Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Mosawer Ahmad Nurzai

Visuelle Spracherkennung für Assistenzroboter durch Dynamic-time-warping

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Mosawer Ahmad Nurzai

Visuelle Spracherkennung für Assistenzroboter durch Dynamic-time-warping

Masterarbeit eingereicht im Rahmen der Masterprüfung

im Studiengang Master of Science Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Andreas Meisel
Zweitgutachter: Prof. Dr. Wolfgang Fohl

Eingereicht am: 31. März 2016

Mosawer Ahmad Nurzai

Thema der Arbeit
Visuelle Spracherkennung für Assistenzroboter durch Dynamic-time-warping

Stichworte
Spracherkennung, visuell, Assistenzroboter, Roboter, DTW, Dynamic Time Warping, ROS, Kinect

Kurzzusammenfassung
Der Einsatz einer akustischen Spracherkennung in der Industrie zur Steuerung/Bedienung von Robotern beherbergt Risiken. Durch eine laute Geräuschkulisse können Kommandos vom System falsch interpretiert werden. In dieser Masterarbeit wird ein visuelles Spracherkennungssystem entwickelt. Die Lippen von Menschen werden mithilfe von Bildverarbeitung gelesen. Es werden Kurzkommandos von einem Assistenzroboter wiedererkannt. Das visuelle Spracherkennungssystem kann unterstützend für ein akustisches Spracherkennungssystem eingesetzt werden. In der Auswertung wird gezeigt, dass das System Kurzkommandos zuverlässig wiedererkennen kann.

Mosawer Ahmad Nurzai

Title of the paper
Visual speech recognition for assistant robots through Dynamic-time-warping

Keywords
speech recognition, visual, assistant robot, robot, DTW, dynamic time warping, ROS, Kinect

Abstract
The use of an acoustic speech recognition in the industry for controlling/handling a robot entails risks. Commands can be misinterpreted from the system in case of high background noise. In this master thesis a visual speech recognition system is developed. The lips of people are read by image processing. Short commands are recognised by an assistant robot. The visual speech recognition system can be used to support an acoustic speech recognition system. In the evaluation, it is shown that the system can reliably recognise short commands.

Inhaltsverzeichnis

1 Einleitung

Die menschliche Sprache ist in seiner Natur *bimodal*, d.h. audiovisuell. Bei einer Aussprache entstehen akustische Wellen, welche die Menschen als das Gehörte wahrnehmen. Lippenbewegungen entstehen ebenfalls bei einer Aussprache. Gemeinsam mit der Bewegung der Zunge und anderen Gesichtsmuskeln kann dies als das Gesehene bezeichnet werden. In der Mensch-zu-Mensch-Kommunikation sind die Hauptinformationskanäle die Sprache (durch die Akustik) und die visuellen Ausdrücke (durch die Lippenbewegung). Beide ergänzen sich und steigern gemeinsam die Effektivität der Kommunikation. Der McGurk-Effekt (2.1) zeigt, dass Sprache nicht nur anhand der Akustik wahrgenommen wird, sondern sie enthält auch visuelle Hinweise. In lauten Umgebungen ist es notwendig sich an visuellen Hinweisen zu orientieren, beispielsweise um sich auf ein Gespräch zu fokussieren (Tsuhan [2001]).

1.1 Motivation

(a) Bedürfnis (b) Befehl (c) Bewerkstelligung

Abbildung 1.1: Szenario: „Hol mir Kaffee!“

In Abbildung 1.1 ist ein typisches Szenario abgebildet, wenn der Mensch an einen assistierenden/helfenden Roboter denkt. In dieser Arbeit wird ein Assistenzroboter (Abschnitt 3.9) eingesetzt, der die Lippen lesen soll. Folgendes Szenario veranschaulicht die Begegnung

Mensch und Roboter in diesem Zusammenhang. Hinter dem Szenario „Hol mir Kaffee!“ stecken verschiedene Schritte. Die drei „B“'s (**B**edürfnis, **B**efehl, **B**ewerkstelligung) stellen diese Schritte dar. Die Arbeit konzentriert sich um das mittlere „B“, um den Befehl. In dieser Phase wird der Befehl vom Roboter wiedererkannt, sei es auf visueller (Lippenlesen) oder akustischer (Spracherkennung) Basis.
Lippenlesen kann in verschiedenen Situationen und Anwendungen unterstützend und hilfreich sein. Lippenlesen kann genutzt werden, um beispielsweise das Gesehene als Textformat in einem Computer-System zu übertragen. Diese Möglichkeit bietet nun weitere Implementierungen in verschiedenen Anwendungsgebieten. Lippenlesen wird von hörbeeinträchtigen Menschen als Kommunikationsmittel (Sprachauffassung) genutzt. Menschen mit Hörschädigung sind oftmals isoliert und die Teilhabe am gesellschaftlichen Leben wird erschwert. Mithilfe einer visuellen Spracherkennung kann die Kommunikation zwischen hörbeeinträchtigten Menschen und Menschen mit gesundem Hörorgan möglich gemacht werden. Da nicht jeder Gebärdensprache beherrscht, kann dieses Hindernis durch ein automatisiertes Lippenlesesystem überwunden werden (Al-Ghanim u. a. [2013]).
Ein weiteres Anwendungsgebiet ist die Medizin. Unter Aufsicht von qualifiziertem Personal können die Lippen von gehörbedingt sprachbeeinträchtigten Menschen gelesen werden, um zu messen und zu beurteilen wie erfolgreich der Patient ist, wenn er versucht eine Mimik des Mundes nachzuahmen. So kann die Qualität der Aussprache bewertet werden (Yargic und Dogan [2013]).
Bei akustischen Spracherkennungssystemen kann die visuelle Spracherkennung unterstützend eingesetzt werden. Ein Spracherkennungssystem, welches in der Industrie bei Robotern eingesetzt wird, um beispielsweise Anweisungen zu geben, ist ein mögliches Anwendungsgebiet für die visuelle Spracherkennung. In der Industrie herrscht oftmals eine laute Geräuschkulisse, somit kann es hilfreich sein, zusätzlich zur akustischen Spracherkennung, das Lippenlesen, also eine visuelle Spracherkennung, einzuführen (Yoshida u. a. [2010]).

1.2 Aufbau der Arbeit

- **2. Kapitel Grundlagen:** Im zweiten Kapitel werden die Grundlagen beschrieben. Es werden akustische und visuelle Spracheinheiten präsentiert, Lippenlesen aus Perspektive der Bildverarbeitung betrachtet und die Prozesskette zur Entwicklung eines visuellen Spracherkennungssystems vorgestellt. Verwendete Lippenmerkmale werden in Raumkurven, sogenannten Trajektorien repräsentiert. Die Grundlagen dazu werden ebenfalls präsentiert.

- **3. Kapitel Analyse:** Das Kapitel 3 umfasst die Analyse des zu entwickelnden Prototyps. Zwei grundlegende Vorgehen (modellbasiert und bildbasiert) werden vorgestellt. Eine Machbarkeitsanalyse zur visuellen Spracherkennung wird durchgeführt und vergleichbare Projekte vorgestellt. Die Prozessschritte der Prozesskette werden analysiert, dabei werden ähnliche Arbeiten vorgestellt. Um die Ähnlichkeit von Trajektorien zu messen wird der Algorithmus DTW (Dynamic-time-warping) eingesetzt. Dieser wird ebenfalls wird präsentiert. Aus der Analyse werden Problemstellungen diskutiert und Anforderungen an das zu realisierende visuelle Spracherkennungssystem gestellt. Es wird die Laborumgebung vorgestellt. Das visuelle Spracherkennungssystem wird von einem Roboter benutzt. Der Roboter, die Kamera und das Roboter-Betriebssystem werden vorgestellt.

- **4. Kapitel Lippenlesen:** Das fünfte Kapitel befasst sich mit der Realisierung des visuellen Spracherkennungssystems. Es wird das Design des Systems aufgezeigt und die Komponenten präsentiert. Die einzelnen Prozessschritte werden realisiert. Die Realisierung wird vorgestellt: die Lokalisierung des Gesichts, der Lippen, wie und welche Lippenmerkmale extrahiert werden, wie die Ähnlichkeitsanalyse der Trajektorien stattfindet und wie das Klassifizieren einer Äußerung zu einem Kommando realisiert wird.

- **5. Kapitel Auswertung:** Bei der Auswertung werden verschiedene Versuche durchgeführt. Insgesamt werden vier Versuche aufgebaut. Die Ergebnisse und Erkenntnisse aus den Versuchen werden bewertet. Anhand der Versuche, der Analyse und der Realisierung werden notwendige Erweiterungen des visuellen Spracherkennungssystems vorgestellt.

- **6. Kapitel Schluss:** Abschließend erfolgt im siebten Kapitel eine Zusammenfassung und Ausblick der Masterarbeit.

Es ist anzumerken, dass kursiv geschriebene Wörter im Glossar zu finden sind. Das Programm zum realisierten visuellen Spracherkennungssystem kann auf https://github.com/Mosa- gefunden werden. Einige Icons werden von VisualPharm [2016] genutzt.

1.3 Zielsetzung

Diese Arbeit befasst sich mit der Entwicklung eines Prototyps zur visuellen Spracherkennung für einen ROS-basierten Assistenzroboter. Der Roboter soll Kurzkommandos erkennen können. In der Entwicklung des Prototyps wird jeder Prozessschritt erläutert. Die Prozesskette beschreibt die notwendigen Schritte für die Entwicklung eines visuellen Spracherkennungssystems. Unter anderem beinhaltet die Prozesskette folgende Schritte:

- Gesichtslokalisierung/-erkennung
- Lippenlokalisierung/-erkennung
- Extraktion der Merkmale
- Training
- Klassifikation

Ein weiteres Ziel ist, die Reduktion der erforderlichen Rechenleistung bei der Ausführung der jeweiligen Methoden in der Prozesskette. Aufgrund der großen Datenmenge sowie der Komplexität der Berechenbarkeit ist der Rechenleistungsbedarf im Bereich der Bildverarbeitung sehr hoch.
In der Spracherkennung wird primär ein akustisches Spracherkennungssystem genutzt und die visuelle Komponente dient als Unterstützung. Der Prototyp soll als ein unterstützendes System eingesetzt werden. Das Hauptaugenmerk der Arbeit liegt auf die Entwicklung dieses Prototyps.

2 Grundlagen

Das Kapitel beschreibt die Grundlagen. Es wird darauf eingegangen, wie Menschen die Lippen lesen, was dabei die Schwierigkeit ist und was für eine Rolle der McGurk-Effekt hat. Lippenlesen wird in diesem System durch Bildverarbeitung realisiert. Hierzu wird die Prozesskette beschrieben. In dieser Prozesskette werden aus den beobachteten Kommandos sich zeitlich ändernde Merkmalsvektoren erzeugt, die als Trajektorien bezeichnet werden, diese werden ebenfalls vorgestellt.

2.1 Lippenlesen

Der McGurk-Effekt zeigt unter anderem auf, dass nicht nur ausschließlich die akustische Information einer Äußerung ausreicht, um Kommunikation korrekt wahrzunehmen. Visuelle Komponenten sind unter anderem Bewegungen/Mimik des Mundbereichs. Diese können die Kommunikation stark beeinflussen, so weit dass etwas Falsches wahrgenommen werden kann. McGurk untersuchte die *Bimodalität* der Sprache mit folgendem Versuch: Menschen wurden gefilmt, wie sie jeweils die Laute /ba/ und /ga/ produzierten. Diese Aufnahmen wurden manipuliert, in dem die Audiospur der beidem Filmaufnahmen vertauscht wurden, so wurde /ba/ gehört und sah /ga/ und umgekehrt. McGurk schaute sich die neuen Aufnahmen an und nahm, statt dem /ba/ oder /ga/, /da/ wahr. Aus diesem Versuch wurde deutlich, dass die visuelle Komponente der Kommunikation einen starken Einfluss auf diese hat. Grund dieser falschen Wahrnehmung ist, dass das Gehirn versucht die *bimodalen* Signale zu synchronisieren. Es will die visuellen Informationen verwerten, um Rückschlüsse auf die akustischen Informationen zu schließen. Das Visuelle ist hierbei eine unterstützende Komponente (McGurk und MacDonald [1976]).
Lippenleser sind fähig die Bedeutung einer Äußerung anhand der Anordnung/Struktur und der Bewegung der sichtbaren Artikulatoren herzuleiten. Hierbei dient der Kontext bzw. die herrschende Situation als weiterer Anhaltspunkt in der Bedeutungsfindung. Die Artikulatoren sind die Zunge, die Lippen die Zähne und die Gesichtsmuskeln (Mimik) des Mundes. Durch die Position der Artikulatoren werden Informationen über den Inhalt der akustischen Äußerung geliefert. Lippenlesen wird häufig von hörbeeinträchtigen Menschen für das Sprachverstehen

benutzt.
Neben dem Lippenlesen gibt es andere Wege in der ein Mensch das Gesehene nutzen kann, um die akustische Kommunikation zu unterstützen. Die visuellen Informationen werden beispielsweise genutzt, um bei einem Gespräch nicht den Fokus zu verlieren. Der Mensch kann so störende Hintergrundgeräusche vom Gesprochenen trennen. Außerdem kann das Visuelle unterstützend sein, falls die Person, durch Akzent/Dialekt, undeutlich spricht (Tsuhan [2001]). Lippenlesen kann unterschiedlich schwer sein in Abhängigkeit der Sprache. In einigen Sprachen können nicht genügend visuelle Informationen über die Artikulatoren vermittelt werden. Lippenlesen wird erschwert bei Sprachen, in der Äußerungen viel über den Mundhohlraum und ausschließlich durch die Stimmbänder geschieht. In Sprachen in der die Kommunikation viel über Lippenbewegungen und einer sichtbaren Zunge geschieht, ist das Lippenlesen einfacher. In dieser Arbeit wird das Lippenlesen in englischer Sprache vollzogen, dabei spielen Kontextinformation keine unterstützende Rolle und es werden nur ausschließlich die Lippen betrachten. Die Zunge, die Zähne und die Gesichtsmuskulatur (Mimik) werden in dieser Arbeit für die Wiedererkennung des Gesprochenen nicht betrachtet.

2.1.1 Phonem/Mundbild

Ein Phonem ist die kleinste unterscheidbare Einheit aus Sicht der Akustik bei einer Äußerung und ein Mundbild ist die optische Repräsentation eines Phonems.

Phonem

Bei der Aussprache von Worten entstehen Laute. Diese Laute werden in der Phonologie und in der Phonetik als Phoneme/Phone (Plural) bezeichnet. Ein Phonem (Singular) ist die kleinste diskrete Einheit, welche isoliert in einer ausgesprochenen Wortkette geteilt werden kann. Phoneme sind daher untereinander sprachlich unterscheidbar. Ein Phonem stellt also ein Sprachlaut dar. Phoneme können aufgeteilt werden in Konsonanten und Vokale (Tsuhan [2001]).

Phonem	Wort
/m/	**m**ap
/p/	**p**it
/r/	**r**un
/l/	**l**eft

Tabelle 2.1: Phonem - Wort

In der Tabelle 2.1 sind einige Phoneme aufgelistet. Zu jedem Phonem ist ein Wort abgebildet.

Mundbild

Bei der Bildung eines Phonems entsteht ein Mundbild (engl. viseme). Das Mundbild ist die visuelle Repräsentation der Aussprache und entsteht durch die Bewegung der Lippen und der darum liegenden Mundregion. Ein Mundbild ist also die kleinste unterscheidbare Bewegung, um ein Laut zu verursachen (Hazen [2006]).
Audiosignale von Konsonanten sind weniger unterscheidbarer als Vokale (Yau u. a. [2006]). Konsonanten sind einfacher zu „sehen“ und schwerer zu „hören“ als Vokale. Mundbilder können konkateniert werden, um Wörter zu bilden. Es gibt weniger Mundbilder als Phoneme, d.h. jedes Mundbild kann mehr als ein Phonem zugeordnet werden, da Zunge und Stimmbänder nicht sichtbar bei der Bildung des Mundbildes sind (Yau u. a. [2006]).

Konsonant	Startframe	Mittelframe	Endframe
/v/			
/m/			
/g/			

Tabelle 2.2: Mundbilder
Quelle: Yau u. a. [2006]

In der Tabelle 2.2 sind einige (englische) Mundbilder mit ihren korrespondierenden Phonemen zu sehen.

2.1.2 Lippenlesen aus Sicht der Bildverarbeitung

In diesem Abschnitt wird das Lippenlesen aus der Sicht der Bildverarbeitung (Computer Vision) vorgestellt.

In der Abbildung 2.1 ist die Verarbeitungskette beim Lippenlesen zu sehen. Diese Prozesskette ist abstrakt und zeigt die üblich notwendigen Schritte beim Lippenlesen (Bacivarov u. a. [2008], Yargic und Dogan [2013], Werda u. a. [2006], Alizadeh u. a. [2008], Sujatha und Krishnan [2012], Shin u. a. [2011]). Im ersten Schritt muss der Körper bzw. das Gesicht erfasst werden. Es können Video- oder Echtzeitaufnahmen benutzt werden. Um die Lippen lokalisieren zu können, ist es vorher notwendig den Bildbereich einzuschränken. Dadurch wird die Suche der Lippen beschleunigt. Aus diesem Grund wird vor der Lippenlokalisierung eine Gesichtserkennung bzw. -lokalisierung durchgeführt. Das Gesicht kann beispielsweise durch eine „Region of Interest“ (ROI) ausgewählt werden. Eine *ROI* ist ein Rechteck, wobei mindestens zwei Eckpunkte bekannt sein müssen oder ein Eckpunkt und die Breite und Höhe der *ROI* (Bacivarov u. a. [2008]). Es gibt verschiedene Arten, um die Lippen zu erkennen bzw. zu lokalisieren, jedoch wird hauptsächlich entweder die modellbasierte oder die bildbasierte Lippenerkennung angewendet (Sujatha und M.Radhakrishnan [2013], 2.1.2, 4.3). Bei Notwendigkeit kann die Auflösung des Bilds der lokalisierten Lippen reduziert werden, somit muss weniger verarbeitet werden. Dies hat den Vorteil, da es die Rechenkomplexität senkt.

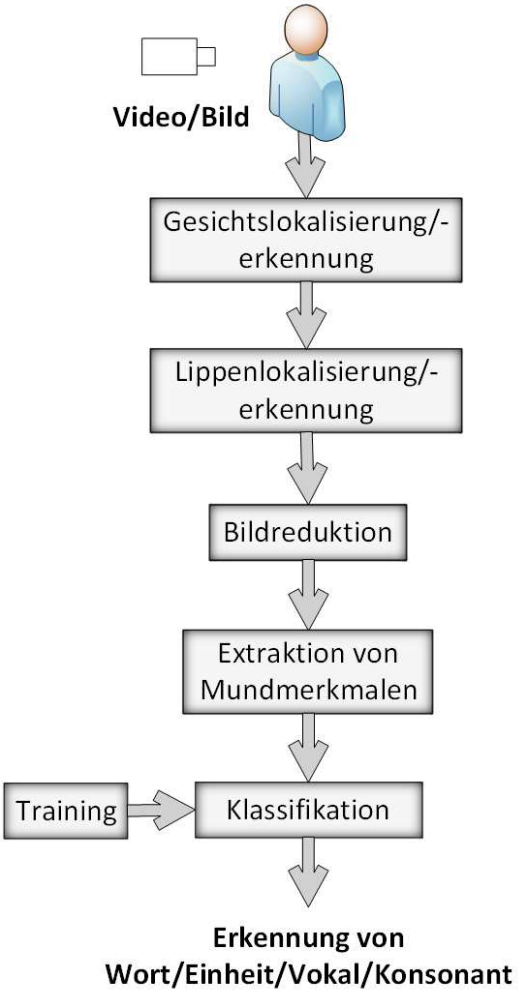


Abbildung 2.1: Prozesskette: Lippenlesen

Nachdem ggf. unbrauchbare Informationen entfernt wurden, werden Merkmale (Features) in der Mundregion bzw. der Lippen extrahiert. Die Merkmale werden benutzt, um die Lippen zu lesen. Die Merkmalsextraktion kann modell- oder bildbasiert sein (2.1.2). $Momente$ können

beispielsweise verwendet werden, um die Bewegungen der Lippen als Merkmale zu repräsentieren (Zernike *Momente* (Yau u. a. [2006])). Schlüssel-Punkte der Lippen (wie die Ecken), die durch bildbasierte Verfahren extrahiert werden, können ebenfalls als Merkmale genutzt werden (Bangsawan u. a. [2015]). Die Merkmale werden einem Klassifizierer übergeben. Der Klassifizierer führt den finalen Schritt zum Lippenlesen aus und erkennt das Gesagte bzw. Gesehene. Der Klassifizierer muss vorher trainiert werden bzw. eine Wissensbasis haben, um die Merkmale beispielsweise auf Mundbilder abzubilden bzw. zu klassifizieren. Ein Mundbild ist die visuelle Repräsentation in der Aussprache von Vokalen und Konsonanten.

Bildbasiertes und modellbasiertes Lippenlesen

Historisch gesehen gibt es zwei grundlegende Ansätze in der visuellen Spracherkennung (Lippenlesen) (Ooi u. a. [2008]). Die erste Methode ist der bildbasierte Ansatz oder intensitätbasierte Ansatz und die zweite Methode ist der modellbasierte Ansatz oder formbasierte (geometrische) Ansatz (Shaikh u. a. [2013]).
Beim bildbasierten Ansatz werden Merkmale direkt von den Grau- und/oder Farbwerten der Mundregion abgeleitet. Die Pixelwerte repräsentieren die Mundregion und sind gleichzeitig ein Merkmalsvektor für die Wiedererkennung. Dies hat den Vorteil, dass ein genaues Verfolgen (Tracking) und die Modellierung der Lippen nicht benötigt werden. Dadurch dass kein statistisches Modell (der Lippen) trainiert werden muss, wird die Berechnungskomplexität des Systems reduziert. Oftmals sind Merkmale bei diesem Ansatz der Mundhohlraum, der Kiefer und die äußere Gesichtsumgebung des Mundes, einschließlich der Lippen. Im Gegensatz zum modellbasierten wird beim bildbasierten Ansatz eine höhere Dimension betrachtet, d.h. jeder Pixel der Mundregion wird für die Erkennung genutzt. Dies ist ein Nachteil für Systeme mit geringer Rechenleistung. Dem kann entgegengewirkt werden, indem eine Dimensionsreduktion angewendet wird. Ein weiterer Nachteil ist, dass dieser Ansatz fehleranfällig sein kann, z.B. bei Veränderungen der Beleuchtung, der Position der Lippen, der Distanz zur Kamera, der Kamerarotation oder in Abhängigkeit von Sprecher.
Beim modellbasierten Ansatz wird die geometrische Form bzw. Kontur des Mundes und der Lippen als Merkmal benutzt und gespeichert. Das Lippenmodell kann durch ein kleines Set von Parametern repräsentiert werden. Beispielsweise kann die Höhe und Breite des Mundes als Parameter dienen. Die Werte können vom binären Bild des Mundes abgeleitet werden. Bei der Benutzung einer Stereokamera können 3-D-Koordinaten, wie beispielsweise Lippenecken und Mittelpunkt von Ober- und Unterlippe, als Parameter bzw. Merkmal genutzt werden. Im Allgemeinen werden bei der modellbasierter Merkmalsextraktion Techniken basierend auf geometrische $Templates$ genutzt. Die Techniken codieren ein Set von Mundform oder Lip-

penkonturen, um die Lippen zu erkennen. Eine Möglichkeit um die Lippen zu modellieren sind sogenannte „Active Shape Models“ (ASM). ASM nutzen *Template*techniken, um Lippenkonturen zu extrahieren. Ein statistisches Konturenmodell wird mit Werten aus dem Bild gefüllt, um so die Lippeninformationen zu erhalten. Vorteil dieser Methode ist, dass sie weniger sensitiv auf Kameraposition/-sicht und Bildstörungen (Lichtverhältnisse) ist. Außerdem basieren sie nur auf der Kontur der Lippen und haben keine Informationen zu anderen Sprechern. Nachteil ist, dass das Modell falsch aufgestellt werden kann, da ein aufwendiges Training des Modells notwendig ist. Hinzu kommt, dass komplexe Algorithmen genutzt werden, um die Markierung der Lippenkonturen zu ermöglichen, dies kann ein Nachteil in einem Echtzeitsystem sein (Shaikh u. a. [2013]).

2.2 Trajektorie

Die extrahierten Merkmale in dieser Arbeit werden durch eine Trajektorie repräsentiert. Eine Trajektorie ist eine Raumkurve (Formel 2.2 und Abb. 2.3). Jeder Punkt der Raumkurve entspricht einem Merkmalsvektor $\overrightarrow{x}(t)$ zu einem gegebenen Zeitpunkt t (Baud u. a. [2007], Yanagisawa und Satoh [2006], D'Urso [2000]):

$$\overrightarrow{x}(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{pmatrix} \tag{2.1}$$

$$f(t) = \overrightarrow{x}(t) \tag{2.2}$$

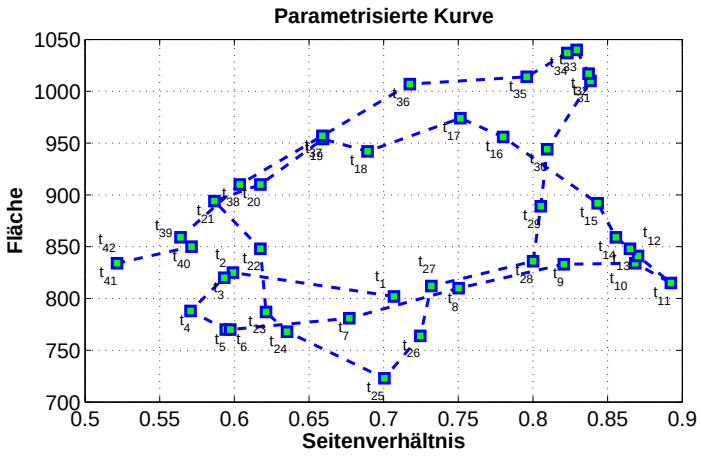


Abbildung 2.2: Parametrisierte Kurve

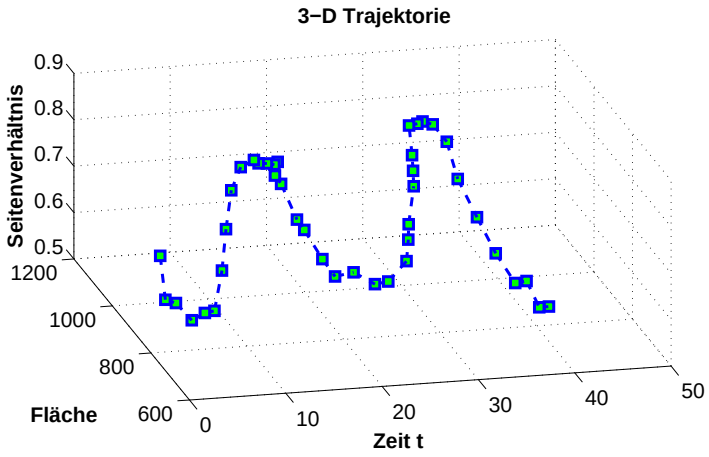


Abbildung 2.3: Trajektorie

Der Merkmalsvektor in Formel 2.1 kann eine Position eines Objektes sein oder auch das Verhalten von Parametern über die Zeit beschreiben. In Abbildung 2.3 ist eine Trajektorie zu sehen, die die Veränderung der Fläche und das Seitenverhältnis des Mundes über die Zeit beschreibt. Die Trajektorie ist entstanden bei der Äußerung eines Kommandos. Eine andere Darstellungsform stellt die parametrisierte Kurve dar (Abb. 2.2). Hier wird die Zeit nicht als Achse aufgezeigt, sondern ggf. an den jeweiligen Parameterpunkten markiert. Abbildung 2.2 ist die parametrisierte Kurve zur Trajaktorie in Abbildung 2.3.

3 Analyse

Das Kapitel befasst sich mit der Analyse des zu entwickelnden visuellen Spracherkennungssystems. Zunächst werden die zwei in 2.1.2 vorgestellten gängigen Vorgehensweisen zu computerbasiertem Lippenlesen vorgestellt. Hierzu werden Vor- und Nachteile und ein abschließendes Fazit aufgezeigt. Anhand einer Machbarkeitsanalyse wird dargestellt, inwieweit ein computerbasiertes Lippenlesen realisiert werden kann. Bei der Entwicklung des Prototyps wird die Implementierung der vorgestellten Prozesskette in 2.1.2 realisiert. Hierzu werden vergleichbare Projekte und ein Fazit zu jedem Prozessschritt vorgestellt. Die Grundlagen des Algorithmus zur Klassifizierung bzw. zur Ähnlichkeitsmessung von Trajektorien wird aufgezeigt. Problemstellungen und Lösungen, die in der Entwicklung des visuellen Spracherkennungssystems auftauchen, werden vorgestellt. Abschließend wird ein Fazit in Form einer Anforderungsanalyse an das System gezogen, in dem zusätzlich ermittelt wird welche Verfahren genutzt werden.

3.1 Modelbasiertes Vorgehen

Wie in 2.1.2 beschrieben wird beim modellbasierten Ansatz ein Modell aufgebaut. In diesem Abschnitt werden die unterschiedlichen Verfahren zur Generierung von geometrischen Formen und Aufstellung von Modellen vorgestellt.

3.1.1 Einfaches Template-matching

In der Arbeit Yingjie u. a. [2011] wird ein System entwickelt, welches anhand der Kontur der Lippen Sprecher identifiziert. Es wird ein einfaches *Template*-matching oder Abgleich von Mustern/*Templates* angewendet. Es werden geometrische Informationen der Lippen genutzt, um das Template aufzustellen. Dabei werden Schlüsselelemente definiert, die für eine erfolgreiche Identifikation, erkannt werden müssen. Die Schlüsselelemente sind definiert durch die Äußerung der Vokale /a/, /o/, /i/, /u/ und der Konsonanten /sh/ und /z/. Es wird von der Annahme ausgegangen, dass diese Schlüsselelemente genügend Unterschiedlichkeiten in der Äußerung haben, um Sprecher zu unterscheiden.

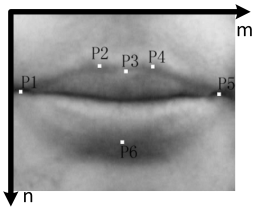


Abbildung 3.1: Schlüsselpunkte
Quelle: Yingjie u. a. [2011]

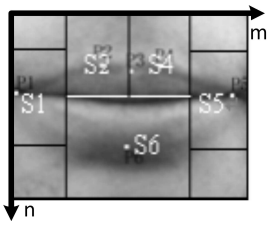


Abbildung 3.2: Regionen um die Schlüsselpunkte
Quelle: Yingjie u. a. [2011]

Für das *Template*-matching werden zunächst im ersten Frame Schlüsselpunkte der Lippen manuell markiert. Diese Schlüsselpunkte spiegeln die Deformation der Lippen am besten (Abb. 3.1) wider. Nun werden die Grauwerte der Regionen um die Schlüsselpunkte in Form eines Fenster der Größe $N * N$ (N freiwählbar) gespeichert. Diese stellen nun die *Templates* dar.

$$D(i,j) = \sum_{m=1}^{N} \sum_{n=1}^{N} |S^{i,j}(m,n) - T(m,n)| \tag{3.1}$$

Beim Abgleich bzw. Matching wird das Bild in Bereiche der Schlüsselpunkte aufgeteilt (Abb. 3.2). $S^{i,j}(m,n)$ ist der Grauwert des Bildpunktes an der Bildausschnittskoordinate (m,n) desjenigen Fensters mit der Ursprungskoordinate (i,j). Nur in den jeweiligen Bereichen wird das jeweilige *Template* fürs Matching genutzt. $T(m,n)$ ist der Grauwert des Bildpunktes an Templatekoordinate (m,n). Es erfolgt ein einfaches Pixelmatching bzw. ein Abgleich von Grauwerten. Die Gleichung 3.1 beschreibt das Matching. i und j sind die Koordinaten des Fensters. Es wird die absolute Differenz vom aktuellen Frame und dem *Template* berechnet. Der Pixelgrauwert gemeinsam mit seinem Fenster, dessen Differenz am geringsten ist, ist der gesuchte Schlüsselpunkt.

Vor- und Nachteile

Vorteil dieses Verfahren ist die niedrige Komplexität in der Implementierung. Nachteil ist, dass die Schlüsselpunkte manuell gesetzt werden müssen. Weiterer Nachteil ist, dass ein einfaches Pixelmatching fehleranfällig ist. Bei unterschiedlichen Lichtverhältnissen oder weiteren Störungen im Bild kann kein erfolgreiches Matching geschehen. Die Fenstergröße kann nur manuell ausgewählt werden, somit ist das Verfahren skalierungsempfindlich. Viele Tests sind notwendig um eine optimale Fenstergröße auszuwählen.

3.1.2 Active Contour Model/Snakes

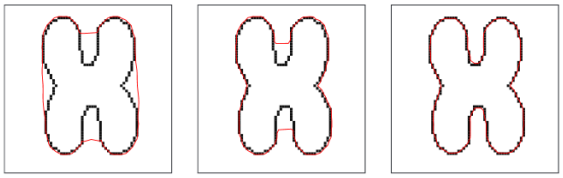


Abbildung 3.3: Iterative Entwicklung eines Snakes
Quelle: Cartas-Ayala [2002]

Ein weiteres modellbasiertes Verfahren sind die sogenannten „Snakes“ bzw. das „Active Contour Model“ (ACM) (Kass u. a. [1988]) . ACM wird in der Bildverarbeitung eingesetzt, um Konturen und Kanten zur Laufzeit zu erkennen. Hierbei wird eine parametrisierte Kurve um die Grenzen eines Objekts oder anderen hervorstechenden Bildmerkmalen gelegt. Die Kurve versucht nun die Kontur zu finden und passt bzw. legt sich an die Kontur an. Eine Energiefunktion E repräsentiert die Kurve. Ziel von ACM ist es, die Energiefunktion zu korrigieren in dem sie minimiert wird. Gleichzeitig erfolgt dadurch das Anpassen der Kurve an der Kontur des Objekts. Die Kontur wird in diesem Zusammenhang „Snake“ genannt.

$$E_{Snake} = E_{intern} + E_{extern} \tag{3.2}$$

Die Energie beschreibt wie die spezifischen Parameter und die Snake und wie die aktuellen Bildinformationen und die Snake zusammenpassen. Die interne Energie E_{intern} (Gleichung 3.2) beschreibt die vom Bild unabhängigen spezifischen Parameter der Snake. Sie passt die Elastizität, Glätte und die Krümmungen der Kurve/Kontur an. Die externe Energie E_{extern} beschreibt die aktuelle Bildinformation. Sie verursacht das Anpassen an existierenden Strukturen des Bildes (rote Kontur ist die Snake in Abb. 3.3). Eine geringe Energie sagt aus, dass die Übereinstimmung von Snake zur Kontur hoch ist, d.h. sie befindet sich an einer Kante. Das Ziel ist es eine Snake mit minimaler Energie zu finden.

Vor- und Nachteile

Vorteil dieser Methode ist, die Beschreibung von Konturen durch eine einzelne Funktion. Außerdem werden die Konturen dynamisch generiert. Durch den internen Energiefaktor (Gleichung 3.2) können sich Snakes flexibel entwickeln, jedoch kann sich dieser Punkt zum Nachteil entwickeln.

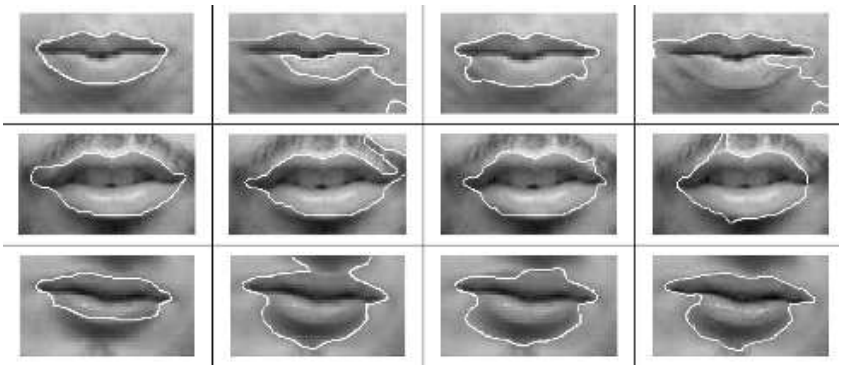


Abbildung 3.4: Fehlerhafte Snakes
Quelle: Chin u. a. [2012]

Es ist notwendig die korrekten Parameter zu finden, um eine auf das Objekt passende Kontur/Snake zu finden. Durch verschiedene Lichtverhältnisse können sich die Snakes verschieden entwickeln. So müssen ständig die Parameter angepasst werden (Abb. 3.4). Ein weiterer Nachteil ist, dass ein initialisierter Snake benötigt wird, welches um das Objekt manuell gelegt werden muss. Problematisch wird ebenfalls die iterative Findung der Snakes bei schnell dynamischen Bewegungen, wie beispielsweise bei Lippenbewegung durch eine Äußerung (Sui u. a. [2013], Chin u. a. [2012]).

Active Shape Model

Abbildung 3.5: ASM: Iteratives Anpassungsverhalten
Quelle: Cootes u. a. [2000]

Das „Active Shape Model“ (ASM) ist eine Erweiterung des ACM (Cootes und Taylor [1992]). Ziel von ACM ist es mithilfe eines statistischen Modells eine Kontur eines Objektes zu beschreiben/-finden. Ein statistisches Modell sagt aus, welche Werte bestimmte Daten annehmen können. Für die Zuordnung werden Wahrscheinlichkeiten genutzt. Beim ASM wird das statistische Modell durch einen Verbund von Koordinatenpunkten im Bild repräsentiert. Die Kontur eines Objekts wird von diesem Verbund von Punkten bestimmt. Es wird hierfür ein Punktverteilungsmodell

(„point distribution model“) genutzt, welches Informationen, wie Abstände der Punkte zueinander beinhaltet. Das Modell wird generiert durch Trainingsbilder. Eine Durchschnittskontur bzw. ein Durchschnittsumriss wird von diesen Trainingsbildern generiert. Hierzu müssen Punkte auf den Trainingsbildern per Hand markiert/gelegt werden. Die Konturen werden durch einen iterativen Prozess gefunden, in dem der Durchschnittsumriss (das Modell) zu den Konturen des Objektes angenähert werden. Ein Verfahren für die Annäherung ist, dass eine euklidische Distanz minimiert wird. Die Distanz ist der Abstand der Kontur des Durchschnittsumrisses und dem tatsächlichen (Bild) Objekt.

Vor- und Nachteil

Vorteil dieser Methode ist, dass sie echtzeitfähig ist, da das Punktverteilungsmodell sich iterativ an die Veränderung anpasst. Durch die Nutzung von vordefinierten Trainingsbildern ist die Kantensuche robuster als beispielsweise das Verfahren des einfachen *Template*-matchings (3.1.1) oder ACM (3.1.2). Das Verfahren und das Modell sind voneinander getrennt, sodass das Trainingsset wiederverwendet werden kann. Nachteil des Verfahrens ist der hohe Trainingsaufwand und die manuelle Setzung von Punkten. Ein großes Set von Trainingsdaten ist notwendig für eine robuste Kantenverfolgung. Verdeckungen und Mehrdeutigkeit von Konturen werden nicht behandelt. Durch sein iteratives Anpassen (Abb. 3.5) ist ASM fehleranfällig für schnelle, dynamische Bewegungen (wie bei Lippenbewegungen in einer Äußerung) (Cootes u. a. [2000]).

3.1.3 Active Appearance Model

Abbildung 3.6: AAM: Iteratives Anpassungsverhalten
Quelle: Cootes u. a. [2001]

Das „Active Appearance Model“ (AAM) ist eine Erweiterung bzw. eine Weiterentwicklung des ACM und des ASM (Cootes u. a. [2001]). In diesem Verfahren wird, genau wie bei ASM, ein statistisches Modell erstellt. Das Modell basiert zusätzlich, neben der Kontur bzw. des Umrisses des Objekts, auf weitere Texturinformationen (bildbasiert) des Bildes. ASM nutzt nur das Modell des Umrisses des Objekts. Die Variationen der Grauwerte werden jedoch in diesem

Modell mit berücksichtigt. Genau wie bei ASM, müssen Markierungen gesetzt werden, diese Markierung erhalten eine weitere Information, den Faktor für die Variation des Grauwertes. Das Modell wird auf das zu verfolgende Objekt gelegt und durch Iterationen wird das Modell aktualisiert, in dem die Unterschiedlichkeiten von Bild und Modell gemessen werden.

Vor- und Nachteile

Vorteil des Modells ist die Echtzeitfähigkeit und das robuste Verfolgen (Tracking) von Objekten. Durch die zusätzlichen Informationen der Grauwerte steigt die Robustheit gegenüber dem Modell von ASM. Nachteil des Modells ist der hohe Aufwand für das Training. Es ist erforderlich eine große Anzahl von Trainingsdaten zur Verfügung zu stellen, damit ein robustes Tracking möglich ist. Durch das Konturenmodell des ASM und die Erweiterung von AAM bzgl. des Modells der Grauwerte wird eine hohe Rechenleistung benötigt, damit eine Echtzeitfähigkeit garantiert wird (Gao u. a. [2010]).

Konsequenzen für die weitere Vorgehensweise

In diesem Abschnitt wird beurteilt, ob die präsentierten Verfahren für das visuelle Spracherkennungssystem in dieser Arbeit geeignet sind. Das einfache *Template*-matching erweist sich als zu ungenau, da die *Templates* durch ein störanfälliges Verfahren ausgewählt werden. Die *Templates* sind fehleranfällig bei Kontraststörungen durch unterschiedliche Lichtverhältnisse und anderen bildbasierten Störungen. Bereits vollzogene Tests haben dies bestätigt. Das manuelle Setzen der Schlüsselpunkte passt nicht zur Anforderung nach einem automatischen visuellen Spracherkennungssystem.
Das „Active Contour Model“ erzeugt ein auf den ersten Blick brauchbares dynamisches Modell, jedoch müssen, durch die Energiefunktion, zu viele Parameter gesetzt werden. Außerdem entstehen fehlerhafte Konturen bei verschiedenen Lichtverhältnissen (durch Schatten, starke Sonneneinstrahlung) (Abb. 3.4), sodass ständig Parameter manuell gesetzt werden müssen. Die Lippenbewegungen haben eine hohe Veränderungsrate. Durch das iterative Suchen und das manuelle Anlegen der Initialkontur wird ein genaues Verfolgen (Tracking) fehlerhaft.

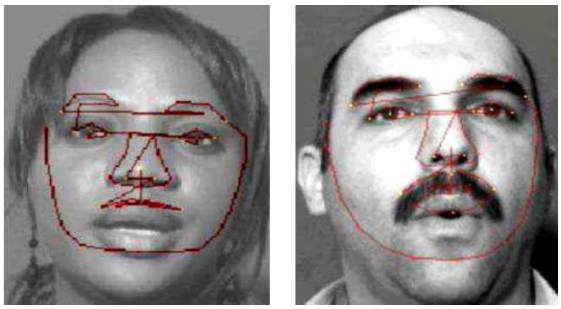


Abbildung 3.7: Fehlerhaftes Verhalten von ASM
Quelle: ur Rehman Butt und Lombardi [2013]

Beim „Active Shape Model“ werden Vorinformationen durch ein tatsächliches Modell genutzt. Dadurch wird ein robustes Verfolgen von Objektkonturen erreicht. Jedoch ist die Komplexität der Entwicklung hoch. Nach ur Rehman Butt und Lombardi [2013] erfolgt kein zuverlässiges Verfolgen der Lippen (Abb. 3.7 links) und es entstehen Probleme wenn die Person einen (Schnurr) Bart (Abb. 3.7 rechts) hat. Laut Yau u. a. [2006] entstehen Fehler bei der Verfolgung (Tracking). Eines der Ziele dieser Arbeit ist es, ein System mit geringer Rechenleistung zu entwickeln, da das System auf einem Roboter mit niedrigen Ressourcen laufen wird. Durch das iterative Verhalten des Modells werden fortlaufende Korrekturen durchgeführt. Daher ist das Verfahren mit geringer Rechenleistung nicht durchführbar.
Das „Active Appearance Model“ stellt das robusteste Verfahren in Bezug auf die Korrektheit des Modells und des Verfolgens (Tracking) dar. Ein zuverlässiges Modell wird jedoch nur dann erstellt, wenn eine große Menge von Trainingsdaten vorhanden ist. Außerdem ist für ein zuverlässiges Verfolgen eine hohe Rechenleistung notwendig, denn es werden zwei Modelle (Kontur- und Grauwertmodell) verwendet. Die Komplexität in der Entwicklung einer robusten und zuverlässigen AAM-Anwendung ist ebenfalls hoch. Aus den vorher genannten Gründen wird von der Verwendung dieses Verfahrens abgesehen.

3.2 Bildbasiertes Vorgehen

Im Abschnitt 2.1.2 wurde dargestellt, dass beim bildbasierten Ansatz Merkmale ausschließlich aus den Grau- und/oder Farbwerten des Bildes abgeleitet werden. Ein Modell für die Merkmalsextraktion, wie beim modellbasierten Vorgehen wird nicht aufgestellt. Einige Verfahren werden vorgestellt und Konsequenzen für das weitere Vorgehen gezogen.
Es existieren vielfältige bildbasierte Ansätze, um Merkmale zu extrahieren, da jedes Anwendungsgebiet verschiedene Anforderungen hat. Außerdem bieten bildbasierte Ansätze keine

Komplettlösungen an, d.h. für verschiedene Prozessketten werden verschiedene/andere Ansätze/Methoden verwendet. Beim modellbasierten Ansatz gibt es die klassischen Vorgehensweisen (siehe Abschnitt 3.1). In diesem Abschnitt werden nur ein Bruchteil und relevante Ansätze vorgestellt.

3.2.1 Momente

Für die Mustererkennung können *Momente* (Hu [1962]) eingesetzt werden. *Momente* sind koordinaten-gewichtete Mittelwerte aus den Grau- und/oder Farbwerten von Bildern. Momente können das Bild anhand von verschiedenen Eigenschaften beschreiben. Mithilfe von geometrischen *Momenten* können die Fläche und der Schwerpunkt eines Bildes beschrieben werden. Bei der Fläche handelt es sich um die Summe der Pixelwerte.

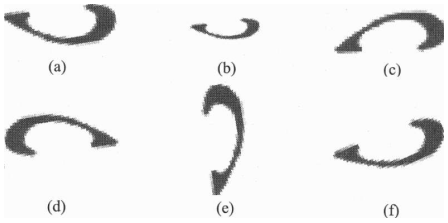


Abbildung 3.8: Objekt
Quelle: Theodoridis und Koutroumbas [2008]

Moments	(a) 0°	(b) Scaled	(c) 180°	(f) 15°	(d) Mirror	(e) 90°
ϕ_1	93.13	91.76	93.13	94.28	93.13	93.13
ϕ_2	58.13	56.60	58.13	58.59	58.13	58.13
ϕ_3	26.70	25.06	26.70	27.00	26.70	26.70
ϕ_4	15.92	14.78	15.92	15.83	15.92	15.92
ϕ_5	3.24	2.80	3.24	3.22	3.24	3.24
ϕ_6	10.70	9.71	10.70	10.57	10.70	10.70
ϕ_7	0.53	0.46	0.53	0.56	−0.53	0.53

Abbildung 3.9: Hu-Momente der Objekte
Quelle: Theodoridis und Koutroumbas [2008]

Die Hu-*Momente* haben verschiedene Invarianzen. Invarianz ist: „Unveränderlichkeit bestimmter mathematischer oder physikalischer Größen gegenüber Koordinatentransformationen“ (wissen.de [2016]). In der Abbildung 3.8 ist ein Objekt in verschiedenen Positionen, Skalierungen und Ausrichtungen zu sehen. Neben dem Bild sind die entsprechenden Hu-*Momente* (Tabelle 3.9) zu sehen. Es kann festgestellt werden, dass die *Momente* nur geringe Abweichungen haben, obwohl die Objekte unterschiedliche Positionen und Größen haben. Weiterentwicklungen der geometrischen *Momente* sind:

- Zentral*moment*: translationsinvariant: Gleiche *Momente*, auch wenn das Bild verschoben wird.
- normalisiertes Zentral*moment*: skalierungsinvariant: Gleiche *Momente*, auch wenn das Bild skaliert (vergrößert und verkleinert) wird.
- Hu-*Momente* (Hu [1962]): rotationsinvariant: Gleiche *Momente*, auch wenn das Bild rotiert (gedreht) wird.

Vor- und Nachteile

Ein Vorteil der Extraktion von Merkmalen durch *Momente* ist, dass durch eine bestimmte Anzahl von Werten (den *Momenten*) ein Bild oder Objekt beschrieben werden kann. Es ist nicht notwendig alle Pixel zu speichern. Weiterer Vorteil ist, dass die Momente invariant zu bestimmten Eigenschaften sind. Dies kann die Wiedererkennung von Objekten erleichtern, da nicht zusätzliche Verfahren notwendig sind, um Invarianzen zu ermöglichen. Ein Nachteil dieser Methode kann sein, dass bei dynamischen Bewegungen (beispielsweise Lippenbewegungen) durch die ständige Berechnung der *Momente* eine hohe Rechenleistung benötigt wird. Die *Momente* sind nicht beleuchtungs- und kontrastinvariant (bei unterschiedlichen Lichtverhältnissen).

3.2.2 Pixelcharakteristika

Eine weitere mögliche Option ist die Betrachtung von charakteristischen Grau- und/oder Farbwerten. Diese Werte können gespeichert und im späteren Prozessverlauf für die Wiedererkennung eingesetzt werden. Es ist ebenfalls möglich die charakteristischen Pixelwerte indirekt für die Wiedererkennung zu nutzen. Dabei werden Vorinformationen des Objekts genutzt und die Pixelwerte innerhalb eines Bildes gesucht (siehe Abschnitt 3.5.3 und 4.4.2). Das Anwendungsgebiet bestimmt wie, welche Pixelwerte und ob Grauwerte und/oder Farbwerte extrahiert werden.
In Chowdhury u. a. [2015] werden Farbwerte von Straßenrändern analysiert, um die Anteile von Erdboden, Gras , Straße und Bäumen zu detektieren. Diese Analyse ist eine wichtige Aufgabe für die Identifizierung von Regionen mit Brandgefahr. Um Typen von Objekten zu identifizieren werden verschiedene Farbinformationen genutzt. Es wird die Vorinformation genutzt, dass Pflanzenwuchs bzw. Gras folgende Farben enthalten: grün, rot, orange, braun und gelb. Ein Mensch kann zusätzlich das Lichtverhältnis einschätzen und aussagen, ob es sich um Gras handelt. In der Bildverarbeitung können unterschiedliche Lichtverhältnisse ein extremer Störfaktor sein. In der Arbeit wird deshalb erwähnt, dass es verschiedene Farbmodelle für verschiedene Anforderungen gibt. Statt dem üblich bekannten RGB-Modell (Rot, Grün, Blau) kann beispielsweise das HSV-Modell genutzt werden. „H“ steht für den Farbwert (hue), „S“ steht für die Farbsättigung (saturation) und „V“ steht für den Helligkeitswert (value). So kann der V-Wert ignoriert werden. Die korrekte Auswahl des Farbmodells ist ein entscheidender Faktor für erfolgreiche Klassifikation der Objekte.

Statistics Parameter	Average			
	|R-G|	*|R-B|*	*|G-B|*	*(R+G+B)/3*
Soil-brown	22	71	48	85
Green Grass	20	17	30	85
Road	3	21	23	85
Tree	9	6	10	65

Abbildung 3.10: Durchschnitt Pixelwerte je Klasse
Quelle: Chowdhury u. a. [2015]

In der Arbeit wurde jedoch das RGB-Modell genutzt, da sie durch erste Tests herausgefunden haben, dass das Modell die höchste Erkennungsrate liefert. Aus dem RGB-Modell definieren sie sich vier Merkmale (Tabelle 3.10). Es werden verschiedene Klassen für die Wiedererkennung bestimmt. Die Klassen spiegeln die möglichen Objekte (Gras , Baum, Straße, Erde) wider. Für jede dieser Klassen bestimmen sie aus Trainingsbilder Farb-Durchschnitte für die jeweiligen Merkmale. Hierfür wird eine *ROI* genutzt, die die Regionen des Bildes analysiert in dem die Merkmale extrahiert und zugeordnet werden.

Vor- und Nachteile

Vorteil dieser Methode ist, dass sie Vorinformationen von Farbwerten nutzen. Sie bestimmen aus ihrem Anwendungsfall heraus, welche charakteristische Farbwerte die jeweiligen Objekte besitzen. Nachteil dieser Methode ist, dass je Bild viele Merkmalsvektoren entstehen. Es wird nur ein Standbild des Straßenrandes analysiert, d.h. würden mehrere Frames analysiert werden, würde eine größere Anzahl von Merkmalsvektoren entstehen. Es ist außerdem nicht aus den Merkmalen ersichtlich, ob Regionen zusammengehören oder welche Regionen ein Objekt darstellen. Es werden nur Regionen des Bildes einer Klasse zugeordnet.

3.2.3 Extraktion von Konturen

Je nach Anwendungsgebiet ist es notwendig, durch bildbasierte Methoden Konturen zu extrahieren. Die Konturen können als Merkmalvektor für eine Wiedererkennung genutzt werden. Im Gegensatz zu den bereits vorgestellten bildbasierten Methoden wird die Wiedererkennung nicht durch zusammenhängende Werte (*Momenten* oder Grau- und/oder Farbwerten) realisiert, sondern es werden charakteristische Konturpunkte im Bild gesucht. Bevor die Konturen untersucht werden können, ist es notwendig die Konturen aus dem Bild zu extrahieren.

Konsequenzen für die weitere Vorgehensweise

Die vorgestellten Verfahren in diesem Abschnitt werden beurteilt, ob sie für das visuelle Spracherkennungssystem in dieser Arbeit geeignet sind. *Momente* können Objekte zuverlässig beschreiben, außerdem bieten sie Invarianzen an. Problem des Verfahrens ist, dass pro Bild/Frame viele *Momente*/Werte entstehen. Je mehr Invarianz eingefordert wird, desto mehr *Momente* sind zu speichern. Für ein visuelles Spracherkennungssystem ist das Verfahren nicht geeignet, da bei einer Äußerung viele Bilder entstehen, somit viele *Momente*. Es müssten viele Parameter miteinander verglichen werden, somit wird ein großer Merkmalsvektor benötigt.
Die Verwendung von charakteristischen Grau- und/oder Farbwerten bietet sich in der Erkennung der Lippen an. Die Farbe der Lippen unterscheidet sich von der Farbe der Haut im Gesicht. So ist ein Verfahren notwendig, welches diese Unterschiedlichkeit extrahiert. In Abschnitt 4.4.3 wird ein passendes Verfahren vorgestellt. Das Verfahren nutzt ein anderes Farbmodell (nicht RGB), denn die Sättigung und Helligkeit haben für gewisse Merkmale der Lippen eine spezifische Eigenschaft, welche in dem Abschnitt präsentiert werden.
Das Extrahieren von Konturen durch bildbasierte Methoden ist notwendig, um Merkmale aus den Lippen zu bestimmen. Dieses Verfahren ist geeignet für die Lippen, da die Lippen eine charakteristische Kontur besitzt. Die Form bzw. die Anatomie der Lippen ist bei jedem Menschen nahezu identisch. Im Abschnitt 4.4.1 wird auf diese Charakteristika eingegangen.
Wie aus den bisherigen Ausführungen ersichtlich, wird das bildbasierte dem modellbasierten Vorgehen vorgezogen, da die Generierung eines Modells eine hohe Rechenleistung und ein großes Set von Trainingsdaten benötigt. Außerdem spielt ebenfalls die Komplexität der Realisierung einer robusten und zuverlässigen modellbasierten Anwendung für die Wahl eine Rolle.

3.3 Machbarkeit eines visuellen Spracherkennungssystems

Es werden zunächst verschiedene Typen von Äußerungen vorgestellt, die üblicherweise bei der Entwicklung eines akustischen Spracherkennungssystems betrachtet werden. Anhand der Typen erfolgt eine Analyse über die Machbarkeit, indem entschieden wird welcher Typ von Äußerung für die Entwicklung am realistischsten ist.
Üblicherweise wird die Erkennung von akustischer Sprache in folgende Typen eingeteilt (Anusuya und Katti [2010]):

- **Isolierte Wörter** Hierbei werden einzelne Wörter erkannt, dabei muss die Umgebung still sein, es darf keine Störgeräusche geben. Je Wort gibt es eine „Zu hören/nicht zu hören“ Phase, d.h. zwischen gesprochenen Wörtern muss es eine Pause geben.

- **Verbundene Wörter** Ähnlich wie bei „isolierte Wörter“ werden Wörter, mit einer Pause zwischen den Wörtern erkannt. Einziger Unterschied ist, dass mehrere Wörter gesagt werden können.
- **kontinuierliche Sprache** Bei kontinuierlicher Sprache kann nahezu natürlich gesprochen werden. Hierbei bestimmt ein Computer(-programm) den Inhalt des Gesagten.
- **spontane Sprache** Hierbei soll die Sprache, die natürlich und nicht eingespielt wirkt, erkannt werden. Ein automatisches as soll dabei Laute, wie „ehm“ und „ah“, die bei natürlicher Sprache entstehen, in der Erkennung mit einbeziehen.

Die Forschung bzgl. der akustischen Spracherkennung ist weit fortgeschritten, sodass sie kontinuierliche, sogar spontane Sprache zuverlässig und robust wiedererkennen kann (Hauswald u. a. [2015], siehe Apple Siri[1], Google Now[2], Microsoft Cortana[3]). Die Forschung bzgl. der visuellen Spracherkennung ist nicht so weit fortgeschritten, wie der akustischen Spracherkennung. Diese Arbeit hatte zu Anfang das Ziel kontinuierliche Sprache anhand von Mundbildern zu erkennen. Vorbild dieses Vorgehen waren die Arbeiten Yau u. a. [2006] und Shaikh u. a. [2013]. In den Arbeiten werden Konsonanten in Mundbildern erkannt. Hintergrund hierbei ist, dass Konsonanten in Audiosignalen mehrdeutiger bzw. schwer erkennbarer sind als Vokale. Die Gesichtsbewegung in Videodaten werden durch sogenannte „Motion History Images“ (MHI) repräsentiert. MHI ist ein *Template*, welches speichert wo und wann Bewegungen in einer

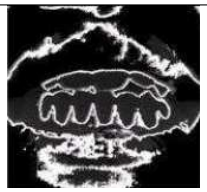
Konsonant	Startframe	Mittelframe	Endframe	MHI
/g/				

Tabelle 3.1: /g/ - Mundbild mit MHI
Quelle: Yau u. a. [2006]

Bildsequenz auftauchen. MHI werden generiert durch „difference of frames“ (DOF). DOF entstehen indem Grauwerte von aufeinanderfolgenden Frames voneinander subtrahiert werden und somit die Unterschiede von einem Frame zum nächsten berechnet werden (Tabelle 3.1). Aus den MHI werden *Momente* als Merkmale extrahiert, um die Konsonanten wiederzuerkennen. In der Arbeit werden nur einzelne Elemente (Mundbilder) isoliert voneinander

[1] http://www.apple.com/de/ios/siri
[2] https://www.google.com/landing/now/
[3] http://windows.microsoft.com/de-de/windows-10/getstarted-what-is-cortana

Mundbild	Phonem	Beispiel
1	/p/, /b/, /m/	put, bed, me
2	/f/, /v/	far, voice
3	/th/, /dh/	think, that
4	/t/, /d/	tick, door
5	/k/, /g/	kick, gate
6	/sh/, /j/, /ch/	she, join, chair
7	/s/, /z/	sick, zeal
8	/n/, /l/	new, lew
9	/r/	rest

Tabelle 3.2: MPEG-4 Standard - Mundbild-Modell - Englische Konsonanten
Quelle: Yau u. a. [2006]

wiedererkannt. Laut Yu u. a. [2009] liefert die Nutzung von Mundbildern bei kontinuierlicher Lippenbewegung begrenzt visuelle Informationen. Es gibt wenig Forschungsergebnisse bzgl. der Identifikation von unterscheidbaren visuellen Sprachelementen, welche den Sprachprozess in der kontinuierlichen Sprache modelliert (Yu u. a. [2009]). In der Theorie können die Mundbilder kombiniert/konkateniert werden, um Wörter zu bilden. In der Praxis jedoch haben einige Mundbilder gleiche Merkmale (Überlappungen) oder Mundbilder werden verfälscht durch vorherige Mundbilder. So wird der Übergang von Mundbild zu Mundbild nicht betrachtet. Außerdem gibt es weniger Mundbilder als Phoneme (siehe 2.1.1 und 3.2). Sie decken nur einen kleinen Subraum von Mundbewegungen. In Tabelle 3.2 ist der MPEG-4 Standard zu sehen, er definiert 9 Mundbilder zu den passenden Phonemen. Aus den genannten Gründen ist ein kontinuierliches visuelles Spracherkennungssystem basierend auf Mundbildern nicht realisierbar.
Aus dieser Analyse wird folgende Schlussfolgerung gezogen. Es werden keine Einheiten/-Mundbilder versucht zu erkennen und daraus Wörter zu bilden. Verbundene Wörter sollen erkannt werden. Ein sogenannter holistischer Ansatz (Sujatha und Krishnan [2012]) wird angewendet. Holistisch bedeutet, dass Etwas als Ganzes betrachtet wird. In diesem Fall werden die Wörter nicht in ihre Einheiten eingeteilt und betrachtet, sondern das Wort als Ganzes soll wiedererkannt werden. Dieses Verfahren wird auch von Hassanat [2014] empfohlen, wenn ein Anwendungsgebiet bekannt ist und die Wörter beschränkt werden können. In dieser Arbeit werden Befehle an einen Assistenzroboter gegeben, so ist ein Anwendungsgebiet gegeben und eine beschränkte Anzahl von Kommandos kann aufgestellt werden.

3.4 Vergleichbare Projekte

Schwierigkeiten und Konsequenzen bzgl. der eigenen Arbeit werden gezogen. Es werden ebenfalls zu den jeweiligen Prozessschritten der vorgestellten Prozesskette in Abschnitt 2.1.2 Methoden/Verfahren vorgestellt. Hierbei geht es explizit um das Lippenlesen. In einem Fazit wird aufgezeigt welches Verfahren bzw. welcher Ansatz für die eigene Arbeit ausgewählt wurde.

Detektion von Lippenpunkten durch Tiefeninformationen

Ziel

Ziel der Arbeit „A Lip Reading Application on MS Kinect Camera“ (Yargic und Dogan [2013]) ist es isolierte türkische Wörter zu erkennen. In der Arbeit werden vordefinierte Lippenpunkte durch Tiefeninformation lokalisiert. Die Winkel zwischen den Lippenpunkten werden als Merkmale genutzt. Für die Klassifikation der Wörter wird ein k-NN (k-Nearest-Neighbor) Klassifikator genutzt. In k-NN wird ein Objekt zu einer Klasse zugeordnet, wenn das Objekt am häufigsten zu den nächsten k zu geordnet wird. Ein k ist ein Vertreter/Repräsentant einer Klasse (Altman [1992]).

Randbedingungen

Die Arbeit nutzt die Tiefeninformation, um Punkte auf dem Gesicht bzw. in der Mundregion abzubilden. Die Methode ist unabhängig zur Kopfposition zur Kamera, da die Winkel anhand von 3D-Positionen berechnet werden. Es ist nicht notwendig den Kopf frontal zur Kamera zu halten. Diese Methodik funktioniert nur mit einer Tiefenkamera oder durch Berechnung der Tiefenpunkte, welche jedoch zusätzliche Rechenleistung benötigt. Die Tiefenkamera bietet jedoch eine Software mit dem es möglich ist ein Modell des Gesichts aufzubauen. Die Lippenaktivität wird mit wenig Rechenaufwand, anhand von aktiven und passiven Bildframes bestimmt.

Vorgehensweise

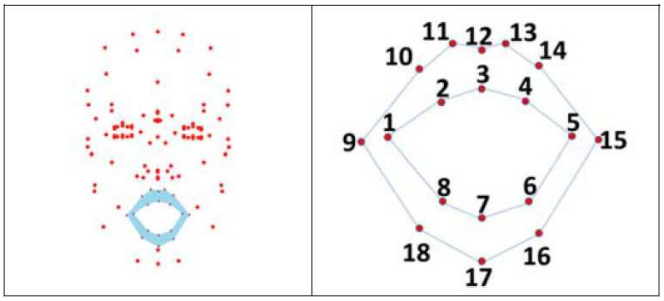


Abbildung 3.11: 18 Lippenpunkte und deren IDs
Quelle: Yargic und Dogan [2013]

1. Die verwendete Kamera bietet eine Face-Tracking-Engine. Die Engine analysiert das eintreffende Bild, um 3-D-Positionen des Kopfes zu berechnen. Hierbei werden 121 vordefinierte Gesichtspunkte im Gesichtsmodel lokalisiert. Die Distanzen zu den Punkten werden auch geliefert. In der Arbeit werden von 121 Punkten 18 Punkte zur Lippenrepräsentation genutzt. Jeder der Punkte hat eine ID (Abb. 3.11).

2. Beim Lippenlesen ist es erforderlich den Anfang- und Endpunkt eines Wortes zu bestimmen. Dies erhöht die Erfolgsrate bei der Klassifikation. Der Begriff „word segmentation“ beschreibt diesen Prozess. Während der Aussprache eines Wortes kann die Lippenaktivität durch eine Abweichung eines spezifischen Lippenmerkmals, während der letzten k Bildframes, entdeckt werden. In der Arbeit werden zwei Winkel benutzt, um die Lippenaktivität zu bestimmen. Diese Winkel beschreiben wie weit der Mund offen ist. Durch einen Schwellwert wird bewertet, ob die Abweichung den entsprechenden Frame als aktiv oder passiv sieht. Hierbei wird die Lippenaktivität bestimmt. Es wird ein einfaches Muster genutzt, um zu bestimmen, wann ein Wort anfängt oder endet. Bei zehn aufeinanderfolgenden Frames mit aktiven oder passiven Frames wird jeweils festgelegt, dass es sich um den Anfang bzw. das Ende des Wortes handelt.

3. Mit dem k-NN-Klassifikator werden die Input-Daten genommen, um die Distanz zum nächsten Nachbarn zu bestimmen. Jedes isolierte Wort ist entsprechend der euklidischen Distanz einem nahen Nachbarn zugeordnet. Durch die Reduzierung der Winkel erfolgte eine erhebliche Reduzierung der Rechenzeit.

Hybrides Verfahren zur Lippenextraktion

Ziel

Ziel der Arbeit „Lip Feature Extraction and Reduction for HMM-Based Visual Seech Recognition“ (Alizadeh u. a. [2008]) ist es Merkmale der Lippen zu extrahieren, dabei werden bildbasierte und modellbasierte Ansätze (hybrider Ansatz) genutzt. In der Arbeit werden verschiedene Schlüsselpunkte der Lippen extrahiert. Sie verwenden einen modellbasierten Ansatz, um die Lippen zu repräsentieren. Es werden die Kanteninformationen und Schlüsselpunktpositionen genutzt für die Konturextraktion.

Randbedingungen

Da in der Arbeit ein Modell für die Lippen genutzt wird, ist es notwendig das Modell durch Trainingsbilder zu generieren.

Vorgehensweise

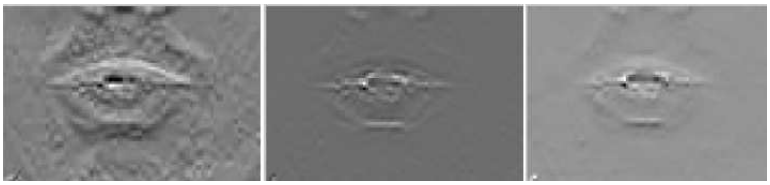


Abbildung 3.12: Kombination von Pseudo-Hue- und Beleuchtungsbilder
Quelle: Alizadeh u. a. [2008]

1. Für die Extraktion der Lippen wird die Transformation „Pseudo-Hue“ (P-H) genutzt. Hierbei erfolgt eine Transformation bzw. Abwandlung des RGB-Farbraums. Das P-H wird wie folgt berechnet:

$$h(x,y) = \frac{R(x,y)}{G(x,y) + R(x,y)} \tag{3.3}$$

   Dabei stehen (x, y) für die Position des Farbwertes. Es wird P-H genutzt, da die Lippen in diesem Farbmodell heller sind als die Haut des Gesichts.

2. Es wird ebenfalls ein Beleuchtungsbild erzeugt, in dem verschiedene Beleuchtungsstärken der Lippen erzeugt werden. Das P-H-Bild und das Beleuchtungsbild werden kombiniert. Drei Bilder werden generiert, die die verschiedenen Konturen der Lippen (Ober- und Unterlippe) verschieden stark zum Vorschein bringen (Abb. 3.12).

3. Durch die Vorbearbeitung können die Schlüsselpunkte durch einen Such-Algorithmus extrahiert werden.

4. Anhand von fünf unabhängigen Kurven wird das Modell erzeugt, hierbei werden die extrahierten Schlüsselpunkte genutzt. Die Schlüsselpunkte dienen zur Konstruktion der Lippenkontur und des *Templates*.

5. Aus dem Modell können verschiedene Merkmale berechnet werden: Lippenbreite, Lippenhöhe und verschiedene Distanzen von Ober- zu Unterlippe.

Konsequenzen für die weitere Vorgehensweise

Im ersten Ansatz wird ein Modell genutzt, um die Lippen zu beschreiben. Das Modell wurde jedoch nicht selbst erzeugt, sondern nutzt eine, von der Kamera mit gelieferte Software. Außerdem ist das Verfahren nur anwendbar mit einer Tiefenbildkamera, jedoch werden bei der Nutzung einer Tiefenbildkamera zusätzlich hohe Ressourcen benötigt. Einige für die Arbeit durchgeführte Tests bestätigen, dass der Rechenbedarf, bei Verwendung einer Tiefenbildkamera hoch ist. Daher wird auf die ausschließliche Nutzung einer Tiefenbildkamera abgesehen. Jedoch kann die Tiefenbildkamera minimal genutzt werden. Hierbei kann die Distanz von Kamera zu Gesicht verwendet werden, um Lippenmerkmale unabhängig zu dieser Entfernung zu machen. Das Verfahren wurde nur mit türkischen Wörtern getestet. Diese Wörter sind nur isolierte Wörter (3.3) und stark unterschiedlich voneinander.
Im zweiten Ansatz wird eine brauchbare bildbasierte Vorverarbeitung durchgeführt, um die Schlüsselpunkte zu bestimmen. Die Kombination von „Pseudo-Hue“ und Beleuchtungsbildern wird in dieser Arbeit ebenfalls verwendet (4.4.2). Die vorgestellte Arbeit nutzt ein Modell, welches Trainingsbilder benötigt. Dies wird nicht in dieser Arbeit genutzt. Die Merkmale der Lippen sind gut gewählt, da sie geometrische bzw. anatomische Eigenschaften der Lippen beschreiben. In dieser Arbeit werden ebenfalls anatomische Lippenmerkmale genutzt. Im nächsten Abschnitt des Kapitels wird einzeln auf die Prozessschritte eingegangen.

3.5 Prozesskettenanalyse

In diesem Abschnitt wird die Prozesskette (Abb. 2.1), die im Abschnitt 2.1.2 vorgestellt wurde, analysiert. Es werden die einzelnen Prozessschritte betrachtet. Hierbei werden ebenfalls vergleichbare Projekte genannt und die dort benutzen Methoden/Verfahren. Es wird entschieden welche Methoden in dieser Arbeit genutzt werden sollen.

3.5.1 Lokalisierung des Gesichts

Bevor die Lippen lokalisiert und dessen Merkmale extrahiert werden können, muss das Gesicht lokalisiert werden. Somit wird das Bild auf das Gesicht begrenzt, dies beschleunigt die Suche nach den Lippen. Die Technologie um Gesichter zu lokalisieren hat sich in den letzten Jahrzehnten verbessert (Sharifara u. a. [2014]). Es ist möglich in Echtzeit Gesichter zu verfolgen (tracking). Außerdem gibt es eine Vielzahl von Anwendungen/Implementation/Verfahren zur Gesichtserkennung. Die Technologie um Gesichter zu lokalisieren scheint ausgereift zu sein. In dieser Arbeit wird eine bereits vorhandene Gesichtserkennung aus der freien Bibliothek „OpenCV“ genutzt. OpenCV ist eine Bibliothek, die für die Sprachen C und C++ geschrieben ist und bietet zahlreiche Verfahren an um Bildverarbeitungen durchzuführen. Für die Gesichtserkennung/-lokalisierung wird OpenCV's „Haar Cascade“ genutzt. In diesem Abschnitt wird die Theorie hinter dem Verfahren erklärt und im Abschnitt 4.2 geht es um die Realisierung in dieser Arbeit. Ein Grund für die Wahl dieser Gesichtserkennung ist, dass sie nicht neu implementiert werden muss und so Vorhandenes wiederverwendet werden kann. Die Gesichtserkennung besitzt ebenfalls verschiedene Trainingsdaten für die Gesichter, somit ist der Aufwand für ein eigenes Training nicht notwendig.

OpenCV's Haar Cascade

Bei „Haar Cascade“ wird ein Klassifizierer verwendet, der für die Erkennung sogenannte Haar-Merkmale nutzt (Abb. 3.13). Die Idee des Verfahrens ist, dass ein Bild klassifiziert wird, indem nach einem Objekt in Bildausschnitten gesucht wird.

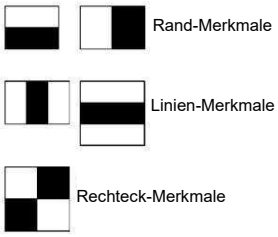


Abbildung 3.13: Haar-Merkmale
Quelle: itseez [2016]

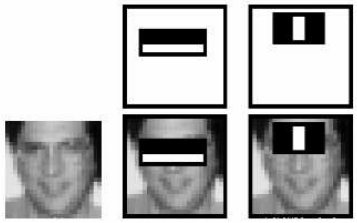


Abbildung 3.14: Haar-Merkmale über Gesichter
Quelle: itseez [2016]

Beim Algorithmus werden positive Bilder (Bilder von Gesichtern) und negative Bilder (Bilder ohne Gesichter) für das Trainieren des Klassifizierers genutzt. Ein Haar-Merkmal ist ein einfacher Wert, der berechnet wird durch Subtraktion der Grauwertsumme unter dem weißen Rechteck von der Grauwertsumme unter dem schwarzen Rechteck (Abb. 3.13). Diese Differenz wird genutzt um Bereiche im Bild kategorisieren zu können. In Abbildung 4.2 werden die Haar-Merkmale auf das Gesicht angewendet. Es werden zwei Haar-Merkmale genutzt. Das linke Haar-Merkmal wird benutzt, um die Augen zu lokalisieren. Dieses Merkmal nutzt die Information, dass die Augenpartie dunkler ist als der Nasenbereich. Beim zweiten Haar-Merkmal (rechts) wird die Helligkeit der Augen mit der über dem Nasenbein verglichen. Diese beiden Merkmale könnten beispielsweise nicht genutzt werden für die Lokalisierung der Lippen, da Haut und Lippen ähnlich dunkel/hell sind (itseez [2016]). Um die Berechnungszeit der Merkmale zu reduzieren, werden Integralbilder benutzt. Die Bildung und wie die Integralbilder genutzt werden, werden in der Arbeit Viola und Jones [2001] beschrieben.

3.5.2 Lokalisierung der Lippen

In der Prozesskette des Lippenlesens ist es notwendig die Lippen im Gesicht zu finden. Es ist deshalb erforderlich die Mundregion (*ROI*) zu lokalisieren. Die Lokalisierung der Lippen ist ein externer Prozess. Es gibt viele Techniken, um die Lippen zu lokalisieren. Bildbasierte Lippenerkennungsmethoden nutzen spatiale (räumliche) Informationen, Pixelgrauwert/Pixelfarbwert und Intensität, Kanten, Ecken und Bewegungen der Lippen, um sie zu lokalisieren (Sujatha und M.Radhakrishnan [2013]). Lippenlokalisierung wird erschwert durch die geringe Graustufenvariation um den Mund herum. Bei der Verwendung von Farbbildern nutzen viele Entwickler rote Lippen, um eine Abgrenzung zur restlichen Mundregion zu schaffen. Jedoch können schlechte Lichtverhältnisse störend in der Erkennung sein (WenJuan u. a. [2010]).

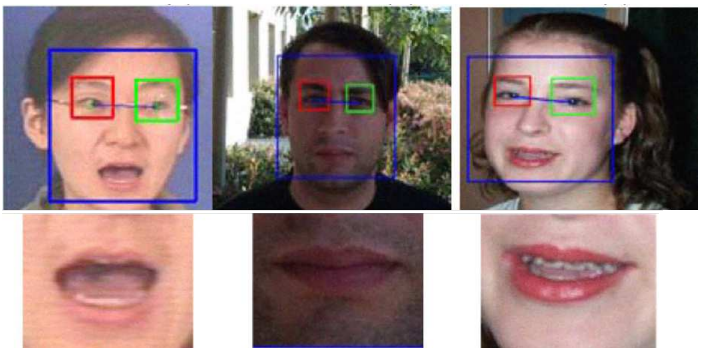


Abbildung 3.15: Lippenlokalisierung anhand der Augen
Quelle: WenJuan u. a. [2010]

In der Arbeit WenJuan u. a. [2010] werden anatomische/geographische Eigenschaften (Çeliktutan u. a. [2013]) des Gesichtes benutzt, um die Lippen zu lokalisieren. Folgende Eigenschaften werden genutzt:

- Die Distanz der Augen entspricht der Breite der Mundregion.
- Die Höhe der Mundregion ist gleich die Hälfte der Größe der Distanz von Auge bis zum Ende des Gesichts.
- Der Mund ist horizontal parallel zu den Augen, dasselbe gilt bei der Rotation.

Sie nutzen OpenCV's Haar-Merkmale, um Augenregion zu lokalisieren. Nachdem die Augenregion lokalisiert wurde, werden die Augäpfel lokalisiert. Nun nutzen sie die anatomischen Eigenschaften (siehe obige Auflistung) um die Lippenregion zu bestimmen (Abb. 3.15). Nachteil dieser Methode ist es vorher die Augen bzw. die Pupille zu erkennen. In dieser Arbeit werden ebenfalls anatomische/geographische Eigenschaften der Lippen für die Lokalisierung genutzt. Dies hat ein Vorteil gegenüber der modellbasierten Lokalisierung, da kein Modell der Lippen benötigt wird. Außerdem erfolgt eine einfache Berechnung. Bei der modellbasierten Lokalisierung sind rechenaufwändige Berechnungen notwendig. Für eine Echtzeitanwendung kann dies von Nachteil sein.

Mund-ROI anhand der Gesichts-ROI

Bezüglich der Lippenlokalisierung orientiert sich diese Arbeit an den Arbeiten Bacivarov u. a. [2008] und Sujatha und M.Radhakrishnan [2013]. Hierbei handelt es sich um bildbasierte Lippenerkennungsmethoden. Es werden zunächst die beiden Methoden vorgestellt, dann werden die Methoden diskutiert. Die in dieser Arbeit realisierte Methode zur Lippenlokalisierung wird in Abschnitt 4.3 vorgestellt.

Bei der ersten Methode (Bacivarov u. a. [2008]) wird die Mundregion mit Hilfe der *ROI* des Gesichts ermittelt bzw. berechnet.

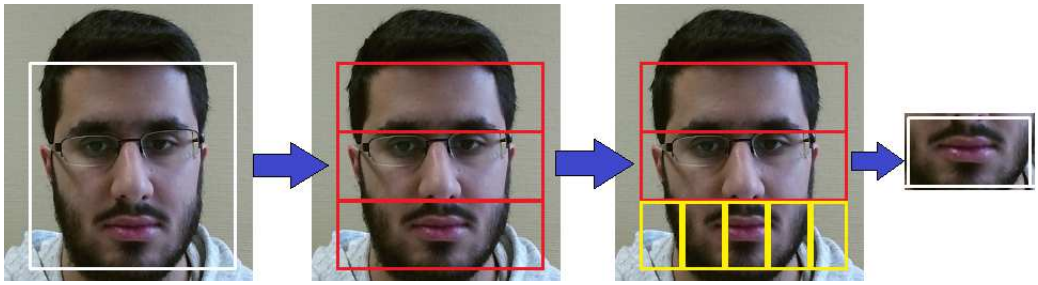


Abbildung 3.16: Verfahren 1: Lippenlokalisierung

Die *ROI* des Mundes befindet sich in der Höhe im letzten Drittel der Gesichts-*ROI*. Die Breite der Mund-*ROI* wird ermittelt, in dem die Gesichts-*ROI* in fünf Segmente geteilt wird. Nach dem ersten Fünftel beginnt und beim letzten Fünftel endet die Breite der Mund-*ROI*. Auf dem obigen Bild 3.16 ist die Prozesskette zur Lokalisierung des *ROI* des Mundes zu sehen.

$$S_H = h_G/3 \tag{3.4}$$

$$S_B = b_G/5 \tag{3.5}$$

$$h_M = S_H \tag{3.6}$$

$$b_M = S_B * 3 \tag{3.7}$$

$$x_M = x_G + S_B \tag{3.8}$$

$$y_M = y_G + S_H * 2 \tag{3.9}$$

In den Gleichungen von 3.4 bis 3.9 ist die Berechnung der Koordinaten der Mund-*ROI* der ersten Methode zu sehen. h_M und b_M stehen für die Höhe und Breite des Mundes. h_G und h_G stehen für die Höhe und Breite des Gesichts. Die Berechnungen erfolgen anhand der obigen Erklärung. Hierbei stehen h für die Höhe, b für die Breite, x für die x-Koordinate in einem Bild und y für die y-Koordinate in einem Bild. S_B und S_H in Gleichungen 3.4 und 3.5 stehen für die Segmentgrößen, jeweils für die Höhe und Breite. Diese Segmentgrößen werden entsprechend der obigen Erklärung genutzt, um die Mundkoordinaten zu bestimmen (Gleichungen 3.8 und 3.9)

Die zweite Methode (Sujatha und M.Radhakrishnan [2013]) benutzt, genau wie die erste Methode, die Gesicht-*ROI*, um die Mund-*ROI* zu berechnen.

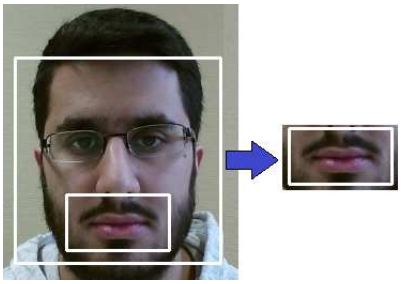


Abbildung 3.17: Verfahren 2: Lippenlokalisierung

Der Mund befindet sich in der Regel in der unteren Hälfte des Gesichts. Auf Basis dieser Annahme wurde in der Arbeit Sujatha und M.Radhakrishnan [2013] ein Algorithmus entwickelt der die Mund-*ROI* berechnet. In der Arbeit wurde die Methode auf 175000 Frames angewendet und validiert. Die Mund-*ROI* wurde mit einer Genauigkeit von 91,15 % lokalisiert. Auf dem Bild 3.17 ist das Resultat der Lippenlokalisierung zu sehen.

$$bf_M = x_G + b_G \tag{3.10}$$

$$hf_M = y_G + b_G \tag{3.11}$$

$$x_M = x_G + (bf_M - x_G)/4 \tag{3.12}$$

$$y_M = y_G + (hf_M - y_G)/1.5 \tag{3.13}$$

$$h_M = (hf_M - (hf_M - y_G)/15) - y_M \tag{3.14}$$

$$b_M = (bf_M - (bf_M - x_G)/4) - x_M \tag{3.15}$$

In den Gleichungen 3.10 bis 3.15 ist der Algorithmus zu sehen, um die Koordinaten der Mund-*ROI* zu berechnen.

Konsequenzen für die weitere Vorgehensweise

Der Vorteil dieser Methoden ist, dass einfache Berechnungen anhand von Koordinaten (des Gesicht-*ROI*) durchgeführt werden. Die Berechnungen sind schneller als beispielsweise die Verwendung von einem geographischen Modell der Lippen oder komplexe Prozeduren, um Ecken und Kanten der Lippen zu lokalisieren (Sujatha und M.Radhakrishnan [2013]). Nachteil dieser Methoden ist, dass die Berechnung der Mund-*ROI* nur möglich ist, wenn das Gesicht frontal zur Kamera gewandt ist.
In der ersten Methode wird eine größere Mund-*ROI* als in der zweiten Methode berechnet, wie auf den Abbildungen 3.16 und 3.17 zu sehen ist. Es werden somit die Lippen vollständig erfasst.

Dies ist auch notwendig, weil es Variationen in der Bewegung des Mundes bei Bildung der Wörter geben kann. Hierbei spielt die Höhe der Mund-*ROI* eine Rolle, da in der Aussprache der Mund vertikal verschieden hohe Größen annehmen kann. Nachteil zur zweiten Methode ist, dass mehr aufgezeichnet wird als notwendig. So kann es sein, dass bei der Merkmalsextraktion, Algorithmen länger benötigen, da die Mund-*ROI* größer ist. Außerdem kann die Erkennung von Wörtern fehleranfällig sein, da nicht nur die Mund-*ROI* betrachtet wird.
Der Vorteil der zweiten Methode gegenüber der ersten Methode ist, dass exakt die Mundregion erfasst wird. Es wird nicht mehr als nötig aufgezeichnet. Die Breite und Höhe der Mundregion grenzt an die Lippen, somit sind überflüssige Pixel nicht vorhanden. Nachteil gegenüber der ersten Methode ist, dass die *ROI* zu minimalistisch ist, da gerade die Höhe der Mund-*ROI* eine entscheidende Rolle in der kompletten Aufzeichnung der Lippen und somit der (visuellen) Sprache spielt.
In dieser Arbeit wird der Vorteil beider Verfahren kombiniert. Im Abschnitt 4.3 wird darauf eingegangen.

3.5.3 Merkmalsextraktion

Im folgenden Abschnitt wird ein Verfahren vorgestellt um Lippenmerkmale zu extrahieren. Der Algorithmus zur Merkmalssuche entstammt aus der Arbeit Bangsawan u. a. [2015]. Die dort beschriebene Merkmalsextraktion der Lippen wird jedoch nicht verwendet. Die Probleme, die dabei entstehen werden in diesem Abschnitt aufgezeigt. In diesem Abschnitt werden ebenfalls mögliche Lippenmerkmale aufgelistet. Das endgültige eingesetzte Verfahren zur Merkmalsextraktion wird im Abschnitt 4.4 vorgestellt.

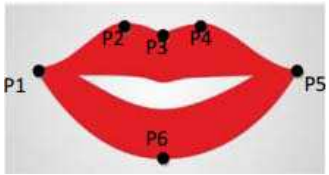


Abbildung 3.18: Sechs Schlüsselpunkte
Quelle: Bangsawan u. a. [2015]

Abbildung 3.19: Segmentierung
Quelle: Bangsawan u. a. [2015]

In der Arbeit Bangsawan u. a. [2015] werden sechs Schlüsselpunkte (Abb. 3.18) der Lippen mithilfe eines adaptiven Schwellwerts segmentiert. Die Schlüsselpunkte der Lippen geben Hinweise auf die Kontur der Lippen. Die Farbinformation der Lippen wird für die Segmentierung genutzt, da die Sättigung der Lippen höher ist als die der Haut. Die Lippensegmentierung wird wie folgt ausgeführt:

1. Berechnung der Farbsättigung:

$$s(x,y) = 2 * tan^{-1}\left(\frac{R(x,y) - G(x,y)}{R(x,y)}\right) \div \pi \tag{3.16}$$

$s(x,y)$ beschreibt die Sättigung, R und G sind die Rot- und Grünkomponenten des Pixels (x,y). Mit der Gleichung 3.16 wird die Sättigung normalisiert zwischen 0 und 1.

2. Der Schwellwert wird adaptiv also anpassend zu verschiedenen Lippengrößen berechnet. Hierbei wird N auf ein Drittel der Fläche der Mund-*ROI* gesetzt. Es wird davon ausgegangen, dass die Lippen ein Drittel der Fläche in dem Mund-*ROI* ausmachen.

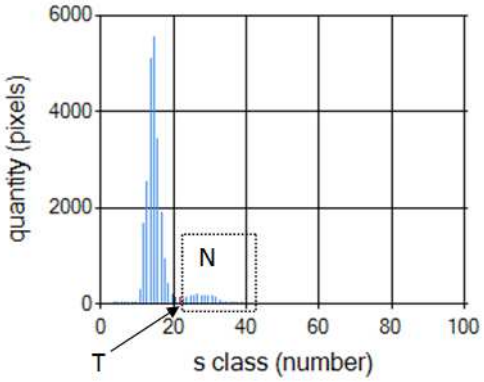
3. 

Abbildung 3.20: Häufigkeitsverteilung der Sättigung
Quelle: Bangsawan u. a. [2015]

Es wird eine Häufigkeitsverteilung (Abb. 3.20) der Sättigung in der Mund-*ROI* erstellt.

4. Da die Lippen einen hohen Sättigungswert haben, befinden sich deren Werte im hohen Bereich (rechts) der Häufigkeitsverteilung (Abb. 3.20). Zwei Drittel der Mund-*ROI* sind keine Lippen. Es wird daher der adaptive Schwellwert anhand der Häufigkeitsverteilung bestimmt, indem die Pixel in der Häufigkeitsverteilung bis zu zwei Drittel der Mund-*ROI* abgezählt werden (Abb. 3.20).

5.

$$l(x,y) = \begin{cases} 255 & s(x,y) \geq T \\ l(x,y) & \text{sonst} \end{cases} \tag{3.17}$$

Die Segmentierung erfolgt indem die Pixel ($l(x,y)$) deren Sättigung ($s(x,y)$) größer gleich dem adaptiven Schwellwert T sind, auf weiß gesetzt werden.

6. Mithilfe eines Suchalgorithmus, der zuerst weiße auftretende Pixel sucht, werden die Schlüsselpunkte im neuen Bild (Abb. 3.19) gesucht.

Konsequenzen für die weitere Vorgehensweise

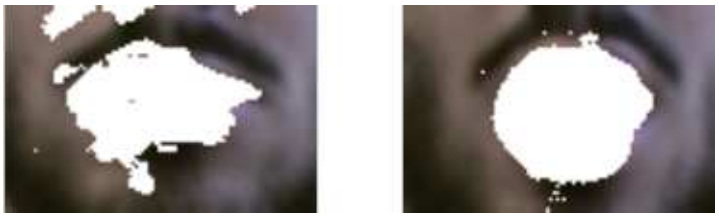


Abbildung 3.21: Fehlgeschlagene Sättigungsmethode

Das präsentierte Verfahren nutzt die Sättigung der Lippen für die Segmentierung. Es werden also Vorinformationen bzw. Pixelcharakteristika (siehe Abschnitt 3.2.2) genutzt. In dieser Arbeit werden ebenfalls Pixelcharakteristika verwendet. Nachteil dieser Methode ist, dass sie nicht geeignet ist für dunkle Lippen (laut Arbeit). Außerdem ist diese Methode fehleranfällig bei geringer Beleuchtung. Ein weiteres Problem ist, dass die Segmentierung nicht bei dynamischen Bewegungen (Lippenbewegungen) funktioniert. Laut der Arbeit gibt es Fehler in der Segmentierung wenn der Mund geöffnet ist. Eigene Tests bestätigen dies (Abb. 3.21). Der verwendete Suchalgorithmus um die Schlüsselpunkte in Lippenbilder zu suchen wird in dieser Arbeit verwendet und im Abschnitt 4.4 präsentiert.

3.5.4 Lippenmerkmale

Es folgt eine Auflistung von möglichen Lippenmerkmalen. Folgende Lippenmerkmale können durch bildbasierte Verfahren extrahiert werden:

- **Sechs Schlüsselpunkte**:

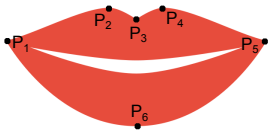


Abbildung 3.22: Sechs Schlüsselpunkte

  Die sechs Schlüsselpunkte (Abb. 3.22) basieren auf anatomische Eigenschaften der Lippen.

- **Breite und Höhe**:

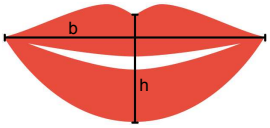


Abbildung 3.23: Breite und Höhe

  Die Breite b und Höhe h (Abb. 3.23) stellen weitere Merkmale dar. Die Breite kann durch die Punkte P_1 und P_5 und die Höhe kann durch die Punkte P_3 und P_6 berechnet werden.

- **Seitenverhältnis**: Das Seitenverhältnis s der Lippen kann anhand der Breite und Höhe berechnet werden. Die Gleichungen lauten b/h oder h/b.

- **Fläche**:

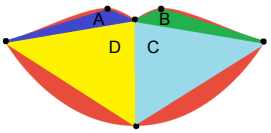


Abbildung 3.24: Fläche

  Die Fläche der Lippen (Abb. 3.24) ist ein weiteres Merkmal. Die Fläche kann berechnet werden, indem aus den sechs Schlüsselpunkte Dreiecke gebildet werden. Der Flächeninhalt der Dreiecke (A, B, C und D) beschreibt die Fläche der Lippen (ur Rehman Butt und Lombardi [2013]).

- **Mundhöhle**: Zähne und Zunge sind bei einer Äußerung teils sichtbar. So kann die Mundhöhle als Merkmalsvektor dienen, indem die Grau- und/oder Farbwerte gespeichert werden (Saitoh u. a. [2008]).

3.6 Dynamic-time-warping

Die vorgestellten Lippenmerkmale in Abschnitt 3.5.4 erzeugen bei einer Äußerung einen Merkmalsvektor über die Zeit. Dieser Merkmalsvektor kann als Trajektorie (siehe Abschnitt 2.2) dargestellt werden. Für eine Ähnlichkeitsanalyse von Trajektorien kann das „Dynamic-time-warping“ (DTW) eingesetzt werden.

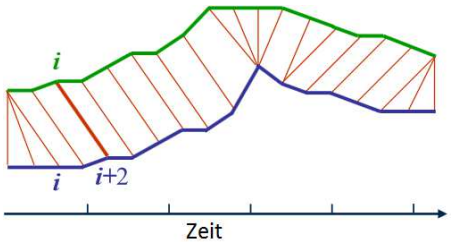


Abbildung 3.25: DTW: Angleichung von Trajektorien
Quelle: Criel und Tsiporkova [2006]

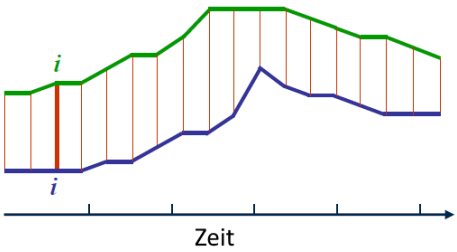


Abbildung 3.26: Einfaches Vergleichen von Trajektorien
Quelle: Criel und Tsiporkova [2006]

Mit DTW ist es möglich ein Abgleich von zwei Trajektorien mit unterschiedlichen Längen, Geschwindigkeiten und Variationen in der Zeit durchzuführen. Variation in der Zeit bedeutet, dass eine Äußerung zeitlich versetzt (zeitliche Verzögerung) auftreten kann. Stauchungen/-Dehnungen der Trajektorie treten auf durch unterschiedliche Sprechgeschwindigkeit.

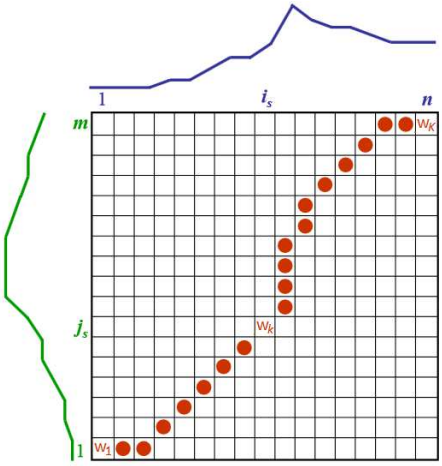


Abbildung 3.27: DTW: Matrix mit „Warping-Pfad“
Quelle: Criel und Tsiporkova [2006]

Durch das DTW werden die Variationen der Trajektorien passend “gewarpt„ (Abb. 3.25). In Abbildung 3.25 ist zu sehen, wie das DTW zwei Trajektorien abgleicht. Es findet eine intuiti-

ve Ähnlichkeitsmessung statt, indem nicht nur die Punkte beider Trajektorien zum Zeitpunkt i abgeglichen werden (wie in Abb. 3.26), sondern Zeitverschiebungen mit berücksichtigt werden. DTW erlaubt somit eine elastische Verschiebung der Zeitachsen (Yanagisawa und Satoh [2006], D'Urso [2000]).
Es wird angenommen, dass zwei zweidimensionale Trajektorien P und Q mit deren Längen n und m gegeben sind. In diesem Fall beschreibt eine Trajektorie eine Sequenz von Merkmalswerten zum Zeitpunkt t:

$$P(t) = p_t \tag{3.18}$$

$$P = p_1, p_2, ..., p_i, ..., p_n \tag{3.19}$$

$$Q(t) = q_t \tag{3.20}$$

$$Q = q_1, q_2, ..., q_j, ..., q_m \tag{3.21}$$

Um DTW anwenden zu können und die Ähnlichkeit zwischen zwei Trajektorien zu bestimmen ist es notwendig eine $n * m$ Matrix zu erstellen. Die Elemente der Matrix (i, j) beinhalten die Distanz $d(p_i, q_j)$ zwischen den zwei Merkmalswerten p_i und q_j. Die Distanz beschreibt in diesem Fall, wie ähnlich sich die Merkmale sind. Sind beide Merkmalswerte gleich so ist die Distanz gleich null. Folgende Distanzfunktionen können angewendet werden:

$$d(p_i, q_j) = (p_i - q_j)^2 \tag{3.22}$$

$$d(p_i, q_j) = |p_i - q_j| \tag{3.23}$$

Die Gleichung 3.23 beschreibt die absolute Distanz. Nach dem die Distanzmatrix erstellt wurde ist der nächste Schritt den besten Abgleich zwischen den Trajektorien P und Q zu finden. Hierfür muss der sogenannte „Warping-Pfad" (WP) W (Gleichung 3.25) gefunden bzw. berechnet werden. Der WP ist ein Pfad von Matrixelementen (Gleichung 3.24), welche die Zuordnung zwischen P und Q definiert (Abb. 3.27).

$$w_k = (i, j)_k \tag{3.24}$$

$$W = w_1, w_2, ..., w_k, ..., w_K \qquad max(n, m) \leq K < n + m + 1 \tag{3.25}$$

1. **Grenzbedingung**:

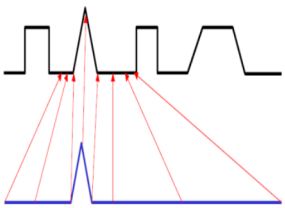


Abbildung 3.28: Grenzbedingung
Quelle: Criel und Tsiporkova [2006]

Es ist notwendig, dass der WP diagonal verläuft. Startpunkt ist $w_1 = (1, 1)$ und Endpunkt ist $w_K = (n, m)$. Diese Bedingung garantiert, dass der Abgleich nicht nur einen Abschnitt einer Trajektorie betrifft (Abb. 3.28).

2. **Kontinuität**:

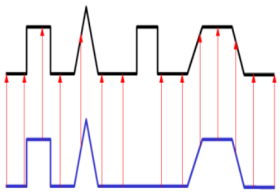


Abbildung 3.29: Kontinuität
Quelle: Criel und Tsiporkova [2006]

Der WP darf keine Werte auslassen, sonst würde er in der Zeit „springen“. Folgendes muss zutreffen: $w_k = (a, b)$ dann ist $w_{k-1} = (a', b')$ bei $a - a' \leq 1$ und $b - b' \leq 1$. Ein Schritt im WP muss auf benachbarte Matrixzellen folgen. Diese Bedingung garantiert, dass wichtige Merkmale nicht ausgelassen werden (Abb. 3.29).

3. **Monotonie**:

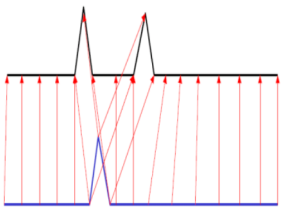


Abbildung 3.30: Monotonie
Quelle: Criel und Tsiporkova [2006]

Der WP darf nicht in negative Zeitrichtung gehen, d.h. die Indizes müssen fortlaufend sein. Folgendes muss zutreffen: $w_k = (a, b)$ dann ist $w_{k-1} = (a', b')$ bei $a - a' \geq 0$ und $b - b' \geq 0$. Diese Bedingung garantiert, dass Merkmale beim Abgleich nicht wiederholt werden (3.30).

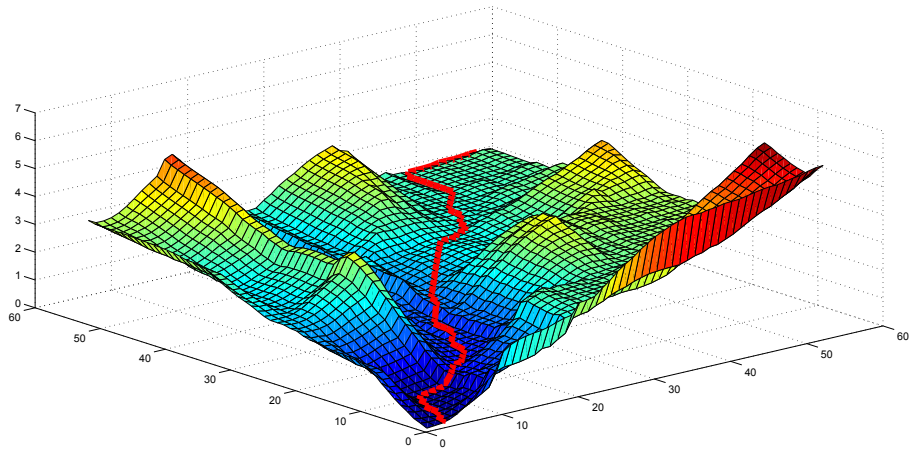


Abbildung 3.31: 3-D Ansicht: Kostenmatrix mit „Warping-Pfad“

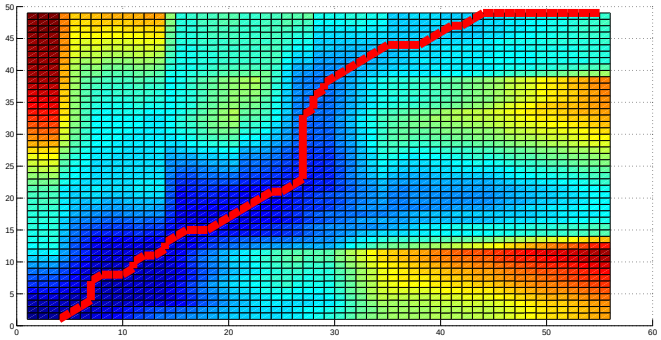


Abbildung 3.32: 2-D Ansicht: Kostenmatrix mit „Warping-Pfad“

Der WP kann sehr effizient durch dynamische Programmierung gefunden werden. Bei der dynamischen Programmierung wird ein Problem gelöst, indem Teilprobleme gelöst (Optimalitätsprinzip von Bellman) werden. Die Teilprobleme werden bearbeitet: Beispielsweise wird in einem Matrixabschnitt von Punkt (i, j) aus der kürzeste gewichtete Nachbarpunkt gesucht. Die Gewichtung und Richtung, also die Lösung des Teilproblems wird zwischengespeichert. Das Lösen der Teilprobleme geschieht iterativ. Die Matrix wird weiter durchlaufen und somit eine Bestwegesuche durchgeführt, indem die gewichteten Teilstände weiter abgespeichert werden. Am Ende werden diese Teilstände für die Gesamtlösung zusammengesetzt. Der kürzeste Weg kann in der Matrix nun abgelesen werden (Si u. a. [2009]). Übertragbar auf dieses Problem

wird die zuvor berechnete Distanzmatrix und eine lokale Bedingung für die Kontinuität (lBK) genutzt. Bei der lBK wird die Schrittweise, in der die Distanzmatrix durchlaufen wird, definiert (Abb. 3.33). Um den WP zu berechnen wird die Distanzmatrix mithilfe des lBK durchlaufen, dabei werden die Ergebnisse in eine neue Kostenmatrix K gespeichert (lösen und zwischenspeichern des Teilproblems). Es wird der Pfadschritt gewählt, der die minimale akkumulierte (aufaddierte) Distanz in der Kostenmatrix hat (Keogh und Pazzani [2001]).

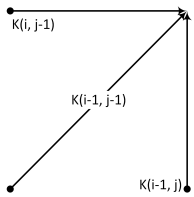


Abbildung 3.33: Lokale Bedingung für die Kontinuität

$$K(i,j) = min \left\{ \begin{array}{c} \text{K(i, j-1)} \\ \text{K(i-1, j)} \\ \text{K(i-1, j-1)} \end{array} \right\} + d(p_i, q_j) \tag{3.26}$$

Die Gleichung 3.26 zeigt den oben beschriebenen Prozess. Der Graph in Abbildung 3.31 stellt die Kostenmatrix dar. Hierbei stellen die roten höhere Kosten dar als die blauen Bereiche. Der Bereich verläuft also vom Blauen bis zum Roten. Der WP läuft entlang der Bereiche mit den geringsten Kosten (blaue Bereiche). Der WP wird anhand von „Backtracking“ (BT) bestimmt. BT ist ein Verfahren welches rückwärts laufend einen Pfad bestimmt. In diesem Fall wird der Pfad mit den geringsten Kosten gefunden. Das BT fängt an der Stelle (n, m) an und endet gemäß Grenzbedingung in $(1, 1)$ (Sakoe und Chiba [1978]).
Der Indikator für die Ähnlichkeit zwischen zwei Trajektorien ist der WP bzw. die Summe der Werte in der Kostenmatrix entlang des WPs. Je geringer die Summe desto ähnlicher sind sich die Trajektorien. Es wird angenommen, dass sich der optimale WP also der beste Abgleich entlang der Diagonale der Kostenmatrix befindet. Dies muss nicht in jeder Domäne der Fall sein. In dieser Arbeit werden Kommandos erkannt. Diese Kommandos sind verbundene Wörter (siehe Abschnitt 3.3) und werden nicht innerhalb einer kontinuierlichen Sprache erkannt. Es ist bekannt wann eine Äußerung anfängt und wann sie endet (siehe Abschnitt 4.3.1). Daher wird der optimale WP in der Diagonale der Matrix gesucht. In anderen Anwendungsgebieten in denen die Trajektorien durch Events an stark unterschiedlichen Zeitpunkten anfangen und enden, verläuft der optimale WP nicht unbedingt entlang der Diagonale (Salvadore und Chan [2004]).

Schritt-Muster und Warping-Fenster

In Abbildung 3.33 ist ein sogenanntes „Step-Pattern“ (Schritt-Muster) zu sehen. Es gibt verschiedene Schritt-Muster. Das Schritt-Muster bestimmt, wie die Kostenmatrix aufgestellt wird. Die Grundidee ist den zulässigen „Warping-Pfad“ zu begrenzen. Hierbei werden lokale Bedingungen aufgestellt, die die Schrittweise, bei der Berechnung der Kostenmatrix, definiert. Die Wahl des optimalen Schritt-Musters basiert auf Fachwissen oder durch Ausprobieren („Try-and-error“) (Rabiner und Juang [1993]).

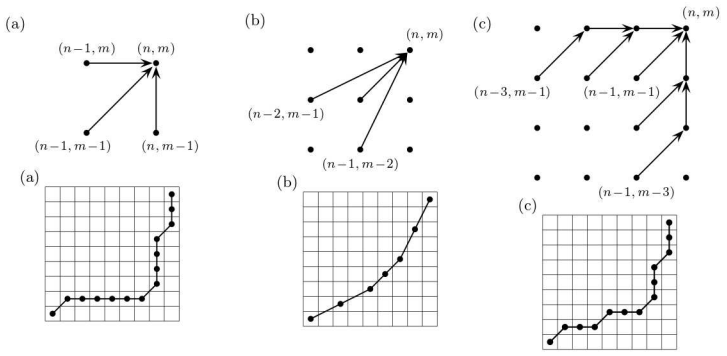


Abbildung 3.34: Verschiedene Schritt-Muster mit WP
Quelle: Müller [2007]

In Abbildung 3.34 sind drei verschiedene Schritt-Muster zu sehen. Die verschiedenen Schritt-Muster haben einen Einfluss auf den WP, da die Kostenmatrizen unterschiedlich sind. Durch die Auswahl des richtiges Schrittmusters (*b)* in Abb. 3.34) kann gewährleistet werden, dass der WP diagonal verläuft und somit ein optimalen Abgleich zwischen den zwei Trajektorien durchführt.

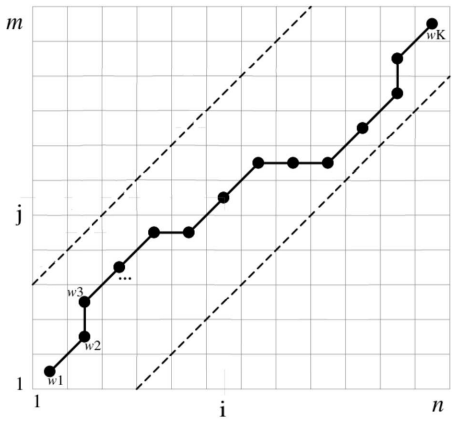


Abbildung 3.35: Warping-Fenster
Quelle: Verändert aus Keogh und Pazzani [2001]

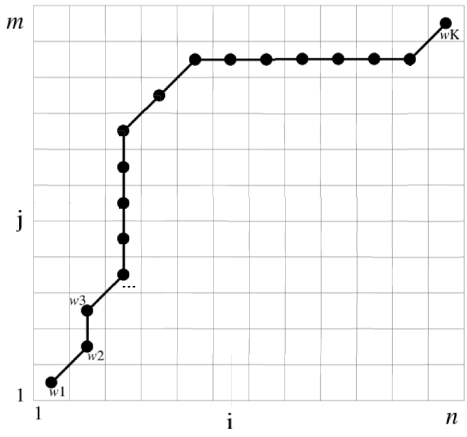


Abbildung 3.36: Nicht optimaler WP
Quelle: Verändert aus Keogh und Pazzani [2001]

Eine weitere Beschränkung für das Finden des „Warping-Pfads“ ist die Nutzung eines „Warping-Fensters“ (Abb. 3.35). Durch die Verwendung eines Warping-Fensters wird gewährleistet, dass der WP entlang der Diagonale verläuft. Die optimale Größe wird anhand von Erfahrung und durch Ausprobieren gefunden. Mithilfe des Warping-Fensters wird ein pathologischer Abgleich von zwei Trajektorien verhindert, d.h. es wird nicht überwiegend ein einzelner Abschnitt der ersten Trajektorie auf die gesamte andere Trajektorie abgebildet (Abb. 3.36) (Paliwal u. a. [1982]).

Konsequenzen für die weitere Vorgehensweise

In dieser Arbeit werden Merkmale des Mundes als Merkmalsvektor erfasst. Diese Merkmale entstehen bei einer Äußerung eines Kommandos. Die Merkmalsvektoren sind Werte über die Zeit und können daher als Trajektorie aufgefasst werden. Das „Dynamic-time-warping“ kann eingesetzt werden, um die Ähnlichkeit von zwei Trajektorien zu messen. DTW ist somit geeignet für Ähnlichkeitsmessung von Äußerungen bzw. Kommandos in dieser Arbeit. Äußerungen können durch Aussprache variieren. Die Sprechgeschwindigkeit der gleichen Äußerung kann unterschiedlich schnell/langsam sein. DTW kann diese verschiedenen Variationen und Geschwindigkeiten durch das „warpen“, in dem Zeitachsen verschoben werden, wie in 3.6 gezeigt, ausgleichen (Yanagisawa und Satoh [2006]). Durch die Vorinformation wann eine Äußerung, ein Kommando ausgesprochen und enden wird, kann angenommen werden, dass das optimale Abgleichen entlang der Diagonale der Kostenmatrix verläuft. Somit können die verschiedenen Bedingungen und Einschränkungen auf den DTW-Algorithmus angewandt werden, um eine hohe Wiedererkennungsrate zu erhalten.

3.7 Problemstellungen

In diesem Abschnitt werden die Problemstellungen, die innerhalb der Analyse erkannt wurden, kurz zusammengefasst und mögliche Lösungen vorgestellt.
Visuelle Spracherkennung für kontinuierliche Sprache ist schwer zu realisieren. Für die kontinuierliche Sprache müssen brauchbare Einheiten definiert werden. Mundbilder sind die visuellen Repräsentanten von Phonemen. Phoneme werden für die akustische Spracherkennung eingesetzt. Jedoch eignen sich Mundbilder nicht (siehe Abschnitt 3.3). Mundbilder geben keine Informationen über den Übergang von Mundbild zu Mundbild. Außerdem sind einem Mundbild mehrere Phoneme als Repräsentant zugewiesen, da Bewegungen die in der Mundhöhle (Zunge, Zähne und Rachen) geschehen, verdeckt werden. Es wird daher keine kontinuierliche Sprache erkannt, sondern es werden Kommandos, also verbundene Wörter erkannt. Die Kommandos sind leicht verständlich für Ausführungen von Robotern. Die Wörter werden abstrakt betrachtet. Sie werden als Ganzes und nicht in ihre Einheiten aufgefasst.
Ziel dieser Arbeit ist es einen Prototyp zu entwickeln. Die Komplexität in der Entwicklung ist hoch, die Arbeit soll als Grundlage dienen. Einzelne Prozessschritte können modular verändert bzw. verbessert werden. Dies ist einer der Gründe, weshalb bildbasierte Verfahren eingesetzt werden. Beim modellbasierten Verfahren sind die Anwendungen komplex, es bedarf eine eigene Studie für die Entwicklung einer solchen Anwendung, um beispielsweise ein robustes und stabiles Lippenverfolgen zu ermöglichen. In modellbasierten Verfahren ist es ebenfalls notwendig ein eigenes sehr großes Trainingsset zu schaffen, um ein zuverlässiges Modell aufzubauen. Bildbasierte Verfahren werden genutzt. Bildbasierte Verfahren sind fehleranfällig bei verschiedenen Lichtverhältnissen und bei sonstigen Störungen im Bild. Die verwendete Lippenlokalisierung kann fehlschlagen, denn es werden Berechnungen genutzt, um die *ROI* zu bestimmen. Es ist ggf. notwendig eine bestimmte Ausrichtung und Distanz zur Kamera einzuhalten, um ein zuverlässiges Lokalisieren zu ermöglichen.
Für die Ähnlichkeitsanalyse von Trajektorien wird DTW eingesetzt. Das zu realisierende System ist eine Echtzeitanwendung, daher kann es sein, dass die ständige Aufstellung einer Matrix und das Suchverfahren rechenaufwendig sind. Um diese Problematik zu lösen, können vorgestellte Einschränkungen/Bedingungen genutzt werden (siehe Abschnitt 3.6). Durch diese Einschränkungen beispielsweise durch die Verwendung eines „Warping-Fenster“ wird das DTW beschleunigt.

3.8 Anforderungen

Im folgenden Abschnitt werden die Anforderungen an das zu realisierende System gestellt. Es wird die Frage beantwortet, was das System machen soll. Dieser Abschnitt dient ebenfalls als Fazit zur Analyse. Es wird Bezug zu jedem Abschnitt in der Analyse genommen.

- Das zu entwickelte visuelle Spracherkennungssystem wird (Kurz-) Kommandos versuchen zu erkennen. Die Kommandos, die das System erkennen kann, sind bekannt. Das System wird auf einem Assistenzroboter laufen. Der Assistenzroboter nimmt die Kommandos zur Echtzeit per Kamera auf und versucht sie zu erkennen.
- Das Lippenlesen wird durch bildbasierte Methoden realisiert. Modellbasiertes Vorgehen sind komplex in der Realisierung, benötigen ein großes Set an Trainingsdaten und Rechenleistung. Die Trainingsdaten haben einen starken Einfluss auf ein zuverlässiges Modell. Bildbasierte Methoden, wie die Nutzung von Pixelcharakteristiken und Extraktion von Konturen werden eingesetzt. Die Lippen haben eine andere Farbe und Sättigung als die Gesichtshaut. Dieses Merkmal wird genutzt für die Segmentierung der Lippen. Die Konturen der Lippen werden extrahiert und typische Lippenmerkmale gefunden (siehe Abschnitt 4.4).
- Das visuelle Spracherkennungssystem wird keine kontinuierliche Sprache versuchen zu erkennen. Die Problematik wurde in der Machbarkeitsanalyse (3.3) erörtert. Es werden verbundene Wörter erkannt. Diese Wörter stellen die Kommandos an den Roboter dar.
- Das System wird nicht als ein ausschließliches Wiedererkennungssystem eingesetzt, sondern es ist als ein, für ein akustisches Spracherkennungssystem, unterstützendes System gedacht. In der Machbarkeitsanalyse (3.3) wurde dargestellt, dass mehrere Phoneme einem Mundbild zugeordnet werden. Der Grund ist, dass viele Laute im Mundhohlraum passieren. Aus diesem Grund wird die Wiedererkennungsrate nicht ausschließlich bei hundert Prozent sein. Im Kapitel „Lippenlesen“ (siehe Abschnitt 4.5.3) wird das Maß für die Erkennungsrate erläutert.
- Für die Gesichtslokalisierung wird die Gesichtserkennung von OpenCV - „Haar Cascade“ - eingesetzt. Das Gesicht muss frontal zur Kamera gerichtet sein, so wird gewährleistet, dass das Gesicht problemlos lokalisiert werden kann.
- Um die Lippen zu lokalisieren werden anatomische Merkmale des Gesichts genutzt. Die Position der Mund-*ROI* wird berechnet. Hierzu wird die Gesichts-*ROI* genutzt. Es ist daher erforderlich, dass die Gesichtserkennung problemlos durchgeführt wird.

- Die sechs Schlüsselpunkte der Lippen werden genutzt, um die das Seitenverhältnis und die Fläche des Mundes zu berechnen. Diese Merkmale werden zur Wiedererkennung einer Äußerung genutzt. Die Erläuterung der Implementierung erfolgt in Abschnitt 4.4.
- Das „Dynamic-time-warping“ wird für die Ähnlichkeitsanalyse von Trajektorien eingesetzt. Hierbei ist darauf zu achten, dass die Echtzeitfähigkeit des Systems nicht gestört wird. Es werden bestimmte in Abschnitt 3.6 vorgestellte Bedingungen und Einschränkungen genutzt, um eine Echtzeitfähigkeit zu garantieren.

Funktionale Anforderung

Die funktionalen Anforderungen beschreiben die Funktionen, die das visuelle Spracherkennungssystem erfüllen sollen. Jede funktionale Anforderung beginnt mit FA (für funktionale Anforderung) und anschließende Nummerierung. Zu jeder FA werden Akteur, Ereignisse, Vor- und Nachbedingungen und das Zeitverhalten beschrieben. Vor- und Nachbedingungen gelten je FA und müssen erfüllt sein/werden. Das Zeitverhalten beschreibt in was für einer Zeit das jeweilige FA bearbeitet werden muss. Die FA werden im Kapitel 4 erfüllt bzw. realisiert.

FA1 - Wiedererkennung einer Äußerung

- **Akteur:** Benutzer, visuelle Spracherkennungssystem
- **Ereignisse:** 1. Benutzer schaut still in die Kamera, 2. Benutzer tätigt eine Äußerung, 3. Benutzer schaut still in die Kamera
- **Vorbedingungen:** Gesicht frontal zur Kamera gerichtet
- **Nachbedingungen:** Output der Befehle mit zugehörigen Wiedererkennungsraten
- **Zeitverhalten:** Output unmittelbar nach der Äußerung

FA2 - Off-line Wiedererkennung: Äußerung aus Datei laden

- **Akteur:** Benutzer, Dateisystem, visuelle Spracherkennungssystem
- **Ereignisse:** 1. Benutzer gibt den Pfad zur Datei der Äußerung, welche die Trajektorie beinhaltet, an, 2. Benutzer klickt auf den Button „Utter“
- **Vorbedingungen:** Gültige (Datei vorhanden, Datei beinhaltet Trajektorie) Datei, Hacken gesetzt bei „Load utterance file“
- **Nachbedingungen:** Output der Befehle mit zugehörigen Wiedererkennungsraten
- **Zeitverhalten:** Output unmittelbar nach Betätigung des Buttons

FA3 - Aufzeichnung von Trajektorien ins Trainingsset

- **Akteur:** Benutzer, Datenbank, visuelle Spracherkennungssystem

- **Ereignisse:** 1. Benutzer gibt Kommandoname an, 2. Wählt gewünschte Merkmale (Fläche, Seitenverhältnis) für die Aufzeichnung, 3. Drückt auf den Button „Record Trajectory“, 4. Ausführung von FA1, 5a. Fürs speichern der Trajektorie Button „Save“, 5b. Fürs nicht speichern Button „Abort“
- **Vorbedingungen:** Datenbank gestartet
- **Nachbedingungen:** a: Speichern der Trajektorie in die Datenbank

FA4 - Änderung der Datenbank für Trajektorien-Trainingsset

- **Akteur:** Benutzer, Datenbank
- **Ereignisse:** 1. Benutzer ändert Name der Datenbank
- **Vorbedingungen:** Datenbank gestartet
- **Nachbedingungen:** Output des aktuellen Trainingsset

FA5 - Aufzeichnung der Ergebnisse der Wiedererkennung

- **Akteur:** Benutzer, LipsRecRecorder
- **Ereignisse:** 1. Benutzer gibt Dateiname ein, 2. Drückt beim LipsRecRecorder auf den Button „Record“, 3. Ausführung von FA1, 4a. Fürs speichern der Ergebnisse der Wiedererkennung Button „Save“, 4b. Fürs nicht speichern Button „Decline“
- **Vorbedingungen:** Gültiger Dateiname
- **Nachbedingungen:** Erzeugung einer Datei mit Werten

FA6 - Aufzeichnung einer Äußerung in eine Datei

- **Akteur:** Benutzer, LipsRecRecorder
- **Ereignisse:** 1. Benutzer gibt Dateiname ein, 2. Hacken setzen bei „Save utterance“, 3. Drückt beim LipsRecRecorder auf den Button „Record“, 4. Ausführung von FA1, 5a. Fürs speichern der Äußerung Button „Save“, 5b. Fürs nicht speichern Button „Decline“
- **Vorbedingungen:** Gültiger Dateiname
- **Nachbedingungen:** Erzeugung einer Datei mit der Äußerung (Trajektorie)

FA7 - Auswahl der Methode für die Ähnlichkeitsmessung

- **Akteur:** Benutzer, visuelles Spracherkennungssystem
- **Ereignisse:** 1a. Auswahl DTW, 1b. Auswahl euklidische Distanz

FA8 - Konfiguration der Parameter für das DTW

- **Akteur:** Benutzer, visuelles Spracherkennungssystem

- **Ereignisse:** -Auswahl der Distanzfunktion, -Auswahl des Schritt-Musters, -Aktivierung der Fensterbeschränkung, -Auswahl der Größe des Fensters für die Fensterbeschränkung, -Aktivierung/Deaktivierung des gewichteten DTW (siehe Abschnitt 4.5.2)
- **Nachbedingungen:** Speichern der Parameter

FA9 - Konfiguration der Paramter für das gewichtete DTW (siehe Abschnitt 4.5.2)

- **Akteur:** Benutzer, visuelles Spracherkennungssystem
- **Ereignisse:** 1. Einstellung des Koeffizienten, 2. Einstellung der Gewichte je Kommando
- **Vorbedingungen:** Gewichtete DTW aktiviert
- **Nachbedingungen:** Speichern der Parameter

FA10 - Training: Generierung von Repräsentanten je Kommando (siehe Abschnitt 4.6.2)

- **Akteur:** Benutzer, visuelles Spracherkennungssystem
- **Ereignisse:** 1. Auswahl welches Kommando trainiert werden soll, 2. Auswahl wie viel Repräsentanten gefunden werden sollen, 3. Starte Training per Betätigung des Buttons „Cluster“
- **Vorbedingungen:** Datenbank gestartet
- **Nachbedingungen:** Speichern der Repräsentanten in die Datenbank
- **Zeitverhalten:** Nicht sofort, bis zu mehrere Minuten

FA11 - Einzeichnen der Lippenmerkmale

- **Akteur:** Benutzer, visuelles Spracherkennungssystem
- **Ereignisse:** 1. Auswahl der Lippenmerkmale in dem auf Lippensymbol geklickt wird
- **Nachbedingungen:** Einzeichnen der Lippenmerkmale in das Echtzeitkamerabild

3.9 Laborumgebung

In diesem Abschnitt wird die Laborumgebung präsentiert. Der Roboter, indem das visuelle Spracherkennungssystem laufen wird, die Kamera und das Betriebssystem, welches vom Roboter genutzt wird, werden vorgestellt.

Abbildung 3.37: Roboter: Robo

3.9.1 Roboter

Der Roboter besteht aus der mobilen Plattform SCITOS G5[1] von der Firma MetraLabs[2] und einem 5-DOF-Arm von der Firma SCHUNK[3]. Mithilfe des SCITOS G5 kann sich der Roboter bewegen und durch den Roboterarm der Firma SCHUNK besteht die Möglichkeit des Greifens. Am SCHUNK-Arm befindet sich eine Kinect[4], womit der Roboter seine Umwelt sensoriell erfassen kann. An der mobilen Plattform befinden sich zwei 2-D-Laserscanner (Leuze[5] und Hokuyo[6]), mit denen eine 2-D-Visualisierung der Umgebung erstellt werden kann.

3.9.2 Kinect V2

Die Kinect V2 (Abb. 3.38) wird in dieser Arbeit genutzt, um das Gesicht und die Lippen zu erfassen. Die Kinect ist ein Sensor für die Spielkonsole Xbox One von Microsoft und liefert verschiedene Komponenten für die Erfassung eines oder mehrerer Spieler bzw. Personen. Folgende Komponenten hat die Kinect:

[1] http://metralabs.com/index.php?option=com_content&view=article&id=70&Itemid=64
[2] http://metralabs.com
[3] http://www.de.schunk.com
[4] http://www.xbox.com/de-DE/Kinect
[5] http://www.leuze-electronic.de
[6] http://www.hokuyo-aut.jp

Abbildung 3.38: Kinect-Sensor
Quelle: Microsoft [2016]

1 3-D-Tiefensensoren: Die Tiefensensoren liefern dreidimensionale Tiefenwerte.
2 RGB-Kamera: Die RGB-Kamera (1920x1080 Auflösung) liefert eine Videoaufzeichnung.
3 Infrarotkamera: Mit der Infrarotkamera ist es beispielsweise möglich ein Bild im Dunkel zu erhalten.
3 Vier Mikrofone (3-D-Mikrofon): Eine Reihe von Mikrofonen, welche beispielsweise zur Spracherkennung oder zur Kommunikation benutzt werden können.

3.9.3 ROS - Robot Operating System

Um den Roboter ansteuern und mit ihm interagieren zu können, wird das spezielle Roboterbetriebssystem *ROS* (Robot Operating System) verwendet, welches eher ein Software-Framework darstellt[7]. *ROS* bietet ein Set von Software Bibliotheken und Tools, um Roboter Applikationen zu erstellen. Obwohl der Name von einem Betriebssystem spricht, ist *ROS* kein Betriebssystem im klassischen Sinne (Prozess- management und Scheduling), sondern es liefert ein strukturiertes Kommunikationslayer über das Host-Betriebssystem. Die Designziele werden wie folgt formuliert (Quigley u. a. [2009]):

- **Peer-to-peer**: Es übernehmen mehrere Rechner verschiedene Berechnungen, wobei rechenintensive Berechnungen auf High-power-offboard (nicht am Roboter) Maschinen durchgeführt werden.

- **Tool-basiert:** Mikro-Kernel-Design, sprich wichtige *ROS* Komponenten sind im Kernel, um mit der Komplexität von *ROS* umzugehen. Knoten (welche mögliche Applikationen darstellen) werden mit zusätzlichen Tools erstellt und ausgeführt.

- **Multi-lingual:** *ROS* will sprachenneutral sein, hierfür bietet *ROS* eine language-neutral interface definition language (IDL) an, welches verwendet werden kann, um zwischen Knoten zu kommunizieren. Die Knoten können durch verschiedene Programmiersprachen entwickelt werden. Mit IDL können bis zu 400 verschiedene Nachrichtentypen

[7] http://www.ros.org

definiert werden. Die Nachrichtentypen repräsentieren unterschiedliche Datenstrukturen.

- **Dünn:** *ROS* ist stark modular aufgebaut. Es werden Standalone-Bibliotheken verwendet. Ziel dabei ist es die Wiederverwendbarkeit zu steigern.
- Frei und Open-source

ROS-Architektur

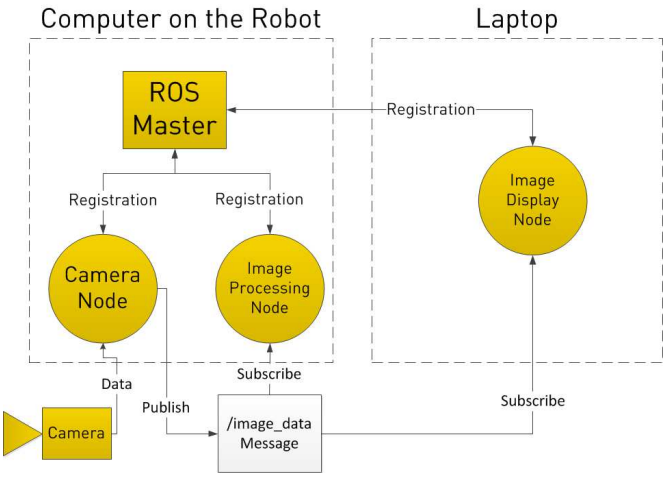


Abbildung 3.39: ROS-Knoten Struktur
Quelle: Robotics [2016]

Publish-Subscribe-Konzept

ROS nutzt das Publish- und Subscribe-Konzept über Topics. In *ROS* gibt es sogenannte Knoten, die Publisher und/oder Subscriber sein können. Dadurch entsteht die Kommunikation zwischen den einzelnen Knoten in einem *ROS*-System. Die Knoten und ihre Topics werden untereinander mithilfe eines Masters vermittelt. Die Knoten können verschiedene Aufgaben (Berechnungen, Ausführungen, Sensoren, Aktuatoren, etc.) übernehmen. Die Kommunikation zwischen den Knoten erfolgt über Topics. Ist ein Knoten A an einem Topic C von Knoten B und seine Nachrichten interessiert, so wird ein Subscribe auf Topic C vollzogen. Publisht Knoten B eine neue Nachricht auf das Topic C, so reagiert Knoten A automatisch auf die neu angekommene Nachricht. Knoten A stellt eine Methode zur Verfügung, die automatisch aufgerufen wird, falls eine neue Nachricht über das Topic C gesendet wird.
Mithilfe der IDL (3.9.3) wird die Art der Nachrichten definiert. Eine Nachricht kann aus

verschiedenen Typen zusammengestellt werden. Das Serialisieren und Deserialisieren wird von *ROS* übernommen. Es erfolgt eine Erläuterung des unteren Codeausschnittes.

```
1 Publisher cmd = n.advertise<geometry_msgs::Twist>("/cmd/sci", 3);
2 geometry_msgs::TwistPtr velocity_msg;
3 velocity_msg(new geometry_msgs::Twist);
4 velocity_msg->linear.x = x;
5 velocity_msg->linear.y = y;
6 velocity_msg->linear.z = z;
7 cmd.publish(velocity_msg);
```

Listing 3.1: Publish einer Nachricht vom Typ Twist

Im Codeausschnitt 3.1 ist zu sehen wie ein Publisher erzeugt wird. Die Erzeugung erfolgt mithilfe der *ROS*-Methode „advertise“ (Zeile 1). Die Kommunikation erfolgt über den Topic */cmd/sci*, der vom Publisher im *ROS*-Netzwerk registriert wird. Der zweite Parameter definiert die Größe der Queue, sprich es können maximal drei Nachrichten gespeichert werden. Werden Nachrichten zu schnell veröffentlicht, so wird die älteste Nachricht aus der Queue ersetzt. Die Nachrichten werden vom Typ *geometry_msgs::Twist* gesendet. Die Nachricht besteht nicht aus einem einfachen Typ, dies wird durch die IDL ermöglicht. In diesem Beispiel wird ein Vektor benutzt, der eine Geschwindigkeit definiert (Zeile 4-6). In Zeile 7 wird die zusammengesetzte Nachricht versendet/veröffentlicht.

```
1 ros::Subscriber sub = n.subscribe("/recognizer/output", 2, cmdParse);
2 ...
3 void cmdParse(const std_msgs::String& cmd) { ... }
```

Listing 3.2: Subscribe einer Nachricht vom Typ String

Im Codeausschnitt 3.2 wird ein Topic */recognizer/output* subscribt, dabei wird die Methode *cmdParse* registriert, sprich treffen neue Nachrichten über diesen Topic so wird automatisch die Methode *cmdParse* aufgerufen. Der zweite Parameter bestimmt die Größe der Queue. Werden Nachrichten zu schnell empfangen, so wird immer die älteste Nachricht in der Queue ersetzt.

Service-Konzept

Ein weiterer Mechanismus den *ROS* anbietet sind Services. Ein Service ist eine weitere Möglichkeit für Knoten miteinander zu kommunizieren. Hierbei wird aktiv von einem Knoten eine Kommunikationsanfrage gestellt. Ein Knoten schickt ein *request* an einen anderen Knoten, der mit einem *response* antwortet. Service können beispielsweise genutzt werden um Knoten zu konfigurieren.

```
1 bool add(beginner_tutorials::AddTwoInts::Request &req,
2 beginner_tutorials::AddTwoInts::Response &res) {
3       res.sum = req.a + req.b;
4       return true;
5 }
6 ...
7 ros::ServiceServer service = n.advertiseService("add_two_ints", add);
```

Listing 3.3: Erstellen eines Service

Im Codeausschnitt 3.3 wird ein Service erstellt, um die Summe von zwei Integer-Werten zu addieren. In Zeile 7 erfolgt die Registrierung des Services im *ROS*-Kommunikationslayer.

```
1 ros::ServiceClient client =
2 n.serviceClient<beginner_tutorials::AddTwoInts>("add_two_ints");
3 beginner_tutorials::AddTwoInts srv;
4 srv.request.a = atoll(argv[1]);
5 srv.request.b = atoll(argv[2]);
6 client.call(srv);
```

Listing 3.4: Aufruf eines Service

Im Codeausschnitt 3.4 erfolgt ein Aufruf des Service *add_two_ints*. Hierbei werden die Request-Parameter gesetzt und in Zeile 6 erfolgt der Aufruf des Service mit Übergabe der Request-Parameter.

4 Lippenlesen

In diesem Kapitel wird die Realisierung des visuellen Spracherkennungssystems präsentiert. Es wird zunächst der Design des Prototyps vorgestellt. Nach der Präsentation des Designs werden die einzelnen Schritte der Prozesskette (Abb. 2.1) vorgestellt.

4.1 Design des visuellen Spracherkennungssystems

Es erfolgt eine Übersicht der vorhandenen Komponenten (der Architektur) und deren Relationen zueinander. Mithilfe eines Sequenzdiagramms werden die Kommunikation zwischen den Komponenten und der Anwendungsverlauf des Systems gezeigt. Abschließend wird das Datenformat für die Trajektorien präsentiert.

4.1.1 Architektur

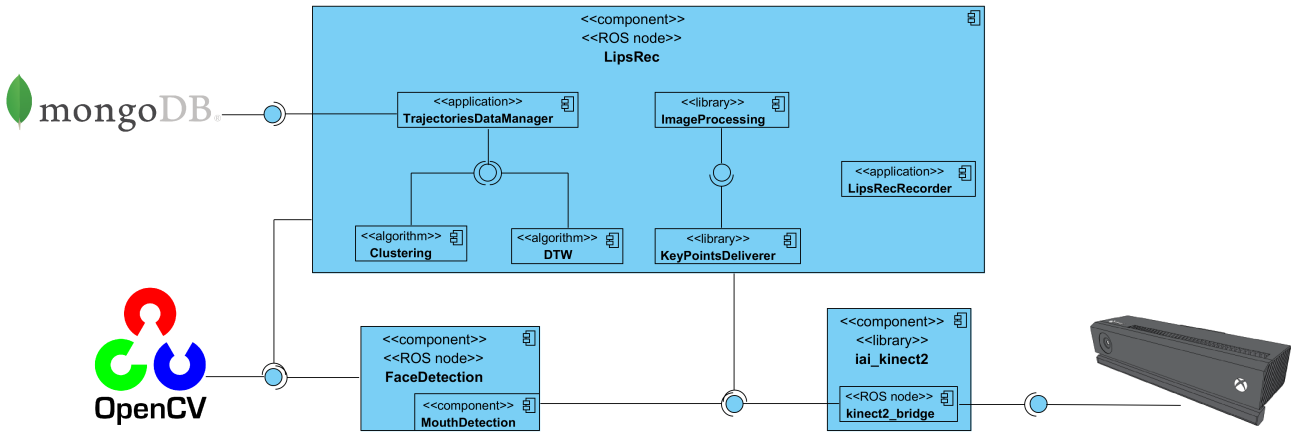


Abbildung 4.1: Architektur von LipsRec

In Abbildung 4.1 ist die Architektur des visuellen Spracherkennungssystems zu sehen. Die einzelnen Komponenten und deren Relationen sind abgebildet. Es folgt eine Erläuterung der Komponenten:

- **LipsRec:** Die Komponente „LipsRec“ ist eine *ROS*-Anwendung. Sie ist die Komponente, die die visuelle Spracherkennung durchführt. „LipsRec“ besteht aus mehreren Unterkomponenten. In der nächsten Aufzählung werden die Unterkomponenten und deren Aufgaben vorgestellt. „LipsRec“ dient als Schnittstelle für den Benutzer. Die GUI ist in Abbildung 4.3 zu sehen. Die einzelnen Funktionalitäten der GUI werden in der folgenden Aufzählung kurz vorgestellt:
  - **TrajectoriesDataManager:** Der „TrajectoriesDataManager“ (TDM) verwaltet die Trajektorien in einer Datenbank (MongoDB siehe Abschnitt 4.1.2). Er kann Trajektorien hinzufügen, entfernen und für Verarbeitungen aufrufen. In Abbildung 4.3 in **Punkt 1** ist das vorhandene Trajektorienset je Kommando zu sehen. Die GUI ist auf der übernächsten Seite zu sehen. Der TDM wird für das Anzeigen des Sets benutzt. In **Punkt 8** ist die Funktionalität zu sehen, um Trajektorien aufzunehmen und Äußerung aus einer Datei zu laden. Die funktionalen Anforderungen FA2 (Off-line Wiedererkennung), FA3 (Aufzeichnung von Trajektorien ins Trainingsset), FA4 (Änderung der Datenbank für Trajektorien-Trainingsset), werden durch diese Komponente erfüllt.
  - **ImageProcessing:** In „ImageProcessing“ sind verschiedene Bildverarbeitungsalgorithmen implementiert, die innerhalb von „LipsRec“ verwendet werden, um Bildvorverarbeitung durchzuführen. In der GUI 4.3 in **Punkt 2** sind einige Bildvorverarbeitungsverfahren zu sehen. In den weiteren Abschnitten wird näher auf die verschiedenen Verfahren eingegangen.
  - **LipsRecRecorder:** Mithilfe des „LipsRecRecorder“ ist es möglich die Ergebnisse bzw. die Wiedererkennungsraten einer Äußerung in eine Datei zu speichern. Zusätzlich kann die Äußerung selbst als Trajektorie gespeichert werden. Es wird der jeweilige Merkmalsvektor aufsteigend nach der Zeit in eine Textdatei gespeichert. In Abbildung 4.3 in **Punkt 3** ist der „LipsRecRecorder“ zu sehen. Die funktionalen Anforderungen FA5 (Aufzeichnung der Ergebnisse der Wiedererkennung) und FA6 (Aufzeichnung einer Äußerung in eine Datei) werden durch den „LipsRecRecorder“ erfüllt.
  - **DTW:** Durch die Komponente DTW wird der „Dynamic-time-warping“- Algorithmus zur Verfügung gestellt. Es können verschiedene Distanzfunktionen, Schritt-Muster und Fenstergrößen eingestellt werden. Zusätzlich kann eingestellt werden, ob ein gewichtetes DTW durchgeführt werden soll. In Abschnitt 4.5.2 wird auf diese Erweiterung eingegangen. In Abbildung 4.3 in **Punkt 4** sind die verschiedenen Ein-

stellungen zu DTW und in **Punkt 5** ist die Kostenmatrix und der „Warping-Pfad“ zur jeweiligen Äußerung und zum jeweiligen Merkmal zu sehen. Die funktionalen Anforderungen FA8 (Konfiguration der Parameter für das DTW) und FA9 (Konfiguration der Parameter für das gewichtete DTW) werden erfüllt.

- **KeyPointsDeliverer:**

Abbildung 4.2: Darstellungsformen der Lippenmerkmale

Der „KeyPointsDeliverer“ (KPD) nimmt das Lippenbild entgegen und führt mittels „ImageProcessing“ Bildvorverarbeitung aus. Es wird vom KPD ein Suchverfahren angewendet, um die Schlüsselpunkte des Mundes (siehe Abbildung 3.22 in Abschnitt 3.5.4) zu extrahieren. Das Verfahren wird in Abschnitt 4.4.3 vorgestellt. In der GUI 4.3 in **Punkt 6** sind die extrahierten Schlüsselpunkte zu sehen. Es können mehrere Darstellungsformen der Lippenmerkmale ausgewählt werden. In Abbildung 4.2 sind die drei möglichen Darstellungsformen zu sehen. Die funktionale Anforderung FA11 (Einzeichnen der Lippenmerkmale) wird erfüllt.

- **Clustering:** Die Äußerungen, die in der Datenbank als Trajektorien gespeichert sind, dienen als Trainingsset. Die Komponente „Clustering“ findet für die jeweiligen Kommandos je k-Repräsentant(en). Das Verfahren und der Algorithmus werden in Abschnitt 4.6.2 vorgestellt. In Abbildung 4.3 in **Punkt 7** ist die Realisierung der Komponente zu sehen. Die funktionale Anforderung FA10 (Training: Generierung von Repräsentanten je Kommando) wird erfüllt.

- **FaceDetection:** Die Komponente „FaceDetection“ ist eine Entwicklung innerhalb dieser Arbeit. Sie ist, genau wie „LipsRec“, eine *ROS*-Anwendung. „FaceDetection“ liefert die *ROI*-Koordinaten des Gesichtes und Mundes. Für die Gesichtserkennung wird OpenCV eingesetzt. In Abschnitt 4.2 werden die Realisierung und auftretende Probleme vorgestellt.

- **iai_kinect2:** Um die Kinect innerhalb in *ROS* verwendet zu können wird die Komponente „iai_kinect2“ (IAIK) verwendet. IAIK ist eine Sammlung von Werkzeugen und Bibliotheken für *ROS*, um die Kinect V2 zu kalibrieren, ihre Tiefenbilder auf der Grafikkarte zu verarbeiten und ihre Bilder anzuzeigen. Es bietet außerdem eine Unterstützung für *ROS*

an, indem es eine Brücke/Schnittstelle zwischen dem Kinect V2 Treiber „libfreenect2“ (Echtler u. a. [2016]) und *ROS* schafft (Wiedemeyer [2014 – 2015]).

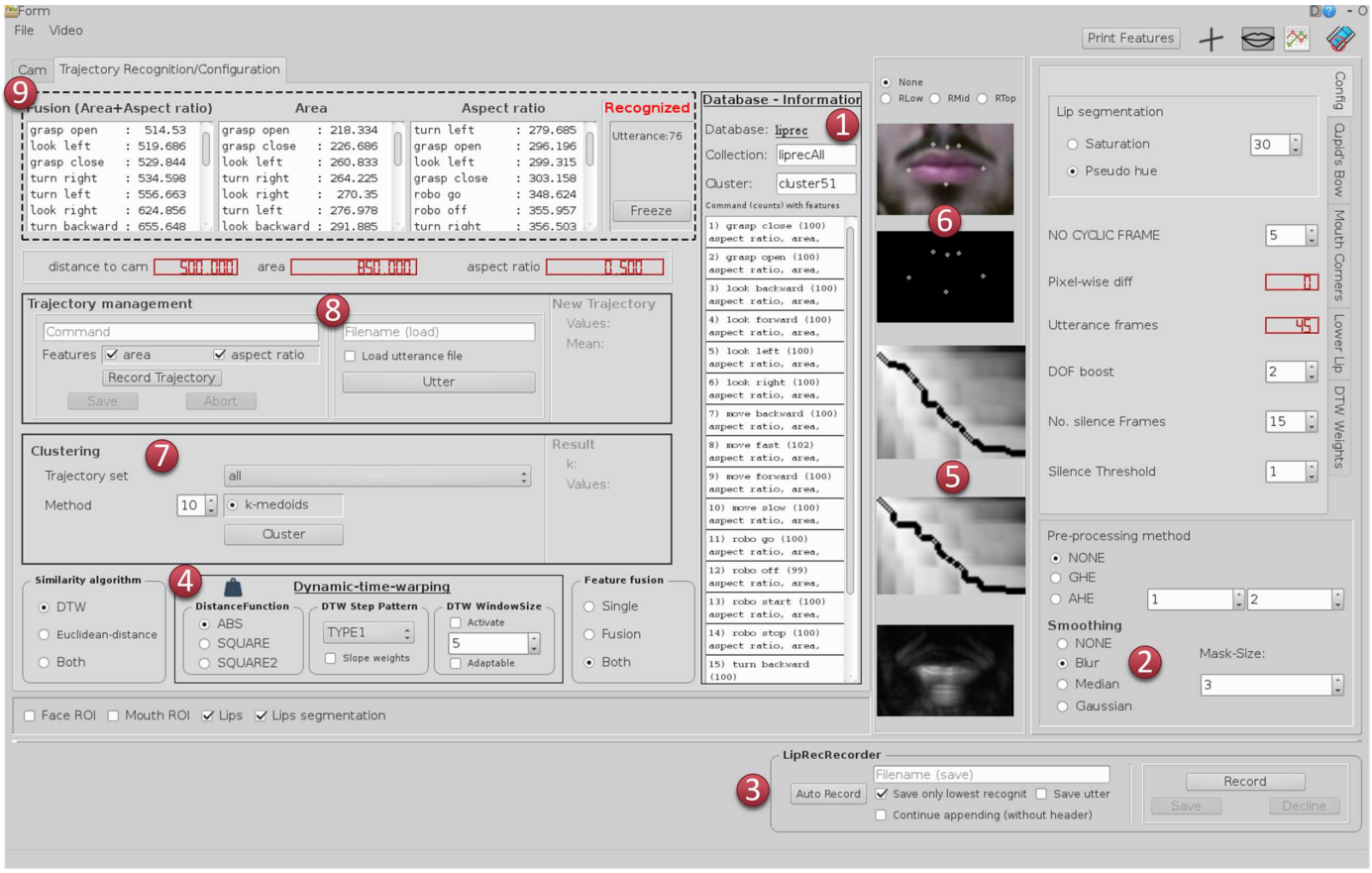


Abbildung 4.3: Benutzeroberfläche (GUI) von LipsRec

4.1.2 Datenformat

In diesem Abschnitt wird das Datenformat präsentiert in der die Trajektorien gespeichert werden. Es wird ebenfalls kurz die verwendete Datenbank vorgestellt.
Die Trajektoriendaten werden in die NoSQL-Datenbank (NoSQL steht für „Not only SQL“) MongoDB gespeichert. NoSQL bedeutet, dass die Daten nicht nur in Form von traditionellen Tabellen (relationales Datenbankmodell) vorliegen müssen (Cattell [2011]). MongoDB nutzt einen dokumentenorientierten Ansatz, d.h. es ist kein Schema, wie in relationalen Datenbanken, notwendig, um die Daten zu speichern. Es werden die Daten also nicht in Zeilen und Spalten gespeichert. Die Daten werden als „Key-value“- Paar gespeichert (MongoDB [2016]).

Die „Key-value“- Paar werden umgewandelt und im XML-Format gespeichert. XML steht für „Extensible Markup Language“ (erweiterbare Auszeichnungssprache) und ist ein text-basiertes Format. Es wird eingesetzt, um Daten zwischen Computersystemen auszutauschen (Bray u. a. [2008]). Ein Datum für eine Trajektorie sieht wie folgt aus:

```
<trajectory>
      <command> robo start </command>
      <feature> aspect ratio </feature>
      <values>
            <val id = 1> 0.5296 </val>
            <val id = 2> 0.5866 </val>
            <val id = 3> 0.5686 </val>
            ...
            <val id = n> 0.1556 </val>
      </values>
</trajectory>
```

Listing 4.1: Trajektorie im XML-Format

Im Code 4.1 sind die Werte und weitere Informationen einer Trajektorie dargestellt, die das Kommando „robo start“ repräsentiert. Diese Trajektorie enthält die Werte vom Merkmalsvektor aufsteigend nach der Zeit. In diesem Fall ist das Merkmal das Seitenverhältnis der Lippen. Die Darstellungsform in 4.1 ist das XML-Format.

4.1.3 Sequenzdiagramm

Im folgenden Abschnitt wird das Sequenzdiagramm vorgestellt, welches die Kommunikation der einzelnen Komponenten beschreibt. Das Sequenzdiagramm in Abbildung 4.4 + 4.5 ist aus Übersichtlichkeit in zwei Bilder geteilt. Es wird nicht jede einzelne Funktionalität des visuellen Spracherkennungssystems vorgestellt, sondern das Sequenzdiagramm beschreibt die Hauptfunktionalität (Wiedererkennung einer Äußerung).

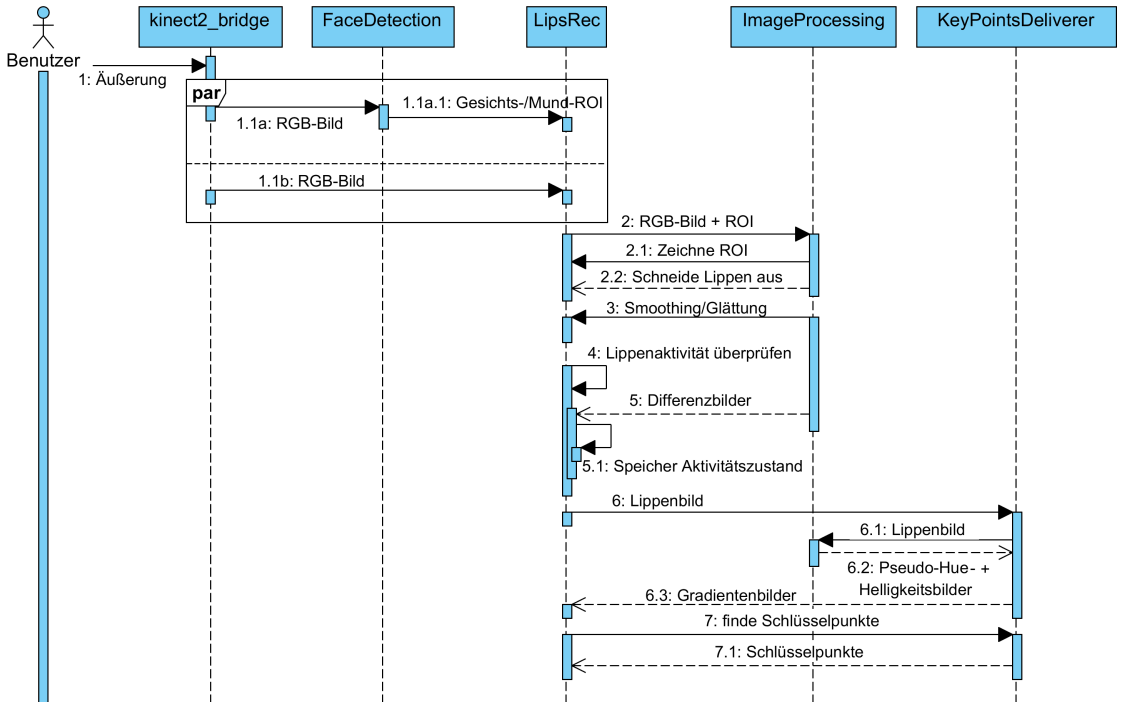


Abbildung 4.4: Teil 1: Sequenzdiagramm für eine Äußerung

- **1: Äußerung:** Der Benutzer tätigt eine Äußerung.
- **1.1a: RGB-Bild:** Das RGB-Bild wird von der Kinect geliefert. Es wird hierfür die *ROS*-Anwendung „kinect2_bridge“ genutzt. Die Komponente „FaceDetection“ nimmt das RGB-Bild entgegen.
- **1.1a.1: Gesichts-/Mund-ROI:** „FaceDetection“ ermittelt die Gesichts- und Mund-*ROI* und sendet diese weiter an das visuelle Spracherkennungssystem „LipsRec“.
- **1.1b: RGB-Bild:** „par“ steht für parallel, d.h. das RGB-Bild der Kinect wird gleichzeitig an „LipsRec“ gesendet.
- **2: RGB-Bild + ROI:** „LipsRec“ liefert Beides an „ImageProcessing“ als Parameter aus.
- **2.1: Zeichne ROI:** Mittels „ImageProcessing“ wird die *ROI* in das RGB-Bild gezeichnet.
- **2.2: Schneide Lippen aus:** Aus dem RGB-Bild werden die Lippen bzw. die Mund-*ROI* ausgeschnitten.

- **3: Smoothing/Glättung:** Das Lippenbild wird vor verarbeitet, indem es geglättet wird. Auf das Verfahren wird in 4.4.2 eingegangen.
- **4: Lippenaktivität überprüfen:** Das visuelle Spracherkennungssystem startet den Vorgang zur Überprüfung, ob eine Lippenaktivität vorliegt. Das Verfahren wird in 4.3.1 beschrieben.
- **5: Differenzbilder:** Für die Überprüfung werden Differenzbilder benötigt. Diese werden mithilfe von „ImageProcessing“ generiert. Ein Differenzbild wird generiert aus den Unterschieden der Grauwerte aus zwei nacheinander folgenden Frames. In Abschnitt 4.3.1 wird darauf eingegangen.
- **5.1: Speicher Aktivitätszustand:** „LipsRec“ speichert sich den Aktivitätszustand.
- **6: Lippenbild:** Das Lippenbild wird der Komponente „KeyPointsDeliverer“ (KPD) übergeben.
- **6.1: Lippenbild:** KPD nutzt ebenfalls „ImageProcessing“.
- **6.2: Pseudo-Hue- + Helligkeitsbilder:** KPD erhält von „ImageProcessing“ zwei Lippenbilder, die verschiedene Darstellungsformen repräsentieren. In Abschnitt 4.4.2 wird auf das Verfahren eingegangen.
- **6.3: Gradientenbilder:** Es werden aus den zwei Lippenbilder Gradientenbilder generiert, die bestimmte Eigenschaften der Lippen zum Vorschein bringen. Das Verfahren und die Eigenschaften werden ebenfalls in Abschnitt 4.4.2 vorgestellt.
- **7: finde Schlüsselpunkte:** „LipsRec“ fordert KPD auf die Schlüsselpunkte zu suchen.
- **7.1: Schlüsselpunkte:** Durch ein Suchverfahren werden die Schlüsselpunkte der Lippen im Bild gefunden. Das Suchverfahren wird in Abschnitt 4.4.3 präsentiert.

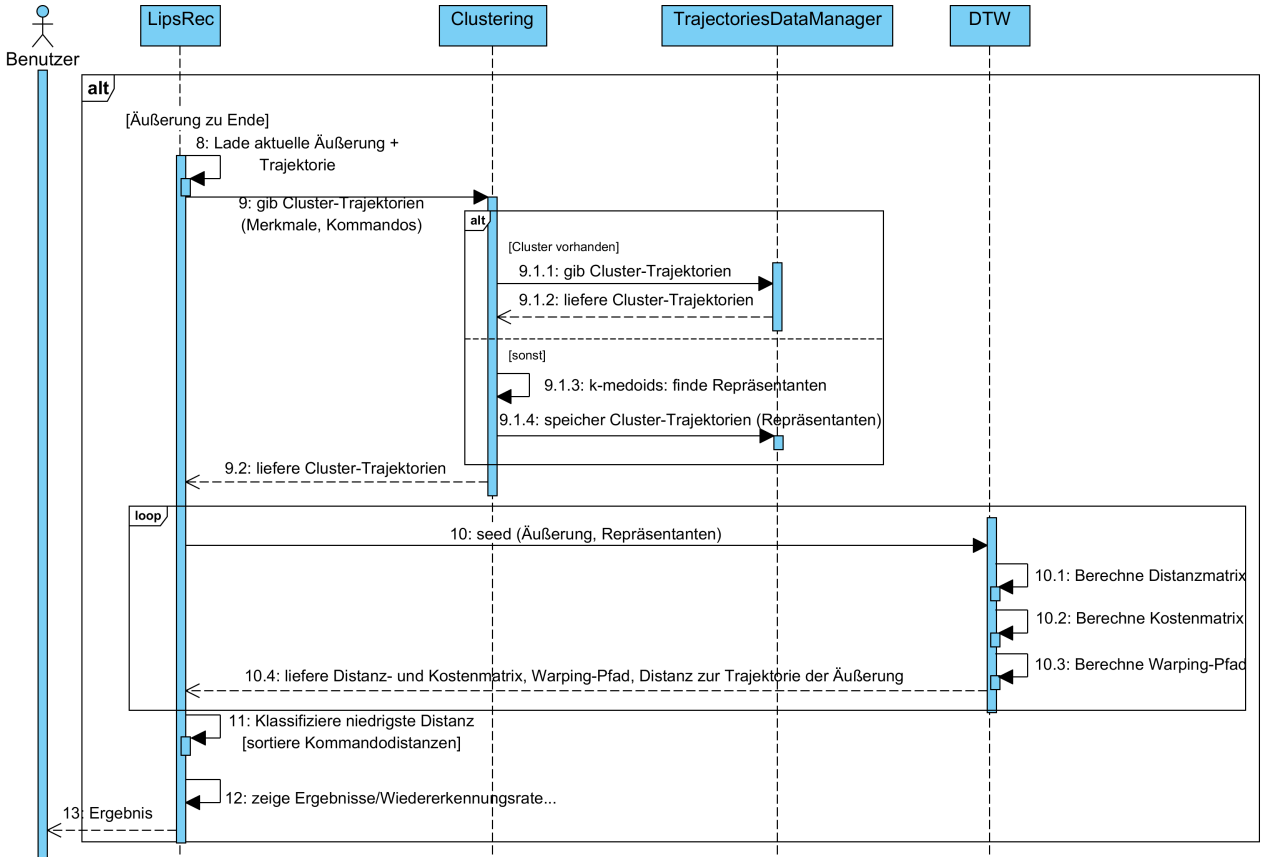


Abbildung 4.5: Teil 2: Sequenzdiagramm für eine Äußerung

- **8: Lade aktuelle Äußerung + Trajektorie:** „alt“ steht für Alternative und führt eine Fallunterscheidung ein. Die folgenden Sequenzen werden nur ausgeführt wenn die Äußerung zu Ende ist. Die aktuelle Äußerung ist gespeichert, genauso wie die Merkmalsvektoren (Trajektorien). Diese werden geladen.
- **9: gib Cluster-Trajektorien (Merkmale, Kommandos):** Um die Trajektorien zu klassifizieren, sind sogenannte Cluster-Trajektorien oder Repräsentanten notwendig. Diese repräsentieren ein Trajektorienset, welche ein Kommando darstellen. „LipsRec“ übergibt der Komponente „Clustering“ das Merkmal und die Kommandos, für die die Repräsentanten gefunden werden sollen.

- **9.1.1: gib Cluster-Trajektorien:** Es gibt eine Fallunterscheidung: Wurden die Repräsentanten bereits gefunden, so werden diese von der Komponente „TrajectoriesDataManager“ (TDM) aus der Datenbank geliefert.
- **9.1.2: liefere Cluster-Trajektorien:** TDM liefert die gewünschten Cluster-Trajektorien.
- **9.1.3: k-medoids: finde Repräsentanten:** Wurden noch keine Repräsentanten gefunden, führt „Clustering“ eine Repräsentantensuche (Clustering) aus. Der Algorithmus, der dafür eingesetzt wird, lautet k-medoids. In Abschnitt 4.6.2 werden das Clustering, also die Repräsentantensuche und der Algorithmus vorgestellt.
- **9.1.4: speicher Cluster-Trajektorien (Repräsentanten):** In TDM werden die neu gefundenen Repräsentanten gespeichert.
- **9.2: liefere Cluster-Trajektorien:** Die Repräsentanten werden ausgeliefert.
- **10: seed (Äußerung, Repräsentanten):** „loop“ leitet eine Schleife ein, denn es wird für jeden Repräsentanten der Kommandos die Distanz zur getätigten Äußerung gemessen. Hierfür werden die Äußerung und die Repräsentanten an DTW übergeben.
- **10.1: Berechne Distanzmatrix:** DTW berechnet, wie in 3.6 beschrieben, die Distanzmatrix.
- **10.2: Berechne Kostenmatrix:** DTW berechnet die Kostenmatrix.
- **10.3: Berechne Warping-Pfad:** Der „Warping-Pfad“ wird berechnet.
- **10.4: liefere DTW-Daten:** Die berechneten Daten werden an „LipsRec“ geliefert.
- **11: Klassifiziere niedrigste Distanz:** „LipsRec“ klassifiziert welcher Repräsentant zur Äußerung die geringste Distanz hat, hier werden die Distanzen je Kommando aufsteigend sortiert.
- **12: zeige Ergebnisse/Wiedererkennungsraten in GUI:** Die Wiedererkennungsraten werden in der GUI dargestellt. Siehe **Punkt 9** in der GUI (Abb. 4.3).
- **13: Ergebnis:** Das Ergebnis wird vom Benutzer wahrgenommen.

4.2 Lokalisierung des Gesichts

In der Prozesskette in Abbildung 2.1 ist der erste Prozessschritt die Gesichtslokalisierung. In der Analyse (3.5.1) wird beschrieben, dass die Gesichtserkennung von OpenCV benutzt wird.

```
1  const char* cascade_name_f = "haarcascade_frontalface_default.xml";
2  cascade_face = (CvHaarClassifierCascade*)cvLoad(cascade_name_f, 0, 0, 0);
3
4  void imageCallback(const sensor_msgs::ImageConstPtr& msg){
5  ...
6  CvSeq *faces = cvHaarDetectObjects(&iplImg, cascade_face, storage,
7  2.0, 3, 0, cvSize( 50,50 ));
8
9  CvRect *e = (CvRect*)cvGetSeqElem(faces, 0);
10 ...
11 }
```

Listing 4.2: OpenCV Haar-Cascade - Gesichtserkennung

In Listing 4.2 ist die Ausführung der Gesichtserkennung mit Hilfe von OpenCV zu sehen. In Zeile 2 wird der antrainierte Klassifizierer von OpenCV für das Gesicht geladen. Antreffende Frames werden in der Methode „imageCallback“ (Zeile 4) bearbeitet. In Zeile 6 werden die Gesichter in den Frames erkannt. Die erkannten Gesichter werden als *ROI* repräsentiert. In Zeile 9 wird beispielsweise auf das erste erkannte Gesicht zugegriffen. Der Typ „CvRect“ repräsentiert die *ROI* und beinhaltet die x- und y-Koordinaten, sowie die Breite und Höhe der *ROI*. Die Komponente FaceDetection sendet die *ROI* über eine *ROS*-Nachricht an alle anderen *ROS*-Knoten, die sich für die *ROI* interessieren.

4.2.1 Beseitigung des Jitters

Die Gesichtserkennung von OpenCV ist zuverlässig in der Findung von Gesichtern, wenn die Gesichter frontal zur Kamera gerichtet sind. Je Frame wird ein neues *ROI* gesendet. Die *ROI* dürfen sich, wenn sich der Kopf nicht bewegt, nicht unterscheiden. Die gesendeten *ROIs* bei nicht veränderter Kopfposition liefern jedoch unterschiedliche *ROIs*. Die *ROIs* unterscheiden sich minimal. Durch leichte Grauwertveränderungung im Bild findet OpenCV ständig neue *ROIs*. Dies verursacht einen Jitter, d.h. die *ROIs* schwanken ständig von Frame zu Frame. Dies ist ein Problem für die Ähnlichkeitsanalyse von Trajektorien. Die Merkmale der Lippen werden von der Mund-*ROI* extrahiert. Um die Mund-*ROI* zu erhalten, wird die Gesichts-*ROI* benötigt. Dadurch, dass die Gesichts-*ROI* schwankt, schwanken die Werte der Merkmale ebenfalls. Es kommt zu fehlerhaften Trajektorien.

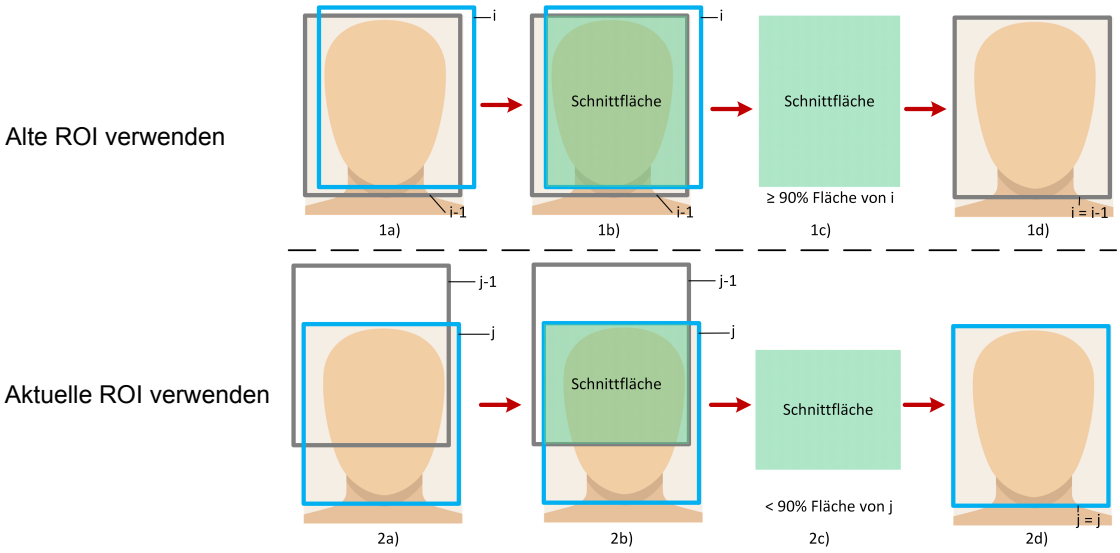


Abbildung 4.6: Beseitigung des Jitters wenn nötig

Für die Beseitigung des Jitters werden Rechtecke (*ROIs*) miteinander verglichen, indem deren Überschneidungen analysiert werden: Um den Jitter zu verhindert, wird die *ROI* des letzten Frames beibehalten, wenn die Schnittfläche des aktuellen und letzten *ROIs* größer gleich ist als ein vordefinierter Anteil der Fläche des aktuellen *ROIs*. Der vordefinierte Anteil ist der Schwellwert und wird in Prozent angegeben. Deckt also die Schnittfläche einen großen Anteil des aktuellen *ROIs*, kann die letzte *ROI* genutzt werden. Die Abbildung 4.6 veranschaulicht die Vorgehensweise zur Beseitigung des Jitters. Die Schritte von 1a bis 1d zeigen auf, wie die aktuelle ROI i nicht verwendet wird. Es wird die alte *ROI* $i-1$ verwendet, da die Schnittfläche 90% (Schwellwert) des aktuellen *ROIs* füllen. Bei den Schritten 2a bis 2d wird die aktuelle *ROI* j verwendet. Die Position hat sich gegenüber zum vorherigen Frame stark verändert. Die Schnittfläche ist kleiner als 90% der Fläche der aktuellen *ROIs*. Die alte *ROI* $j-1$ wird nicht verwendet.

4.3 Lokalisierung der Lippen

In der Analyse (3.5.2) wurden zwei Verfahren vorgestellt, mit denen es möglich ist die Mund-*ROI* anhand der Gesichts-*ROI* zu lokalisieren/berechnen.

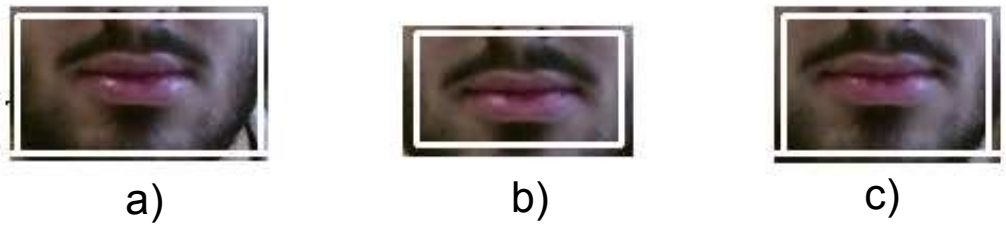


Abbildung 4.7: Vergleich der Verfahren zur Lippenlokalisierung

In dieser Arbeit werden beide Methoden kombiniert, um den Vorteil der schmalen (in der Breite) Mund-*ROI* von der Methode 2 und hohen Mund-*ROI* von der Methode 1 zu erhalten. Die Abbildung 4.7 vergleicht die drei Methoden. Abbildung c) ist der hybride Ansatz, also die Kombination von a) und b).

$$bf_M = x_G + b_G \tag{4.1}$$

$$b_M = (bf_M - (bf_M - x_G)/4) - x_M \tag{4.2}$$

$$x_M = x_G + (bf_M - x_G)/4 \tag{4.3}$$

$$S_H = h_G/3 \tag{4.4}$$

$$h_M = S_H \tag{4.5}$$

$$y_M = y_G + S_G * 2 \tag{4.6}$$

In den Gleichungen 4.1 bis 4.3 wird die Breite (b_M) und die x-Koordinate (x_M) der Mund-ROI, anhand des Algorithmus welches in der Arbeit Bacivarov u. a. [2008] vorgestellt wird, berechnet. In den Gleichungen 4.4 bis 4.6 wird die Höhe (h_M) und die y-Koordinate (y_M) der Mund-ROI, anhand des Algorithmus welches in der Arbeit Sujatha und M.Radhakrishnan [2013] vorgestellt wird, berechnet. S_H steht für ein Segment der Höhe. Zusammengefasst wird die Höhe von Methode 1 (Bild a) genutzt, da beim Sprechen sich die Lippen vertikal bewegen. Die Höhe von Methode 2 (Bild b) ist zu gering, jedoch wird aus der Methode 2 (Bild b) die Breite genommen, damit nicht Überflüssiges verarbeitet wird. In den Gleichungen 4.1 bis 4.6 wurden die Berechnungen, die in der Analyse vorgestellt werden, jeweils für die Höhe und Breite kombiniert.

4.3.1 Aktivierung einer Äußerung

Dieses visuelle Spracherkennungssystem erkennt automatisch ob eine Äußerung getätigt wird und wann diese endet. Für das Verfahren werden Differenzbilder genutzt. Ein Differenzbild beschreibt die Unterschiede der Farb-/Grauwerte von zwei aufeinanderfolgenden Frames.

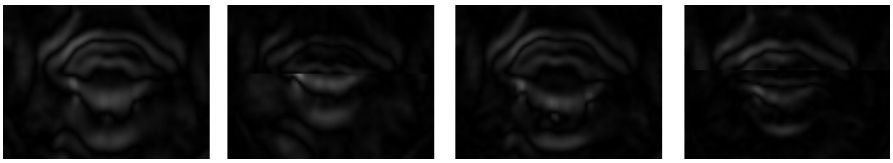


Abbildung 4.8: Verschiedene Differenzbilder während einer Äußerung

In Abbildung 4.8 sind einige Differenzbilder zu sehen.

$$Differenzbild(x, y) = |A(x, y) - B(x, y)| \tag{4.7}$$

Die Gleichung 4.7 berechnet ein Differenzbild. A und B sind aufeinanderfolgende Frames. Es wird die absolute Differenz der Grauwerte an Position (x, y) berechnet und in einem neuen Frame (Differenzbild) gespeichert. Um die Bearbeitungszeit zu reduzieren, wird ein Differenzbild aus den zuvor berechneten Mund-*ROI* erstellt und nicht vom kompletten Bild.

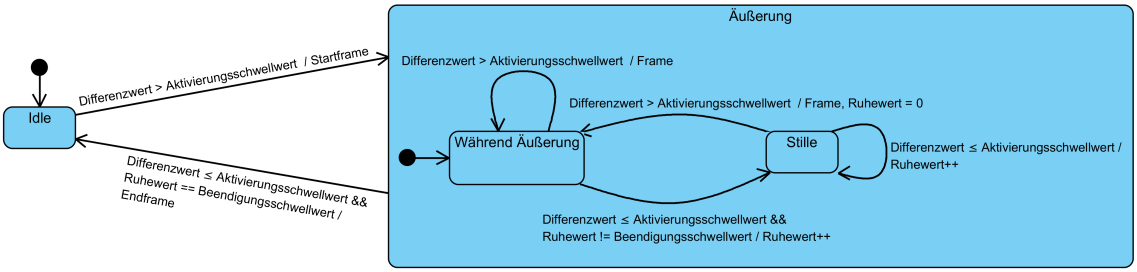


Abbildung 4.9: Zustandsautomat für die Aktivierung/Beendigung einer Äußerung

In Abbildung 4.9 ist der Zustandsautomat zu sehen, der bestimmt wann eine Äußerung anfängt und wann sie endet.

$$d = \sum_{i=0}^{N} \sum_{j=0}^{M} |A(i, j) - B(i, j)| \tag{4.8}$$

Die Gleichung 4.8 berechnet einen Wert, um die Differenz von zwei aufeinanderfolgenden Frames zu messen. N und M sind die Breite und Höhe der Frames von A und B. Der Messwert d wird für die Bestimmung einer Aktivierung/Beendigung einer Äußerung genutzt. Der Wert ist die Summe der absoluten Differenzen der Grauwerte von zwei aufeinanderfolgenden Frames. Es erfolgt eine Erläuterung des Zustandsautomaten:
Der Zustandsautomat ist am Anfang in Zustand „Idle“. Wenn eine Person anfängt eine Äußerung zu tätigen, erhöht sich der Differenzwert d (Gleichung 4.8). Überschreitet der Differenzwert

einen vorher festgelegten Schwellwert (Aktivierungsschwellwert), so wird dies als Anfang der Äußerung gesehen. Der Aktivierungsschwellwert basiert auf Erfahrung und liegt bei 0 bis 1. Es wird der Startframe gespeichert. Der Zustandsautomat befindet sich in Zustand „Äußerung“. Dieser Zustand hat ein Unterzustandsautomat. Der Unterzustandsautomat befindet sich jetzt im Zustand „Während Äußerung“ und bleibt dort wenn weiterhin Aktivität gemessen wird. Wird nun keine Aktivität gemessen, d.h. der Differenzwert ist kleiner gleich Aktivierungsschwellwert, so wechselt der Automat in den Zustand „Stille“. Der Ruhewert ist ein Zähler, der die Ruhephase in einer Äußerung angibt. Er wird hochgezählt wenn weiterhin in diesem Zustand keine Aktivität gemessen wird. Wird Aktivität gemessen, so wird zurück in den Zustand „Während Äußerung“ gewechselt und der Ruhewert wird zurückgesetzt. Wird jedoch weiterhin keine Aktivität gemessen und der Ruhewert erreicht den Beendigungsschwellwert so wird in den Zustand „Idle“ gewechselt und die Äußerung ist zu Ende. Der Endframe wird gespeichert. Der Beendigungsschwellwert wird höher gesetzt bei hohen FPS (Bilder pro Sekunde) als bei geringen FPS, da der Ruhewert je Frame betrachtet wird und wenn mehr Frames vorhanden sind, ist die Ruhephase ebenfalls länger (gemessen an den Frames).

4.4 Merkmalsextraktion

In diesem Abschnitt wird das Verfahren zur Extraktion von Lippenmerkmalen, welches in dieser Arbeit verwendet wird, vorgestellt. Es werden zunächst die sechs Schlüsselpunkte vorgestellt. Die Lippenbilder werden vor verarbeitet, um bestimmte Eigenschaften der Lippen zum Vorschein zu bringen. Das Verfahren wird präsentiert. Der Suchalgorithmus, um die Merkmale in den vor verarbeiteten Lippenbildern zu extrahieren, wird ebenfalls vorgestellt. Die extrahierten Merkmale müssen normalisiert werden, das Verfahren hierfür wird dargestellt.

4.4.1 Sechs Schlüsselpunkte

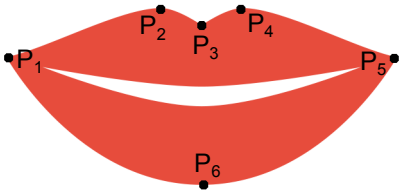


Abbildung 4.10: Sechs Schlüsselpunkte

In dieser Arbeit dienen die sechs Schlüsselpunkte (Abb. 4.10) als Grundlage für weitere Merkmale. Es ist erforderlich diese sechs Schlüsselpunkte zu extrahieren. Die Schlüsselpunkte basieren

auf anatomische Eigenschaften der Lippen. P_1 und P_5 stellen die Ecken der Lippen dar. Durch P_6 wird der unterste Punkt der Unterlippe beschrieben. Der sogenannte Kupidobogen ist der Bogen der Oberlippe, der durch die Punkte P_2, P_3 und P_4 beschrieben wird (Bangsawan u. a. [2015]).

4.4.2 Vorverarbeitung der Lippenbilder

In diesem Abschnitt werden die antreffenden Lippenbilder vor verarbeitet. Die verwendeten Verfahren werden vorgestellt.

Smoothing/Bildglättung

Das sogenannte „Smoothing“ (Bildglättung) wird eingesetzt, um ein Bild aus verschiedenen Gründen zu glätten. Es kann beispielsweise eingesetzt werden, um Details eines Bildes zu reduzieren, um die groben Strukturen/Muster zu erhalten. Ein weiterer Grund ist es Störungen/-Bildrauschen im Bild zu beseitigen. Störungen können auftreten durch eine geringe Auflösung des Bildes, unterschiedliche Lichtverhältnisse, Aufnahmeprobleme etc. (Farooque und Rohankar [2013]).

Abbildung 4.11: Glättung der Lippen

In dieser Arbeit werden die Algorithmen zur Bildglättung von OpenCV genutzt. Es wird die Methode „blur“ für das Glätten genutzt. Die Methode nimmt ein Lippenbild entgegen und legt eine Maske je Pixel auf. Für das Pixel wird ein neuer Farbwert berechnet, indem der Mittelwert aller Pixelfarbwerte, die unter der Maske sind, genommen wird. Die Maskengröße bestimmt wie viele Pixel für die Mittelwertberechnung mit einbezogen werden (itseez [2016]). In Abbildung 4.11 ist die Bildglättung eines Lippenbildes zu sehen.

Gradientenbilder

Nachdem das Lippenbild geglättet ist, werden sogenannte Gradientenbilder erzeugt. Gradienten werden unter anderem eingesetzt, um Kanten in einem Bild zu entdecken, dabei werden Veränderungen von

Abbildung 4.12: Gradientenverlauf
Quelle: Meisel [2016]

Verläufen von Grauwerten betrachtet (Abb. 4.12). Ein Gradient ist ein Vektor, der die Richtung und Stärke der Steigung von diesem Verlauf von Grauwerten angibt (Sharifi u. a. [2002]).
Wie in der Analyse in Abschnitt „Vergleichbare Projekte“ 3.4 vorgestellt wurde, werden bestimmte Eigenschaften der Lippen zum Vorschein gebracht, bevor die Gradientenbilder erzeugt werden.

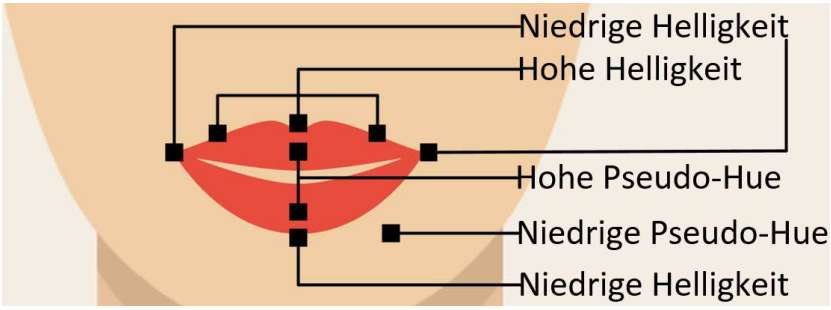


Abbildung 4.13: Verteilung von Psuedo-Hue- und Helligkeitswerten

Es wird die Eigenschaft genutzt, dass sich der Farbton der Lippen von der restlichen Gesichtshaut unterscheidet. Für die Vorverarbeitung wird das Farbmodell RGB umgewandelt. Es wird die Pseudo-Hue-Darstellung (P-H) genommen.

$$h(x,y) = \frac{R(x,y)}{G(x,y) + R(x,y)} \tag{4.9}$$

In Gleichung 4.9 ist die Berechnung des P-Hs zu sehen. x, y sind die Koordinaten des Pixels. R und G sind der Rot- und Grünwert des Pixels. In P-H-Darstellung haben die Lippen einen höheren Wert als die Gesichtshaut (Chetty und Wagner [2004]). Beleuchtung/Helligkeit ist ein wichtiger Faktor für die Unterscheidung der Lippen. Für gewöhnlich scheint die Beleuchtungsquelle von oben, somit ist der obere Teil der Oberlippe stärker/besser beleuchtet, als der untere Teil der Unterlippe. Der untere Teil der Unterlippe hat durch die Beleuchtungsquelle einen Schatten und der obere Teil der Oberlippe wird beleuchtet (Husain [2011]). Die Ecken der Lippen liegen in einem dunklen Bereich und haben einen hohen P-H-Wert (Alizadeh u. a. [2008]).

$$l(x,y) = 0.2126 * R(x,y) + 0.7152 * G(x,y) + 0.0722 * B(x,y) \tag{4.10}$$

Für die Berechnung der Helligkeit eines bestimmen Pixels wird die Gleichung 4.10 eingesetzt. Die Gleichung stammt aus der Hellempfindlichkeitskurve. Aus der wird abgeleitet, dass der Mensch auf grünes Licht 10 Mal empfindlicher als auf rotes oder blaues Licht reagiert. Blaues Licht wird am geringsten wahrgenommen (ScanDig [2016]). Dies spiegelt sich auch in der

Gleichung wider.

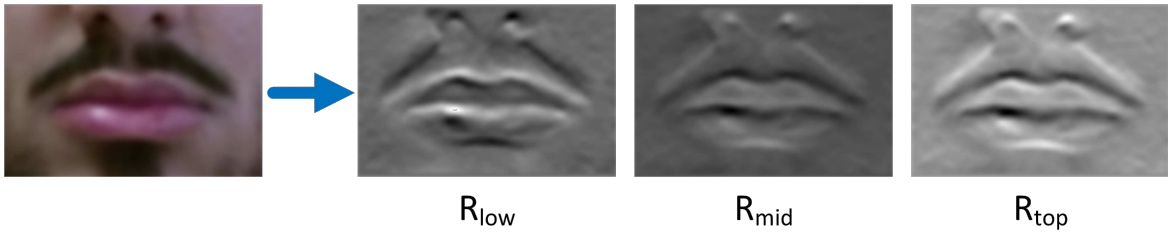


Abbildung 4.14: Die drei Gradientenbilder

Die Verteilung der P-H- und Helligkeitswerte im Gesicht und in der Lippenregion (Abb. 4.13) werden genutzt für die Extraktion der Schlüsselpunkte der Lippen. In Abbildung 4.13 sind diese Verteilungen aufgezeigt. Die Schlüsselpunkte der Lippen werden von den Gradientenbildern extrahiert. Zunächst werden die P-H- (Gleichung 4.9) und Helligkeitsbilder (Gleichung 4.10) der Lippen generiert. Beide Darstellungen werden im nächsten Schritt miteinander kombiniert.

$$R_{low}(x,y) = \nabla^y(h(x,y) + l(x,y)) \tag{4.11}$$

$$R_{mid}(x,y) = \nabla^y(h(x,y) - l(x,y)) * h(x,y) \tag{4.12}$$

$$R_{top}(x,y) = \nabla^y(h(x,y) - l(x,y)) \tag{4.13}$$

Die Gleichungen 4.11, 4.12 und 4.13 kombinieren die P-H- und Helligkeitsbilder der Lippen (Chetty und Wagner [2004] und Alizadeh u. a. [2008]). ∇ (Nabla) ist ein Operator, der die Gradientenbilder erzeugt. Der Exponent beschreibt die Richtung der Gradientenbilder (x oder y). Für die Erzeugung der Gradientenbilder wird der sogenannte Sobel-Filter (∇) eingesetzt. Der Sobel-Filter bringt Kanten, also Orte der größten Änderung, zum Vorschein. Mithilfe von sogenannten Faltungsmasken werden die Gradientenbilder erzeugt. Die Faltungsmasken werden auf das Ursprungsbild gelegt und der betrachtete Pixelwert (Mittelpunkt der Maske) wird entsprechend der Maske neu berechnet.

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{4.14}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{4.15}$$

In Gleichung 4.14 wird die x-Richtung und in Gleichung 4.15 die y-Richtung des Gradienten berechnet (Sharifi u. a. [2002]). Es werden drei neue Lippenbilder erzeugt R_{low}, R_{mid} und R_{top}.

Die jeweiligen Gradientenbilder haben die geforderten Werteverteilungen, die in Abbildung 4.13 beschrieben werden. R_{low} wird eingesetzt um Schlüsselpunkt P_6 zu finden. Es hat niedrige Werte für den unteren Teil der Unterlippe. R_{mid} wird eingesetzt um die Schlüsselpunkte P_1 und P_5 (Ecken der Lippen) zu finden. Es hat hohe Werte bei den Ecken der Lippen. R_{top} wird eingesetzt um die Schlüsselpunkte P_2, P_3 und P_4 zu finden. Es hat hohe Werte für die Schlüsselpunkte des Kupidobogens. In Abbildung 4.14 sind die drei Gradientenbilder zu sehen. Sie wurden aus den geglätteten Lippenbildern erzeugt. Die jeweiligen Lippenteile (Oberlippe, Unterlippe) werden stark zum Vorschein gebracht.

4.4.3 Realisierung

Im folgenden Abschnitt wird das Verfahren, um die Schlüsselpunkte in den Gradientenbildern zu finden, vorgestellt.

Point-of-interest-Suche

Ein „Point-of-interest“ (POI) ist ein Punkt der in Frage kommen kann, um ein Schlüsselpunkt zu sein. *POIs* werden in den Gradientenbildern gesucht. Das Gradientenbild R_{low} ist geeignet, um *POIs* im unteren Teil der Unterlippe zu finden. Mit dem Gradientenbild R_{mid} werden *POIs* im Bereich der Ecken der Lippen gesucht. Das Gradientenbild R_{top} ist geeignet, um *POIs* im oberen Teil der Oberlippe zu finden.
Um die *POIs* zu finden, werden sogenannte „Lines-of-interest“ (LOI) eingesetzt. *LOIs* werden eingesetzt um den Übergang/Verlauf der Grauwerte zu finden. Es werden also die Kanten in den jeweiligen Gradientenbildern gesucht. Ein *LOI* ist senkrecht und hat eine bestimmte Pixellänge. Um zu entscheiden, ob eine *POI* auf der Kontur liegt, werden folgende Gleichungen angewendet:

$$M = \text{LOI-Länge}/2 + 1 \tag{4.16}$$

$$a_{Pgs} = \sum_{k=1}^{M} R_{low}(i, j-k) + R_{low}(i, j+k) \tag{4.17}$$

$$d_{Pm} = \frac{a_{Pgs}}{\text{LOI-Länge}} \tag{4.18}$$

$$diff_{Pm} = \begin{cases} 100 - (\frac{R_{low}(i,j)*100}{d_{Pm}}) & d_{Pm} > 0 \\ 0 & sonst \end{cases} \tag{4.19}$$

In den Gleichungen 4.16 bis 4.19 wird das Gradientenbild R_{low} durchsucht. Das Verfahren ist

dasselbe für die anderen Gradientenbilder. Die Pixel des Gradientenbilds werden durchlaufen. Ein Pixelwert wird wie folgt betrachtet: Der Mittelpunkt der *LOI* wird auf den Pixelwert gelegt und die Pixelgrauwertsumme a_{Pgs}, also die Pixelwerte, die auf der Linie sind, wird berechnet. Gleichung 4.17 beschreibt diesen Vorgang. i und j sind die x- und y-Koordinaten der Grauwerte. Nachdem die Pixelgrauwertsumme a_{Pgs} berechnet wurde, wird für diese *LOI* das arithmetische Mittel d_{Pm} (Pm = Mittelwert der Pixelgrauwerte) über die Grauwerte berechnet (Gleichung 4.18). Um nun entscheiden zu können, ob das betrachtete Pixel ($R_{low}(i,j)$) eine *POI* ist, wird die prozentuale Differenz des Grauwertes $diff_{Pm}$ vom Pixel zum Mittelwert berechnet (Gleichung 4.19). Ist $diff_{Pm}$ größer als ein festgelegter Schwellwert, so ist das betrachtete Pixel ein *POI*. Der Schwellwert wird nach Erfahrung festgelegt (20% bis 40%). Die *POI* beschreibt einen Kantenpunkt.

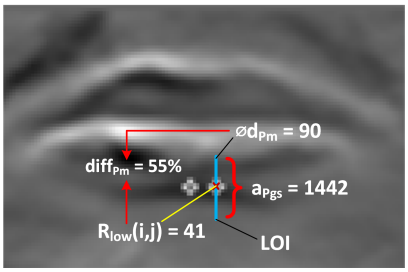


Abbildung 4.15: Einsatz einer *LOI*

In Abbildung 4.15 ist zu sehen, wie das Verfahren angewendet wird. Die *LOI* wird auf ein Pixel gelegt. In diesem Beispiel wurde bereits ein *POI* gefunden und ein wahrscheinlicher *POI* wird untersucht. Der untersuchte Grauwert ist ein *POI*. Der Wert zeigt, dass der Grauwert dunkler ist als der Mittelwert der *LOI*. Eine prozentuale Differenz von 55% ist gegeben, somit ist sie größer als der Schwellwert (20% bis 40%) und der Punkt ist ein *POI*.

Suchverfahren der Schlüsselpunkte

Um entscheiden zu können, ob ein *POI* ein Schlüsselpunkt der Lippe ist, wird ein an Bangsawan u. a. [2015] angelehntes abgeändertes Suchverfahren eingesetzt.

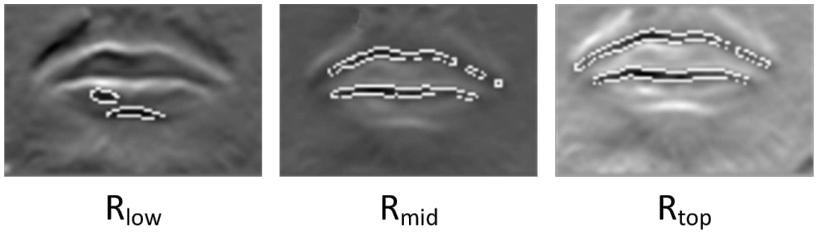


Abbildung 4.16: Aus *POI* generierte Konturen

Bevor das Suchverfahren eingesetzt wird, werden die *POIs* miteinander verbunden, um eine Kontur zu bilden (Abb. 4.16). Beim Suchverfahren werden sechs Schlüsselpunkte der Lippen gesucht. Die Konturpunkte werden auf bestimmte Eigenschaften untersucht. Es folgt eine Erläuterung des Verfahrens:

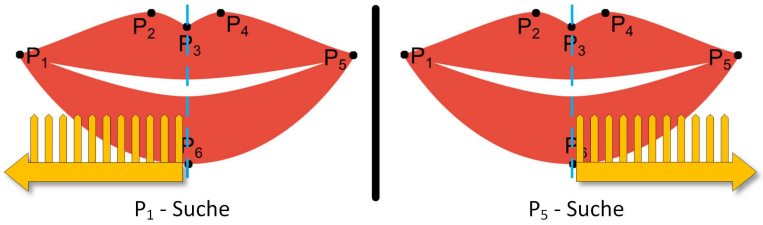


Abbildung 4.17: Suchverfahren für die Eckpunkte

Zunächst werden die Ecken der Lippen gesucht. Um die linke Ecke des Mundes (P_1) zu finden, wird das Gradientenbild R_{mid} genutzt. Wie bereits in 4.4.2 beschrieben, liefert das Bild die besten Informationen für die Ecken der Lippen (Ecken der Lippen sind dunkel). In Abbildung 4.17 ist der Suchverlauf dargestellt. Um beispielsweise den Schlüsselpunkt P_1 zu finden, wird die Suche in der Mitte des Lippenbildes gestartet. Der große Pfeil zeigt die Richtung an, in der das Bild gesucht wird. Die Spalten werden von rechts nach links nacheinander durchsucht, dabei werden die Spalten von unten nach oben durchlaufen. Die jeweiligen Grauwerte werden betrachtet. Die Konturen sind in Weiß gezeichnet. Es wird also nach weißen Konturpunkten gesucht. Der Schlüsselpunkt P_1 ist der am weitesten links außen liegende Konturpunkt (*POI*). Die Lippen werden von innen nach außen durchsucht, da es sein kann, dass sonst der äußerste Punkt der Kontur der Oberlippe fälschlicherweise als Eckpunkt klassifiziert wird. Dies liegt daran, da die Oberlippe weiter Außen verläuft als die Kontur für die Ecken. Dies ist in Bild R_{top} zu sehen. Die Oberlippenkontur liegt höher als die Kontur für die Ecken. Das wird genutzt, um zu entscheiden, ob es sich um die Oberlippenkontur oder Kontur für die Ecken handelt. Als Vergleich wird die Höhe des vorherigen Konturpunktes genutzt. Dies kann nur funktionieren, wenn von Innen nach Außen gesucht wird, da zuerst die Kontur für die

Ecken betrachtet wird. Das Suchverfahren für den Schlüsselpunkt P_5 verläuft genau andersrum.

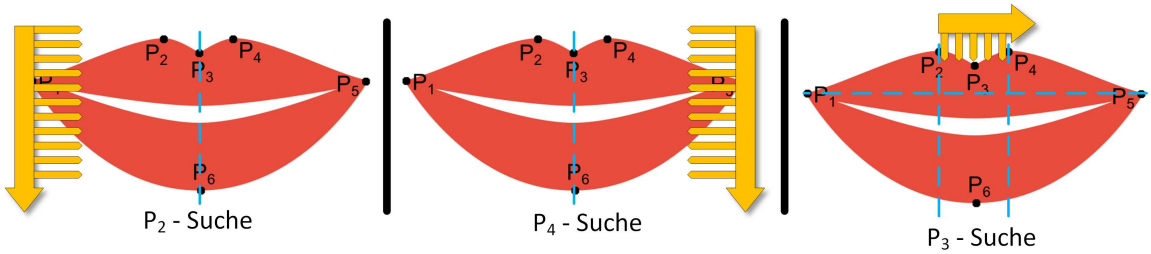


Abbildung 4.18: Suchverfahren für den Kupidobogen

Beim nächstes Schritt werden die Schlüsselpunkte des Kupidobogens gesucht (Abb. 4.18). Das Gradientenbild R_{top} wird verwendet (siehe Abschnitt 4.4.2). Die Schlüsselpunkte P_2 und P_4 sind die Punkte die am höchsten auf der Oberlippe liegen. Um den Schlüsselpunkt P_2 zu finden, wird das Bild zeilenweise von oben nach unten durchsucht. Die Zeilen werden von links nach rechts durchlaufen. Dabei wird das Lippenbild nur zur Hälfte durchlaufen. Der Grauwert der weiß ist und am höchsten liegt ist Punkt P_2. Das gleiche Verfahren, bloß umgekehrt, wird für die Suche von P_4 genutzt. Um den Schlüsselpunkt P_3 zu finden, werden die beiden zuvor gefunden Punkte P_2 und P_4 als Einschränkung des Suchraums genutzt. P_3 liegt vertieft zwischen P_2 und P_4. Es werden die Spalten von rechts nach links durchsucht, dabei werden sie von oben nach unten durchlaufen. Der Konturpunkt der am tiefsten liegt, ist der Schlüsselpunkte P_3. Eine weitere Einschränkung ist, dass die Suche in der Tiefe nicht bis zum unteren Bildende erfolgt, es werden die Höhen der Schlüsselpunkte P_1 und P_2 genutzt. Es werden sonst falsche Konturpunkte gewählt, nämlich die Konturen, die für die Suche der Ecken genutzt werden.

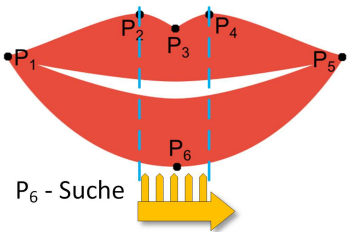


Abbildung 4.19: Suchverfahren für die tiefste Stelle

Der letzte Schritt ist die Suche des Schlüsselpunkts P_6 (Abb. 4.19). Hierfür wird das Gradientenbild R_{low} verwendet. Das Gradientenbild bringt den unteren Bereich der Unterlippe

zum Vorschein (siehe Abschnitt 4.4.2). Der Schlüsselpunkt P_6 liegt anatomisch mittig und an der tiefsten Stelle. Diese Information wird genutzt für die Suche. Zur Einschränkung des Suchraums werden die x-Koordinaten der Schlüsselpunkte P_2 und P_4 genutzt. Es werden die Spalten des Gradientenbildes von links nach rechts durchsucht. Dabei werden die Spalten von unten nach oben durchlaufen. Der Grauwert der weiß ist und am tiefsten liegt ist der gesuchte Schlüsselpunkt P_6.

Generierung der Merkmale

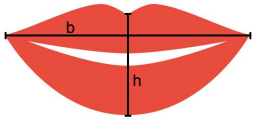


Abbildung 4.20: Breite und Höhe

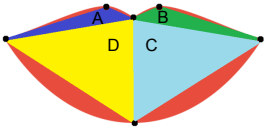


Abbildung 4.21: Fläche

Aus den gefundenen Schlüsselpunkten werden zwei weitere Merkmale generiert, die für die Wiedererkennung genutzt werden. Das Seitenverhältnis der Lippen wird verwendet. Um das Seitenverhältnis zu berechnen, wird die Breite und Höhe der Lippen verwendet. Die Breite wird aus den Schlüsselpunkten P_1 und P_5 und die Höhe P_3 und P_6 berechnet.

$$Seitenverhältnis = \frac{h_{Mund}}{b_{Mund}} \tag{4.20}$$

$$a = d_{SP_{12}} = |\vec{SP_1} - \vec{SP_2}| \tag{4.21}$$

$$b = d_{SP_{23}} = |\vec{SP_2} - \vec{SP_3}| \tag{4.22}$$

$$c = d_{SP_{13}} = |\vec{SP_1} - \vec{SP_3}| \tag{4.23}$$

$$s = \frac{a+b+c}{2} \tag{4.24}$$

$$s = \sqrt{s*(s-a)*(s-b)*(s-c)} \tag{4.25}$$

Die Gleichung 4.20 berechnet das Seitenverhältnis des Mundes. Um die Fläche des Mundes zu berechnen, werden mithilfe der Schlüsselpunkte Dreiecke gebildet (siehe Abb. 4.21). Die unterschiedlichen Flächen der Dreiecke werden zur Berechnung der Gesamtfläche addiert. Für die Berechnung des Flächeninhalts eines Dreiecks wird der Satz des Heron angewendet. Der Satz des Heron kann genutzt werden, wenn alle drei Seitenlängen gegeben sind. Um die Seitenlängen zu berechnen, werden die Gleichungen 4.21 bis 4.23 angewendet. SP steht für

Schlüsselpunkt. Die euklidische Distanz der Schlüsselpunkte wird berechnet. In Gleichung 4.24 wird der halbe Umfang des Dreiecks berechnet, dieser wird in Gleichung 4.25 benutzt, um die Fläche des Dreiecks zu berechnen.

Normalisierung:

Die Merkmale können sich bei gleicher Äußerung stark unterscheiden, wenn die Distanz des Kopfes zur Kamera unterschiedlich ist. Die extrahierten Merkmale (Seitenverhältnis und Fläche der Lippen) müssen daher normalisiert werden, d.h. sie werden unabhängig zur Distanz. Das Seitenverhältnis an sich ist ein unabhängiges Maß, da es ein Verhältnis ist. Es bedarf keiner Normalisierung. Die Fläche der Lippen vergrößert sich je näher die Person an die Kamera geht und verkleinert sich, wenn sich die Person von ihr entfernt. Für die Normalisierung der Fläche wird die Tiefenkamera bzw. das Tiefenbild der Kinect genutzt. Es wird die Linsengleichung/Abbildungsgleichung genutzt. In dieser Gleichung wird der Zusammenhang zwischen der Gegenstandsgröße (Lippenfläche in der Realität), Bildgröße (Lippenfläche im Bild), Gegenstandsweite (Distanz der Lippenfläche zur Kinect) und Bildweite (nicht beachtete Distanz von Kamera zur Linse) beschrieben.

$$\frac{B}{G} = \frac{b}{g} \tag{4.26}$$

$$G = B * \frac{g}{b} \tag{4.27}$$

In Gleichung 4.26 wird das Verhältnis von Bildgröße (B), Gegenstandsgröße (G), Bildweite (b) und Gegenstandsweite (g) beschrieben. Für die normierte Fläche ist es notwendig die Gegenstandsgröße zu berechnen (Gleichung 4.27), dabei kann die Bildweite b vernachlässigt werden (ScanDig [2016]). Beide normalisierten Merkmale werden zusätzlich, damit sie untereinander im DTW-Algorithmus vergleichbar und kombinierbar sind, in einer Skala im Bereich von 0 bis 1 gewandelt.

4.5 Ähnlichkeitsanalyse von Trajektorien

In diesem Abschnitt wird das DTW mit einer Methode, die die euklidische Distanz zur Ähnlichkeitsmessung nutzt, verglichen. In dieser Arbeit wurde das DTW erweitert, indem Gewichtungen verwendet werden. Die Erweiterung wird ebenfalls vorgestellt.
Bei einer Äußerung entstehen aus den Merkmalsvektoren (Seitenverhältnis und Fläche der

Lippen) über die Zeit Trajektorien. Diese Trajektorien können miteinander verglichen werden, um Ähnlichkeiten zu messen:

4.5.1 Dynamic-time-warping gegen Euklidische Distanz

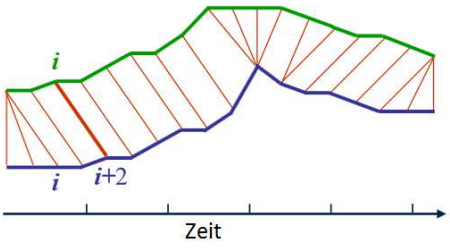


Abbildung 4.22: DTW
Quelle: Criel und Tsiporkova [2006]

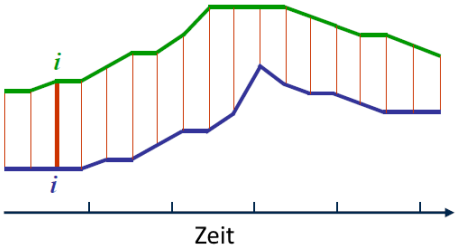


Abbildung 4.23: Euklidische Distanz: Vergleich von Trajektorien
Quelle: Criel und Tsiporkova [2006]

Ein Verfahren um Ähnlichkeit von Trajektorien zu messen, ist die Verwendung der euklidischen Distanz. Bei der Methode werden die Merkmalsvektoren der Trajektorien A und B zum Zeitpunkt i miteinander verglichen, indem die euklidische Distanz berechnet wird. Es findet also ein Punkt-zu-Punkt-Vergleich statt (siehe Abb. 4.23). Je geringer die Distanz desto höher die Ähnlichkeit.

$$D_e(A,B) = \sqrt{\sum_{i=1}^{N}(a_i - b_i)^2} \tag{4.28}$$

In Gleichung 4.28 wird die euklidische Distanz D_e der Trajektorien A und B berechnet. N ist die Länge der Trajektorie, a_i und b_i sind die Merkmalswerte zum Zeitpunkt i (Zhang u. a. [2006]).

Nachteil der euklidischen Ähnlichkeitsmessung ist, dass beide Trajektorien gleichlang sein müssen. Sind die Trajektorien nicht gleichlang, so muss die längere Trajektorie entweder am Anfang oder am Ende gekürzt werden. Dies ist ein weiterer Nachteil, da Daten/Informationen verloren gehen. Dieses Verfahren erlaubt nicht das Vergleichen von Trajektorien mit starken lokalen temporalen Variationen (Ali u. a. [2014]), da nur ein Punkt-zu-Punkt-Vergleich stattfindet. Für größere Datensätze ist die euklidische Distanz geeignet, da das Verfahren keine rechenintensive Berechnung durchführt. Laut Fangerau u. a. [2012] bietet das Verfahren eine hohe Genauigkeit bei großen Datensätzen, jedoch ist das Verfahren anfällig für Störungen innerhalb der Trajektorien.

DTW ist robuster als das Verfahren, welches die euklidische Distanz (VED) nutzt. Dies liegt daran, da DTW Zeitstörungen und Zeitverschiebung ausgleicht (siehe Abschnitt 3.6) (Zhang u. a. [2014]). DTW ist nicht geeignet für große Datensätze, da die Berechnungen kostenintensiv

sind. In dieser Arbeit werden Kurzkommandos (siehe Abschnitt 5.1) wiedererkannt, diese stellen keine langen Trajektorien dar. Verschieden schnell gesprochene Äußerungen kann DTW durch seine Warpeigenschaft ausgleichen (Yanagisawa und Satoh [2006]). In dieser Arbeit werden Äußerungen per DTW und VED miteinander verglichen. Die Ergebnisse und Erkenntnisse werden in Abschnitt 5 vorgestellt.

4.5.2 Weighted Dynamic-time-warping

DTW ist geeignet für Ähnlichkeitsmessungen bei Trajektorien mit zeitlichen Variationen in der Äußerung. Ebenfalls gleicht DTW die Variationen aus, wenn gleiche Kommandos unterschiedlich lang ausgesprochen werden. Die Wiedererkennung bzw. die Ähnlichkeitsmessung weist jedoch Fehler auf, wenn sehr kurze Kommandos mit sehr langen Kommandos verglichen werden. So kann es vorkommen, dass geringe Distanzen berechnet werden, wenn kurze Kommandos/Äußerungen mit langen Kommandos/Äußerungen verglichen werden. Dies führt dazu, dass falsche Kommandos erkannt werden. Der Grund für dieses Verhalten ist, dass kurze Kommandos auch kürzere Trajektorien besitzen und somit auch kleinere Distanz- und Kostenmatrizen erstellt werden. Der „Warping-Pfad“ ist daher ebenfalls kurz und die Distanz dementsprechend gering. Dieses Problem kann beseitigt werden indem Gewichtungen für die Kommandos eingeführt werden. Die Gewichtungen hängen von der Länge eines Kommandos ab, d.h. kürzere Kommandos haben eine höhere Gewichtung als lange Kommandos. So können die Distanzen angepasst werden. Es folgt eine Darstellung der Erweiterung:
Die Gewichtung wird je Kommando berechnet, dabei spielt die Länge des Kommandos eine Rolle. Die Gewichtung ist ein Prozentwert, der zusätzlich zur berechneten Distanz von zwei Trajektorien hinzuaddiert wird.

$$l_M = \frac{\sum_{i=0}^{N=k} r_i^{\text{Länge}}}{N} \tag{4.29}$$

$$l_{Max} = max(r_{i,k}^{\text{Länge}}) \tag{4.30}$$

$$g_{moveforward} = \frac{(l_{Max} - l_M) * a_{MF}}{100} \tag{4.31}$$

Die Gleichungen 4.29 bis 4.31 zeigen die Berechnung einer Gewichtung für ein Kommando. Als Datengrundlage dienen die Trajektorien, die als Repräsentanten je Kommando fungieren. Der Algorithmus für die Suche der Repräsentanten wird in Abschnitt 4.6.2 behandelt. Es gibt k-Repräsentanten je Kommando. In Gleichung 4.29 wird der Mittelwert l_M der Trajektorien-

länge der k-Repräsentanten berechnet. Dabei steht $r_i^{\text{Länge}}$ für die Länge der Trajektorie des i-ten Repräsentanten. Der Wert der längsten Trajektorie aller Kommandos l_{Max}, also nicht nur des betrachteten Kommandos, wird in Gleichung 4.30 gespeichert. Als letzten Schritt wird in Gleichung 4.31 die Gewichtung $g_{moveforward}$ für das Kommando „move forward“ berechnet. Dabei bestimmt a_{MF} (MF = Maximal Faktor) einen vordefinierten Prozentwert, der beschreibt wie viel maximal zur Distanz dazu addiert werden kann. Ein exemplarisches Beispiel: Wenn l_{Max} 80 ist und die Durchschnittslänge des betrachteten Kommandos ebenfalls 80 wird nichts zur Distanz hinzuaddiert. a_{MF} kann frei gewählt werden, in dieser Arbeit ist er auf 30 % gesetzt. Es können also maximal 30 % zur Distanz hinzuaddiert werden. Im Abschnitt 5 erfolgt eine Auswertung der Erweiterung.

$$d_{\text{WP}} = \sum_{i=1}^{N} w_i \tag{4.32}$$

$$d_g = d_{\text{WP}} + (d_{\text{WP}} * g_{moveforward}) \tag{4.33}$$

In Gleichung 4.33 wird die Gewichtung zur Distanz hinzuaddiert. Die aufsummierten Distanzen entlang des „Warping-Pfads“ beschreiben die Distanz d_{WP} zweier Trajektorien (siehe Abschnitt 3.6).N ist die Länge des WPs.

4.5.3 Das Maß für die Wiedererkennung

Das visuelle Spracherkennungssystem ist ein unterstützendes System. In der Machbarkeitsanalyse wurden die Gründe genannt (Abschnitt 3.3). Viele Laute geschehen im Mundhohlraum (Zunge, Zähne, Rachen). Die Wörter können nicht ausschließlich durch die Lippen zuverlässig erkannt werden. Aus diesem Grund wird für die Wiedererkennungsrate ein Ranking eingesetzt. Folgendes Beispiel verdeutlicht das Ranking:

1. Die Aussage „move forward“ wird getätigt.
2. Die Distanz der Aussage zu den jeweiligen Repräsentanten je Kommando wird durch (W)DTW berechnet.
3. Die Distanzen und die dazugehörigen Kommandos werden aufsteigend sortiert.
4. Die fünf geringsten Distanzen werden betrachtet. Ist die Aussage „move forward“ einer der Kommandos mit den fünf geringsten Distanzen, so wird folgende Wiedererkennungsrate vergeben: Ist „move forward“ auf Platz 1 (geringste Distanz): 1 Punkt, Platz 2: $\frac{1}{2}$, Platz 3: $\frac{1}{3}$, Platz 4: $\frac{1}{4}$ und Platz 5: $\frac{1}{5}$, sonst 0 Punkte.

Für die Auswertung wird das Maß zur Wiedererkennung verwendet (siehe Abschnitt 5).

4.6 Klassifikation

Im folgenden Abschnitt wird dargestellt, wie eine Äußerung (in Form einer Trajektorie) klassifiziert wird. Es wird also klassifiziert zu welchem Kommando die Äußerung gehört. Für dieses Verfahren werden die Distanzen aus dem gewichteten DTW (WDTW), normalen DTW und VED (Verfahren, welches die euklidische Distanz verwendet) verwendet. Es wird zunächst die verwendete Methode zur Klassifizierung „k-Nearest-Neighbor“ (k-NN) vorgestellt. Es ist je Kommando erforderlich ein Datenset in Form von Trajektorien zu erstellen. Innerhalb des Trajektoriens vcets werden k-Repräsentanten gesucht, die verwendet werden für die Ähnlichkeitsmessung bei eintreffender Äußerung. Für das Training wird ein Clustering-Verfahren verwendet. Das Training bzw. Clustering wird ebenfalls präsentiert.

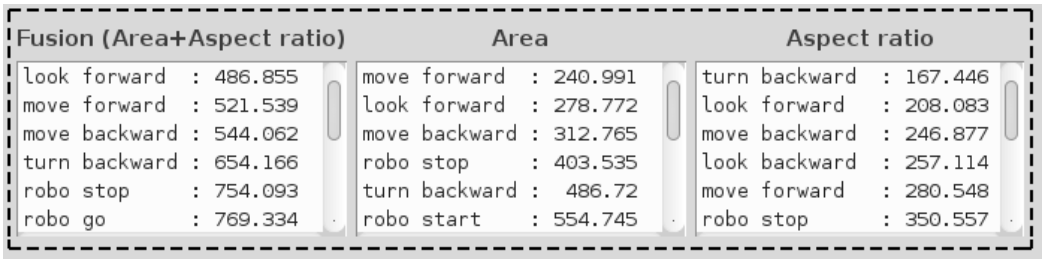


Abbildung 4.24: Distanzen: Fusion, Fläche und Seitenverhältnis

In dieser Arbeit werden zwei Merkmale für die Wiederkennung genutzt: das Seitenverhältnis und die Fläche der Lippen. Die Distanzen werden je Merkmal berechnet, d.h. WDTW wird zweimal je Äußerung ausgeführt. Für die Klassifikation werden beide Distanzen miteinander fusioniert, indem sie addiert werden. In Untersuchungen wurde festgestellt, dass die Wiedererkennung erheblich gesteigert wird, wenn die Merkmale zusammen betrachtet werden. In Abbildung 4.24 sind die Distanzen zu einer Äußerung („look forward“) je Kommando zu sehen. In diesem Test wurde „look forward“ gesagt. Werden ausschließlich einzeln die Distanzen für die Fläche oder das Seitenverhältnis betrachtet, so hat „look forward“ nur die zweit geringste Distanz je Merkmal. Werden beide Distanzen jedoch addiert, so haben sie die geringste Distanz aller Kommandos und dementsprechend wurde „look forward“ erkannt. Die Fusion der Distanzen der Merkmale ist nur möglich, da sie vorher normalisiert wurden.

4.6.1 k-Nearest-Neighbor

Für die Klassifikation wird der „k-Nearest-Neighbor“ (k-NN) genutzt. k-NN ist ein einfacher Klassifikator, denn er prüft anhand von Distanzen seinen nächsten Nachbarn. Es wird nicht nur sein nächster Nachbar, sondern seine nächsten k-Nachbarn gesucht. In diesem Fall ist der Nachbar ein Kommando. k ist frei wählbar. In k-NN wird ein Objekt (Äußerung) zu einer Klasse (Kommando) zugeordnet, wenn das Objekt am häufigsten zu den nächsten k zu geordnet wird. Ein k ist ein Vertreter/Repräsentant einer Klasse (Altman [1992]).
Durch das Training (siehe Abschnitt 4.6.2) werden k-Repräsentanten je Kommando gefunden. In dieser Arbeit werden diese Repräsentanten genutzt für den k-NN-Algorithmus. Eine Äußerung wird also untersucht zu welcher dieser Repräsentanten je Kommando sie die geringste Distanz hat. Die Distanz wird berechnet durch (W)DTW.

4.6.2 Training/Clustering

In diesem Abschnitt wird das Verfahren beschrieben um Repräsentanten je Kommando zu finden. Je Kommando gibt es ein Trainingsset. Das Trainingsset besteht aus Trajektorien, die vorab aufgesagt/aufgezeichnet und in die Datenbank gespeichert werden. Die aus dem Training gefundenen Trajektorien repräsentieren das Trainingsset bzw. das Kommando. Wie viele Repräsentanten gefunden werden können, ist frei wählbar. Es ist erforderlich verschiedene Repräsentanten je Kommando zu haben, da Äußerungen des gleichen Kommandos variieren können. Trajektorien gleicher Äußerung können sich durch Sprechgeschwindigkeit, Art der Aussprache (Dialekt, Akzent, Nuscheln) und wenn für ein Trainingsset verschiedene Sprecher genommen werden, stark unterscheiden.
Für das Training wird ein Clustering-Algorithmus verwendet. Clustering ist eine Methode, um eine endliche Menge von Klassen oder Gruppen (Cluster) in vorhandenen Daten zu identifizieren. Dabei haben die Daten/Objekte im gleichen Cluster gleiche Eigenschaften und sind sich somit ähnlich. Es werden ebenfalls interessante Verteilungen der Daten/Objekte identifiziert. Dabei werden Distanzen der Objekte zueinander bestimmt oder andere Ähnlichkeitsmessungen (DTW) durchgeführt, um ihre Gruppe (Cluster) zu bestimmen (Halkidi u. a. [2001]). In dieser Arbeit wird der Clustering-Algorithmus k-Medoids verwendet. Er ist verwandt mit dem bekannten Clustering-Algorithmus k-Means. k-Means ist ein iterativer Algorithmus:

1. Beim Algorithmus werden zufällige k Objekte als Cluster-Zentren aus einem Datenset gewählt. k ist frei wählbar.

2. Die übrigen Daten werden den vorher zufällig gewählten nächsten Cluster-Zentren zugeordnet. Hierbei bestimmt die euklidische Distanz die Zuordnung.

3. Nun wird für jedes Cluster der Durchschnittswert (Mean) der Objekte berechnet. Dieser für jedes Cluster berechnete Wert ist das neue Cluster-Zentrum.

4. Der Algorithmus wiederholt sich (Schritte 2 bis 4) und es werden wieder alle Objekte zu den neuen Cluster-Zentren zugeordnet. Dies geschieht so lange, bis die Cluster-Zentren (Mittelwerte) sich nicht mehr verändern (Rujasiri und Chomtee [2009]).

k-Means ist nur geeignet wenn die Objekte numerische Daten sind, da sich die Cluster-Zentren anhand des Mittelwerts der Daten ändern.

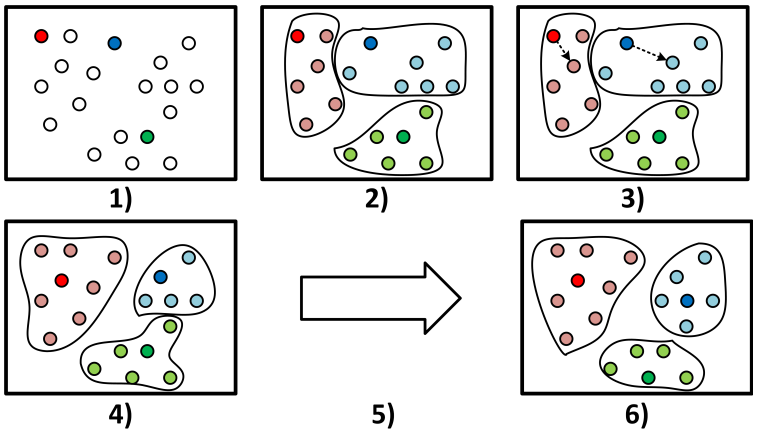


Abbildung 4.25: Clustering-Algorithmus: k-Medoids

Für das clustern von Trajektorien kann k-Means nicht eingesetzt werden, da Trajektorien Merkmalsvektoren sind. Es kann aus dem Merkmalsvektor keine Distanz durch Bildung des Mittelwerts bestimmt werden. Für ein Trajektorienset ist also der Mittelpunkt nicht definierbar. Es wird der Clustering-Algorithmus k-Medoids eingesetzt. Bei k-Medoids ist das Objekt selbst das Cluster-Zentrum (Medoid) und nicht ein berechneter Mittelwert. In diesem Fall werden Trajektorien zu Cluster-Zentren ausgewählt. k-Medoids verläuft ähnlich wie k-Means (Sandhya und VijayKumar [2013]):

1. In einem Trajektorienset werden zufällig k Trajektorien als Medoids ausgewählt. k ist frei wählbar.

2. Die übrigen Trajektorien werden jeweils zu den nächsten k Medoids zugeordnet. Der nächste Medoids ist der mit der geringsten Distanz. Die Distanz zu den jeweiligen Medoids wird durch DTW bestimmt.

3. Es wird innerhalb eines Clusters der neue Medoid gesucht. Es wird die Distanz der Trajektorien untereinander berechnet. Die Trajektorie, die die geringste Distanz zu allen anderen Trajektorien hat, wird als neues Medoid gewählt.

4. Gibt es Änderungen in den Clustern, d.h. wurde ein neues Medoid gefunden/gewählt, so werden die Punkte 2 bis 4 wiederholt.

In Abbildung 4.25 ist der k-Medoids-Algorithmus dargestellt. Schritt 1 im Bild ist der erste Schritt des Algorithmus. Die dunkleren Punkte stellen die Medoids dar. Beim zweiten Schritt im Bild werden die Cluster gebildet (Schritt 2 im Algorithmus). Die neuen Medoids werden in Schritt 3 im Bild gesucht und gesetzt (Schritt 3 im Algorithmus). Schritt 4 im Bild: Nachdem die neuen Medoids gesetzt wurden, wird der Algorithmus ab Schritt 2 aus wiederholt. Die Schritte 2 bis 4 des Algorithmus werden so lange ausgeführt bis die passenden Medoids gefunden werden (Schritte 5 und 6 im Bild).
Im Abschnitt 4.1.2 wurde das Datenformat für eine Trajektorie dargestellt. Die Repräsentanten je Kommandos werden ebenfalls in MongoDB gespeichert. Die Darstellung sieht wie folgt aus:

```
<trajectory>
      <command> look forward </command>
      <feature> area </feature>
      <clusterMethod> k-medoids </clusterMethod>
      <values>
            <k1>
                  <val id = 1> 822.5 </val>
                  <val id = 2> 780.01 </val>
                  ...
                  <val id = n> 820.5 </val>
            </k1>
            <k2>
                  <val id = 1> 835.01 </val>
                  <val id = 2> 884.51 </val>
                  ...
                  <val id = n> 896.98 </val>
            </k2>
            ...
            <kn>
                  ...
            </kn>
      </values>
</trajectory>
```

Listing 4.3: Trajektorien als Repräsentanten im XML-Format

Im Code 4.3 werden das Kommando, das Merkmal und die Methode zur Repräsentantensuche gespeichert. Da es k beliebige Repräsentanten geben kann, werden je Datum k Trajektorien gespeichert.

5 Auswertung

In diesem Kapitel erfolgt eine Auswertung des visuellen Spracherkennungssystems. Dabei werden verschiedene Versuche durchgeführt. Der Versuchsaufbau wird erläutert. Zunächst wird in einem Versuch ermittelt welche Cluster-Parameter die besten Wiedererkennungsraten liefern. Im zweiten Versuch werden vordefinierte Kommandos aufgesagt. Es wird ermittelt welche Parameter (DTW-Einschränkungen) die besten Wiedererkennungsraten liefern. In Versuch 3 wird WDTW mit dem Verfahren, welches die euklidische Distanz nutzt, verglichen, indem untersucht wird welches Verfahren eine höhere Wiedererkennungsrate liefert. Im vierten Versuch tritt eine Person gegen das visuelle Spracherkennungssystem an. Die Person versucht die Lippen zu lesen. Zur gleichen Zeit wird das visuelle Spracherkennungssystem eingesetzt. Beide Ergebnisse werden miteinander verglichen.
Anschließend erfolgt eine Bewertung der Versuche und Erkenntnisse aus den Ergebnissen. Aus den Erkenntnissen der Versuche und der Analyse werden notwendige Erweiterungen für den Prototyp vorgestellt.

5.1 Versuchsaufbau

In den Versuchen wird das visuelle Spracherkennungssystem „LipsRec“ (Kurz) Kommandos erkennen.

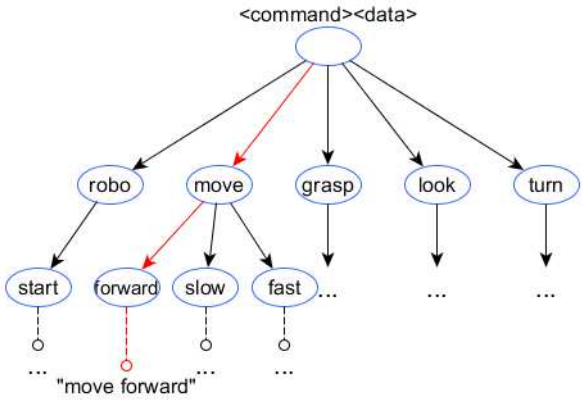


Abbildung 5.1: Aufbau - Kurzkommandos

In Abbildung 5.1 ist der Aufbau der Kommandos zu sehen. Die Grammatik besteht aus einem „command“ und einem „data“. Dabei bilden diese beiden Wörter ein „order“. In der Abbildung ist der Pfad des Befehls „move forward“ abgebildet.
Die Befehle, die dem Roboter gegeben werden können, sind: „robo“, „move“, „grasp“, „look“ und „turn“.

$$< robo > (< go > | < start > | < stop > | < off >) \tag{5.1}$$
$$< move > (< forward > | < backward > | < slow > | < fast >) \tag{5.2}$$
$$< grasp > (< close > | < open >) \tag{5.3}$$
$$< look > (< right > | < left > | < forward > | < backward >) \tag{5.4}$$
$$< turn > (< left > | < right > | < backward >) \tag{5.5}$$

In den Formeln 5.1 bis 5.5 sind die möglichen Kombinationen der Wörter zu sehen, die ein Kommando bilden können. Es sind insgesamt 17 Kommandos. Für jedes der 17 Kommandos wird ein Trainingsset angelegt. Es werden 50 Äußerungen je Kommando aufgezeichnet. In dieser Arbeit werden zwei Lippenmerkmale (Seitenverhältnis und Fläche der Lippen) genutzt, daher werden zwei Mal 50 2-D-Trajektorien je Kommando gespeichert.

Wertung der Wiedererkennungsraten

Um beurteilen zu können welche Cluster-Parameter in Versuch 1 und welche DTW-Paramter in Versuch 2 höhere Wiedererkennungsraten liefern, welches Verfahren besser ist in Versuch 3

und welche Wiedererkennungsrate Mensch oder Maschine in Versuch 4 höher ist, wird das in Abschnitt 4.5.3 Maß für die Wiedererkennungsrate verwendet. Es wird je Konfiguration (Versuch 1 und 2), Verfahren (Versuch 3) oder Art (Versuch 4) ein Ranking, gemäß wie in Abschnitt 4.5.3 beschrieben, aufgestellt. Es werden 50 Äußerungen getätigt und das Ranking aufgestellt. Das Ranking stellt die Häufigkeit auf, wie wahrscheinlich das richtige Kommando zur Äußerung zugeordnet wird. Anhand des folgenden Beispiels wird die Wertung verdeutlicht:

move forward	Rang	Mensch	Maschine
	1	31	30
	2	14	6
	3	3	4
	4	0	2
	5	1	8
Wiedererkennungsrate		39,4	36,43

Tabelle 5.1: Auswertung eines Versuchs

Das Kommando „move forward“ wird von einer Person und dem visuellen Spracherkennungssystem wiedererkannt. Das Kommando wird 50 Mal aufgesagt. In Tabelle 5.1 ist Ranking + Wiedererkennungsrate zu sehen. Die Person hat 31 Äußerungen direkt als „move forward“ erkannt (Rang 1). In 14 Fällen wird die Äußerung am zweit wahrscheinlichsten wiedererkannt und beispielsweise erkennt die Person (fälschlicherweise) „move backward“ eher usw. Die Maschine (visuelle Spracherkennungssystem „LipsRec“) trägt anhand der Distanzen zu den jeweiligen Repräsentanten die Häufigkeiten der Ränge automatisch ein. Anhand des Maßes für die Wiedererkennungsrate, welches in Abschnitt 4.5.3 vorgestellt wurde, werden die Wiedererkennungsraten berechnet:

$$Mensch = 1 * 31 + \frac{1}{2} * 14 + \frac{1}{3} * 3 + \frac{1}{4} * 0 + \frac{1}{5} * 1 = 39,4 \tag{5.6}$$

$$Maschine = 1 * 30 + \frac{1}{2} * 6 + \frac{1}{3} * 4 + \frac{1}{4} * 2 + \frac{1}{5} * 8 = 36,43 \tag{5.7}$$

5.1.1 Versuch 1 - Suche der besten Repräsentanten (Clustering)

Um zu bestimmen welche Repräsentanten die besten Wiedererkennungsraten liefern, werden unterschiedliche Cluster-Parameter untersucht. Durch das k-Medoids Clustering-Verfahren (siehe Abschnitt 4.6.2) werden Cluster aufgestellt bzw. die jeweiligen Repräsentanten gefunden. Es wird untersucht wie viele k-Repräsentanten notwendig sind für eine hohe Wiedererkennungsrate. Um die Anzahl der Tests einzuschränken werden fünf Kommandos („robo

start“, „move forward“, „grasp close“, „look left“ und „turn right“) benutzt. Die fünf Kommandos werden 50 Mal je 10 verschiedene Cluster-Parameter aufgesagt. Damit die Anzahl der Tests des Versuchs nicht steigt, wird nicht jede Kombination von Parametern getestet. Es werden Erfolg versprechende Parameter aus Erfahrung ausgewählt. Der Versuch hat 2500 (10 Cluster-Paramter $*$ 5 Kommandos $*$ 50 Äußerungen) Tests:

Cluster	k	DTW: Schritt-Muster	DTW: Warping-Fenster aktiviert	DTW: Größe des Warping-Fensters
1	1	a	Nein	-
2	5	b	Nein	-
3	10	a	Nein	-
4	1	a	Ja	8
5	1	c	Ja	5
6	5	a	Ja	5
7	5	a	Ja	8
8	10	a	Ja	5
9	10	a	Ja	8
10	10	a	Ja	12

Tabelle 5.2: Versuchsparameter des Versuchs 1

In Tabelle 5.2 sind die Versuchsparameter aufgelistet. Die unterschiedlichen Schritt-Muster sind in Abschnitt DTW 3.6 zu sehen. Das k beschreibt wie viele Repräsentanten gesucht werden. Es werden also gemäß Tabelle zehn Cluster gebildet. Jeder dieser zehn Cluster hat Repräsentanten, die für die Wiedererkennung (Ähnlichkeitsmessung per DTW) genutzt werden. Es werden die drei Cluster (und deren Repräsentanten) gesucht, mit denen die höchste Wiedererkennungsrate erreicht wird. Aus diesem Test erfolgt die Erkenntnis, dass ein höheres k bessere Wiedererkennungsraten liefert.

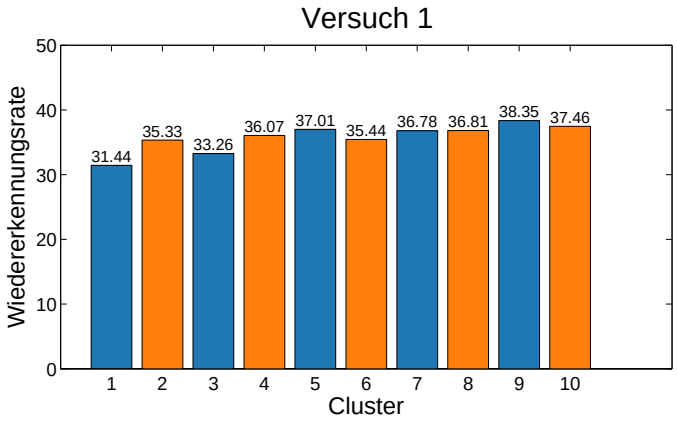


Abbildung 5.2: Wiedererkennungsraten des ersten Versuchs

In der Abbildung 5.2 ist das Ergebnis des ersten Tests zu sehen. Mit den Repräsentanten der Cluster 9, 10 und 5 wird die höchste Wiedererkennungsrate erreicht. Beim Versuch werden

fünf Kommandos getestet, der Mittelwert der Wiedererkennungsraten ist in der Abbildung 5.2 zu sehen.

5.1.2 Versuch 2 - Vollständiges Kommandoset

In Versuch 2 werden die Repräsentanten der Cluster 9, 10 und 5 genutzt, um zu testen welche DTW-Parameter die besten Wiedererkennungsraten liefern. Hierbei wird das komplette Kommandoset getestet, d.h. alle 17 Kommandos. Die Repräsentanten werden verwendet für die Klassifikation der Äußerungen. Es werden nicht alle Kombinationen der DTW-Parameter getestet, da die Anzahl der Tests zu hoch wäre. Aus diesem Grund werden Parameter aus Erfahrung ausgewählt. Insgesamt werden 17850 Tests durchgeführt: 3 Cluster $*$ 17 Kommandos $*$ 50 Äußerungen $*$ 7 Parametereinstellungen. In diesem Test wird ebenfalls analysiert, ob die Erweiterung von DTW, WDTW eine bessere Wiedererkennungsrate liefert oder nicht.

Test/ Parametereinstellung	Cluster	k	Schritt-Muster	Warping-Fenster aktiviert	Größe des Warping-Fensters	WDTW aktiviert	WDTW: $a_{MaxFaktor}$
1	9	10	a	Nein	-	Nein	-
2	9	10	a	Nein	-	Ja	30 %
3	9	10	a	Ja	5	Nein	-
4	9	10	a	Ja	8	Nein	-
5	9	10	a	Ja	5	Ja	30 %
6	9	10	a	Ja	8	Ja	30 %
7	9	10	c	Ja	8	Ja	30 %
8	10	10	a	Nein	-	Nein	-
9	10	10	a	Nein	-	Ja	30 %
10	10	10	a	Ja	5	Nein	-
11	10	10	a	Ja	8	Nein	-
12	10	10	a	Ja	5	Ja	30 %
13	10	10	a	Ja	8	Ja	30 %
14	10	10	c	Ja	8	Ja	30 %
15	5	1	a	Nein	-	Nein	-
16	5	1	a	Nein	-	Ja	30 %
17	5	1	a	Ja	5	Nein	-
18	5	1	a	Ja	8	Nein	-
19	5	1	a	Ja	5	Ja	30 %
20	5	1	a	Ja	8	Ja	30 %
21	5	1	c	Ja	8	Ja	30 %

Tabelle 5.3: Versuchsparameter des Versuchs 2

In Tabelle 5.3 sind die Versuchsparameter des zweiten Versuchs zu sehen. Es werden 21 Parametereinstellungen (3 Cluster $*$ 7 Parametereinstellungen) genutzt.

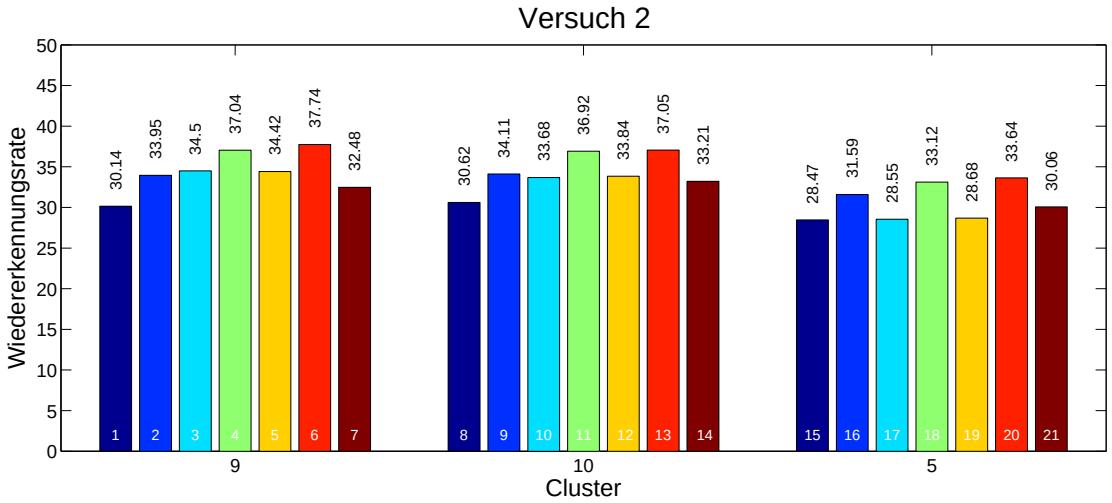


Abbildung 5.3: Wiedererkennungsraten des zweiten Versuchs

In Abbildung 5.3 sind die Wiedererkennungsraten des zweiten Versuchs abgebildet. Das Säulendiagramm ist anhand der Cluster gruppiert. Die Wiedererkennungsraten der jeweiligen Parametereinstellungen sind auf den jeweiligen Säulen platziert. Die Säulen sind nummeriert anhand Tests bzw. der Parametereinstellungen gemäß Tabelle 5.3. Die jeweiligen Säulen stellen den Mittelwert der Wiedererkennungsraten aus den 17 Kommandos je 50 Äußerungen dar. Die Parametereinstellung Nummer 6 liefert die höchste Wiedererkennungsrate:

- Repräsentanten: 10 aus dem Cluster 9
- Schritt-Muster: a
- Warping-Fenster: aktiviert
- Größe des Warping-Fenster: 8
- WDTW aktiviert: ja
- a_{MF}: 30 %

WDTW gegen DTW

Es wird untersucht, ob WDTW oder DTW eine höhere Wiedererkennungsrate liefert: Bei der Untersuchung werden jeweils gleiche Parametereinstellungen (Tabelle 5.3) miteinander verglichen mit dem Unterschied ob WDTW aktiviert ist oder nicht. Die Parametereinstellung Nummer 1 nutzt keine Gewichtung und Nummer 2 nutzt WDTW. Mit der Nutzung steigt die Wiedererkennungsrate um 3,81 Punkte. Dasselbe Verhalten ist bei den Parametereinstellungen Nummer 8, 9 (Steigerung um 3,49) und 15, 16 (Steigerung um 3,12) zu beobachten. Die erwähnten Parametereinstellungen nutzen kein „Warping-Fenster“. Wird ein „Warping-Fenster“ genutzt, so gibt es nur eine geringe Steigerung der Wiedererkennungsrate gegenüber den

Parametereinstellungen, die kein WDTW nutzen. Es gibt eine Steigerung um 0,7 bei Parametereinstellungen von 4 auf 6, bei Parametereinstellungen 10 auf 12 eine Steigerung um 0,16, bei Parametereinstellungen 11 auf 13 eine Steigerung um 0,13, bei Parametereinstellungen 17 auf 19 eine Steigerung um 0,13 und bei Parametereinstellungen 18 auf 20 eine Steigerung um 0,52. Die geringe Steigerung der Wiedererkennungsrate ist begründet, da in den zuvor beobachteten Parametereinstellungen ein „Warping-Fenster“ genutzt wird. Das „Warping-Fenster“ schränkt den Suchbereich/-raum für den „Warping-Pfad“ so weit ein, dass die Gewichtungen keinen großen Einfluss haben. Wird kein „Warping-Fenster“ eingesetzt, so ist der Suchbereich beim DTW größer und der „Warping-Pfad“ verläuft möglicherweise nicht entlang der Diagonale der Matrix (pathologisches Warping siehe Abschnitt 3.6). Der Pfad entlang der Diagonale ist für diese Domäne der beste Vergleich (siehe Abschnitt 3.6). Aus diesem Grunde können die Gewichtungen bei WDTW einen besseren Ausgleich schaffen, daher kommt die höhere Steigerung der Wiedererkennungsrate bei Nutzung von WDTW. Bei Nutzung eines „Warping-Fenster“ wird der Suchbereich auf die Diagonale beschränkt, so haben ungleiche Trajektorien hohe Distanzen an der Diagonale. Aus diesem Grund gibt es keine hohe Steigerung der Wiedererkennungsrate bei Nutzung von Gewichtungen.

Erkenntnisse

Die Parametereinstellung 6 liefert die höchste Wiedererkennungsrate. Wie in den Versuchen zu vor werden alle 17 Kommandos untersucht/getestet. Alle 17 Kommandos werden 50 Mal aufgesagt. Die Ergebnisse aus dieser Einstellung werden analysiert:

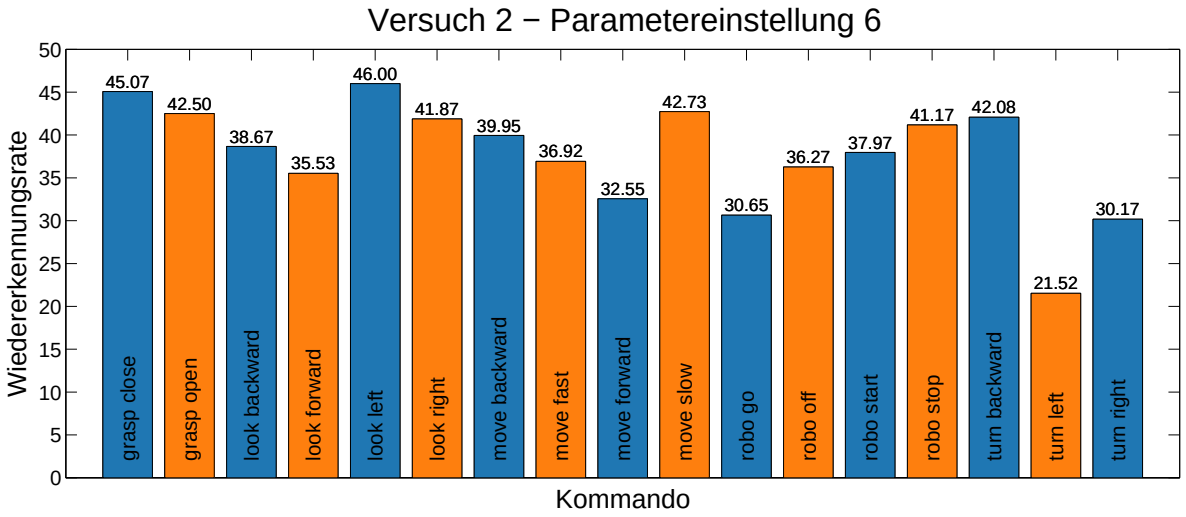


Abbildung 5.4: Wiedererkennungsraten der Parametereinstellung 6

In Abbildung 5.4 sind die Wiedererkennungsraten der Parametereinstellung 6 zu sehen. Auffällig ist, dass die Kommandos „grasp close“ und „grasp open“ eine hohe Wiedererkennungsrate haben. Dies ist zu begründen, da die Wortkombinationen sich stark von den anderen Kommandos unterscheiden. Sie enthalten Wörter, die nicht in den anderen Kommandos enthalten sind. Eine weitere Auffälligkeit ist die niedrige Wiedererkennungsrate der Kommandos „turn left“ und „turn right“. Dies liegt an der Ähnlichkeit der Lippenbewegung der Wörter „turn“ und „left“:

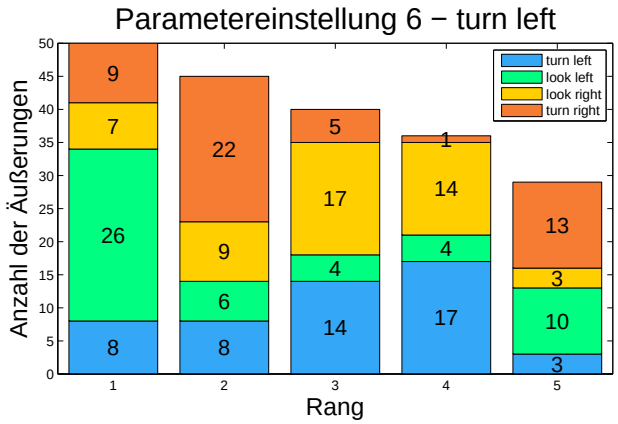


Abbildung 5.5: Rangverteilung von „turn left“

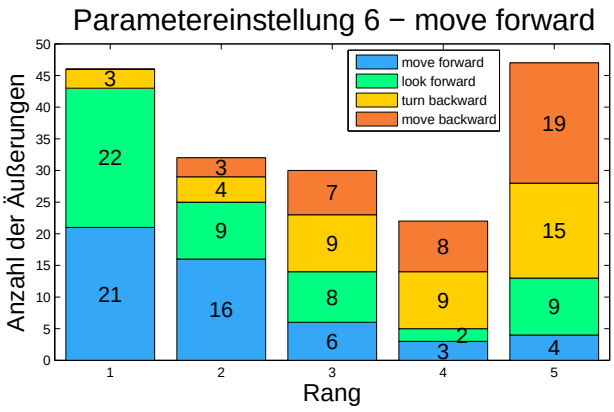


Abbildung 5.6: Rangverteilung von „move forward“

In der Grafik 5.5 ist die Verteilung der Ränge zu „turn left“ zu sehen. Wie bereits beschrieben, wird „look left“ oft mit einer geringeren Distanz als „turn left“ klassifiziert. In der Abbildung 5.6

ist die Verteilung der Ränge zu „move forward“ zu sehen. „look forward“ wird 22 Mal (häufiger als „move forward“) als Rang 1 klassifiziert. Das Wort „forward“ und das Wortsegment „ward“ ist in den abgebildeten Kommandos vorhanden, d.h. die Kommandos ähneln sich. Aus diesem Grund ist eine unregelmäßige Platzierung von „move forward“ auf Rang 1 festzustellen.

5.1.3 Versuch 3 - WDTW gegen euklidisches Distanzverfahren

Im Versuch 3 werden die Wiedererkennungsraten des WDTW mit der Parametereinstellung 6 (höchste Winderkennungsrate) und der Ähnlichkeitsmessung, welches die euklidische Distanz nutzt, verglichen. Für den Versuch werden 2550 Tests durchgeführt: 3 Cluster $*$ 17 Kommandos $*$ 50 Äußerungen. Die Repräsentanten der drei Cluster mit den höchsten Winderkennungsraten (siehe Versuch 1: 5.1.1) werden verwendet.

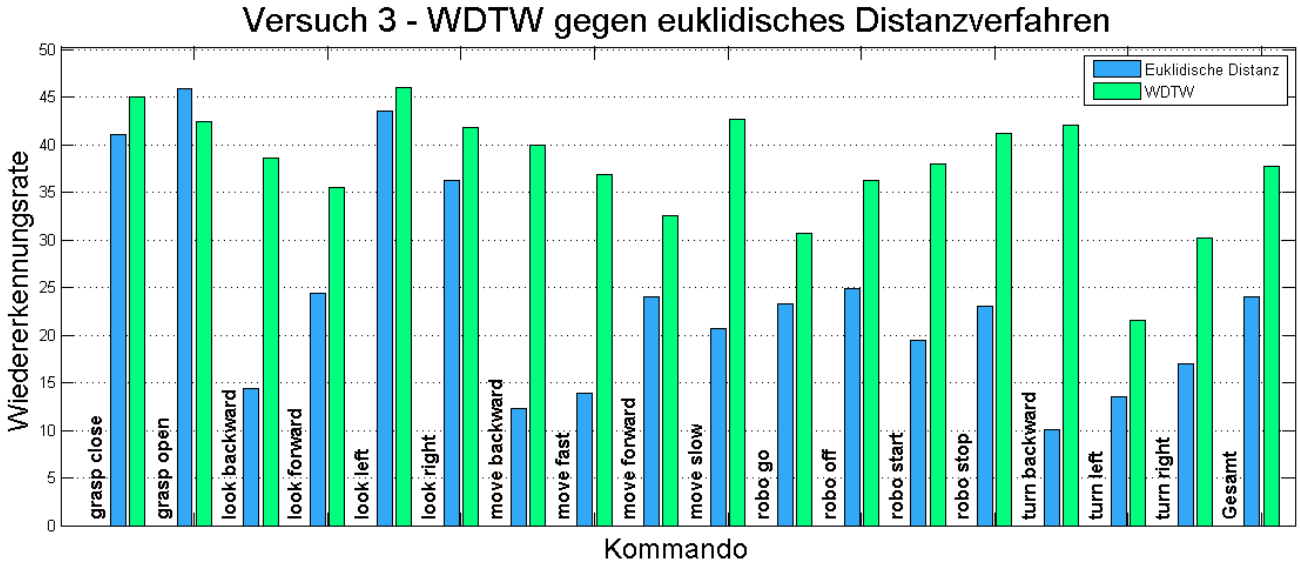


Abbildung 5.7: Wiedererkennungsraten: WDTW gegen euklidisches Distanzverfahren

In Abbildung 5.7 sind die Wiedererkennungsraten des dritten Versuchs zu sehen. Beim euklidischen Distanzverfahren hatten die Repräsentanten des Clusters 9 die höchsten Wiedererkennungsraten, diese werden zum Vergleich in der Abbildung genutzt. WDTW hat eine höhere Gesamtwiedererkennungsrate (37,74) als das euklidische Distanzverfahren (24). Durch die Nutzung von WDTW ist eine Steigerung der Wiedererkennungsrate um 27,5 % (bei 50 Äußerungen je Kommando) möglich.

5.1.4 Versuch 4 - Mensch gegen Maschine

Im Versuch 4 tritt eine Person (Mensch) gegen das visuelle Spracherkennungssystem (Maschine) an. Insgesamt werden 85 Äußerungen getätigt bzw. Tests durchgeführt: 17 Kommandos $*$ 5 Äußerungen. Die Parametereinstellung mit der höchsten Wiedererkennungsrate wird eingesetzt (6). Die Person versucht gleichzeitig mit dem visuellen Spracherkennungssystem die Lippen zu lesen. Um die Winderkennungsraten des visuellen Spracherkennungssystems mit der Klassifizierung der Person verglichen werden kann, notiert sich die Person je Äußerung, wie wahrscheinlich ein Kommando gesagt wurde, d.h. die Person stellt ein Ranking auf (Top 5).

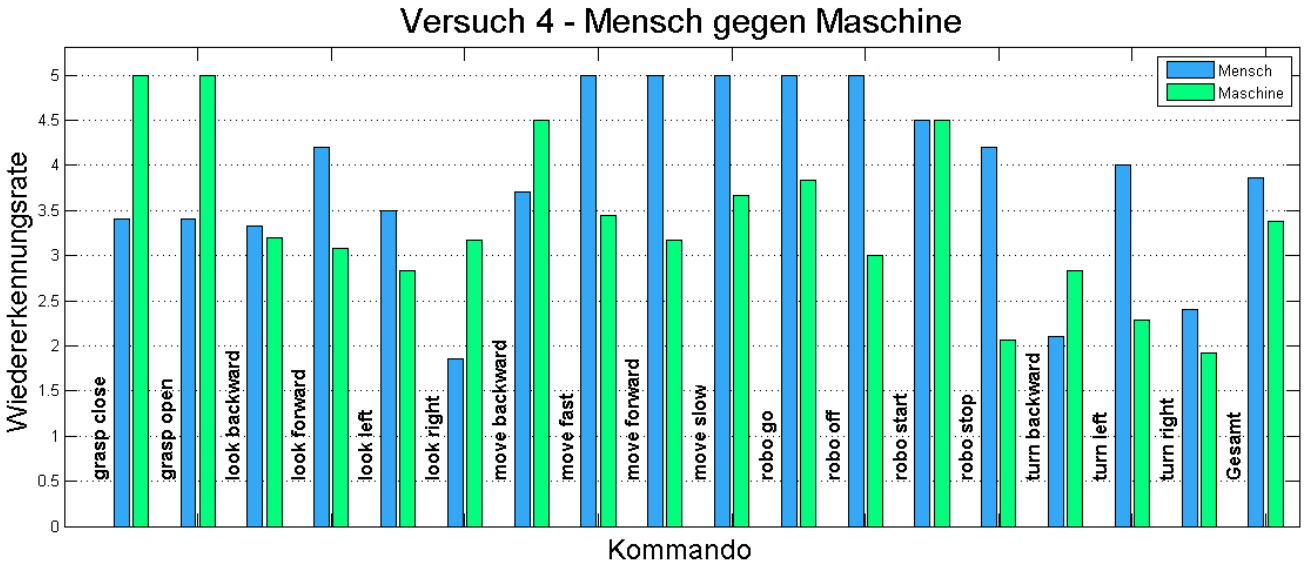


Abbildung 5.8: Wiedererkennungsraten: Mensch gegen Maschine

In der Abbildung 5.8 sind die Wiedererkennungsraten des Versuchs zu sehen. Insgesamt ist die Wiedererkennungsrate der Person (3,86 von 5) höher als des visuellen Spracherkennungssystems (3,38 von 5). Die Wiedererkennungsrate der Person ist um 9,6 % (bei 5 Äußerungen je Kommando) höher als des visuellen Spracherkennungssystems. Auffällig ist, dass die Person Schwierigkeiten in der Klassifizierung der Wortgruppe „grasp close“ und „grasp open“ hat. Das visuelle Spracherkennungssystem kann diese Kommandos zuverlässig klassifizieren, da sie sich stark von den anderen Kommandos unterscheiden. In der Auswertung der Wiedererkennungsraten wurde deutlich, dass die Person „turn right“ oft als „look right“ und umgekehrt klassifizierte. Diese fehlerhafte Klassifizierung führt ebenfalls das visuelle Spracherkennungssystem aus (siehe Versuch 2: 5.1.2). Der Grund ist, dass diese Wortkombinationen von der Lippenbewegung ähnlich verlaufen.

5.2 Bewertung

Im folgenden Abschnitt werden die durchgeführten Versuche bewertet und Erkenntnisse aus ihnen gezogen. Die höchste Wiedererkennungsrate, welche mit WDTW erreicht wurde, ist 37,74 (siehe Versuch 2: Parametereinstellung 6). Das ist eine Wiedererkennung von 75,48 % bei 50 Äußerungen je Kommando. Eine höhere Wiedererkennung kann nicht erreicht werden, da nur die Lippenbewegungen erfasst werden. Der Mundhohlraum wird nicht betrachtet. Der Mundhohlraum ist, durch die Zunge, Rachen und Zähne, stark beteiligt in der Bildung von Sprache. Das Problem wird in der Machbarkeitsanalyse diskutiert (Abschnitt 3.3). Ziel des visuellen Spracherkennungssystems ist es jedoch als ein unterstützendes System für ein akustisches Spracherkennungssystem eingesetzt zu werden. Aus diesem Grund ist die Wiedererkennungsrate ausreichend.
In Versuch 1 werden Cluster gebildet, um Repräsentanten zu finden. Bei hohen k, d.h. einer hohen Anzahl von Repräsentanten, ist der Zeitaufwand erheblich höher als bei niedrigen k. Die Suche nach Repräsentanten muss jedoch nur einmalig geschehen, so dass das Training beispielsweise über Nacht durchgeführt werden kann.
Die Kommandos „turn left“, „turn right“ und „look left“ ähneln sich in der Lippenbewegung (siehe Versuch 2: 5.1.2). Aus diesem Grund sind die Wiedererkennungsraten der Kommandos niedriger als der Mittelwert. Um den Merkmalsvektor zu erweitern und somit die Wiedererkennungsrate zu erhöhen, kann der Mundhohlraum genutzt werden. Die Zunge ist in einigen Äußerungen zu sehen, so kann beispielsweise der Mittelwert der Pixelgrauwerte der Zunge/des Mundhohlraums als ein weiteres Merkmal genutzt werden. Die genannten Kommandos werden dadurch besser unterschieden.
Im vierten Versuch hat der Mensch eine höhere Wiedererkennungsrate als das visuelle Spracherkennungssystem erreicht. Dies liegt daran, da der Mensch geübter ist und mehr vom Mund wahrnehmen kann als das System. Die höhere Wiedererkennungsrate liegt ebenfalls daran, da der Mensch die Bewegungen der Zunge für das Lippenlesen ergänzend miteinbeziehen kann.
WDTW erhöht die Wiedererkennungsrate. Wird ein „Warping-Fenster“ benutzt, so erhöht sich die Winderkennungsrate nur minimal (Grund siehe Versuch 2: 5.1.2). Das euklidische Distanzverfahren hat, wie erwartet, eine deutlich geringere Wiedererkennungsrate als (W)DTW, da die beim euklidischen Distanzverfahren die Variationen in der Zeit nicht berücksichtigt werden. Vor- und Nachteile des Verfahrens werden im Abschnitt „Dynamic-time-warping vs. Euklidische Distanz“ 4.5.1 behandelt.

5.3 Notwendige Erweiterungen

Es werden notwendige/mögliche Erweiterungen für das visuelle Spracherkennungssystem diskutiert. Es werden die Erkenntnisse und die Ergebnisse der Versuche genommen, um zu beurteilen, weshalb Erweiterungen erforderlich sind. Schwierigkeiten in der Realisierung des Systems werden ebenfalls mit berücksichtigt. Die Darstellung erfolgt anhand der Prozesskette (siehe Abschnitt 2.1.2):
Die Lippen können nur zuverlässig lokalisiert werden, wenn das Gesicht frontal zur Kamera gerichtet ist. Wenn das Gesicht nach rechts, links, unten oder nach oben gewendet wird, so entstehen fehlerhafte Merkmalsvektoren, da die Lippenmerkmale nicht korrekt gefunden werden. Die verwendete Merkmalsextraktion (Pseudo-Hue und Lichtbilder) ist störanfällig bei starker Sonnenstrahlung. Die Eigenschaft, dass die Lippenecken in dunklen Bereichen liegen, wenn eine Lichtquelle von oben nach unten strahlt, wird aufgehoben. Aus den genannten Gründen ist ein zuverlässiges und robustes Tracking (Verfolgen) der Lippen notwendig. Dies kann durch ein modellbasiertes Tracking erreicht werden. Für dieses Verfahren wird ein Modell der Lippen aufgestellt. Aus dem Modell können die sechs Schlüsselpunkte der Lippen extrahiert werden. Durch Bildung des Modells können weitere Merkmale genutzt werden, wie beispielsweise verschiedene Winkel zwischen der Ober- und Unterlippe. Dies kann die Wiedererkennung erhöhen.
Für ein zuverlässiges Lippenlesen und Erhöhung der Wiedererkennungsrate ist es notwendig den Mundhohlraum, besonders die Zunge als Merkmal zu erfassen. Ein Modell der Zunge ist schwer realisierbar, da die Zunge während der Äußerung zum größten Teil verdeckt ist. Für die Erweiterung können bildbasierte Verfahren eingesetzt werden. Zunächst kann der Bereich der Mundhöhle durch modellbasierte Verfahren zuverlässig lokalisiert werden. Nachdem der Bereich lokalisiert wurden, werden die durch bildbasierte Verfahren die Pixelgrauwerte des Bereiches gespeichert (beispielsweise durch Mittelwertbildung).
Für die Klassifikation können andere Verfahren eingesetzt werden, jedoch ist (W)DTW zuverlässig genug, um Zeitvariationen auszugleichen und Trajektorien (also Äußerungen) miteinander zu vergleichen. Wie bereits beschrieben ist ein zuverlässiges und robustes Verfolgen der Lippen und Erweiterung des Merkmalraumes notwendig für eine Steigerung der Wiedererkennungsrate.

6 Schluss

6.1 Zusammenfassung

Die Arbeit beschreibt die Entwicklung eines visuellen Spracherkennungssystems. Ziel der Arbeit ist es einen funktionsfähigen Prototyp zu realisieren. In den Grundlagen wurde die Prozesskette zur Entwicklung eines visuellen Spracherkennungssystems vorgestellt. Die Analyse stellte fest, dass in der vorliegenden Arbeit bildbasierte Methoden für die Merkmalsextraktion benutzt werden. Modellbasierte Verfahren sind komplex in der Realisierung, benötigen ein großes Set an Trainingsdaten und viel Rechenleistung. Die Machbarkeitsanalyse zeigt, dass eine kontinuierliche visuelle Spracherkennung schwer realisierbar ist. In der Arbeit wird daher ein holistischer Ansatz genutzt, d.h. es werden Wörter als Ganzes betrachtet und wiedererkannt. Aus der Machbarkeitsanalyse geht ebenfalls hervor, dass das System eine Unterstützung für eine akustische Spracherkennung sein soll. In der Prozesskettenanalyse werden bildbasierte Methoden vorgestellt aus denen Techniken für das eigene Vorgehen verwendet werden. Die Mund-*ROI* wird durch die anatomische Eigenschaft, dass sich der Mund im unteren Teil des Gesichts befindet, berechnet. Verschiedene Lippenmerkmale werden vorgestellt und entschieden, dass das Seitenverhältnis und die Fläche der Lippen genutzt werden. Werden die Merkmale bei einer Äußerung eines Kommandos über die Zeit betrachtet, entstehen Merkmalsvektoren. Ein Merkmalsvektor kann als Trajektorie betrachtet werden. Durch DTW können zwei Äußerungen, also Trajektorien miteinander verglichen werden. Die Ähnlichkeit beider Trajektorien wird gemessen, um zu klassifizieren welches Kommando geäußert wurde. Im Kapitel Lippenlesen wird gezeigt, wie die Merkmale der Lippen extrahiert werden. Die Lippenbilder werden vor verarbeitet, indem Gradientenbilder erzeugt werden. Die Gradientenbilder betonen Eigenschaften (Ecken der Lippen, oberen Teil der Oberlippe, unteren Teil der der Unterlippe) der Lippen, um die Merkmale zuverlässiger zu extrahieren. Das Suchverfahren in den Gradientenbildern wird vorgestellt. Durch Gewichtungen wird das DTW erweitert. Dies ist im Fall, wenn kein „Warping-Fenster“ genutzt wird, notwendig, da Äußerungen unterschiedlich lang sein können. Eine Äußerung wird zu einem Kommando klassifiziert, indem die Ähnlichkeit der Repräsentanten je Kommando und der Äußerung durch DTW gemessen wird. Aus einem

Trainingsset werden Repräsentanten gefunden. Es existiert je Kommando ein Trainingsset. Die Repräsentanten werden durch den Clustering-Algorithmus k-Medoids gefunden. In der Auswertung wird gezeigt, dass das visuelle Spracherkennungssystem Kommandos mit einer zuverlässigen Wiedererkennungsrate erkennt. In einem Test wird deutlich, dass der Mensch Lippen genauer/zuverlässiger lesen kann als das System, da der Mensch den Mundhohlraum, besonders die Zunge für die Klassifizierung mit einbezieht. Abschließend kann festgestellt werden, dass ein visuelles Spracherkennungssystem entwickelt wurde, welches verbundene Wörter zuverlässig erkennt und unterstützend für ein akustisches Spracherkennungssystem eingesetzt werden kann.

6.2 Ausblick

In dieser Arbeit wird ein Prototyp entwickelt. Die Realisierung läuft entlang der Prozesskette, die in den Grundlagen vorgestellt wird. Der Prototyp kann verbessert werden, indem die einzelnen Prozessschritte erweitert bzw. weiterentwickelt werden. Der erste Schritt wäre ein robustes Verfolgen der Lippen zu ermöglichen, beispielsweise durch modellbasierte Ansätze. Das robuste Verfolgen der Lippen stellt eine eigene Studie dar. Die Erweiterung des Merkmalsvektors muss ebenfalls betrachtet werden, da eine Erhöhung der Wiedererkennungsrate dadurch möglich ist. Eine zuverlässigere Klassifikation von Äußerungen ist möglich, wenn das Trainingsset vergrößert wird. Das Trainingsset je Kommando muss verschiedene Variationen dieses Kommandos beinhalten. Es müssen also viele Äußerungen aufgezeichnet werden, indem ein Kommando schneller, langsamer, von verschiedenen Sprechern, undeutlich, mit Akzent und mit Dialekt aufgesagt wird. Beim Clustering werden dann aussagekräftigere Repräsentanten gefunden. Die Vision ist, dass ein System entwickelt wird, welches kontinuierliche Sprache ausschließlich durch das Visuelle wiedererkennt. Hierzu ist es notwendig brauchbare visuelle Spracheinheiten zu finden und zu formalisieren.

Literaturverzeichnis

[Al-Ghanim u. a. 2013] Al-Ghanim, A. ; AL-Oboud, N. ; Al-Haidary, R. ; Al-Zeer, S. ; Altammami, S. ; Mahmoud, H.A.: I See What You Say (ISWYS): Arabic lip reading system. In: *Current Trends in Information Technology (CTIT), 2013 International Conference on*, Dec 2013, S. 11–17

[Ali u. a. 2014] Ali, B. B. ; Masmoudi, Y. ; Dhouib, S.: Tabu search for dynamic time warping global constraint learning. In: *Soft Computing and Pattern Recognition (SoCPaR), 2014 6th International Conference of*, Aug 2014, S. 376–381

[Alizadeh u. a. 2008] Alizadeh, S. ; Boostani, R. ; Asadpour, V.: Lip feature extraction and reduction for HMM-based visual speech recognition systems. In: *Signal Processing, 2008. ICSP 2008. 9th International Conference on*, Oct 2008, S. 561–564

[Altman 1992] Altman, N. S.: An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. In: *American Statistician* 46 (1992), S. 175–185

[Anusuya und Katti 2010] Anusuya, M. A. ; Katti, S. K.: Speech Recognition by Machine, A Review. In: *CoRR* abs/1001.2267 (2010). – URL http://arxiv.org/abs/1001.2267

[Bacivarov u. a. 2008] Bacivarov, I. ; Ionita, M.C. ; Corcoran, P.: A combined approach to feature extraction for mouth characterization and tracking. In: *Signals and Systems Conference, 208. (ISSC 2008). IET Irish*, June 2008, S. 156–161. – ISSN 0537-9989

[Bangsawan u. a. 2015] Bangsawan, H.T. ; Mardiyanto, R. ; Sardjono, T.A.: Six key points lip's feature extraction using adaptive threshold segmentation. In: *Intelligent Technology and Its Applications (ISITIA), 2015 International Seminar on*, May 2015, S. 261–266

[Baud u. a. 2007] Baud, O. ; El-Bied, Y. ; Honore, N. ; Taupin, O.: Trajectory Comparison for Civil Aircraft. In: *Aerospace Conference, 2007 IEEE*, March 2007, S. 1–9. – ISSN 1095-323X

[Bray u. a. 2008] Bray, Tim ; Paoli, Jean ; Sperberg-McQueen, C. M. ; Maler, Eve ; Yergeau, François: *Extensible Markup Language (XML) 1.0 (Fifth Edition)*. World Wide Web Consortium, Recommendation REC-xml-20081126. November 2008

[Cartas-Ayala 2002] Cartas-Ayala, Alejandro: *Gradient Vector Flow Snakes.* 2002. – URL http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV1011/cartas.pdf. – Zugriffsdatum: 05.01.2016

[Cattell 2011] Cattell, Rick: Scalable SQL and NoSQL Data Stores. In: *SIGMOD Rec.* 39 (2011), Mai, Nr. 4, S. 12–27. – URL http://doi.acm.org/10.1145/1978915.1978919. – ISSN 0163-5808

[Çeliktutan u. a. 2013] Çeliktutan, Oya ; Ulukaya, Sezer ; Sankur, Bülent: A Comparative Study of Face Landmarking Techniques. In: *EURASIP J. Image and Video Processing* 2013 (2013), S. 13. – URL http://dx.doi.org/10.1186/1687-5281-2013-13

[Chetty und Wagner 2004] Chetty, Girija ; Wagner, Michael: Automated lip feature extraction for liveness verification in audio-video authentication. In: *Proceedings of Image and Vision Computing* (2004), S. 17–22

[Chin u. a. 2012] Chin, Siew W. ; Seng, Kah P. ; Ang, Li-Minn: Lips Contour Detection and Tracking Using Watershed Region-Based Active Contour Model and Modified H_∞. In: *Circuits and Systems for Video Technology, IEEE Transactions on* 22 (2012), June, Nr. 6, S. 869–874. – ISSN 1051-8215

[Chowdhury u. a. 2015] Chowdhury, S. ; Tom, M. ; Verma, B. ; Zhang, Mengjie: Pixel characteristics based feature extraction approach for roadside object detection. In: *Neural Networks (IJCNN), 2015 International Joint Conference on*, July 2015, S. 1–8

[Cootes u. a. 2000] Cootes, Tim ; Baldock, ER ; Graham, J: An introduction to active shape models. In: *Image processing and analysis* (2000), S. 223–248

[Cootes u. a. 2001] Cootes, Timothy F. ; Edwards, Gareth J. ; Taylor, Christopher J.: Active appearance models. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* (2001), Nr. 6, S. 681–685

[Cootes und Taylor 1992] Cootes, Timothy F. ; Taylor, Christopher J.: Active shape models - Smart Snakes. In: *BMVC92*. Springer London, 1992, S. 266–275

[Criel und Tsiporkova 2006] Criel, J ; Tsiporkova, Elena: Gene Time Eχpression Warper: a tool for alignment, template matching and visualization of gene expression time series. In: *BIOINFORMATICS* 22 (2006), Nr. 2, S. 251–252. – URL http://dx.doi.org/10.1093/bioinformatics/bit787; http://www.mathcs.emory.edu/~lxiong/cs730_s13/share/slides/searching_sigkdd2012_DTW.pdf. – ISSN 1367-4803

[D'Urso 2000] D'Urso, Pierpaolo: Dissimilarity measures for time trajectories. In: *Journal of the Italian Statistical Society* 9 (2000), Nr. 1-3, S. 53–83. – URL http://dx.doi.org/10.1007/BF03178958. – ISSN 1121-9130

[Echtler u. a. 2016] Echtler, Florian ; Kerl, Christian ; Xiang, Lingzhu ; Wiedemeyer, Thiemo ; Lars ; Gordon, Ryan ; hanyazou ; laborer2008 ; Wareham, Rich ; Goldhoorn, Matthias ; Facioni, Francisco ; gaborpapp ; alberth ; Fuchs, Steffen ; Federico ; jmtatsch ; Blake, Joshua ; Jungkurth, Henning ; Mingze, Yuan ; vinouz ; Coleman, Dave ; Rawat, Rahul ; Reynolds, Paul ; Viau, P.E. ; Ludique ; Alistair ; Billingham, James: *libfreenect2: Release 0.1.1.* Januar 2016. – URL http://dx.doi.org/10.5281/zenodo.45314

[Fangerau u. a. 2012] Fangerau, J. ; Höckendorf, B. ; Wittbrodt, J. ; Leitte, H.: Similarity analysis of cell movements in video microscopy. In: *Biological Data Visualization (BioVis), 2012 IEEE Symposium on*, Oct 2012, S. 69–76

[Farooque und Rohankar 2013] Farooque, Mohd A. ; Rohankar, Jayant S.: Survey on various noises and techniques for denoising the color image. In: *International Journal of Application or Innovation in Engineering & Management (IJAIEM)* 2 (2013), Nr. 11, S. 217–221

[Gao u. a. 2010] Gao, Xinbo ; Su, Ya ; Li, Xuelong ; Tao, Dacheng: A review of active appearance models. In: *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 40 (2010), Nr. 2, S. 145–158

[Halkidi u. a. 2001] Halkidi, M. ; Batistakis, Y. ; Vazirgiannis, M.: Clustering algorithms and validity measures. In: *Scientific and Statistical Database Management, 2001. SSDBM 2001. Proceedings. Thirteenth International Conference on*, 2001, S. 3–22. – ISSN 1099-3371

[Hassanat 2014] Hassanat, Ahmad B. A.: Visual Speech Recognition. In: *CoRR* abs/1409.1411 (2014). – URL http://arxiv.org/abs/1409.1411

[Hauswald u. a. 2015] Hauswald, J. ; Kang, Yiping ; Laurenzano, M.A. ; Chen, Quan ; Li, Cheng ; Mudge, T. ; Dreslinski, R.G. ; Mars, J. ; Tang, Lingjia: DjiNN and Tonic: DNN as a service and its implications for future warehouse scale computers. In: *Computer Architecture (ISCA), 2015 ACM/IEEE 42nd Annual International Symposium on*, June 2015, S. 27–40

[Hazen 2006] Hazen, T.J.: Visual model structures and synchrony constraints for audio-visual speech recognition. In: *Audio, Speech, and Language Processing, IEEE Transactions on* 14 (2006), May, Nr. 3, S. 1082–1089. – ISSN 1558-7916

[Hu 1962] Hu, Ming-Kuei: Visual pattern recognition by moment invariants. In: *information Theory, IRE Transactions on* 8 (1962), Nr. 2, S. 179–187

[Husain 2011] Husain, Benafsh N.: Face Detection And Lip Localization. (2011)

[itseez 2016] ITSEEZ: *OpenCV*. 2016. – URL http://opencv.org/. – Zugriffsdatum: 25.01.2016

[Kass u. a. 1988] KASS, Michael ; WITKIN, Andrew ; TERZOPOULOS, Demetri: Snakes: Active contour models. In: *INTERNATIONAL JOURNAL OF COMPUTER VISION* 1 (1988), Nr. 4, S. 321–331

[Keogh und Pazzani 2001] KEOGH, Eamonn J. ; PAZZANI, Michael J.: Derivative Dynamic Time Warping. In: *In First SIAM International Conference on Data Mining SDM 2001*, 2001

[McGurk und MacDonald 1976] MCGURK, Harry ; MACDONALD, John: Hearing lips and seeing voices. In: *Nature* 264 (1976), 12, S. 746–748

[Meisel 2016] MEISEL, Andreas: *RV Robot Vision - Merkmalsextraktion.* 2016

[Microsoft 2016] MICROSOFT: *Kinect.* https://support.xbox.com/de-DE/browse/xbox-one/accessories/Kinect. 2016. – Zugriffsdatum: 15.02.2016

[MongoDB 2016] MONGODB, Inc.: *MongoDB.* https://www.mongodb.com. 2016. – Zugriffsdatum: 15.02.2016

[Müller 2007] MÜLLER, Meinard: *Information Retrieval for Music and Motion.* Secaucus, NJ, USA : Springer-Verlag New York, Inc., 2007. – ISBN 3540740473

[Ooi u. a. 2008] OOI, Wei C. ; JEON, Changwon ; KIM, Kihyeon ; HAN, D.K. ; KO, Hanseok: Effective lip localization and tracking for achieving multimodal speech recognition. In: *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, Aug 2008, S. 90–93

[Paliwal u. a. 1982] PALIWAL, K.K. ; AGARWAL, Anant ; SINHA, Sarvajit S.: A modification over Sakoe and Chiba's dynamic time warping algorithm for isolated word recognition. In: *Signal Processing* 4 (1982), Nr. 4, S. 329–333. – URL http://www.sciencedirect.com/science/article/pii/0165168482900093. – ISSN 0165-1684

[Quigley u. a. 2009] QUIGLEY, Morgan ; CONLEY, Ken ; GERKEY, Brian P. ; FAUST, Josh ; FOOTE, Tully ; LEIBS, Jeremy ; WHEELER, Rob ; NG, Andrew Y.: ROS: an open-source Robot Operating System. In: *ICRA Workshop on Open Source Software*, 2009

[Rabiner und Juang 1993] RABINER, Lawrence ; JUANG, Biing-Hwang: *Fundamentals of Speech Recognition.* Upper Saddle River, NJ, USA : Prentice-Hall, Inc., 1993. – ISBN 0-13-015157-2

[ur Rehman Butt und Lombardi 2013] Rehman Butt, W. ur ; Lombardi, L.: A survey of automatic lip reading approaches. In: *Digital Information Management (ICDIM), 2013 Eighth International Conference on*, Sept 2013, S. 299–302

[Robotics 2016] Robotics, Clearpath: *ROS 101: Intro to the Robot Operating System.* 2016. – URL http://robohub.org/ros-101-intro-to-the-robot-operating-system/. – Zugriffsdatum: 15.02.2016

[Rujasiri und Chomtee 2009] Rujasiri, Piyatida ; Chomtee, Boonorm: Comparison of clustering techniques for cluster analysis. In: *Nat. Sci* 43 (2009), S. 378–388

[Saitoh u. a. 2008] Saitoh, T. ; Morishita, K. ; Konishi, R.: Analysis of efficient lip reading method for various languages. In: *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, Dec 2008, S. 1–4. – ISSN 1051-4651

[Sakoe und Chiba 1978] Sakoe, H. ; Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26 (1978), Feb, Nr. 1, S. 43–49. – ISSN 0096-3518

[Salvadore und Chan 2004] Salvadore, S. ; Chan, P.: FastDTW: Toward accurate dynamic time warping in linear time and space. In: *3rd Workshop on Mining Temporal and Sequential Data*, 2004

[Sandhya und VijayKumar 2013] Sandhya, K. ; VijayKumar, K.: Classification of ECG Heart beats using Dynamic Time Warping. In: *INTERNATIONAL JOURNAL FOR DEVELOPMENT OF COMPUTER SCIENCE & TECHNOLOGY* 1 (2013), July-Aug, Nr. 7

[ScanDig 2016] ScanDig: *Glossar Scanner, Digitalkameras, Bildbearbeitung.* 2016. – URL http://www.filmscanner.info/Glossar_H.html,http://www.scandig.info/Linsen.html. – Zugriffsdatum: 06.03.2016

[Shaikh u. a. 2013] Shaikh, Ayaz A. ; Kumar, Dinesh K. ; Gubbi, Jayavardhana: Automatic Visual Speech Segmentation and Recognition Using Directional Motion History Images and Zernike Moments. In: *Vis. Comput.* 29 (2013), Oktober, Nr. 10, S. 969–982. – URL http://dx.doi.org/10.1007/s00371-012-0751-7. – ISSN 0178-2789

[Sharifara u. a. 2014] Sharifara, A. ; Mohd Rahim, M.S. ; Anisi, Y.: A general review of human face detection including a study of neural networks and Haar feature-based cascade classifier in face detection. In: *Biometrics and Security Technologies (ISBAST), 2014 International Symposium on*, Aug 2014, S. 73–78

[Sharifi u. a. 2002] Sharifi, M. ; Fathy, M. ; Mahmoudi, M. T.: A classified and comparative study of edge detection algorithms. In: *Information Technology: Coding and Computing, 2002. Proceedings. International Conference on*, April 2002, S. 117–120

[Shin u. a. 2011] Shin, Jongju ; Lee, Jin ; Kim, Daijin: Real-time Lip Reading System for Isolated Korean Word Recognition. In: *Pattern Recogn.* 44 (2011), März, Nr. 3, S. 559–571. – URL http://dx.doi.org/10.1016/j.patcog.2010.09.011. – ISSN 0031-3203

[Shutler 2002] Shutler, Jamie: *Statistical Moments*. 2002. – URL http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/SHUTLER3/CVonline_moments.html. – Zugriffsdatum: 01.03.2016

[Si u. a. 2009] Si, Jennie ; Yang, Lei ; Lu, Chao ; Sun, Jian ; Mei, Shengwei: Approximate dynamic programming for continuous state and control problems. In: *Control and Automation, 2009. MED '09. 17th Mediterranean Conference on*, June 2009, S. 1415–1420

[Sui u. a. 2013] Sui, Chao ; Bennamoun, M. ; Togneri, R. ; Haque, S.: A lip extraction algorithm using region-based ACM with automatic contour initialization. In: *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, Jan 2013, S. 275–280. – ISSN 1550-5790

[Sujatha und Krishnan 2012] Sujatha, P. ; Krishnan, M.R.: Lip feature extraction for visual speech recognition using Hidden Markov Model. In: *Computing, Communication and Applications (ICCCA), 2012 International Conference on*, Feb 2012, S. 1–5

[Sujatha und M.Radhakrishnan 2013] Sujatha, P. ; M.Radhakrishnan, M.: SPEAKER - INDEPENDENT VISUAL LIP ACTIVITY DETECTION FOR HUMAN - COMPUTER INTERACTION. In: *IJRET: International Journal of Research in Engineering and Technology*, Nov 2013

[Theodoridis und Koutroumbas 2008] Theodoridis, Sergios ; Koutroumbas, Konstantinos: *Pattern Recognition, Fourth Edition*. 4th. Academic Press, 2008. – ISBN 1597492728, 9781597492720

[Tsuhan 2001] Tsuhan, Chen: Audiovisual Speech Processing: Lip Reading and Lip Synchronization. In: *Signal Processing Magazine, IEEE* 18 (2001), Jan, Nr. 1, S. 9–21. – ISSN 1053-5888

[Viola und Jones 2001] Viola, Paul ; Jones, Michael: Rapid object detection using a boosted cascade of simple features, 2001, S. 511–518

[VisualPharm 2016] VisualPharm: *Icons8*. https://icons8.com. 2016. – Zugriffsdatum: 01.01.2016

[WenJuan u. a. 2010] WenJuan, Yao ; YaLing, Liang ; Minghui, Du: A real-time lip localization and tacking for lip reading. In: *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on* Bd. 6, Aug 2010, S. V6–363–V6–366. – ISSN 2154-7491

[Werda u. a. 2006] Werda, S. ; Mahdi, W. ; Tmar, M. ; Hamadou, A.B.: ALiFE: Automatic Lip Feature Extraction: A New Approach for Speech Recognition Application. In: *Information and Communication Technologies, 2006. ICTTA '06. 2nd* Bd. 2, 2006, S. 2963–2968

[Wiedemeyer 2014 – 2015] Wiedemeyer, Thiemo: *IAI Kinect2*. https://github.com/code-iai/iai_kinect2. 2014 – 2015. – Zugriffsdatum: 10.01.2016

[wissen.de 2016] wissen.de: *Invarianz*. http://www.wissen.de/fremdwort/invarianz. 2016. – Zugriffsdatum: 26.03.2016

[Yanagisawa und Satoh 2006] Yanagisawa, Y. ; Satoh, T.: Clustering Multidimensional Trajectories based on Shape and Velocity. In: *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on*, 2006, S. 12–12

[Yargic und Dogan 2013] Yargic, A. ; Dogan, M.: A lip reading application on MS Kinect camera. In: *Innovations in Intelligent Systems and Applications (INISTA), 2013 IEEE International Symposium on*, June 2013, S. 1–5

[Yau u. a. 2006] Yau, Wai C. ; Kumar, Dinesh K. ; Arjunan, Sridhar P.: Voiceless Speech Recognition Using Dynamic Visual Speech Features. In: *Proceedings of the HCSNet Workshop on Use of Vision in Human-computer Interaction - Volume 56*. Darlinghurst, Australia, Australia : Australian Computer Society, Inc., 2006 (VisHCI '06), S. 93–101. – URL http://dl.acm.org/citation.cfm?id=1273385.1273401

[Yingjie u. a. 2011] Yingjie, Meng ; Haiyan, Zhang ; Yingjie, Hu ; Jinyang, Li: Lip information extraction based on the fusion of geometry and motion features. In: *Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on* Bd. 4, July 2011, S. 2186–2190

[Yoshida u. a. 2010] Yoshida, T. ; Nakadai, K. ; Okuno, H.G.: Two-layered audio-visual speech recognition for robots in noisy environments. In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, Oct 2010, S. 988–993. – ISSN 2153-0858

[Yu u. a. 2009] Yu, Dahai ; Ghita, Ovidiu ; Sutherland, Alistair ; Whelan, Paul F.: *Advances in Image and Video Technology: Third Pacific Rim Symposium, PSIVT 2009, Tokyo, Japan, January 13-16, 2009. Proceedings*. Kap. A Novel Visual Speech Representation and HMM

Classification for Visual Speech Recognition, S. 398–409. Berlin, Heidelberg : Springer Berlin Heidelberg, 2009. – URL http://dx.doi.org/10.1007/978-3-540-92957-4_35. – ISBN 978-3-540-92957-4

[Zhang u. a. 2014] Zhang, Z. ; Tang, L. ; Tang, P.: Local feature based dynamic time warping. In: *Data Science and Advanced Analytics (DSAA), 2014 International Conference on*, Oct 2014, S. 425–429

[Zhang u. a. 2006] Zhang, Zhang ; Huang, Kaiqi ; Tan, Tieniu: Comparison of Similarity Measures for Trajectory Clustering in Outdoor Surveillance Scenes. In: *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on* Bd. 3, 2006, S. 1135–1138. – ISSN 1051-4651

Abbildungsverzeichnis

Abkürzungsverzeichnis

bimodal ..... bi = zwei, modal = Art, Seite 6

LOI ..... Line of Interest: LOIs werden eingesetzt um den Übergang/Verlauf der Grauwerte in den Lippenbilder zu finden. Sie werden eingesetzt, um die Schlüsselpunkte des Kupidoboden, der Unterlippe und der Ecken der Lippen zu finden. Es werden die Kanten in den jeweiligen Gradientenbildern gesucht. Ein LOI ist senkrecht und hat eine bestimmte Pixellänge., Seite 77

Momente ..... Momente sind gewichtete Mittelwerte. Die Mittelwerte werden berechnet aus den Pixelintensitaten (Grau-/Farbwert). Die Mittelwerte werden genutzt, um ein Bild zu beschreiben (Shutler [2002])., Seite 14

POI ..... Point of Interest: Ein POI ist ein Punkt der in Frage kommen kann, um ein Schlüsselpunkt der Lippen zu sein. Ein POI hat x- und y-Koordinaten und befindet sich auf einem Bild., Seite 77

ROI ..... Region of Interest: Eine ROI ist ein Rechteck, wobei mindestens zwei Eckpunkte bekannt sein müssen oder ein Eckpunkt und die Breite und Höhe der ROI. ROI sind Bereiche, die in spezifischen Kontexten interessant sein können und wichtige Bildinformationen beinhalten., Seite 13

ROS ..... Robot Operating System: Software-Framework zur Bedienung/Steuerung von Robotern, Seite 8

Template ..... Eine Schablone (Template) eines Objekts wird erstellt, indem Trainingsdaten für die Modellierung genutzt werden. Es wird genutzt um ein Abgleich zu einem Objekt durchzuführen. , Seite 14

Versicherung über die Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 31. März 2016 Mosawer Ahmad Nurzai