Brandenburgische
Technische Universität
Cottbus - Senftenberg

LS mt lehrstuhl medientechnik

-Masterarbeit-

HMM-basiertes Analyse-Synthese-System zur Verarbeitung von 3D-Trajektorien

Brandenburgische technische Universität
Cottbus - Senftenberg

Lehrstuhl: Medientechnik

Name:	**Thomas Werner**
Matrikelnummer:	**3036956**
Betreuer:	**Prof. Dr.-Ing. habil. Christian Hentschel,** **M.Sc. Stephan Rogge,** **Dr.-Ing. Ronald Römer,**

Abgabeort:	**Cottbus**
Datum:	**25. August 2014**

Brandenburgische
Technische Universität
Cottbus - Senftenberg

LS mt lehrstuhl medientechnik
Lehrstuhl Medientechnik
Prof. Dr.-Ing. C. Hentschel

Masterarbeit

Name: Werner, Thomas

Matrikel: 3036956 **Ausgabe:** 17.03.2014

HMM-basiertes Analyse-Synthese-System zur Verarbeitung von 3D-Trajektorien

HMM-based analysis-synthesis-system for processing of 3D-trajectories

Im Bereich der Signalverarbeitung werden für Detektionsaufgaben häufig Hidden-Markov-Modelle (HMM) verwendet. Sie haben sich als zuverlässiges Werkzeug für die statische Schrift-, Sprach- und Gestenerkennung etabliert. Auf Basis von extrahierten Merkmalsvektoren werden verborgene Zustände mit entsprechenden Übergangs- und Emissionswahrscheinlichkeiten in einem Modell zusammengefasst, welches dann üblicherweise als Ergebnis einer sogenannten „Trainingsphase" vorliegt.
Ziel dieser Arbeit ist die Verwendung von HMM-basierten Algorithmen zur Erkennung von vorgegebenen Gesten auf dreidimensionalen Trajektorien zu untersuchen und exemplarisch ein Analyse-System mit Hilfe von Matlab zu implementieren. Um einen Überblick gängiger HMM-basierter Gestenerkennungsverfahren zu erlangen, ist eine Literarturrecherche durchzuführen. Sie soll unter anderem darüber Aufschluss geben, welche Merkmalsvektoren für die Analyse geeignet sind. Dies ist bei der Implementierung zu berücksichtigen. Zudem ist ein Synthese-System zu realisieren, das unter Verwendung der trainierten Modelle Trajektorien synthetisch nachzeichnet.
Zur Evaluierung der realisierten Algorithmen sollen vier vorgegebene Gesten für Training und Analyse verwendet werden. Hierzu sind in einem Laborversuch von mehreren Probanden mit Hilfe eines Trackingsystems 3D-Trajekorien aufzuzeichnen und zum einen Teil als Trainings- zum anderen Teil als Testdaten zu verwenden.

Teilaufgaben:

- Entwicklung und Implementierung von HMM-Analyse- und Trainings-Algorithmen zur Erkennung von 4 symboltragenden Trajektorien
- Entwicklung eines Datenformates zur permanenten Speicherung der aus Matlab erzeugten HMM, Trainings- und Testdaten
- Verwendung eines Garbage- / Filler sowie No-Grammar-Modells zur Filterung nicht-symboltragender Gesten (Robustheitserhöhung)
- Evaluierung (SNR-Analyse) der entwickelten Verfahren / Algorithmen mit Hilfe
    - Synthetischer Trainings- und Testdaten
    - Realer Trainings- und Testdaten aus einem Laborversuch

Betreuer: M. Sc. Stephan Rogge

Ch. Hentschel

Prof. Dr.-Ing. C. Hentschel

Zusammenfassung

Ziel dieser Arbeit ist die Erkennung von Gesten mit Hilfe von Hidden-Markov-Modellen im dreidimensionalen Raum. Die Grundlage bilden dabei im Labor für Virtual Reality,kurz „ivi“ genannt, aufgezeichnete Trajektorien des Lehrstuhls Medientechnik der BTU. Der verwendete Gestensatz umfasst die vier Symbole: „4“, „K“, „Kreis“ und „Dreieck“. Zu Beginn werden auf Basis des Merkmals der Orientierung Gesten synthetisch erzeugt. In einem anschließenden Erkennungsprozess erfolgt die Evaluation verschiedener HMM-Strukturen mit diskreter und kontinuierlicher Wahrscheinlichkeitsmodellierung. Verschiedene Verfahren zur Klassifikation werden dabei überprüft. Dafür notwendige Softwarekomponenten und Algorithmen werden mit Matlab unter Verwendung der HMM-Toolbox von Kevin Murphy (MIT, PMTK3) erstellt. Diese gewährleisten die Umsetzung von Synthese-, Trainings- und Analyseprozessen. Ein speziell hierfür entwickeltes XML-Format dient der permanenten Speicherung von Modellstrukturen und Parametern, sowie der erzeugten Trajektorien. Auf diese Weise wird die Möglichkeit einer späteren Reproduzierbarkeit der Arbeitsergebnisse und der dabei durchgeführten Schritte sichergestellt. Für eine erfolgreiche Erkennung der im Labor aufgezeichneten Daten, ist es notwendig, den Merkmalsvektor um die Distanz und die Geschwindigkeit zu erweitert. Des weiteren kommt eine modifizierte Modellstruktur zum Einsatz, die neben den Zuständen zur Bewegungsmodellierung zusätzlich Übergangszustände zwischen diesen enthält. Eine Evaluation dieser erfolgt in einem Laborversuch, bei dem von mehreren Probanden Gesten aufgezeichnet und anschließend erkannt werden. Darüber hinaus wird ein Garbage-Modell eingesetzt, dessen Aufgabe die Erkennung von nicht symboltragenden Bewegungen ist. In einem abschließenden Experiment wird ein Verfahren zur Online-Erkennung auf Basis des „Threshold-Modell-Ansatzes“ vorgestellt.

abstract

The aim of this work is the recognition of gestures in three-dimensional space with the help of Hidden Markov Models. As a basis serve the in the Virtual Reality Laboratory „ivi“ recorded trajectories of the department Media Technology at BTU. The used gesture set consists of four symbols: „4“, „K“, „circle“ and „triangle“. Based on the feature of the orientation gestures are produced synthetically at the beginning. In a subsequent recognition process, the evaluation of different HMM structures with discrete and continuous probability modeling is done. Besides, various methods of classification are checked. Necessary software components and algorithms are created with Matlab using the HMM toolbox of Kevin Murphy (MIT, PMTK3). These ensure the implementation of synthesis, training and analysis processes. A specially developed XML format is used for permanent storage of model structures and parameters, as well as the generated trajectories. In this way the possibility of subsequent reproducibility of results and the steps performed thereby is ensured. For detecting the data recorded in the laboratory, the feature vector is extended by the distance and the speed. Furthermore, a modified model structure is used, which encloses in addition to the conditions for motion modeling the transition states between these. Furthermore there is used a modified model structure which encloses, in addition to the states of motion modeling, the transition states between these. An evaluation of this takes place in a laboratory test in which the gestures of several test persons are recorded and detected afterwards. In addition, a garbage-model is used, whose task is the recognition of non-symbol-bearing movements. In a final experiment, a method for Online detection based on the „threshold model approach“ is presented.

Abkürzungen und Formelzeichen

α	Winkel Alpha
arccos	Arkuskosinus
arcsin	Arkussinus
arctan	Arkustangens
c	Gewichtung für Normalverteilung
ca	Auswertung: zutreffendes als richtig angenommen (engl. correct alarms)
cd	Auswertung: nicht zutreffendes als falsch angenommen (engl. correct dismissals)
CHMM	kontinuierliches → HMM (engl. continuous HMM)
cos	Kosinusfunktion
ϵ	Distanz Epsilon
DHMM	diskretes → HMM (engl. discrete HMM)
fa	Auswertung: nicht zutreffendes, als richtig angenommen (engl. false alarms)
fd	Auswertung: zutreffendes, als falsch angenommen (engl. false dismissals)
g	Verstärkungsfaktor (z.B.: Signal, Rauschen)
σ	Standardabweichung
θ	Winkel Theta
HMM	Hidden-Markov-Modell (Plural: HMMs)
Inf/ -Inf	Unendlich in positive/ negative Richtung (Matlab)
LR-Modelle	Links-Rechts-Modelle (engl. left right)
$M\{X\}$	Mischverteilung
$N\{X\}$	Normalverteilung
NaN	keine gültige Zahl (engl. not a number; Matlab: z.B.: Division durch 0)
$n.d.$	nicht definiert (z.B.: Division durch 0)
O	Menge der Observationen, auch als Emissionen bezeichnet
P	Wahrscheinlichkeit (engl. probability)
r	Vektorbetrag
S	Zustand bzgl. HMM (engl. state)
sin	Sinusfunktion
SL	math. Fenster zum Erfassen einer bestimmten Anzahl von Werten aus einer Sequenz (engl. sliding window)

SNR	Signal-Rausch-Verhältnis (engl. signal noise ratio)
tan	Tangensfunktion
TMA	Threshold-Modell-Ansatz
$Var\{X\}$	Varianz
x	x-Koordinate im kartesischen Raum
X	Zufallsvariable X
y	y-Koordinate im kartesischen Raum
z	z-Koordinate im kartesischen Raum

Abbildungsverzeichnis

Tabellenverzeichnis

Inhaltsverzeichnis

1 Einführung

Mit seiner enormen Vielfalt, seinem stetig wachsenden Anwendungspool und seinen sich schnell verändernden technischen Systemen fordert der Bereich rund um Multimedia nicht nur von Entwicklern ein hohes Maß an Anpassungsfähigkeit, sondern auch von seinen Nutzern. Vor allem Kurzlebigkeit und eine hohe Komplexität, meist in Verbindung mit einer unüberschaubaren Anzahl von Eingabe- und Steuermöglichkeiten, sowie einem hohen Konfigurationsaufwand, können zu einer Überforderung der Anwender führen. Um dem entgegenzuwirken, ist es notwendig, dass die Bedienungen von Anwendungen und technischen Systemen möglichst intuitiv, schnell erlernbar und allgemeingültig gestaltet werden.

1.1 Motivation

Als eigenes Forschungsfeld, mit einer Vielzahl von wissenschaftlichen Veröffentlichungen und einem stark zunehmenden Anwendungsfeld, stellt die Steuerung mit der Hand, als ein Teilgebiet der Erkennung von Gesten, eine praktische, einfache, berührungslose und natürliche Bedienmöglichkeit für immer komplexer werdende Aufgaben dar [CRR11]. So sind bereits in den Bereichen der Spielindustrie, Automobilbrache, „Virtual Reality“, Robotersteuerung oder „smart home“ Anwendungen dieser zu finden. Als Beispiele seien hier „Easy Open“ , „Kinect“ oder „LEAP“ genannt.
Neben dem hohen Potenzial für intuitive Mensch-Maschine-Schnittstellen gibt es bei der Umsetzung eine Vielzahl von technischen Problemen zu berücksichtigen und zu bewältigen, damit ein zuverlässiger Einsatz dieser Technologie ermöglicht werden kann. Als Hauptprobleme werden die Unterscheidung von Gesten und nicht-symboltragenden Bewegungen aus einem kontinuierlichen Eingabedatenstrom, sowie die unterschiedliche Ausführung einer Geste durch ein und die selbe Person genannt. Letzteres hat in Verbindung mit mehreren Nutzern eine hohe Variation in Ausführungsgeschwindigkeit, Form, Größe und Lokalität jeder einzelnen Geste zur Folge [Elm10].
Zusätzlich müssen Position und Ausrichtung des Anwenders berücksichtigt werden. Betrachtet man noch die Dimension des Raumes, in dem Bewegungen ausgeführt werden, so nimmt die Komplexität der Erkennung bei dem Übergang von 2D zu 3D weiter zu, da die Rotation und die Raumtiefe als weitere Einflussfaktoren hinzukommen.
Als Folge der steigenden technischen und mathematischen zu erfüllenden Anforderungen ist das Thema Gestenerkennung zunehmend in den Bereich des „soft computing“ verschoben worden, was bedeutet, dass die Algorithmen Probleme nicht aufgrund von harten Entscheidungen lösen, sondern auf statistische Methoden zurückgreifen [CRR11]. Eine dieser sind die Hidden-Markov-Modelle, welche den Schwerpunkt der hier vorliegenden Ausarbeitung bilden. Ziel ist es, die bisher verwendeten Strukturen und Merkmalsrepräsentationen dieser Modelle zu optimieren, um die Erkennungsraten im zweidimensionalen und im dreidimensionalen Raum für hochauflösende Systeme zu verbessern.

1.2 Aufbau der Arbeit

Zunächst erfolgt die Darstellung der wesentlichen Struktur eines Mustererkennungsprozesses. Anschließend wird ein Rückblick auf die bisher genutzten Verfahren und Ansätze dazu gegeben. Mit der Konzentration auf Hidden-Markov-Modelle wird dabei erläutert, wie diese strukturell aufgebaut sind, welche Merkmale für die Beschreibung von Gesten aus Bewegungsabläufen extrahiert werden, wie die Umsetzung im dreidimensionalen Raum realisiert wird und wie eine Unterscheidung zwischen Gesten und beliebigen Bewegungen erfolgt. Aufbauend auf den daraus gewonnen Erkenntnissen erfolgt die Formulierung eines eigenen Ansatzes, der durch Modifikation bekannter Methoden eine Optimierung in Form von steigenden Erkennungsraten erreicht.
Inhaltlich wird dieser in den zwei Abschnitten Synthese und Analyse behandelt. Ein erstes Vorexperiment dient dazu, eine Evaluation der verwendeten Algorithmen und eine Bestätigung der eingesetzten Modellstrukturen und Parameter auf der Basis von künstlich erzeugten Daten zu liefern.
Auf diese folgend wird in einem zweiten Vorexperiment die Eignung des Systems für die Erkennung von Menschen ausgeführten Gesten überprüft. Es wird gezeigt, warum dabei eine Anpassung der verwendeten Merkmale und Parameter nötig ist und welche zusätzlichen Aspekte bei der Erkennung der von real ausgeführten Bewegungen berücksichtigt werden müssen. Dabei getroffene Annahmen werden schließlich mittels von Probanden aufgezeichneten Bewegungen evaluiert.
In einem abschließenden Szenario werden noch einmal alle in dieser Arbeit gewonnen Erkenntnisse zusammengefasst und am Beispiel einer auf Hidden-Markov-Modellen basierenden Umsetzung, unter Anwendung des Threshold-Modell-Ansatzes, demonstriert.

Ferner wird die Bezeichnung „Hidden-Markov-Modell“, in Anlehnung an die gängige Schreibweise wissenschaftlicher Publikationen, mit „HMM“, bzw. „HMMs “ (Plural) abgekürzt.

2 Grundlagen

Bei der Erkennung von Gesten handelt sich um ein Mustererkennungsproblem, dessen Lösung eine Reihe von Einzelschritten unterschiedlicher Komplexität beinhaltet. Die wesentlichen für das Verständnis dieser Arbeit notwendigen theoretischen Aspekte dazu werden im Folgenden genannt.

2.1 Prozess der Gestenerkennung

Eine gute Beschreibung zu diesem Thema ist unter [Ric11] zu finden. Demnach ist der Ausgangspunkt eine Datenquelle bzw. ein Eingabedatenstrom. In der Regel wird auf diese eine Vorverarbeitung vorgenommen, um die Daten hinsichtlich bestimmter für den Musterkennungsprozess relevanter Faktoren zu optimieren und so eine bessere Unterscheidung von Nutzsignal und eventuellen Störsignalanteilen zu erreichen.
Mit der Annahme, bei dem Eingangssignal handle es sich um die Repräsentation eines speziellen Musters, nimmt die Segmentierung eine disjunkte Einteilung zusammengehöriger Merkmale vor. Als Beispiel sei hier die Unterteilung eines vorliegenden Bildes in Regionen gleicher Farben genannt [Ric11].
Ist die Detektion der Hand im Eingabestrom abgeschlossen, erfolgt die Merkmalsextraktion. Diese dient dazu, die vorliegenden Daten auf eine geeignete Repräsentationsform zu abstrahieren, welche das gesuchte Muster in seinen Eigenschaften möglichst treffend beschreibt.
Genau genommen finden bei der Betrachtung der gesamten Prozesskette zwei Segmentierungen statt. Mit dem ersten wird die Detektion der Hand vorgenommen. Der zweite teilt die Folge von Merkmalen, die den HMMs als Beobachtungsreihe zugeführt wird, durch Detektion von Start- und Endpunkten einer gesuchten Geste in einzelne Bewegungsabschnitte ein [Elm10].
Letztendlich findet eine Klassifikation statt, in der ein Klassifikator die zugeführten Eigenschaften einer Musterklasse zuweist. Eine Klasse definiert dabei verschiedene Zustände, die durch das Auftreten bestimmter Merkmalsfolgen ein gesuchtes Muster repräsentieren.
Abbildung 2.1 stellt den Prozess der Mustererkennung grafisch dar:

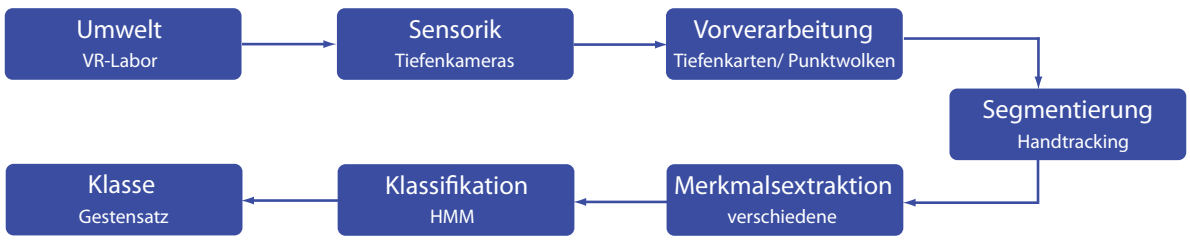


Abbildung 2.1: Schematische Darstellung eines Mustererkennungsprozesses im VR-Labor

Auf die ersten vier Teilprozesse wird nicht weiter eingegangen. Das daraus resultierende Ergebnis, eine erfolgreiche Ortung der Hand im dreidimensionalen Raum, wird als gegeben vorausgesetzt. Somit handelt es sich bei den für die Merkmalsextraktion verwendeten Daten um vierdimensionale Vektoren, von denen jeder einzelne zum Zeitpunkt t die Raumkoordinaten x, y und z enthält.

$$Trajektorie(n) = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} \tag{2.1}$$

Die Berechnung der Merkmale basiert demzufolge auf räumlichen und zeitlichen Informationen. Abbildung 2.2 zeigt die Entstehung der Trajektorien am Beispiel einer im zweidimensionalen Raum ausgeführten Bewegung:

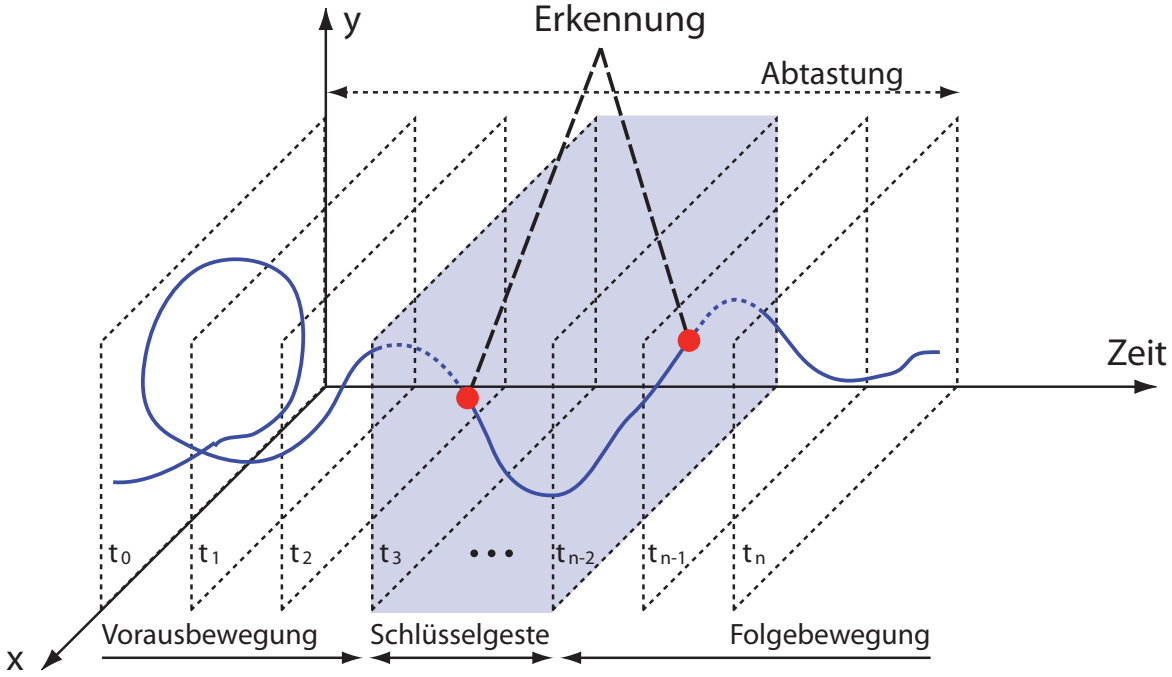


Abbildung 2.2: Erkennung relevanter Bewegungen in einem kontinuierlichen 2-dimensionalen Datenstrom [Elm10]

2.2 Extraktion von Merkmalen

Nach [Ric11] werden Merkmale als numerische Werte aufgefasst, die von rohen Messdaten abstrahieren. Ein guter Überblick über die typischen Verfahren der Merkmalsextraktion lässt sich aus [CeS95] gewinnen.

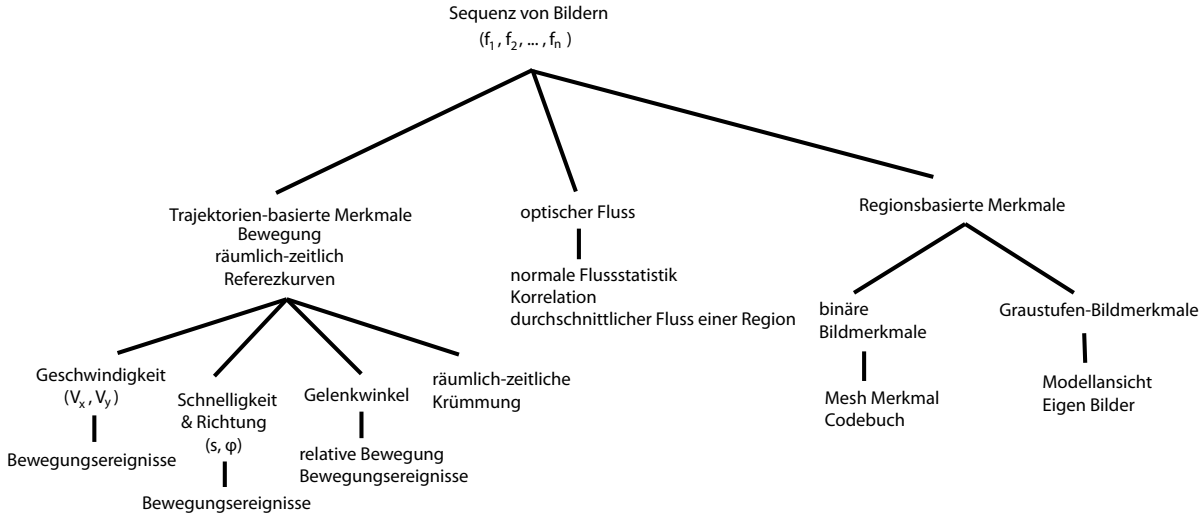


Abbildung 2.3: Verfahren zur Merkmalsextraktion auf der Grundlage von Bildsequenzen [Ces95]

Wie sich in Abbildung 2.3 erkennen lässt, gibt es drei Hauptgruppen von Merkmaltypen: trajektorien-basierte, optischer Fluss- und regionsbasierte Merkmale. Dagegen wählt [Ric11] eine etwas andere Unterteilung in:

- (1) silhouettenbasierte Merkmale
- (2) farbbasierte Merkmale
- (3) strukturbasierte Merkmale

Weitere Ausführungen zu Punkt (3) beschreiben ähnliche Ansätze wie sie aus Abbildung 2.3 bekannt sind:

- räumliche-zeitliche Repräsentation
- Einzelpostur-basierte Ansätze
- dichte Flussfelder
- Trajektorienanalyse

[YSB01], [Ric11], [Elm10] und weitere zeigen, dass sich trajektorienbasierte Merkmale für die Gestenerkennung sehr gut eignen, weil es sich um sehr dynamische Merkmale handelt. Da die Schnittstelle zum VR-Labor (s. Kapitel 6) über (x, y, z)-Koordinaten gebildet wird, bietet sich dieser Ansatz auch hier an.

Davon abstrahierte Repräsentationen können ein- oder mehrdimensional sein, je nachdem, wie viele unterschiedliche, voneinander unabhängige, Merkmale berechnet werden. Bei letzteren spricht man von einem Merkmalsvektor. Werte können, je nach dem gewählten Modell, in kontinuierlicher oder diskreter Form vorkommen.

2.2.1 Orientierung

Ein in der Literatur häufig zu findender Ansatz verwendet das Merkmal der Orientierung in Kombination mit diskreten HMMs [LeK99][EHM08]. In [YSB01] wird gezeigt, dass eine Kombination von Lokalität, Winkel und Geschwindigkeit gute Ergebnisse erreicht. Eine Bestätigung dessen ist in der Dissertation von [Elm10] zu finden, in der drei verschiedene Merkmale zur Orientierung berechnet werden.
Im zweidimensionalen Raum wird der Winkel zwischen zwei aufeinander folgenden Punkten zur X-Achse ermittelt.

$$\phi_t = arctan\left(\frac{y_{t+1} - y_t}{x_{t+1} - x_t}\right);\ f\ddot{u}r\ t = 1, 2, ..., T-1 \tag{2.2}$$

Anschließend erfolgt durch Division mit der Zahl 20 eine Quantisierung des 360° Kreises auf 18 gleich große Sektoren. Diesen sind die Symbole von 1 bis 18 zugewiesen (siehe 2.4).

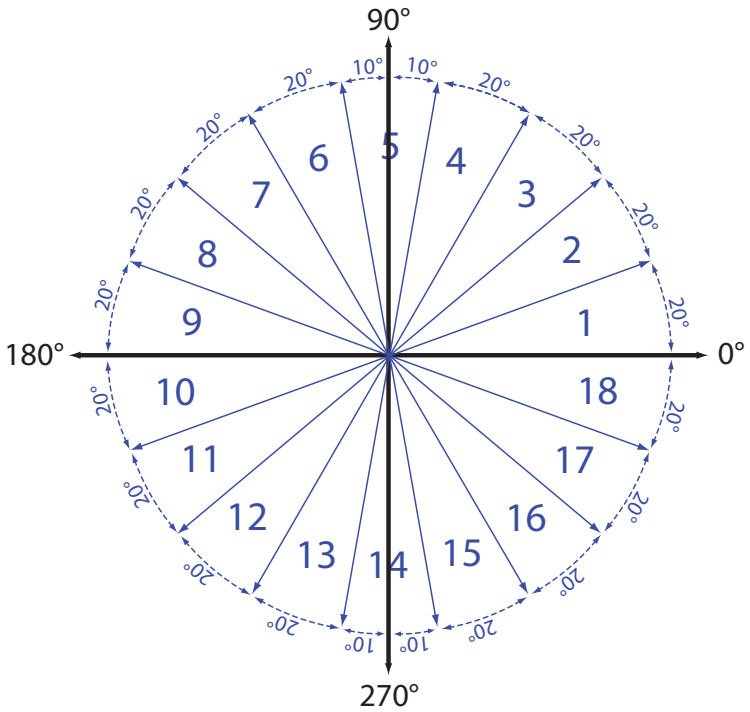


Abbildung 2.4: Quantisierung des Kreises auf 18 gleich große Sektoren [Elm10]

Während [LeK99] 16 Sektoren verwendet, setzen neuere Publikation (mitunter vom selben Autor) 18 Segmente ein. Diese bilden das Codebuch für die Emissionen, das sich formal wie folgt darstellen lässt:

$$V = \{1, 2, 3, ..., 18\} \tag{2.3}$$

2.2.2 Geschwindigkeit

An zweiter Stelle bzw. oft auch in Kombination mit dem Merkmal der Orientierung wird die Geschwindigkeit berechnet. Diese sei hier der Vollständigkeit halber angegeben, da sie im Verlauf der Arbeit noch eine wichtige Rolle einnehmen wird:

$$V_t = \sqrt{\left(\frac{x_{t+1} - x_t}{t}\right)^2 + \left(\frac{y_{t+1} - y_t}{t}\right)^2} \tag{2.4}$$

[Elm10] hat hier gezeigt, dass sich die Geschwindigkeit je nach Geste an den Eckpunkten einer ausgeführten Symbolbewegung verlangsamen kann. Um diese Veränderung zu erfassen, werden die Modelle, die auf Basis der Orientierung arbeiten, um dieses ergänzt. Genauere Zusammenhänge dafür jedoch nicht angegeben, sodass hier ein möglicher Anknüpfungspunkt für diese Arbeit besteht.

2.3 Hidden-Markov-Modelle

Sind die Merkmale aus einer Bewegungsaufnahme berechnet worden, erfolgt als letzter Punkt der Prozesskette die Klassifikation. Auf den Gebieten der Spracherkennung, Handschrifterkennung und in der Analyse biologischer Sequenzen haben sich HMMs als Standard-Werkzeug für Aufgaben der Musterkennung von zeitlich sequentiell vorliegenden Daten etabliert [GeF03]. Dabei liegen ihre Stärken vor allem auf der Erkennung und Klassifizierung informationstragender Abschnitte einer Sequenz von Daten (z.B.: Bildsequenzen). Im Bereich der Gestenerkennung sind ebenfalls diverse Ansätze auf Basis von HMMs zu finden, die gute bis sehr gute Erkennungsraten liefern.
Ein Model λ besteht grundlegend aus einer Menge von Zuständen, für die es festgelegte Übergangswahrscheinlichkeiten gibt [Fin03]. Jeder Zustand umfasst Parameter, welche die Wahrscheinlichkeit zu einer Beobachtung (konkreter Wert eines Merkmals) wiedergeben. Außerdem muss eine Definition über den Eintrittpunkt bzw. den Startzustand eins Modelles angegeben werden. Somit lässt sich ein HMM grundlegend durch drei Komponenten vollständig beschrieben:

- eine Matrix A, in der die Übergangswahrscheinlichkeiten der einzelnen Zustände definiert sind

  $$A = \{a_{ij} | a_{ij} = P(S_t = j | S_{t-1} = i)\} \tag{2.5}$$

  Die Matrix A bildet sich aus den Wahrscheinlichkeiten a für die Zustandsübergänge von i nach j, wobei a_{ij} die Wahrscheinlichkeit für den Zustandswechsel $S_t = j$, unter der Bedingung $S_{t-1} = i$ angibt. Die typische Darstellungsform für ein Modell mit n Zuständen lautet:

  $$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \tag{2.6}$$

- ein Vektor, der die Startwahrscheinlichkeiten für die einzelnen Zustände enthält, d.h. alle möglichen Eintrittspunkte für das Modell angibt

  $$\pi = \{\pi_i | \pi_i = P(S_1 = i)\} \tag{2.7}$$

- zustandsspezifische Emissionsverteilungen

$$\{b_j(o_k)|b_j(o_k) = P(O_t = o_k|S_t = j)\} \tag{2.8}$$

Ausgangsbasis für die Erzeugung und Verwendung von HMMs bilden sequentiell vorliegende Beobachtungswerte zu den gewählten Merkmalen O:

$$O = \{o_1, o_2, ..., o_n\} \tag{2.9}$$

An dieser Stelle seien noch einmal kurz die beiden Begriffe „Emissionen“ und „Observationen“ betrachtet. Ersteres meint hier Beobachtungen, die aus den Modellparameter hervorgehen (z.B.: Synthese). Letzteres umfasst weitestgehend alle Beobachtungen, die einem Modell im Trainings- oder Erkennungsprozess zugeführt werden. Aus einer Folge von Beobachtungen sind die durchlaufenden Zustände nicht ersichtlich, sondern in dieser verborgen (engl. „hidden“). Eine wichtige Eigenschaft von HMMs ist dabei die Segmentierung. Diese wird im Modell durch die Zuordnung von Observation und Zustand automatisch vorgenommen.

2.3.1 Diskrete Modelle

Die Beschreibung der Emissionswahrscheinlichkeiten hängt stark vom Modelltyp ab. Im diskreten Fall werden diese durch eine Matrix B repräsentiert, welche die diskreten Wahrscheinlichkeitsverteilungen enthält.

$$B = \{b_{jk}|b_{jk} = P(O_t = o_k|S_t = j)\}$$

Da die Anzahl der möglichen Observationen endlich ist und jede einzelne dieser symbolischen Charakter hat, spricht man in der Literatur auch von einem Codebuch [Fin03]. Dieses lässt sich beschreiben durch die Menge aller möglichen zu beobachtenden Symbole:

$$V = \{v_1, v_2, ..., v_n\} \tag{2.10}$$

2.3.2 Kontinuierliche Modelle

Handelt es sich bei den Emissionen um vektorielle Größen, werden kontinuierliche Modelle (CHMMs) eingesetzt, die die Beobachtungen mit Hilfe von Mischverteilungen auf Basis der Gaußschen Dichtefunktion (Normalverteilung) modellieren. Somit wird jede Dimension des Merkmalvektors als Zufallsvariable mit dazugehöriger Verteilung aufgefasst. Formal lässt sich die Darstellung wie folgt beschreiben [Fin03]:

$$b_j(X) = \sum_{k=1}^{M_j} c_{jk} N(X|\mu_{jk}, K_{jk}) = \sum_{k=1}^{M_j} c_{jk} g_{jk}(X) \tag{2.11}$$

- mit j als aktueller Zustand
- mit M als Mischverteilung
- mit N als Normalverteilung
- mit c als Gewichtung der Normalverteilung

- mit μ als Mittelwert der Normalverteilung
- mit K als Kovarianzmatrix der Zufallsvariablen pro Zustand

Wie im späteren Verlauf der Arbeit noch gezeigt wird, lassen sich die vorliegenden Daten mit nur einer Normalverteilung pro Zufallsvariablen beschreiben, sodass alle Gewichtungen $c_j = 1$ sind und deshalb keine Mischverteilungen zum Einsatz kommen sondern nur Normalverteilungen. Daher lässt sich die Emissionsmodellierung auf folgende Darstellung vereinfachen:

$$b_j(X) = N(X|\mu_{jk}, K_{jk}) = g_{jk}(X) \tag{2.12}$$

2.3.3 Topologie

Eine vorgegebene Topologie definiert, welche Zustandsübergänge möglich sind (jedoch nicht mit welcher Wahrscheinlichkeit). Dabei hat sich in der Praxis gezeigt, dass eine geringe Anzahl von Zustandsübergänge oft ausreicht, um gewünschte Ergebnisse zu erzielen [Fin03][LLK04]. Neben weiteren sind hier besonders zwei Modellvarianten interessant:

- lineare Modelle (L-Modell)
- ergodische Modelle (E-Modell)

Wie aus der Bezeichnung zu entnehmen ist, kann man sich lineare Modelle als eine Kette von Zuständen vorstellen, bei der Übergänge nur von einem Zustand zum nächsten möglich sind. Darüber hinaus ist ein Verweilen in den einzelnen Zuständen möglich. Das bedeutet, dass ein Zustand, der erreicht wurde, sich selbst wieder als Nachfolgezustand haben kann. In der Literatur wird in diesem Zusammenhang von „self loops“ oder „self transitions“ gesprochen [Elm10]. Abbildung 2.5 zeigt ein Beispiel einer linearen Struktur.

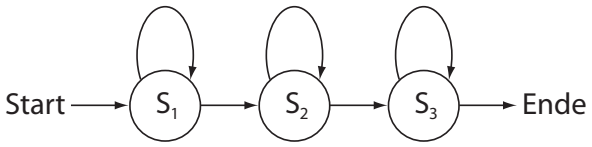


Abbildung 2.5: HMM: lineare Modellstruktur [Fin03]

In [LLK04] wird gezeigt, dass lineare Modellstrukturen die besten Erkennungsraten für Gesten liefern und zudem auch noch den Aufwand für Berechnungen gering halten. Zu Ersterem wird die starke Bindung zwischen Modellstruktur und der Struktur einer ausgeführten Geste genannt, welche bei Modellen mit vielen Zustandsübergängen schneller verloren gehen kann. Verglichen wurden dabei lineare Modelle, LR-Modelle und ergodische Modelle. [Elm10] bestätigt dies ebenfalls.
Die Anzahl der Modellzustände wird auf Grundlage des zu erkennenden Musters geschätzt. Dabei wird jedem möglichst gerade verlaufenden Linien-Segment ein eigener Zustand zugeordnet. Dies sei am Beispiel des Dreiecks einmal versinnbildlicht:

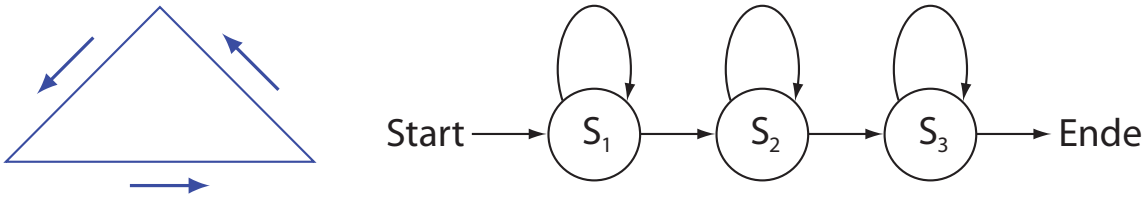


Abbildung 2.6: Topologie - Dreieck

Absolute Strukturfreiheit, und deshalb in der Praxis eher selten eingesetzt werden ergodische Modelle. Bei diesen kann jeder Zustand sich selbst und jeden weiteren erreichen (Abbildung 2.7). Im Gegensatz zu einigen anderen Topologien gibt es hier nicht nur einen Übergang ausgehend vom Start- und Endzustand. Jeder Zustand kann sowohl Eintrittspunkt in das Modell, sowie Endzustand sein. Ergodische Strukturen eignen sich vor allem für die Erkennung beliebiger, nicht einer Musterklasse zugeordneter Signalanteile. In dem Zusammenhang wird dieser Modelltyp deshalb häufig als „Garbage“-Modell bezeichnet.

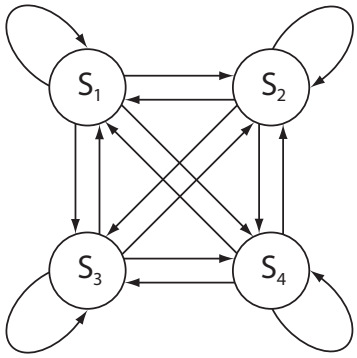


Abbildung 2.7: HMM: ergodische Modellstruktur [Fin03]

In der Regel werden Struktur und Parameter für das jeweilige Modell zunächst geschätzt. Ein standardisiertes Verfahren gibt es hierfür jedoch nicht. So wird die Struktur oft auf Grundlage von Erfahrungswerten festgelegt, wobei dem meist ein Evaluationsprozess voraus geht, bei dem verschiedene Modelltypen bezüglich der gewählten Merkmalsbeschreibung überprüft werden. Ausgangspunkt sind dabei meist stichprobenartig ausgewählte Daten, aus denen die stochastischen Eigenschaften für jedes Merkmal abgeleitet werden. Diese Vorgehensweise liegt auch der in dieser Arbeit verwendeten Quelle zu Grunde[LLK04].

2.3.4 Training

Nachdem die initialen Parameter für ein HMM festgelegt worden sind, erfolgt ein Trainingsprozess, bei dem diese bezüglich einer vorliegenden Menge von Trainingsdaten so optimiert werden, das die Berechnung der Produktionswahrscheinlichkeit maximal ausfällt. Dafür wird der Baum-Welsch-Algorithmus eingesetzt[Fin03]. Da dieser nur ein lokales Maximum bezüglich aller möglichen Modelle liefert, ist es für einen erfolgreichen Erkennungsprozess wichtig, dass im ersten Schritt schon eine möglichst treffende Modellstruktur- und Parameterfestlegung vorgenommen wurde [Fin03]. Das Training wird daraufhin mehrmals d.h. in mehreren Iteration durchgeführt. Für eine detaillierte Beschreibung zu dem Verfahren und den verwendeten Algorithmen sei an dieser Stelle auf [Fin03], [Rab89] oder [LLK04] verwiesen.

2.3.5 Klassifizierung

Ist das Training erfolgt, kann nun die Erkennungsrate ermittelt werden. Dazu wird die Produktionswahrscheinlichkeit für jedes Modell im Bezug auf eine vorliegende Folge von Beobachtungen berechnet $P(O|\lambda)$ [Fin03]. Dabei darf die Folge der Observationen nicht aus dem Trainingsdatensatz stammen. Über eine $argmax()$-Entscheidung wird das beste Modelle ausgewählt.
Da der Raum aller errechenbaren und darstellbaren Zahlen aufgrund der physikalisch bedingten Grenzen der Computerhardware begrenzt ist, wird für die Repräsentation der Wahrscheinlichkeiten eine logarithmische Form, meistens zur natürlichen Basis e, gewählt. Diese wird als *loglikelihood* bezeichnet. So können auch sehr geringe Werte noch berechnet und anzeigt werden. Bei der logarithmischen Wahrscheinlichkeitsberechnung ergibt demzufolge $0 = 100\%$.
Da das HMM-basierte System, wie es hier offline eingesetzt wird, große Parallelen mit klassischen Suchsystemen aufweist, wie sie in [ISc04] erläutert werden, können für die Auswertung der Klassifikation die dafür charakteristischen Kennzahlen genutzt werden. Das Ergebnis der Zuordnungen eines oder mehrerer Modelle zu einem Satz gegebener Emissionsfolgen kann durch zwei verschiedene Fehlerarten und zwei korrekten Zuweisungen beschrieben werden:

- ca: correct alarms - Anzahl der richtig zugeordneten Sequenzen zu einem Modell
- fa: false alarms - Anzahl der falsch zugeordneten Sequenzen zu einem Modell
- cd: correct dismissals - Anzahl der richtig abgelehnten Sequenzen zu einem Modell
- fd: false dismissals - Anzahl der falsch abgelehnten Sequenzen zu einem Modell

Aus diesen lassen sich die Werte für „recall“ und „precision“ ableiten. Nach [AnH08] lassen sich folgende Definitionen angeben:

$$recall_{\lambda_n} = \frac{ca_{\lambda_n}}{(ca_{\lambda_n} + fd_{\lambda_n})} \times 100\% \tag{2.13}$$

$$precision_{\lambda_n} = \frac{ca_{\lambda_n}}{(ca_{\lambda_n} + fa_{\lambda_n})} \times 100\% \tag{2.14}$$

Während *recall* die Zuverlässigkeit eines HMMs dahingehend bestimmt, dass alle zutreffenden Sequenzen auch richtig klassifiziert werden, gibt *precision* an, wie gut das System

eine nicht zutreffende Trajektorie als solche erkennt. Bei beiden Kennzahlen ist ein Wert von 100% anzustreben. Da die Entscheidung in der Klassifikation meist ein HMM zurückliefert, wird oft ergänzend eine Verwechslungsmatrix angegeben, aus der hervorgeht, welche Symbole nicht eindeutig auseinander gehalten werden können. Tabelle 2.1 zeigt ein Beispiel für ein optimales Ergebnis

Modell	**wird erkannt als (in %)**			
	$\boldsymbol{\lambda_4}$	$\boldsymbol{\lambda_K}$	$\boldsymbol{\lambda_{Kreis}}$	$\boldsymbol{\lambda_{Dreieck}}$
$\boldsymbol{\lambda_4}$	100	0	0	0
$\boldsymbol{\lambda_K}$	0	100	0	0
$\boldsymbol{\lambda_{Kreis}}$	0	0	100	0
$\boldsymbol{\lambda_{Dreieck}}$	0	0	0	100

Tabelle 2.1: Beispiel - Verwechslungsmatrix

Um die Bedeutung der hier in einem begrenzten Rahmen ermittelten Ergebnisse für die tatsächlich vorhandene Parameterlage feststellen zu können und um direkt eine statistische Signifikanz der durchgeführten Messungen zu erkennen, wird für Aufgaben der Detektion ein Konfidenzintervall angeben [Ges14]. In der Regel wird das 95% Konfidenzintervall c_{95} als ausreichend betrachtet, sodass dieses auch hier Verwendung findet.
Bestimmt wird das Intervall nach [Ges14] durch:

$$c_{95} \approx \pm 2\sqrt{\frac{m - m^2}{N - 1}} \tag{2.15}$$

N gibt die Anzahl der Stichproben an und m beschreibt das Verhältnis der korrekten Klassifizierungen C zur Gesamtstichprobenanzahl N:

$$m = \frac{C}{N} \tag{2.16}$$

3 Lösungsansatz

Ein Problem, das in der Literatur in Verbindung mit HMMs auftritt, ist die Repräsentation von Gesten durch geeignete Merkmale, sodass eine robuste und gleichzeitig variationsreiche Beschreibung und Erkennung von Bewegungen erfolgen kann. Ausgangspunkt der bisher umgesetzten Lösungen sind die von Probanden aufgezeichneten Bewegungsabläufe, auf deren Grundlage die Parameter hinsichtlich der Verwendung von HMMs geschätzt werden [Elm10][LeK99][Ric11]. Im Gegensatz zu dieser Verfahrensweise wird in dieser Arbeit ein Ansatz vorgestellt, bei dem ein kleiner Gestensatz zunächst synthetisch erzeugt wird. Es sei hier die Arbeitsthese aufgestellt, dass Modelle, die in der Lage sind eine Geste wohlgeformt zu generieren, vom Menschen ausgeführte Bewegungsabläufe auch zuverlässig erkennen. Anschließend erfolgt die Evaluierung der dabei ermittelten Modelle in einem Analyseprozess auf der Grundlage von natürlichen Daten.
Die im Rahmen dieser Arbeit durchgeführten Untersuchungen werden „offline“ vorgenommen, d.h. alle Gesten werden in einzelnen Dateien abgelegt und als einzelne Trajektorie verwendet. Die Erkennung läuft dann in einem späteren Prozess ab. Das hat den Vorteil, dass das erwartete Ergebnis mit dem tatsächlich eintreffenden genau verglichen und analysiert werden kann, da beides bekannt ist. Außerdem wird die Problematik der Detektion von Anfangs- und Endpunkten innerhalb eines kontinuierlichen Datenstroms in den Hintergrund geschoben. Dabei sei vorweggenommen, dass im Kapitel 6 mit dem „Threshold-Modell-Ansatz“ ein entsprechendes Verfahren dazu demonstriert wird. Außerdem kann hier eine Unterscheidung bezüglich der Relevanz der aktuellen Bewegung für eine gesuchte Geste vorgenommen werden.
Ein weiterer Aspekt, der bei dem „offline“ -Verfahren an Bedeutung verliert, ist die Performanz bzw. die Echtzeitfähigkeit des Systems. Da hier der Prozess der Erkennung im Vordergrund steht, wird darauf nicht weiter eingegangen.

3.1 Synthese

Nachdem die Betrachtung der theoretischen Grundlagen abgeschlossen ist, erfolgt an dieser Stelle der Übergang zu den praktischen Ausführungen. Eine besondere Bedeutung kommt dem Schritt der Synthese deshalb zu, weil diese Herangehensweise von der typischen Vorgehensweise im Bereich der Gestenerkennung mit HMMs abweicht.
Speziell in diesem Kapitel geht es um das synthetische Erzeugen bzw. das Generieren von Emissionsfolgen durch ein vorher definiertes HMM. Dabei sollen vor allem die eingesetzten Algorithmen überprüft werden, um deren Korrektheit sicher zu stellen. Eine Rückführung auf fehlerhafte Implementierung bei später auftretenden Problemen soll dabei ausgeschlossen werden.

3.1.1 Symbolsatz

Der hier zugrunde liegende Symbolsatz umfasst 4 Gesten, die so gewählt sind, dass sie für die Klassifizierung, unter Berücksichtigung der Vorbetrachtungen, unterschiedliche Schwierigkeitsgrade aufweisen bzw. Probleme mit sich bringen können. Da sowohl Anfangs- und Endpunkt, als auch die Art und Weise, wie ein Symbol gezeichnet wird, vom jeweiligen Benutzer abhängt, wird die Folge der zu zeichnenden Zustände fest vorgegeben.

"4": Besondere Beachtung findet bei der Ziffer 4 der waagerechte Strich in Form einer Rechtsbewegung, da dieser die Modellstruktur eines HMMs entscheidend beeinflusst. Bewegt sich die Gesten-ausführende Person nach der Rechtsbewegung gleich nach oben kann dass HMM mit 4 Zuständen gebildet werde. Erfolgt jedoch erst eine Linksbewegung, bevor es nach oben weiter geht, werden 5 Zustände benötigt. Somit ergeben sich grundsätzlich 2 Modellstrukturen.

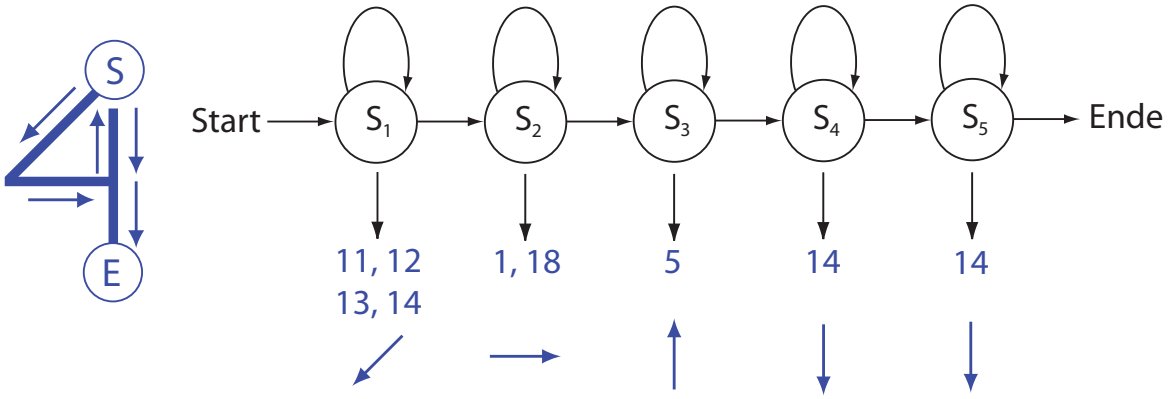


Abbildung 3.1: Geste „4“ - diskret

"K": Das großgeschriebene „K “ weist in den Zuständen 1, 2, 3 und 5 ähnliche Eigenschaften wie die Zustände der „4“ auf und wurde deshalb ausgewählt, weil anzunehmen ist, dass die einzelnen Segmente in ihrer Parametermodellierung mit denen der „4 “ korrespondieren. Hauptunterscheidungsmerkmal der beiden Symbole ist vor allem die unterschiedliche Reihenfolge der Zustände und die Anzahl der enthaltenen Modellzustände. Darüber hinaus ist anzunehmen, dass schräge Linien stärker variieren, als horizontale oder vertikale.

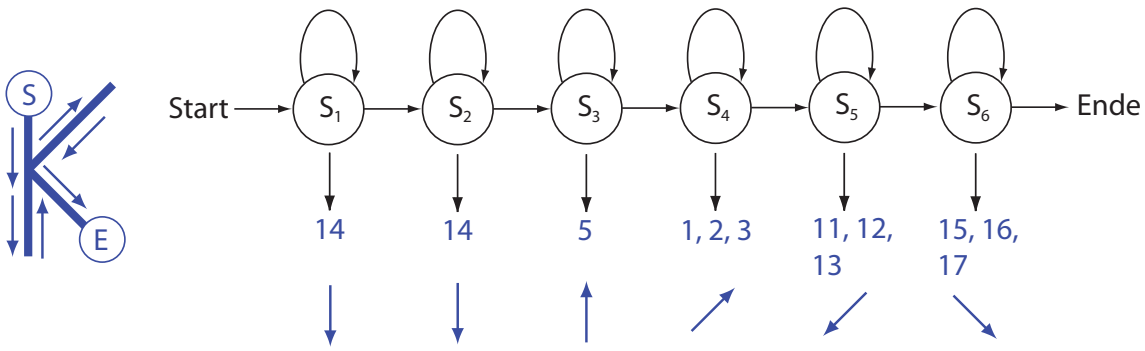


Abbildung 3.2: Geste „K“ - diskret

"Kreis": Die Tatsache, dass der Kreis theoretisch jeden möglichen Winkel zwischen 0° und 360° enthalten kann, macht ihn zu einer interessanten Herausforderung für die Gestenerkennung. Bei der Ausführung sind zwei Richtungen möglich. So kann der Kreis mit oder gegen den Uhrzeigersinn gerichtet sein. Der Startpunkt und Endpunkt könnte überall auf der Kreislinie liegen, ohne dass sich das Symbol ändern würde. Mit einer Einteilung des Kreises in vier Segmente mit ausreichender Varianz sind pro Ausführungsrichtung jeweils vier unterschiedliche Startpunkte möglich. Somit ist anzunehmen, dass sich der Kreis durch insgesamt maximal acht Modelle beschreiben lässt. Die hier gewählte Variante beginnt rechts und bewegt sich dann gegen den Uhrzeigersinn.

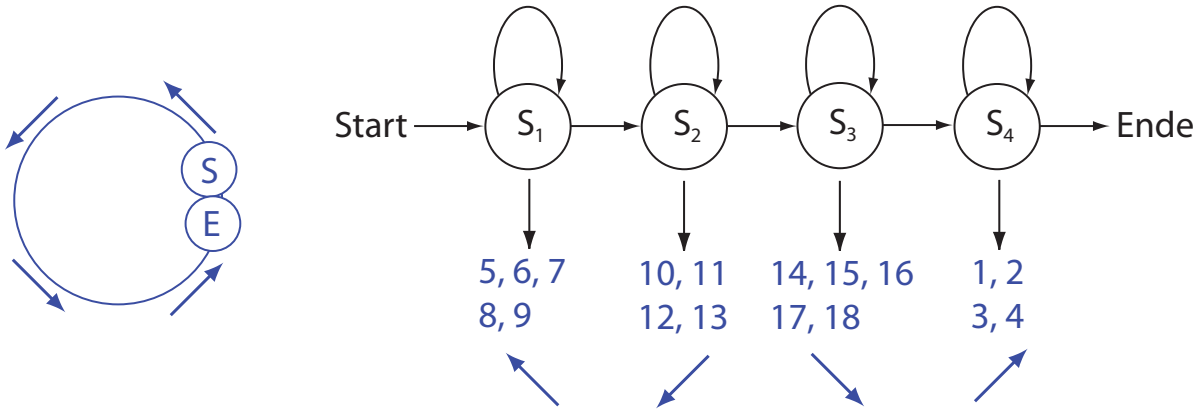


Abbildung 3.3: Geste „Kreis“ - diskret

"Dreieck": Als Vertreter für kurze Modellstrukturen kann das Dreieck mit seinen drei Zuständen angesehen werden. Hier sind vor allem Korrespondenzen mit dem Kreis zu erwarten, da sich bei Ungenauigkeiten in der Bewegungsausführung mit einer zunehmenden Abrundung der Ecken durchaus zirkuläre Eigenschaften erkennen lassen. Beim Dreieck allerdings ist der Startpunkt klar erkennbar. Es sind zwei Ausführungsrichtungen mit jeweils drei unterschiedlichen Startpunkten möglich. Hier sind maximal sechs unterschiedliche Modelle für die Erkennung zu erwarten. Die hier modellierte Variante ist in Abbildung 3.4 zu sehen.

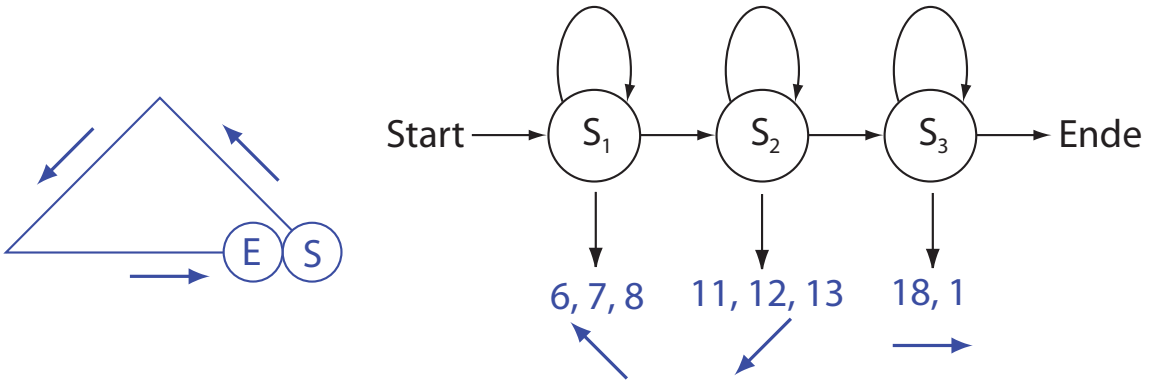


Abbildung 3.4: Geste „Dreieck“ - diskret

Zunächst werden die Symbole durch diskrete Modelle repräsentiert. Aus der Dissertation von [Elm10] lassen sich HMM-Strukturen für die Symbole „4“ und „K“ entnehmen, während die Modelle für „Kreis“ und „Dreieck“ nach [LeK99] strukturiert werden können. Deren grafische Darstellung wurde in den vier vorangegangenen Abbildungen bereits gezeigt. Bei der Betrachtung der ersten beiden Symbole fällt auf, dass Linien, die bei idealer Gestenausführung die doppelte Länge im Vergleich zu anderen im Symbol enthaltenen Linien besitzen, mit zwei Zuständen belegt werden (Beispiel: K - Zustand 1 & 2). Die Emissionsmodellierung ist hierbei jedoch identisch. Aus letzterer Quelle lässt sich lediglich die Anzahl der verwendeten Zustände entnehmen, die für eine bestimmte Geste verwendet wurde. Die genaue Zuordnung ist allerdings unklar. Daher werden hier die unter [Elm10] erwähnten Kriterien für die Festlegung der Modellstruktur verwendet.

$$a_{ii} = 1 - \frac{1}{d} \quad mit \quad d = \frac{\varnothing T}{N} \tag{3.1}$$

$$\varnothing T = durchschnittliche\ Länge\ der\ Observationsfolge \tag{3.2}$$

$$N = Anzahl\ der\ Modellzustände \tag{3.3}$$

Die Abtastrate des VR-Labors liegt zwischen 25 und 30 Hz und löst damit doppelt so hoch auf, wie bisherige Systeme. Da die Synthese zunächst nur auf Annahmen basiert, wird die vermutliche Länge einer aufgenommenen Trajektorie zu einer Geste basierend aus den Kennzahlen der bisher betrachteten Publikationen erstmal auf die doppelte Anzahl von Merkmalen geschätzt. Somit ergeben sich für die hier benutzten Symbole zu erwartende durchschnittliche Sequenzlängen von:

- $\varnothing T_4 = 94$
- $\varnothing T_K = 128$
- $\varnothing T_{Kreis} = 72$
- $\varnothing T_{Dreieck} = 66$

Da in [Elm10] und [LeK99] weder „Kreis“ noch „Dreieck“ explizit ausgewiesen werden, bzw. sich keine Angaben zur durchschnittlichen Sequenzlänge finden lassen, sind hier Messwerte zu ähnlichen Symbolen herangezogen worden. So ist in [Elm10] für den Buchstaben „O“ eine durchschnittliche Länge von Observationen von $\varnothing T_O = 36$ angeben. Beim Kreis wird ein äquivalentes Verhalten geschätzt. Weiter werden für die Buchstaben „N“ und „Z“ Kennzahlen von $\varnothing T_N = 35$ und $\varnothing T_Z = 32$ genannt. Da beide Gesten mit drei Zuständen modelliert wurden, wird für das hier verwendete „Dreieck“ ein ähnliches Verhalten geschätzt. Nach der Multiplikation mit 2 ergeben sich die zuvor beschriebenen Werte.
Nach den Restriktionen, die ein L-Modell mit sich bringt, können die Modelle von dem Startzustand aus jeweils nur über den ersten Zustand passiert werden. Damit ergibt sich für den Startvektor mit den A-prior-Wahrscheinlichkeiten:

$$\pi = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{3.4}$$

Die Matrix für die Zustandsübergangswahrscheinlichkeiten sieht so aus:

$$A = \begin{pmatrix} a_{11} & 1 - a_{12} & 0 & \dots & 0 \\ 0 & a_{22} & 1 - a_{22} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} \tag{3.5}$$

3.1.2 Diskrete Modellierung

Entsprechend einer diskreten Emissionsmodellierung ergibt sich für die B-Matrix mit einer Anzahl von m Symbolen pro Zustand und n Zuständen:

$$B = \begin{pmatrix} b_{11} & b_{12} & \dots & b_1 \\ b_{21} & b_{22} & \dots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nm} \end{pmatrix} = \begin{pmatrix} \frac{1}{m} & \frac{1}{m} & \dots & \frac{1}{m} \\ \frac{1}{m} & \frac{1}{m} & \dots & \frac{1}{m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{m} & \frac{1}{m} & \dots & \frac{1}{m} \end{pmatrix} \tag{3.6}$$

Diese Modellierung wurde primär für die Erkennung von Gesten entwickelt, wobei der normalerweise ein Trainingsprozess vorausgeht. Würde man diese Modelle ohne Training für die Synthese einsetzen, würden sie nach der Formel 3.6 jedes der 18 Symbole mit gleicher Wahrscheinlichkeit emittieren. Für ein sinnvolles Ergebnis der Synthese muss die B-Matrix daher neu geschätzt werden. Die geänderte konkrete Belegung der Wahrscheinlichkeiten für die Observationssymbole lässt aus Tabelle 3.1 entnehmen. Diese basieren auf den Eingangs vorgestellten vier Geste.

Symbol	Zustand	Emission v	P(v) in %
4	1	11, 12, 13, 14	25
	2	1, 18	50
	3	5	100
	4	14	100
K	1	14	100
	2	5	100
	3	2, 3, 4	33.33
	4	11, 12, 13	33.33
	5	15, 16, 17	33.33
Kreis	1	5, 6, 7, 8, 9	20
	2	10, 11, 12, 13	25
	3	14, 15, 16, 17, 18	20
	4	1, 2, 3, 4	25
Dreieck	1	6, 7, 8	33.33
	2	11, 12, 13	33.33
	3	1, 18	50

Tabelle 3.1: DHMM - Emissionswahrscheinlichkeiten

Außerdem wurden neben den am Anfang vorgestellten Modellen noch zwei weitere hinzugenommen. Darunter sind eine zusätzliche Version für die „4“ und das „K“ . Bei beiden Symbolen wurde der doppelt vorhandene Zustand zu einem zusammengefasst. Da in [Elm10] neben der Orientierung noch andere diskrete Merkmale (z.B.: Lokalität) getestet wurden, hat die Aufteilung in zwei Zustände durchaus ihre Berechtigung. Da hier zunächst nur auf Basis der Orientierung gearbeitet wird, ist eine Zusammenfassung der Zustände möglich. Diese müssen dann allerdings die doppelte Anzahl von Emissionen generieren, wie die gleichen Zustände in den Ausgangsmodellen, so dass hier eine Anpassung der Übergangswahrscheinlichkeiten mit der doppelten Anzahl geschätzter Werte vorgenommen wurde. Aus der Synthese ist zu erwarten, dass sich die von den modifizierten Modellen erzeugten Trajektorien in ihrem Verhalten nicht wesentlich von den Ausgangsmodellen unterscheiden. Die in Tabelle 3.1 definierten Emissionen beziehen sich bereits auf die modifizierte Modellvariante für „4“ und „K“.
Bevor die Ergebnisse der diskreten Modelle präsentiert werden, sei an dieser Stelle noch erwähnt, dass der aus der Bibliothek zur Verfügung gestellte Algorithmus für die Generierung von Observationen nicht verwendet wurde, da er zwar erlaubt, Sequenzen in gewünschter Länge zu erzeugen, nicht aber garantiert, dass wirklich alle Zustände durchlaufen werden. Deshalb wurde ein eigener Algorithmus entwickelt, welcher Letzteres garantiert. Allerdings lässt sich hier die Länge der Beobachtungsfolge nicht vorgeben. Wie spätere Untersuchungen zeigen werden, ist die Länge der Emissionsfolge stark abhängig von der Ausführungsgeschwindigkeit der Gesten. Daher ist eine Längenvariation für Trainingssequenzen durchaus sinnvoll. Außerdem bestimmen die Übergangswahrscheinlichkeiten der Modellzustände, wie lang im Mittel eine Sequenz ist. Entsprechend einer statistischen Verteilung können dabei auch sehr kurze und längere Sequenzen entstehen, die aus praktischer Sichtweise betrachtet.

	λ_4 Var 1	λ_4 Var 2	λ_K Var 1	λ_K Var 2	λ_{Kreis}	$\lambda_{Dreieck}$
n_O	150	150	150	150	150	150
$\varnothing T_{geschätzt}$	94	94	128	128	72	66
$\varnothing T_{Synthese1}$	92.64	92.98	125.7	129.42	72.62	67.72
$\varnothing T_{Synthese2}$	90.08	101.74	126.12	139.41	73.47	63.68
$\varnothing T_{Synthese3}$	91.33	94.73	125.7	130.64	69.91	66.04
$\varnothing T_{Synthese4}$	92.12	97.86	126.39	127.91	68.95	64.98
$\varnothing T_{Synthese5}$	98.15	95.43	129.52	121.2	74.42	71
a_{ii} in %	94.68	97.34	95.31	97.66	94.44	95.45
$a_{i(i+1)}$ in %	5.32	2.66	4.69	2.34	5.56	4.55

Tabelle 3.2: DHMM - durchschnittliche Anzahl von Emissionen und Zustandsübergangswahrscheinlichkeiten

Um sicherzustellen, dass die Synthese auch die erwartete Sequenzlänge generiert, sind insgesamt fünf Synthesedurchläufe mit jeweils 150 generierten Emissionsfolgen durchgeführt worden. Wie die Tabelle 3.2 zeigt, sind die Erwartungen der geschätzten Anzahl von Emissionen im Mittel auch eingetroffen. Außerdem konnte bestätigt werden, dass die beiden Modelle mit den reduzierten Zuständen kein abweichendes Verhalten zeigen und somit ohne Bedenken verwendet werden können.
Als Letztes muss noch gewährleistet werden, dass sich die einzelnen Emissionen auch ähnlich zu der festgelegten Wahrscheinlichkeitsverteilung verhalten. Diese lässt erwarten, dass im Mittel über alle Sequenzen jeder Zustand annähernd gleich oft passiert wird, d.h. die gleiche Anzahl von Werten erzeugt. Die Bestätigung dafür ist aus Tabelle 3.3 ersichtlich :

Zustand	λ_4 : $\varnothing N_O$	λ_K : $\varnothing N_O$	λ_{Kreis} : $\varnothing N_O$	$\lambda_{Dreieck}$: $\varnothing N_O$
1	17.45	19.34	19.31	23.34
2	22.4	22.91	16.48	22.77
3	18.66	19.25	17.84	20.76
4	20.76	18.42	19.59	nicht vorhanden
5	20.33	22.16	nicht vorhanden	nicht vorhanden
6	nicht vorhanden	18.51	nicht vorhanden	nicht vorhanden

Tabelle 3.3: DHMM - durchschnittliche Anzahl Emission pro Zustand

Um eine synthetisch erzeugte Trajektorie so darzustellen, dass sie für den Menschen als Symbol erkannt werden kann, müssen zuvor einige Annahmen getroffen werden. Bis auf die Tatsache, dass die Emissionen zeitlich nacheinander generiert worden sind, lassen sich keine weiteren zeitlichen oder räumlichen Informationen aus dem eindimensionalen Merkmal der Orientierung gewinnen. Zumindest Letztere sind aber notwendig, um ein Symbol im zweidimensionalen Raum zeichnen zu können. damit kann festgestellt, dass es sich im Vergleich zu den Ausgangsdaten mit xyz-Koordinaten und Zeitstempel t um eine erhebliche Relevanzreduktion bezüglich der für das Zeichnen einer Geste notwendigen Informationen handelt.
Um trotzdem eine Visualisierung zu ermöglichen, wird im Folgenden angenommen, dass jedes erzeugte Symbol eine Länge von 1 im zweidimensionalen Koordinatensystem hat. Unter der Voraussetzung einer kontinuierlichen Abtastrate entspräche das einer Bewegungsausführung mit konstanter Geschwindigkeit. Weiterhin ist der Winkel zu jedem Emissionssymbol nicht eindeutig bestimmbar, da jedes Symbol einen Sektor von 20° repräsentiert.

sentiert. Bei den nachstehenden Abbildungen wird daher angenommen, dass immer der Winkel hinter dem entsprechenden Symbol steht, der die Mitte eines Sektors darstellt: Somit ergeben sich für die einzelnen Sektoren die Werte:

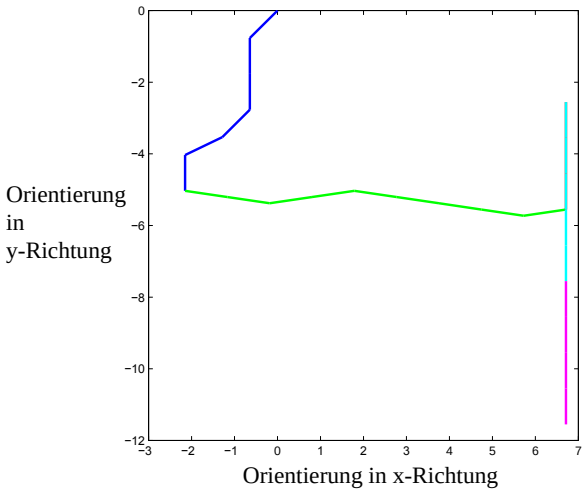
0	1	2	3	4	5	6	7	8	9
10°	30°	50°	70°	90°	110°	130°	150°	70°	190°

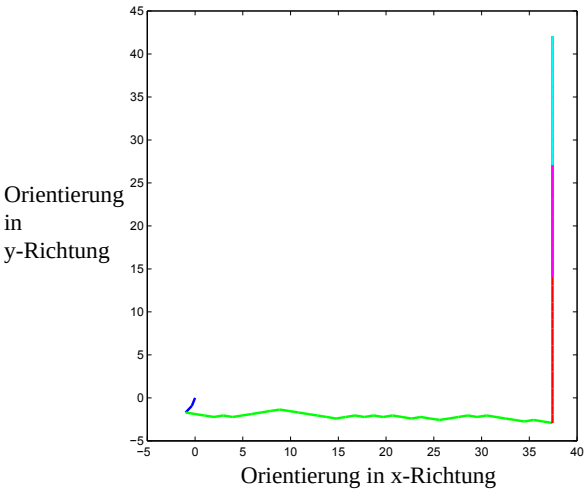
Tabelle 3.4: DHMM - Zuordnung Sektor und Winkel (1)

0	10	11	12	13	14	15	16	17	18
10°	190°	210°	230°	250°	270°	290°	310°	330°	350°

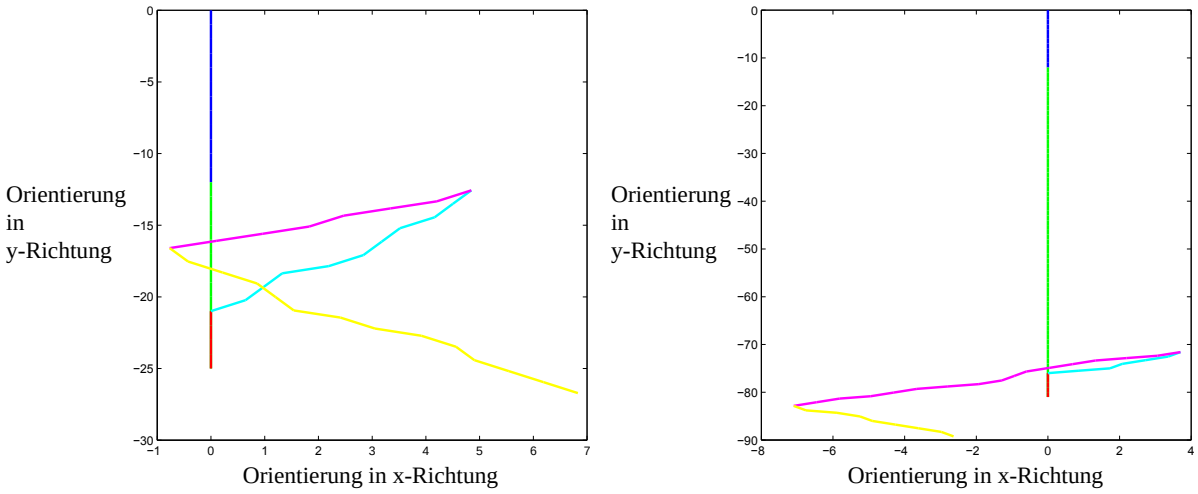
Tabelle 3.5: DHMM - Zuordnung Sektor und Winkel (2)

Die folgenden Abbildungen zeigen immer jeweils ein Positivbeispiel und ein Negativbeispiel für jedes Symbol. Die zuerst genannten bestätigen, dass die Modelle in der Lage sind auch wohlgeformte Gesten zu erzeugen. Die Negativbeispiele beinhalten meist einen Zustand, der deutlich öfter Emissionen generiert, als die anderen. Als Folge dessen wirken die Symbole entartet. Jedes Liniensegment gleicher Farbe kennzeichnet Emissionen, die aus dem selben Zustand generiert wurden.

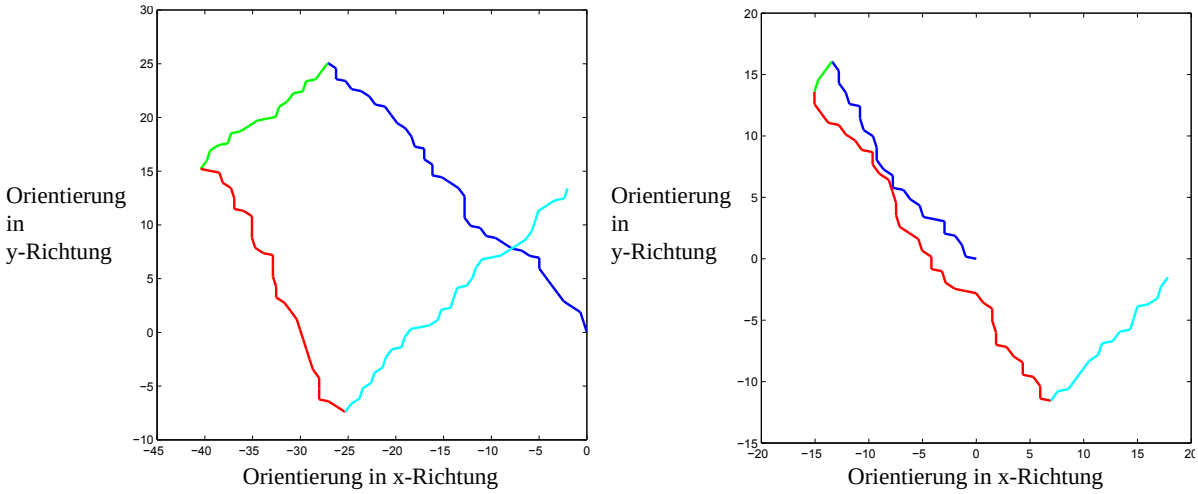




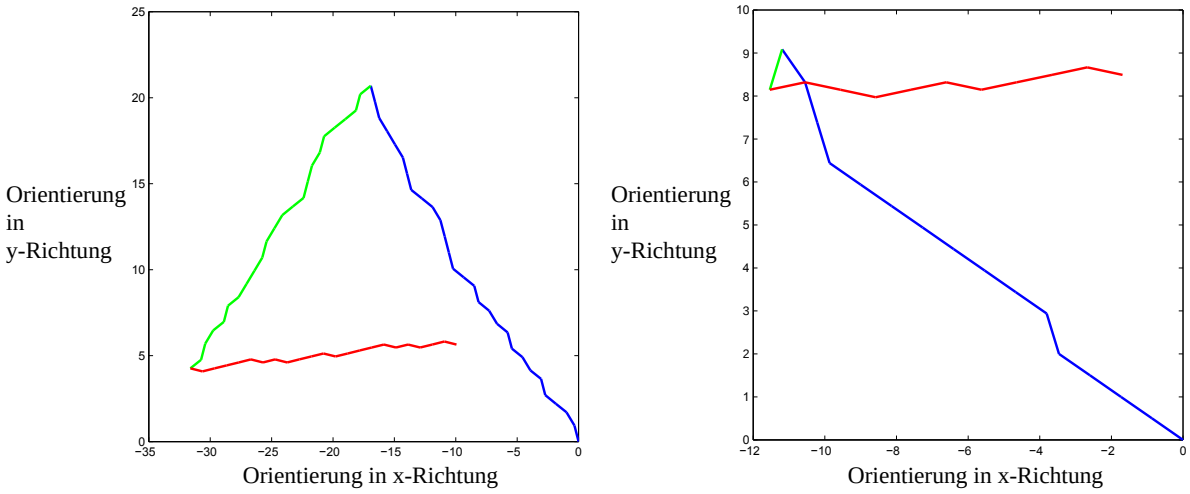
Die obere Abbildung lässt gut erkennen, dass schräge Liniensegmente eine größere Varianz aufweisen und trotzdem eindeutig zu dem gleichen Symbol zugeordnet werden können, währende horizontale und waagerechte Linien auch eindeutig als solche wahrgenommen werden.
Rechts lässt sich die Vier fast nicht mehr erkennen, da die hier generierten Emissionen nicht gleichmäßig verteilt sind. In der Visualisierung überlagern sich die Beobachtungen der letzten beiden Zustände, sodass sich zuerst an ein Dreieck denken lässt.
Bei genauer Betrachtung der nicht-linearen Achseneinteilung, handelt es sich bei dem Positivbeispiel für das Symbol „K“ eigentlich auch um eine entartete Abbildung (unten). Dennoch zeigt es, dass der Buchstabe in seiner typischen Form gezeichnet werden kann.



Ähnlich wie bei dem Symbol „4“ lässt sich in der rechten Abbildung der Buchstabe „K“ nicht mehr wirklich erkennen.



Interessant sind die Beobachtungen aus der oberen Grafik für den Kreis. Dieser ähnelt in seiner Erscheinungsform eher einem Rechteck. Die Ursache dafür liegt in der initialen Parameterschätzung. Dafür, dass der Kreis nur 4 verschiedene Zustände beinhaltet, können die möglichen Emissionen jeden Winkel generieren.



Bei dem Dreieck sei noch eine Anmerkung zur linken Figur der oberen Abbildung gemacht. Aufgrund der Lage von Sektor 1 und 18 kann bei den für das Zeichnen getroffenen Annahmen keine horizontale Linie erzeugt werden. Da 0° nicht abgebildet werden, kann entsteht in Zustand 3 ein ständig die Richtung ändernder Verlauf.
Als Fazit lässt sich dabei festhalten, dass sich trotz größerer Varianzen und unterschiedlichsten Ausprägungsformen des gleichen Symbols alle Zustände eindeutig wiedererkennen lassen.

3.1.3 Kontinuierliche Modellierung

Wie die Abbildung 3.5 nochmal verdeutlicht, handelt es sich bei der diskreten Modellierung aufgrund der Quantisierung um harte Entscheidungen, bei denen jedes Observationssymbol einen Bereich von 20° beschreibt.

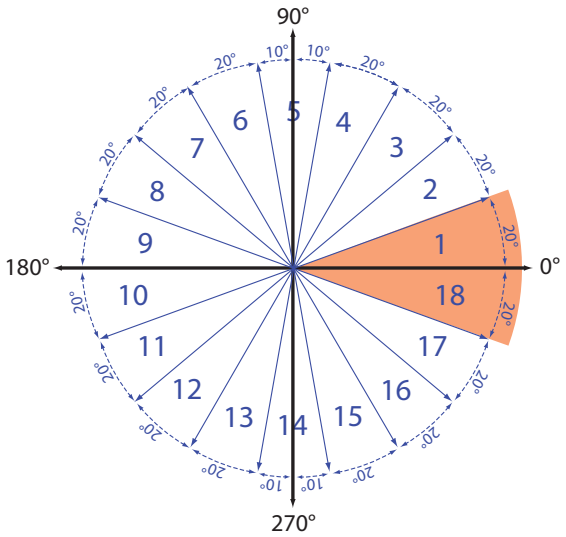


Abbildung 3.5: DHMM - Modellierung einer Rechtsbewegung belegt Observationen mit einer Streuung von 40° mit Wahrscheinlichkeiten

Soll beispielsweise eine horizontale Bewegung nach rechts modelliert werden, müssen gleich 2 Sektoren (1 und 18) mit Wahrscheinlichkeiten belegt werden, da 0° genau auf der Grenze zwischen diesen liegt. Somit würde das dafür verwendete Modell bereits in einem Bereich von 40° Emissionen generieren, obwohl nur eine waagerechte Bewegung modelliert werden soll. Dadurch wirken die durch Synthese erzeugten Emissionen sehr kantig und unnatürlich und entsprechen damit nicht den in der Realität erzeugten Gesten. Um dem zu Beginn formulierten Anspruch gerecht zu werden, die Modelle so zu parametrisieren, dass sie in der Lage sind, die Symbole auch zu erzeugen, ist eine kontinuierliche Modellierung notwendig. Dabei bleiben die Anzahl der Modellzustände, die Übergangswahrscheinlichkeiten und die Modellstruktur zunächst gleich. Lediglich die Emissionswahrscheinlichkeiten werden mittels gewichteten Normalverteilungen in den kontinuierlichen Wertebereich transportiert.
In einem ersten Ansatz werden die 20° breiten Sektoren nun durch Normalverteilungen repräsentiert, deren Mittelwerte auch dem Mittelpunkt eines jeden Sektors entsprechen. Die Varianz σ wird dabei ebenfalls auf die Breite der Sektoren gesetzt. Waagerechte und senkrechte Liniensegmente werden mit einer Standardabweichung von 20° modelliert. Da es anzunehmen ist, dass schräge Linien eine größere Varianz besitzen, wird hier zunächst eine Standardabweichung von 40° angenommen. Um ein Intervall mit einer gewünschten Breite zu erhalten ist bei der Verwendung von Normalverteilungen eine Umrechnung notwendig:

$$Var\,(X) = \sigma^2 \tag{3.7}$$

Die gewünschte Breite wird als Standardabweichung angenommen und über Gleichung 3.7 in die Varianz überführt.

3.1.4 Kartesische Koordinaten

Bei genauerer Betrachtung der Darstellung von Winkel als Wertebereich im kartesischen Koordinatensystem tritt allerdings beim Kreis ein Problem auf, dass sich auch auf andere Symbole übertragen lässt. Da 0° und 360° im Bezug auf die Orientierung die gleiche Aussage liefern, bedeutet das auch, dass Winkel kleiner 360° und größer 0°, aufgrund der zirkularen Eigenschaft des Kreises direkt aneinander anschließende Wertebereiche darstellen sollten. In der bisherigen Form, lässt sich das nicht realisieren. Wie es Abbildung 3.6 darstellt, müssen zwei Normalverteilungen (= eine Mischverteilung) eingesetzt werden, wenn der Wertebereich um 0° bzw. 360° erfasst werden soll. Gleichzeitig adressiert Abbildung 3.6 ein dabei auftretendes Problem. Da die Bedingung gilt, dass die Summe über die Integrationsergebnisse aller Normalverteilungen gleich 1 sein muss, wäre eine Gewichtung dieser notwendig. Die Modellierung ist jetzt zwar mathematisch korrekt, spiegelt aber nicht das ursprünglich gewünschte Verhalten wider, da Observationen um den Punkt der Unstetigkeit nur noch mit der um die Hälfte reduzierten Wahrscheinlichkeit emittiert werden können.

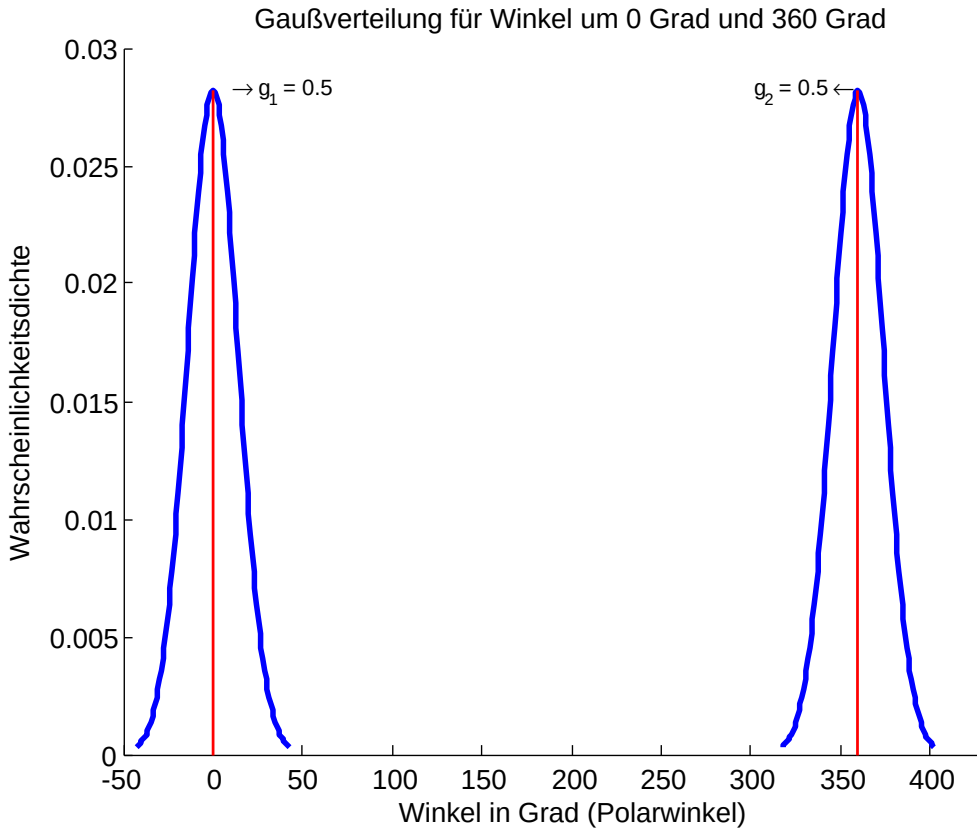


Abbildung 3.6: Kontinuierliche Wahrscheinlichkeitsmodellierung an der Sprungstelle von 0° zu 360°

Um das Problem umgehen zu können ist jedoch eine Transformation der Koordinaten notwendig.

3.1.5 Polarkoordinaten

Für die Abbildung aller Winkel auf ein periodisches Signal bietet sich eine Transformation zwischen polaren und kartesischen Koordinaten an.

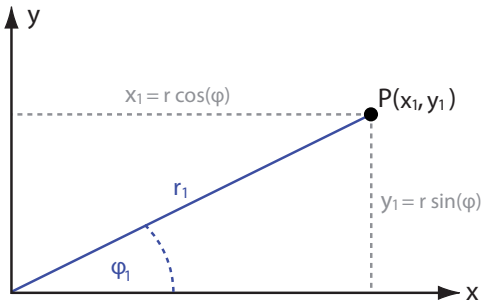


Abbildung 3.7: Im Polarkoordinatensystem wird der Punkt $P(x_1, y_1)$ durch den Polarwinkel ϕ_1 und den Radius r_1 beschrieben. Eine Umrechnung zwischen den beiden Systemen erfolgt über die Winkelfunktionen cos und *sin* [Roo14].

Wie es Abbildung 3.7 zeigt, lässt sich ein Punkt durch Nutzung einfacher Winkelfunktionen von einem System in das andere überführen. Ein erheblicher Vorteil ergibt sich daraus für die Darstellung des Kreises in Polarkoordinaten. Die für die Orientierung berechneten Winkel werden als Winkelkoordinate in einem Polarkoordinatensystem betrachtet. Da hier nur die Ausrichtung des Vektors und nicht die Länge von Bedeutung ist, wird $r = 1$ gesetzt, wobei r der Betrag des Vektors ist. Anschließend erfolgt eine Umrechnung von Polarkoordinaten in die kartesische Form:

$$x = r\ \cos(\phi) \tag{3.8}$$

$$y = r\ \sin(\phi) \tag{3.9}$$

Mit $r = 1$ ergibt sich für $x = \cos(\phi)$ und für $y = \sin(\phi)$. Da der Wertebereich auf den Bereich zwischen 0° und 360° ($[0, 360]$) begrenzt ist, liefert das Ergebnis der Koordinatentransformation eine exakte Abbildung der Kosinus- und Sinusfunktion im Bereich von 0 bis 2π, dargestellt in Abbildung 3.8.

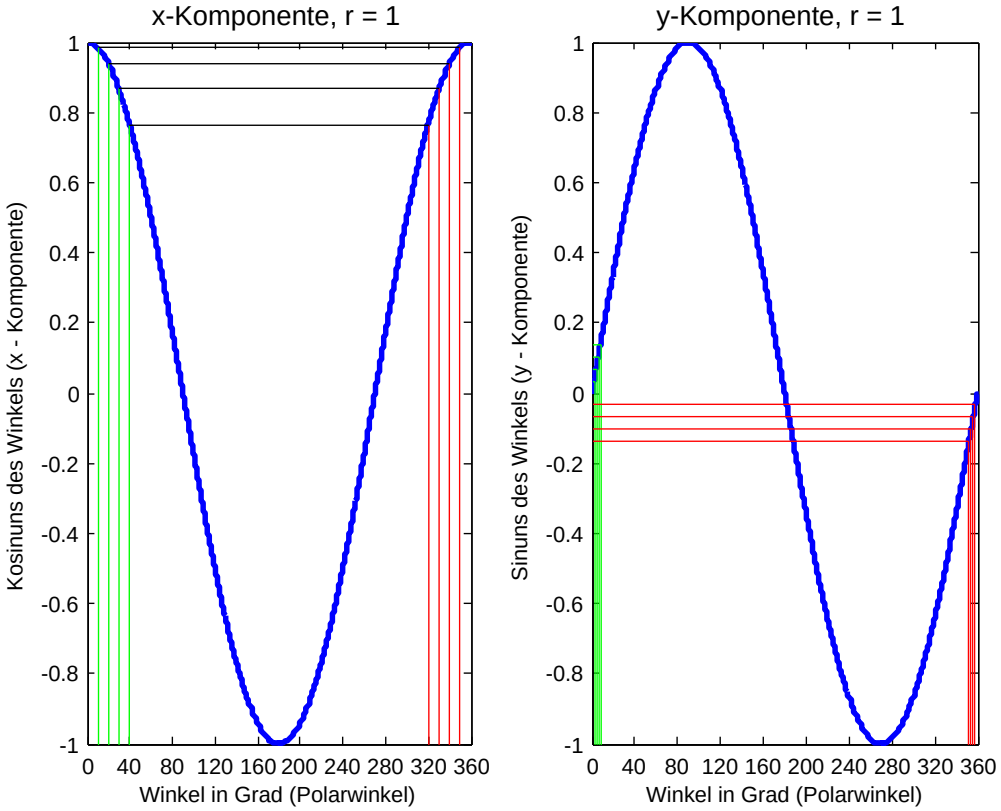


Abbildung 3.8: Koordinatentransformation polar - kartesisch: x- und y-Komponente

Die horizontalen Linien zeigen, dass die Winkel um den Bereich von 0° bzw. 360° durch die Kosinusfunktion auf den gleichen Wertebereich abgebildet und durch die Sinusfunktion zu benachbarten Wertebereichen werden. Somit ist eine Wahrscheinlichkeitsmodellierung für den Sprung der Werte über eine einzige Normalverteilung möglich. Eine Rücktransformation kann ohne Weiteres über Gleichung 3.10 realisiert werden.

$$\phi = \begin{cases} \arctan(\frac{y}{x}) & x > 0 \\ \arctan(\frac{y}{x}) + \pi & x < 0 \, und \, y \geqslant 0 \\ \arctan(\frac{y}{x}) - \pi & x < 0 \, und \, y < 0 \\ \frac{\pi}{2} & x = 0 \, und \, y > 0 \\ -\frac{\pi}{2} & x = 0 \, und \, y < 0 \\ n.d. & y = 0 \end{cases} \tag{3.10}$$

Der Abstand (Vektorlänge) lässt sich berechnen durch:

$$r = \sqrt{x^2 + y^2} \tag{3.11}$$

Eine vorliegende Observation wird jetzt nicht mehr nur durch ein Symbol beschrieben, sondern durch einen zweidimensionalen Merkmalsvektor:

$$O(n) = \begin{pmatrix} \cos(\alpha_n) \\ \sin(\alpha_n) \end{pmatrix} \tag{3.12}$$

Um eine Vorstellung der Charakteristik der neu definierten Wahrscheinlichkeitsmodellierung zu bekommen, zeigt Abb. 3.9 nochmal den Zusammenhang zwischen Modelltopologie und Emissionsgenerierung am Beispiel des Dreiecks.

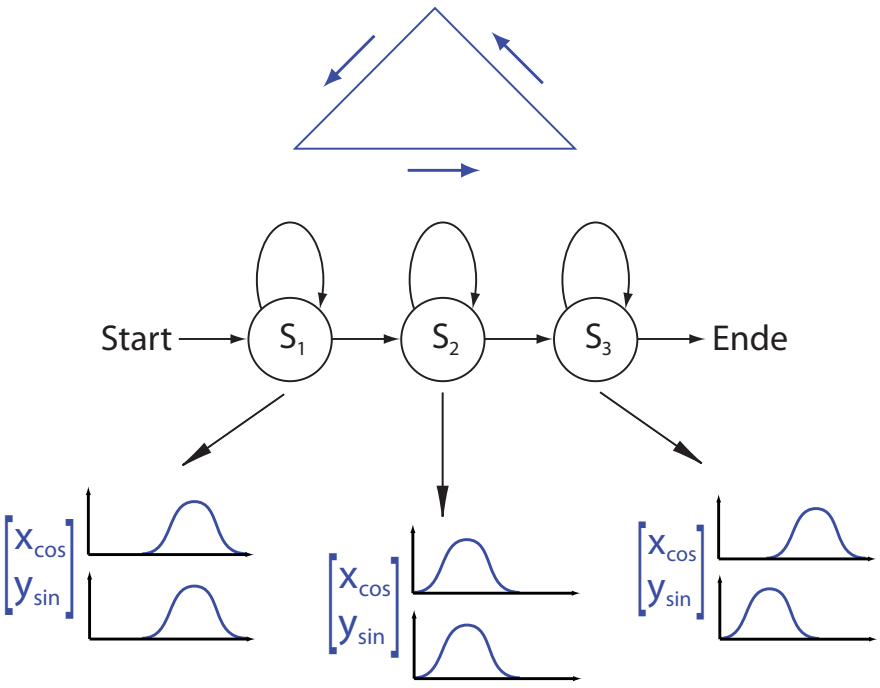


Abbildung 3.9: CHMM: 2-dimensionaler Merkmalsvektor am Beispiel des Dreiecks

Ein dabei unbedingt zu beachtender Punkt ist die Nichtlinearität dieser Transformation. Während bisher eine bestimmte Intervallbreite auf beliebige Mittelwerte verschoben werden konnte, müssen nun die Intervallbreiten in Abhängigkeit des Mittelwertes, der

Kosinus- und der Sinusfunktion für jede Wertänderung bei initialen Modellen berechnet werden. Die Grafik 3.10 veranschaulicht diese Problematik:

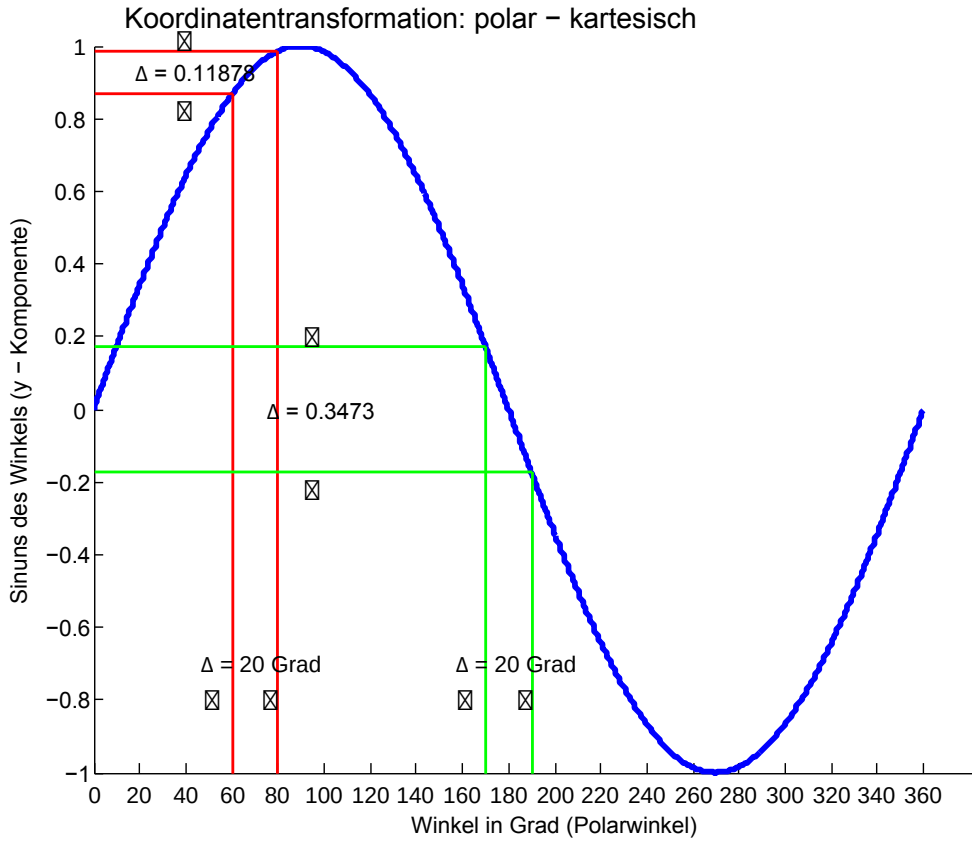


Abbildung 3.10: Koordinatentransformation polar - kartesisch: Nichtlinearität

Bei der Visualisierung müssen bezüglich des Winkels keine Annahmen gemacht werden, da dieser nach der Rücktransformation direkt entnommen werden kann. Für den Abstand zwischen zwei Punkten wird die Normierung 1 angenommen. Im Folgenden ist je Symbol ein Positivbeispiel dargestellt:

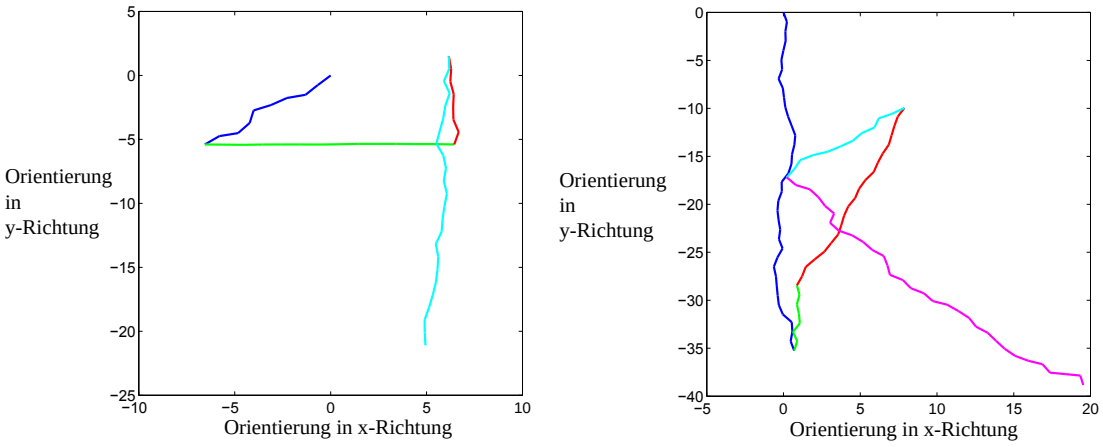


Abbildung 3.11: Synthese: 4 und K, kontinuierlich - positiv

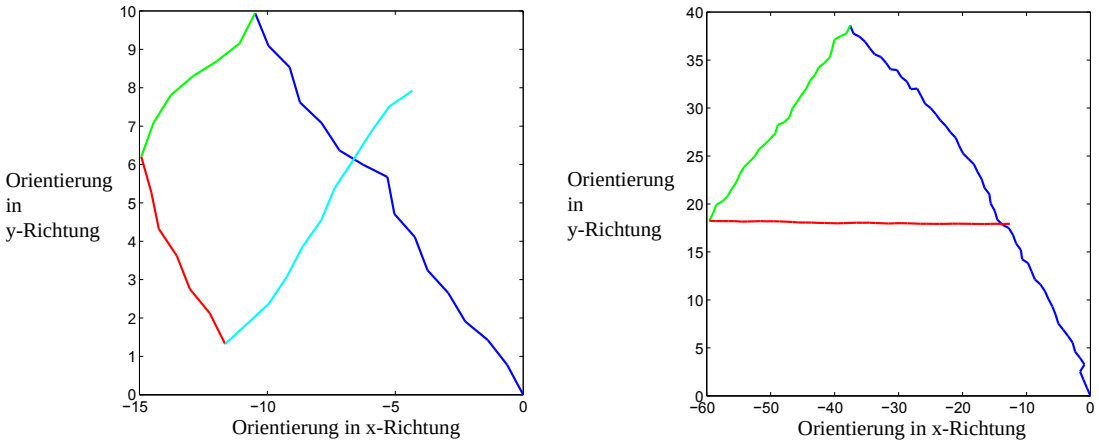


Abbildung 3.12: Synthese: Kreis und Dreieck, kontinuierlich - positiv

Die durch Synthese erzeugten Daten mit kontinuierlichen Modellen zeigen schon mehr Dynamik, erlauben weiche Übergange bei Richtungsänderungen und weisen eine deutlich größere Varianz in der Symbolerzeugung auf. Letzteres bedeutet auch, dass sich die Modelle im Trainingsprozess besser an die tatsächlich ausgeführten Gesten anpassen können. Während bei diskreten Modellen Wahrscheinlichkeiten für nur 20° große Sektoren geändert werden können, wird nun im reellen Zahlenraum gearbeitet. Als Nachteil sei hier erwähnt, dass dadurch allerdings der Berechnungsaufwand steigt.

3.1.6 Ergebnisse

Es konnte gezeigt werden, dass diskrete Modelle dazu in der Lage sind, aufgrund der Quantisierung die Symbole in grober Form zu erzeugen. Weiterhin kann festgehalten werden, dass die gewählte Form der Merkmale geeignet ist, um in Verbindung mit einer linearen Modellstruktur verschiedene Symbole eindeutig voneinander zu unterscheiden, insofern die Emissionen das Modellverhalten widerspiegeln. Außerdem konnte die Festlegung der initialen Zustandsübergangswahrscheinlichkeiten nach [Elm10] und [Lek99] bestätigt werden, da die Synthese gezeigt hat, das auch annähernd wohlgeformte Symbole generiert werden können. Um eine Trajektorie als Geste zeichnen zu können, müssen Annahmen bezüglich des Winkels und der Distanz getroffen werden, da die gewählte Merkmalsrepräsentation nicht ausreicht, um ein Symbol vollständig in seiner ursprünglichen Form zu beschreiben.
Wird nur das Merkmal der Orientierung verwendet, können Modellzustände mit ähnlichen oder gleichen Emissionen zusammengefasst werden. Im Gegensatz zu diskreten Modellen ist in einem kontinuierlichen HMM mehr Dynamik und Varianz in der Emissionsmodellierung zu beobachten. Um einen Winkel im kontinuierlichen Wertebereich mit Wahrscheinlichkeiten zu belegen, ist eine Koordinatentransformation notwendig.

3.2 Analyse

Um die Algorithmen für den Analyseprozess zu evaluieren wurden zunächst die Synthese-Modelle für die Erkennung der von ihnen selbst generierten Daten eingesetzt. Das Training ist an dieser Stelle nicht notwendig, da die Modelle die „Testdaten “ bereits optimal

modellieren sollten. Weiter lässt sich davon ableiten, dass die Zuverlässigkeit für jedes Modell bezüglich der generierten Sequenzen bei 100 % liegen sollte. Eine Bestätigung für diese Annahme für diskrete Modelle lässt sich aus Tabelle 3.10 entnehmen.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	50	0	150	0	100	100 ± 0.061391
λ_K	50	0	150	0	100	100 ± 0.061391
λ_{Kreis}	50	0	150	0	100	100 ± 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 ± 0.061391

Tabelle 3.10: Analyse - DHMM, synthetische Daten und Synthesemodelle, ohne Training

Überraschender Weise fällt die Klassifizierung für die kontinuierlichen Modelle unerwartet schlecht aus. Zu entnehmen ist dies aus Tabelle 3.11.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	9	7	143	41	56.25	18 ± 0.029391
λ_K	5	0	150	45	100	10 ± 0.022135
λ_{Kreis}	49	75	75	1	39.5161	98 ± 0.060976
$\lambda_{Dreieck}$	50	5	145	0	90.9091	100 ± 0.061391

Tabelle 3.11: Analyse - CHMM, synthetische Daten und Synthesemodelle ohne Training

Während die Erbebnisse für den Kreis und das Dreieck bei 100% bzw. 98% liegen, werden das K und die 4 nur zu 10% bzw. 18% richtig erkannt. Bei der Erstellung der Modelle wurde bereits angemerkt, dass CHMMs eine Kovarianz beinhalten, welche die Abhängigkeiten zwischen mehreren Zufallsvariablen beschreibt. Ohne Training sind diese alle 0, da nur die Varianz in der Hauptdiagonalen gesetzt wird. Da der Kreis alle möglichen Winkel emittieren kann, liefert er auch für die Sequenzen von K und 4 weitestgehend die höchsten Log-Likelihood-Ergebnisse. Dieses Verhalten ist in der Verwechslungsmatrix zu beobachten:

Modell	wird erkannt als (in %)			
	λ_4	λ_K	λ_{Kreis}	$\lambda_{Dreieck}$
λ_4	18	0	74	8
λ_K	12	10	76	2
λ_{Kreis}	2	0	98	0
$\lambda_{Dreieck}$	0	0	0	100

Tabelle 3.12: Analyse - CHMM, synthetische Daten und Synthesemodelle

An dieser Stelle lässt sich die Vermutung aufstellen, dass die gewählte Modellierung des Gestensatzes nicht ausreicht, um jedes Symbol eindeutig zu beschreiben. Um diese zu bestätigen, wird im nächsten Kapitel eine weitere Untersuchung vorgenommen, bei der gezeigt wird, wie sich die Erkennungsrate für trainierte Modelle verhält.

4 Vorexperimente I

Die bisher im Erkennungsprozess eingesetzten Synthese-Modelle werden nun durch trainierte Modelle ersetzt. Somit dient dieses Kapitel vor allem auch der Evaluation des Trainingsalgorithmus, welche mittels der bisher synthetisch erzeugten Daten durchgeführt wird.

4.1 Verfahrens- und Problembetrachtung des Modelltrainings

Die Grundlagen für das Modelltraining wurden bereits in Kapitel 2.3.4 beschrieben. Jetzt geht es darum, die konkreten Rahmenbedingungen und Parameter für den Baum-Welch-Algorithmus festzulegen. Dazu gilt es vor allem zwei Fragestellungen genauer zu betrachten:

- Wann soll der Algorithmus für das Training determinieren?
- Wie viele Sequenzen werden für das Training verwendet?

Zu Erstgenanntem wird in [Elm10] eine maximale Anzahl von 10 Durchläufen als ausreichend angegeben. Für die Realisierung des Trainingsprozesses wird in dieser Arbeit die Matlab-Implementierung von Kevin Murphy verwendet. Die Determination dieser Funktion kann dabei über zwei Parameter gesteuert werden:

- die Anzahl der Iteration: hier mit 10 festgelegt
- Schwellwert für die Differenz der Log-Likelihood-Funktionsergebnisse: initial mit 1×10^{-4} festgelegt

Mit jeder Iteration wird die Wahrscheinlichkeit für das zu trainierende Modell bezüglich der gegebenen Trainingsdaten berechnet. Somit kann nach jedem Durchlauf die Differenz dieser ermittelt werden. Dadurch lässt sich feststellen, ob weitere Iteration noch signifikante Änderungen in der Modelloptimierung erreichen können. Das Training ist beendet, wenn die Differenz der letzten beiden errechneten Wahrscheinlichkeiten den Schwellwert unterschreitet oder die maximale Anzahl an Iterationen erreicht ist.
Je nach verwendeter Literatur unterscheiden sich die Angaben über die Anzahl der benutzten Trainings- und Testdaten. So verwendet [LeK99] 200 Sequenzen für das Training und 50 für die Ermittelung der Erkennungsrate. [Elm10] hingegen optimiert die Modelle mit 42 verschieden Gesten und führt die Analyse mit 18 aufgenommenen Trajektorien durch. Im Durchschnitt liegt das Verhältnis der beiden Datensätze bei 3 : 1, d.h. 2/3 Trainingsdaten und 1/3 Testdaten. Entsprechend der im vorherigen Kapitel generierten Anzahl von Trajektorien gehen somit 100 Sequenzen für jedes HMM in das Training und 50 in die Analyse mit ein.
Anhand unterschiedlicher Signal-Rausch-Verhältnisse kann die Robustheit der Erkennung bestimmt werden. In mehreren Iteration wird zu jeder einzelnen Sequenz ein Weißes Rauschen addiert, wobei das SNR mit jeder Iterationen abnimmt. Am Ende jeden Durchlaufes wird ein Klassifikationsprozess eingeleitet, der zu jedem HMM die Erkennungsrate in

Relation zu den modifizierten Beobachtungsfolgen liefert. Berechnen lässt sich das Signal-Rausch-Verhältnis durch:

$$SNR = 20\log_{10}(\frac{\sigma_{signal}}{\sigma_{noise}}) \tag{4.1}$$

Um ein Störsignal mit einem fest definierten SNR in Bezug auf ein gegebenes Eingangssignal zu erhalten, wird erst ein Rauschen mit der Standardabweichung $\sigma = 1$ generiert. Die Anzahl und die Dimension der enthaltenen Werte muss dabei mit denen des Eingangssignals übereinstimmen. Als nächstes erfolgt die Bestimmung des Verstärkungsfaktors g, mit dem das Störsignal multipliziert werden muss, um das gewünschte Verhältnis der beiden Signale zu erhalten. Durch das Rauschen mit $\sigma = 1$ ergibt sich:

$$SNR = 20\log_{10}(\frac{\sigma_{signal}}{1}) \tag{4.2}$$

SNR mit unbekanntem Verstärkungsfaktor g:

$$SNR = 20\log_{10}(\frac{\sigma_{signal}}{1 \times g}) \tag{4.3}$$

Umstellen nach g:

$$g = 10^{(\frac{SNR}{20})}\ \frac{\sigma_{signal}}{\sigma_{noise}} = 10^{(\frac{SNR}{20})}\ \frac{\sigma_{signal}}{1} = 10^{(\frac{SNR}{20})}\ \sigma_{signal} \tag{4.4}$$

Das Rauschen wird allerdings nur den xy(z)-Koordinaten hinzugefügt. Würde man auch den Vektor mit den Zeitinformationen verändern, könnte es zu kausalen Widersprüchen kommen, beispielsweise, wenn der Startzeitpunkt t_0 einer Sequenz in den negativen Wertebereich verschoben wird.

$$\boxed{g = 10^{(\frac{SNR}{20})}\ \sigma_{signal}}$$

Abbildung 4.1: SNR:Verstärkungsfaktor

4.2 Analyse: Künstliche Daten und diskrete Modelle

Im Rahmen des Kapitel „Synthese“ wurde erwähnt, dass [Elm10] bei der Festlegung der initialen Parameter die Wahrscheinlichkeiten über alle Emissionssymbole gleich verteilt. Daher wurden diese neu verteilt, damit sie der Synthese eher gerecht werden und nur Emissionen hervorbringen, die auch zu der Geste passen. An dieser Stelle soll anhand des EM-Trainings überprüft werden, ob es signifikante Unterschiede hinsichtlich der LogLikelihood-Berechnung von Trainingsdaten und Modellen gibt. Das Ergebnis dazu lässt sich aus Tabelle 4.1 entnehmen.

Parameter	HMM	recall in %	precision in %	loglik	$EM_{Iterationen}$
Modelltyp: initial	λ_4	100	100	-4613.27	7
$N_{Trainingsdaten} = 100$	λ_K	100	100	-11182.02	10
$N_{Testdaten} = 50$	λ_{Kreis}	100	100	-11804.97	9
$EM_{Iter\_max} = 10$; $EM_{Threshold} = 10^{-04}$	$\lambda_{Dreieck}$	100	100	-7417.91	7
Modelltyp: synthese	λ_4	100	100	-4613.26	3
$N_{Trainingsdaten} = 100$	λ_K	100	100	-8646.89	3
$N_{Testdaten} = 50$	λ_{Kreis}	100	100	-11804.96	3
$EM_{Iter\_max} = 10$; $EM_{Threshold} = 10^{-04}$	$\lambda_{Dreieck}$	100	100	-7417.9	3

Tabelle 4.1: DHMM - Ergebnisse der Analyse

Das EM-Training für die Synthese-Modelle benötigt weniger Iterationen, als die der mit nach [Elm10] festgelegten Parametern. Eine Hauptursache liegt darin, dass die Sequenzen von den selben Modellen erzeugt worden sind und die Verteilungen denen der Observationen weitestgehend entsprechen.
Vor allem beim „K “ fällt auf, dass die LogLikelihood-Ergebnisse unterschiedlich ausfallen. Für das Synthese-Modell wird ein erkennbarer größerer Wert errechnet. Schaut man sich die Wahrscheinlichkeitsverteilung des trainierten initialen Modelles an, so kann man feststellen, dass Zustand 2 und 3 vorrangig eine Bewegung nach oben modellieren, während Zustand 4 korrekt eine Bewegung nach rechts oben abbildet und der letzte Zustand die im Synthese Modell gedachten Zustande 4 und 5 zusammenfasst und somit zwei unterschiedliche Richtungen emittiert. Das Modell würde demnach vom Ursprungsmodell abweichend so aussehen:

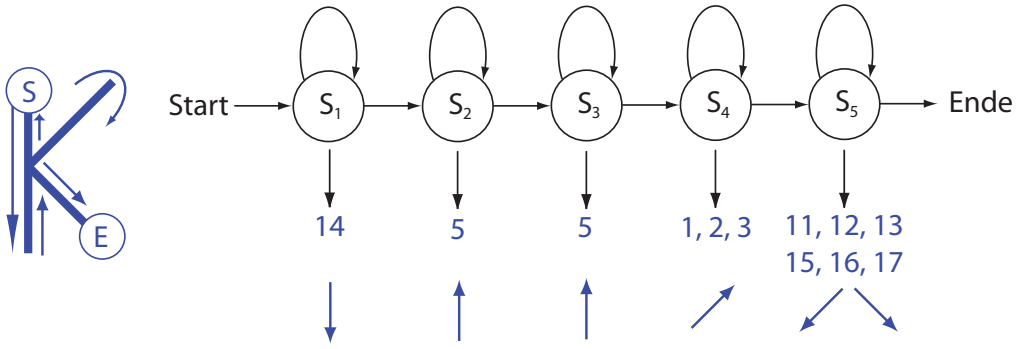


Abbildung 4.2: Geste „K“ - diskret, nach dem Training

Daher wird empfohlen, auch initiale Emissionswahrscheinlichkeiten an das reale Verhalten anzupassen.
Des Weiteren konnte die Angabe über die Anzahl der Iterationen bestätigt werden, da gezeigt wurde, dass nach einer bereits geringen Anzahl von Durchläufen keine signifikanten Änderungen der Wahrscheinlichkeit mehr auftreten und das Training nach wenigen Durchläufen beendet ist.

Auch das Ergebnis der Klassifikation entspricht den Erwartungen, da alle Sequenzen richtig zugeordnet worden sind:

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	50	0	150	0	100	100 ± 0.061391
λ_K	50	0	150	0	100	100 ± 0.061391
λ_{Kreis}	50	0	150	0	100	100 ± 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 ± 0.061391

Tabelle 4.2: Analyse - DHMM, Synthetische Daten, Modelle nach [Elm10]

Tabelle 4.2 zeigt die Ergebnisse für die von [Elm10] definierten Modelle, obwohl zu Beginn alle Emissionssymbole die gleiche Wahrscheinlichkeitsverteilung haben, konnten die Trajektorien nach dem Trainingsprozess korrekt zugeordnet werde. Deshalb kann eine korrekte Funktionsweise des Baum-Welch-Algorithmus angenommen werden.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	50	0	150	0	100	100 ± 0.061391
λ_K	50	0	150	0	100	100 ± 0.061391
λ_{Kreis}	50	0	150	0	100	100 ± 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 ± 0.061391

Tabelle 4.3: Analyse - DHMM, synthetische Daten und Synthesemodelle

4.3 Analyse: Künstliche Daten und kontinuierliche Modelle

Die Erwartungen bezüglich der Erkennungsrate für synthetisch erzeugte kontinuierliche Emissionen sind die gleichen, wie die der diskreten Modellierung, immer unter der Annahme, dass die Modelle auch die von sich selbst generierten Sequenzen erkennen.
Zunächst sei auch hier eine genauere Betrachtung des Trainingsprozesse vorgenommen. Im Gegensatz zu den diskreten Modellen, werden nur die Synthese-Modelle gemessen, da eine kontinuierliche Gleichverteilung über alle möglichen Beobachtungen, auch wenn der Wertebereich aufgrund der Winkelfunktionen begrenzt ist, nicht sinnvoll erscheint.

Parameter	HMM	recall in %	precision in %	loglik	$EM_{Iterationen}$
Modelltyp: Synthese	λ_4	100	100	3070.57	4
$N_{Trainingsdaten} = 100$	λ_K	100	100	-8381.77	6
$N_{Testdaten} = 50$	λ_{Kreis}	100	100	-14679.05	4
$EM_{Iter\_max} = 10$; $EM_{Threshold} = 10^{-04}$	$\lambda_{Dreieck}$	100	100	6183.45	4

Tabelle 4.4: CHMM - Ergebnisse der Analyse

Tabelle 4.4 zeigt auch hier, dass 10 Wiederholungen für das Training ausreichen, da der Baum-Welch-Algorithmus im Durchschnitt in weniger als der Hälfte der maximal durchlaufenden Iterationen beendet ist.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	50	0	150	0	100	100 ± 0.061391
λ_K	50	0	150	0	100	100 ± 0.061391
λ_{Kreis}	50	0	150	0	100	100 ± 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 ± 0.061391

Tabelle 4.5: Analyse - CHMM, synthetische Daten und Synthesemodelle

Gemäß den Erwartungen wurden auch bei den kontinuierlichen Modellen alle Testsequenzen richtig zugeordnet. Im Vergleich mit den DHMMs fällt auf, dass für das „K“ schon mehr als die doppelte Anzahl an Iterationen während des Trainings notwendig ist. Darin scheint sich die eingangs getroffene Annahme widerzuspiegeln, dass je mehr schräge Linien in einem Symbol vorhanden sind, dass desto größer die Varianz und damit auch der Berechnungsaufwand wird.
Daher kann vermutet werden, dass die Klassifizierung für kontinuierliche HMMs 1. nicht ohne Training zuverlässig verlaufen kann und 2. die gewählte Merkmalsrepräsentation nicht ausreicht, um die Modelle eindeutig voneinander zu unterscheiden. Dennoch konnten die Synthese- und Trainingsergebnisse überzeugen, sodass an der Verwendung kontinuierlicher Modelle zunächst festgehalten wird.
Mittels diskreter Modelle ist es möglich eine HMM-Struktur zu entwickeln, die in der Lage ist, die gewünschten Gesten zu erzeugen. Weiterhin konnte die korrekte Funktionsweise der bisher verwendeten Algorithmen für Training und Klassifizierung bestätigt werden. Der Fokus der nächsten Untersuchungen liegt auf CHMMs. Daher wird auf DHMMs im Verlauf nur noch in Form von Verweisen eingegangen. Ist künftig die Rede von „HMMs“ so sind, insofern nicht weiter spezifiziert, kontinuierliche Modelle gemeint.

5 Vorexperimente II

In diesem Kapitel folgt nun der Übergang zu den real von Menschen ausgeführten Bewegungen. Zu Beginn konnte nur ein sehr kleiner Testdatensatz verwendet werden der:

- für jede Geste nur 11 Aufnahmen enthält
- keine fest definierten Anfangs- und Endpunkte besitzt
- keine explizite Richtung vorgibt, sodass sowohl Kreis, als auch Dreieck gegen und mit dem Uhrzeigersinn verlaufend vorliegen

5.1 Analyse: Reale Daten und kontinuierliche Modelle

Die Visualisierung der Gesten ist im Anhang zu finden. Für jede Geste sei aber an dieser Stelle in Abbildung 5.1 ein Beispiel gegeben.

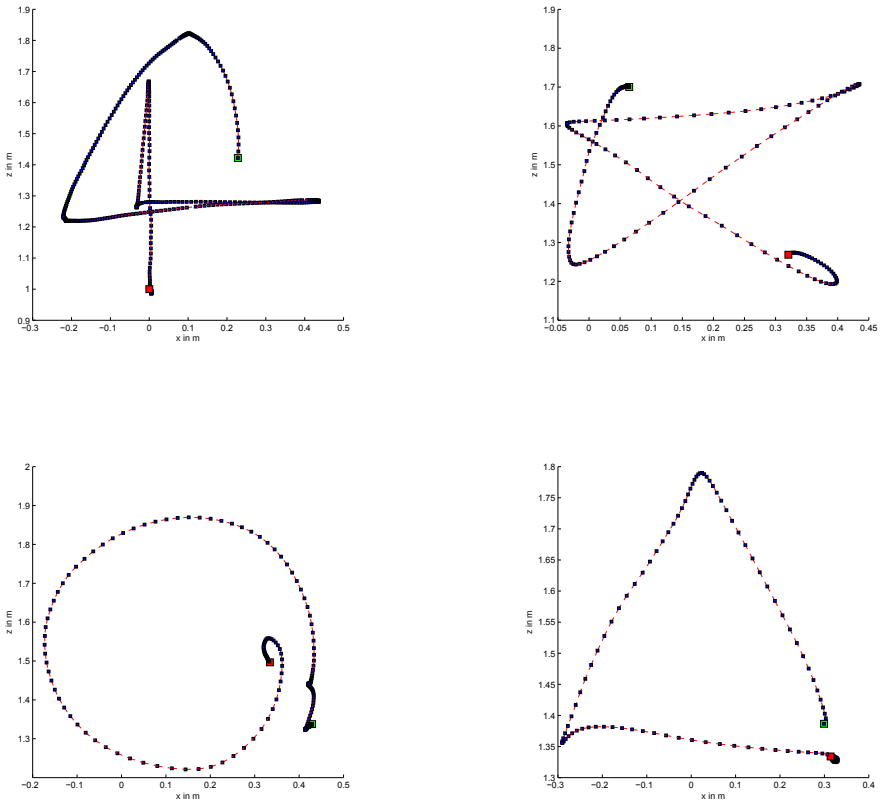


Abbildung 5.1: Beispiel - VR-Testdaten

Neben den bereits negativ genannten Aspekten ist ein Hauptproblem die geringe Anzahl vorhandener Gesten pro Modell. Würde jetzt noch eine Einteilung in Trainings- und Testdaten vorgenommen werden, so würde die Anzahl der für das Testen verwertbaren Trajektorien weiter nach unten sinken. Daher wurde hier ein experimenteller Ansatz zur Vervielfältigung der vorhanden Datensätze gewählt, der wie folgt abläuft:

- Im ersten Schritt muss gewährleistet werden, dass die Modelle dazu in der Lage sind, die für sie bestimmten Trajektorien sicher zu erkennen. So werden die vorhanden Sequenzen vollständig als Trainings- und als Testdaten verwendet.
- danach findet eine SNR-Analyse statt, die den vorhandenen Daten ein weißes Rauschen hinzufügt. Gesucht wird die dB-Grenze, an der die Klassifikation gerade noch alle Modelle korrekt bestimmen kann.
- Anschließend findet der eigentliche Trainings- und Erkennungsprozess statt, bei dem die Datensätze durch Hinzufügen eines zufälligen Rauschens anhand des ermittelten SNR-Verhältnisses vervielfältigt werden, bis ein ausreichen großer Datensatz entsteht. Das zufallsbestimmte Rauschen gewährleistet, dass sich die Testdaten von den Trainingsdaten unterscheiden.

Im ersten Schritt gilt zu überprüfen, ob die bisherigen Modelle die Trajektorien erfolgreich im zweidimensionalen Raum erkennen können. Die Umwandlung von dreidimensionalen Koordinaten in diese kann deshalb ohne Probleme vorgenommen werden, da alle Gesten parallel zur XY-Ebene ausgeführt wurden.

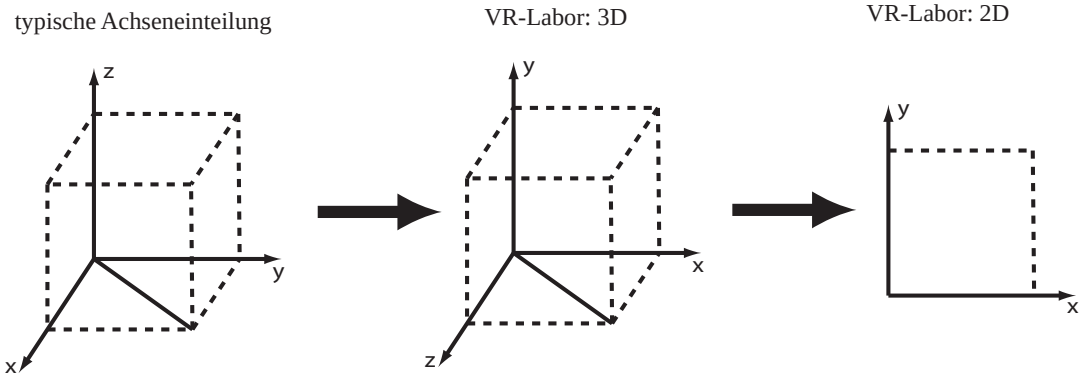


Abbildung 5.2: Koordinatensystem - VR-Labor, abweichende Achseneinteilung

Da die Orientierung auf der Basis des Differenzvektors zweier Punkte ermittelt wird, kann die Z-Ebene ignoriert werden. Dabei sei darauf hingewiesen, dass die Achseneinteilung des VR-Labors von der üblichen Beschriftung abweicht, wie es in Abbildung 5.2 zu sehen ist. Es zeigt sich ein ähnliches Verhalten, wie bei die Erkennung auf synthetische Daten ohne Training. Der Kreis weist als einziges Symbol einen 100% Wert vor (Tabelle 5.1). Darüber hinaus werden über die Hälfte aller anderen Symbole als Kreis erkannt.

Modell	wird erkannt als (in %)			
	λ_4	λ_K	λ_{Kreis}	$\lambda_{Dreieck}$
λ_4	16.6667	0	75	8.3333
λ_K	8.3333	0	66.6667	25
λ_{Kreis}	0	0	100	0
$\lambda_{Dreieck}$	0	0	58.3333	41.6667

Tabelle 5.1: Verwechslungsmatrix Testdaten - CHMM, VR-Daten und HMM-Merkmale: *acos*, *asin*

5.2 Erweiterung der Merkmale

Die Ursache für die deutlich schlechter ausfallenden Resultate, im Vergleich zu den synthetischen Daten, liegt in der Beschaffenheit der aufgezeichneten Trajektorien. Ein Problem, das sich hier zeigt, ist, dass der Anfang und das Ende einer Geste in einer kontinuierlichen Erkennung nicht genau bestimmt sind. Ein weiterer entscheidender, auf die Erkennungsrate Einfluss nehmender Effekt ist in den Übergängen zwischen zwei Liniensegmenten (an den „Ecken“) zu beobachten. An dieser Stelle verliert das Orientierungsmerkmal seine Bedeutung als Alleinstellungsmerkmal. Daraus resultiert eine enorme Beeinflussung des Trainingsprozesses, da die Varianz der möglichen Winkelrichtungen deutlich größer wird, als die, die für das eigentliche Liniensegment modelliert wurde.

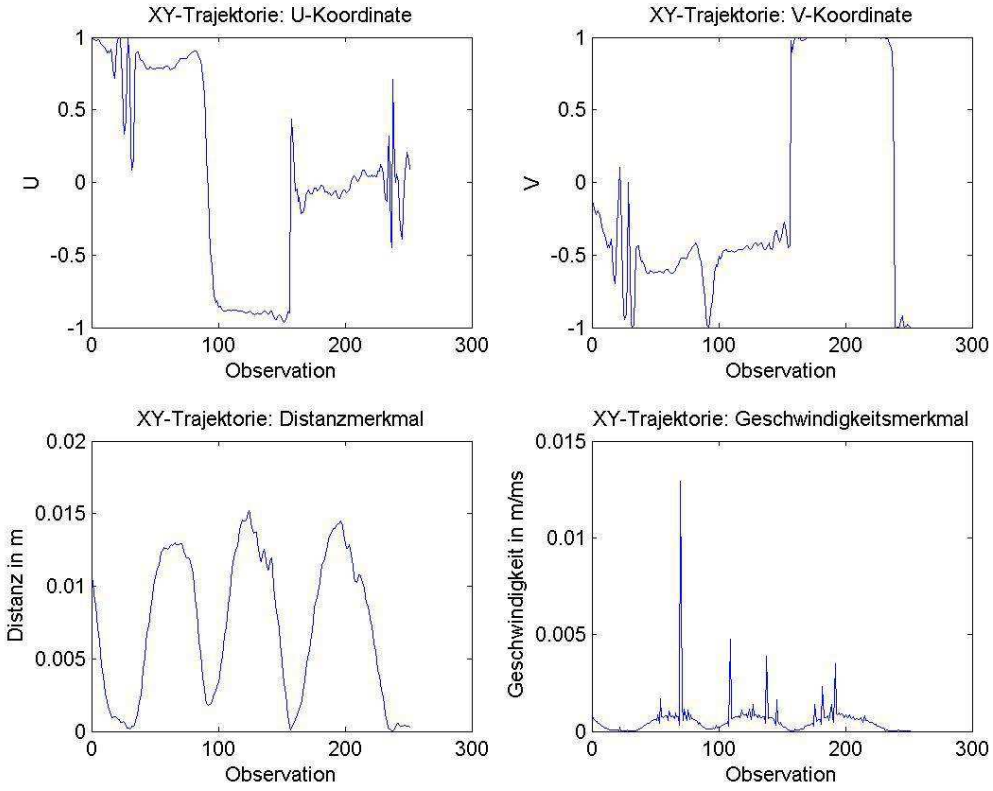


Abbildung 5.3: Korrelation von Merkmalen - Beispiel Dreieck

Aus der Darstellung 5.3 lässt sich die Vermutung aufstellen, dass die Distanzen zwischen den einzelnen xy-Punkten in diesen Übergangspunkten kürzer ausfallen, als die der eigentlichen Bewegungen. Besonders deutlich zeigt sich diese Charakteristik in dem zeitlichen Verlauf des Merkmals. Da die Distanz ein absolutes Merkmal ist, wird hier zusätzlich die Geschwindigkeit mit hinzugenommen. [Elm10] hatte darauf hingewiesen, dass diese eine wichtige Rolle an den Wendepunkten eines Bewegungsverlaufes einnehmen kann. Schlussfolgerungen wurden allerdings nicht angegeben. Der Vorteil der Geschwindigkeit ist, dass sie unabhängig von Auflösung und Abtastrate ist. Für die Distanz im zweidimensionalen Raum wird folgende Formel benutzt:

$$s = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{5.1}$$

Die Geschwindigkeit lässt sich aus der Distanz zweier Punkte und den dazu gegebenen Zeitwerten errechnen:

$$v = \frac{s}{\Delta t} \tag{5.2}$$

Somit ergibt sich ein neuer vierdimensionaler Merkmalsvektor mit:

$$O(n) = \begin{pmatrix} \cos(\phi_n) \\ \sin(\phi_n) \\ s_n \\ v_n \end{pmatrix} \tag{5.3}$$

Als nächstes gilt es daher, die gerade erläuterten Aspekte in die Modellstrukturen einfließen zu lassen. Die Bisherigen reichen für die Abbildung des signifikanten Unterschiedes in der Distanz und der Geschwindigkeit mit ihren wenigen Zuständen nicht aus, da die anfänglichen Annahmen ohne Berücksichtigung von Liniensegmentübergängen und mit Fokus auf der reinen kontinuierlichen Bewegung der Hand getroffen wurden. Das Einfügen von Übergangszuständen an Modellanfang, -ende und zwischen jedem Segmentzustand erfüllt die Bedingungen. Somit umfassen die Modelle jetzt grundlegend zwei verschiedene Arten von Zuständen. Die einen modellieren den eigentlichen Bewegungsvorgang innerhalb eines Liniensegmentes. Die zweite Art von Zuständen bildet immer einen Übergang zwischen diesen ab und kann Pausenzustand aufgefasst werden. In der Vorstellung sollen damit Anfang und Ende, sowie Ecken innerhalb der Symbole erfasst werden. Abbildung 5.4 zeigt das Schema der neuen Modellstruktur:

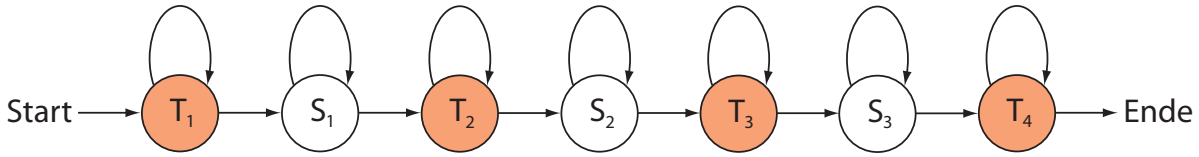


Abbildung 5.4: HMM Topologie mit Übergangszuständen: beispielhafte Darstellung eines Symbols mit 3 Bewegungszuständen („S“ -Zustände). Zu Beginn, am Ende und zwischen den eigentlichen Segment-Zuständen befinden sich die Pausen zur Modellierung von Übergängen zwischen zwei Linien (z.B.: an Ecken)

Somit ergibt sich für die Anzahl der Zustände für die modifizierten Modelle folgende Formel:

$$N_{neu} = 2N_{alt} + 1 \tag{5.4}$$

Die Wahrscheinlichkeitsverteilung innerhalb der Gestensegmente bleibt dabei gleich, mit Ausnahme der neu hinzukommenden Verteilungen für die Distanz und die Geschwindigkeit. Diese wurden auf Grundlage der vorhanden Daten so geschätzt, dass jeweils die Maxima in die bisherigen Zustände als Mittelwert einfließen und die Minima in die neu ergänzten Übergangszustände. Die Standardabweichung wurde aus dem Mittel von Maxima und Minima gebildet. Die Varianz der Winkelfunktionen wurde bei einem Mittelwert von 0 auf 1 gesetzt, da davon auszugehen ist, dass sehr viele verschiedene Winkel an den Ecken auftreten können.
Da die Aufteilung der Observationen pro Sequenz stark variiert und auch nur geschätzt werden kann, sind hier alle Übergangswahrscheinlichkeiten auf 50% gesetzt. Eine Anpassung wird durch das Training vorgenommen.
Schließlich wird eine Wiederholung des Trainings- und Erkennungsprozesses mit den neuen Erkenntnissen durchgeführt.

Modell	**wird erkannt als (in %)**			
	$\boldsymbol{\lambda_4}$	$\boldsymbol{\lambda_K}$	$\boldsymbol{\lambda_{Kreis}}$	$\boldsymbol{\lambda_{Dreieck}}$
$\boldsymbol{\lambda_4}$	100	0	0	0
$\boldsymbol{\lambda_K}$	0	91.6667	0	8.3333
$\boldsymbol{\lambda_{Kreis}}$	0	0	66.6667	33.3333
$\boldsymbol{\lambda_{Dreieck}}$	8.3333	0	8.3333	83.3333

Tabelle 5.2: Verwechslungsmatrix Testdaten - CHMM, VR-Daten und HMM-Merkmale: *acos*, *asin*, *Distanz* und *Geschwindigkeit*

Die beiden Tabellen zeigen, das sich die Erkennung der Symbole deutlich verbessert hat. Allerdings werden immer noch nicht für jedes Symbol 100% erreicht. Die sind für die Vervielfältigung des Datensatzes unbedingt notwendig, damit das optimale SNR ermittelt werden kann. Deshalb wurden alle aufgezeichneten Gesten entfernt, deren Richtung nicht mit den Modellen übereinstimmt, sodass am Ende für jede Geste noch 5 Sequenzen bleiben. Das Ergebnis der Analyse trifft nun auch die geforderten 100%, wie es Tabelle 5.3 zeigt:

HMM	**ca**	**fa**	**cd**	**fd**	**precision in %**	**Erkennungsrate (recall) in %**
$\boldsymbol{\lambda_4}$	5	0	15	0	100	100 ± 0.19868
$\boldsymbol{\lambda_K}$	5	0	15	0	100	100 ± 0.19868
$\boldsymbol{\lambda_{Kreis}}$	5	0	15	0	100	100 ± 0.19868
$\boldsymbol{\lambda_{Dreieck}}$	5	0	15	0	100	100 ± 0.19868

Tabelle 5.3: Analyse Testdaten (gefiltert) - CHMM, VR-Daten und HMM-Merkmale: *acos*, *asin*, *Distanz* und *Geschwindigkeit*

Die Evaluation (Abb. 5.5) zeigt für die einzelnen Modelle sehr unterschiedliche Werte, sodass erstmal ein Wert von 40dB übernommen wird, da die Klassifikation bis dahin

noch relativ zuverlässig funktioniert. Um auf die gleiche Anzahl Trainings- und Testdaten zu kommen, wie sie bei der Synthese vorliegen, wird Ersteres um 20 vervielfältigt (jede Sequenz wird durch Rauschen auf 20 Sequenzen erweitert) und letzteres um den Faktor 10 reproduziert.

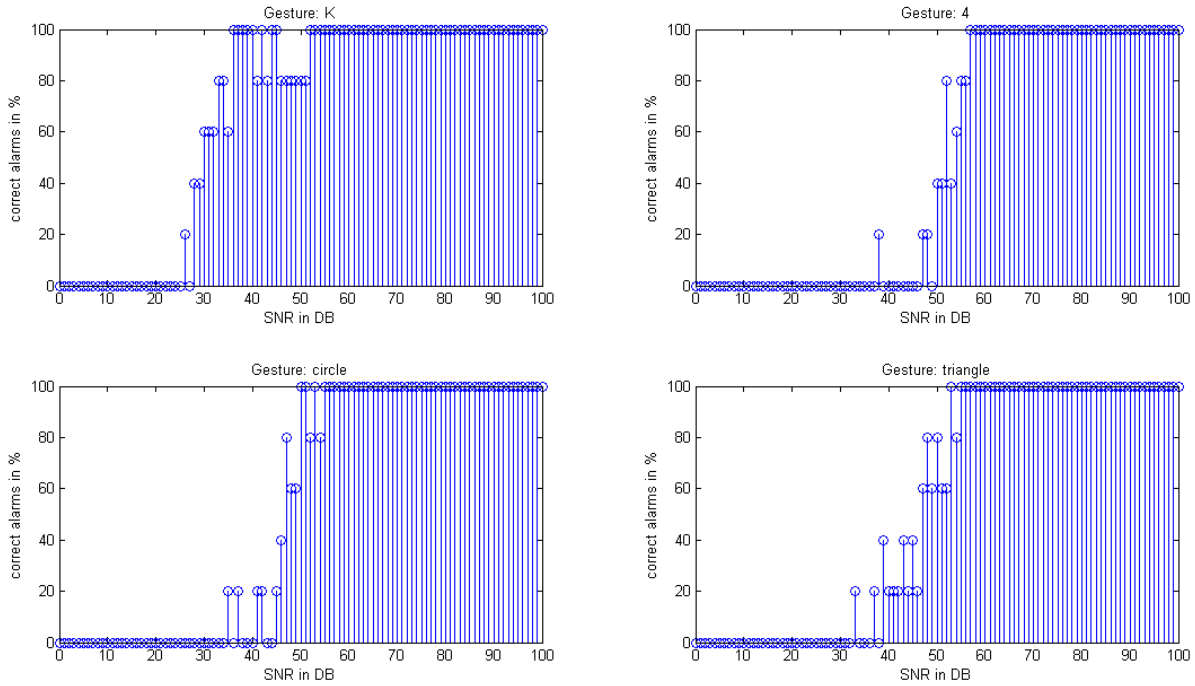


Abbildung 5.5: Evaluation - SNR, Trackingdaten (Testset)

Das Ergebnis ist eine Erkennungsrate von 100% für alle Symbole, und das für einen auf 200 Sequenzen erweiterten Datensatz pro Geste:

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	50	0	150	0	100	100 $\pm$ 0.061391
λ_K	50	0	150	0	100	100 $\pm$ 0.061391
λ_{Kreis}	50	0	150	0	100	100 $\pm$ 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 $\pm$ 0.061391

Tabelle 5.4: Analyse Testdaten (SNR) - CHMM, VR-Daten und HMM-Merkmale: *acos*, *asin*, *Distanz* und *Geschwindigkeit*

Nachdem nun Realdaten im zweidimensionalen Raum zuverlässig erkannt werden können, erfolgt nun der Übergang in den dreidimensionalen Raum.

5.3 Dreidimensionaler Merkmalsraum

Um nun dreidimensionale kartesische Raumpunkte im Sinne der Orientierung in Bezug auf einen Mittelpunkt eindeutig beschreiben zu können, sind im Gegensatz zu den Polarkoordinaten zwei Winkel notwendig. Deshalb werden zwei aufeinander folgende Punkte

$P(x_1, y_1, z_1)$, $P(x_2, y_2, z_2)$ hinsichtlich ihrer Ausrichtung durch Kugelkoordinaten beschrieben.

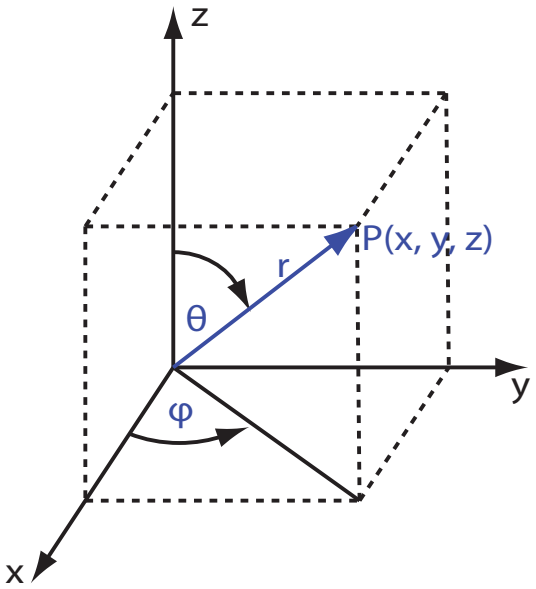


Abbildung 5.6: Im Kugelkoordinatensystem wird der Punkt $P(x_1, y_1, z_1)$ durch den Polarwinkel ϕ_1, den Azimutwinkel θ und den Radius r_1 zum Koordinatenursprung beschrieben. Eine Umrechnung zwischen den beiden Systemen erfolgt über Winkelfunktionen [Roo14].

Der dadurch entstehende Merkmalsvektor einer Observation umfasst nun die sechs Dimensionen:

$$O(n) = \begin{pmatrix} \cos(\phi_n) \\ \sin(\phi_n) \\ \cos(\theta_n) \\ \sin(\theta_n) \\ s_n \\ v_n \end{pmatrix} \tag{5.5}$$

Mit dem Wissen, dass alle Gesten parallel zur xy-Ebene ausgeführt werden, scheint die Betrachtung des zweiten Winkels in den Hintergrund zu rücken. Daher wird jeder Zustand in seiner Emissionsmodellierung für θ mit dem Mittelwert 0 und der Varianz 1 initialisiert. Wie im zweidimensionalen Raum wird zunächst eine Überprüfung des ganzen Testdatensatzes durchgeführt. Im Gegensatz, zu dem Vorherigen fallen die Ergebnisse hier deutlich besser aus (Tabelle 5.5). Obwohl Kreis- und Dreieck-Sequenzen in beide Richtungen gezeichnet worden sind, wird nur eine einzige Trajektorie falsch zugeordnet. Somit bildet der zweite Raumwinkel ein unverzichtbares Element für die Robustheit der Erkennung, da sich darin für die Geste typische Beobachtungen verbergen. Außerdem scheinen die hinzugefügten Übergangszustände starke Strukturinformationen zu beinhalten.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	12	1	35	0	92.3077	100 ± 0.12632
λ_K	12	0	36	0	100	100 ± 0.12632
λ_{Kreis}	12	0	36	0	100	100 ± 0.12632
$\lambda_{Dreieck}$	11	0	36	1	100	91.6667 ± 0.12261

Tabelle 5.5: Analyse Testdaten - CHMM, VR-Daten und HMM-Merkmale: $acos(\phi)$, $asin(\phi)$, $acos(\theta)$, $asin(\theta)$, *Distanz* und *Geschwindigkeit*

Die nächste Messung umfasst die durch SNR vervielfältigten Sequenzen. Für alle Testdaten wurde ein Rauschabstand von $45dB$ verwendet. Da hier jedes Symbol nun 11 Testdaten hat, wurden die Faktoren 10 für Trainingsdaten und 5 für Testdaten zur Sequenz-Reproduktion benutzt.

Modell	wird erkannt als (in %)			
	λ_4	λ_K	λ_{Kreis}	$\lambda_{Dreieck}$
λ_4	100	0	0	0
λ_K	0.83333	98.3333	0	0.83333
λ_{Kreis}	0.83333	1.6667	97.5	0
$\lambda_{Dreieck}$	38.3333	0	0	61.6667

Tabelle 5.6: Verwechslungsmatrix Testdaten (SNR) - CHMM, VR-Daten und HMM-Merkmale: ewline $acos(\phi)$, $asin(\phi)$, $acos(\theta)$, $asin(\theta)$, *Distanz* und *Geschwindigkeit*

Da die aus der Tabelle 5.6 zu entnehmende Robustheit die 100% nicht für jedes Symbol erreicht, wurde noch eine Messung mit dem gefilterten Datensatz (5 Sequenzen pro Geste mal 20/ 10) durchgeführt. Hier zeigen die Ergebnisse das gewünschte Verhalten:

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	50	0	150	0	100	100 ± 0.061391
λ_K	50	0	150	0	100	100 ± 0.061391
λ_{Kreis}	50	0	150	0	100	100 ± 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 ± 0.061391

Tabelle 5.7: Analyse Testdaten (gefiltert) - CHMM, VR-Daten und HMM-Merkmale: $acos(\phi)$, $asin(\phi)$, $acos(\theta)$, $asin(\theta)$, *Distanz* und *Geschwindigkeit*

6 Versuchsaufbau und Evaluierung

Schwerpunkt dieses Kapitels bildet die Evaluierung der in den vorangegangenen Abschnitten beschriebenen Verfahren und Gegebenheiten für den dreidimensionalen Raum. Da es nun möglich ist, die Trajektorien synthetisch zu erzeugen und die HMMs für den Erkennungsprozess erfolgreich einzusetzen, steht nun eine Untersuchung der Robustheit, Flexibilität und Eignung der strukturellen und parametrischen Eigenschaften der ermittelten Modelle an. In diesem Rahmen wurde eine Studie mit insgesamt 22 Probanden durchgeführt. Zunächst folgt ein kurzer Einblick in die dafür notwendige Datenverwaltung und Verarbeitung des Systems gegeben.

6.1 Softwarearchitektur

Auf eine komplette Darstellung aller verwendeten Funktionen und Konstrukte wird an dieser Stelle verzichtet und auf die der beiliegenden CD enthaltenen Dokumentation verwiesen. Vielmehr geht es hier darum, die wesentlichen Komponenten zu umranden und die Anforderungen zu benennen, die nötig sind, um die zu Beginn beschriebene Problemstellung zu lösen, eine nachvollziehbare Darstellung der dafür unternommenen Schritte zu erhalten und schließlich eine zu jedem Zeitpunkt reproduzierbare Form der Ergebnisse zu schaffen.

6.1.1 Anforderungen

Die Anforderungen an das zu erstellende „Synthese-Analyse-System“ lassen sich grob in drei Punkten zusammenfassen:

1. permanente Speicherung von Modellstrukturen, Parametern und Emissionsfolgen
2. grafische Darstellung von Trainings- und Analysedaten
3. Visualisierung sämtlicher Prozesse, um deren Wirkung analysieren zu können

Punkt 1 beinhaltet das Festlegen eines Formates, das dazu in der Lage ist, die Struktur eines HMMs vor und nach dem Trainingsprozess vollständig zu erfassen. Hierfür wurde eine XML-Repräsentation gewählt, mit der sich strukturbasierte Konstrukte (wie HHMs) sehr gut darstellen lassen und zusätzlich theoretisch ein System-übergreifender Austausch möglich ist, da ein XML-Dokument unabhängig von der Matlab-Umgebung ist.
Ebenso wird jede Observationsfolge in einem XML-Dokument bzw. einer CSV-Datei abgelegt. Der Vorteil hier ist, dass sich so Meta-Informationen, wie z.B.: eine Merkmalsbeschreibung oder Wertebereiche hinzufügen lassen, die aus der reinen Observationsmatrix nicht ersichtlich wären. Synthetisch erzeugte Daten weisen ein zusätzliches Tag, welches die einzelnen Beobachtungsvektoren einem Zustand zuordnet. Wie zu Beginn erwähnt, bilden dreidimensionale Raumkoordinaten die Basis für die Gestenerkennung. Da HMMs aber meist nicht auf diesen Daten arbeiten, sonder auf abstrahierten Merkmalen, ist es

notwendig, grafisch zu zeigen, dass Emissionsfolge und Merkmalsfolge korrelieren. Zusätzlich soll ersichtlich sein, dass die gewählte Beschreibung von Eigenschaften dazu geeignet ist, um unterschiedliche Gesten eindeutig abzubilden. Letzteres, was mit Punkt 2 erfüllt werden soll, ist die Visualisierung von synthetischen Gesten, um schlussfolgern zu können, ob ein Modell dazu in der Lage ist, eine Geste auch wirklich zu emittieren.
Der unter Punkt 3 genannte Aspekt umfasst im Wesentlichen die Trainings-, Synthese- und Analyseprozesse. Es soll gezeigt werden können, wie das Training die Parameter der HMMs beeinflusst, welche Observationen aus welchen Zuständen generiert werden, wie Wahrscheinlichkeiten verschiedener Modelle verglichen werden und wie die Zuordnung von Sequenz und Modell vorgenommen wird.

6.1.2 Bibliotheken

Da der Kern der Arbeit weitestgehend mathematische Problemstellungen umfasst und dabei ein hoher Grad an Flexibilität und Visualisierung wünschenswert ist, wurde die Programmiersprache und -umgebung „Matlab“ in der Version 7.11.0.584 (R2010b) eingesetzt. Die hier zur Verfügung stehende Vielfalt an Software-Paketen (in Matlab als „Toolboxes“ bezeichnet) und Schnittstellen erfüllen die wichtigsten Kriterien, die für die Realisierung dieser Arbeit notwendig sind. Schließlich galt es, die geeignete Toolbox für die hier beschriebene Umsetzung zu finden. Dabei seien die wichtigsten in Frage kommenden Bibliotheken kurz in einer tabellarischen Gegenüberstellung vorgestellt:

Toolbox	**Beschreibung**	**Vorteile**	**Nachteile**
Statistics Toolbox	enthält zahlreiche Algorithmen für Statistik und maschinelles Lernen Quelle: [MaW14]	großer Funktionsumfang für gesamten Bereich der Statistik von MathWorks herausgegeben	Fokus auf diskreten Modellen im HMM-Kontext wird meist auf alternative Bibliotheken ausgewichen kostenpflichtig
PMTK3	steht für „Probabilistic Modeling Toolkit“ und bietet riesigen Funktionsumfang zur Modellierung, Visualisierung und Beschreibung statistischer Prozesse mit Bezug auf konkrete Anwendungsfelder uvm. zur Unterstützung des Buches „Machine learning: a probabilistic perspective“ von Murphy entwickelt Quelle: [Mur14]	Nachfolger von HMM Toolbox (Murphy) riesiger Funktionsumfang frei verfügbar	uneingeschränkte Nutzung setzt diverse kostenpflichtige Matlab-Module voraus, sowie weitere Software-Komponenten keine reine Matlab-Implementierung seit 2011 keine Feature-Erweitrung mehr, sondern nur noch Fehlerbeseitigung

HMM Toolbox (Kevin Murphy)	Schwerpunkt: HMMs basierend auf diskreten, normal- oder mischverteilten Emissionen Quelle: [Mur05]	Funktionen erfüllen Anforderungen an vorliegende Problemstellung viele Funktionen mit Demos zum Thema HMM Vorgänger von PMTK keine Zusatzmodule notwendig frei verfügbar	letzte Aktualisierung 2005 weniger Funktionen zur Visualisierung von Teilprozessen ausführliche Dokumentation ist nur direkt im Quellcode zu finden

Tabelle 6.1: Vergleich von Matlab-Bibliotheken

Nach dem Vergleich der drei näher beleuchteten Bibliotheken fiel die Auswahl auf die von Kevin Murphy entwickelte "HMM"Matlab-Toolbox, da hier alle benötigten Funktionen für das Erzeugen, Trainieren und Auswerten von HMMs vorhanden sind. Der überschaubare aber dennoch ausreichende Funktionsumfang ermöglicht es, eigene Implementierungen und Ideen leichter in die Bibliothek zu integrieren, sodass zusätzlich Algorithmen z.B.: aus dem Bereich Visualisierung o.ä. hinzugefügt werden können. Ein weiterer Vorteil dieser Bibliothek liegt darin, dass neben der bekannten Netlab-Toolbox noch drei weitere Pakete enthalten sind. Dazu gehört ein HMM-Paket, welches die gängigen Algorithmen für Training und Analyse, sowie Funktionen zum Generieren von Testsequenzen enthält. Außerdem sind die Verzeichnisse „KPMstats“ und „KPMtools“ enthalten, die zusätzlich Funktionen zur Lösung statistischer Problemstellungen beinhalten. Einziger Nachteil ist die etwas sparsame Dokumentation, denn es wurde nur im Quellcode kommentiert. Eine Übersicht über alle vorhanden Funktionen ist leider nicht enthalten.

6.1.3 Implementierung

Da die benutzten Softwarekomponenten im Vorab genannt wurden und der dieser Arbeit separat beiliegende Beschreibung der verwendeten Funktionen sowie dem Quellcode bereits detaillierte Informationen zur Implementierung vorliegen bzw. nachgeschlagen werden können, wird die zugrunde liegende Architektur nur kurz umrissen. Da Matlab auf der Basis von Matrizen arbeitet, muss im Großen und Ganzen auf die Implementierung von HMMs nicht weiter eingegangen werden, da die formale Beschreibung aus Kapitel 2.3 gezeigt hat, das sich die hier vorliegenden Modelle weitestgehend durch Matrizen und Vektoren beschreiben lassen. Eine Bemerkung sei allerdings noch zur programmatischen Darstellung der Emissionen gemacht. Während für die Realisierung diskreter Emissionen lediglich eine Matrix notwendig ist, werden für kontinuierliche Modelle mit Normal- bzw. Mischverteilungen drei Komponenten benötigt. Diese lassen sich, unter den in Kapitel 2.3.2 getroffenen Annahmen wie folgt beschreiben:

- eine Matrix μ, welche die Mittelwerte der einzelnen Normalverteilungen pro Zustand enthält:

$$\mu = \begin{pmatrix} \mu_{N_1 1} & \mu_{N_1 2} & \dots & \mu_{N_1 n} \\ \mu_{N_2 1} & \mu_{N_2 2} & \dots & \mu_{N_2 n} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{N_n 1} & \mu_{N_n 2} & \dots & \mu_{N_n n} \end{pmatrix} \tag{6.1}$$

  jede Normalverteilung beschreibt dabei eine Dimension (Zufallsvariable) des Merkmalsvektors

- einen Vektor c, welcher die Gewichtungen 1 der einzelnen Matrizen pro Zustand enthält,

$$c = \begin{bmatrix} c_1 & c_2 & \dots & c_n \end{bmatrix} \tag{6.2}$$

- einen Tensor K in der Matlab-Umgebung auch als σ bezeichnet, der die Kovarianzmatrizen der einzelnen Zufallsvariablen enthält

$$\sigma = \begin{pmatrix} cov_{11} & cov_{12} & \dots & cov_{1n} \\ cov_{21} & cov_{22} & \dots & cov_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ cov_{n1} & cov_{n2} & \dots & cov_{nn} \end{pmatrix} \tag{6.3}$$

Zur kompakten Darstellung und Speicherung von HMMs wurde ein objektorientierter Ansatz gewählt, der alle zu einem HMM gehörenden Eigenschaften in einer Klasse zusammenfasst. Dazu gehören neben den zur vollständigen Beschreibung eines HMM notwendigen Komponenten zusätzliche Information, wie z.B.: Modellname, repräsentiertes Merkmal und andere. Mehr Informationen sind in Kapitel 6.1.4 zu finden. Darüber hinaus lassen sich Objekt-Eigenschaftsbeziehungen sehr gut im XML-Format darstellen.
Für das Erzeugen und Lesen von XML-Dokumenten wird der in Matlab zur Verfügung stehende DOM-Parser verwendet, der ein XML-Dokument nach dem Standard des „W3C“ als Baumstruktur repräsentiert. Dieses Objekt befindet sich dabei die ganze Zeit im Speicher des Rechners. Das ist immer dann sinnvoll, wenn während einer Applikation Elemente öfter geändert oder gelesen werden sollen(z.B.: Ändern von HMM-Parametern nach einem Trainingsprozess) [UZI14].

6.1.4 XML-Darstellung

Um eine Reproduzierbarkeit der vorliegenden Ergebnisse zu gewährleisten, ist es zwingend notwendig, alle Observationsfolgen und verwendeten Modelle in einer Datei abzulegen, um sie so später wieder aufrufen zu können. Hier wurde eine XML-Struktur entwickelt, die alle benötigten Werte und Parameter beschreibt. Darüber hinaus kann das Format dazu verwendet werden, um Daten programmiersprach- oder systemübergreifend zu verarbeiten.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<hmm name="left" symbol="triangle" description="gesture" class="geometric" type="
    discrete" states="1" date="2014-03-22">
  <state number="1" startProbability="100" endProbability="0">
    <observation name="angle" type="orientation">
      <value probability="50" component="true">1</value>
      <value probability="0" component="true">2</value>
      <value probability="0" component="true">3</value>
      <value probability="0" component="true">4</value>
      <value probability="0" component="true">5</value>
      <value probability="0" component="true">6</value>
      <value probability="0" component="true">7</value>
      <value probability="0" component="true">8</value>
      <value probability="0" component="true">9</value>
      <value probability="0" component="true">10</value>
      <value probability="0" component="true">11</value>
      <value probability="0" component="true">12</value>
      <value probability="0" component="true">13</value>
      <value probability="0" component="true">14</value>
      <value probability="0" component="true">15</value>
      <value probability="0" component="true">16</value>
      <value probability="0" component="true">17</value>
      <value probability="50" component="true">18</value>
    </observation>
  </state>
</hmm>
```

Listing 6.1: XML: DHMM Struktur

Listing 6.1 zeigt einen Auszug einer beispielhaften XML-Datei für ein diskretes HMM mit einer Gesamtzustandsanzahl von 1. Für das HMM wird ein Name festgelegt, eine Symbolzuordnung vorgenommen, optional eine Beschreibung, eine Klasse, ein Typ und ein Datum definiert, sowie die Anzahl der enthaltenen Zustände angegeben. Explizit gesetzt werden müssen hier Name und Symbol. Beschreibung, Klasse und Typ sind dafür gedacht, wenn im Nachgang der Arbeit die entwickelten Algorithmen für andere Bereiche außer der Gestenerkennung benutzt werden sollten. Sie dienen dabei der Zuordnung.
Für jeden Zustand wird eine Nummer (optional), die a priori-Wahrscheinlichkeit und die für einen Übergang in den Endzustand definiert. Alle Wahrscheinlichkeiten werden generell in Prozent angegeben. Die Nummer ist optional, da die Zustände beim Einlesen gezählt werden. Das Tag könnte später für eine Evaluierung in Kombination mit dem *<states>*-Tag des HMM genutzt werden, um die in dem XML-Dokument definierte Modellstruktur auf Korrektheit zu überprüfen.
Jeder Zustand enthält alle möglichen Emissionen mit den dazu gehörigen Wahrscheinlichkeiten. Name, Typ und Komponente sind dabei optional. Die Komponente gibt an, ob die Emission zu dem eigentlichen Symbol gehört (*true*), oder dazu genutzt wird, um Absetzvorgänge zu erfassen (*false*). Würde zum Beispiel ein „X“ mit einem Stift geschrieben werden, müsste zwischen der ersten und der zweiten Linie des Buchstaben abgesetzt und der Schreibvorgang unterbrochen werden. Das HMM muss diesen Absetzvorgang jedoch ebenfalls modellieren. In diesem Fall werden Observationen erzeugt, die nicht zum Symbol gehören.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<hmm name="left" symbol="triangle" description="gesture" class="geometric" type="
    continuous" states="3" date="2014-03-24">
  <state number="1" startProbability="100" endProbability="0">
    <observation name="angle" type="orientation">
      <gauss component="true">
        <mu>0</mu>
        <sigma>1</sigma>
        <weight>1</weight>
        <subinfo>Meta-Info</subinfo>
      </gauss>
    </observation>
    <observation name="velocity" type="velocity">
      <gauss component="true">
        <mu>0</mu>
        <sigma>1</sigma>
        <weight>1</weight>
      </gauss>
    </observation>
  </state>
</hmm>
```

Listing 6.2: XML: MHMM Struktur

Wie in *Listing 6.2* zu sehen ist, bleibt die Grundstruktur für CHMMs gleich. Lediglich die Emissionsmodellierung unterscheidet sich von dem Vorangegangenen. So können hier eine oder mehrere Gauß-Komponenten mit den typischen Parametern, wie Mittelwert, Varianz und Gewicht (wichtig für die Verwendung von Mixturen, da die Summe über alle 1 sein muss) beschrieben werden. Das *<subinfo>*-Tag bietet eine Möglichkeit, um Zusatzinformationen, z.B.: für eine Beschreibung der Merkmale, abzulegen (z.B. *<subinfo>*Winkel von 0° bis 20°*</subinfo>*).

```
  <stateTransitions>
    <transition start="1" destination="1">94.680851</transition>
    <transition start="1" destination="2">5.319149</transition>
    <transition start="2" destination="2">94.680851</transition>
    <transition start="2" destination="3">5.319149</transition>
    <transition start="3" destination="3">100</transition>
  </stateTransitions>
```

Listing 6.3: XML: HMM Zustandsübergänge

Die Zustandsübergänge werden für diskrete und kontinuierliche Modelle auf die gleiche Art beschrieben, sodass an dieser Stelle nur ein Beispiel unter *Listing 6.3* angegeben wird. Definiert werden hier nur Übergänge, die eine Wahrscheinlichkeit größer 0 aufweisen. Alle weiteren Übergänge werden in der A-Matrix beim Einlesen auf 0 gesetzt. Das *<start>*-Tag gibt an, wo ein Übergang beginnt und das *<destination>*-Tag definiert den Zielzustand. Bei einem L-Modell muss der letzte Zustand immer 100 % betragen, da wie bereits beschrieben, die Wahrscheinlichkeiten für einen Übergang in den Endzustand in einem separaten Vektor erfasst werden.

```
<?xml version="1.0" encoding="utf-8"?>
<observation-data date="2014-08-19" name="(triangle) RU_U_LU_RU" time="17
    :12:50:098">
   <state number="1">
      <observationValue number="1">
         <value>6</value>
         <value>8</value>
         <value>6</value>
      </observationValue>
    <observationValue number="1">
         <value>0.1</value>
         <value>0.2</value>
         <value>0.1</value>
      </observationValue>
   </state>
   <state number="2">
      <observationValue number="1">
         <value>12</value>
         <value>13</value>
         <value>11</value>
         <value>11</value>
      </observationValue>
    <observationValue number="1">
         <value>0.5</value>
         <value>0.45</value>
         <value>0.6</value>
         <value>0.324</value>
      </observationValue>
   </state>
</observation-data>
```

Listing 6.4: XML: Observationen

Schließlich folgt noch ein Beispiel für das Speichern von Observationsfolgen. Wie in *Listing 6.4* dargestellt, sind alle zu einem Merkmal gehörenden Observationen unter einem *<observationValue>*-Tag zusammengefasst. Handelt es sich um einen Merkmalsvektor, gibt es entsprechend seiner Dimension, mehrere dieser Tags. Alle so erfassten Merkmale müssen am Ende die gleiche Anzahl von Werten enthalten. Das *<state>*-Tag wird nur geschrieben, wenn bekannt ist, welche Observationen zu welchem Zustand gehören. Das ist hauptsächlich bei der Synthese möglich, da der dafür entwickelte Algorithmus diese Information liefert. Die im Labor aufgezeichneten Daten haben das nicht und werden einfach unter einem *<state>*-Tag gespeichert.

6.2 Probanden

Zu Beginn der Durchführung hatte jeder Proband die Möglichkeit, jedes Symbol fünfmal nach eigenem Ermessen auszuführen. So konnten Start- und Endpunkt, sowie Bewegungsabfolge frei und individuell, unabhängig von vorhandenen Modellstrukturen gewählt werden. Ziel dabei ist es, die Robustheit des Systems hinsichtlich einer größeren Varianz möglicher Strukturen zu überprüfen. Alle beliebig ausgeführten Trajektorien werden ausschließlich für den Analyse-Prozess und nicht für den Trainingsprozess verwendet, da deren struktureller Aufbau erstmal unbekannt ist und nicht zwangsläufig die in den Modellen definierte Zustandsfolge widerspiegelt. So lassen sich in der Auswertung Rückschlüsse über die Allgemeingültigkeit und Mächtigkeit der entwickelten Modelle treffen.

In weiteren Durchläufen sind den Probanden jeweils zwei Strukturen pro Symbol vorgegeben worden. Diese umfassen die bereits bekannten und bis hierher verwendeten 4 CHMMs. Zusätzlich wurden für jedes Symbol ein weiteres Modell mit einer sich meist nur in einem Zustand unterscheidende Ausführungsform ergänzt. Der Gestensatz umfasst damit die nachstehenden Bewegungen:

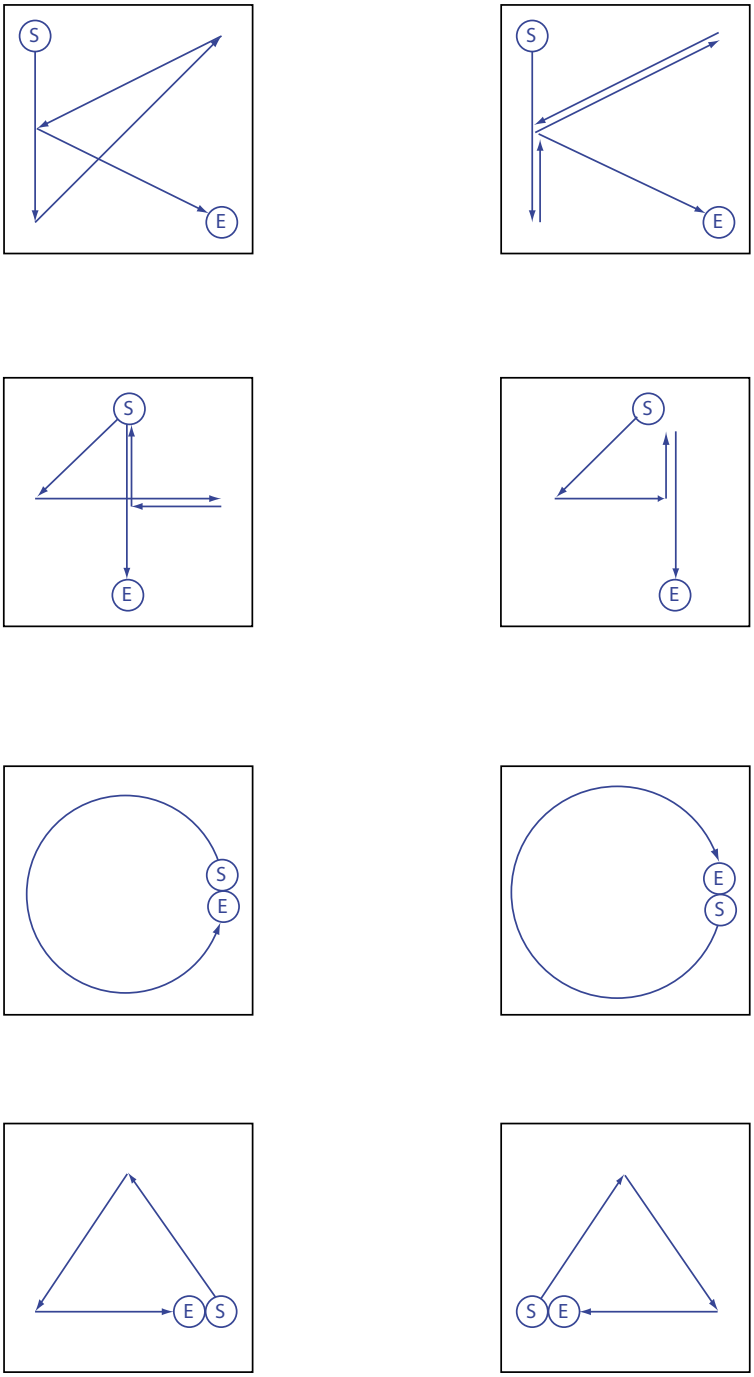


Abbildung 6.1: Gestensatz - Vorgaben für Probanden, „S“ steht für Startpunkt und „E“ für Endpunkt

Aufgrund der bereits vorgenommenen Untersuchungen lassen sich im Vorab schon einige Vermutungen bezüglich des erwarteten Ergebnissen aufstellen:

- Eine hohe Anzahl von Probanden kann die Varianz vergrößern. Dies kann den Erkennungsprozess erschweren und zu einer größeren Fehlerrate führen.
- Ohne der konkreten Vorgabe der gewünschten Gestenausführung, können Bewegungsabläufe auftreten, die nicht zu den Modellen passen.
- Für eine erfolgreiche Erkennung müssen alle Probanden von der selben Position aus agieren und die Ausführung der Gesten möglichst in der selben räumlichen Ebene vornehmen.
- Große Varianzen in der Geschwindigkeit nehmen großen Einfluss auf den Erkennungsprozess.
- Große Varianzen in der Ausdehnung der ausgeführten Geste nehmen bei geringen Varianzen in der Geschwindigkeit geringeren Einfluss auf den Erkennungsprozess.
- Bei gleicher Positionierung aller Probanden ist der Bereich der Gestenausführung in der Ebene beliebig verschiebbar.
- Je detaillierter die Anweisungen ausfallen, desto einheitlicher werden die Gesten ausgeführt. Je weniger Anweisungen gegeben werden, desto mehr Varianz ist zu erwarten.
- Wird die gleiche Geste mehrmals hintereinander ausgeführt, ist zu erwarten, das sich die einzelnen Bewegungsabläufe bezüglich jedes Probanden gleichen.
- Die sich stark unterscheidenden Gesten, die bereits untersucht wurden, lassen aufgrund der guten Erkennungsergebnisse erwarten, dass die vorhandenen Modelle die einzelnen Gesten bereits sehr gut beschreiben ()zumal an diese Stelle auf korrekte Ausführung der Gesten geachtet wird).

Eine genaue Auswertung kann dem Anhang entnommen werden, da sich dort sowohl Messungen zur Merkmalsdarstellung, als auch zu verschiedenen Datensätzen befinden.

HMM	**ca**	**fa**	**cd**	**fd**	**precision in %**	**Erkennungsrate (recall) in %**
λ_4	44	0	132	0	100	100 $\pm$ 0.065465
λ_K	44	0	132	0	100	100 $\pm$ 0.065465
λ_{Kreis}	44	0	132	0	100	100 $\pm$ 0.065465
$\lambda_{Dreieck}$	44	0	132	0	100	100 $\pm$ 0.065465

Tabelle 6.2: Analyse Probanden - CHMM, Gesten 1 - 4

Das Fazit der Tabelle 6.2 lässt sich so zusammentragen, dass mit der gewählten Merkmalsrepräsentation 100% Erkennungsgenauigkeit erreicht wird. Des Weiteren lässt sich festhalten, dass die Messungen, in der beide Symbolvarianten gleichzeitig eingesetzt wur-

den, gezeigt haben, dass der Großteil, der Sequenzen zu nur einem Modell klassifiziert wird (Tabelle unten).

Modell	wird erkannt als (in %)							
	λ_4	λ_4 (2)	λ_K	λ_K (2)	λ_{Kr}	λ_{Kr} (2)	λ_{Dr}	λ_{Dr} (2)
λ_4	97.7273	2.2727	0	0	0	0	0	0
λ_4 (Var2)	83.3333	16.6667	0	0	0	0	0	0
λ_K	0	0	93.1818	6.8182	0	0	0	0
λ_K (Var2)	0	0	2.2727	97.7273	0	0	0	0
λ_{Kreis}	0	0	0	0	100	0	0	0
λ_{Kreis} (Var2)	6.8182	0	9.0909	0	0	84.0909	0	0
$\lambda_{Dreieck}$	0	0	0	0	0	0	100	0
$\lambda_{Dreieck}$ (Var2)	0	0	0	0	4.5455	0	0	95.4545

Tabelle 6.3: Analyse Probanden - CHMM, Gesten 1 - 8

Besonders interessant ist an dieser Stelle nochmal die Untersuchung der Bedeutung der Übergangszustände. Bei den nächsten beiden Messungen wurden nach dem Training einmal die eigentlichen Bewegungszustände aus denn Modellen entfernt und zum anderen die Übergangszustände. Dabei soll überprüft werden, ob nach dem Training eine Reduktion der Zustände möglich ist.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	43	9	123	1	82.6923	97.7273 $\pm$ 0.064962
λ_K	38	12	120	6	76	86.3636 $\pm$ 0.062206
λ_{Kreis}	36	0	132	8	100	81.8182 $\pm$ 0.060984
$\lambda_{Dreieck}$	38	0	132	6	100	86.3636 $\pm$ 0.062206

Tabelle 6.4: Analyse Probanden - CHMM, Gesten 1 - 4, ohne Linien-Segment-Zustand

Ein recht überraschendes Ergebnis (Tabelle oben) und damit ein guter Anknüpfungspunkt an diese Arbeit zeigt das Verhalten der Zwischenzustände. Werden nur diese verwendet, so liegen die Erkennungsraten für jedes Modell immer noch bei über 76%. Zum einen lässt sich das damit erklären, dass diese Zustände dafür gedacht sind, um eine große Varianz an Winkeln abzudecken und zum anderen lässt sich vermuten, dass hier eventuell Zustände reduziert werden könnten. Weitere Untersuchungen diesbezüglich sind aber im Rahmen dieser Arbeit nicht möglich.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	0	132	0	44	0	0 ± 0
λ_K	0	0	132	44	NaN	0 ± 0
λ_{Kreis}	0	0	132	44	NaN	0 ± 0
$\lambda_{Dreieck}$	0	0	132	44	NaN	0 ± 0

Tabelle 6.5: Analyse Probanden - CHMM, Gesten 1 - 4, ohne Zwischenzustand

Kommt nur die Anwendung der Linien-Segment-Zustände zum Einsatz, so ist keine Klassifizierung möglich, da die Wahrscheinlichkeiten so niedrig sind, dass sie vom Rechner nicht mehr darstellbar sind (-Inf). Dies bestätigt diese Art von Zuständen bezüglich ihrer Aufgabe, möglichst lineare Bewegungsabläufe zu modellieren. Besteht das Modell nur aus solchen, die Trajektorie hingegen aus großer Winkel-Varianz, sind genau solche Werte von der Klassifizierung zu erwarten.

6.3 Erhöhung der Robustheit durch Garbage-Modell

Der Erkennungsprozess nach dem vorangegangenem Verfahren führt dazu, dass immer ein Modell zurück geliefert wird, selbst wenn die Bewertung sehr niedrig ist. Die Ursache dafür liegt in der argmax() - Entscheidung, die es zwingend notwendig macht, sich für ein HMM zu entscheiden. Das führt dazu, dass auch beliebig ausgeführte Bewegungen immer auf eine relevante Geste abgebildet werden.
Eine in der Literatur zu findende Möglichkeit, um informationstragende von beliebigen Beobachtungen zu unterscheiden, ist die Verwendung von sogenannten Garbage-Modellen [elm10][Ric11][LeK99]. Dabei wird ein zusätzliches HMM zu den vorhanden ergänzt, welches alle nicht relevanten Bewegungen erfassen soll. Erzeugt wird dieses aus den übrigen Modellen der Schlüsselgesten. Als Bildungsvorschrift sind drei grundlegende Schritte durchzuführen [Elm10]:

- alle Zustände der vorhandenen Modelle werden mit ihren Emissions- und Übergangswahrscheinlichkeiten für „self loops“ zu dem Garbage-Modell hinzugefügt.
- Neuschätzung aller Emissionswahrscheinlichkeiten durch:

$$garbage(b_j(\lambda)) = \frac{1}{\sqrt{2\pi\sigma}} \times \exp\left(\frac{b_j(\lambda)^2}{2\sigma^2}\right) \tag{6.4}$$

- Setzen aller Wahrscheinlichkeiten für Zustandsübergänge durch:

$$\hat{a}_{ij} = \frac{1 - a_{ij}}{N - 1};\ f\ddot{u}r\ alle\ j,\ i \neq j \tag{6.5}$$

Abbildung 6.2 zeigt die auf dieser Grundlage entstehende ergodische Struktur des Garbage-Modelles:

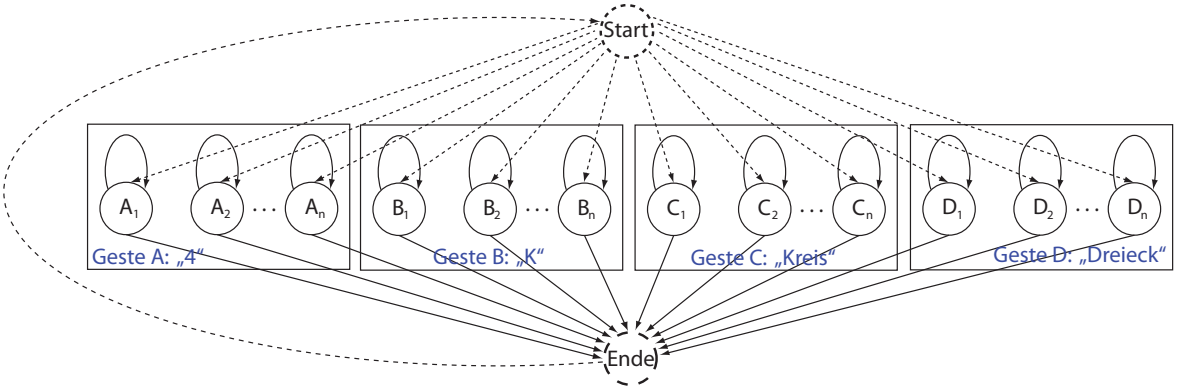


Abbildung 6.2: HMM Topologie: Vereinfachte Darstellung der ergodischen Strukturform des Garbage-Modelles am Beispiel der vier Modelle: λ_4, λ_K, λ_{Kreis} und $\lambda_{Dreieck}$ [elm10]

Eine Anmerkung sei noch zur zweiten Bildungsvorschrift gemacht. Die aus Quelle [Elm10] vorgegebenen Schritte beziehen sich auf diskrete Modelle. Da in dieser Arbeit letztendlich mit kontinuierlichen HMMs gearbeitet wird, bei denen die Emissionsmodellierung durch Normalverteilungen erfolgt, wird hier anstelle der angegebenen Formel die Kovarianzmatrix mit einem experimentell ermittelten Faktor c multipliziert. Somit ergibt sich dafür die folgende Gleichung:

$$garbage(K_j(\lambda)) = c \times garbage(K_j(\lambda)) \tag{6.6}$$

Die Bedingung für den folgenden Test sind die selben, wie aus dem vorherigen:

- CHMMs für den dreidimensionalen Raum mit 6-dimensionalem Merkmalsvektor
- Trainingsdaten: 20 × 5 durch Rauschen variierte Sequenzen
- Testdaten: 10 × 5 durch Rauschen variierte Sequenzen
- Für das Garbage-Modell selbst sind keine Sequenzen vorhanden

Nach dem Trainingsprozess wird aus den dadurch hervorgegangenen Modellen ein Garbage-Modell erzeugt und als gleichberechtigtes HMM in die Klassifikation integriert. Bei direkt übernommenen Emissionsparametern der Ausgangsmodelle neigt das Garbage-Modell dazu, bei den meisten Sequenzen aufgrund seiner hohen Produktionswahrscheinlichkeit als zugehörige Klasse bestimmt zu werden. Eine Ursache dafür könnte in dem hier verwendeten sehr kleinen Gestensatz liegen. Bezieht man sich auf die Bildungsvorschrift, so hängen die Übergangswahrscheinlichkeiten von der Gesamtanzahl der Zustände aller Modelle ab. Bei einer größeren Anzahl verwendeter HMMs sinkt demzufolge die Wahrscheinlichkeit eines Zustandsüberganges innerhalb des Garbage-Modelles. Dabei sei noch erwähnt das normalerweise eine Zustandsreduzierung stattfindet, bei der Zustände mit gleicher oder sehr ähnlicher Emissionsgenerierung zusammengefasst werden. Nach [Elm10] liegt der Vorteil lediglich in der Reduzierung des Berechnungsaufwandes, sodass dieser Schritt hier ausgelassen wird. Die nächsten vier Tabellen zeigen jeweils ein Trainings- und Erkennungsprozess für die Faktoren: $c = 20$, $c = 10$, $c = 5$ und $c = 1$. Letzteres bedeutet, dass hier die Zustände der Ausgangsmodelle 1:1 übernommen werden. So zeigt Tabelle 6.9,

dass hier fast alle Sequenzen vom Garbage-Modell „aufgefangen“ werden. $c = 10$ stellt einen Grenzwert für die hier gegebenen Bedingungen dar. Wird c noch kleiner, sinkt auch die Erkennungsrate zunehmend.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	50	0	150	0	100	100 ± 0.061391
λ_K	50	0	150	0	100	100 ± 0.061391
λ_{Kreis}	50	0	150	0	100	100 ± 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 ± 0.061391
$\lambda_{Garbage}$	0	0	200	0	NaN	NaN ± 0

Tabelle 6.6: Analyse Garbage-Model - CHMM, VR-Testdaten, Faktor $c = 20$

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	48	0	150	2	100	96 ± 0.06055
λ_K	50	0	150	0	100	100 ± 0.061391
λ_{Kreis}	50	0	150	0	100	100 ± 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 ± 0.061391
$\lambda_{Garbage}$	0	2	198	0	0	NaN ± 0

Tabelle 6.7: Analyse Garbage-Model - CHMM, VR-Testdaten, Faktor $c = 10$

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	40	0	150	10	100	80 ± 0.05671
λ_K	50	0	150	0	100	100 ± 0.061391
λ_{Kreis}	50	0	150	0	100	100 ± 0.061391
$\lambda_{Dreieck}$	50	0	150	0	100	100 ± 0.061391
$\lambda_{Garbage}$	0	10	190	0	0	NaN ± 0

Tabelle 6.8: Analyse Garbage-Model - CHMM, VR-Testdaten, Faktor $c = 5$

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	0	0	150	50	NaN	0 ± 0
λ_K	0	0	150	50	NaN	0 ± 0
λ_{Kreis}	39	0	150	11	100	78 ± 0.056172
$\lambda_{Dreieck}$	0	0	150	50	NaN	0 ± 0
$\lambda_{Garbage}$	0	161	39	0	0	NaN ± 0

Tabelle 6.9: Analyse Garbage-Model - CHMM, VR-Testdaten, Faktor $c = 1$

In der nachstehenden Tabelle sind die Ergebnisse für den Erkennungsprozess für die aufgezeichneten Bewegungen der Probanden zu finden. Daraus lässt sich erkennen, dass alle

bis auf zwei Sequenzen richtig erkannt werden. Durch Änderung des Faktors c lässt sich das Ergebnis weiter verbessern.

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	37	0	132	7	100	84.0909 ± 0.061604
λ_K	43	0	132	1	100	97.7273 ± 0.064962
λ_{Kreis}	42	0	132	2	100	95.4545 ± 0.064443
$\lambda_{Dreieck}$	39	0	132	5	100	88.6364 ± 0.06279
$\lambda_{Garbage}$	0	15	161	0	0	NaN ± 0

Tabelle 6.10: Analyse Probanden - CHMM, Gesten 1 - 4, Garbage-Modell

Eine ähnliche Struktur wie Garbage-Modelle liefern Grammatik-freie Modelle, bei denen die Emissionen ebenfalls von den Ausgangsmodellen übernommen werden. Allerdings werden hier alle Zustandsübergänge über die Anzahl der vorliegenden Zustände gleich verteilt. Diese Modell kann als Konfidenz-Modell eingesetzt werden.Im Hinblick auf den noch folgenden Threshold-Modell-Ansatz wird hier jedoch der Ansatz eines Garbage-Modelles weiter verfolgt.

6.3.1 Threshold-Modell-Ansatz

Die Bedeutung eines Garbage-Modelles und deren Einfluss auf den Erkennungsprozess soll noch einmal in einem abschließenden Szenario demonstriert werden. Dafür wurde der „Threshold-Modell-Ansatz“ aus [LeK99] implementiert. Von „Simulation“ wird deshalb gesprochen, weil die Trajektorien bereits aufgezeichnet wurden und der Erkennungsprozess deshalb „offline“ stattfindet. Eine der wichtigsten Aufgaben der Online-Erkennung ist die Detektion von Start- und Endpunkten einer Geste in einem kontinuierlichen Eingabedatenstroms. Dieser wird hier simuliert, indem mehrere Trajektorien miteinander verkettet werden und so als Ergebnis einen einzelnen Datensatz liefern, der mehrere gesuchte Symbole enthält. Zu erwarten ist, dass insofern die Trajektorie keiner vorhanden Modellstruktur ähnelt, eine Zuordnung zu dem Garbage-Modell erfolgt.
Die Methode, die diesem Verfahren zu Grunde liegt, wird in der Literatur als „sliding window“ genannt. Zu Beginn funktioniert diese Technik wie ein Puffer. Entsprechend einer definierten Anzahl werden die anliegenden Emissionen gesammelt. Anschließend wird ein Klassifikationsprozess ausgelöst, wie er bisher verwendet wurde. Fällt die Entscheidung aufgrund der höheren Wahrscheinlichkeit auf das Garbage-Modell, so wird der älteste im Puffer enthaltene Wert entfernt und die nächste neu eintreffende Emission mit berücksichtigt. Bildlich kann man sich das wie eine Vorwärtsbewegung des Fensters über die Observationsfolge vorstellen, wobei sich die Position pro Durchlauf um 1 erhöht. Dabei sind auch andere Werte wählbar. Diese Vorgehensweise wiederholt sich solange, bis die Klassifizierung anstelle des Garbage-Modelles ein anderes Modell aus dem Gestensatz zurück liefert. Ab diesem Zeitpunkt werden alle folgenden Observationen mit in den Puffer gelegt und nach jeder Hinzunahme die Wahrscheinlichkeit für die Modelle erneut berechnet und zwar so lange, bis das Garbage-Modell als passende Klasse ermittelt wird. Abbildung 6.3 verdeutlicht nochmal den Ablauf des Algorithmus.

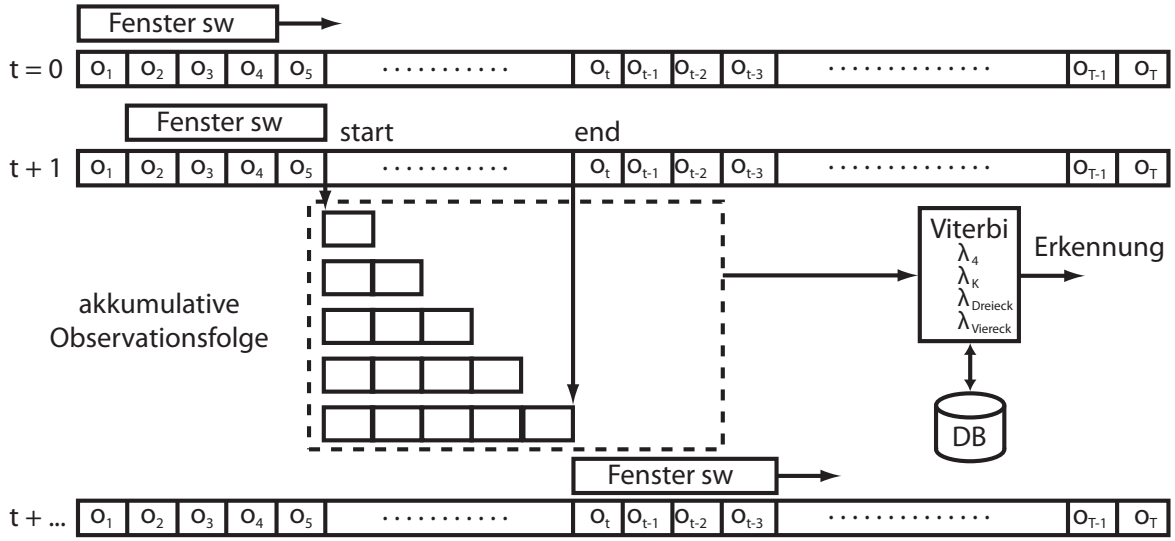


Abbildung 6.3: „sliding window“ [LeK99]

Der in [LeK99] beschriebene Algorithmus verwendet für die Berechnung der Wahrscheinlichkeiten bei einer durchschnittlichen Sequenzlänge von ca. 20 Emissionen eine Fenstergröße von 5. Überträgt man das vorgeschlagene Verhalten auf die hier vorliegenden Signaleigenschaften, so liegt die initiale SL-Größe bei ca. 20. Das SL sollte groß genug sein, um einen signifikanten Unterschied in der Wahrscheinlichkeitsberechung zu erreichen, wenn ein Übergang von nicht symboltragenden Gesten zu relevanten Emissionen erfolgt. Außerdem ist anzunehmen, dass, wenn nach dem Threshold-Modell-Ansatz gearbeitet wird, der erste und letzte Zustand der hier entwickelten Gesten-Modelle überflüssig sind, da diese ähnlich dem Garbage-Modell dazu dienen sollen, nicht klar definierte Bewegungen abzufangen, um so einen Übergang in die relevante Gestenbewegung zu schaffen. Diese Bereiche der Emissionsfolge würden nun durch das Garbage-Modell modelliert werden. Ein sich daraus zu erhoffendes Resultat ist die Erhöhung der Robustheit bei der Gestenerkennung.

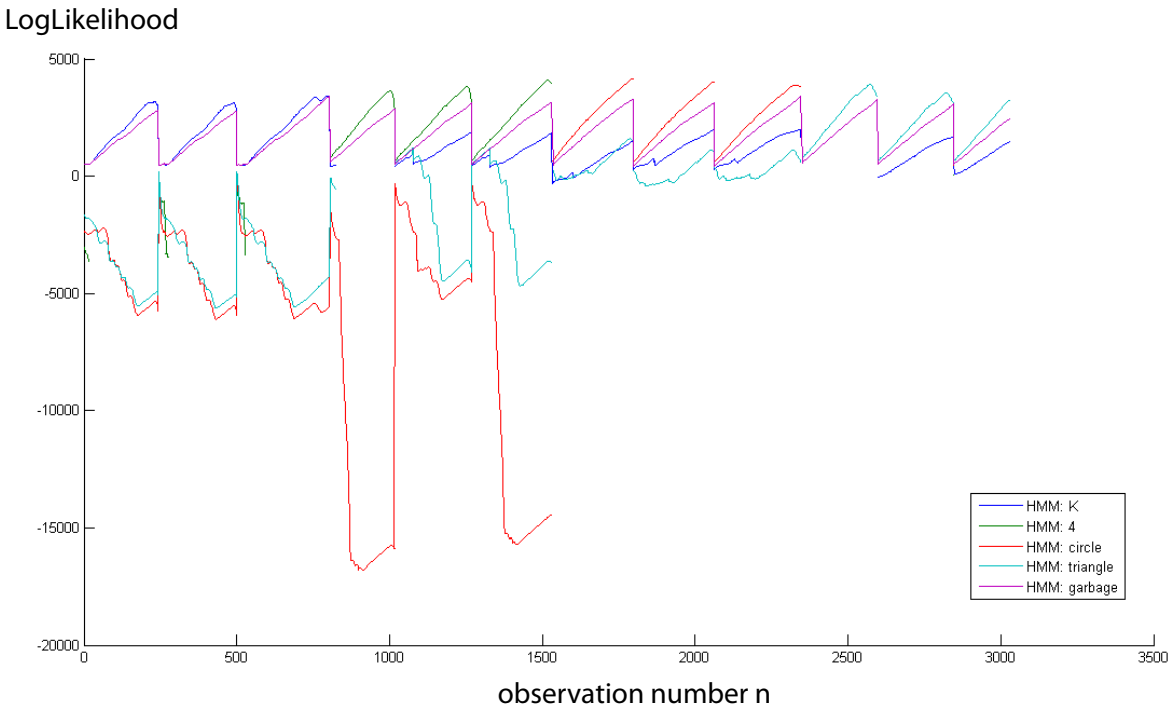


Abbildung 6.4: Threshold-Modell-Ansatz: für jedes Symbol sind 3 Trajektorien in dem Datensatz integriert

Abbildung 6.4 zeigt das Ergebnis der Anwendung des Threshhold-Modell-Ansatzes. Wie sich erkennen, lässt werden die Erwartungen an das Verfahren erfüllt. Anhand der lokalen Maxima lassen sich die unterschiedlichen Gesten in der Sequenz klar erkennen. Dabei sei zu erwähnen, dass es in den Übergängen der Wahrscheinlichkeitsübertretung verschiedener Modelle zu häufigen Wechseln des in der Wahrscheinlichket dominierenden Modelles kommen kann, da sich Garbage-Modell und Übergangszustände der Schlüsselmodelle in den Parameter sehr ähneln können.

7 Fazit

Neben einem abschließendem Fazit in dem nochmal die wesentlichen Erkenntnisse aus dieser Arbeit herausgegriffen werden, sind in diesem Kapitel Anknüpfungspunkte und Vorschläge für weitere zu untersuchende Aspekte bezüglich der Gestenerkennung mit HMMs zu finden.

7.1 Ergebnisse

Im Rahmen dieser Arbeit konnte gezeigt, dass eine Gestenerkennung mit Hidden-Markov-Modellen auf Basis der Merkmale Orientierung, Distanz und Geschwindigkeit in dem hier begrenzten Rahmen zuverlässige und überzeugende Erkennungsraten für hochauflösende Systeme liefert. Dabei hat sich die Herangehensweise, mit der Synthese zu beginnen, als sehr hilfreich erwiesen. Es konnte festgestellt werden, dass die mit dem Trackingsystem aufgezeichneten Daten sich in ihrer Charakteristik deutlich von den synthetisch generierten Daten unterscheiden, was letztendlich die Hinzunahme weiterer Merkmale und Modellzustände zwingend notwendig machte. In Abhängigkeit der Varianz der ausgeführten Gesten ist eine Merkmalsreduktion möglich. Der Vorteil der Geschwindigkeit liegt darin, dass das Merkmal auch bei geänderter Abtastrate benutzt werden kann, wenn es einmal auf einen bestimmte Datensatz trainiert wurde. Dabei konnte erkannt werden, dass es sich bei den zusätzlich Übergangszuständen um stark Struktur-tragende Elemente enthält. Eine weitere Erkenntnis konnte im Bezug auf die Start- und Endpunkte von Gesten gewonnen werden. Dabei hat sich gezeigt, das es selbst bei einem „offline“ Verfahren sehr schwer sein kann, eine Geste zu erkennen, wenn zu Beginn am Ende kurze Abschnitte beliebiger Bewegungen enthalten sind. Eine detailierte und hinreichende Visualisierung während des Analyseprozesses der zugrunde liegenden Sachverhalte ist unverzichtbar für eine genaue Reflektion der einzelnen Prozesse.

7.2 Ausblick

Da der Prozess der Gestenerkennung mit Hidden-Markov-Modellen diverse Teilprozesse und auch verschiedene Problemstellungen umfasst, werden an dieser Stelle mögliche Anknüpfungspunkte an diese Arbeit dargelegt, die sich aus der vorangegangenen Reflexion der Ergebnisse ergeben haben. Dabei werden Aspekte betrachtet, bei denen es sinnvoll sein kann, aufgrund der hier erlangten Kenntnisse, diese genauer zu untersuchen. Als Beispiel sei an dieser Stelle der Threshold-Modell-Ansatz genannt, da er in dem hier vorgelegten Rahmen nur am Rande betrachtet werden konnte. Die dabei gewonnen Einblicke haben jedoch gezeigt, dass das Verfahren durchaus Potenzial für einen „online“ -Erkennungsprozess hat und es sich daher lohnen würde, diesbezüglich weitere Untersuchungen anzustellen. In Analogie zu den Prozessen der Gestenerkennung sind die im Folgenden genannten Punkte so geordnet, dass sie thematisch bei der Datenaufnahme beginnen und im Prozess der Klassifizierung enden.

7.3 Performanz

Berechnungen von Wahrscheinlichkeiten bei einer großen Anzahl von Modellen mit kontinuierlicher Emissionsmodellierung und einer hohen Anzahl von Zuständen pro Modell können im Bezug auf die Echtzeitfähigkeit eines Systems sehr viele Ressourcen verbrauchen unter Umständen Latenzen verursachen. Für eine gegebene Folge von Observationen muss während der Klassifizierung die Produktionswahrscheinlichkeit für alle Modelle berechnet werden. Dieser Vorgang könnte zum Beispiel für jedes Model parallel ablaufen. Interessant wäre an dieser Stelle eine Untersuchung aller für die Erkennung mit HMMs nötigen Berechnungsschritte, um feststellen zu können, welche Operationen sich aus Systemtechnischer Sicht zusammenfassen bzw. parallelisieren lassen.

7.4 Koordinatenbestimmung aus Nutzerposition

Die Ergebnisse der Analyse und des Evaluationsprozesses haben gezeigt, dass der in dieser Arbeit gewählte Ansatz der Merkmalsextraktion im zweidimensionalen Raum sehr gut funktioniert und unabhängig von der aktuellen Position des Nutzers ist. Im dreidimensionalen Raum hingegen lässt sich eine Positionsunabhängigkeit nicht ohne Weiteres erreichen. Aufgrund der zusätzlichen Raumachse ist das Merkmal der Orientierung sehr anfällig für Positionsänderungen des Anwenders. Das heißt, selbst wenn eine Geste in der exakt gleichen Ausführung zweimal hintereinander von derselben Position vollzogen wird, kann sie nicht erkannt werden, wenn sich der Nutzer z.B. um 45 Grad dreht. Ein Ansatz um das Problem zu lösen, ist das „plane fitting“ , bei dem eine Ebene gesucht wird, welche die ausgeführte Geste in den zweidimensionalen Raum projiziert. Neben dem erhöhten Berechnungsaufwand kann es hier weitere Probleme geben, da die Zuordnung der Achsen nicht aus den Hauptkomponenten einfach entnommen werden kann. Ein möglicher Ansatz zur Lösung des Problems könnte eine individuelle Anpassung des Systems an die Nutzerpostion sein. Dabei bestimmt das Trackingsystem sein Koordinatensystem in Abhängigkeit der Ausrichtung des Anwenders und richtet den Koordinatenursprung auf dessen Position aus.

7.4.1 Modelltopologie und Parameter

Ein entscheidender Faktor für den Erfolg des Erkennungsprozesses bilden die gewählten Startparameter und Modellstrukturen. Auch da liegen verschiedene Ansätze vor, wie sich diese optimal bestimmen lassen könnten. Ein standardisiertes Verfahren gibt dafür noch nicht. In dieser Arbeit konnte gezeigt werden, wie das Hinzufügen oder Entfernen von Modellzuständen, die Erkennung stark beeinflussen kann. Der Ansatz, die Untersuchungen mit der Synthese zu beginnen, hat sich letztendlich bewährt, da zu Beginn festgestellt werden konnte, dass die Modelle dazu in der Lage sind, die Gesten zu emittieren. So konnten im Verlauf der Ausarbeitung Probleme die bei der Erkennung realer Daten aufgetreten sind, leicht adressiert und gelöst werden und Parameter besser an die Anforderungen angepasst werden. Anhand der Merkmale Distanz und Geschwindigkeit konnte allein schon an der grafischen Darstellung erkannt werden, wie viele Modellzustände für jede Geste in etwa benötigt werden, um diese optimal zu beschreiben. Als Anschlussthema biete es sich daher an, detaillierter auf die Zusammenhänge von gewählter Merkmalsrepräsentation, Modellstruktur und initial bestimmte Parameter zu schauen.

7.4.2 Threshold-Modell-Ansatz

Wie in Kapitel 6.4 gezeigt wurde, kann mit Hilfe des Garbage-Modelles großen Einfluss auf die Erkennung genommen werden. Eine genaue Untersuchung bezüglich der Struktur und Wahrscheinlichkeitsverteilung könnte durchaus die Zuverlässigkeit der einzelnen Modelle erhöhen und eine bessere Trennung zwischen relevanten Gesten und nicht symboltragenden Trajektorien erreichen. Darüber hinaus scheint die Verwendung des Garbage-Modelles als Schwellwert für die Start- und Endpunktdetektion durchaus Potenzial zu haben. Der Threshold-Modell-Ansatz, welcher im Prinzip als ein Garbage-Modell-gesteuerter Ansatz bezeichnet werden kann und aufgrund des begrenzten zeitlichen und inhaltlichen Rahmens dieser Arbeit nur beiläufig betrachte werden konnte, hat in ersten Eindrücken gezeigt, wie effektiv die Kombination aus Nicht-Gesten und Gestenmodellen sein kann. Daher empfiehlt es sich hier, eine genauere Analyse zu betreiben, um dessen Verhalten zu erfassen und eine Beurteilung hinsichtlich seiner Zuverlässigkeit vornehmen zu können.

7.4.3 Hierarchische Modellstrukturen

Die Analyse der hier verwendeten Gesten hat gezeigt, dass ein einziges Modell meist nicht ausreicht um alle verschiedenen Ausführungen bezüglich eines Symboles zu erfassen. Eine Frage die sich dabei ergibt, ist die nach einer geeigneten Struktur bzw. Hierarchie, bei der alle zu einem Symbol gehörenden Modelle zusammengefasst werden können. Des Weiteren hat der Threshold-Model-Ansatz gezeigt, das es bei dem Übergang zwischen zwei Gesten, die in einer Beobachtungsfolge enthalten sind, zu vielen kurzen Wechseln zwischen dem aktuell erkannten Modell kommen kann, da hier Garbage-Modell und Gestemodelle ähnliche Produktionswahrscheinlichkeiten aufweisen. Das führt dazu, dass das Modell mit der höchsten Wahrscheinlichkeit in kurzen Zeitintervallen wechseln kann. Eine übergeordnete Struktur könnte an dieser Stelle eingreifen, und beispielsweise anhand der Anzahl der seit der letzten Klassifikation eintreffenden Observationen feststellen, ob diese praktisch ausreicht um eine Geste komplett ausführen zu können. So würde sich hier die Möglichkeit einer Validierung ergeben. Eine Untersuchung dieser Effekte könnte darüber Aufschluss geben.

7.4.4 Übergangszustände

Der Analyseprozess hat klar gezeigt, wie wichtig die Rolle der Übergangszustände innerhalb der Schlüsselmodelle ist und das sich selbst und unter Ausschluss der eigentlichen Gestensegmente noch relativ hohe Erkennungsraten erreichen lassen. Ein Erschließen der Bedeutung von Struktur enthaltene Elementen innerhalb einer bereits vorhanden Modellstruktur kann offenbar großen positiven Einfluss auf die Systemleistung nehmen. Daher empfiehlt es sich das Thema weiter zu untersuchen.

8 Literatur

[AnH08] Heinrich, A.: Information Retrieval 1. Otto-Friedrich-Universität Bamberg, 2008.

[BHK97] Barret, C.; Hughey, R.; Karplus, K.: Scoring hidden markov models. University of California, 1997.

[BOP96] Brand, M.; Oliver, N.; Pentland, A.: Coupled hidden markov models for complex action recognition. MIT Media Lab, 1996.

[CeS95] Cédras, C.; Shah, M.: Motion-based recognition: a survey, in: Image and Vision Computing Volume 13, Elsevier Science B.V. (Hrsg.), 1995.

[CRR11] Chaudhary, A. et al.: Intelligent approaches to interact with machines using hand gesture recognition in natural way: a survey, in: International Journal of Computer Science & Engineering Survey (IJCSES), Vol. 2, No. 1, 2011.

[CTY12] Caon, M. et al.: Extending the interaction area for view-invariant 3D gesture recognition. University of Bedfordshire, University of Applied Science of Western Switzerland, 2012.

[EHA08] Elmezain, M. et al.: A hidden markov model-based continuous gesture recognition system for hand motion trajectory. Otto-von-Guericke-Universitaet Magdeburg, 2008.

[EHM08] Elmezain, M.; Al-Hamadi, A., Michaelis, B.: Real-time capable system for hand gesture recognition using hidden markov models in stereo color image Sequences. Otto-von-Guericke-Universitaet Magdeburg, 2008.

[Elm10] Elmezain, M.O.S.M.: Dissertation, Hand gesture spotting and recognition using HMMs and CRFs in color image sequences. Otto-von-Guericke-Universitaet Magdeburg, 2010.

[Fin03] Fink, G.A.: Mustererkennung mit Markov-Modellen. B.G.Teubner Verlag, 2003.

[Ges14] Geßler, P.: Stimmenauthentifizierung. BTU Cottbus-Senftenberg, 2014.

[ISc04] Schmitt, I.: Dissertation, Multimedia-Datenbanken: Retrieval, Suchalgorithmen und Anfragebearbeitung. Otto-von-Guericke-Universitaet Magdeburg, 2004.

[JuJ09] Jurafsky, D.; Martin, J.M.: Speech and language processing. Pearson Education, 2009.

[KhI12] Khan, Z.R.; Ibraheem, N.A.: Survey on gesture recognition for hand image postures. Aligarh Muslim University, 2012.

[LeK99] Lee, H.-K.; Kim, J.H.: An HMM-based threshold model approach for gesture recognition, in: IEEE Transactions on pattern analysis and machine intelligence, VOL. 21, NO. 10, 1999.

[LLK03] Liu, N.; Lovell, B. C.; Kootsookos, P. J.: Evaluation of HMM training algorithms for letter hand gesture recognition. University of Queensland, Brisbane, Australia 4072, 2003.

[LLK04] Liu, N. et al.: Model structure selection & training algorithms for a HMM gesture recognition system. University of Queensland, Brisbane, Australia 4072, 2004.

[MaW14] MathWorks: Statistics Toolbox. MathWorks, URL: http://www.mathworks.de/de/help/stats/index.html, 2014.

[Mur05] Murphy, K.: Hidden Markov Model (HMM) Toolbox for Matlab. Massachusetts Institute of Technology, URL: http://www.cs.ubc.ca/ murphyk/Software/HMM/hmm.html, 2005

[Mur14] Murphy, K. et al.: Probabilistic modeling toolkit for Matlab/ Octave (PMTK3). Massachusetts Institute of Technology, URL: https://github.com/probml/pmtk3, 2014

[NaW96] Nam, Y.; Wohn, Y. K.: Recognition of space-time hand-gestures using hidden markov model. Korea Advanced Institute of Science and Technologies,1996.

[PSH97] Pavlovic, I. V.; Sharma, R.; Huang T. S.: Visual interpretation of hand gestures for human-computer interaction: a review, in: IEEE Transactions on pattern analysis and machine intelligence, VOL. 19, NO. 7, 1997.

[Rab89] Rabiner, L.R.: A tutorial on Hidden Markov Models an selected applications in speech recognition, in: IEEE, VOL. 77, NO. 2, 1989.

[RaQ08] Rajko, S.; Quian, G.: HMM parameter reduction for practical gesture recognition. Arizona State University, Tempe, AZ, 2008.

[Ric11] Richarz, J.: Dissertation, Videobasierte Gestenerkennung in einer intelligenten Umgebung. Technische Universitaet Dortmund, 2011.

[Roo14] Roolfs, G: Koordinatensysteme. Niedersächsisches Landesinstitut für schulische Qualitätsentwicklung, URL: http://nibis.ni.schule.de/ lbs-gym/Verschiedenespdf/Koordinatensysteme.pdf, 2014.

[SDD12] Song, Y.; Demirdjian, D.; Davis, R.: Continuous body and hand gesture recognition for natural human-computer interaction. Massachusetts Institute of Technology, 2012.

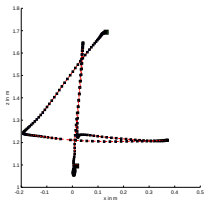
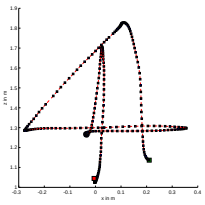
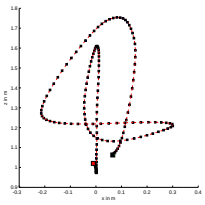
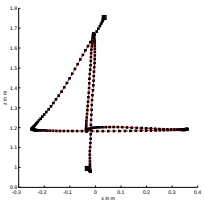
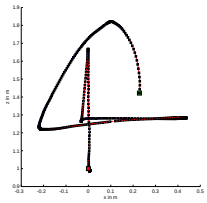
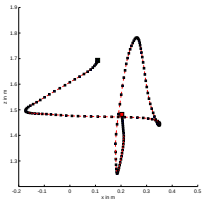
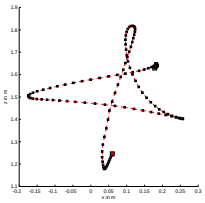
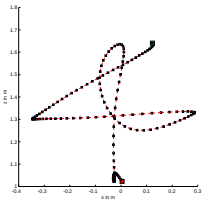
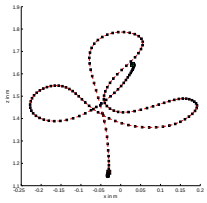
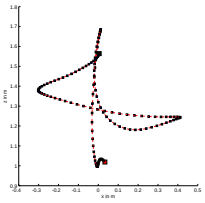
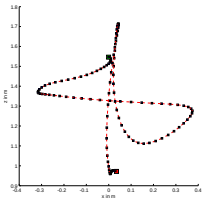
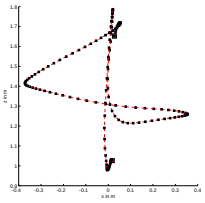
[UZI14] o.V.; XML Extensible Markup Language, URL: http://www.uzi-web.de/, 2014.

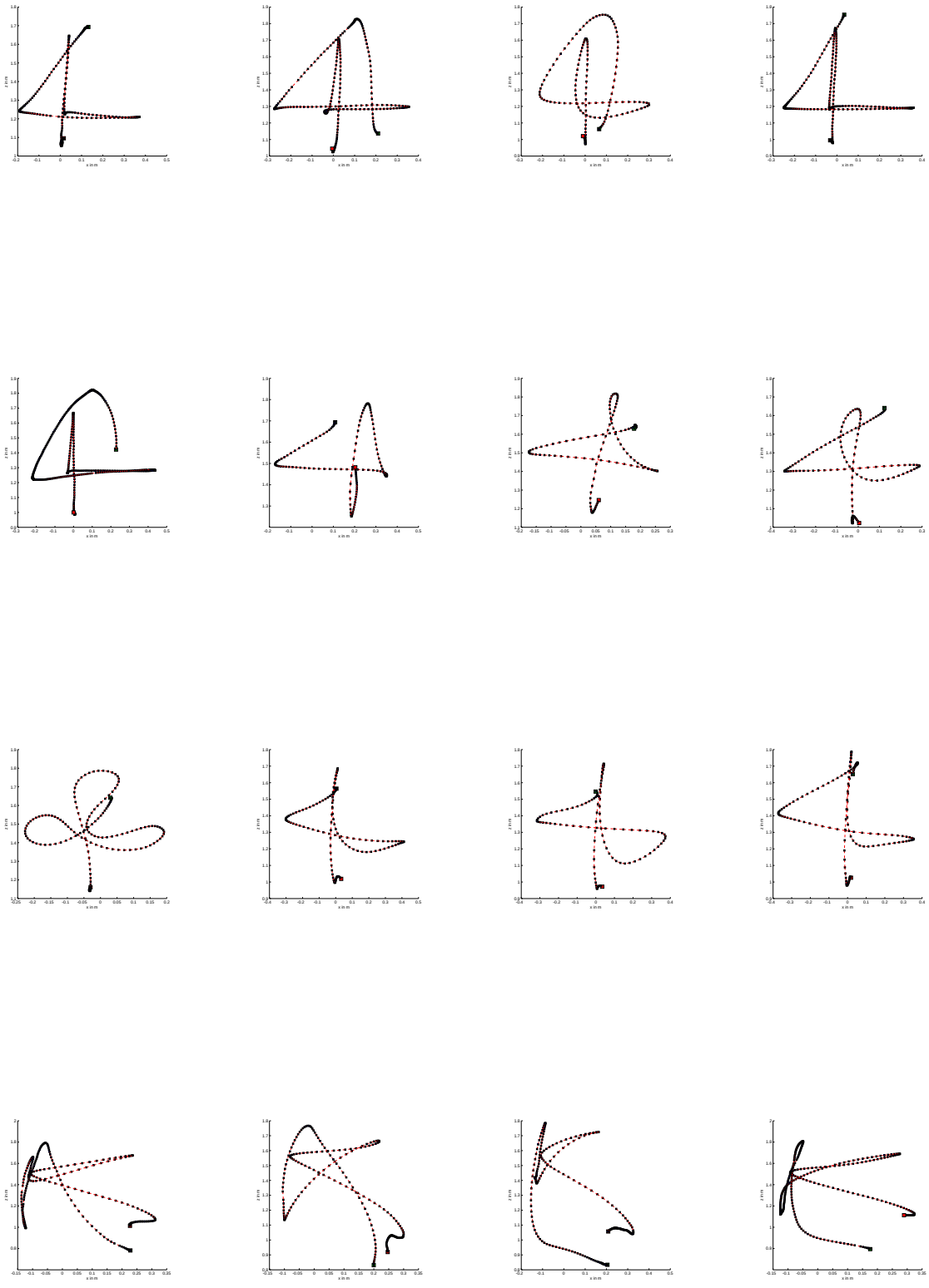
[Vio99] La Viola Jr., J. J.: A survey of hand posture and gesture recognition techniques and technology. Brown University Providence, Rhode Island 02912, 1999.

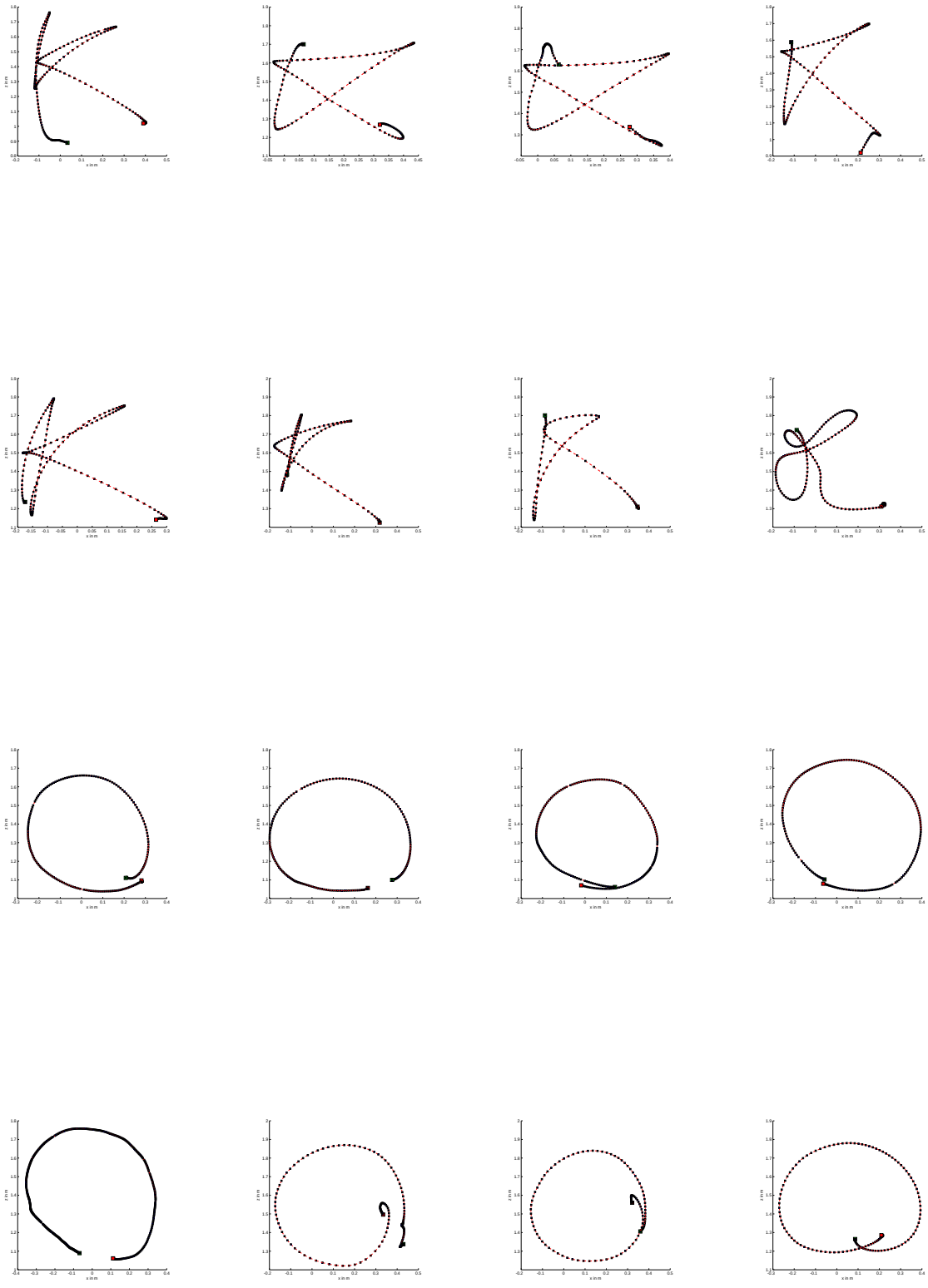
[YSB01] Yoon, H.-S. et al.: Hand gesture recognition using combined features of location, angle and velocity, in: Elsevier Science Ltd. (Hrsg.), Pattern Recognition 34, S. 1491-1501, 2001.

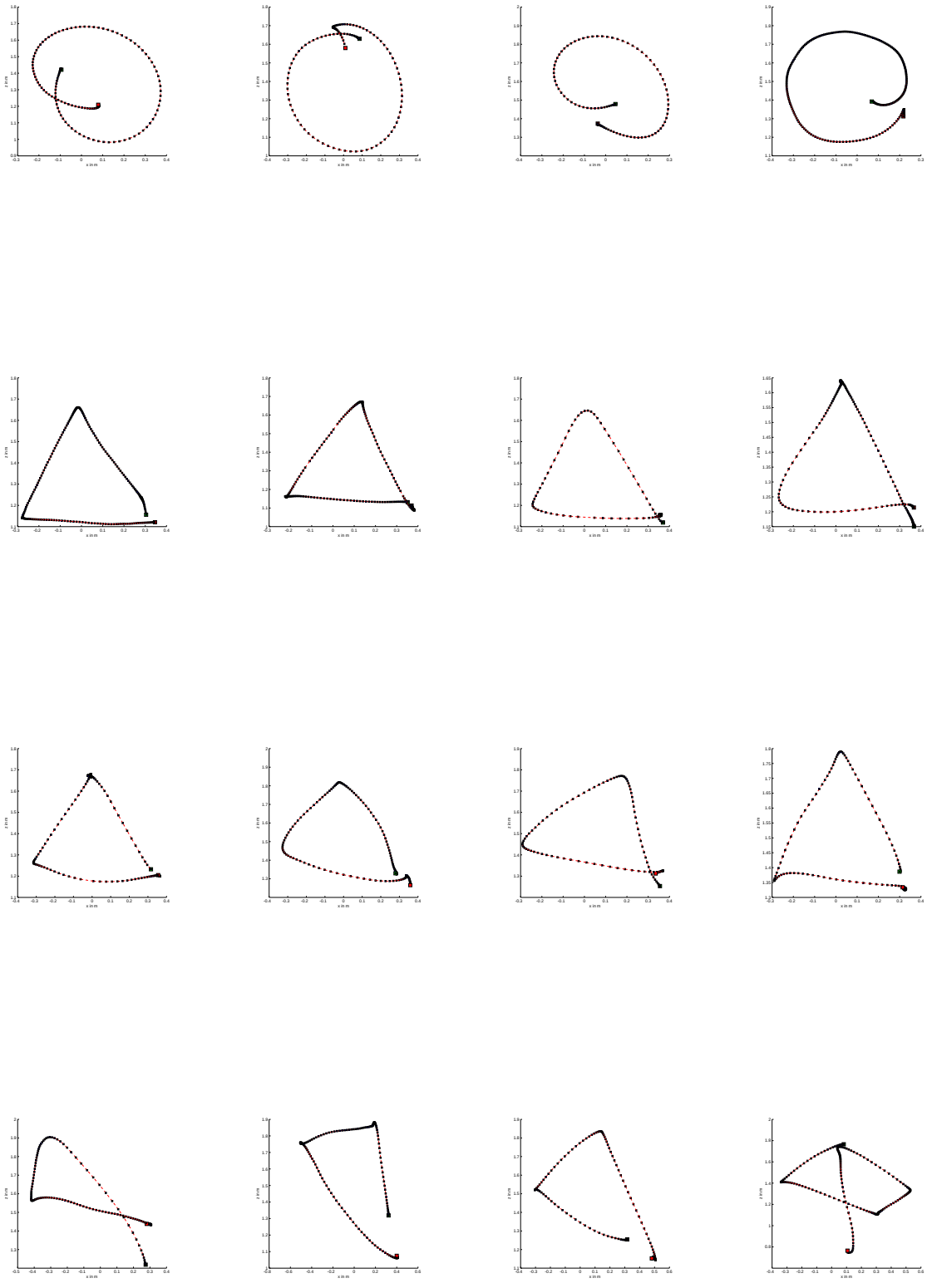
A Anhang

A.1 VR-Labor: Testdaten

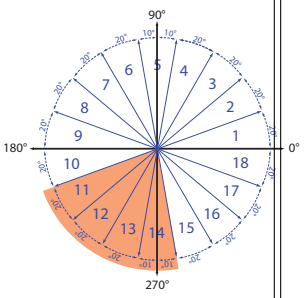
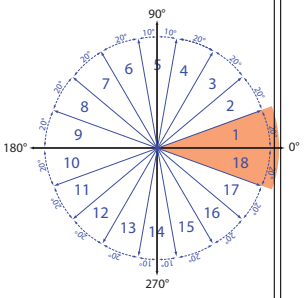
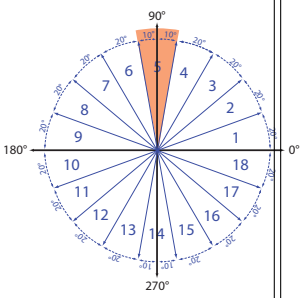
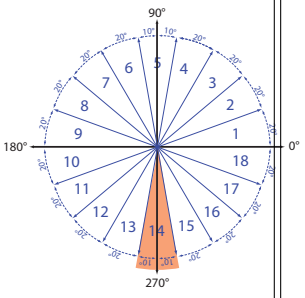
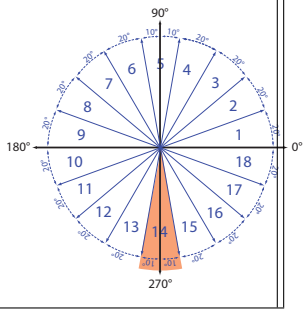













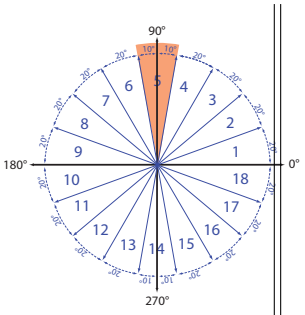
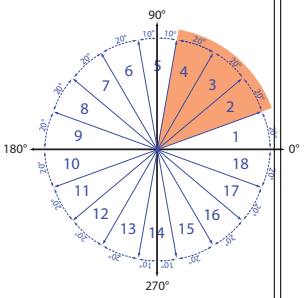
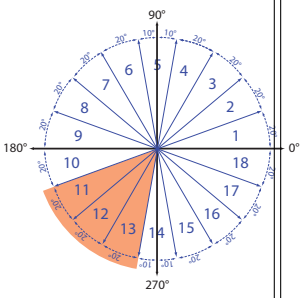
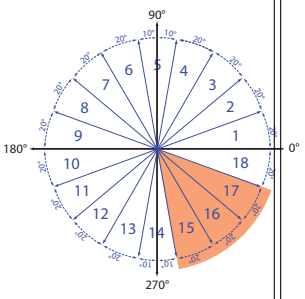
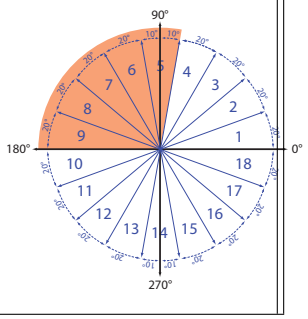




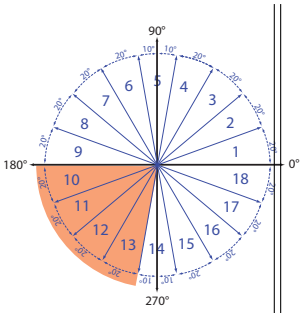
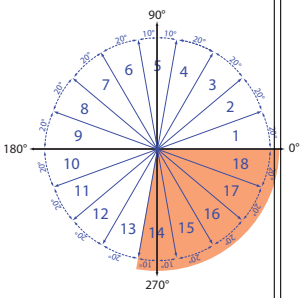
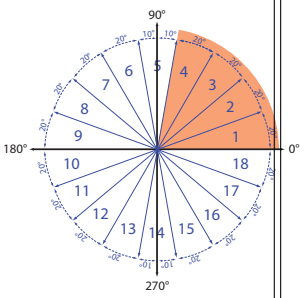
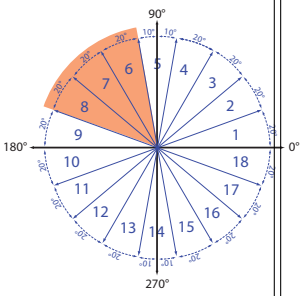
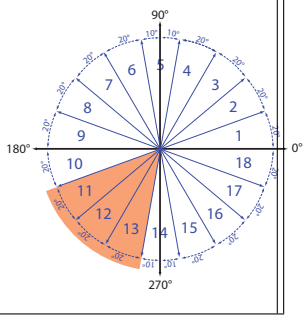


A.2 DHMM: Observationswahrscheinlichkeiten

Symbol	Zustand	Emission v	P(v) in %	Grafik
4	1	11, 12, 13, 14	25	
	2	1, 18	50	
	3	5	100	
	4	14	100	
	1	14	100	

	3	5	100	
	4	2, 3, 4	33.33	
	5	11, 12, 13	33.33	
	6	15, 16, 17	33.33	
	1	5, 6, 7, 8, 9	20	

Kreis

	2	10, 11, 12, 13	25	
	3	14, 15, 16, 17, 18	20	
	4	1, 2, 3, 4	25	
Dreieck	1	6, 7, 8	33.33	
	2	11, 12, 13	33.33	

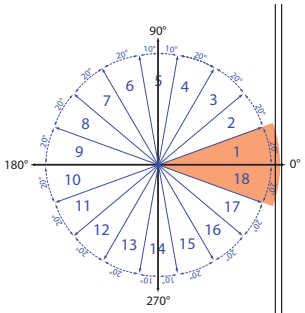
	3	1, 18	50	

Tabelle A.2: Emissionswahrscheinlichkeiten: DHMMs

A.3 Ergebnisse: Probanden-Analyse

Ergebnisse: Gestenerkennung, Modelle 1 - 4

Merkmal: xyOfSpherical, distance, velocity

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	44	0	132	0	100	100 ± 0.065465
λ_K	44	0	132	0	100	100 ± 0.065465
λ_{Kreis}	44	0	132	0	100	100 ± 0.065465
$\lambda_{Dreieck}$	44	0	132	0	100	100 ± 0.065465

Tabelle A.3: Analyse Probanden - CHMM, Gesten 1 - 4

Ergebnisse: Gestenerkennung, Modelle 1 - 4, mit Garbage-Modell

Merkmal: xyOfSpherical, distance, velocity

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	37	0	132	7	100	84.0909 ± 0.061604
λ_K	43	0	132	1	100	97.7273 ± 0.064962
λ_{Kreis}	42	0	132	2	100	95.4545 ± 0.064443
$\lambda_{Dreieck}$	39	0	132	5	100	88.6364 ± 0.06279
$\lambda_{Garbage}$	0	15	161	0	0	NaN ± 0

Tabelle A.4: Analyse Probanden - CHMM, Gesten 1 - 4, Garbage-Modell

Ergebnisse: Gestenerkennung, Modelle 1 - 8

Merkmal: xyOfSpherical, distance, velocity

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4 **Var 1**	43	38	268	1	53.0864	97.7273 ± 0.035144
λ_4 **Var 2**	7	1	307	35	87.5	16.6667 ± 0.014988
λ_K **Var 1**	41	5	301	3	89.1304	93.1818 ± 0.034429
λ_K **Var 2**	43	3	303	1	93.4783	97.7273 ± 0.035144
λ_{Kreis} **Var 1**	44	2	304	0	95.6522	100 ± 0.035492
λ_{Kreis} **Var 2**	37	0	306	7	100	84.0909 ± 0.032917
$\lambda_{Dreieck}$ **Var 1**	44	0	306	0	100	100 ± 0.035492
$\lambda_{Dreieck}$ **Var 2**	42	0	306	2	100	95.4545 ± 0.03479

Tabelle A.5: Analyse Probanden - CHMM, Gesten 1 - 8

Modell	wird erkannt als (in %)							
	λ_4	λ_4 (2)	λ_K	λ_K (2)	λ_{Kr}	λ_{Kr} (2)	λ_{Dr}	λ_{Dr} (2)
λ_4	97.7273	2.2727	0	0	0	0	0	0
λ_4 (Var2)	83.3333	16.6667	0	0	0	0	0	0
λ_K	0	0	93.1818	6.8182	0	0	0	0
λ_K (Var2)	0	0	2.2727	97.7273	0	0	0	0
λ_{Kreis}	0	0	0	0	100	0	0	0
λ_{Kreis} (Var2)	6.8182	0	9.0909	0	0	84.0909	0	0
$\lambda_{Dreieck}$	0	0	0	0	0	0	100	0
$\lambda_{Dreieck}$ (Var2)	0	0	0	0	4.5455	0	0	95.4545

Tabelle A.6: Analyse Probanden - CHMM, Gesten 1 - 8

Ergebnisse: Gestenerkennung, Modelle 1 - 8, modellspezifische Testdaten + Daten der frei wählbare Ausführungsart

Merkmal: xyOfSpherical, distance, velocity

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4 Var 1	41	71	411	3	36.6071	93.1818 $\pm$ 0.023401
λ_4 Var 2	15	24	416	71	38.4615	17.4419 $\pm$ 0.014528
λ_K Var 1	35	11	471	9	76.087	79.5455 $\pm$ 0.021754
λ_K Var 2	78	26	412	10	75	88.6364 $\pm$ 0.031021
λ_{Kreis} Var 1	43	15	467	1	74.1379	97.7273 $\pm$ 0.023915
λ_{Kreis} Var 2	56	1	437	32	98.2456	63.6364 $\pm$ 0.026922
$\lambda_{Dreieck}$ Var 1	37	6	476	7	86.0465	84.0909 $\pm$ 0.022321
$\lambda_{Dreieck}$ Var 2	64	3	435	24	95.5224	72.7273 $\pm$ 0.028535

Tabelle A.7: Analyse Probanden - CHMM, Gesten 1 - 8, strukturfreier Datensatz

Modell	wird erkannt als (in %)							
	λ_4	λ_4 (2)	λ_K	λ_K (2)	λ_{Kr}	λ_{Kr} (2)	λ_{Dr}	λ_{Dr} (2)
λ_4	93.1818	6.8182	0	0	0	0	0	0
λ_4 (Var2)	77.907	17.4419	1.1628	2.3256	0	0	1.1628	0
λ_K	0	0	79.5455	20.4545	0	0	0	0
λ_K (Var2)	0	0	9.0909	88.6364	1.1364	0	0	1.1364
λ_{Kreis}	0	0	0	0	97.7273	2.2727	0	0
λ_{Kreis} (Var2)	2.2727	3.4091	1.1364	13.6364	13.6364	63.6364	0	2.2727
$\lambda_{Dreieck}$	0	15.9091	0	0	0	0	84.0909	0
$\lambda_{Dreieck}$ (Var2)	2.2727	12.5	1.1364	3.4091	2.2727	0	5.6818	72.7273

Tabelle A.8: Analyse Probanden - CHMM, Gesten 1 - 8, strukturfreier datensatz

Ergebnisse: Gestenerkennung, Modelle 1 - 4,

Merkmal: xyOfSpherical, velocity

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	44	2	130	0	95.6522	100 $\pm$ 0.065465
λ_K	44	0	132	0	100	100 $\pm$ 0.065465
λ_{Kreis}	44	0	132	0	100	100 $\pm$ 0.065465
$\lambda_{Dreieck}$	42	0	132	2	100	95.4545 $\pm$ 0.064443

Tabelle A.9: Analyse Probanden - CHMM, Gesten 1 - 4, Merkmal - Winkel und Geschwindigkeit

Modell	wird erkannt als (in %)			
	λ_4	λ_K	λ_{Kreis}	$\lambda_{Dreieck}$
λ_4	100	0	0	0
λ_K	0	100	0	0
λ_{Kreis}	0	0	100	0
$\lambda_{Dreieck}$	4.5455	0	0	95.4545

Tabelle A.10: Analyse Probanden - CHMM, Gesten 1 - 4, Merkmal - Winkel und Geschwindigkeit

Ergebnisse: Gestenerkennung, Modelle 1 - 4,

Merkmal: xyOfSpherical, distance

HMM	ca	fa	cd	fd	precision in %	Erkennungsrate (recall) in %
λ_4	44	0	132	0	100	100 $\pm$ 0.065465
λ_K	44	0	132	0	100	100 $\pm$ 0.065465
λ_{Kreis}	43	0	132	1	100	97.7273 $\pm$ 0.064962
$\lambda_{Dreieck}$	44	1	131	0	97.7778	100 $\pm$ 0.065465

Tabelle A.11: Analyse Probanden - CHMM, Gesten 1 - 4, Winkel und Distanz

Modell	wird erkannt als (in %)			
	λ_4	λ_K	λ_{Kreis}	$\lambda_{Dreieck}$
λ_4	100	0	0	0
λ_K	0	100	0	0
λ_{Kreis}	0	0	97.7273	2.2727
$\lambda_{Dreieck}$	0	0	0	100

Tabelle A.12: Analyse Probanden - CHMM, Gesten 1 - 4, Merkmal - Winkel und Distanz

A.4 Matlab: Beispiele

Die Tabelle enthält die auf der CD befindlichen Beispiel-Scripte und liefert zu jedem eine kurze Beschreibung.

Beispiel	Beschreibung
Example 1	diskrete Synthese-Modelle mit dem Merkmal „Orientierung (quantisiert)“ zum Generieren von 150 Sequenzen, Struktur nach [Elm10]
Example 1.1	diskrete Synthese-Modelle mit dem Merkmal „angle (quantisiert)“ zum Generieren von 150 Sequenzen, Symbol K und 4 um doppelten Zustand reduziert
Example 2	kontinuierliche Synthese-Modelle mit dem Merkmal „xyOfPolarAngle)“ zum Generieren von 150 Sequenzen, Symbol K und 4 um doppelten Zustand reduziert
Example 2.1	Laden der von 2.1 synthetisch erzeugten Sequenzen mit dem Merkmal „xyOfPolarAngle)“
Example 3	Training und Erkennung mit diskreten initialen Modellen nach [Elm10] unter Nutzung der von 1 synthetisch erzeugten Sequenzen mit dem Merkmal „angle (quantisiert)“
Example 3.1	Training und Erkennung mit diskreten Synthese-Modellen unter Nutzung der von 1 synthetisch erzeugten Sequenzen mit dem Merkmal „angle (quantisiert))“
Example 3.2	Erkennung ohne Training mit diskreten Synthese-Modellen unter Nutzung der von 1 synthetisch erzeugten Sequenzen mit dem Merkmal „angle (quantisiert))“
Example 4	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der von 2 synthetisch erzeugten Sequenzen mit dem Merkmal „xyOfPolarAngle “
Example 4.1	Erkennung ohne Training mit kontinuierlichen Synthese-Modellen unter Nutzung der von 2 synthetisch erzeugten Sequenzen mit dem Merkmal „xyOfPolarAngle “
Example 4.2	Erkennung ohne Training mit kontinuierlichen Synthese-Modellen (ohne circle) unter Nutzung der von 2 synthetisch erzeugten Sequenzen mit dem Merkmal „xyOfPolarAngle “
Example 5	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der VR-Testdaten mit dem Merkmal „xyOfPolarAngle “
Example 5.1	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der gefilterten VR-Testdaten mit dem Merkmal „xyOfPolarAngle “
Example 5.2	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der gefilterten VR-Testdaten mit den Merkmalen „xyOfPolarAngle distance velocity “
Example 5.3	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der gefilterten VR-Testdaten (SNR-erweitert) mit den Merkmalen „xyOfPolarAngle distance velocity “
Example 6	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der VR-Testdaten mit den Merkmalen „xyOfSphericalAngle distance velocity “
Example 6.1	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der VR-Testdaten (SNR-erweitert) mit den Merkmalen „xyOfSphericalAngle distance velocity “
Example 6.2	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der gefilterten VR-Testdaten (SNR-erweitert) mit den Merkmalen „xyOfSphericalAngle distance velocity “
Example 7	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der gefilterten VR-Testdaten (SNR-erweitert) mit den Merkmalen „xyOfSphericalAngle distance velocity “ und der Verwendung eines Garbage-Modelles
Example 10 - 12	Training und Erkennung mit kontinuierlichen Synthese-Modellen unter Nutzung der von Probanden aufgezeichneten Daten. Diverse Parameter sind einstellbar: Merkmale, Garbage-Modell, State-Deleting, freie Datensatzwahl

Eidestattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Masterarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe. Mit meiner Unterschrift stimme ich einer Veröffentlichung der Arbeit oder Teilen aus der Arbeit zu.