einzelnen Bildpunkt i und j der beiden Bilder:

$$[\Delta P]_{i,j} = \left| [P_n]_{i,j} - [P_{n-2}]_{i,j} \right| \quad (3.5)$$

Für die Information, ob eine Veränderung vorhanden ist, ist es unerheblich, ob der Übergang von hell nach dunkel oder umgekehrt stattfindet; daher wird eine absolute Differenz gebildet. Es wird ein Abstand $d = 3$ der Bilder angenommen, woraus sich in der Gl. 3.5 die Indizes der Bilder zu n und $n - 2$ ergeben. Der Abstand wurde dergestalt gewählt, dass der Zustandswechsel zwischen offenem und geschlossenem Lid in dem Differenzbild vollständig erfasst wird und somit in einer maximalen Differenz resultiert.

Das Differenzbild ist in Abbildung 3-6 dargestellt.

Die Helligkeit in dem Differenzbild ist proportional zu der Veränderung zwischen den Ausgangsbildern. Hier werden also alle Bewegungen und damit auch die Bewegung des Augenlids hervorgehoben. Für die weitere Bewertung ist zunächst die Tatsache, dass sich etwas bewegt hat, interessant. In den Wertabstufungen des Differenzbildes steckt jedoch zudem noch die Information, mit welcher Intensität (Geschwindigkeit) sich etwas verändert hat. Um hier eine weitere Informationsreduktion zu erzielen, wird das Graustufen-Differenzbild mit einem Schwellwertverfahren in ein Binärbild überführt.

3.2.2. Die Binarisierung

Das Schwellwertverfahren ist ein Segmentierungsverfahren. Die Bildpunkte eines Bildes werden dabei unterschiedlichen Gruppen zugeordnet, wobei zur Erzeugung



Abbildung 3-6: Differenzbild eines unwillkürlichen Lidschlages

eines Binärbildes nur zwei Gruppen definiert werden: Diejenige Gruppe der Bildpunkte, deren Wert unterhalb eines Schwellwerts liegt und jene Gruppe, deren Wert größer oder gleich dem Schwellwert ist. Die Transformationsvorschrift S für jeden Bildpunkt $[P]_{i,j}$ lautet also:

$$S([P]_{i,j}) = \begin{cases} p_{min} & \text{wenn } [P]_{i,j} < s \\ p_{max} & \text{wenn } [P]_{i,j} \geq s \end{cases} \quad (3.6)$$

Wenn der Wert eines Bildpunktes die Schwelle s erreicht, so wird dem Bildpunkt der maximale Wert im Wertebereich p_{max} zugewiesen; ansonsten erhält der Bildpunkt den minimale Wert p_{min} . Der Wert für die Schwelle s muss experimentell ermittelt werden. Zur Bestimmung des Schwellwerts wurden daher sechs Differenzbilder von Lidbewegungen als Basis ausgewählt, deren Ursprungsbilder durchschnittlich gut ausgeleuchtet waren. Die Schwelle wurde gerade so niedrig gewählt, dass die Lidbewegung im Binärbild erhalten blieb. Der so experimentell ermittelte Wert aus den Versuchsbildern führt zu einer Festlegung von $s = 10\%$ von p_{max} . Die einzelnen ermittelten Werte werden in Kapitel B.1.1 näher dargestellt.

Das Ergebnis der Binarisierung aus dem Differenzbild aus Abbildung 3-6 ist in Abbildung 3-7 dargestellt.

3.2.3. Morphologische Filterung

Das binarisierte Bild hebt nun deutlich die Augenregion vor; es enthält jedoch noch einige unerwünschte Eigenschaften, die in Abbildung 3-6 gekennzeichnet sind:



Abbildung 3-7: Binarisiertes Differenzbild eines unwillkürlichen Lidschlages

1. Leichte Bewegungen des Kopfes führen schon zu starken Bilddifferenzen entlang der Kopfform, diese Bewegungselemente sind von eher länglicher aber eher schmaler Natur.
2. Das Kamerarauschen bzw. die Kompressionsartefakte führen ebenfalls zu Differenzen, die als kleinere Punkte sichtbar sind.
3. Schließlich ist die Augenregion nicht als durchgehende Fläche vorhanden, sondern enthält „Ausfransungen“, da Teile der Bewegungsdifferenz unterhalb der Binarisierungsschwelle liegen können.

Diese Störungen können insgesamt dazu führen, dass mit dem Verfahren nicht entschieden werden kann, ob Augenregionen im Differenzbild enthalten sind. Einerseits können die Eigenschaften 1 und 2 dafür verantwortlich sein, dass eine zu große Menge von irrelevanten Regionen in den Entscheidungsprozess mit einbezogen werden müsste. Andererseits kann Eigenschaft 3 dazu führen, dass es unmöglich wird, eindeutige Kriterien für Augenregionen zu finden, da die Fläche, die die Augenregionen beschreibt, zuviele unterschiedliche Muster annehmen kann.

Zur Reduktion dieser Probleme wird das Binärbild vor der Verarbeitung durch den Entscheidungsprozess mit Hilfe von morphologischen Operatoren gefiltert. Eine hierfür geeignete Operation ist das „Öffnen“. Der Öffnungsoperator führt die elementaren Operationen Erosion und Dilation⁹ hintereinander aus.

Die Erosion entfernt dabei zunächst kleinere Störungen und auch schmale Kanten; danach verbindet die Dilation Ausfransungen von ursprünglich zusammengehörenden

⁹Dazu [SSSS01], S.75ff.

den Flächen dann wieder miteinander.

Die Wahl eines geeigneten Strukturelements erfolgte hierbei ebenfalls experimentell anhand der vorhandenen Differenzbilder; deren Auswertung wird in Kapitel B.1.2 näher erläutert. Die Auswahl der in Frage kommenden Strukturelemente kann anhand einiger besonderer Störungseigenschaften eingeschränkt werden:

1. Die Störungen, die durch Kanten der Kopfbewegung verursacht werden, verlaufen linienförmig mit geringer Dicke; dies kann im Strukturelement dadurch berücksichtigt werden, dass es eine Ausdehnung sowohl in vertikaler als auch in horizontaler Richtung enthält. Dadurch können die linienförmigen Kanten durch den Erosionsprozess entfernt werden.
2. Die Rausch- und Kompressionsartefakte sind meist Flächen mit Ausdehnungen kleiner fünf Bildpunkte; somit werden durch die Erosion diese Flächen entfernt, wenn das Strukturelement selbst eine größere Fläche einnimmt.
3. Im Dilationsprozess werden die Flächen wieder zusammengeführt. Hier ist es zweckmäßig, die Ausdehnung des Strukturelements bezüglich der Fläche nicht wesentlich größer werden zu lassen als die größten zu entfernenden Flächen.

Das hieraus resultierende Strukturelement $M_e = (\text{Scheibe}, 5)$ wird in Abbildung 3-8 dargestellt.

0	0	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	0	0

Abbildung 3-8: Binäres Strukturelement

Das Strukturelement ist scheibenförmig angelegt mit einer Wirkfläche von $A_{struct} = 69$ Bildpunkten.

Abbildung 3-9 zeigt die Anwendung der Öffnungsfunktion auf das Differenzbild bestehend aus den Einzelschritten.

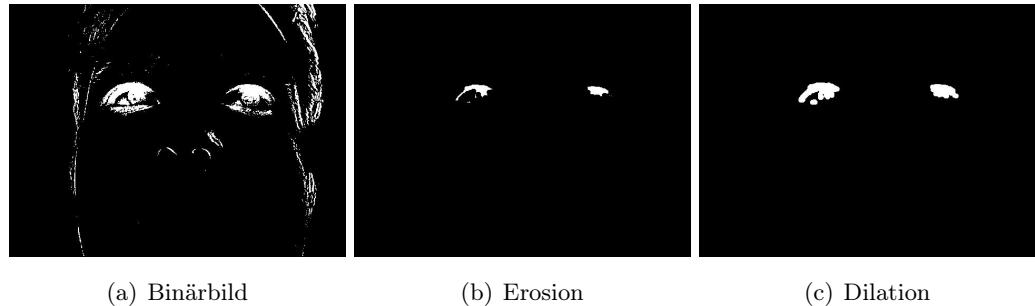


Abbildung 3-9: Anwendung der morphologischen Operatoren

Durch die Scheibenform werden bei der Erosion die Rauschartefakte sowie die störenden Kanten durchgehend entfernt, so dass nur noch die Augenflächen übrig bleiben. Deren Gesamtfläche wird durch den Dilationsoperator geglättet.

Eine detailliertere Ansicht des resultierenden Binärbildes nach der Anwendung der morphologischen Operationen ist in Abbildung 3-10 zu sehen. Dieses Binärbild bildet das Eingangssignal für den nächsten Schritt im Gesamtablauf der Entscheidungsfindung.



Abbildung 3-10: Morphologisch bearbeitetes Binärbild

3.2.4. Ermittlung der Augenregion

Im Entscheidungsprozess muss aus den binären Bilddaten die Information extrahiert werden, ob die Daten einen Lidschlag zeigen. Die Bilddaten beinhalten nur noch abstrakte Muster, die zusammenhängende Flächen darstellen. Daher muss der

Entscheider die Informationen aus den geometrischen Eigenschaften dieser Flächen beziehen.

Zu diesem Zweck wurde ein Algorithmus entwickelt, der die geometrischen Eigenschaften der Augenlider modelliert und einen Vergleich der Bildinformation mit dem hinterlegten Modell durchführt.

Die weitere Verarbeitung erfolgt nun nicht mehr auf der Basis einzelner Bildpunkte, sondern bezieht sich vielmehr auf abstrakte Muster. Die Überführung in abstrakte Muster wird dabei durch die Zuordnung der Bildpunkte in zusammenhängende Regionen mit Hilfe eines sogenannten *Labeling*-Verfahrens realisiert. Mit diesem Verfahren werden die Bildpunkte, die eine Nachbarschaft mit anderen Bildpunkten eingehen und somit eine einheitliche Fläche bilden, mit einer gemeinsamen Markierung versehen, dem sog. Label.¹⁰ Dieses Label ist eine fortlaufende Nummerierung.

Die Definition, welcher Punkt ein Nachbar eines Bildpunktes ist, kann auf zwei Arten erfolgen, einerseits als 4er-Nachbarschaft und andererseits als 8er-Nachbarschaft.

Beide Nachbarschaftsarten sind in Abbildung 3-11 skizziert.

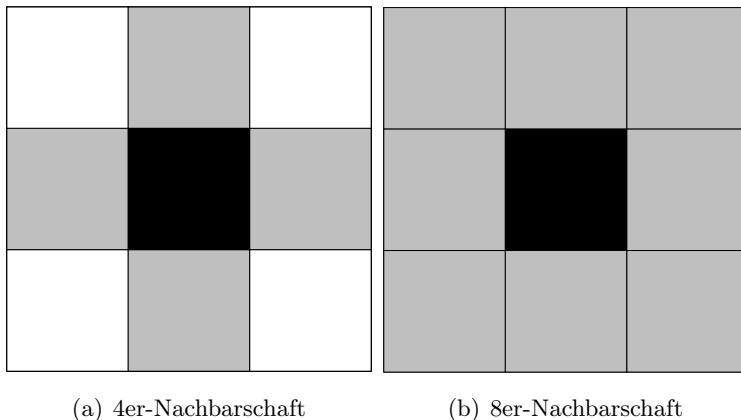


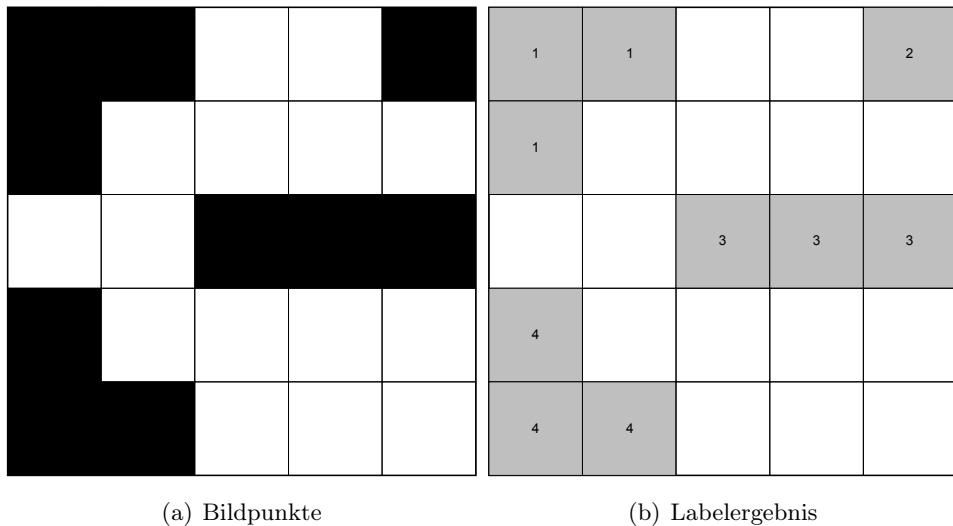
Abbildung 3-11: Nachbarschaftsarten

Der Referenzbildpunkt ist hierbei $p(r, c)$, wobei r den Zeilenindex und c den Spaltenindex innerhalb der Bildmatrix angibt. Die 4er-Nachbarschaft ist somit definiert als die Nachbarbildpunkte $p_n(r - 1, c)$, $p_n(r + 1, c)$, $p_n(r, c - 1)$ und $p_n(r, c + 1)$. Bei der 8er-Nachbarschaft werden zusätzlich noch die zum Referenzbildpunkt diagonal liegenden Punkte $p_n(r - 1, c - 1)$, $p_n(r - 1, c + 1)$, $p_n(r + 1, c - 1)$ und $p_n(r + 1, c + 1)$ mit berücksichtigt.

Die Abbildung 3-12 zeigt die beispielhafte Anwendung des Labeling.

Die Bildinformation liegt nun in einer Menge von Regionen R vor, die geometrische

¹⁰[SSSS01], S. 69ff.



(a) Bildpunkte

(b) Labelergebnis

Abbildung 3-12: Beispiel des Labeling-Verfahrens

Eigenschaften besitzen:

$$\mathcal{R} := \{R_0, R_1, \dots, R_n\} \quad (3.7)$$

Jeder Region R_i wird durch den Labeling-Prozess eine fortlaufende Nummerierung im Bereich $0 \dots n$ zugewiesen; somit können die optisch zusammengehörenden Flächen im Programmlauf voneinander unterschieden werden. Zur Veranschaulichung wurden die unterschiedlichen Labels der Augenregionen mit Farbwerten versehen und in Abbildung 3-13 gezeichnet. Diese Regionen werden in der Bildverarbeitung

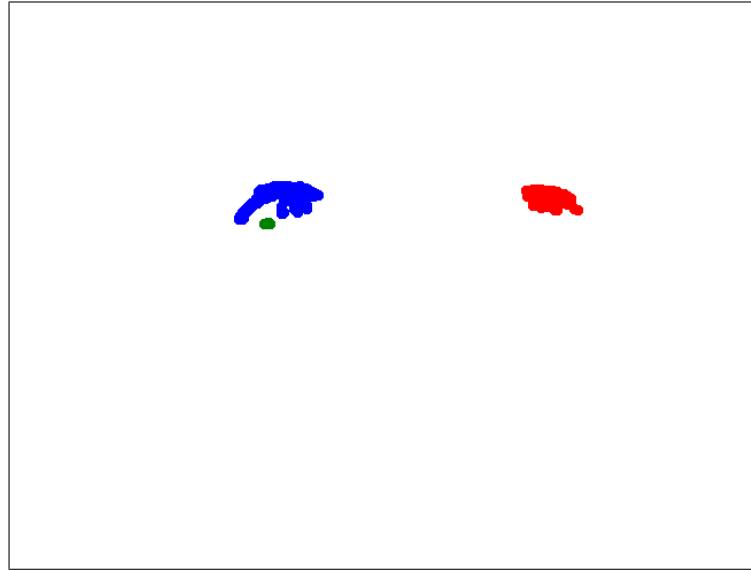


Abbildung 3-13: Ein mit Labels versehenes Binärbild

auch als *Blobs* bezeichnet. Jeder Blob besitzt nun geometrische Eigenschaften, die aus den vorliegenden Binärdaten errechnet werden können:

Fläche $A(R_i)$: Die Fläche $A(R_i)$ beschreibt die Summe aller zur Region gehörenden Bildpunkte:

$$A(R_i) = \sum_{(n,m) \in R_i} 1 \quad (3.8)$$

umgebendes Rechteck B : Das umgebende Rechteck, auch BoundingBox genannt, entspricht den Koordinaten eines Rechtecks, das so groß ist, dass alle Punkte der Region in ihm liegen. Das umgebende Rechteck wird somit definiert durch die Koordination der oberen linken B_{ol} und der unteren rechten Ecke B_{ur} :

$$B_{ol}(R_i) = (\min \{n \mid (n, m) \in R_i\}, \min \{m \mid (n, m) \in R_i\}) \quad (3.9)$$

$$B_{ur}(R_i) = (\max \{n \mid (n, m) \in R_i\}, \max \{m \mid (n, m) \in R_i\}) \quad (3.10)$$

Weitere Eigenschaften des umgebenden Rechtecks sind dessen Breite und Höhe:

$$w_B(R_i) = \max \{n \mid (n, m) \in R_i\} - \min \{n \mid (n, m) \in R_i\} \quad (3.11)$$

$$h_B(R_i) = \max \{m \mid (n, m) \in R_i\} - \min \{m \mid (n, m) \in R_i\} \quad (3.12)$$

Centroid $C(R_i)$: Der Centroid $C(R_i)$ beschreibt den Schwerpunkt der Masse, die die Region bildet:

$$\bar{n} = \frac{1}{A(R_i)} \sum_{(n,m) \in R_i} n \quad (3.13)$$

$$\bar{m} = \frac{1}{A(R_i)} \sum_{(n,m) \in R_i} m \quad (3.14)$$

Mittelpunkt $M(R_i)$: Der Mittelpunkt $M(R_i)$ gibt den Punkt in der Mitte des umgebenden Rechtecks an.

Seitenverhältnis $Ar(R_i)$: Das Seitenverhältnis $Ar(R_i)$ gibt den Verhältnisfaktor von Breite zu Höhe des umgebenden Rechtecks an:

$$Ar(R_i) = \frac{w_B(R_i)}{h_B(R_i)} \quad (3.15)$$

Zur vollständigen Bildung eines anthropologischen Modells der Augenregionen müssen weitere Eigenschaften berücksichtigt werden, die sich aus der Verknüpfung zweier Regionen zu Paaren ergeben. Diese paarweise Zuordnung zu Tupeln T_R lässt sich mathematisch beschreiben als:

$$T_{R_{i,j}}(R_i, R_j) \in \mathcal{R} \times \mathcal{R} : R_i < R_j \quad (3.16)$$

Die zusätzlichen Eigenschaften der Tupel, die sich aus der Verknüpfung der Regionen ergeben, sind:

Abstand $d(T_{R_i,j})$: Der Abstand $d(T_{R_i,j})$ gibt die Distanz zwischen den Mittelpunkten $M(R_i)$ und $M(R_j)$ an. Er ist definiert als

$$d(T_{R_i,j}) = \sqrt{(M_x(R_i) - M_x(R_j))^2 + (M_y(R_i) - M_y(R_j))^2} \quad (3.17)$$

Winkel $\alpha(T_{R_i,j})$: Der Winkel $\alpha(T_{R_i,j})$ beschreibt den Winkel zwischen der Geraden, die durch die Mittelpunkte $M(R_i)$ und $M(R_j)$ der Regionen definiert ist und der Bildebene. Er wird wie folgt berechnet:

$$\alpha(T_{R_i,j}) = \arcsin \left(\frac{M_y(R_i) - M_y(R_j)}{d(T_{R_i,j})} \right) \quad (3.18)$$

Normierte Breite der Regionen $w(R_i)$ und $w(R_j)$: Die normierte Breite der Regionen setzt die Breite des umgebenden Rechtecks mit dem Abstand $d(T_{R_i,j})$ der Regionen ins Verhältnis :

$$w(R_i) = \frac{B_w(R_i)}{d(T_{R_i,j})} \quad (3.19)$$

$$w(R_j) = \frac{B_w(R_j)}{d(T_{R_i,j})} \quad (3.20)$$

Erfüllungsgrad der Gleichheit der Seitenverhältnisse $g_{A_r}(T_{R_i,j})$: Der Erfüllungsgrad für die Gleichheit der Seitenverhältnisse ist ein Indikator, inwieweit die Seitenverhältnisse der Regionen miteinander identisch sind. Der Erfüllungsgrad nimmt einen Wert zwischen $0 < g_{A_r} \leq 1$ an, wobei eins bedeutet, dass die Seitenverhältnisse identisch sind.

Erfüllungsgrad der Gleichheit der Regionenflächen $g_A(T_{R_i,j})$: Dieser Erfüllungsgrad beschreibt die Gleichheit der Flächen der Regionen zueinander. Auch er nimmt einen Wert zwischen $0 < g_A \leq 1$ an, wobei 1 wiederum bedeutet, dass die Flächen identisch sind.

In Abbildung 3-14 sind diese Eigenschaften nochmals am Beispiel skizziert.

Abbildung 3-15 zeigt die Paarbildung; hier wird durch Linien verdeutlicht, dass die einzelnen Regionen Paare miteinander bilden.

3.2.5. Filterung der Augenregion

Die Eigenschaften der einzelnen Regionen sowie der Paare werden nun gefiltert. Diese Filterung enthält das anthropologische Modell der Augenregionen und führt dazu, dass nur noch ein Regionenpaar ausgewählt wird, dessen Merkmale wiederum auf einen Lidschlag hinweisen müssen. Diese Parameter wurden empirisch festgelegt. Im vorliegenden Fall wurden dazu die oben genannten Eigenschaften zu zehn Lidschlagereignissen tabellarisch erfasst und sortiert. Die Parametrisierung des Modells ist in Kapitel B.1.4 näher aufgeführt.

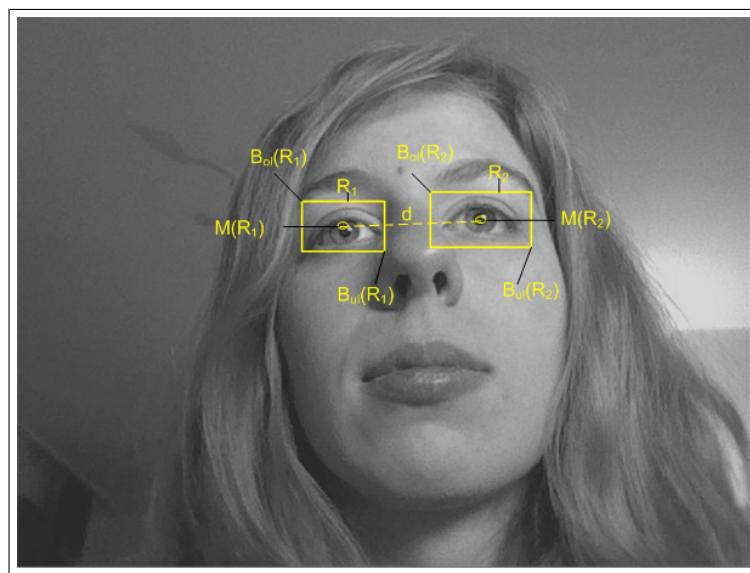


Abbildung 3-14: Eigenschaften der Regionen

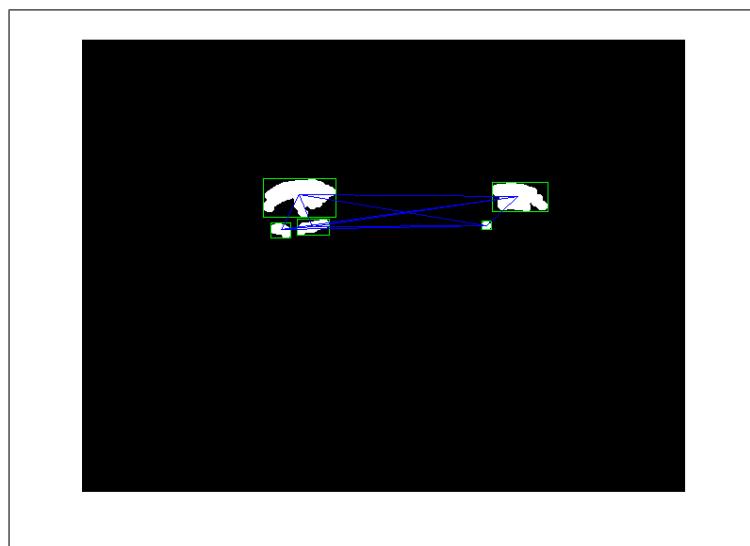


Abbildung 3-15: Paarbildung der Regionen

Die Filterkette umfasst die Kriterien nach Abbildung 3-16.

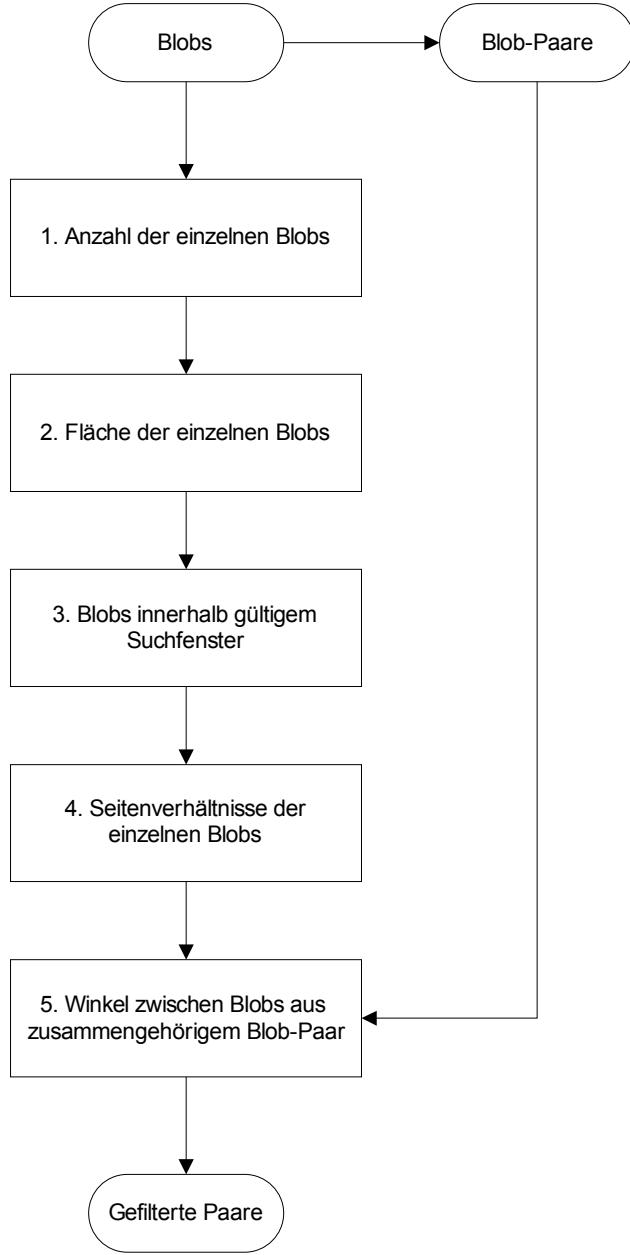


Abbildung 3-16: Filterkette der Regionen

1. Es wird zunächst die Anzahl der ermittelten Regionen (Blobs) gefiltert. Wenn zuwenige oder zuviele Blobs gefunden wurden, so wird die weitere Suche für das vorliegende Differenzbild abgebrochen. Zuviele Blobs ($n > 10$) deuten nämlich darauf hin, dass andere Bewegungskomponenten im Bild vorhanden sind, die nicht alleine von dem Lidschlag herrühren. Bei weniger als $n = 2$ Blobs ist dagegen keine Paarbildung mehr möglich und eine weitere Betrachtung somit zwecklos.

2. Wenn man annimmt, dass sich die Abbildungsgröße der Augen nur in einem bestimmten Bereich befindet, können die entstehenden Flächen für die Augenregionen auch nur in diesem Bereich liegen. Dieser Filterschritt sortiert somit alle Blobs aus, die außerhalb der festgelegten Grenzen für die Größe der Augenregionen liegen. Die Grenzen für gültige Flächengrößen sind dabei auf $0,25\% \leq A(R_i) \leq 0,8\%$ der Gesamtfläche des Bildes festgelegt.
3. Im nächsten Schritt werden die übriggebliebenen Blobs daraufhin gefiltert, ob ihr Mittelpunkt innerhalb eines Suchfensters liegt. Hier wird wiederum die Annahme getroffen, dass die Kameraausrichtung die Augen in der Mitte des Bildes und nicht am äußeren Rand abbildet. Das gewählte Suchfenster beschneidet die Bildgröße um 10%.
4. Daraufhin werden die Blobs nach ihren Seitenverhältnissen gefiltert. Diese Filterung lässt bereits anthropologische Merkmale einfließen, nämlich, dass die Augenregionen nicht beliebige Seitenverhältnisse annehmen können; es ist beispielsweise nicht möglich, dass eine Augenregion deutlich höher als breit ist. Das Seitenverhältnis ist begrenzt auf $1 \leq Ar(R_i) \leq 3,5$.
5. Der letzte Schritt bezieht die Filterung nicht mehr auf einzelne Blobs, sondern berücksichtigt Eigenschaften von Blob-Paaren. Hier wird der Winkel in der Bildebene der Paare zueinander bewertet. Es wird vorausgesetzt, dass der Benutzer den Kopf nur innerhalb bestimmter Winkel neigt; daher werden Paare, die diesen Winkel unter- oder überschreiten herausgefiltert. Die Winkel sind begrenzt auf $0^\circ \leq \alpha(T_{R_{i,j}}) \leq 18^\circ$.

Es liegt nun eine Anzahl von Blob-Paaren vor, die den Filterkriterien entsprechen. Zur weiteren Eingrenzung auf die Augenregionen werden die verbleibenden Paare anhand ihrer Eigenschaften gewichtet. Zu diesem Zweck wird aus weiteren Eigenschaften ein Bewertungsindex I gebildet, der einen Mindestwert erreichen muss. Erreichen mehrere Paare den Mindestwert, so ist das Paar mit dem höchsten Wert der Kandidat für die Augenregionen. In Abbildung 3-17 wird dargestellt, wie der Bewertungsindex gebildet wird. Die Bewertung erfolgt dabei anhand folgender vier Kriterien:

1. Die Gleichheit der Seitenverhältnisse $g_{Ar}(T_{R_{i,j}})$ geht direkt mit einer Gewichtung zwischen 0..1 in die Bewertung ein. Dadurch wird dasjenige Paar bevorzugt, dessen Seitenverhältnisse ähnlich zueinander sind.
2. Weiterhin wird direkt die Gleichheit der Regionenflächen $g_A(T_{R_{i,j}})$ dem Bewertungsindex aufgeschlagen. Hier werden wiederum diejenigen Paare bevorzugt, deren Flächen ähnlich zueinander sind.
3. Ein weiteres Kriterium ist die normierte Breite $w(R_i)$ für die erste Region eines Paars. Die normierte Breite beschreibt das Verhältnis zwischen der Breite

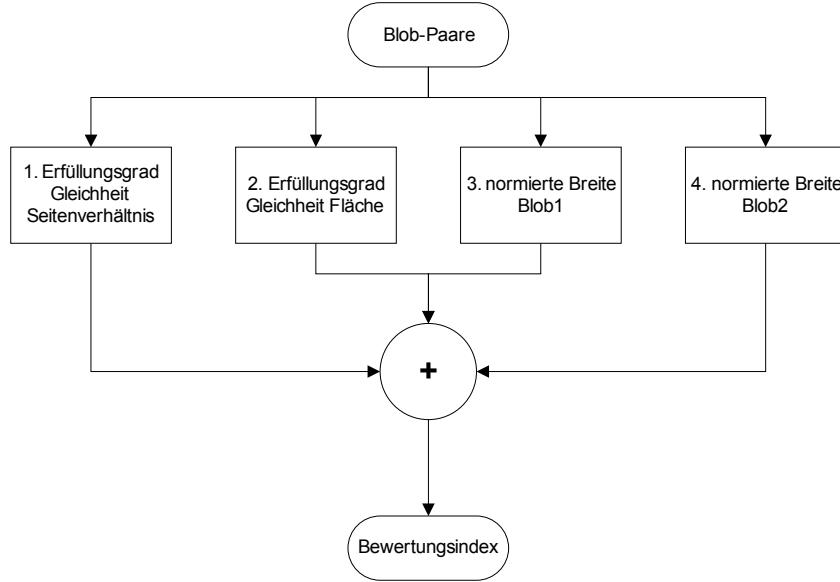


Abbildung 3-17: Berechnung des Bewertungsindex

der Augenregionen und dem Abstand der beiden Regionen. Im Durchschnitt liegt sie bei einem Wert von $w = 0,32$; der sinnvolle Wertebereich reicht von $w = 0,2$ bis $w = 0,5$. Die normierte Breite wird nach Abbildung 3-18 so skaliert, dass der optimale Wert von $w = 0,32$ in dem höchsten Aufschlag von 1 resultiert, während andere Werte entsprechend niedriger skaliert werden.

4. Als letztes wird dem Bewertungsindex schließlich die normierte Breite $w(R_j)$ für die zweite Region des Paars aufgeschlagen.

Ein Paar gilt dann als potentielles Augenpaar, wenn der Bewertungsindex einen Wert von $I = 2,9$ überschreitet. Das Paar mit dem höchsten Bewertungsindex repräsentiert nun jene Regionen im Bild, an denen ein Lidschlag stattfand. Die weiteren Prozessschritte greifen nun auf diese Information zurück.

Sollte kein gültiges Augenpaar gefunden werden, wird in den nachfolgenden Bildern nach potentiellen Augenpaaren weitergesucht.

Abbildung 3-19 skizziert zusammenfassend die Verarbeitungsschritte für die Lokalisierung der Augen.

3.3. Augenverfolgung und Zustandsbestimmung

Nachdem die Augenregionen lokalisiert wurden, sorgt die Ablaufsteuerung für einen Wechsel in den Systemzustand „Augenverfolgung“ und „Lidzustandsbestimmung“. In diesem Systemzustand wird zunächst aus den lokalisierten Regionen diejenige mit der größten Fläche ausgewählt. Im weiteren Programmablauf wird dann der

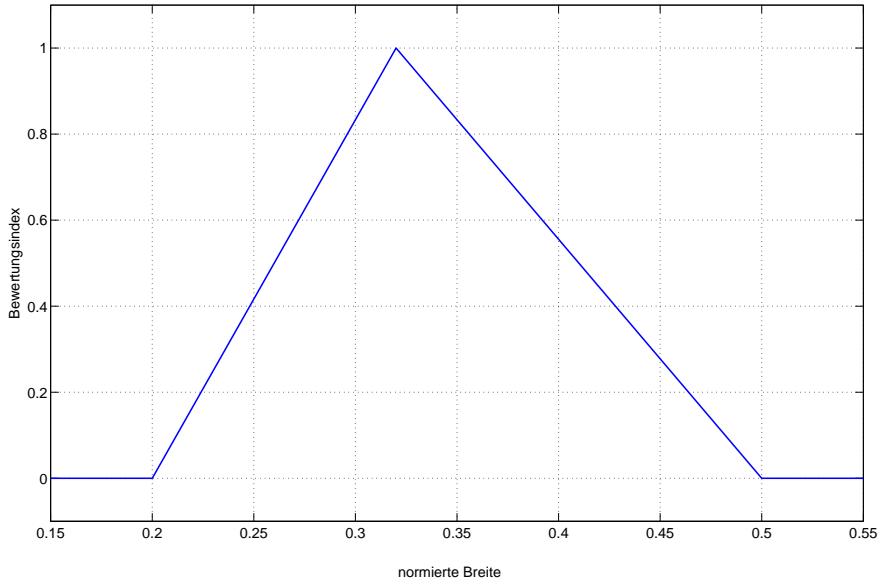


Abbildung 3-18: Skalierung der normierten Breite

aktuelle Öffnungszustand des gewählten Auges bestimmt sowie dessen aktuelle Position mitgeführt. Ist eine Mitführung der Position nicht mehr akkurat möglich, so wird der Systemzustand wieder zurückgesetzt und der Programmlauf setzt bei der Lokalisation der Augen neu an.

Grundlegende Konzepte zur Lösung dieser Problemstellung wurden in den Arbeiten von Betke ([BGF]) veröffentlicht. Das Verfolgen und die Zustandsbestimmung werden dort nach dem Prinzip der Kreuzkorrelation gelöst: Ein Bildbereich wird auf eine Ähnlichkeit mit einer zuvor festgelegten Maske des offenen Auges hin verglichen. Das Korrelationsverfahren resultiert in einem Wert, der angibt, wie gut die Maske in einen Bildbereich hineinpasst (korreliert). Der Punkt im Bildbereich, in dem das Maximum der Korrelation auftritt, entspricht somit demjenigen Punkt, an dem die Maske und das Bild am besten übereinstimmen. Zudem enthält der Korrelationswert ein Maß, wie gut die Maske überhaupt passt; ein sehr hoher Korrelationswert deutet dabei darauf hin, dass das Maskenbild in dem Bildbereich annähernd vollständig vorhanden ist.

3.3.1. Initialisierung der Verfolgung

In einem ersten Ansatz wurde angenommen, dass die Augenlokalisierung ein gültiges Regionenpaar beim Zustandsübergang vom offenen zum geschlossenem Lid findet, d.h. dass das aktuell im Videodatenstrom anliegende Bild P_n die gerade geschlossenen Augen repräsentiert. Das Bild, das im Datenstrom zwei Zeiteinheiten vorher

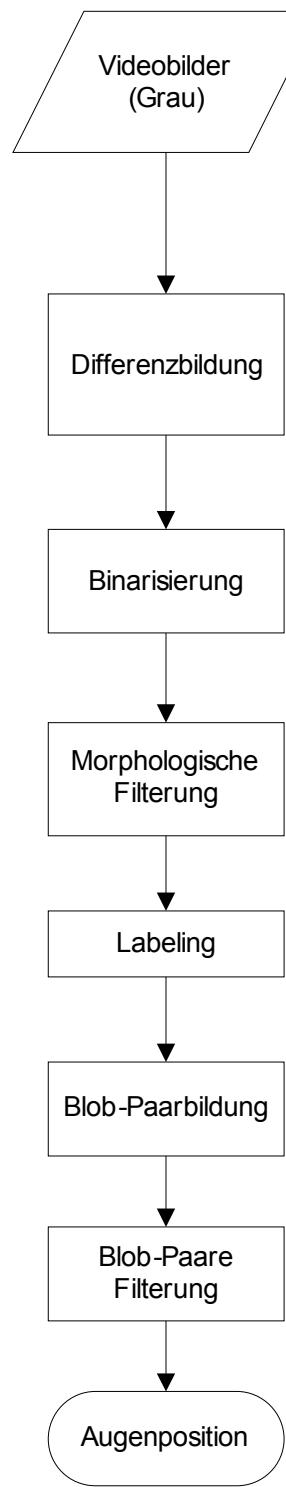


Abbildung 3-19: Ablauf der Lokalisation

anlag (P_{n-2}), enthält somit die offenen Augen. In den abgebildeten Daten ist die Region mit der größten Fläche die linksseitige, weshalb im folgenden auch nur noch diese Region betrachtet wird.

Die Maske des offenen Auges kann also durch das Extrahieren des Bereiches, der durch die Lokalisation ermittelt wurde, aus dem alten Bild generiert werden. Da durch die Differenzbildung nur diejenigen Teile mit der größten Bewegungsintensität hervorgehoben werden, ist es möglich, dass die ermittelten Regionen nicht den vollständigen Augenbereich erfassen. Daher wird die Größe der aus den Bildern zu extrahierenden Maske um das 1,5fache vergrößert. Abbildung 3-20 zeigt die Maskenextraktion für das aktuelle Bild P_n und Abbildung 3-21 für das vorherige Bild P_{n-2} . Dabei sind im Ausgangsbild die detektierten Regionen grün und die resultierenden Maskengrößen rot markiert.

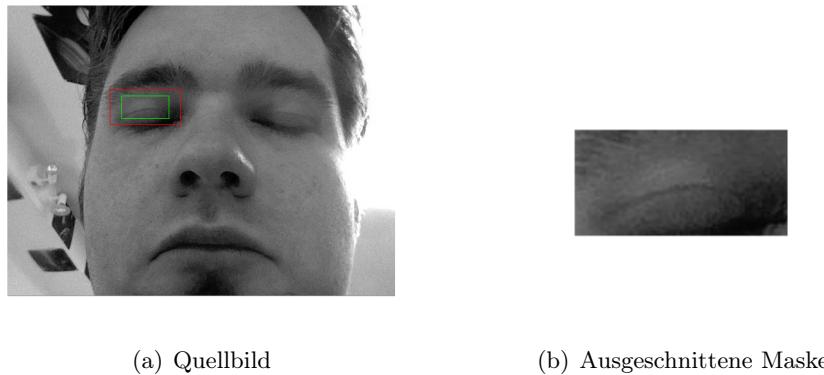


Abbildung 3-20: Maskenbildung des aktuellen Bildes P_n

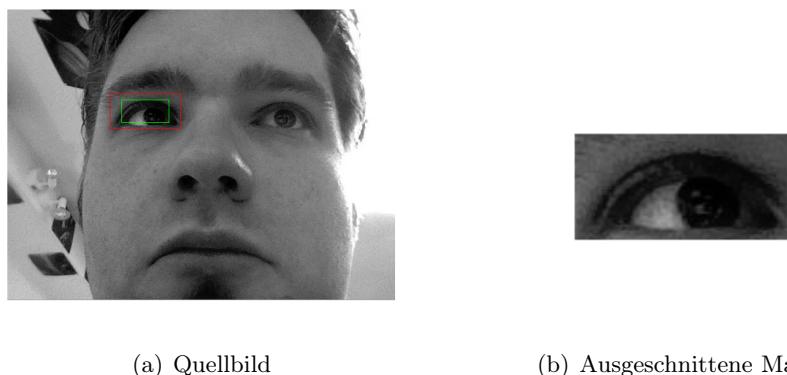


Abbildung 3-21: Maskenbildung des vorhergehenden Bildes P_{n-2}

Bei den ersten Versuchen hat sich jedoch eine gravierende Schwäche dieses Verfahrens gezeigt: Die Annahme, dass der Zustandsübergang von offen nach geschlossen durch die Lokalisation gefunden wird, ist nicht immer wahr. Es kann nämlich auch

der umgekehrte Übergang von geschlossen nach offen stattfinden. Eine aus diesen Ausgangsdaten generierte Maske enthält somit fälschlicherweise ein Bild des geschlossenen Auges, was eine Fehlerkennung zur Folge hat.

Zur Lösung dieses Problems ist es daher notwendig, ein Kriterium zu finden, das es ermöglicht, zu bewerten, welches der Bilder das geöffnete Auge zeigt.

Eine herausragende Charakteristik des Auges ist die Tatsache, dass der Augapfel bei gesunden Menschen eine weiße Farbe besitzt. Der weiße Augapfel wird vom hautfarbenen Lid im geschlossenen Zustand vollständig überdeckt. Eine zielführende Methode zur Unterscheidung des Öffnungszustands wäre demnach die Auswertung, ob ein weißer Anteil in der Bildmaske vorhanden ist.¹¹

Zur Bestimmung des Weißanteils wird ab diesem Punkt im Verfahrensablauf auf die aus der Kamera vorliegende Farbinformation zurückgegriffen. Die Bilder liegen im RGB-Farbraum vor, d.h. jeder Bildpunkt $p(R, G, B)$ besteht aus einem 3-Tupel, das die Farbe des Punktes in einem Farbwürfel mit den Kanten Rot, Grün und Blau festlegt (vgl. Abbildung 3-22). Die für die weitere Betrachtung interessante Information

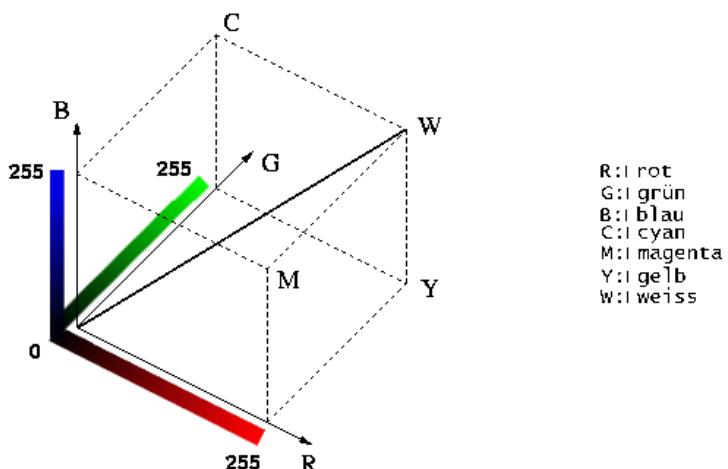


Abbildung 3-22: RGB Farbwürfel¹²

liegt jedoch nicht im Farbanteil vor, sondern vielmehr in der Farbsättigung. Letztere ist ein Maß dafür, wie stark ein Bildpunkt eine Farbe wiedergibt (Buntheit). Eine geringere Sättigung entspricht folglich einer farblosen, also weißen, Farbinformation. In der Farbenlehre existieren neben dem RGB-Farbraum weitere Farträume, die die Farbinformation auf unterschiedliche Weise in Komponenten zerlegen. Ein Farbraum, der als Teilkomponente die Farbsättigung enthält, ist der HSV-Farbraum, in dem das 3-Tupel aus den folgenden Komponenten besteht:

¹¹[BMM00], S. 2f.

¹²Quelle: GIMP 2.6.5 User-Manual.

- Hue (Farbton),
- Saturation (Sättigung) und
- Value (Intensität).

Die Farbe des Bildpunktes im HSV-Farbraum definiert sich dabei aus einem Kegel, dessen Kreis den Farbton angibt. Die Distanz von Grundfläche zu Spitze entspricht der Intensität der Farbe, während der Abstand zwischen Lot und Außenkreis ihre Sättigung darstellt. Somit kann ein RGB-Punkt nach der Vorschrift

$$H = \begin{cases} H_1 & \text{wenn } B \leq G \\ 2\pi - H_1 & \text{wenn } B > G \end{cases} \quad (3.21)$$

$$S = \frac{\max(R, G, B) - \min(R, G, B)}{\max(R, G, B)} \quad (3.22)$$

$$V = \frac{\max(R, G, B)}{255} \quad (3.23)$$

mit

$$H_1 = \arccos \left(\frac{0,5 [(R - G) + (R - B)]}{\sqrt{(R - G)(R - G) + (R - B)(G - B)}} \right) \quad (3.24)$$

in einen HSV-Punkt überführt werden. Abbildung 3-23 zeigt den HSV-Farbraum.

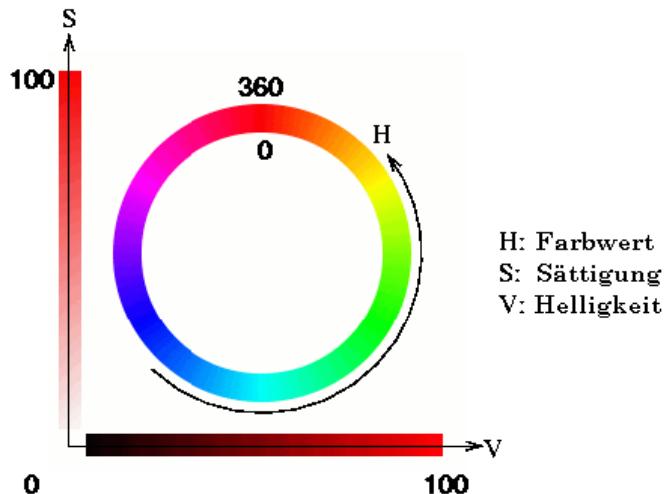


Abbildung 3-23: HSV-Farbraum¹³

Die ausgeschnittenen Masken aus dem Vorgängerbild und dem aktuellen Bild werden somit in den HSV-Farbraum gewandelt und die Sättigungsebene der Bilder als neues Grauwertbild extrahiert. Dieses Grauwertbild zeigt nun an den weißen Stellen des Augapfels einen dunklen Wert, da hier keine Farbsättigung vorliegt. Abbildung 3-24 zeigt die Masken und die dazugehörigen Sättigungsbilder.

¹³Quelle: GIMP 2.6.5 User-Manual.

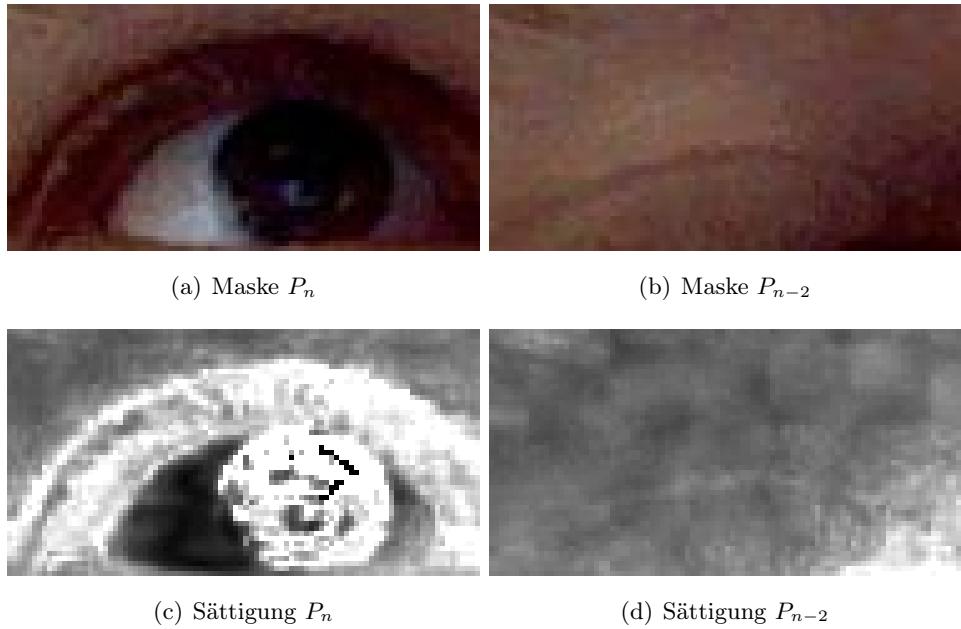


Abbildung 3-24: Maske mit dazugehöriger Sättigung

Wie die Histogramme in Abbildung 3-25 zeigen, liegt der Wertebereich der Sättigung des Augapfels in den unteren 10 Prozent des Histogramms. Das Sättigungsbild

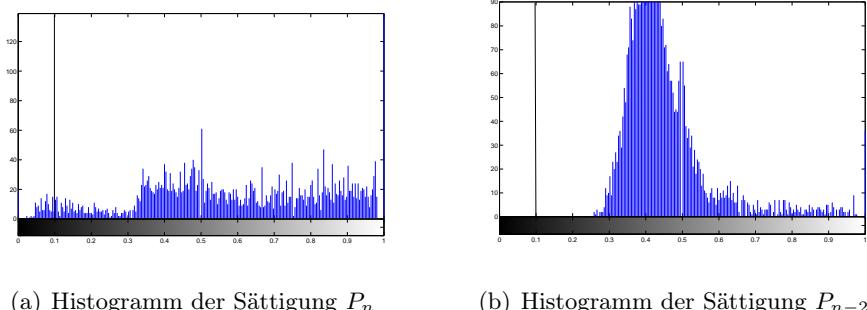


Abbildung 3-25: Sättigungshistogramme

wird nun derart mit einem Schwellwertoperator bearbeitet, dass alle Bildpunkte die unterhalb der Schwelle von $s = 10\%$ liegen, den Maximalwert erhalten, während alle anderen Bildpunkte den Wert 0 bekommen. Nach der Anwendung der Schwellwertbildung liegen nun binäre Bilder vor (Abbildung 3-26).

In diesen binarisierten Sättigungsbildern wird nun die Anzahl der Bildpunkte gezählt, die nicht 0 gesetzt wurden (positive Punkte); aus ihnen können nachfolgende Informationen abgeleitet:

1. Es wird bestimmt, welche Maske das geöffnete Auge zeigt; hier ist auschlag-

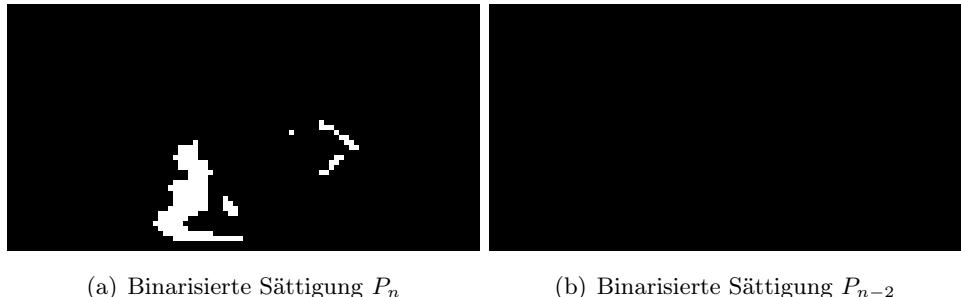


Abbildung 3-26: Binarisierte Sättigungen

gebend, welches Maskenbild die größte Anzahl an positiven Punkten besitzt.

2. Es wird geprüft, ob in mindestens einer der Masken mindestens 0,1% der Maskenfläche mit positiven Punkten besetzt ist. Ist dies nicht der Fall, so wird die Region als unzulässig gewertet und der Algorithmus fährt mit der Lokalisation der Augen fort.
3. Die Maske mit der größeren Anzahl an positiven Punkten muss mindestens 65% mehr solcher Punkte beinhalten, damit die Region als gültig gewertet wird.

Wenn das Verfahren die Masken als gültig qualifiziert hat, ist nun bekannt, welche Masken welchen Augenzustand zeigen, so dass mit der Augenverfolgung und der Auswertung des Lidzustands fortgesetzt werden kann.

3.3.2. Verfolgung der Augenposition

Die Lokalisation der Augenpositionen ist in Bezug auf die Rechenleistung ein sehr aufwändiger Prozess, da hierbei auch das ganze Bild in seiner vollen Auflösung betrachtet wird. Wenn die Position der Augen initial ermittelt wurde, ist es daher wünschenswert, die Position in jedem neu angelieferten Bild nach einem Verfahren zu ermitteln, das weniger rechenintensiv arbeitet.

Die Aufgabe der Augenverfolgung ist es im Grunde, ein Teilbild, nämlich die bei der Lokalisation ermittelte Maske, in einem größeren Bild wiederzufinden. Ein Verfahren aus der Signalverarbeitung, das für diese Aufgabe geeignet ist, ist die Kreuzkorrelation¹⁴. Im zweidimensionalen Anwendungsfall wird die Kreuzkorrelation auch mit *Template Matching* bezeichnet. Das Prinzip ist in Abbildung 3-27 dargestellt: Die Maske wird über das Bild geschoben und für jeden Punkt ein Korrelationskoeffizient

¹⁴[Lew95].

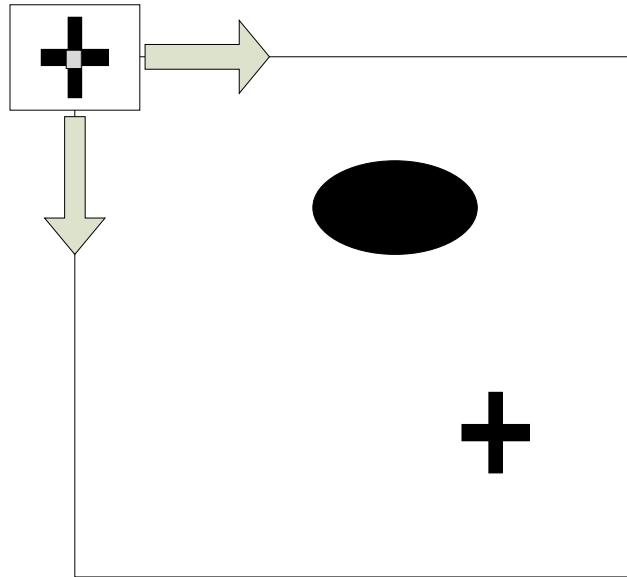


Abbildung 3-27: Template Matching

nach

$$R_{corr}(x, y) = \sum_{x', y'} [T(x', y') \cdot I(x + x', y + y')]^2 \quad (3.25)$$

berechnet, wobei T das Maskenbild ist, während I das abzusuchende Bild darstellt. Die Gleichung 3.25 kann nach derart erweitert werden, dass die Koeffizienten mit den Durchschnitten der Punktswerte normiert werden:

$$R_{normcorr}(x, y) = \frac{\sum_{x', y'} [T(x', y') \cdot I(x + x', y + y')]^2}{\sqrt{\sum_{x', y'} T(x', y')^2 \cdot I(x + x', y + y')^2}} \quad (3.26)$$

Dadurch wird die Methode unempfindlicher gegen Varianzen der Beleuchtungsintensität. Das Ergebnis ist eine neue Bildmatrix, die Punktswerte im Intervall $[0; 1]$ annimmt. Ein Wert von $R = 0$ bedeutet dabei keinerlei Übereinstimmung mit der Maske, während $R = 1$ eine vollkommene Übereinstimmung mit der Maske anzeigen.

Die Eingangsparameter für die Anwendung von Gleichung 3.26 sind die Graustufenmasken der ermittelten Augenregion sowie ein Ausschnitt des Eingangsbildes. Durch die Rechenkomplexität des Template-Matchings ist es nicht sinnvoll, als Eingangsbild das gesamte Bild mit einer Auflösung von $w = 640$ und $h = 480$ Punkten heranzuziehen, da dies bei einer durchschnittlichen Maskengröße von $w_m = 40$ und $h_m = 30$ Punkten zu einem Rechenaufwand führen würde, der nicht mehr innerhalb der Echtzeitanforderungen realisierbar ist. Aus diesem Grund wird nur ein gewählter Ausschnitt aus dem Eingangsbild verglichen. Die Größe des Ausschnitts wurde auf das 2,2fache der Maskengröße festgelegt; sein Mittelpunkt wird initial deckungsgleich mit der Augenregion positioniert. Nach der Ermittlung der neuen Position wird der Ausschnitt entsprechend mitgeführt. Durch die Eingrenzung des

Ausschnitts wird allerdings prinzipbedingt die Geschwindigkeit der Kopf- oder Kamerabewegung eingeschränkt, bei der das System noch die Position verfolgen kann.

Hier wurde zunächst die Strategie verfolgt, nur die Maske des offenen Auges heranzuziehen. Dies hatte jedoch zu Folge, dass bei aktuell geschlossenem Auge ein Maximum der Korrelation nicht mehr am Auge berechnet wurde, sondern an anderen Kantenübergängen wie der Augenbraue oder der Nasenwand. Die Verfolgungsleistung konnte dadurch erhöht werden, dass die Korrelation sowohl mit der Maske des offenen als auch mit derjenigen des geschlossenen Auges durchgeführt wird. Es wird dann die Position der Korrelation verwendet, deren Absolutwert den höchsten Wert aufweist.

Wenn bei keiner Maske ein Korrelationswert über dem Wert von $R = 0,55$ gefunden wurde, so wird die weitere Verfolgung beendet und der Algorithmus beginnt wieder bei der Lokalisation der Augen. Wurde dagegen bei mindestens einer Maske der Korrelationswert $R > 0,55$ erreicht, so wird die Position p_s des Suchfensters auf die ermittelte Position der Maske p_m um den Betrag $(\Delta x, \Delta y)$ verschoben, deren Korrelationswert am höchsten ist. Abbildung 3-28 skizziert den Verschiebeprozess.

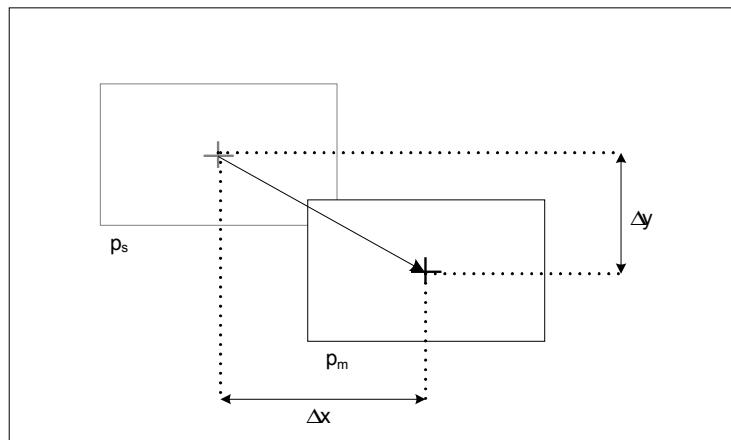


Abbildung 3-28: Nachführen des Suchfensters

Damit ist gewährleistet, dass die Augenposition kontinuierlich nachgeführt wird und der Öffnungszustand bestimmt werden kann.

3.3.3. Bestimmung des Öffnungszustands

Zur Auswertung des aktuellen Lidzustands wurde zunächst der einfache Ansatz verfolgt, als Kriterium die Korrelation mit der Maske des geöffneten Auges zu nutzen. Hier wurde die Entscheidung für den Lidzustand S_{eye} anhand einer Fallunterschei-

dung nach Gleichung 3.27 vorgenommen:

$$S_{eye} = \begin{cases} 1 & \text{wenn } R_{max} > 0,8 \\ 0 & \text{wenn } R_{max} > 0,55 \vee R_{max} \leq 0,8 \\ \text{ungültig} & \text{wenn } R_{max} \leq 0,55 \end{cases} \quad (3.27)$$

Dieser Ansatz konnte bereits erfolgreich den Zustand des Auges detektieren, wenn die Pupille bei geöffnetem Auge in der gleichen Stellung wie bei der Aufnahme verharrt. Abbildung 3-29 zeigt hierzu graphisch den realen Öffnungszustand sowie den Korrelationswert mit der offenen Maske über eine Menge von Bildern aus einem Testvideo.

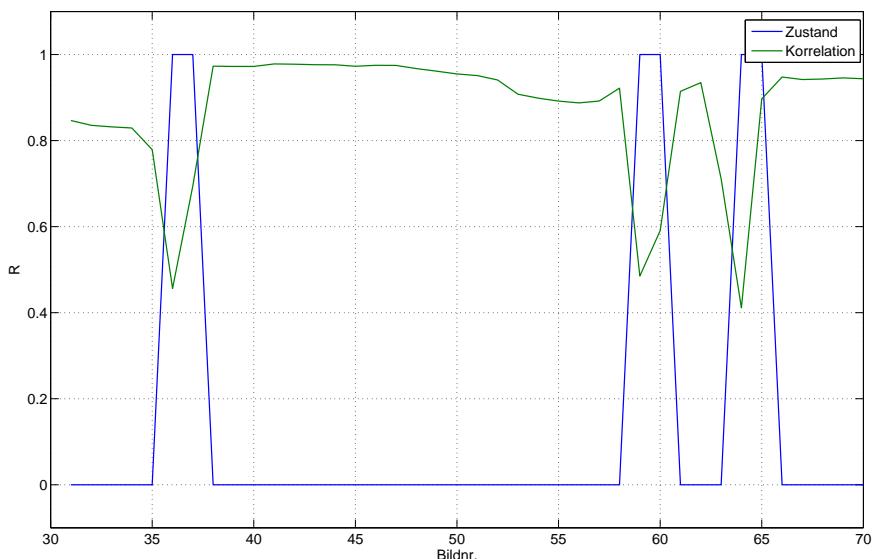


Abbildung 3-29: Korrelationswert in den Lidzuständen

Bei den Testläufen mit Daniela hat sich jedoch gezeigt, dass sie nicht in der Lage ist, die Pupille koordiniert ruhig zu halten; dies zeigte sich besonders deutlich bei besonders starken Emotionen, wie etwa der Freude über einen geglückten Schaltversuch oder auch bei Ablenkung durch äußere Gegebenheiten (z.B. Betreten des Raumes durch Personen). Die Stellung der Pupille verändert das zu vergleichende Bild derart, dass ein offenes Auge in Bezug auf den Korrelationswert nicht mehr von einem geschlossenen Auge unterscheidbar ist. Abbildung 3-30 stellt den Problemfall mit den resultierenden Korrelationsfaktoren dar.

Es ist zu sehen, dass im rechten Bild das Lid geöffnet ist, der Korrelationswert jedoch auf $R = 0,782$ sinkt, was zur Folge hat, dass das Lid als geschlossen erkannt wird. Dieses Problem konnte jedoch umgangen werden, indem der Entscheidungsprozess durch weitere Kriterien erweitert wurde. Diese zusätzlichen Kriterien bestehen aus

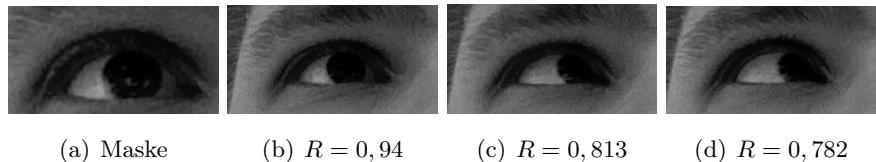


Abbildung 3-30: Fehlerhafte Zustandserkennung

1. einer Verkopplung mit der offenen als auch mit der geschlossenen Maske sowie
2. der Berücksichtigung der Sättigungsinformation in Bezug auf die Sättigung der zugrundeliegenden Masken.

Die Sättigungsinformation aus der Maskenwahl konnte für diesen Ansatz weitergenutzt werden und dient nun auch zur Bestimmung des aktuellen Lidzustands.

Die Auswertung des Lidzustands erfolgt schließlich über den in Abbildung 3-31 dargestellten Entscheidungsbaum:

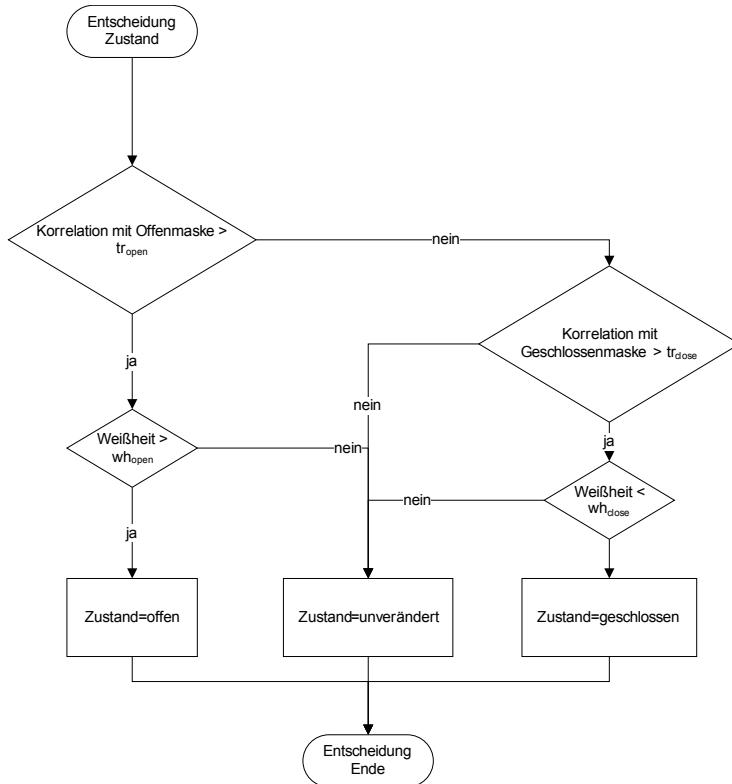


Abbildung 3-31: Entscheidungsbaum zum Augenzustand

Die Schwellwerte für die Korrelation tr_{open} und tr_{close} entsprechen den zuvor ermittelten Grenzwerten von S_{eye} aus Gl. 3.27. Der Schwellwert wh_{open} wurde so

festgelegt, dass mindestens 65% der Anzahl von weißen Punkten bezogen auf die Anzahl der weißen Punkte in der Offenmaske sichtbar sind. Die Schwelle wh_{close} wurde so gewählt, das nur maximal 10% der Anzahl von weißen Punkten sichtbar sein darf. Dies führt zu stabilen Entscheidungen.

Abbildung 3-32 skizziert zusammenfassend die Verarbeitungsschritte für die Verfolgung der Augen.

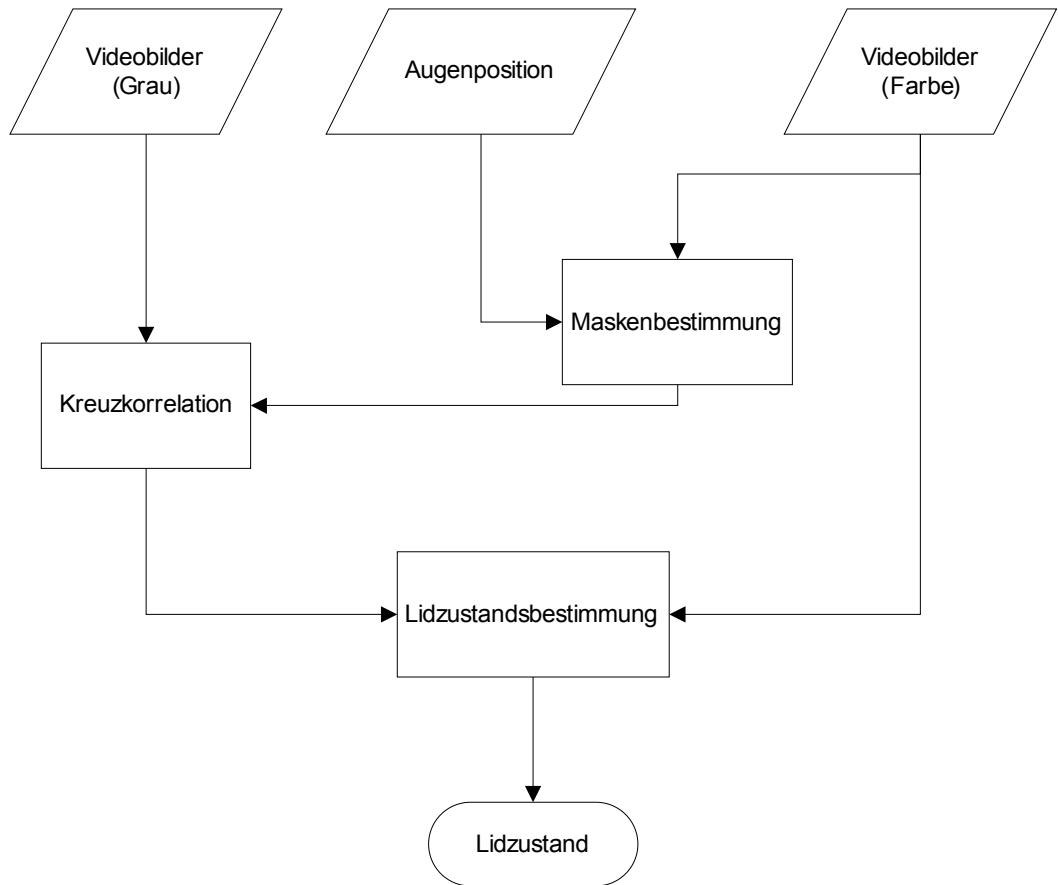


Abbildung 3-32: Ablauf der Augenverfolgung

Der Algorithmus mit der vorgestellten Parametrisierung ermöglicht es, die Augen zu lokalisieren und ihren Lidzustand zu bestimmen. Der Lidzustand liegt hierbei in binärer Form vor und wird von den Ausgabemodulen weiterverarbeitet.

4. Entwicklung der Software *sensbli*

Die Untersuchung des oben beschriebenen Algorithmus und dessen echtzeitfähige Implementierung sind mit teilweise gegensätzlichen Anforderungen verbunden. Die Untersuchung eines Algorithmus erfordert eine schnelle prototypische Umsetzung von neuen Erkenntnissen sowie den nachvollziehbaren Zugriff auf Zwischenergebnisse in jedem Verarbeitungsschritt. Die echtzeitfähige Implementierung sollte jedoch hinsichtlich der Verarbeitungszeit des Algorithmus optimal sein, was etwa die Ausgabe von Zwischenergebnissen verbietet; es ist ebenfalls nicht möglich, in einem Echtzeitsystem wahlfrei auf die Eingangsdaten zuzugreifen. Die Implementierung der Software wurde aus diesen Gründen zweistufig unter dem Arbeitstitel *sensbli*¹⁵ durchgeführt.

Zunächst erfolgte eine prototypische Implementierung und Untersuchung der algorithmischen Zusammenhänge in der technisch-wissenschaftlichen Programmiersprache MATLAB. Hierbei lag der Fokus auf dem Zugriff und der Visualisierung der einzelnen Teilschritte und deren anfallenden Ergebnisdaten. Nach der vollständigen Parametrisierung des Algorithmus wurde das Verfahren dann in eine echtzeitfähige Implementierung überführt. Damit die Überführung des Prototyps in die Echtzeitsoftware ohne einen erhöhten Strukturierungsaufwand gelingen konnte, war es notwendig, dass die Architektur beider Implementierungen identisch ist.

4.1. Softwarerarchitektur

Jede Software, die über triviale Aufgaben hinaus Daten verarbeitet, stellt ein komplexes Gebilde aus einzelnen Teilelementen dar. Der Entwurf einer solchen komplexen Software setzt voraus, dass die Aufgabenstellung akkurat erfasst und die Software in kleinere Teile zerlegt wird. Die Architektur beschreibt die Zerlegung und das Zusammenspiel der entstandenen Teilelemente eines Softwaresystems anhand standardisierter Sprachmittel aus verschiedenen Sichtweisen.¹⁶ Die Teilelemente einer Software sind die Komponenten; sie werden wie folgt definiert:

Komponenten sind modulare Teile eines Systems, die ihren Inhalt und somit komplexes Verhalten transparent kapseln und in ihrer Umgebung als austauschbare Einheiten mit klar definierten Schnittstellen auftauchen.¹⁷

¹⁵ Abkürzung für **SenseBlink** („erkenne Blinzeln“)

¹⁶ Dies geschieht ähnlich der Architektur im Bauwesen, die sich jeweils unterschiedlicher Pläne für Statik, Konstruktion und Elektro bedient.

¹⁷ Vgl. [RH06], S. 48.

Die Komponentenzerlegung erfolgt somit unter nachfolgend dargestellten Gesichtspunkten:¹⁸

Modularisierung: Die Komplexität einer Aufgabenstellung wird durch die Zerlegung in kleinere, voneinander unabhängige Teilmodule reduziert. Die Komponenten sind dann lose miteinander gekoppelt.

Kapselung: Die Komponenten verstecken Methoden und Daten, die für andere Komponenten nicht von Belang sind. Die Verknüpfung von Komponenten erfolgt nur über wohldefinierte Schnittstellen.

Trennung von Zuständigkeiten: Jeder Komponente wird nur eine Aufgabe zugewiesen. Dies ermöglicht einen problemlosen Austausch von Komponenten; Teile der Aufgabenstellung können verändert werden, ohne dass andere Komponenten angepasst werden müssen.

In Kapitel Methodenbeschreibung wurden die sequentiell voneinander abhängigen Ablaufschritte identifiziert und unterteilt. Auch der Datenfluss für den Ablauf wurde in Abbildung 3-4 spezifiziert. Im folgenden wird nun die Umsetzung dieser Zerlegung in eine Softwarekomponentenarchitektur beschrieben.

Der grundlegende Aufbau einer Signalverarbeitungskette nach Abbildung 4-33 besteht aus der Eingabe, der Verarbeitung und der Ausgabe. Im vorliegenden Fall ist die Eingabe der Videodatenstrom, die Verarbeitung der Erkennungsalgorithmus und die Ausgabe die Bereitstellung der Lidzustandsinformation. Dieser grundlegen-

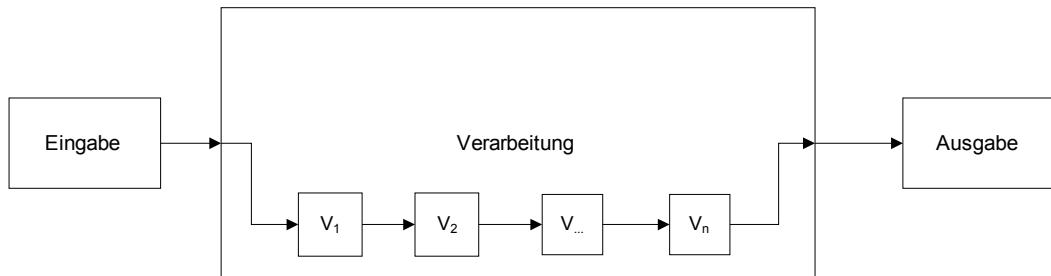


Abbildung 4-33: Struktur eines signalverarbeitenden Systems

de Ablauf wird in einem signalverarbeitenden System zyklisch abgearbeitet, solange Eingangsdaten anliegen. Die Verarbeitung kann dabei beliebig komplex werden und mitunter aus mehreren Verarbeitungsschritten V_1 bis V_n bestehen, die wie im vorliegenden Fall voneinander abhängen. Die Verarbeitungsprozesse selbst sind eine Aneinanderreihung von Operationen auf Eingangsdaten, die Ergebnisse bereitzustellen.

¹⁸Siehe [RH06], S. 72ff.

Die Darstellung der Architektur erfolgt aus mehreren Sichtweisen, um einzelne Eigenschaften der Software gezielt hervorzuheben. In dieser Arbeit wird die Sicht auf die beiden wesentlichen Eigenschaften der Software beschränkt, nämlich auf das Zusammenspiel ihrer Komponenten und auf die physikalische Sichtweise.

4.1.1. Komponentensichtweise

Die Software besteht aus den beiden Grundkomponenten „Prozesse“ und „Prozess-Manager“. Die Prozesse abstrahieren den Ablauf einer Prozesskette, während der Prozess-Manager das Zusammenspiel mehrerer Prozessketten koordiniert.¹⁹ Abbildung 4-34 stellt einen Prozess mit seinen Schnittstellen dar. Jeder Prozess besitzt

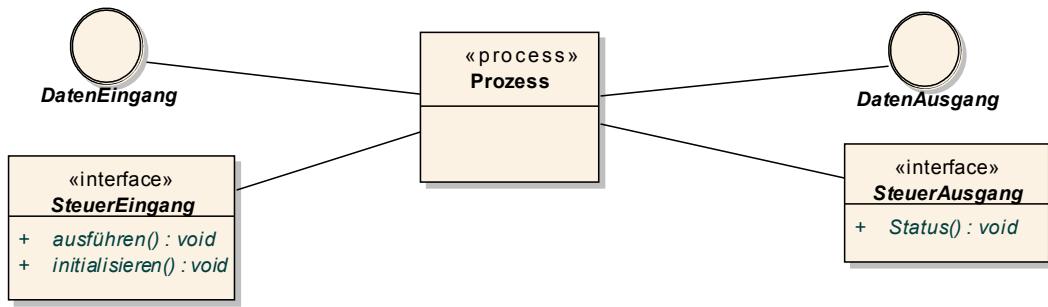


Abbildung 4-34: Prozesskomponente

Ein- und Ausgabeschnittstellen, die sich wiederum in Daten- und Steuerschnittstellen unterteilen lassen. Über die Datenschnittstelle *DatenEingang* werden die zu verarbeitenden Daten bereitgestellt und das Ergebnis über die Schnittstelle *DatenAusgang* nach außen geführt. Die Steuerschnittstellen *SteuerEingang* und *SteuerAusgang* ermöglichen die Kontrolle des Prozesses. Die Ausführung eines Prozesses wird über die Methode *ausführen()* gestartet. Da ein Prozess zustandsbehaftet sein kann, besteht die Möglichkeit über *initialisieren()* den inneren Zustand zurückzusetzen. Über die Methode *Status()* kann schließlich der Ausführungsstatus des Prozesses abgefragt werden.

Abbildung 4-35 zeigt die komplette Prozesskette für den Erkennungsalgorithmus.

In der Komponentenansicht werden die Steuerschnittstellen der Prozesse mit *Control* und die Datenschnittstellen mit *Data* bezeichnet. Die Komponente *VideoInput* abstrahiert das Einlesen der Videodaten; diese Videodaten werden dem Eingangspuffer *ImageBuffer* übergeben. Der Eingangspuffer wird selbst als eigenständiger Prozess ausgeführt, da der Puffer zustandsbehaftet ist. Dies liegt darin begründet,

¹⁹Vgl. dazu Execution Control Pattern, [Dou], S. 8f.

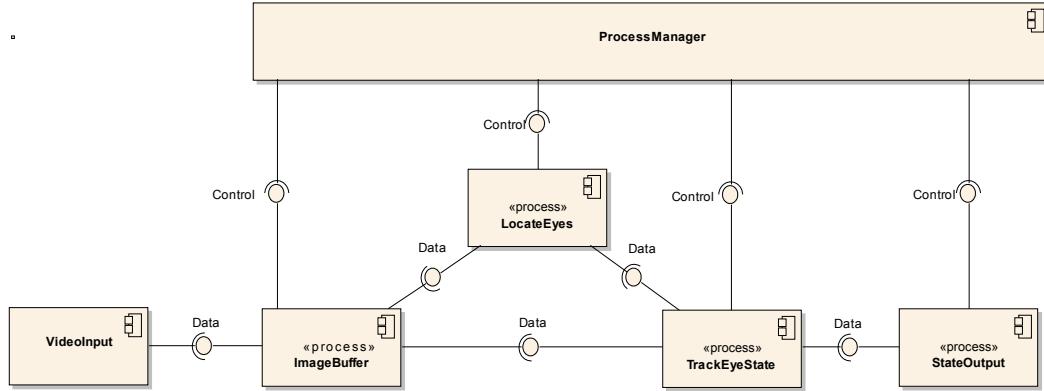


Abbildung 4-35: Komponentenansicht der Prozesskette

dass der Puffer eine Mindestanzahl von 3 Bildern aus dem Datenstrom vorhalten muss, um die Differenzbildung für die Lokalisation zu ermöglichen. Die Komponenten für die Lokalisation der Augen *LocateEyes* und der Verfolgung *TrackEyeState* erhalten beide Daten aus dem Bildpuffer; die Verfolgung greift zudem auch noch auf Prozessdaten aus der Lokalisation zurück. Der Augenzustand wird schließlich über die Komponente *StateOutput* an übergeordnete Systemkomponenten bereitgestellt.

Die Steuerung der Prozesse wird von der Komponente *ProcessManager* koordiniert. Sie verwaltet die Prozesse über ihre Steuerschnittstellen und kontrolliert dadurch deren Ablauffreihenfolge. Die Steuerung wird als Zustandsautomat ausgeführt, dessen Zustandsdiagramm in Abbildung 4-36 skizziert ist.

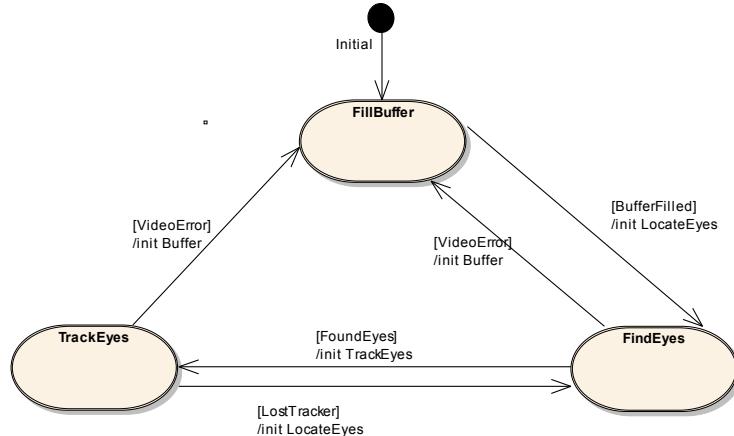


Abbildung 4-36: Zustandsautomat Prozessmanager

Beim Start des Programmlaufs beginnt die Steuerung im Zustand *FillBuffer*. Hier verbleibt sie so lange, bis der Puffer mit der Mindestanzahl von zu bearbeitenden Bildern gefüllt ist. Danach wechselt der Zustand zu *LocateEyes* und initialisiert den

LocateEyes-Prozess. Sobald Augenpositionen gefunden wurden, wird in den Zustand *TrackEyes* gewechselt und dessen zugehöriger Prozess *TrackEyeState* initialisiert. Verliert der Prozess *TrackEyeState* die Augenposition, so kehrt das Verfahren wieder in den Zustand *LocateEyes* zurück. Sollte ein fehlerhafter Programmablauf im Rahmen der Videobildverarbeitung²⁰ auftreten, so wird der Eingangspuffer neu initialisiert und in den Zustand *FillBuffer* zurückgewechselt.

Der zeitliche Ablauf der Zustandswechsel wird aus dem Sequenzdiagramm 4-37 ersichtlich.

Dieses Sequenzdiagramm²¹ verdeutlicht den Ablauf in den einzelnen Systemzuständen und zeigt, welche Kommunikationen zwischen den Komponenten erfolgen.

4.1.2. Physikalische Sichtweise

Die physikalische Sichtweise (Abbildung 4-38) zeigt das Zusammenspiel der Komponenten auf unterschiedlichen Abstraktionsschichten.

Auf der niedrigsten Ebene befinden sich diejenigen Geräte, die für die Systemfunktion relevant sind. Sie werden durch Treiber auf der Betriebssystemebene abstrahiert. Auf die so abstrahierten Geräte kann mittels Programmbibliotheken zugegriffen werden, die sich in der *SoftwareService*-Ebene befinden. Auf dieser Ebene finden sich auch die Servicebibliotheken zur Nutzung von mathematischen Funktionen für die Bildverarbeitung, so dass für die Arbeit keine grundlegenden Verarbeitungsfunktionen wie Schwellwertbildung oder morphologische Operatoren manuell implementiert werden müssen. Über der Serviceschicht befindet sich der Algorithmus. Er kommuniziert den Augenzustand über eine Socketbibliothek als UDP-Nachricht an ein Netzwerk. Hierdurch wird eine architektonische Unabhängigkeit der Ausgabe ermöglicht, da die Ausgabe über die Relaiskarte nicht mehr zwingend im gleichen Rechnersystem implementiert sein muss. Die eingerahmten Komponenten sind durch die Kommunikation über Netzwerkverbindungen wiederum auch auf andere Rechnersysteme verlagerbar.

Bei dieser Betrachtungsweise tritt auch der wesentliche Unterschied zwischen der Prototypenimplementierung und der Echtzeitsoftware zutage; der Algorithmus bleibt gleich, nur die Service- und die Betriebssystemschicht verändern sich. Anstelle der Serviceschicht wird die Umgebung von MATLAB verwendet. Die Eingangsdaten werden im Prototypen als Videodateien zur Verfügung gestellt und die Ausgabe des Augenzustands über textuelle Meldungen simuliert.

²⁰Etwas eine Division durch Null oder ungültige Videobilddaten.

²¹Vgl. UML-Sequenzdiagramme, [RHQ⁺05], S.407ff.

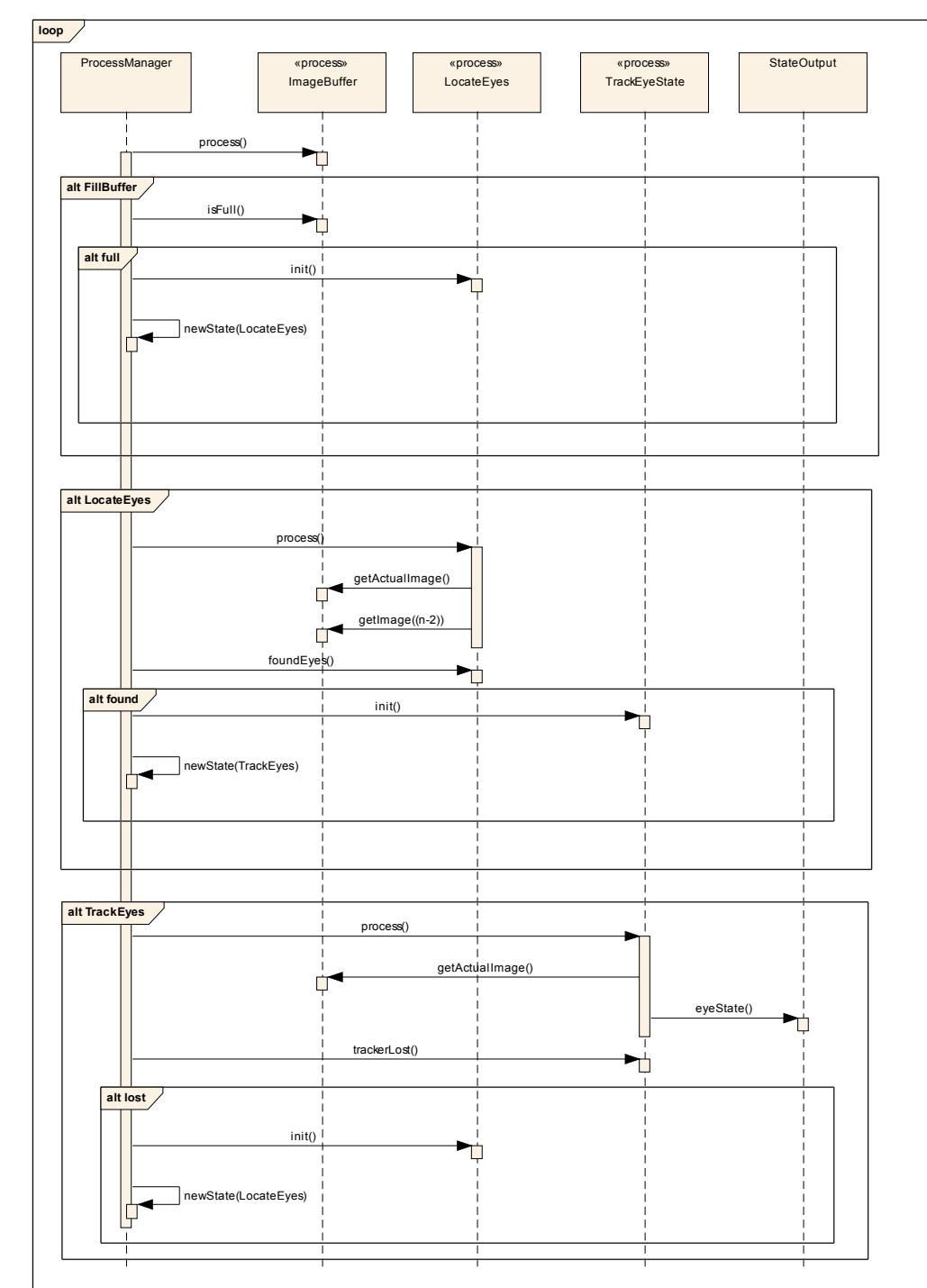


Abbildung 4-37: Sequenzdiagramm Zustandswechsel

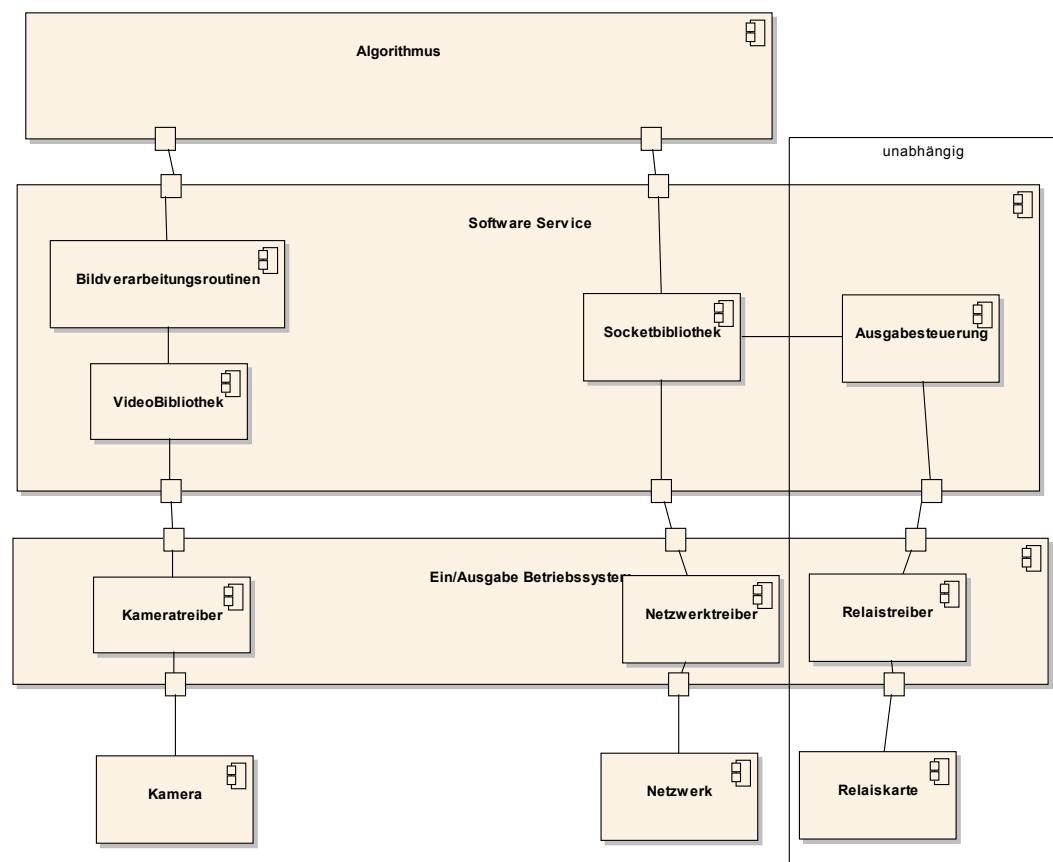


Abbildung 4-38: Physikalische Schichten

4.2. Prototyp

Die Kernaufgabe des Prototypen war die Untersuchung des Erkennungsalgorithmus. Zur Untersuchung war es erforderlich, dass die Verarbeitungsschritte des Ablaufs sowohl graphisch als auch numerisch nachvollziehbar sind. Der Prototyp wurde in der MATLAB-Umgebung²² entworfen, was den Vorteil hatte, dass die Umgebung bereits Hilfswerzeuge für die Ergebnisdarstellung sowie eine Sammlung von grundlegenden Signalverarbeitungsalgorithmen zur Verfügung stellte. Eine Verarbeitung in Echtzeit ist mit MATLAB zwar nicht möglich, dies ist jedoch auch nicht notwendig, da zur Untersuchung der Algorithmen die Nachvollziehbarkeit der Eingangsdaten gewährleistet sein muss. Die Eingangsdaten liegen für den Prototypen als unkomprimierte²³ AVI-Videosequenzen vor.

Ein weiterer Unterschied zwischen dem Prototypen und der späteren Echtzeitsoftware liegt darin, dass nicht nur das Auge mit der größten Region ausgewertet wird, sondern beide Augenregionen nacheinander ausgewertet werden. Diese Vorgehensweise ermöglichte es, eine größere Anzahl an Testdaten für die Algorithmen zur Lidzustandserkennung zu gewinnen.

Die vollständigen Quellen für den Prototypen befinden sich auf der CD-ROM unter */prototype/source*.

4.2.1. Modulkonzept

Die Implementierung des objektorientierten Komponentenkonzepthes konnte aufgrund der mangelnden Unterstützung der Objektorientierung durch MATLAB nicht unmittelbar umgesetzt werden. Aus diesem Grund wurde ein Modulkonzept entwickelt, dass eine objektorientierte Umsetzung auch im Prototyp ermöglichte: MATLAB kennt nur Funktionen in Dateien, d.h. jede Funktion ist eine eigenständige Datei. Zur Umsetzung der Komponenten wurden die Dateien nach dem Schema *komp_methode.m* angelegt.

komp bezeichnet den Namen der jeweiligen Komponente, während *methode* den Namen der zugehörigen Methode beschreibt. Die Zustände der Komponenten werden durch Strukturvariablen festgelegt, die den gleichen Namen wie die Komponente tragen. Diese Strukturvariablen bestehen aus den einzelnen Zustandsvariablen (bspw. *komp.var1*, *komp.var2*). Die Strukturvariablen werden innerhalb der Komponentendateien als globale Variablen mit dem Schlüsselwort *global* angelegt, so dass jede Datei innerhalb der Komponente auf die Daten zugreifen kann. Durch die Deklaration als globale Variablen ist zwar keine Kapselung der Daten gewährleistet, aber der Inhalt der Daten kann von außen leicht abgefragt werden.

²²Die verwendete Version war MATLAB R2008a.

²³UNIX-Versionen von MATLAB können nur unkomprimierte Videodateien verarbeiten

4.2.2. Implementierung

Den Einstiegspunkt für den Prototyp bildet das Modul `sensbli.m`, was die Darstellung der grafischen Oberfläche behandelt (Abbildung 4-39). Die GUI²⁴ besteht im

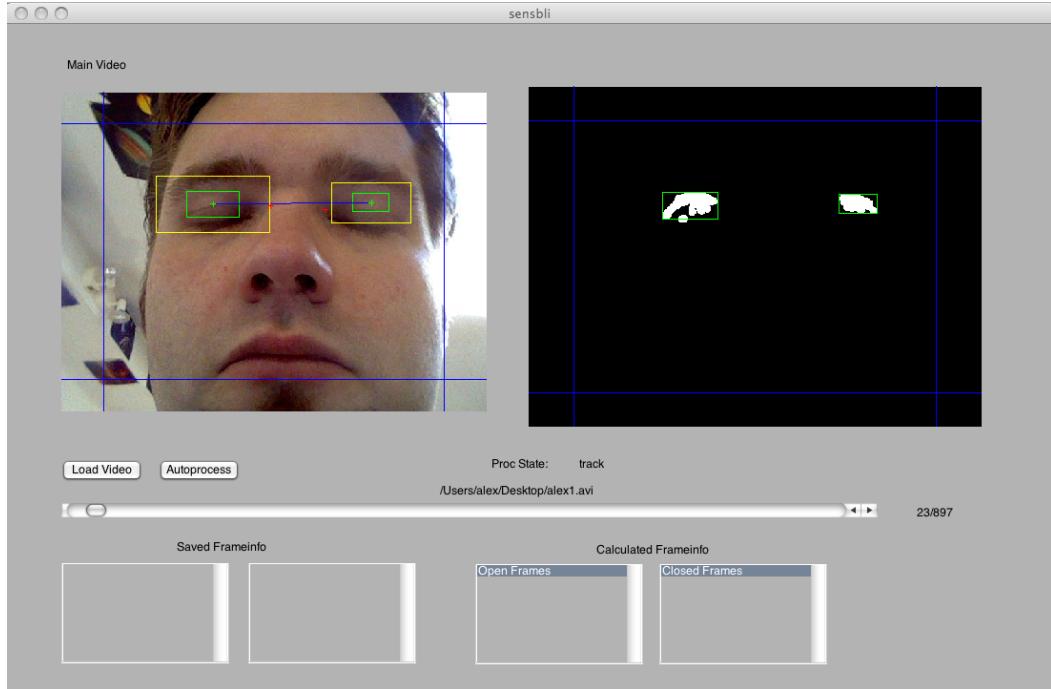


Abbildung 4-39: Prototypen-GUI

Wesentlichen aus zwei Grafiken, die zum einen das aktuelle Videobild mit Markierungen der erkannten Regionen und die Suchfenster und zum anderen die Zwischenstufen für die Erkennung der Augenregionen darstellen. Weitere Bedienelemente sind eine Zeitleiste, die einen wahlfreien Zugriff auf die Videodaten ermöglicht, sowie zwei Schaltknöpfe. Dabei öffnet der Schaltknopf *LoadVideo* einen Dialog zum Laden der Eingangsdaten; der Schaltknopf *AutoProcess* startet die Verarbeitung der Videosequenz von ersten bis zum letzten Bild.

Die Bilder können über die Zeitleiste oder die *AutoProcess*-Funktion in der Videosequenz weitergeschaltet werden. Bei jedem neuen Bild wird der Bildpuffer gefüllt und der Algorithmus gemäß dem Zustandsautomat nach Abbildung 4-36 ausgeführt. Der Zustandsautomat befindet sich in dem Modul *bldetect*. Da die Verarbeitung nur dann sinnvolle Ergebnisse liefern kann, wenn die Daten aufeinanderfolgende Bildelemente erhalten, wird das Zurückspulen oder das Springen der Bildfolgen um einen anderen Wert als $n + 1$ wie ein Fehler (*VideoError* als Bedingung im Zustandsautomat) gewertet, woraufhin der Zustandsautomat in die Pufferinitialisierung zurückspringt. Der aktuelle Zustand wird im Textfeld *ProcState* ausgegeben.

²⁴Graphical User Interface.

Die Videodarstellungen enthalten zusätzliche grafische Elemente, die den Algorithmus visualisieren.

Sowohl in der Videoansicht als auch in der Regionenansicht werden diejenigen Regionen, die als gültige Augenpositionen ermittelt wurden, mit einem grünen Rechteck umrandet. Erkannte Blobs, die nicht den Kriterien für gültige Regionen entsprechen, werden dagegen mit einem roten Rechteck umrandet.

Wurden Regionen erkannt, dann wechselt das System in den Tracking-Zustand; es werden in der Videoansicht weitere Markierungen sichtbar: Die Suchfenster für die Masken werden gelb umrandet. Die aktuell gefundenen Positionen der Masken innerhalb der Suchregionen werden für die Masken der geöffneten Augen mit grünen Kreuzen und für die Masken der geschlossenen Augen mit roten Kreuzen versehen.

Im Tracking-Zustand wird zudem ein weiteres Fenster mit visualisierten Zwischenergebnissen für jede Augenregion geöffnet; Abbildung 4-40 zeigt die Graphen für ein Auge.

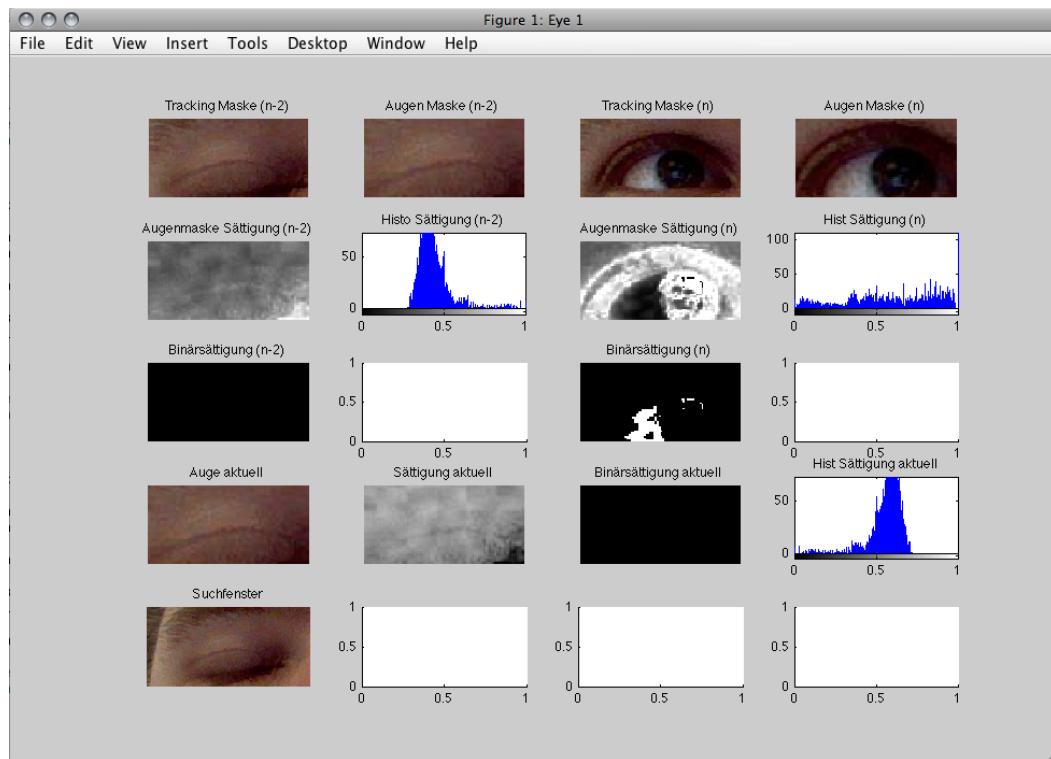


Abbildung 4-40: Prototypen Visualisierung des Trackings

Der Lidzustand wird auf der MATLAB-Konsole nach dem Durchlaufen des Tracking-Prozesses in der Variablen *eyestate* ausgegeben. Weitere Zwischenergebnisse können ebenfalls über die Konsole abgefragt werden, indem auf die globalen Zustandsvariablen zugegriffen wird.

Tabelle 4-4 führt eine Beschreibung der Teilmodule des Prototypencodes auf.

Unter der Prototypenumgebung wurden die notwendigen Algorithmen und Parametrisierungen, wie sie in Kapitel 3 beschrieben wurden, festgelegt.

4.3. Echtzeitsoftware

Zur Implementierung der Echtzeitvariante war es notwendig, entsprechende Umgebungsvoraussetzungen zu schaffen. Neben der Wahl einer geeigneten Programmiersprache spielt hier die Auswahl der Bildverarbeitungsbibliotheken eine entscheidende Rolle.

4.3.1. Programmierumgebung und Bibliotheken

Entsprechend den Anforderungen an das System wurde die Betriebssystemumgebung auf Linux festgelegt. Daher wurde als Grundstruktur *video4linux* als standardisierte Schnittstelle für den Kamerazugriff ausgewählt. Die Entscheidung für die Programmiersprache fiel auf C++, da sie verschiedene Vorteile in sich vereint. Sie ist

- nahe an der Hardware und damit geschwindigkeitsoptimiert;
- eine direkte Umsetzung der Komponentenarchitektur ist durch ihren objekt-orientierten Ansatz möglich und
- es besteht eine gute Unterstützung von Seiten des Linux-Betriebssystems.

Die Schnittstellen zur Netzwerkkommunikation sind bereits in den Standardbibliotheken²⁵ des Betriebssystems enthalten. Als Bildverarbeitungsbibliothek wurde OpenCV ausgewählt. Bei OpenCV handelt es sich um ein freies Projekt im Sinne der GNU LGPL, das von dem Unternehmen Intel vorangetrieben wird. Die Vorteile von OpenCV sind

- ihre Quelloffenheit;
- sie enthält alle Verarbeitungsfunktionen, die auch MATLAB bietet;
- die Bibliothek kann aus der Sprache C++ verwendet werden;
- sie ist auf den Plattformen Linux, MacOS und Windows verfügbar und
- es existieren umfangreiche Dokumentationen und Lehrbücher.

Die Bibliothek wurde für die vorliegende Arbeit aus den Subversion-Archiven in der Version 1.0.7 kompiliert. Ein weiterer Aspekt von OpenCV ist zudem, dass sie automatisch IPP-Bibliotheken in der Laufzeitumgebung nutzt, sofern diese vorhanden

²⁵GNU libc.

Komponente	Modul	
ProzessManager	bldetect	
	Modulfunktion	Beschreibung
	bldetect_init.m	Initialisierung des Prozess-Managers
	bldetect_process.m	Ausführen des Zustandsautomaten
	bldetect_getstate.m	Rückgabe des aktuellen Prozesszustands
Bildpuffer	imgbuffer	
	Modulfunktion	Beschreibung
	imgbuffer_init.m	Initialisierung des Bildpuffers
	imgbuffer_add.m	Bild zu dem Puffer hinzufügen
	imgbuffer_getactual.m	Aktuelles Bild aus dem Puffer holen
	imgbuffer_getlast.m	Bild $n - 2$ aus dem Puffer holen
	imgbuffer_isfull.m	Rückgabe eines Signals, ob der Puffer voll ist
LocateEyes	findeyes	
	Modulfunktion	Beschreibung
	findeyes_init.m	Initialisierung der Augenlokalisati-on
	findeyes_process.m	Verarbeitungsprozess der Lokalisa-tion
	findeyes_found.m	Rückgabe, ob Augen gefunden wur-den
	findeyes_pos.m	Rückgabe der Augenregionen
TrackEyeState	trackeyes	
	Modulfunktion	Beschreibung
	trackeyes_init.m	Initialisierung der Augenverfolgung
	trackeyes_process.m	Verarbeitungsprozess der Augen-verfolgung
	trackeyes_lost.m	Rückgabe, ob die Verfolgung die Augen verloren hat

Tabelle 4-4: Module des Prototypen

sind. Hierdurch ist eine Beschleunigung der Rechenschritte auf Intel-Plattformen möglich.

4.3.2. Implementierung

Analog zu der Implementierung im Prototyp werden die Komponenten auch in der Echtzeitvariante in Teilmodule aufgespalten. Der Einstiegspunkt für die Applikation bildet das Modul *sensbli.cpp*²⁶, das Kommandozeilenoptionen auswertet und die Kontrolle an den ApplikationsManager *CAppManager.cpp* übergibt. Der Applikationsmanager initialisiert dann die Kameraschnittstelle und die Komponente *ImageBuffer*. Zusätzlich enthält der ApplikationsManager auch Teifunktionen der Komponente *ProcessManager*, die für die Koordination des Bildpuffers verantwortlich sind. Die Bilddaten werden von letzterem zyklisch von der Kamera abgeholt und in den Bildpuffer geladen. Sobald der Bildpuffer gefüllt ist, wird je nach Kommandozeilenoption entweder ein Kalibrierprozess oder die Verarbeitungskette aufgerufen. Dabei ist der Kalibrierprozess eine einfache Darstellung des Videobildes mit einer Maske.

Die Routinen zur Initialisierung der Kamera und für das Auslesen des Videostroms werden von OpenCV plattformunabhängig abstrahiert. Zur Initialisierung wird die Routine `cvCaptureFromCAM(CV_CAP_ANY)` aufgerufen. Durch das Argument `CV_CAP_ANY` wird die erste verfügbare Kamera im System ausgewählt und mit ihrer Standardauflösung von $w = 640$ und $h = 480$ Bildpunkten initialisiert. Die Videobilder werden mit der Routine `cvQueryFrame()` von der Kamera geholt. Die weiteren Softwaremodule und ihre Komponentenzuordnung sind in Tabelle 4-5 aufgeführt.

Komponente	Modul
ProcessManager	C BlinkProcessor.cpp und C AppManager.cpp
Bildpuffer	imagedb/CImageDB.cpp
LocateEyes	eyefinder/CEyeFinder.cpp
TrackEyeState	eyetracker/CEyeTracker.cpp und eyestatehandler/CEyeStateHandler.cpp

Tabelle 4-5: Module der Echtzeitanwendung

Die Komponententeile sind in Klassen aufgeteilt, deren Zusammenhänge in Abbildung 4-41 dargestellt sind.

Die Information, welchen Zustand das Augenlid eingenommen hat, wird mit Hilfe

²⁶Im folgenden werden die C++ Klassendateien als Module beschrieben; die Modulschnittstellen befinden sich gemäß C++ Konvention in zugehörigen .h Headerdateien.

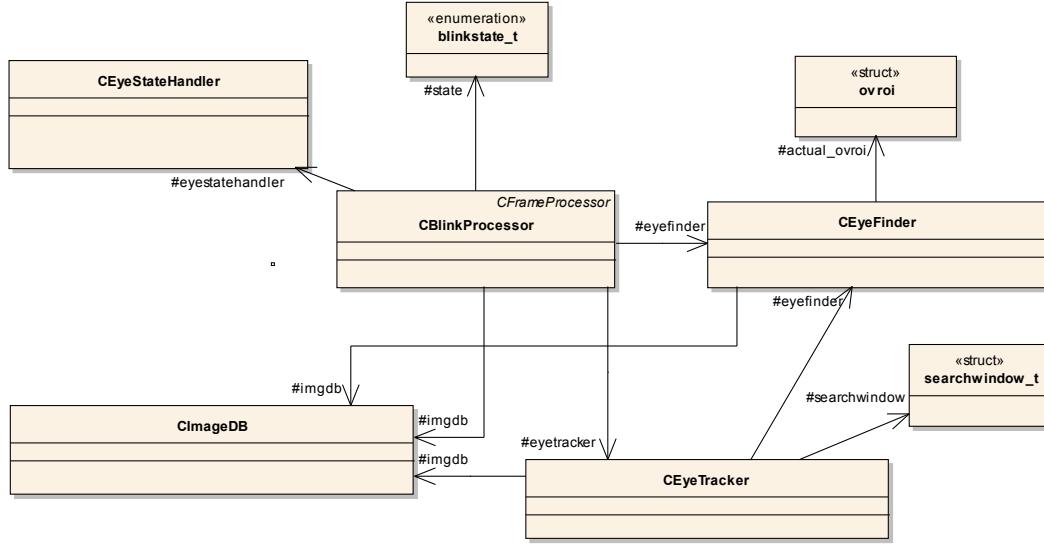


Abbildung 4-41: Klassendiagramm sensbli

von UDP- Nachrichten an das Netzwerk versendet; der Aufbau dieser Nachrichten wird im Rahmen der Betrachtung des Ausgabemoduls in Kapitel 4.4.1 beschrieben.

Die Software kann auf Quellcodeebene konfiguriert werden, wozu die Datei `config.h` dient. Die Konfigurationsparameter sind in Tabelle 4-6 aufgeführt.

Der vollständige Quellcode ist auf der beiliegenden CD-ROM unter `/software/sensbli/src` zu finden.

4.3.3. Programmlauf

Die Software *sensbli* kann in den beiden Betriebsmodi

- Normalbetrieb und
- Kalibrieren

betrieben werden.

Der Kalibriermodus gibt Hilfestellung bei der Ausrichtung der Kamera auf den Benutzer, indem das Videobild mit einer Maske überlagert wird. Die Maske zeigt das stilisierte Bild eines Gesichts. Durch das Kalibrieren wird der sichtbare Ausschnitt des Benutzers dergestalt eingepasst, dass die Augengröße ein Mindestmaß erreicht, um eine Auswertung der Farbsättigung zu ermöglichen. Der Ablauf der Kalibrierung ist in Abbildung 4-42 zu sehen.

Wird das Programm mit dem Kommando

```
./sensbli -calib
```

Parameter	Beschreibung
CFG_CAPTURE_FROM_FILE	Wenn gesetzt, wird der Videostrom nicht von der Kamera eingelesen, sondern aus einer Videodatei (siehe auch CFG_INPUT_FILE).
CFG_INPUT_FILE	Liest Videodaten aus der angegebenen Datei aus, wenn CFG_CAPTURE_FROM_FILE = 1 gesetzt ist.
CFG_DBG_HANDSTEP	Wenn gesetzt, werden die Videobilder nur durch Tastendruck weitergeschaltet. Gilt nur für CFG_CAPTURE_FROM_FILE = 1.
CFG_DBG_MAINSTATE	Wenn gesetzt, wird der Zustand der Prozesskette ausgegeben.
CFG_DBG_LOGFILE	Wenn gesetzt, werden Variablen bei der Regionensuche in eine CSV-Datei geschrieben.
CFG_SERVER_ADDR	Hier wird festgelegt, an welche IP-Adresse die Nachricht einer Lidzustandsänderung geschickt wird.
CFG_SERVER_PORT	Hier wird festgelegt, an welchen IP-Port die Nachricht einer Lidzustandsänderung geschickt wird.

Tabelle 4-6: Konfigurationsparameter



Abbildung 4-42: Bildschirmaufnahme während des Kalibrierens

gestartet, so arbeitet es im Modus „Kalibrieren“. Wenn das Programm dagegen mit dem Kommando

```
./sensbli
```

ohne Kommandozeilenoptionen gestartet wird, wird der Erkennungsalgorithmus ausgeführt und es wird ein Live-Videobild mit grafischen Markierungen angezeigt. Die Software versendet dabei bei erkannten Lidzustandsänderungen entsprechende Nachrichten.

Durch Drücken der ESC-Taste kann der Programmlauf beendet werden.

4.4. Ausgabemodul

Die Verarbeitung der von der Auswertesoftware versendeten UDP-Nachrichten und die darauf folgende Ansteuerung einer externen Relaiskarte wurde in eine eigene Software ausgelagert. Dies hat den Vorteil, dass der Auswerterechner und der Steuereinheit voneinander entkoppelt werden können. Während der Entwicklungsphase konnte zudem die Steuerung der Relaiskarte durch einfache Textausgaben ersetzt werden.

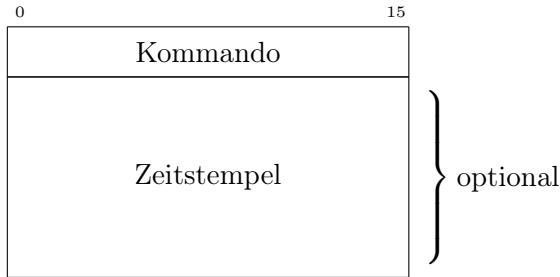
4.4.1. UDP-Nachrichten

Die Kommunikation zwischen Auswerte- und Steuersoftware erfolgt über UDP-Nachrichten gemäß dem Standard für die Internetprotokollfamilie RFC768. UDP ist ein verbindungsloses Protokoll, d.h. es werden einfache Nachrichten (Datagramme) über eine IP-Verbindung versendet. Dies ist für die vorliegende Aufgabe ausreichend, da die Nachrichten so kurz sind, dass sie innerhalb der maximalen Länge eines Paketes Platz finden. Somit sind Probleme mit sich überholenden Paketen, die in Netzen auftreten können, hier ausgeschlossen.

UDP-Nachrichten werden in einem IP-Telegramm versendet; sie haben folgenden Aufbau:

0	8	16	24	31	
Quellport		Zielport			
Länge		Prüfsumme			
Nutzdaten					}

Der Zielport ist in der Auswertesoftware mit dem Parameter `CFG_SERVER_PORT` konfiguriert. Die restlichen Felder des Kopfes werden von der Socket-Bibliothek des Betriebssystems ausgefüllt. Die Nutzdaten werden von der Auswertesoftware generiert und haben nachfolgenden Aufbau:



Das Kommando ist ein String bestehend aus zwei Zeichen, er kann die in Tabelle 4-7 aufgelisteten Werte annehmen. Der optionale Zeitstempel gibt den Wert einer intern

Kommando	Beschreibung
UP	Die Auswertesoftware wurde gestartet.
DN	Die Auswertesoftware wurde beendet.
EC	Es wurde das Schließen des Augenlids erkannt.
EO	Es wurde das Öffnen des Augenlids erkannt.
TL	Während der Augenverfolgung wurde das Auge verloren.

Tabelle 4-7: Übersicht der UDP-Kommandos

laufenden Uhr mit einer Auflösung von 100ms an und kann zur Plausibilisierung der Nachrichten genutzt werden. Die Nachrichten haben eine Gesamtlänge von 10 Byte.

4.4.2. Implementierung des Ausgabemoduls

Das Ausgabemodul hat die Aufgabe, einen UDP-Server bereitzustellen, der die oben beschriebenen Nachrichten entgegennimmt und verarbeitet. Die Ansteuerung der Relaiskarte erfolgt dann nach einer Strategie, die vorher per Konfiguration festgelegt wurde. Die verschiedenen Schaltstrategien sind gemäß den Anforderungen aus Kapitel 1.2.1:

1. **While:** Solange das Lid geschlossen (offen) ist, soll der Ausgang aktiv sein.
2. **Toggle:** Wenn das Lid geschlossen (geöffnet) wird, soll der Ausgang seinen Zustand wechseln.
3. **Timed:** Wenn das Lid geschlossen (geöffnet) wird, soll der Ausgang für eine einstellbare Zeit aktiv werden.

Die Implementierung des Ausgabemoduls erfolgte in der Programmiersprache Python, da hier das Aufsetzen eines UDP-Servers und die Zeichenkettenbehandlung der eintreffenden Nachrichten einfach zu handhaben sind.

Das Ausgabemodul besteht aus

1. dem Eingabeprozess, der die eintreffenden UDP-Nachrichten entgegennimmt, und dem

2. Ausgabeprozess.

Der Eingabeprozess ist in dem Python-Modul blactuator.py implementiert. Das Modul ist auch gleichzeitig das Hauptprogramm und verantwortlich für das Einlesen der Konfiguration und das Erzeugen der Prozesse. Der Ausgabeprozess wird in einem Zyklus von 100ms aufgerufen und verarbeitet in der Zwischenzeit eingetroffene Nachrichten in einem Zustandautomaten. Die Nachrichten sind als Ereignisse ausgelegt. Wenn eine Nachricht verarbeitet wurde, wird diese also gelöscht und liegt somit nicht mehr an.

In den folgenden Zustandsdiagrammen werden die Nachrichten als *msg* bezeichnet. Die Nachricht *on* ist je nach Konfiguration dem Schließen oder dem Öffnen des Lidschlags zugeordnet; die Nachricht *off* entsprechend dem gegenteiligen Ereignis. Der Zustandsautomat für den While-Modus wird in Abbildung 4-43 skizziert.

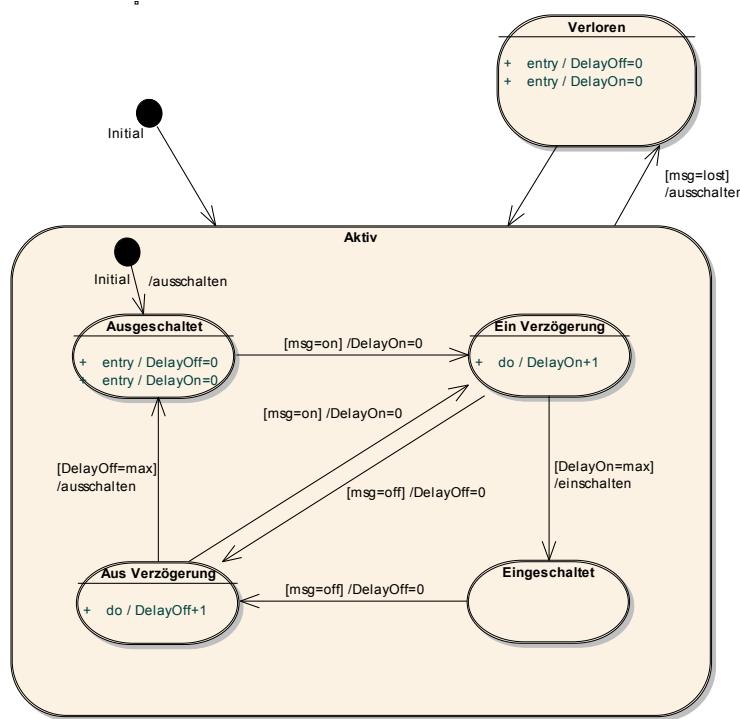


Abbildung 4-43: Zustandsautomat in dem While-Modus

Die Zustandsautomaten bieten für das Ein- und Ausschalten (*DelayOn*, *DelayOff*) Verzögerungen an, deren Zeit konfigurierbar ist. Wenn die Nachricht *lost* eintrifft (die Augenverfolgung hat versagt), so wird der Relaisausgang abgeschaltet und die Verzögerungszähler im Zustand *Verloren* werden gelöscht. Die Verarbeitung beginnt dann von vorne im *Aktiv*-Zustand. Die Zustandsautomaten für den Toggle- und Timed-Modus verhalten sich bezüglich der Verloren-Behandlung gleichartig, weshalb in den Diagrammen für den Toggle-Modus (Abbildung 4-44) und den Timed-

Modus (Abbildung 4-45) nur noch der jeweilige Aktiv-Zustand gezeigt wird.

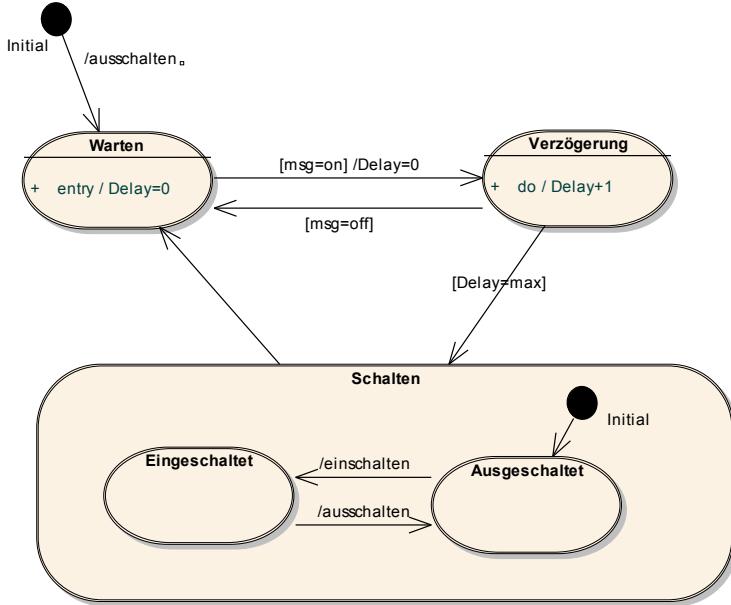


Abbildung 4-44: Zustandsautomat in dem Toggle-Modus

Die Aktionen „Einschalten“ und „Ausschalten“ steuern das Relais an. Zu diesem Zwecke wurden die der Karte²⁷ beiliegenden Treiber und Anwendungsbibliotheken verwendet. Abbildung 4-46 zeigt ein Foto der verwendeten Relaiskarte.

Da die Bibliotheken nur für die Sprache C zur Verfügung stehen, wurde mit Hilfe des Werkzeugs SWIG eine Umsetzung der C-Schnittstellen für Python generiert. So konnten die Bibliotheksfunktionen angesprochen werden. SWIG generiert Module für Python anhand von Schnittstellenbeschreibungen.

Die Schnittstellenbeschreibung für die Kartenbibliothek ist auf der CD-ROM unter `/software/outputmodule/swigdef` zu finden.

Das Ausgabemodul wird auf der Kommandozeile durch den Aufruf von

```
python blactuator.py
```

gestartet. Das Modul öffnet dann einen UDP-Server und wartet im Hintergrund auf die Nachrichten.

Verschiedene Parameter des Ausgabemoduls können über die Konfigurationsdatei `blactuator.cfg` konfiguriert werden. Die Tabelle 4-8 listet die wichtigen einstellbaren Parameter auf.

Der Quellcode für das Ausgabemodul ist auf der CD-ROM unter `/software/outputmodule` zu finden.

²⁷Bei der Relaiskarte handelt es sich um eine USBREL8-Karte von Conrad-Elektronik.

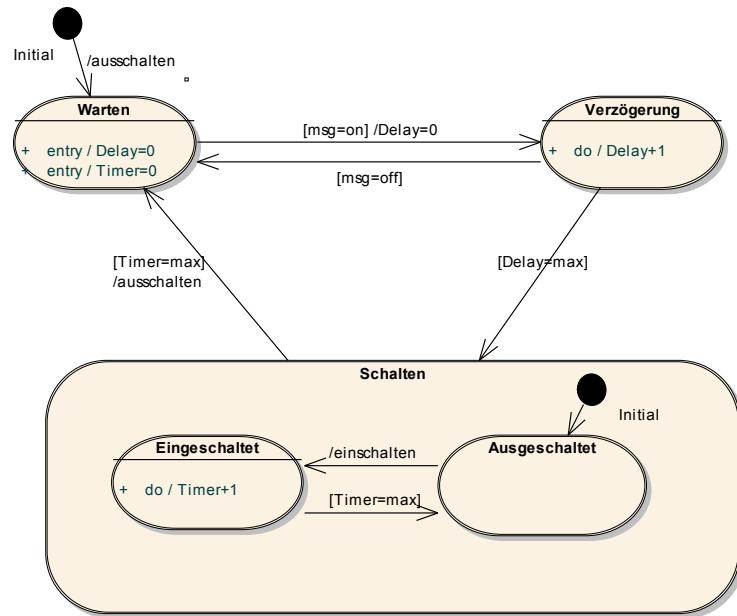


Abbildung 4-45: Zustandsautomat in dem Timed-Modus

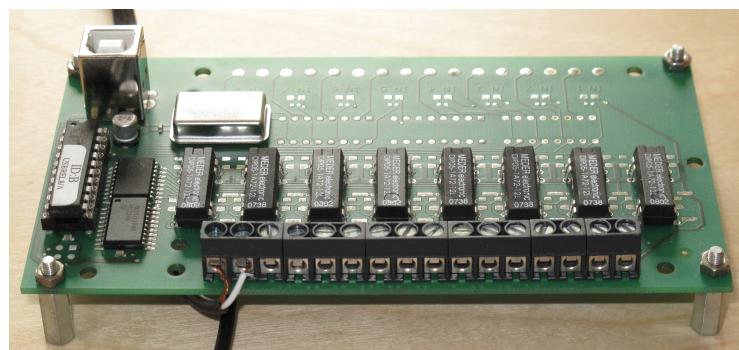


Abbildung 4-46: Relaiskarte

Parameter	Beschreibung
Sektion: hardware	
board	Auswahl der Relaiskarte (derzeit wird nur k8055 unterstützt)
port	UDP-Port, auf den sich der Server bindet
Sektion: k8055	
output	Kanalnummer der Relaiskarte
Sektion: event	
mode	Auswahl der Ansteuerstrategie: <code>while</code> , <code>toggle</code> oder <code>timed</code>
Sektion: modewhile , modetoggle , modetimed	
triggerevent	<code>close</code> =Lid schließen wird als <i>on</i> -Nachricht gewertet, <code>open</code> =Lid öffnen wird als <i>on</i> -Nachricht gewertet
minduration	Verzögerungszeit in 100ms-Raster
Sektion: modetimed	
ontime	Dauer des Einschaltzustandes in 100ms-Raster

Tabelle 4-8: Konfiguration des Ausgabemoduls

5. Untersuchung der Leistungsfähigkeit

Eine grundlegende Frage bei der Entwicklung von Bildverarbeitungssystemen ist es, mit welchem Aufwand²⁸ eine bestimmte Erkennungsleistung erreicht werden kann.

Da bei der vorliegenden Aufgabe bestimmte Anforderungen an die Reaktionszeit des Systems gestellt wurden, muss bei der Betrachtung der Erkennungsleistung auch die Laufzeit des Algorithmus bewertet werden.

Da die Parametrisierung des Algorithmus größtenteils experimentell bestimmt wurde und die Eingangsdaten durch ihre mehrdimensionale Struktur (Farbbild über Zeit) einen großen Kombinationsraum aufspannen, basiert die Untersuchung der Erkennungsleistung auf statistischen Methoden. Die Untersuchung der inhaltlichen Erkennungsleistung wurde dabei getrennt von der Laufzeitanalyse im Prototyp durchgeführt; die Laufzeitanalyse erfolgte in der Echtzeitsoftware.

Beide Untersuchungen wurden mit Testdaten durchgeführt, die mit der im Echtzeitssystem verwendeten Kamera aufgezeichnet wurden. Die Aufnahmen erfolgten mit einer Auflösung von 640x480 Bildpunkten und einer Bildrate von 20fps mit drei verschiedenen Personen und unter unterschiedlichen Bedingungen.

Für die Aufnahme wurden die Testpersonen angewiesen, ruhig zu sitzen und unterschiedlich lange zu blinzeln. Da die Information für die Algorithmen in dem Zustandswechsel des Augenlids von „offen“ nach „geschlossen“ und umgekehrt liegt, wurden für jedes Testvideo die Bildindizes derjenigen Stellen, an denen ein solcher Zustandswechsel auftritt, markiert. Diese Markierungen sind in ASCII-Dateien hinterlegt, so dass auf die Informationen zur Auswertung zurückgegriffen werden kann. Tabelle 5-9 beschreibt die Videoaufnahmen und in Stichworten die Aufnahmebedingungen. Die Testdaten sind in komprimierter Form auf der CD-ROM im Verzeichnis */testdaten* enthalten.

5.1. Erkennungsleistung

Die Erkennungsleistungen für die beiden wesentlichen Teilaufgaben „Augenlokalisierung“ und „Augenverfolgung“ werden getrennt voneinander betrachtet, da sie unterschiedliche Bedeutungen haben.

Die Augenlokalisierung sucht die Position der Augen anhand auftretender unwillkürlicher Lidschlagereignisse. Die Erkennungsleistung für die Lokalisierung sagt somit aus, wie oft der Benutzer unwillkürlich blinzeln muss, bis eine initiale Position gefunden wurde. Eine Leistung von 100% würde demnach bedeuten, dass mit einem einzelnen Lidschlag eine gültige Position ermittelt werden kann.

²⁸Dabei sind Aufwände einerseits Entwicklungsaufwände und andererseits Aufwände bzgl. der Rechenleistung der Plattform.

Videodatei	Beschreibung
alex1.avi	Testperson Alex; Tageslichtbeleuchtung seitlich durch ein Fenster
alex2.avi	Testperson Alex; künstliche Beleuchtung durch Halogenlampen von hinten und indirekt von oben; eine zweite Person durchläuft den Aufnahmebereich
bettina1.avi	Testperson Bettina; Brillenträgerin; Beleuchtung durch Halogenlampen von hinten und von vorne
bettina2.avi	Testperson Bettina; ohne Brille Beleuchtung durch Halogenlampen von hinten und von vorne; Weißabgleich fehlerhaft
arne1.avi	Testperson Arne; Beleuchtung durch Halogenlampen von hinten und von vorne; in Bewegung; Größe des Bildausschnitts nicht kalibriert
arne2.avi	Testperson Arne; Beleuchtung durch Halogenlampen von hinten, von vorne und von oben; in Bewegung; Größe des Bildausschnitts nicht kalibriert

Tabelle 5-9: Übersicht über die Testvideosequenzen

Die Erkennungsleistung der Augenverfolgung bewertet dagegen, mit welcher Wahrscheinlichkeit der korrekte Lidzustand des nun lokalisierten Auges ermittelt wird. Dieses Maß ist besonders wichtig, da das Schaltignal aus der Augenverfolgung heraus generiert wird.

Beide Untersuchungen wurden im Prototyp durchgeführt, was den Zugriff auf interne Daten für die Auswertung ermöglichte.

5.1.1. Erkennungsleistung der Augenlokalisierung

Zur Untersuchung der Augenlokalisierung über alle Lidschlagereignisse in einem Testvideo wurde der Prototyp modifiziert. Nach einer erfolgreichen Lokalisation und dem Wechsel in die Augenverfolgung wird nun eine gescheiterte Verfolgung simuliert, so dass die Verarbeitung wieder mit der Lokalisation beginnt. Dadurch kann jedes Videobild als Eingangsdatum für die Lokalisation der Augen dienen.

Der Prototyp liefert vier Listen, die jeweils denjenigen Bildindex im Video angeben, bei dem ein Ereignis stattfindet. Diese Listen sind die manuell markierten Bildindizes, an denen ein Übergang des Lids von offen nach geschlossen stattfand, der umgekehrte Fall von geschlossen nach offen, eine Liste, bei der ein Lidschlag durch den Lokalisationsalgorithmus erkannt wurde sowie eine Liste derjenigen Indizes, die von dem Lokalisationsalgorithmus zurückgewiesen wurden.

Diese Listen werden durch einen Auswertealgorithmus im MATLAB-Skript *mon-*

state.m weiterverarbeitet. Der Auswertealgorithmus vergleicht die manuell markierten Daten mit den von den Erkennungsalgorithmen stammenden Daten und bildet daraus statistische Aussagen über die Anzahl der korrekt erkannten Lidschläge. Zur Ermittlung der korrekt bzw. falsch erkannten Lidschläge wird für jeden vom Algorithmus erkannten Lidschlag folgende Strategie angewendet:

- Der Bildindex und dessen Umgebung werden in den Listen der manuell bestimmten Lidschlägen gesucht; Abbildung 5-47 zeigt das Prinzip dieser Umgebungssuche. Die schwarze Linie repräsentiert dabei den manuell markierten

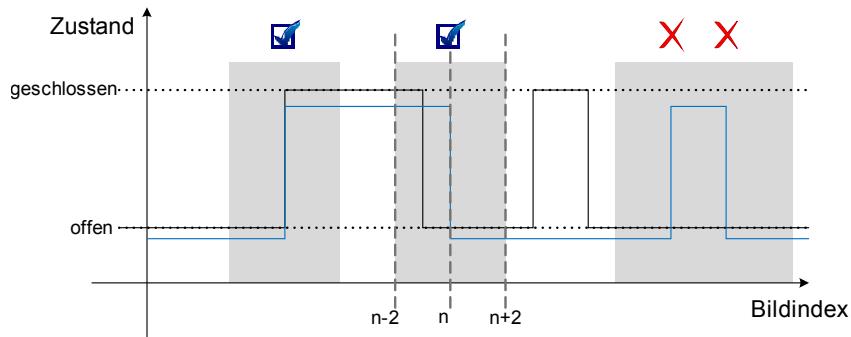


Abbildung 5-47: Umgebungssuche

Lidzustand, während die blaue Linie den erkannten Lidschlag aufzeigt. Die abgesuchten Umgebungen sind grau unterlegt. Die Umgebung umfasst einen Abstand von zwei Bildindizes und erlaubt somit eine entsprechende Unschärfe im Bildindex, da eine manuelle Markierung schon dann gesetzt werden kann, wenn das Lid noch im Schließen oder Öffnen begriffen ist.

- Wurde in der abgesuchten Umgebung ein markierter Lidschlag gefunden, so wird die Anzahl der korrekt gefundenen Lidschläge inkrementiert.
- Ist in der Umgebung kein Lidschlag vorhanden, so wird die Anzahl der fehl detektierten Lidschläge inkrementiert.

Aus der Anzahl der korrekt und falsch erkannten Übergängen werden die Erkennungswahrscheinlichkeiten nach folgenden Formeln berechnet:

$$P_k = \frac{k}{n_l} \quad (5.1)$$

$$P_f = \frac{f}{k + f} \quad (5.2)$$

Die Erkennungswahrscheinlichkeit P_k errechnet sich somit nach Gl. 5.1 aus der Anzahl k der korrekt erkannten Lidschläge im Verhältnis zur Anzahl n_l der manuell markierten Lidschläge. Die Wahrscheinlichkeit der Fehldetektion P_f ist nach Gl. 5.2 der Quotient aus der Anzahl der fehl detektierten Lidschläge f geteilt durch die Summe aller detektierten Lidschläge. Tabelle 5-10 zeigt die Ergebnisse dieser Gleichung.

gen zu den Testvideos sowie den Durchschnitt der Erkennungswahrscheinlichkeiten über alle Testvideos.

Video	n_l	k	f	P_k	P_f
alex1	66	65	5	0,985	0,071
alex2	62	50	0	0,807	0
bettina1	60	41	3	0,683	0,073
bettina2	99	87	1	0,879	0,014
arne1	74	12	0	0,162	0
arne2	96	27	0	0,281	0
\emptyset	76,17	47	1,5	0,633	0,026

Tabelle 5-10: Erkennungsleistung der Augenlokalisierung

Die Erkennungsleistung der Augenlokalisierung wird durch den Wert P_k bestimmt. Er liegt im Durchschnitt bei $P_k = 0,633$, das heißt, dass von 100 unwillkürlichen Lidschlägen 63 erkannt würden – anders gesagt sind im Durchschnitt 1,58 Lidschläge notwendig, bis eine initiale Position der Augen gefunden wird.

Die durchschnittliche Fehlerrate P_f liegt bei 0,026. Dies bedeutet, dass von 100 erkannten Lidschlägen 2,6 Lidschläge an falschen Positionen erkannt werden.

5.1.2. Erkennungsleistung der Augenverfolgung

Um die Erkennungsleistung der Augenverfolgung zu bewerten, wurden aus dem Prototyp die errechneten Lidzustände dem Auswerteskript *monstatetrack.m* zugeführt. Das Skript ermittelt folgende Kennzahlen aus den Daten des Prototyps:

1. Die Anzahl der Bilder f_a nach der Erstinitialisierung der Verfolgung,
2. die Anzahl der Bilder f_v , bei denen die Augenverfolgung einen gültigen Wert liefert,
3. die Anzahl der Bilder f_l , bei denen die Augenverfolgung die Position der Augen verloren hat,
4. die Anzahl der Bilder f_s , bei denen das Resultat des Lidzustands mit dem echten Lidzustand in einer Umgebung von $+ - 2$ Bildern übereinstimmt und
5. die Anzahl der Bilder f_d , bei denen das Resultat mit dem echten Lidzustand nicht übereinstimmt.

Aus diesen Werten errechnet das Skript die folgenden Kennwerte:

$$P_l = \frac{f_l}{f_a} \quad (5.3)$$

$$P_v = \frac{f_s}{f_v} \quad (5.4)$$

$$P_d = \frac{f_d}{f_v} \quad (5.5)$$

Der Wert P_l nach Gl. 5.3 gibt die Wahrscheinlichkeit dafür an, dass die Verfolgung die Position der Augen verliert und das Programm wieder von neuem mit der Lokalisierung beginnen muss. P_v nach Gl. 5.4 gibt hingegen die Wahrscheinlichkeit dafür an, dass das ermittelte Resultat des Lidzustands dem wahren Lidzustand entspricht. Analog hierzu drückt P_d aus Gl. 5.5 die Wahrscheinlichkeit dafür aus, dass das Ergebnis falsch ist; P_d lässt sich auch ausdrücken als $1 - P_v$. Tabelle 5-11 zeigt die Kennwerte zu den Testdaten sowie deren Durchschnitt.

Video	f_{ges}	f_a	f_v	f_l	f_s	f_d	P_l	P_v	P_d
alex1	897	873	764	109	590	174	0,125	0,772	0,228
alex2	892	842	679	163	657	22	0,194	0,968	0,032
bettina1	1516	1476	1292	184	813	479	0,13	0,629	0,371
bettina2	1718	1695	981	714	549	432	0,421	0,56	0,44
arne1	1700	1380	1011	369	687	324	0,267	0,680	0,32
arne2	1740	1691	1158	533	1065	93	0,031	0,920	0,08
\emptyset	1410,5	1326,17	980,83	345,33	726,83	254	0,195	0,753	0,247

Tabelle 5-11: Erkennungsleistung der Augenverfolgung

Die durchschnittliche Erkennungsleistung des Lidzustands liegt über die Testdaten gemittelt bei $P_v = 0,753$, d.h. dass in ca. 75% des Videostroms, bei der die Augenverfolgung eine gültige Position ermittelt hat, eine richtige Aussage über den Lidzustand getroffen wird. In durchschnittlich 25% ($P_d = 0,247$) des Videostroms wird allerdings eine Falschaussage getroffen.

Im Durchschnitt liegt die Wahrscheinlichkeit dafür, dass die Verfolgung ihre Position verliert, bei $P_l = 0,195$, also bei ca. 20%.

Unter den Testdaten sind jedoch insbesondere die drei Aufnahmen bettina2, arne1 und arne2 unter Voraussetzungen aufgenommen worden, die außerhalb der Auslegung der Filter liegen. Es ist daher zu erwarten, dass die Erkennungsleistung deutlich höher liegen kann, wenn der Kalibrierprozess eingehalten wird.

5.2. Laufzeitanalyse

Die Laufzeit des Algorithmus wurde mit der Echtzeitversion der Software untersucht, die dazu mit Hilfe des GNU-Profilierung-Werkzeugs *gprof* instrumentiert wurde. Als Eingangsdaten für den Algorithmus wurden die Testvideosequenzen alex1 und alex2 eingelesen und verarbeitet. Während des Programmlaufs wurden dann statistische Daten erzeugt, die durch das *gprof*-Werkzeug ausgewertet wurden.

Die Umgebung, in der die Laufzeitanalyse durchgeführt wurde, bestand aus folgenden Komponenten:

- Intel Core2Duo 2,2GHz Prozessor
- 2GB Arbeitsspeicher mit 667 MHz Busfrequenz
- Linux-Kernel 2.6.28
- GNU-Compiler 4.3.1

Die wichtige Information zur Laufzeit befindet sich in der Aufrufdauer der Funktion *CBlinkProcessor.process()*²⁹, da dies der Einsprungspunkt für den vollständigen Algorithmus ist. Die Laufzeit dieser Funktion ist somit mit der Laufzeit des gesamten Algorithmus gleichzusetzen. Tabelle 5-12 stellt die Laufzeit für mehrere Durchläufe der Testvideos sowie den daraus resultierenden Durchschnitt dar.

Testdurchlauf	Laufzeit (ms)
alex1,1	32
alex1,2	28
alex1,3	32
alex2,1	36
alex2,2	36
alex2,3	38
∅	33,67

Tabelle 5-12: Laufzeitverhalten der Software

Die durchschnittliche Laufzeit des Algorithmus beträgt 33,67ms, was zu einer Bildrate von

$$r = \frac{1}{33,67\text{ms}} = 29,7 \frac{\text{Bilder}}{\text{s}} \quad (5.6)$$

führt. Da Lidschläge schon mit einer Bildrate von $r = 20 \frac{\text{Bilder}}{\text{s}}$ aufgelöst werden, reicht diese Bildrate für die Problemstellung aus.

²⁹Siehe hierzu die Beschreibung der Implementierung in Kap. 4.3.2.

6. Zusammenfassung und Ausblick

Die vorliegende Arbeit hat gezeigt, dass ein technisches System realisierbar ist, dass Steuerinformation ableiten kann, indem es den Lidschlag einer Person analysiert.

6.1. Erfüllungsgrad der Anforderungen

Das System sollte den bereits vorhandenen optoelektronischen Lidschlagsensor ersetzen. Die Hauptanforderung bestand also darin, zumindest eine ebenbürtige Funktionalität zu dem bestehenden Sensor zu erreichen.

Dem abschließenden Urteil der Eltern zufolge liefert das neue System eine subjektiv deutlich bessere Erkennung verglichen mit dem ursprünglichen Sensor. Hieraus lässt sich zum einen ableiten, dass die Anforderungsdefinitionen³⁰ bereits richtig aufgestellt wurden und dass zum anderen diese Anforderungen auch durch das System erfüllt werden können.

6.1.1. Erfüllungsgrad der funktionalen Anforderungen

Tabelle 6-13 stellt die funktionalen Anforderungen und deren Erfüllungsgrad gegenüber.

Anforderung	Erfüllungsgrad
1f	Erfüllt.
2f	Erfüllt; durch die Verwendung einer Kamera ist der Abstand zur Person wählbar.
3f	Erfüllt mit einer Gesamterkennungswahrscheinlichkeit von 75%.
4f	Erfüllt; das Schaltsignal wird über UDP bereitgestellt.
5f	Erfüllt; die Relaisplatine wird durch den Lidschlag geschaltet.
6f	Erfüllt; die Schaltstrategie ist konfigurierbar.
7f	Erfüllt; die Reaktionszeit des Systems liegt bei unter 400ms.

Tabelle 6-13: Erfüllungsgrad der funktionalen Anforderungen

³⁰Wie in Kapitel 1 festgelegt, wurden die Anforderungen größtenteils aus den praktischen Erfahrungen mit dem bereits vorhandenen Sensor abgeleitet.

6.1.2. Erfüllungsgrad der nichtfunktionalen Anforderungen

Tabelle 6-14 stellt die nichtfunktionalen Anforderungen und deren Erfüllungsgrad gegenüber.

Anforderung	Erfüllungsgrad
1n	Erfüllt, aber Kalibrierungsprozess notwendig.
2n	Erfüllt, aber Kalibrierungsprozess notwendig.
3n	Erfüllt; das System verwendet eine Standard-Webcam sowie einen Standard-PC.
4n	Teilweise erfüllt; keine Bedienung durch Nutzer notwendig, allerdings ist die Installation noch nicht implementiert.
5n	Erfüllt; eine Portierung auf andere Plattformen als Linux ist denkbar.
6n	Erfüllt; die verwendete Relaisplatine ist über USB angebunden und ermöglicht ein potentialfreies Schalten.

Tabelle 6-14: Erfüllungsgrad der nichtfunktionalen Anforderungen

6.2. Potentielle Verfahrensverbesserungen

Das hier vorgestellte System bietet jedoch immer noch Potential für Verbesserungen und Erweiterungen des Verfahrens, die hier kurz erwähnt werden sollen. Die gesamte Erkennungsleistung liegt bereits bei 75%. Eine hundertprozentige Erkennung wird bei einem Bildverarbeitungssystem nicht zu erreichen sein; die Erkennungsleistung sollte sich jedoch erhöhen, wenn eine höhere Anzahl an unterschiedlichen Eingangsdaten zur Parametrisierung herangezogen wird. Die Menge der unterschiedlichen Datensätze von drei Personen ist im Vergleich als niedrig anzusehen.

Die Erkennungsleistung könnte ebenfalls noch erhöht werden, wenn weitere Kriterien zur Lidzustandserkennung herangezogen werden könnten. Solche weiteren Merkmale sind beispielsweise charakteristische Hell-/Dunkelübergänge von Augapfel und Pupille, die nur bei offenem Lid vorhanden sind.

Neben der Erkennungsleistung besteht auch Verbesserungspotential in der Handhabung des Systems. Das System muss mit Hilfe einer Ausrichtungsmaske zunächst kalibriert werden, um zu gewährleisten, dass die Augen des Benutzers mit einer Mindestgröße erscheinen und die Lage im Raum in gewissen Grenzen definiert ist. Es wäre demnach noch zu untersuchen, inwieweit weitere Verfahren oder Kriterien eine solche Kalibrierung überflüssig machen könnten.

Einen weiteren Optimierungsaspekt bildet die Anforderung an die Rechenleistung des Prozessors. Der entwickelte Algorithmus arbeitet auf einem 2,2GHz-Prozessor

mit genügend großer Geschwindigkeit. Im Hinblick auf die Kosten des Systems wäre jedoch der Einsatz sogenannter Netbooks zu bevorzugen, die allerdings nur eine Geschwindigkeit von 1,6GHz haben, was zur Ausführung des Algorithmus in Echtzeit nicht ausreichend ist. Hier könnte eine Optimierung darin bestehen, die Bildauflösung der Eingangsdaten zu reduzieren. Es wäre dann zu untersuchen, ob die Erkennungsqualität erhalten bleibt und die Anforderung an die Rechenleistung entsprechend sinkt.

6.3. Fazit

Abschließend soll ein besonderer Aspekt dieser Arbeit hervorgehoben werden:

Sie hat neben allen technischen Aspekten auch gezeigt, dass durch den Einsatz von Technik Menschen, deren Interaktionsmöglichkeiten durch Behinderungen eingeschränkt sind, ermöglicht werden kann, ihre Interaktion zu erweitern und so ihre Lebensqualität zu verbessern.

Dies sollte auch weiterhin die Kernaufgabe der Technik sein: Die Technik soll dem Menschen dienen, seine Lebensqualität zu verbessern.

A. Literaturverzeichnis

Literatur

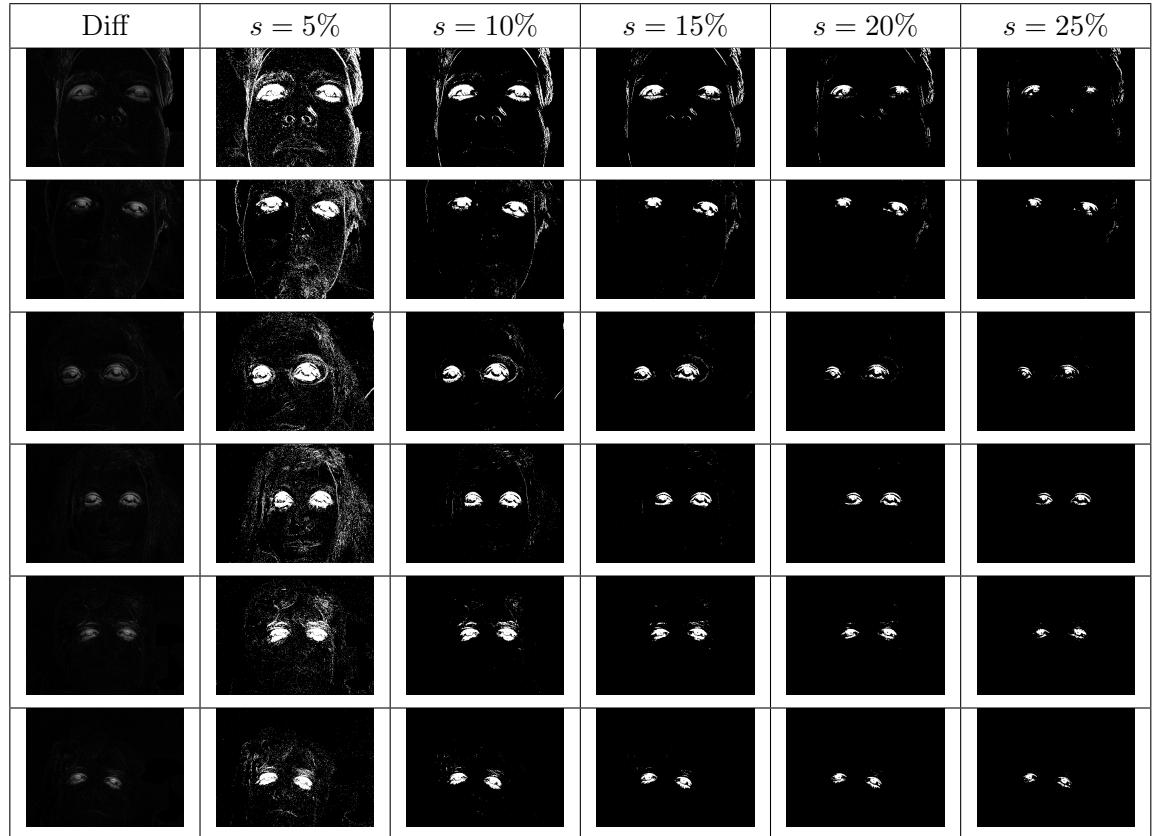
- [BGF] BETKE, Margrit ; GIPS, James ; FEMING, Peter: The Camera Mouse: Visual Tracking of Body Features to Provide Computer Access for People With Severe Disabilities. In: *IEEE Transactions on neural systems and rehabilitation engineering* 10, März 2002, S. 1–10
- [BMM00] BETKE, Margrit ; MULLALLY, William J. ; MAGEE, John J.: Active detection of eye scleras in real time. In: *IEEE CVPR Workshop on Human Modeling, Analysis and Synthesis*, 2000
- [Dou] DOUGLASS, Bruce P.: Architecting Real-Time Systems with Patterns. In: *Embedded Systems Conference* 1, 1998, S. 1–38
- [ELE] ELECOK, Beratungssystem: *Arbeitskreis ELECOK*, <http://www.elecok.de>. – Stand 28. April 2009
- [GBGB] GRAUMAN, Kristen ; BETKE, Margrit ; GIPS, James ; BRADSKI, Gary R.: Communication via Eye Blinks – Detection and Duration Analysis in Real Time. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1, Dezember 2001, S. 1010–1017
- [Hor08] HORNICEK, Gerhard: *Projektaufgabenstellung für eine Diplom-/ Masterarbeit*. Januar 2008
- [Kle] KLEIN, Martin: *Implantatfixierte Orbitaepithese mit myoelektrisch gesteuertem beweglichem Oberlid*, Diss., <http://edoc.hu-berlin.de/docviews/abstract.php?id=20027>. – Stand 28. April 2009
- [Lew95] LEWIS, J. P.: Fast normalized cross-correlation. In: *Vision Interface*, Canadian Image Processing and Pattern Recognition Society, 1995, 120–123
- [RH06] REUSSNER, Ralf ; HASSELBRING, Wilhelm: *Handbuch der Software-Architektur*. Heidelberg, 2006
- [RHQ⁺05] RUPP, Chris ; HAHN, Jürgen ; QUEINS, Stefan ; JECKLE, Mario ; ZENGLER, Barbara: *UML2 glasklar: Praxiswissen für die UML-Modellierung und -Zertifizierung*. 2. Auflage,. München, 2005
- [RKG97] REINDERS, Marcel J. ; KOCH, R.W.C. ; GERBRANDS, Jan J.: Locating Facial Features in Image Sequences using Neural Networks. In: *2nd International Conference on Automatic Face and Gesture Recognition*, 1997, S. 230–235

- [SBB02] SCHAEL, Marc ; BAHLMANN, Claus ; BURKHARDT, Hans: *Praktikumsversuch Klassifikatorentwurf*. Albert-Ludwigs-Universität Freiburg, Februar 2002
- [SSSS01] SHAPIRO, Linda G. ; STOCKMAN, George C. ; SHAPIRO, Linda G. ; STOCKMAN, George: *Computer Vision*. Prentice Hall, 2001
- [VPY02] VENKATESH, B.S. ; PALANIVEL, S. ; YEGNANARAYANA, B.: Face detection and Recognition in an Image Sequence using Eigenedginess. In: *Indian Conference on Computer Vision, Graphics and Image Processing, Ahmedabad, India*, 2002, S. 97–101
- [Wah89] WAHL, Friedrich M.: *Digitale Bildsignalverarbeitung – Grundlagen, Verfahren, Beispiele*. Berlin, 1989
- [YKA] YANG, Ming-Hsuan ; KRIEGMAN, David J. ; AHUJA, Narendra: Detecting Faces in Images: A Survey. In: *IEEE Transactions on Pattern Analysis and machine intelligence* , Januar 2002, S. 34–58
- [ZQ] ZHIWEI, Zhu ; QIANG, Ji: Robust Pose Invariant Facial Feature Detection and Tracking in Real-Time. In: *International Conference on Pattern Recognition* 1, 2006, S. 1092–1095

B. Experimentelle Auswertungen

B.1. Augenlokalisierung

B.1.1. Ermittlung der Binarisierungsschwelle

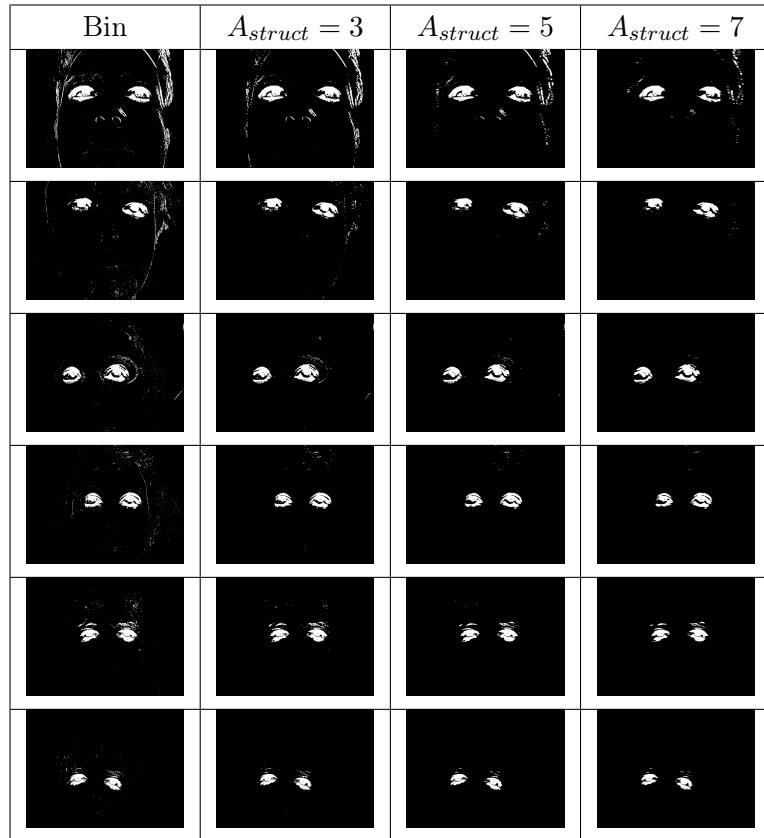


Die optimale Schwelle für die Binarisierung liegt bei $s = 10\%$. Diese Schwelle bewirkt, dass die Augenlider noch hervortreten, aber andere Bewegungen und Bildrauschen ausgeblendet werden. Bei den Schwellen mit $s < 10\%$ treten die Störungen wieder stärker hervor. Wird eine Schwelle von $s > 10\%$ ausgewählt, so besteht die Gefahr, dass nicht mehr das vollständige Lid erfasst und die Fläche optisch zerrissen wird.

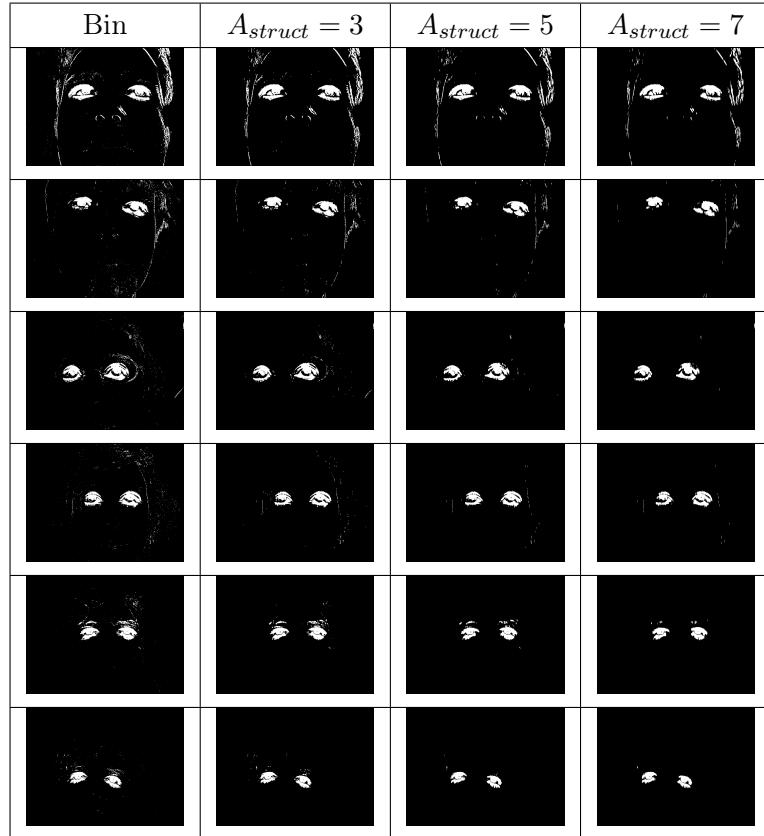
B.1.2. Ermittlung des Strukturelements

B.1.3. Strukturelemente

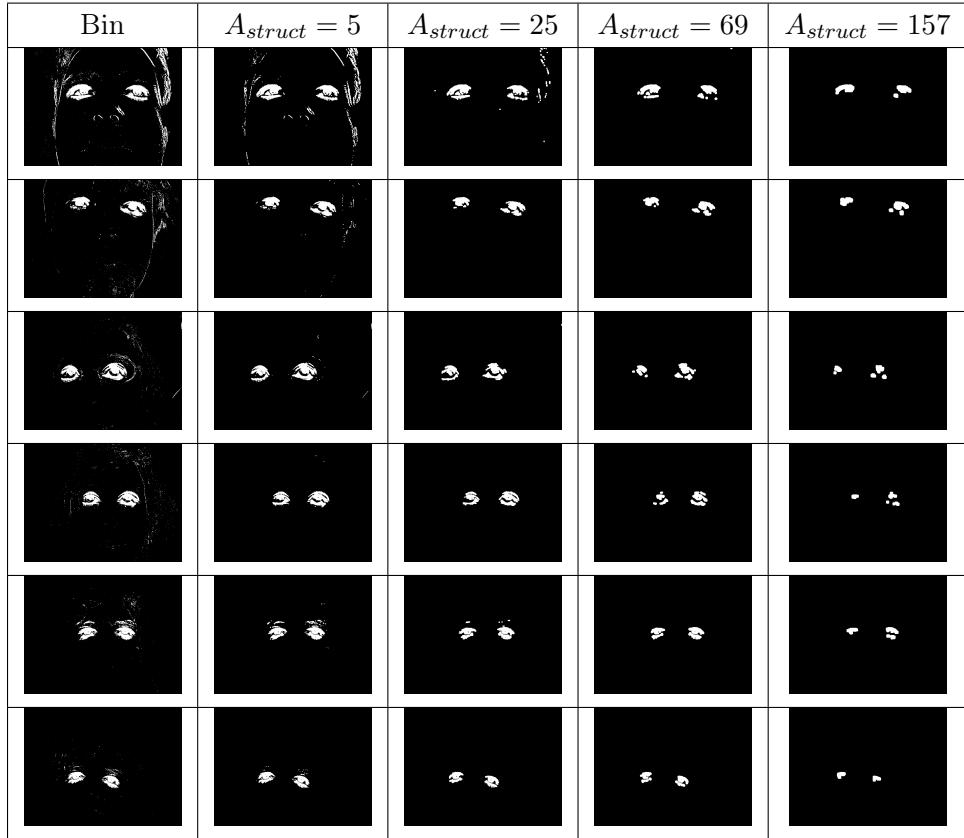
Element: horizontale Linie



Die Anwendung der morphologischen Operation mit unterschiedlich großen horizontalen Linien ergibt eine Verbesserung von Störungen durch Rauschen. Ebenso wird die vertikale Bewegung an Kanten des Gesichts wirkungsvoll eliminiert. Die Lidregionen werden jedoch nicht zu einer homogenen Fläche verschmolzen.

Element: vertikale Linie

Die Anwendung der morphologischen Operation mit unterschiedlich großen vertikalen Linien ergibt eine Verbesserung gegen Störungen durch Rauschen. Die störenden vertikalen Bewegungen werden hier jedoch nicht eliminiert. Eine Verschmelzung der Lidflächen findet vergleichbar mit dem Element der horizontalen Linie nicht statt.

Element: Scheibe

Die morphologische Bearbeitung mit dem Strukturelement Scheibe zeigt ab einer Strukturfläche $A_{struct} \geq 25$ Wirkung gegen Störungen durch Bildrauschen. Die störenden vertikalen Bewegungen können wirkungsvoll ausgeblendet werden. Die optimale Glättung im Lidbereich, die zu homogenen Flächen führt, ist mit einer Wirkfläche von $A_{struct} = 69$ gegeben.

B.1.4. Parametrisierung des anthropologischen Modells

Um die Parameter zur anthropologischen Filterung bestimmen zu können, wurden die Eigenschaften von zehn Lidschlagereignissen ausgewertet. Die Auswertung erfolgte über das Extrahieren der geometrischen Eigenschaften aus den Binärbildern während den Lidschlügen. Die Eigenschaften wurden in Tabelle B-48 zusammengetragen und daraus die Kennwerte ermittelt. Die Regionen sind in der Tabelle bereits zu allen Paarkombinationen zusammengeführt. Die Paare, die den echten Lidschlügen entsprechen, sind grau unterlegt dargestellt.

Index	Region 1								Region 2								xdelta	ydelta	distance	distangle	normwidth1	normwidth2	basenormwidth1	basenormwidth2	aspectratio	arareatio	ovvalid	validity
Region 1	Region 2	Mx	My	Bbox-w	Bbox-h	Area	Aspect	Mx	My	Bbox-w	Bbox-h	Area	Aspect	xdelta	ydelta	distance	distangle	normwidth1	normwidth2	basenormwidth1	basenormwidth2	aspectratio	arareatio	ovvalid	validity			
-f- nr:25																												
1	2	231.607	164.688	84	43	2258	1.950	229.150	198.493	76	21	935	3.620	2.457	33.805	33.900	85.800	2.480	2.240	0.000	0.000	0.540	0.414	0	0.000			
1	3	231.607	164.688	84	43	2258	1.950	467.466	167.691	76	40	1985	1.900	235.859	3.003	236.000	0.729	0.356	0.322	0.799	0.988	0.973	0.879	1	3.640			
2	3	229.150	198.493	76	21	935	3.620	467.466	167.691	76	40	1985	1.900	238.316	30.802	240.000	7.360	0.316	0.316	0.969	0.969	0.525	0.471	0	0.000			
-f- nr:66																												
1	2	183.890	171.431	89	46	2239	1.930	183.413	204.258	67	21	791	3.190	0.476	32.827	32.800	89.200	2.710	2.040	0.000	0.000	0.606	0.353	0	0.000			
1	3	183.890	171.431	89	46	2239	1.930	423.895	156.005	83	39	1990	2.130	240.005	15.426	241.000	3.680	0.370	0.345	0.722	0.860	0.909	0.889	1	3.380			
2	3	183.413	204.258	67	21	791	3.190	423.895	156.005	83	39	1990	2.130	240.482	48.253	245.000	11.300	0.273	0.338	0.610	0.898	0.667	0.397	1	2.570			
-f- nr:91																												
1	2	242.095	169.339	85	37	1986	2.300	240.134	198.787	69	22	876	3.140	1.962	29.448	29.500	86.200	2.880	2.340	0.000	0.000	0.732	0.441	0	0.000			
1	3	242.095	169.339	85	37	1986	2.300	470.933	168.809	72	33	1725	2.180	228.838	0.530	229.000	0.133	0.371	0.315	0.714	0.955	0.950	0.869	1	3.490			
2	3	240.134	198.787	69	22	876	3.140	470.933	168.809	72	33	1725	2.180	230.800	29.978	233.000	7.400	0.296	0.309	0.804	0.911	0.696	0.508	1	2.920			
-f- nr:93																												
1	2	240.684	169.665	96	38	2205	2.530	237.313	198.225	67	21	770	3.190	3.371	28.559	28.800	83.300	3.340	2.330	0.000	0.000	0.792	0.349	0	0.000			
1	3	240.684	169.665	96	38	2205	2.530	474.986	169.066	81	36	1760	2.250	234.302	0.599	234.000	0.147	0.410	0.346	0.502	0.857	0.891	0.796	1	3.050			
2	3	237.313	198.225	67	21	770	3.190	474.986	169.066	81	36	1760	2.250	237.673	29.159	239.000	6.990	0.280	0.338	0.665	0.899	0.705	0.438	1	2.710			
-f- nr:119																												
1	2	239.571	168.457	88	38	2058	2.320	236.656	197.776	68	22	812	3.090	2.914	29.319	29.500	84.300	2.990	2.310	0.000	0.000	0.749	0.395	0	0.000			
1	3	239.571	168.457	88	38	2058	2.320	467.080	168.003	76	35	1774	2.170	227.509	0.453	228.000	0.114	0.387	0.334	0.629	0.922	0.938	0.862	1	3.350			
2	3	236.656	197.776	68	22	812	3.090	467.080	168.003	76	35	1774	2.170	230.423	29.772	232.000	7.360	0.293	0.327	0.772	0.961	0.703	0.458	1	2.890			
-f- nr:159																												
1	2	238.649	163.351	94	43	2358	2.190	235.520	197.039	76	20	990	3.800	3.129	33.689	33.800	84.700	2.780	2.250	0.000	0.000	0.575	0.420	0	0.000			
1	3	238.649	163.351	94	43	2358	2.190	475.407	161.526	71	36	1549	1.970	236.758	1.825	237.000	0.442	0.397	0.300	0.572	0.832	0.902	0.857	1	2.960			
2	3	235.520	197.039	76	20	990	3.800	475.407	161.526	71	36	1549	1.970	239.887	35.514	243.000	8.420	0.313	0.293	0.945	0.773	0.519	0.639	0	0.000			
-f- nr:281																												
1	2	231.973	173.838	91	40	2188	2.270	229.813	203.346	65	20	812	3.250	2.160	29.508	29.600	85.800	3.080	2.200	0.000	0.000	0.700	0.371	0	0.000			
1	3	231.973	173.838	91	40	2188	2.270	465.426	172.944	78	34	1719	2.290	233.453	0.894	233.000	0.219	0.390	0.334	0.612	0.922	0.992	0.786	1	3.310			
2	3	229.813	203.346	65	20	812	3.250	465.426	172.944	78	34	1719	2.290	235.613	30.402	238.000	7.350	0.274	0.328	0.613	0.954	0.706	0.472	1	2.750			
-f- nr:390																												
1	2	206.174	176.714	94	39	2256	2.410	207.534	209.953	72	28	927	2.570	1.360	33.238	33.300	87.700	2.830	2.160	0.000	0.000	0.937	0.411	0	0.000			
1	3	206.174	176.714	94	39	2256	2.410	439.945	171.505	79	36	1802	2.190	233.771	5.209	234.000	1.280	0.402	0.338	0.544	0.901	0.910	0.799	1	3.150			
2	3	207.534	209.953	72	28	927	2.570	439.945	171.505	79	36	1802	2.190	232.411	38.448	236.000	9.390	0.306	0.335	0.880	0.915	0.853	0.514	1	3.160			
-f- nr:630																												
1	2	194.213	185.078	83	40	2071	2.080	194.232	214.516	70	18	822	3.890	0.020	29.438	29.400	90.000	2.820	2.380	0.000	0.000	0.534	0.397	0	0.000			
1	3	194.213	185.078	83	40	2071	2.080	421.490	180.284	70	40	1709	1.750	227.278	4.794	227.000	1.210	0.365	0.308	0.749	0.899	0.843	0.825	1	3.320			
2	3	194.232	214.516	70	18	822	3.890	421.490	180.284	70	40	1709	1.750	227.258	34.232	230.000	8.570	0.305	0.305	0.872	0.872	0.450	0.481	0	0.000			
-f- nr:729																												
1	2	200.919	187.502	84	37	1692	2.270	432.640	181.644	65	33	1340	1.970	231.721	5.858	232.000	1.450	0.362	0.280	0.765	0.670	0.868	0.792	1	3.090			

Abbildung B-48: Regioneneigenschaften