Fachhochschule Köln
Cologne University of Applied Sciences

MASTERTHESIS

vorgelegt an der Fachhochschule Köln
Campus Gummersbach
im Studiengang Medieninformatik

Untersuchungen zur Varianzreduktion beschleunigungsbasierter 3D-Gestendaten

ausgearbeitet von
Daniel Bertram
xarfk@gmx.net

Gummersbach, im April 2012

erster Prüfer
Prof. Dr. Wolfgang Konen

zweiter Prüfer
Prof. Dr. Gerhard Hartmann

Zusammenfassung

Verschiedene Arbeiten haben sich bereits mit der Klassifikation von 3D-Gestendaten beschäftigt, wobei die Varianz aller verwendeten Verfahren zwei Gemeinsamkeiten besitzt. Keine der Arbeiten erzielt benutzerunabhängig ähnlich gute Ergebnisse wie im benutzerabhängigen Fall und keine Arbeit erzielt eine 100% Erkennung. Diese Arbeit untersuchte am Beispiel einer Gestenerkennung mittels *Slow Feature Analysis* (SFA), die auf einem iPhone umgesetzt wurde, welche Unterschiede zwischen benutzerabhängiger und benutzerunabhängiger Erkennung bestimmbar sind und wie sich diese in ihrer Varianz reduzieren lassen.

Für die Betrachtungen wurden zwei personendisjunkte Datensätze verwendet. Es wurden verschiedene Einflüsse bestimmt und durch Operatoren zur Invarianz überführt. Durch die bestimmten Operatoren ist es möglich, die Erkennungswahrscheinlichkeit im benutzerabhängigen und im benutzerunabhängigen Fall zu steigern. Es wurden drei Invarianzen erschaffen, die Rotationsinvarianz durch eine Datensatzrotation mittels Quaternionen, die Gestensegmentierung durch Ruheauslöschung sowie dem Ausklingen um Bewegungsabbrüche zu kompensieren.

Die SFA hat sich als robustes und zuverlässiges Verfahren erwiesen, welches durch seine Analyseeigenschaften sogar für rotierte Gesten korrekt klassifizierbare Eigenschaftsvektoren bestimmt. Durch die unterschiedlichen Methoden wurde die Erkennungswahrscheinlichkeit im benutzerunabhängigen Fall gesteigert und durch die Überführung von Einflussfaktoren zur Invarianz die Benutzbarkeit für einen „untrainierten“ Benutzer deutlich erhöht. Weiterhin wurde festgestellt, dass die Betrachtung der Datensätze im Bild- und Frequenzbereich zu unterschiedlichen Fehlerkennungen führen, diese somit unterschiedliche Informationen für die SFA besitzen.

Inhaltsverzeichnis

Abbildungsverzeichnis **5**

Tabellenverzeichnis **6**

1 Einleitung **8**
1.1 Motivation 8
1.2 Problemstellung und Ziele 8
1.3 Aufbau der Arbeit 9

2 Grundlagen **10**
2.1 Stand der Wissenschaft 10
2.2 Slow Feature Analysis (SFA) 13
2.3 Gauss-Klassifikator 16
2.4 Gesten 18
2.5 Geräterotation versus Gestenrotation 19
2.6 Test- und Trainingsdatenerhebung 20
2.7 Datensatzbetrachtung 22
2.7.1 Grobeinteilung des Trainingsdatensatzes 22
2.7.2 Feineinteilung des Trainingsdatensatzes 23
2.8 Konfusionsmatrix 25

3 Signalbetrachtungen **26**
3.1 Signaleigenschaften 26
3.1.1 Vertrautheit mit dem Gerät 26
3.1.2 Start- und Endhaltung des Gerätes 26
3.1.3 Gestenlänge 27
3.1.4 Start- und Endimpuls 28
3.2 g-Vektor 30
3.3 Spektrale Ähnlichkeit 32
3.4 Ausgleichsrechnungen 35
3.4.1 Translation zur Idealstartposition 35
3.4.2 Translation zu 0 37
3.4.3 Zufälliges Verrauschen 39

4 Systemeinfluss **41**
4.1 Trainingszustand 41
4.2 Klassifikation 42
4.2.1 Der „perfekte“ Datensatz 44
4.2.2 Unterabtastung in der Vorverarbeitung 47
4.3 Expansionsartefakte in $avg1$ 49
4.4 Gaussdimensionen 49

5 Benutzereinfluss **52**
5.1 Ruhebetrachtung 53
5.2 Automatische Segmenttrennung 53
5.3 Winkelkompensation 56
5.4 Winkelkompensation mittels Quaternion 57
5.5 Test des Quaternionausgleichs 62
5.6 Bewegungsabruch und Ausklingen 65
5.7 Methodenkombination 67
5.7.1 Filterbetrachtung 68
5.7.2 Operatorbetrachtung 69
5.7.3 Operatoreinfluss 70

6 Fazit **72**

Abbildungsverzeichnis

1 Zwei Gaussklassen **1** und **2** nach den beiden Eigenschaften **A** und **B** . . . 17
2 Die sechs Gesten, die im Trainingsset und Testset aufgenommen wurden, der Kreis, das X, das Z, das Epsilon, das Herz und das Unendlichkeitszeichen. . . . 19
3 Erkennungswahrscheinlichkeiten für den Testdatensatz, getrennt nach positiven und negativen Datensätzen . . . 24
4 Beträge der Start- und Endbeschleunigungbeschleunigung, sortiert über alle Datensätze im Trainingsdatensatz . . . 29
5 Achsenausrichtung des iPhones . . . 30
6 Start- und Endbeschleunigungen der Kreisgeste . . . 31
7 Betrag der Beschleunigung des ersten und und aller Elemente des Trainingsdatensatzes . . . 32
8 Spektrum von drei Kreis- und drei X-Gesten, jeweils die dritte ist negativ 34
9 Die konkatenierten Eingabevektoren der X-, Y- und Z-Komponente, durch eine zufällige Sortierung verrauscht und Original . . . 40
10 Fehlerquoten der Trainings- und Testmenge für unterschiedliche Cross Validation Einteilungen auf allen Datensätzen . . . 42
11 Mittelwerte der Hauptdiagonalen der inversen Kovarianzmatrizen für unterschiedliche Cross Validation Einteilungen . . . 43
12 Datensatz vor und nach einer Tiefpassfilterung mittels Butterworth Filter dritten Grades . . . 47
13 Expansionsartefakte in dem Vektor $avg1$ . . . 49
14 Fehlerwahrscheinlichkeit des Trainings- und Testdatensatzes für unterschiedliche Gaussdimensionen . . . 50
15 (a) stellt eine korrekt durchgeführte Geste vom Typ *X* dar, (b) eine inkorrekte Durchführung. . . . 52
16 Rohdaten des Datensatzes 431 . . . 53
17 Fehlerwahrscheinlichkeit der Test- und Trainingsmenge des Trainingsdatensatzes sowie die Fehlerwahrscheinlichkeit des Testdatensatzes für unterschiedliche α . . . 55
18 Graphische Darstellung der Quaternion-Hyperfläche . . . 59

Tabellenverzeichnis

1 Erkennungssicherheit auf dem vom Trainingsdatensatz trainierten Modells 23
2 Konfusionsmatrix mit den Klassen A, B und C 25
3 Mittelwerte aller Start- und Endbeschleunigungen auf dem Trainingsdatensatz 27
4 Mittelwerte aller Start- und Endbeschleunigungen auf dem Testdatensatz . 27
5 Die von MATLAB bestimmte Konfusionsmatrix des Trainingsdatensatzes nach der Vorverarbeitung mittels FFT 33
6 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 von dem Trainingsdatensatz 36
7 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testdatensatz 36
8 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach einer Translation zur Idealstartposition 37
9 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testdatensatz nach einer Translation zur Idealstartposition 37
10 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach der Translation zu 0 38
11 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testset nach einer Translation zu 0 38
12 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 ausschließlich auf den positiven Datensätzen. 45
13 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 ausschließlich auf den positiven Datensätzen. 46
14 Die von MATLAB bestimmte Konfusionsmatrix, der Gesten G1 bis G6 auf dem Testset 56
15 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach dem Training des Modells mit rotierten Datensätzen 60
16 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testdatensatz nach einer Translation zur Idealstartposition 61
17 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach dem Training des Modells mit rotierten Datensätzen und anschließender Translation zu 0 61

18 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach dem Training des Modells mit rotierten Datensätzen, dessen Ursprungslage durch den Mittelwert bestimmt wurde 62
19 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem liegend aufgenommenen Datensatz 63
20 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem liegend aufgenommenen Datensatz nach der Rotation mittels Quaternion 64
21 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem liegend aufgenommenen Datensatz nach der Rotation mittels Quaternion und einer Translation zu 0 64
22 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach dem Ausklingen 66
23 Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testset nach dem Ausklingen 66
24 Überblick über den Einfluss der Operatoren untereinander 71
25 Überblick über die Erkennungswahrscheinlichkeiten der unterschiedlichen Methoden 72

1 Einleitung

1.1 Motivation

Seit dem kommerziellen Durchbruch der Wii und der iOS-Gerätefamilie ist *Bewegung* als Eingabemodalität verstärkt in den Vordergrund gerückt. Verschiedene Arbeiten haben sich als Aufgabe gestellt, die Leistungsfähigkeit unterschiedlicher Klassifikationsverfahren an 3D-Gestendaten mit diesen Eingabegeräten zu bestimmen. Durch ein vorhergehendes Projekt, bei dem eine 3D-Gestenerkennung auf einem iPhone mittels SFA umgesetzt wurde, konnten verschiedene Aussagen und Erkenntnisse dieser Arbeiten bestätigt werden. Die Leistungsfähigkeit der SFA zum Klassifizieren von 3D-Gestendaten wurde zuvor von Hein[1] mittels Wii-Mote bestimmt und eine Umsetzung auf dem iPhone erzielte teilweise sehr gute Erkennungswahrscheinlichkeiten. Wenn jedoch ein Benutzer, der nicht im Trainingsdatensatz enthalten oder jemand, der mit dieser Art der Interaktion ungeübt war, Gesten durchführte, brach die Erkennungsleistung dramatisch ein. Nach einer Recherche ähnlicher Arbeit ist aufgefallen, dass keine Arbeit diese Erkennungsdiskrepanz zwischen benutzerabhängiger und benutzerunabhängiger Erkennung beleuchtet. Die Diskrepanz in der Erkennungsleistung scheint als gegeben akzeptiert zu sein.

1.2 Problemstellung und Ziele

Nachdem das Interesse an dieser Problematik geweckt war, tauchten die ersten Probleme auf. Da es keine vergleichbaren Arbeiten gab, gab es keinen „Leitfaden“, wie man sich dieser Problematik nähern konnte. Zahlreiche Fragen begleiteten die ersten Schritte dieser Arbeit: Wie vergleicht man die Eigenschaften in der Erkennung? Welche Einflüsse gibt es und wie kann man diese bestimmen? Gibt es charakteristische Eigenschaften, die sich zwischen benutzerabhängiger und benutzerunabhängiger Erkennung unterscheiden lassen? Welchen Einfluss haben die einzelnen Teilschritte, vom rohen Eingangsvektor bis hin zum Klassifikationsergebniss, auf die Erkennung? Wenn sich Einflüsse oder Faktoren bestimmen lassen, wie kann man diese ausgleichen?

Beginnend mit einer Reihe von Signalbetrachtungen zwischen Test- und Trainingsdatensatz sowie zwischen positiv und negativ erkannten Datensätzen, folgten zahlreiche Untersuchungen, um diesen Fragen auf die Spur zu kommen. Es hat sich schnell gezeigt, dass die klassischen Betrachtungsformen in dieser Problemstellung eher aussageschwach ausfielen.

Während der Aufnahme der Datensätze entschwand der Gedanke, eine 100% Erkennung zu erzielen. Die Personen ließen ihrem menschlichen Spieltrieb freie Bahn und nahmen die wildesten Variationen der vorgegebenen Gesten auf; ein Umstand der sich später als Vor- und Nachteil herausstellen sollte. Durch den Freiheitsgrad der Bewegung war es möglich, eine robuste Datenmenge aufzuzeichnen, die sich nahe an einer reellen Benutzung orientiert.

Als Ziel dieser Arbeit steht der Gedanke im Vordergrund, die Einflüsse und Eigenschaften des Systems auf die Erkennung so stark wie möglich vom Menschen zu entkoppeln. Das System soll trainiert eine robuste Erkennungsleistung anbieten und nicht durch seine Empfindlichkeiten den Menschen „trainieren“.

1.3 Aufbau der Arbeit

Im Folgenden werden zuerst einige Grundlagen beleuchtet. Diese umfassen den aktuellen Stand der Wissenschaft zu der Problematik der 3D-Gestenerkennung sowie einen Überblick über die Slow Feature Analysis und dem verwendeten Gauss-Klassifikator. Anschließend werden die verwendeten Gesten, deren Unterscheidung und die Aufnahme und Eigenschaften des Trainings- und Testdatensatzes beschrieben.
Da in dieser Arbeit Konfusionsmatrizen häufig zur Beurteilung der Modifikationen auf Erkennung herangezogen werden, schließt eine kurze Beschreibung dieser den Grundlagenabschnitt ab.

Im Abschnitt Signalbetrachtungen wird allgemein nach Einflüssen der Rohdaten auf die Erkennungswahrscheinlichkeit gesucht.
Der Einfluss der SFA, der Modellerzeugung und des Klassifikators auf die Erkennung wird im Abschnitt Systemeinfluss genauer betrachtet.

Im Anschluss folgen die Einflüsse des Menschen auf die Erkennungsleistung. Neben dem Bewegungsabbruch und der Ruhe wird der Einfluss der Geräterotation dargestellt sowie Möglichkeiten, wie diese zu Invarianzen überführt werden können. Als Abschluss werden die Ergebnisse zusammengefasst und ein Ausblick gegeben.

2 Grundlagen

Die Zahl der mobilen Geräte, die über Beschleunigungssensoren verfügen, steigt kontinuierlich an und der Gedanke, 3D-Gesten zur Steuerung von Anwendungen zu verwenden, wurde bereits von verschiedenen Autoren aufgegriffen und mit unterschiedlichen Verfahren beleuchtet. Alle Verfahren zeigen auf, dass teilweise sehr gute Erkennungsquoten in benutzerabhängigen Tests möglich sind, die Erkennungswahrscheinlichkeit bei benutzerunabhängigen Tests aber immer schlechter ist. Unabhängig davon, ob die Anwendungen benutzerabhängig oder benutzerunabhängig getestet wurden, erzielt kein Verfahren eine Erkennungsquote von annähernd 100% . Die Frage ist: Warum? Die Frage, welche Ausprägungen von Eigenschaften ein Gestendatensatz besitzen muss um falsch erkannt zu werden, wurde bis jetzt nicht betrachtet. Was ist an benutzerunabhängigen Datensätzen anders, dass sie eine geringere Erkennungswahrscheinlichkeit erzielen? Gedanken, die Erkennung durch eine Veränderung der Eingangsbetrachtungen zu verbessern, wurden von Campell [2] durchgeführt und dieser Gedanke ist seitdem in keiner weiteren Arbeit aufgegriffen worden.

Das Gebiet der Gestenerkennung ist aktuell von großem Interesse. Mit welchen Problemen es zu kämpfen hat, zeigt zum Beispiel die *Harry Potter* [3] App für das iPhone von Warner Bros. In dieser muss ein Anwender Zaubersprüche und Flüche mittels Gesten auslösen. Dazu muss jedoch das Gerät (absolut) gerade gehalten werden und während der Ausführung sollte der Benutzer jede Form der Rotation verhindern, um eine Erkennung möglich zu machen.

Seit März 2012 ist ein Programm namens *Flutter*[1] in einer Alpha Version verfügbar. Dieses nimmt Gesten mittels einer Webcam kontinuierlich auf und benutzt vordefinierte Gesten zum Steuern von Software wie zum Beispiel iTunes, YouTube oder ähnliche. Das verwendete Verfahren ist nicht beschrieben.

Im Folgenden werden Arbeiten der 3D-Gestenerkennung vorgestellt.

2.1 Stand der Wissenschaft

Eine der ältesten Arbeiten in dem Gebiet der räumlichen Gestenerkennung stammt aus dem Jahr 1996 von Campell et al. [2], in der die Bewegungen einer Person von einem

[1] https://flutter.io/

Computer gesteuerten Kamerasystem namens *STIVE* (Stereo Interactive Virtual Environment) beobachtet, analysiert und mittels HMM klassifiziert wurden. Als Gestenset wurden verschiedene T‘ai Chi Bewegungen ausgewählt. Die Autoren versuchten, Varianzen der Personen und der aufgezeichneten Bewegungen mit verschiedenen Methoden entgegenzuwirken. So testeten sie unter anderem das Modell mit den Ableitungen des Eingangssignals (Beschleunigungen), unterschiedlichen Formen der Winkelbeschreibung, deren Ableitungen oder eines DTW (Dynamic Time Warp) transformierten Signals zu trainieren und hielten Vor- und Nachteile der jeweiligen Methode fest. Insgesamt wurde das System mit zehn unterschiedlichen Kombinationen trainiert und getestet.

Perrin et al. [4] haben ein Laser-gestütztes System benutzt, um 3D-Gesten im Raum zu erkennen. Ein Laser wird mit einem Spiegel auf die Fingerspitze der Person gerichtet und folgt dieser Fingerspitze. Ein zweiter Spiegel nimmt die Reflexion des Lasers auf dem Finger auf, aus den Winkeln der beiden Spiegel lässt sich die Position des Fingers im Raum bestimmen. Die Autoren trainierten einem HMM basiertem System sechs Gesten an, die aus vereinfachten Formen der Buchstaben *A, B, C, D, E* und *S* bestanden. Die Trainingsparameter wurden durch die Geschwindigkeit, die Beschleunigung und die Ausrichtung der Spiegel gegeben. Die Autoren stießen auf Probleme, wenn Anfangs- und Endpunkt einer Geste identisch waren oder wenn es kurze Pausen während der Durchführung gab, wie zum Beispiel beim Zeichnen des zweiten Bogens des Buchstabens *B*. Sie führten die Probleme teilweise auf die HMM-basierten Methoden zur Erkennung zurück, weil diese den Prozess und die Entwicklung der Zustände beobachten, nicht aber für Gesten, die einen Moment der Ruhe beinhalten, geeignet sind. Eine Erkennungsquote gaben die Autoren nicht an.

2008 untersuchte Laviola [5] et al. die Leistungsfähigkeit von zwei Algorithmen in der Gestenerkennung mittels WiiMote. Es wurde ein Set von 8500 Gesten aufgenommen, bestehend aus 25 unterschiedlichen Gesten. Zur Erkennung wurden zwei Algorithmen verwendet, ein *AdaBoost*- und ein linearer Klassifikator. Der lineare Klassifikator basiert auf einer Arbeit von Rubine [6], in der zweidimensionale Gesten mittels eines Abstandsmaßes und eines linearen Klassifikators erkannt werden. *AdaBoost* [7] ist ein adaptives System, welches eine Kombination „schwacher“ Lernverfahren verwendet. Sheng [8] hat *AdaBoost* auf dreidimensionale Daten angewendet und eine Anfälligkeit bei verrauschten Daten festgestellt. Beide Algorithmen erkennen benutzerabhängig alle 25 Gesten mit einer Wahrscheinlichkeit von mehr als 90%. Benutzerunabhängig wurden neun Gesten mit

einer Wahrscheinlichkeit von über 90% erkannt, wenn das System mit 100 Datensätzen pro Geste trainiert wurde.

Klingmann [9] hat 2009 ein weiteres HMM auf einem iPhone trainiert und mit diesem eine benutzerabhängige Erkennungswahrscheinlichkeit von $\approx 90\%$ erzielt. Trainiert wurde das System mit fünf Gesten (Kreis, Quadrat, Dreieck, Z, Bowling) und unterschiedlich großen Trainingssets. Auffällig ist, dass innerhalb des Trainingssets starke Differenzen in der Erkennung der unterschiedlichen Gesten bestehen, auf die der Autor nicht eingeht.

Ein Verfahren, welches auf 3D-Bilddaten basiert, hat Holte [10] entwickelt und dabei ausschließlich die Bewegungen charakteristischer Gesten auswertet. Es gibt zwei Gestensets. Eines besteht aus elf primitiven Gesten, die entweder mit einem oder zwei Armen durchgeführt werden, während das zweite aus vier Gesten besteht: *nach rechts zeigen*, *den Arm heben*, *klatschen* sowie *winken*. Diese Bilddaten wurden mit einer speziellen Infrarot-Kamera aufgezeichnet und mittels wahrscheinlichkeitstheoretischen Bearbeitungsabstandes[2] klassifiziert. Die Autoren erreichten eine Erkennungsrate von 82,9%, wobei der Winkel der Aufnahmen keinen (oder nur marginalen) Einfluss auf die Erkennung hatte.

Wu [11] et al. hat Gestendaten von zwölf Gesten mit einer WiiMote aufgezeichnet und verschiedene Verfahren, diese zu analysieren, mit dem Ziel der Benutzerunabhängigkeit, getestet. Neben HMM, DTW, C4.5 und einem naiven Bayes Klassifikator wurde das von den Autoren entwickelte Verfahren FDSVM (Frame-Based Descriptor and multiclass Support Vector Machine) getestet, welches neben den zeitlichen auch spektrale Eigenschaften mit in die Analyse aufnimmt. Die Klassifikation mittels SVM erfolgt laut den Autoren nicht linear. Als spektrale Eigenschaften hat Wu neben dem Kosinus-0-Anteil, die Energie, die Entropie, die Standardabweichung und den Korrelationsfaktor mit in die Analyse aufgenommen. Erzielt wurde eine benutzerunabhängige Erkennungsquote von 89,29% für die zwölf Gesten.

Ebenfalls mittels WiiMote und HMM hat Prekopcsák [12] Arbeiten im Bereich 3D-Gestenerkennnung mit automatischer Gestensegmentierung durchgeführt. Die Segmentierung, also das Markieren von Gestenbeginn und Gestenende, erfolgte über einen Schwellwert. Von vier Personen wurden insgesamt 400 Gesten, der insgesamt 10 verschiedenen Gesten, aufgenommen, wobei die einzelnen Gesten nicht näher in der Arbeit beschrieben

[2] Probabilistic Edit Distance

wurden. Neben einer Klassifikation mittels HMM wurde auch eine Klassifikation mittels Support Vector Machine durchgeführt. Im benutzerabhängingen Training erzielte das System einer Erkennungswahrscheinlichkeit von $\approx 96\%$.

Einen gänzlich anderen Ansatz verfolgten Cutler et al. [13]. Diese haben Gesten wie springen, drehen, mit den Armen flattern und ähnliche mit einer Kamera in Echtzeit mittels optischen Flusses analysiert und mit einem nicht näher beschriebenen Zustandsautomaten klassifiziert. Als Ziel nannten die Autoren eine interaktive Umgebung für Kinder zu erschaffen. Eine Erkennungswahrscheinlichkeit gaben die Autoren nicht an.

Zusammenfassend fällt auf, dass alle betrachteten Arbeiten mit teilweise völlig unterschiedlichen Ansätzen jeweils eine benutzerabhängige Erkennungswahrscheinlichkeit von $\approx 90\%$ erzielen. Außer in Campell et al. [2] geht keine der Arbeiten einem Verbesserungspotenzial oder den Gründen für diese Erkennungswahrscheinlichkeit auf die Spur.

Im Folgenden werden verschiedene Grundlagen zu dieser Arbeit dargestellt. Im Anschluss wird auf den Einfluss der Erkennungsmethode und dem Einfluss des Menschen eingegangen.

2.2 Slow Feature Analysis (SFA)

Die Slow Feature Analysis ist ein Verfahren zur Merkmalsgewinnung hochdimensionaler, zeitlicher Daten. Diese Idee wurde 1998 von Wiskott [14] als Ansatz zum unüberwachten Erlernen invarianter und nicht korrelierender Eingangsdaten vorgestellt. Zur Extraktion von Informationen versucht die SFA, in den Eingangsdaten den langsamsten Prozess oder „die treibende Kraft“ zu bestimmen. Der Grundgedanke ist hierbei, dass sich ein Signal nur langsam verändert, während die Sensoren durch die Abtastung dieses verrauschen. Berkes [15] hat gezeigt, dass die Interklassenvarianz der langsamsten Prozesse hoch ist, während die Intraklassenvarianz niedrig ausfällt.

Bei der SFA handelt es sich um ein Verfahren, welches mit einem „One-shot learning“ trainiert wird. In diesem wird das Modell einer SFA nur einmal mit einer Trainingsmenge trainiert. Soll das Modell erweitert oder verändert werden, wird das alte Modell durch das neu erstellte Modell vollständig ersetzt. Die Anwendung einer SFA erfolgt mit einem zuvor erstellten Modell. Insgesamt lassen sich hierbei zwei Schritte klar abgrenzen: die Modellerzeugung, die als unüberwachtes Training stattfindet und die Anwendung auf

einem zuvor trainierten Modell.

Nach Walter [16] muss, um diese Aufgabe zu lösen, für eine Zeitreihe $x(t)$ die Eingabe-Ausgabe-Funktion $g(x)$ gefunden werden, für die das Ausgabesignal $y(t) = [y_1(t) \dots y_N(t)]$ mit $y_n(t) := g_n(x(t))$ gilt:

$$\langle \dot{y}_n^2 \rangle = \text{Min} \tag{1}$$

Wobei folgende Nebenbedingungen für das zeitliche Mittel gelten, welches durch spitze Klammern dargestellt wird:

$$\langle y_n \rangle = 0 \tag{2}$$

$$\langle y_n^2 \rangle = 1 \tag{3}$$

$$\forall n' < n : \langle y_n' y_n \rangle = 0 \tag{4}$$

Gleichung 2 besagt die Mittelwertfreiheit, Gleichung 3 die Standardabweichung und Gleichung 4 steht für die Dekorrelation des Ausgabesignals. Durch diese wird sicher gestellt, dass die Ausgabesignale unterschiedliche Informationen besitzen. Weitere Informationen zur SFA können Walter [16] oder Berkes [15] entnommen werden.
Für diese Arbeit wurde eine Umsetzung der SFA in MATLAB verwendet [17].

Da für diese Arbeit die Anwendung der SFA von besonderem Interesse ist, sollen die Teilschritte zur Anwendung zuerst grob und im Anschluss präzise dargestellt werden. Diese sind:

1. **Vorverarbeitung**:
   Bestehend aus der Längennormierung, Vektorkonkatination der X, Y und Z-Komponente sowie der anschließenden Amplidutennormierung. Erweitert wird der normierte Vektor um den Faktor der Längennormierung und Amplitudennormierung.

2. **Signalanalyse**:
   Für diese wird zuerst der bei der Vorverarbeitung bestimmte Mittelwert entfernt, der Eingabevektor einem Sphering unterzogen, expandiert und im Anschluss mit den ebenfalls zuvor bestimmten Eigenvektoren multipliziert.

3. **Klassifikation**:
   Reduktion auf $GaussDim$ Elemente, Entfernung des Klassenmittelwerts, der Wahrscheinlichkeitsbestimmung, Hauptkomponentenbetrachtung und der anschließenden Wahl des Maximums aus den bestimmten Wahrscheinlichkeiten. Der Index des Maximums stellt den Index der Geste.

Alle Punkte sind in dieser Arbeit von Interesse. Aus dem Vektor innerhalb von Punkt 1 können rohe Informationen vor der eigentlichen Analyse gezogen werden. Auch eine direkte Manipulation der rohen Eingangsdaten ist an dieser Stelle möglich. Punkt 2, die Signalanalyse, die im Grunde eine Transformation des vorverarbeiteten Eingabevektor in einen zu klassifizierenden Bestimmungsvektor durchführt, ist bei der Anwendung einer SFA entscheidend. Durch den gesamten Prozess und seine Komplexität ist es von Bedeutung, Teilaspekte zu isolieren und diese zu optimieren. Die wichtigsten Bestandteile beziehungsweise Teilschritte, die die Qualität der Erkennung beeinflussen, werden später noch vorgestellt. Der Klassifikation mittels Gauss-Klassifikator ist der folgende Abschnitt gewidmet. Die Eigenschaften des Modells, den Einfluss von Teilen und die Abhängigkeiten untereinander, werden in den darauf folgenden Abschnitten betrachtet. Zuvor soll die Anwendung der SFA beschrieben werden.

Zur Anwendung der SFA auf einem trainierten Modell wird vom normierten Eingangsvektor v zuerst der im Training bestimmte Mittelwert aller Eingangsvektoren m_0 subtrahiert.

$$v_m = v - m_0 \tag{5}$$

Im Anschluss wird der vom Mittelwert befreite Vektor v_m mit der Sphering-Matrix W_0 multipliziert.

$$v_w = W_0 v_m \tag{6}$$

Während v_m noch mit v in der Dimension übereingestimmt hat, so hat der gespherte Vektor v_w durch die Multiplikation die Dimension der Zeilen der Sphering-Matrix ange-

nommen. Dieser „reduzierte“ Vektor wird im nächsten Schritt durch eine Monommultiplikation ersten Grades expandiert.

$$e = \text{expand}(v_w) \tag{7}$$

Die Expansion soll hier kurz an einem Beispiel beschrieben werden. Der Vektor t besteht aus den beiden Elementen x_1 und x_2. Der expandierte Vektor e besteht aus den Elementen x_1, ${x_1}^2$, x_1x_2, x_2, ${x_2}^2$. Von dem expandierten Vektor e wird der Mittelwert aller expandierten Vektoren e_0 des Trainingsdatensatzes subtrahiert und im Anschluss mit den zuvor ermittelten Eigenvektoren ${w_j}^T$ multipliziert.

$$y_{in} = {w_j}^T(e - e_0) \tag{8}$$

Der so erzeugte Vektor y_{in} kann nun der Klassifikation zugeführt werden.

2.3 Gauss-Klassifikator

Die Klassifikation des durch die SFA bestimmten Vektors erfolgt durch einen Gauss-Klassifikator. Der Gauss-Klassifikator ist ein deskriminativer, relativ einfach zu trainierender Klassifikator, der zum Training vollständig klassifizierte Trainingsdaten benötigt. Somit ist ein Gauss-Klassifikator ein überwachtes System. Jeder Trainingsdatensatz wird als ein Punkt in einer M dimensionalen Hyperfläche aufgefasst. Über diese Punkte wird für die Klassen k für $k \in 1...G$, wobei G die Anzahl der möglichen Klassen ist, pro Klasse ein Mittelpunkt und die Klassenabweichung bestimmt. Abbildung 1 verdeutlicht dieses an einem einfachen Beispiel. Die beiden Klassen **1** und **2** wurden nach den Eigenschaften **A** und **B** aus den Trainingsdaten erstellt. Nun soll der neue Datensatz **X** klassifiziert werden. Obwohl sich dieser räumlich näher an der Klasse **1** befindet, wird dieser Klasse **2** zugeordnet. Grund hierfür ist die starke Ausprägung von **1** auf der Eigenschaft **B**. **X** befindet sich, wenn die Standardabweichung betrachtet wird, weiter von **1** entfernt als zu **2**.

Aus diesen Eigenschaften ergibt sich der Vorteil, dass ein Datensatz relativ einfach mit geringem Rechenaufwand einer Klasse zugeordnet werden kann. Unabhängig davon, wie viele Datensätze für das Training verwendet wurden, hängt der Rechenaufwand alleinig von der Dimension M und der Anzahl der Gestenklassen G ab. Jedoch ergibt sich der Nachteil, dass eine Klasse k_h mit hoher Varianz, einer schwachen Ausprägung in der Hyperfläche, zu zahlreichen neuen Punkten einen kleineren Abstand aufweist als eine gut

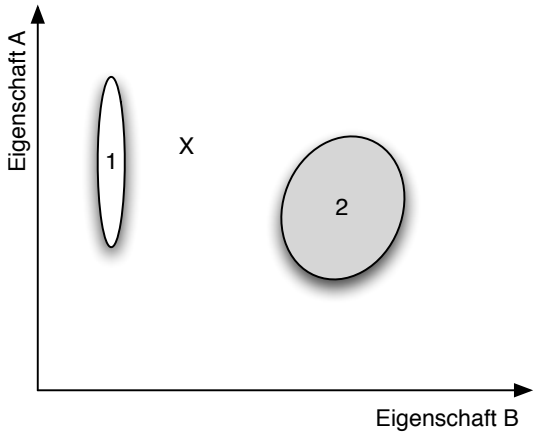

Abbildung 1. Zwei Gaussklassen 1 und 2 nach den beiden Eigenschaften A und B

spezifizierte Klasse k_l mit geringer Varianz. Wenn ein Punkt auch nur knapp neben k_l liegt, kann er bereits zu k_h klassifiziert werden. Mit dieser Problematik sind leider alle Klassifikationsverfahren in unterschiedlicher Ausprägung belastet und die Gradwanderung zwischen über- und unterangepassten Klassifikator bestimmt maßgebend über seine Brauchbarkeit. [18]

Das Training des Klassifikators erfolgt über die Klassifikationsvektoren aus dem Trainingsdatensatz, die der Klasse $k = 1...G$ angehören. Zum Anwenden eines trainierten Gauss-Klassifikators wird der durch die SFA verarbeiteten Eingabevektor y_{in} zuerst auf die zu betrachtenden Gauss-Elemente reduziert und im Anschluss der klassenspezifische Mittelwert $y_{0,k}$ der jeweiligen Gestenklasse subtrahiert:

$$y_k = y_{in,gauss} \quad gauss \in 1...\text{GaussDim} \tag{9}$$

$$a_k = y_k - y_{0,k} \tag{10}$$

Im Anschluss wird die Klassenwahrscheinlichkeit $p_{(y|k)}$ der jeweiligen Klasse k bestimmt, wobei f_k ein klassenspezifischer Vorfaktor ist, der von dem Betrag der Determinante der Klassenkovarianzmatrix abhängt. iG_k ist die inverse Kovarianzmatrix.

$$p_{(y|k)} = f_k \exp^{\left(\frac{{a_k}^T \, iG_k \, a_k}{2}\right)} \tag{11}$$

Da jede Geste mit einer anderen a-priori Wahrscheinlichkeit P_k eintritt, wird diese entfernt. Hierzu wird die Summe s aller Klassenwahrscheinlichkeiten in Relation zu ihrer a-priori Wahrscheinlichkeit gebildet.

$$s = \sum_{k=1}^{G} p_{(y|k)} P_k \tag{12}$$

Mit dieser Summe und der a-priori Wahrscheinlichkeit lässt sich die eigentlich Klassenwahrscheinlichkeit r_k normieren durch

$$r_k = \frac{p_{(y|k)} P_k}{s} \tag{13}$$

Der Index mit dem maximalen Betrag in r_k entspricht der Klasse, zu dem der Eingabevektor y_{in} gehört.

2.4 Gesten

Im Folgenden sollen kurz die Gesten vorgestellt werden, die in den Datensätzen enthalten sind. Gesten stellen Ikonisierungen von Zeichen dar, die mit dem iPhone in die Luft „gemalt“ werden. Hierzu wird ein auf das iPhone angepasstes Gestenset verwendet, weil (hoch)dynamische Gesten, wie zum Beispiel die Frisbee- oder Bowlinggeste mit einem iPhone, welches deutlich weniger ergonomisch gebaut ist als zum Beispiel ein Wii-Controller, zu kostspieligen „Fehlversuchen“ führen können und der Beschleunigungssensor vom Typ LIS332DLH [19], der in den iOS Geräten verbaut ist, nur bis zu 2,1g in X, Y und Z-Richtung aufzeichnen kann. Es wurden folgende Kriterien erstellt, die die Gesten erfüllen müssen:

- **Einfachheit**
  Eine Geste darf keine komplizierte Struktur oder Muster beinhalten. Jede Person sollte die Geste sofort oder nach wenigen Versuchen „zeichnen“ können.

- **Varianz**
  Um die Robustheit des Systems zu testen, sollte eine Geste verschiedene Wege zur Umsetzung anbieten.

- **Geschwindigkeit**
  Jede Geste sollte auch in knapper Zeit in hinreichender Geschwindigkeit ausführbar

sein. Da die Feinmotorik des Menschen mit zunehmender Ausführungsgeschwindigkeit abnimmt, muss die Geste in ihrer Struktur eine gewisse Robustheit aufweisen.

Es wurden sechs Gesten ausgewählt. Drei Gesten besitzen einen kontinuierlichen Charakter (Kreis, Epsilon, Unendlichkeit). Drei Gesten besitzen mindestens ein diskreten Moment in der Durchführung, in der das Gerät für einen kurzen Moment still gehalten wird und / oder eine drastische Beschleunigung durchgeführt wird (X, Z und Herz). In Abbildung 2 werden die verwendeten Gesten dargestellt.

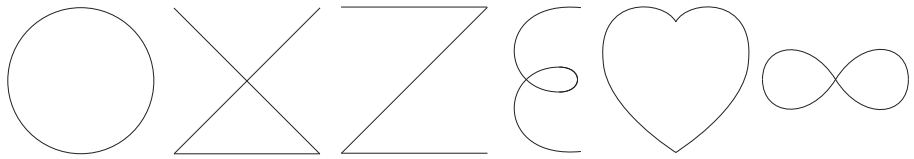

Abbildung 2. Die sechs Gesten, die im Trainingsset und Testset aufgenommen wurden, der Kreis, das X, das Z, das Epsilon, das Herz und das Unendlichkeitszeichen.

2.5 Geräterotation versus Gestenrotation

Wenn in dieser Arbeit der Begriff Rotation verwendet wird, ist immer die Gestenrotation gemeint. Unterschieden werden kann zwischen der Gestenrotation und der Geräterotation. Eine Geräterotation ist der Fall, wenn eine Geste in ihrer Bewegung richtig durchgeführt wird, aber die Lage des Gerätes deutlich von der Normalstellung abweicht. Dieses tritt zum Beispiel auf, wenn Gesten im Liegen durchgeführt werden. Eine Gestenrotation ist eine Geste, die relativ zum Menschen betrachtet, in einer anderen Hauptachse durchgeführt wird. Gestenrotationen können und werden teilweise durch das generierte Modell der SFA erkannt. Bedingungen für die Erkennung sind:

- **Symmetrie**: die Geste muss symmetrisch sein. So kann die Kreisgeste im und gegen den Uhrzeigersinn durchgeführt werden, wobei eine Durchführung gegen den Uhrzeigersinn einer Rotation um 180 Grad auf der Y-Achse entspricht.
- **Umsetzung**: der Mensch muss in der Lage sein, die rotierte Bewegung durchzuführen, ohne die „Natur der Bewegung“ zu verändern oder zu verzerren.

Wenn eine Gestenrotation durchgeführt wird, handelt es sich im Grunde um eine neue, eine andere Geste. Diese Semantik bleibt der SFA jedoch verborgen, wenn die Bewegung

in ihrer Natur der nicht rotierten Bewegung entspricht. Insbesondere bei der Kreisgeste zeigt sich die Vielfalt der Gestenrotation. So kann diese Geste links und rechts drehend durchgeführt werden, was einer Rotation um die Z-Achse entspricht. Auch kann diese Geste an jedem Punkt des Kreises beginnen, was jeweils eine Rotation um die X-Achse und Y-Achse bedeutet.

Da es sich bei der Gestenrotation im Grunde um eine andere Geste handelt, wird „nur“ versucht, die Geräterotation zu kompensieren. Diese ist zwar auch durch einen menschlichen Einfluss entstanden, sollte aber Aufgrund der Bewegungsnatur einer Geste durch eine entsprechende Kompensation getilgt werden können.

2.6 Test- und Trainingsdatenerhebung

Die Generation des Trainingsdatensatzes erfolgte mit einer dafür angefertigten iPhone Anwendung. Mit dieser konnten schnell und unkompliziert die unterschiedlichen Gestendaten aufgezeichnet werden. Die Daten wurden direkt in die Konsole ausgegeben und konnten so über das Terminal ausgelesen werden. Dafür musste das Gerät an einen Computer angeschlossen sein. Dieser Prozess wurde so gewählt, weil während der Erhebung erhebliche Datenmengen anfallen und eine Einschränkung des Aufnahmeprozesses nicht gewünscht war. Jede Person sollte jede Geste ungefähr 10 Mal durchführen. Zu der Aufzeichnung der Gesten wurden keinerlei Vorgaben gemacht, es stand allen Personen frei, wie sie die Gesten durchführen wollen. Dieses führte unter anderem zu einer sehr hohen Varianz in der Kreis- und Epsilongeste, die links und rechts drehend an fast jeder Position begonnen wurde. Einige Personen verhielten sich sehr konsequent und führten einen Gestentyp immer gleich aus, während ein Teil der Personen hohe Variationen in der Durchführung an den Tag legten. Sie wollten das noch nicht vorhandene System testen.[3]

Der Testdatensatz wurde mit einer erweiterten Version der Anwendung direkt auf dem Gerät aufgezeichnet. Diese Version besitzt zum einen das vollständig generierte Modell des Trainingsdatensatzes inklusive der Klassifikation, sodass die Benutzer direkt sehen konnten ob „ihre Geste“ richtig erkannt wurde und zum anderen wurden die aufgezeichneten Rohdaten nach jeder Aufzeichnung automatisch lokal zwischengespeichert, sodass eine Beeinflussung des Aufnahmeprozesses ausgeschlossen werden kann. Alle Datensätze wurden mit dem gleichen Gerät aufgenommen, um Schwankungen zu vermeiden.

[3]Dass aus den teilweise sehr extremen Gesten dennoch eine relativ gute Erkennung wurde, überrascht (mich) immer wieder.

Das Datenformat zur Aufnahme der Daten wurde von Hein[1] übernommen, um Tests in MATLAB ohne Anpassungen durchführen zu können. Die Datensätze wurden im CSV-Format gespeichert, wobei eine Zeile acht Information beinhaltet:

1. **Session ID**
   Die Session ID dient der Identifikation eines Datensatzes und darf nur einmalig vergeben sein. Um dies zu gewährleisten, wurde die Session ID automatisch nach jeder Geste gespeichert und inkrementiert.

2. **Gesten ID**
   Die Gesten ID stellt für das System die Trainingsreferenz dar. Nur durch diese ID kann das System mit den vorhandenen Daten trainiert werden. Bei der Aufzeichnung des Trainingsdatensatzes war die Geste vorgeschrieben, während bei der Aufzeichnung des Testdatensatzes der Benutzer nach jeder Geste vom Gerät durch ein *Pop-Over* gefragt wurde, welche Geste durchgeführt wurde.

3. **Timestamp**
   Beim Aufzeichnen wurde dieses Feld leer gelassen. Durch die Eigenschaften des Gerätes, dass die Daten lokal zur Verfügung stehen und keine Varianzen z.B. durch eine Übertragung auftreten, können äquidistante Daten angenommen werden.

4. **X-Beschleunigung**
   Beinhaltet die aktuelle Beschleunigung in X-Richtung.

5. **Y-Beschleunigung**
   Beinhaltet die aktuelle Beschleunigung in Y-Richtung.

6. **Z-Beschleunigung**
   Beinhaltet die aktuelle Beschleunigung in Z-Richtung.

7. **Index**
   Der Index stellt die Chronologie[4] eines Datensatzes zur Verfügung.

8. **Personen ID**
   Um Varianzen oder Tests innerhalb eines Datensatzes durchführen zu können, wurde auf dem Trainingsdatensatz auch die Personen ID aufgezeichnet.

[4]Hinweis: MATLAB beginnt leider bei 1 und nicht wie üblich bei 0 zu zählen.

2.7 Datensatzbetrachtung

Für die Tests und Betrachtungen wurde auf zwei Sets an Datensätzen zurückgegriffen. Ein Trainingsdatensatz, bestehend aus 892 Datensätzen, die von 14 Personen aufgenommen wurden und ein Testdatensatz, bestehend aus 238 Datensätzen, die von 11 Personen aufgenommen wurden. Keine der 11 Personen aus dem Testdatensatz ist im Trainingsdatensatz enthalten. Beide Datensätze liegen als Rohdaten vor, sodass auf die Beschleunigungsdaten der X-, Y- und Z-Achse zugegriffen werden kann. Diese Daten wurden mit 100Hz erfasst. Alle Gesten kommen ungefähr mit gleicher Häufigkeit in den jeweiligen Datensätzen vor. Zur Betrachtung des Trainingsdatensatzes wurden zwei Einteilungen durchgeführt, eine Grobeinteilung nach allgemeiner Erkennung und eine Feineinteilung nach Klassifikationswahrscheinlichkeit.

2.7.1 Grobeinteilung des Trainingsdatensatzes

Zu einer groben Einteilung der Trainingsdaten wird der Sachverhalt ausgenutzt, dass das erstellte Modell von der Anzahl der zum Training verwendeten Datensätze abhängt. Aus dem Trainingsdatensatz wird eine Trainingsmenge zum trainieren und eine Testmenge zur Überprüfung des Modells ausgewählt. Durch unterschiedliche Mengenzuteilungen kann die SFA so mit leicht unterschiedlichen Modellen getestet werden. So wurde unter anderem eine Verteilung von Trainingsmenge zu Testmenge von 50% zu 50%, 75% zu 25%, 80% zu 20% und 90% zu 10% angewendet, was unterschiedlichen Einteilungen der Cross Validation entspricht.

Es wurde geprüft, welche Datensätze von der SFA mit unterschiedlichen Cross Validation Einteilungen erkannt wurden, welche nicht erkannt wurden und welche Übereinstimmungen bei den unterschiedlichen Verteilungen auftrat. Dadurch ergibt sich folgende Einteilung, die für diese Arbeit verwendet wird:

- **Positive** Datensätze sind Datensätze, die von allen Einteilungen erkannt wurden.
- **Negative** Datensätze sind Datensätze, die von allen Einteilungen nicht erkannt wurden.
- **Labile** Datensätze sind Datensätze, die von nur einem oder mehreren Einteilungen, aber nicht von allen erfolgreich erkannt wurden.

Der Grundgedanke zu dieser Einteilung ist, dass positive Datensätze etwas innehaben, was sie von allen Einteilungen richtig erkennen lässt. Negative Datensätze haben Eigenschaften, durch die sie anderen Gesten zugeordnet werden und labile Datensätze scheinen genau diese Eigenschaften in schwächerer oder anders ausgeprägter Natur zu besitzen, da sie teilweise erkannt werden.

2.7.2 Feineinteilung des Trainingsdatensatzes

Zur „feinen“ Einteilung der Daten wurden die positiven und negativen Datensätze nach ihrer Wahrscheinlichkeit, mit der sie erkannt werden, eingestuft. Auffällig hierbei ist die Entschlossenheit, mit der das System klassifiziert, wie in Tabelle 1 zu sehen ist. Mit einer Sicherheit von 100% werden insgesamt 136 Datensätze erkannt. Davon sind 135 korrekt erkannt und ein Datensatz ist ein Trugschluss. Interessant ist, dass nur 44 Datensätze, das entspricht 4,9%, mit einer Sicherheit von kleiner gleich 65% erkannt werden. Das Modell scheint sich vollständig und höchst präzise an den Trainingsdatensatz angepasst zu haben.

Sicherheit in %	**positive Datensätze**	**negative Datensätze**
$= 100\%$	135	1
$\leq 99\%$	658	7
$\leq 95\%$	717	22
$\leq 90\%$	747	30
$\leq 85\%$	763	36
$\geq 55\%$	7	14
$\geq 60\%$	11	20
$\geq 65\%$	16	28

Tabelle 1. Erkennungssicherheit auf dem vom Trainingsdatensatz trainierten Modells

Unter den Datensätzen, die mit 100% Wahrscheinlichkeit erkannt werden, wurde eine interessante Verteilung festgestellt. So gibt es keine *Herz*geste unter diesen Datensätzen, *Kreis*- und *Unendlichkeit*gesten kommen jeweils sechsmal vor. Die *X*-Geste ist elfmal vertreten, weit abgeschlagen von der *Z*-Geste mit 46 Vorkommnissen. Über die Hälfte aller Datensätze, die mit 100% Wahrscheinlichkeit erkannt werden, fallen auf die *Epsilon*-Geste. Eine der Ursachen für die Entschlossenheit kann sein, dass die Datensätze des Trainingsdatensatzes auf dem Modell angewandt wurden, mit dem das System auch trainiert wurde.

Auf dem Testdatensatz verhält sich der Klassifikator ähnlich entschlossen, obwohl alle

Personen aus dem Testdatensatz nicht im Trainingsdatensatz enthalten sind. Der Mittelwert aller Erkennungswahrscheinlichkeiten auf dem gesamten Testdatensatz beträgt 91%! Von den 247 aufgezeichneten Gesten werden 54 mit 100% Wahrscheinlichkeit klassifiziert, 9 davon werden falsch klassifiziert. In dem Datensatz der negativ erkannten Datensätze befinden sich neben den 9 100% noch weitere 25 Datensätze, die eine Erkennungswahrscheinlichkeit von über 99% besitzen. Die Erkennungswahrscheinlichkeiten der positiven und negativen Datensätze sind in Abbildung 3 dargestellt. Während bei den positiven Datensätze nur 19 Datensätze eine Erkennungswahrscheinlichkeit von unter 90% besitzen, was $\approx 14\%$ entspricht, besitzt die Menge der negativen Datensätze 41 Datensätze mit einer Erkennungswahrscheinlichkeit von unter 90%. Mit sinkender Erkennungswahrscheinlichkeit steigt die Wahrscheinlichkeit der Fehlerkennung.

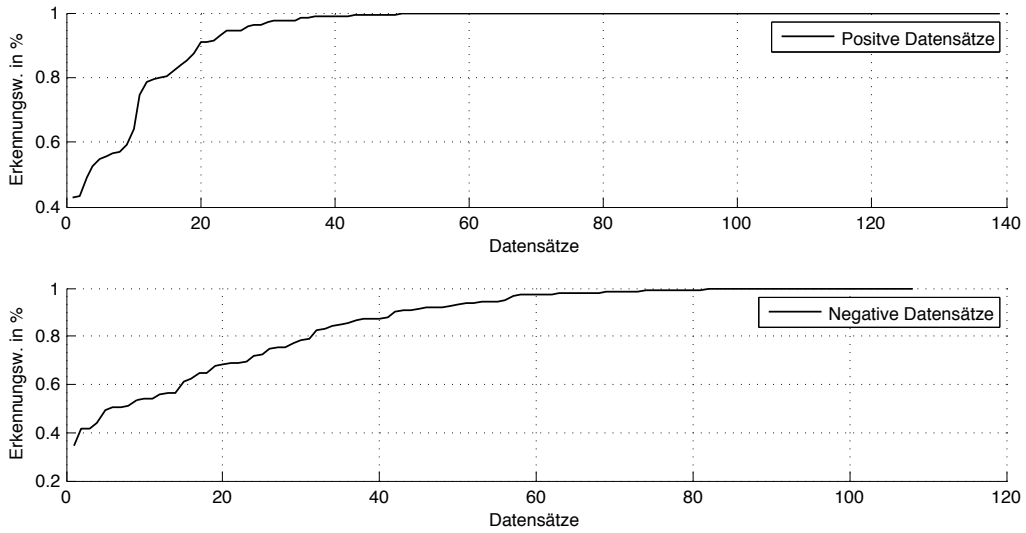

Abbildung 3. Erkennungswahrscheinlichkeiten für den Testdatensatz, getrennt nach positiven und negativen Datensätzen

In den negativen Datensätzen ist noch eine weitere Besonderheit gegeben. Wenn diese mit einer Wahrscheinlichkeit von über 90% erkannt werden, ist die Wahrscheinlichkeit in den anderen Klassen meist relativ gleich verteilt. Mit sinkender Erkennungswahrscheinlichkeit gibt es in der Regel eine oder zwei weitere Klassen mit einem Betrag über einem Prozent, jedoch konnte keine signifikante Wahrscheinlichkeit bestimmt werden, dass der zweithöchste Betrag zu der richtigen Gestenklasse gehört.

2.8 Konfusionsmatrix

In den folgenden Abschnitten werden immer wieder Konfusionsmatrizen zur Darstellung der Ergebnisse verwendet. Diese sind ein Werkzeug um übersichtlich die Erkennungseigenschaften eines Klassifikators zu repräsentieren. Tabelle 2 zeigt eine Beispielmatrix. Die oberste Zeile gibt die wahren Klassen an, die erste Spalte die zugeordneten Klassen.

	A	B	C
A	10	0	2
B	0	9	2
C	0	1	6

Tabelle 2. Konfusionsmatrix mit den Klassen A, B und C

Zu erkennen ist, dass die Klasse A (Spalte 2) ohne eine Fehlerkennung erkannt wurde, die Klasse B eine Fehlerkennung zu Klasse C besitzt und Klasse C jeweils zwei Fehlerkennungen zu Klasse A und B besitzt. Durch diese Form kann der Einfluss einer Methode oder einer Änderung auf die Erkennungsqualität überschaubar dargestellt werden. Weiterhin sind Muster, in Form von verstärkten Fehlerkennungen, gut zu erkennen. Wenn eine Konfusionsmatrix *nur* aus der Hauptdiagonalen besteht, dann gibt es keine Fehlerkennung.

Im Folgenden soll zuerst der Einfluss der gewählten Methoden betrachtet werden. Hierbei wird nicht auf Alternativen eingegangen, sondern dieser Abschnitt beschäftigt sich ausschließlich mit der Fragestellung, ob ein Einfluss auf die Erkennungswahrscheinlichkeit vorliegt. Darauf werden die Gestendaten des Trainings- und Testdatensatz beleuchtet. Versucht wird, Korrelationen zwischen Signaleigenschaften und der Erkennungswahrscheinlichkeit zu finden und festzustellen, welche dieser Korrelationen hauptsächlich den Menschen als Ursache haben. Wenn es diese Einflüsse gibt, lassen sich diese reduzieren beziehungsweise sogar vollständig kompensieren?

3 Signalbetrachtungen

3.1 Signaleigenschaften

Bevor auf den Einfluss des Erkennungsprozesses oder andere Faktoren wie der Mensch oder allgemeine Signaleigenschaften eingegangen wird, soll hier eine allgemeine Signalbetrachtung stattfinden. In dieser werden die Eigenschaften der Eingangssignale analysiert und ggf. auch verändert, um deren Einfluss auf die Erkennung zu bestimmen. Die Betrachtungen versuchen Auffälligkeiten zu entdecken, insbesondere im Bezug auf die Verteilung dieser Eigenschaften bei den positiv und negativ erkannten Datensätze.

3.1.1 Vertrautheit mit dem Gerät

Es hat sich herausgestellt, dass im benutzerunabhängigen Test die Erkennungsquote stark davon abhing, ob die Benutzer vorher schon mit einem iPhone oder ähnlichem Gerät interagiert haben oder nicht. Benutzer, die zum ersten Mal ein iPhone in der Hand hielten, erzielten teilweise dramatische schlechte Erkennungsquoten. Die Durchführung der Gesten war sehr stockend, die Natur der Bewegung war nicht gegeben. Ebenfalls ist aufgefallen, dass Benutzer der Anwendung das Gerät nicht wie üblich in der Hand hielten, sondern „künstlich gerade“ oder versuchten, besonders präzise die Gesten zu zeichnen, was ebenfalls Einfluss auf die Bewegung hatte.

Zusammenfassend muss an dieser Stelle gesagt werden, dass Benutzer, die vertraut mit einem Gerät von dem Format eines iPhones waren, bessere Ergebnisse erzielten haben. Leider lässt sich nicht ausschließen, dass wahrscheinlich auch eine Hemmschwelle existiert, ein Gerät schnell zu bewegen.

3.1.2 Start- und Endhaltung des Gerätes

Der Trainings- und der Testdatensatz wurden bezüglich der Start- und Endhaltung des Gerätes untersucht. Diese Start- und Endhaltung beschreibt die Haltung des Gerätes im Raum. Diese wird jeweils durch das erste und letzte Element im Datensatz bestimmt. Ziel dieser Betrachtung war zum einen das Bestimmen einer Standardstarthaltung und zum anderen, wie sich im Mittel die Haltung des Gerätes im Verlauf einer Geste verhält.

Tabelle 3 stellt die bestimmten Werte des Trainingsdatensatzes dar. Es ist erkennbar, dass das Gerät zu Beginn einer Geste eine annähernd gleich starke Ausprägung in Y und

	Beschleunigung in X	Beschleunigung in Y	Beschleunigung in Z
Starthaltung	-0,098g	-0,62g	-0,65g
Endhaltung	-0,08g	-0,46g	-0,66g

Tabelle 3. Mittelwerte aller Start- und Endbeschleunigungen auf dem Trainingsdatensatz

Z Richtung besitzt, was, wenn man das Gerät in der Hand hält, einer sehr angenehmen Haltung in einem Winkel von $\approx 45°$ entspricht. Die X-Komponente ist geringfügig nach links ausgeprägt. Die Vermutung liegt nahe, dass dieser Betrag durch ein Ungleichgewicht der Personen besteht: alle Personen, die das Trainingsset aufgenommen haben, sind Rechtshänder. Der Daumenballen der haltenden rechten Hand stellt (in bequemer Handhaltung) den höchsten Druckpunkt auf das Gerät dar, was dieses leicht nach links kippen lässt.

Auf dem Testdatensatz ergeben sich ähnliche Werte wie auf dem Trainingsdatensatz. Die Ausprägung der X-Komponente zum Gestenstart ist nur marginal ausgeprägt und weist eine leichte Neigung nach rechts zum Gestenende auf. Das Gerät ist stärker auf der Z-Achse gedreht, sodass die Y-Komponente dominanter ausgeprägt ist. Interessant diesbezüglich ist, dass der Y-Anteil in beiden Datensätzen zum Ende einer Geste abnimmt, was ergonomisch begründet ist: Die meisten Gesten enden räumlich gesehen unterhalb ihrer Starthöhe. Tabelle 4 stellt die Durchschnittsbeschleunigung auf dem Testdatensatz dar.

	Beschleunigung in X	Beschleunigung in Y	Beschleunigung in Z
Starthaltung	0,027g	-0,78g	-0,51g
Endhaltung	0,13g	-0,51g	-0,69g

Tabelle 4. Mittelwerte aller Start- und Endbeschleunigungen auf dem Testdatensatz

3.1.3 Gestenlänge

Die durchschnittliche Länge einer Geste im Trainingsdatensatz beträgt 153 Elemente. Die kürzeste Geste ist mit 56 Elementen nur knapp länger als eine halbe Sekunde, die längste Geste besitzt eine Länge von 359 Elementen, was $\approx 3{,}5$ Sekunden entspricht. Überraschenderweise sind die kürzeste und die längste Geste nicht in der Menge der negativen Datensätze erhalten, sie werden korrekt erkannt. Alle negativen Datensätze besitzen eine

Durchschnittslänge von 170 Elementen und sind somit ein wenig länger als der Durchschnitt über alle Datensätze, liegen jedoch gleich verteilt um die Durchschnittslänge und zeigen keine besondere Ausprägung oder Anhäufung bei bestimmten Werten. Die positiven Datensätze besitzen eine Durchschnittslänge von 151 Elemente.

Obwohl der Testdatensatz von einer anderen Benutzergruppe aufgenommen wurde, gibt es erstaunlich viele Übereinstimmungen zu dem Trainingsdatensatz. So beträgt die durchschnittliche Länge 160 Elemente. Mit 47 Elementen ist der kürzeste Datensatz keine halbe Sekunde lang. Der längste Datensatz bemisst sich auf 349 Elemente. Auch in der Verteilung von positiven und negativen Datensätzen tritt keine Besonderheit hervor. Ebenfalls sind die positiven Datensätze mit 148 Elementen etwas kürzer als der Durchschnitt und die negativen, die mit 174 Elementen ebenfalls etwas länger sind.

Die Ergebnisse lassen darauf schließen, dass die Amplituden und Längennormierung den Einfluss der eigentlichen Gestenlänge massiv zu unterdrücken vermag. Dennoch fallen die negativen Datensätze in der Regel länger aus als der Durchschnitt. Die Notwendigkeit einer vordefinierten Mindestlänge, wie sie in anderen Arbeiten vorhanden ist (zum Beispiel in [12]), scheint nicht gegeben zu sein. Der Einfluss der Unterabtastung wird in dieser Arbeit noch bestimmt, da die Reduktion der Vektorlänge nach der Längennormierung durch eine Mittelwertbildung durchgeführt wird, was zu einer Verletzung des Nyquist-Theorems führt.

3.1.4 Start- und Endimpuls

Bei der Betrachtung des ersten und letzten Elementes aus dem Trainingsdatensatz fällt auf, dass einige Datensätze eine Initial- oder Endbeschleunigung besitzen, die deutlich von dem zu erwartenden Wert von $\approx$ 1g abweicht. Abbildung 4 stellt den euklidischen Betrag w der Beschleunigung der halben Summe des ersten und letzten Elementes pro Datensatz v mit der Länge N dar.

$$w = \frac{\left(v_{x(1)}^2 + v_{y(1)}^2 + v_{z(1)}^2\right)^{\frac{1}{2}} + \left(v_{x(N)}^2 + v_{y(N)}^2 + v_{z(N)}^2\right)^{\frac{1}{2}}}{2} \tag{14}$$

Nach Bestimmung der Beschleunigung wurde der Datensatz jeweils aufsteigend sortiert. Während das Mittel aller Datensätze nahe dem Betrag von 1g liegt, gibt es deutliche Abweichler nach beiden Seiten. Auch ist gut zu erkennen, dass sich die Verteilung der

negativen Datensätze in den Beschleunigungsbeträgen so wie die Gesamtheit der Datensätze verhält. Um auszuschließen, dass die negativen Datensätze die Beschleunigungen in dem Gesamtbild sind, wurde ebenfalls eine Betrachtung der positiven Datensätze durchgeführt, welche die Gleichverteilung bestätigt hat.

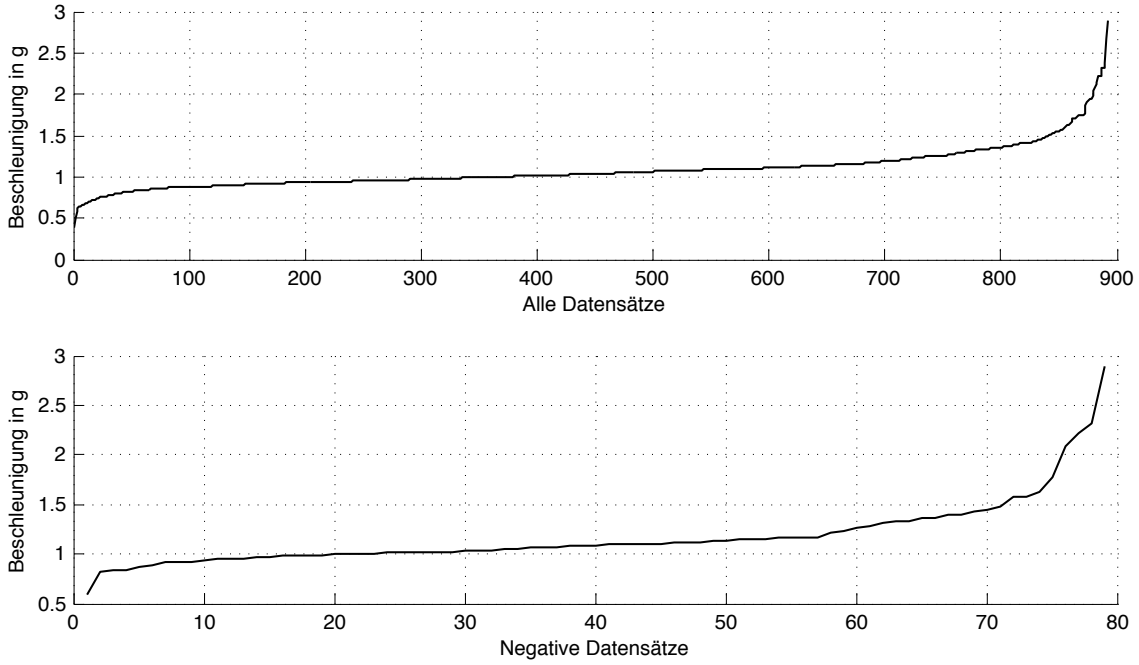

Abbildung 4. Beträge der Start- und Endbeschleunigungbeschleunigung, sortiert über alle Datensätze im Trainingsdatensatz

Für die Auffälligkeiten in den Start- und Endbeschleunigungen kann es folgende Ursachen geben:

- **Versetztes Auslösen** Eine Ursache für diese Auffälligkeit kann sein, dass die Personen den Start der Geste erst nach Beginn der Durchführung markiert haben beziehungsweise vorzeitig die Aufnahme beendet haben, obwohl die Geste noch durchgeführt wurde. Letzteres konnte durch Beobachtungen beim Aufzeichnen des Testdatensatzes bestätigt werden. Der Effekt des vorzeitigen Beendens einer Geste wird im Abschnitt **Bewegungsabruch und Ausklingen** im Detail betrachtet.

- **Hammer-On** beschreibt eine Spieltechnik von Saiteninstrumenten, die von Gitarren- oder Bassspielern verwendet wird. Hierbei wird die Saite mit einem Fingerschlag der Greifhand zum Schwingen gebracht. Eine Vermutung ist, dass Probanden zu Beginn der Geste den Daumen sehr schwungvoll auf das Gerät zum Aufzeichnen

gelegt haben und dieser Impuls einen kurzen aber starken Ausschlag, insbesondere auf der Z-Achse, verursacht hat. Abstrakter betrachtet ist auch das impulsartige Beginnen einer Geste eine Form des Hammer-On, man kann es als Reißen des Gerätes ansehen, welches insbesondere auf der Y-Achse zu starken Amplituden führt. Bei der Einzelbetrachtung sind auch exakt diese Phänomene sichtbar. Der X-Anteil besitzt nur in 5% der Fälle eine Amplitude über 1g, welcher sich vermutlich auf die Durchführung der Z-Gesten zurückführen lässt.

- **Sensoreigenschaften** An dieser Stelle muss auch der Beschleunigungssensor betrachtet werden. Alle Gesten wurden mit einem iPhone 4 aufgezeichnet, dessen Beschleunigungssensor *LIS331*[19] von STMicroelectronics hergestellt wird. Neben kleinen Abweichungen in der Eichung oder Präzisionsschwankungen bei Temperaturveränderungen kann eine Trägheit des Sensors nicht ausgeschlossen werden, die sich leider aus dem Datenblatt nicht beantworten lässt.

3.2 g-Vektor

Unter der Annahme, dass eine Geste in einem ruhigen Moment des Gerätes gestartet wird, stellen die ersten Elemente eines Datensatzes die Orientierung des Gerätes im Raum dar. Die Werte des Beschleunigungssensors in X, Y und Z Richtung werden in diesem Fall ausschließlich von der Erdbeschleunigung bestimmt. Die Lage des Koordinatensystems des Gerätes ist wie in Abbildung 5 dargestellt. In diesem Abschnitt soll untersucht werden, ob es möglich ist, Rückschlüsse über die Lage des Gerätes im Raum zu treffen und diese für weitere Betrachtungen zu benutzen.

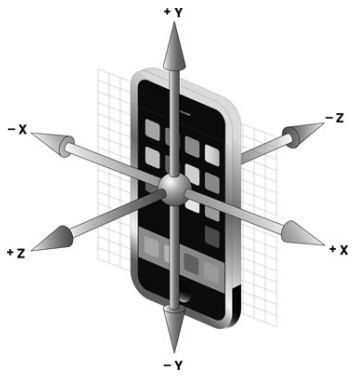

Abbildung 5. Achsenausrichtung des iPhones

Wie bereits im Abschnitt *Start- und Endhaltung des Gerätes* beschrieben, beträgt die

Starthaltung aller Gesten im Trainingsdatensatz im Mittel -0,098g auf der X-Achse, -0,62g auf der Y-Achse und -0,65g auf der Z-Achse. Im Folgenden wird angenommen, dass diese Werte eine Form der „Idealstartposition“ für das Aufzeichnen einer Geste sind. Abbildung 6 zeigt die Start- und Endbeschleunigungen der Kreisgeste. Bei den positiven Datensätzen wurde die Pfeilspitze zu Gunsten der Übersichtlichkeit weggelassen. Es ist zu erkennen, dass die negativen Datensätze relativ gleichverteilt vorkommen und sich keine dominante Anordnung oder Richtung ergibt. Bei den anderen fünf Gesten ist der Initialwinkel teilweise ein anderer, jedoch ist bei keiner Geste eine auffällige Verteilung der positiven oder negativen Datensätze erkennbar.

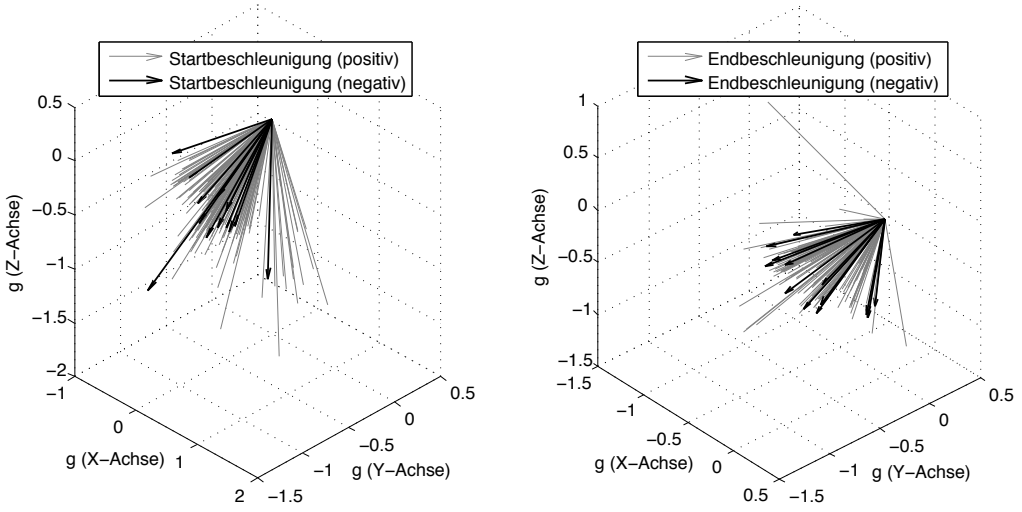

Abbildung 6. Start- und Endbeschleunigungen der Kreisgeste

Es wird angenommen, dass ein Benutzer das Gerät während der gesamten Geste in ungefähr der gleichen Lage belässt und sich die Bewegung durch ihre Langsamkeit nur als ein störendes Rauschen auf diese Lage äußert. Darauf wurden die Datensätze genauer in ihrer Orientierung untersucht. Hierzu wurden alle Datensätze des Trainingsdatensatzes, nach der Vorverarbeitung, in ihrer Amplitude untersucht. Der euklidische Abstand der X-, Y- und Z-Komponente des ersten Elementes und von allen Elementen wurde ermittelt. Wie in Abbildung 7 zu sehen ist, gibt es mitunter intensive Schwankungen innerhalb des ersten Elementes, im Mittel schwanken die meisten Datensätze jedoch um Wert 1g. Das Mittel vom Mittelwert aller Datensätze beträgt 0,964g. Wenn nur die negativen Datensätze betrachtet werden, liegt dieser Mittelwert bei 0,965g. Die positiven Datensätze besitzen einen Mittelwert von 0,963g. Der Mittelwert aller ersten Elemente beträgt 1,19g, nur die negativen Datensätze erreichen einen Betrag von 1,32g. Da die Maxima nicht in

die negativen Datensätze fallen, kann an dieser Stelle nicht von einem Einfluss gesprochen werden.

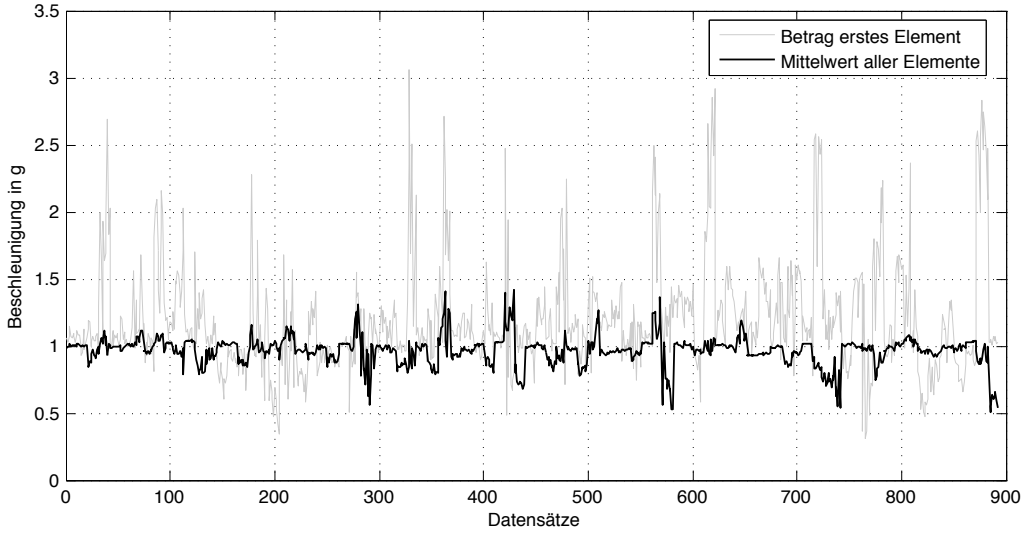

Abbildung 7. Betrag der Beschleunigung des ersten und und aller Elemente des Trainingsdatensatzes

3.3 Spektrale Ähnlichkeit

Der Grundgedanke zur Betrachtung der spektralen Ähnlichkeit war, herauszufinden, ob es eine Korrelation zwischen den Spektren einer Klasse gibt. Wenn es diese gibt, existiert eine Korrelation zwischen den Spektren der Datensätze, die besonders gut erkannt werden. Wenn weiterhin Auffälligkeiten in den Spektren von Datensätzen zu erkennen sind, die eher als negativ erkannt werden, lassen diese Rückschlüsse auf noch näher zu bestimmende Eigenschaften zu.

Um diese zu bestimmen, wird auf das Konzept der spektralen Ähnlichkeit zurückgegriffen, wie es zum Beispiel [20] vorgestellt hat. Einen weiteren sehr interessanten Ansatz haben [21] vorgestellt, in dem eine spektrale Betrachtung einer nicht kontinuierlichen Zeitreihe vorgestellt wird.

Die Idee von [20] ist, dass es eine Korrelation zwischen den Spektren ähnlicher Signale gibt. Um diese zu bestimmen, ist das Amplitudenspektrum aller vorverarbeiteten Datensätze zu bilden und im Anschluss mit einer DTW (Dynamic Time Warp) zu analy-

sieren. Als erster Schritt wurde auf die allgemeine Erkennungsleistung der SFA eingegangen, wenn diese auf die Spektren des Trainingsdatensatzes angewendet wird. Von jedem Datensatz wurden die Spektren der Rohdaten mittels FFT gebildet. Durch die hermitschen Eigenschaften des Spektrums der Rohdaten wurde nach der FFT die erste Hälfte der Frequenzkanäle für die neuen Eingangsvektoren verwendet. So konnte der gesamte Vorverarbeitungsprozess im Anschluss ohne Änderungen fortgesetzt werden.

Die Erkennungswahrscheinlichkeit auf dem Trainingsdatensatz liegt auf der Trainingsmenge bei 83,6% und auf der Testmenge 73,4%. Tabelle 5 stellt die Erkennungsergebnisse des Trainingsdatensatzes dar. Die Erkennungswahrscheinlichkeit auf dem Testdatensatz beträgt 38%.

	G1	G2	G3	G4	G5	G6
G1	93	8	7	2	18	17
G2	9	114	6	9	3	7
G3	1	11	117	3	6	5
G4	9	5	2	116	3	6
G5	19	2	6	4	121	21
G6	12	12	6	8	10	94

Tabelle 5. Die von MATLAB bestimmte Konfusionsmatrix des Trainingsdatensatzes nach der Vorverarbeitung mittels FFT

Es zeigt sich, dass im benutzerabhängigen Fall die Eigenschaften der Spektren eine moderate Erkennungsleistung bieten, im benutzerunabhängigen Fall stark sinken. Ursache hierfür kann die später noch vorgestellte *Ruhe* und das *Ausklingen* sein. Die inverse Kovarianzmatrix besitzt im Mittel nur einen Wert von $\approx 2{,}16$, was dafür spricht, dass die Spektren zwar charakteristische Informationen in sich tragen, diese aber durch die spektrale Betrachtung nur schwach ausgeprägt zur Geltung kommen.

Während es auf dem Trainingsdatensatz ohne jede Modifikation zu 82 Fehlerkennungen kommt, erreicht eine Vorverarbeitung mit FFT 146 Fehlerkennungen. Interessant ist hier, dass nur 35 Datensätze die Schnittmenge der als negativ erkannten Datensätzen bilden. Durch die FFT wird ein Perspektivwechsel auf die Eingangsdaten durchgeführt, der dazu führt, dass andere Elemente Einfluss auf die Erkennung nehmen. Um diese Einflüsse erfassbar zu machen, muss das Spektrum nach seinen Eigenschaften analysiert werden.

Die Analyse der Spektren mit Hilfe der DTW entspringt dem Sachverhalt, dass Spektren mitunter eine hohe Aussagekraft besitzen, aber nur schwer untereinander vergleichbar sind. Um Konkatenationartefakte auszuschließen, welche nach der Fouriertransformation nicht mehr als Störung identifizierbar, aber vorhanden wären, wurden die Beschleunigungsvektoren mit einem anderen Verfahren konkateniert als sonst. In diesem Fall wurde der gesamte Vektor der Y-Komponente durch eine Verschiebung direkt an das letzte Element der X-Komponente angefügt, die Z-Komponente wurde an das letzte Element der Y-Komponente geschoben. Durch diese Harmonisierung beinhaltet der vorverarbeitete Eingangsvektor keine Sprungstellen und durch die Amplitudennormierung werden auch größere Schwankungen in den Spektren innerhalb einer Klasse reduziert. Abbildung 8 zeigt jeweils drei Spektren der Kreis- und der X-Geste. Die Gesten *Kreis 1* und *Kreis 2* sind positiv, *Kreis 3* ist negativ, gleiches gilt für die X-Geste. Während sich für die Kreis-

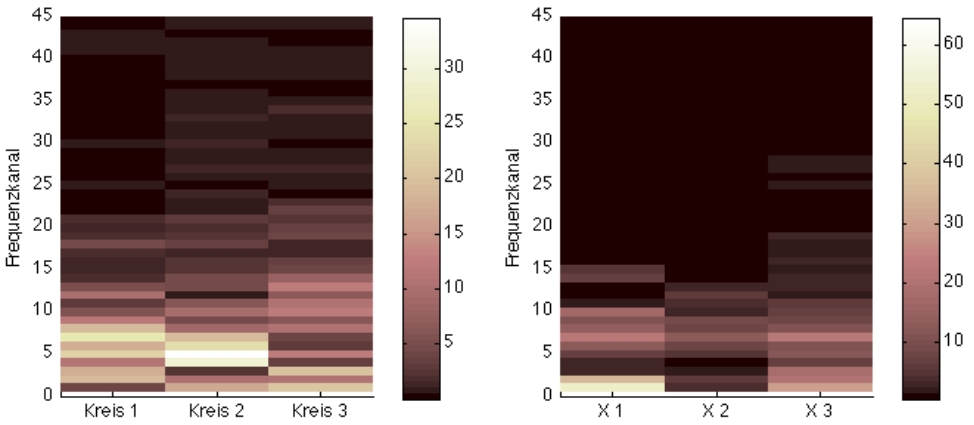

Abbildung 8. Spektrum von drei Kreis- und drei X-Gesten, jeweils die dritte ist negativ

gesten keine offensichtliche Struktur in den Frequenzanteilen offenbart, wirkt es bei den X-Gesten als sei *X 2* vom Betrag der Schatten von *X 1*. *X 3* verhält sich bis zum 10. Kanal recht gleichmäßig in der Amplitude, was die beiden anderen X-Gesten nicht sind. Weiterhin besitzt *X 3* noch ein paar Nachschwinger kurz vor dem 30. Kanal, über dessen Einfluss nur spekuliert werden kann. Diese Einschätzung besitzt leider deutliche Probleme:

- **Kanalbreite**:
  Die Länge des zu analysierenden Signals beträgt nur 90 Elemente. Um eine Vergleichbarkeit zwischen den Spektren herzustellen, sollte nicht das vorverarbeitete

Eingangssignal betrachtet werden. Die rohen Eingangsdaten variieren wie bereits dargestellt stark in ihrer Länge.

- **Bewertung**:
  Die Einschätzung wurde von Hand erstellt. Zwar wirkt *X 2* recht ähnlich zu *X 1*, bei den Kreisgesten ist diese Ähnlichkeit nicht oder nicht offensichtlich gegeben.

Zur automatischen Auswertung kann wie bereits beschrieben die DTW angewendet werden. Für MATLAB gibt es DTW-Toolboxen [22][23], die diese Funktionalität zur Verfügung stellen. Die Korrelationsbestimmung der Spektren zwischen den Datensätzen kann in dieser Arbeit nicht betrachtet werden.

3.4 Ausgleichsrechnungen

Mit der bestimmten „Idealstartposition“ wurde versucht, eine Translation von dem jeweiligen Datensatz zu der Idealstartposition hin durchzuführen. Diese Translation war der Start einer kleinen Reihe von Versuchen, die jeweils mit den Beträgen der jeweiligen Beschleunigungskomponente experimentiert haben. Jeder Ausgleich wird anhand seiner Erkennungsquote in der Anwendung und anhand der Konfusionsmatrix bewertet.

Die Anwendung des Trainingssets auf dem von MATLAB generierten Modell erzielt eine Erkennungsquote von 91,1% auf der Trainingsmenge und 85,4% auf der Testmenge, mit der Konfusionsmatrix 6. Auf dem Testdatensatz wird eine Erkennungswahrscheinlichkeit von 56,2% bestimmt, mit der Konfusionsmatrix 7. Die relativ gute Erkennung der Geste G3 (*Z*) und G4 (*Epsilon*) bleibt in beiden Konfusionsmatrizen erhalten, auch die hohen Verwechslungsquoten der Geste G1 (*Kreis*) zu der Geste G5 (*Herz*) sowie der Geste G6 (*Unendlichkeit*) zu der Geste G2 (*X*) bleiben bestehen. Interessant ist die Verwechslungsquote von Geste G5 (*Herz*). Während G5 auf dem Trainingsdatenzsatz eine relativ hohe Verwechslungsquote mit G1 aufweist, verlagert sich die Verwechslung auf dem Testdatensatz fast vollständig zu den beiden Gesten G2 und G6.

3.4.1 Translation zur Idealstartposition

Als einfachste Methode, die Varianz in der Startposition x_s, y_s, z_s zu kompensieren, wurde eine Translation zur Idealstartposition durchgeführt. Bei dieser wurde jedes Element des eines Datensatzes zu seiner Differenz zu der Startposition betrachtet und diese von jedem

	G1	G2	G3	G4	G5	G6
G1	114	4	1	0	17	3
G2	10	124	5	6	7	12
G3	2	10	133	0	2	0
G4	1	1	5	131	2	2
G5	13	5	0	3	130	3
G6	3	8	0	2	3	130

Tabelle 6. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 von dem Trainingsdatensatz

	G1	G2	G3	G4	G5	G6
G1	23	9	3	2	3	1
G2	3	22	1	2	10	9
G3	1	3	24	4	1	0
G4	2	9	4	28	0	0
G5	12	2	0	0	22	1
G6	4	3	1	2	16	20

Tabelle 7. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testdatensatz

Element des Datensatzes s subtrahiert.

$$x_d = s_{x(1)} - x_s \tag{15}$$

$$y_d = s_{y(1)} - y_s \tag{16}$$

$$z_d = s_{z(1)} - z_s \tag{17}$$

Die Werte x_s, y_s, z_s besitzen nun den Abstand der Startposition einer Geste zu dem Mittel aller Gestenstartpositionen. Vom Datensatz wird nun diese Abweichung entfernt.

$$s'_{k(n)} = s_{k(n)} - k_d;\ k = \{x, y, z\} \forall n = 1...N \tag{18}$$

Nach Durchführung dieses einfachen Differenzausgleichs erhört sich die Erkennungsquote von 91,1% auf 91,9% auf der Trainingsmenge und von 85,4% auf 85,5% auf der Testmenge. Bei Betrachtung der Konfusionsmatrix 8 fällt auf, dass die Erkennungswahrscheinlichkeit der Kreisgeste verbessert wurde, jedoch die Epsilongeste unter diesem Ausgleich leidet. Da beide hohe Fehler in der jeweiligen Klasse besitzen, entsteht die Vermutung, dass der Initialwinkel für diese beiden Klassen besonders wichtig ist.

	G1	G2	G3	G4	G5	G6
G1	120	6	0	0	21	2
G2	7	127	4	5	11	8
G3	1	5	132	2	1	3
G4	1	3	5	131	1	1
G5	13	8	2	3	124	7
G6	1	3	1	1	3	129

Tabelle 8. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach einer Translation zur Idealstartposition

Auf dem Testdatensatz erhöht sich die Erkennungsquote von 56,2% auf 60,7%, wobei alle Gesten bis auf die Kreisgeste (G1) von der Translation profitieren, was im Gegensatz zu dem Trainingsdatensatz steht. Die Gefahr von Geste G1 mit G5 „verwechselt“ zu werden, beträgt fast 50%.

	G1	G2	G3	G4	G5	G6
G1	20	7	1	0	2	1
G2	2	26	1	1	11	6
G3	0	0	24	3	1	1
G4	5	11	1	32	0	0
G5	16	3	6	1	27	2
G6	2	1	0	1	11	21

Tabelle 9. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testdatensatz nach einer Translation zur Idealstartposition

3.4.2 Translation zu 0

Bei der Betrachtung der einzelnen Datensätze ist die Frage nach dem Einfluss der Achsenkonkatenation aufgekommen. Der Initialwert der Beschleunigung auf der jeweilen Achse führt auch nach der Vorverarbeitung zu einem Sprung im konkatenierten Signal. Da die Amplitude der Signalsprünge nicht von der Natur der Bewegung abhängen, sondern von dem Winkel mit dem das Gerät gehalten wird, entstand die Vermutung, dass eine Reduktion oder sogar eine Beseitigung dieser Konkatenierungsartefakt einen Einfluss auf die Erkennungswahrscheinlichkeit besitzen könnte. Um diesen Einfluss zu bestimmbar

zu machen und gegebenenfalls zu beseitigen, wurde eine Translation zu 0 durchgeführt.

Zur Tilgung der Differenz wurde von jedem Element aus den Beschleunigungen der X-, Y- und Z-Achse pro Datensatz d das erste Element pro Beschleunigungsanteil vom gesamten Datensatz abgezogen.

$$\begin{aligned} d'_x(n) &= d_x(n) - d_x(1) \\ d'_y(n) &= d_y(n) - d_y(1) \\ d'_z(n) &= d_z(n) - d_z(1) \end{aligned} \tag{19}$$

Nach dem Angleichen der Vektoren ergibt sich folgende Konfusionsmatrix auf der Trainingsmenge:

	G1	G2	G3	G4	G5	G6
G1	128	5	0	0	12	2
G2	5	130	8	7	5	8
G3	1	5	129	0	2	2
G4	2	1	6	130	1	1
G5	7	6	1	3	139	6
G6	0	5	0	2	2	131

Tabelle 10. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach der Translation zu 0

	G1	G2	G3	G4	G5	G6
G1	28	4	0	1	3	1
G2	3	25	7	1	6	8
G3	0	0	22	3	0	0
G4	3	17	3	30	2	0
G5	10	2	1	1	29	1
G6	1	0	0	2	12	21

Tabelle 11. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testset nach einer Translation zu 0

Es zeigt sich, dass der Ausgleich die Erkennungswahrscheinlichkeit auf der Trainingsmenge von 91,1% auf 93,4%, auf der Testmenge von 85,4% auf 88,2% erhöht. Auf dem Testset erhöht sich die Erkennungswahrscheinlichkeit von 56,2% auf 62,7%, wobei alle Gesten leicht in ihrer Erkennung angeglichen werden. Keine der Gesten sticht durch eine besonders hohe oder niedrige Erkennungsquote heraus. Dieser positive Einfluss lässt

die Vermutung entstehen, dass das Artefakt, das beim Konkatenieren der Achsenkomponenten entsteht, von der SFA als Bestandteil der Geste gewertet wird. Diese Vermutung wurde jedoch falsifiziert!

Die SFA nimmt vom analysieren Signal nicht an, dass dieses als Form eines Amplituden-Zeit-Signal vorliegt, also eine harmonische Form nicht vorausgesetzt wird. Woher kommt dann dieser Effekt?

Wenn das Konkatenierungsartefakt einen Einfluss auf die Erkennungswahrscheinlichkeit besitzt, dann sollte der Trainingsdatensatz, durch das Einbringen zahlreicher weiterer Artefakte, in seiner Erkennungswahrscheinlichkeit sinken. Um dieses zu testen, wurde das Muster einer zufälligen Permutation der vorverarbeiteten Beschleunigungsvektors erstellt. Für das Training und die Anwendung wurde jeweils die gleiche Permutation verwendet. Durch diese wurde jede Komponente zufällig, aber über alle Datensätze gleich verrauscht, was zu einem Signal führte, welches fast nur noch aus (hohen) Amplitudensprüngen bestand (Abbildung 9). Nach der Anwendung ergibt sich exakt das gleiche Ergebnis in der Erkennung, als wäre die Modifikation nicht durchgeführt worden. Die Verbesserung in der Erkennung erfolgte demnach nicht aufgrund der Reduktion des Konkatenierungsartefaktes.

Der Angleich der Initialwerte aller drei Beschleunigungsvektoren zu 0 entspricht einer Homogenisierung der Starthaltung des Gerätes. Durch diesen Ausgleich wurde der Winkel, mit dem das Gerät im Raum gehalten wurde oder auch die initiale Beschleunigungen, nicht mehr als Merkmal der Geste gewertet.

3.4.3 Zufälliges Verrauschen

Im vorigen Abschnitt kam die Vermutung auf, dass das Konkatenierungsartefakt einen Einfluss auf die Erkennung besitzt. Durch die Verschiebung tritt eine Verbesserung in der Erkennungswahrscheinlichkeit ein. Da die SFA jedoch nach dem treibenden Prozess in einem Eingabevektor sucht, der kein Amplituden-Zeit-Signal sein muss, überrascht dieses Verhalten.

Um dieses zu überprüfen, wurde ein absichtliches, zufälliges Umsortieren der vorverarbeiteten Eingabevektoren durchgeführt. Da für das Training und die Anwendung die gleiche zufällige Verteilung vorherschen muss, wurden zuerst drei Indexvektoren mit dem

MATLAB Befehl *randperm*[5] erzeugt. Im Anschluss wurde jeweils der Eingabevektor, der zur X-, Y- und Z-Achse gehört, durch dieses Indexvektoren „sortiert“. Abbildung 9 zeigt beispielhaft einen Datensatz vor und nach der Umsortierung. Zu erkennen ist, dass die Umsortierung starke Schwankungen verursacht, die nur noch entfernt an die Original Eingabevektoren erinnern. Das zufällige Sortieren, welches zu einem verrauschten Einga-

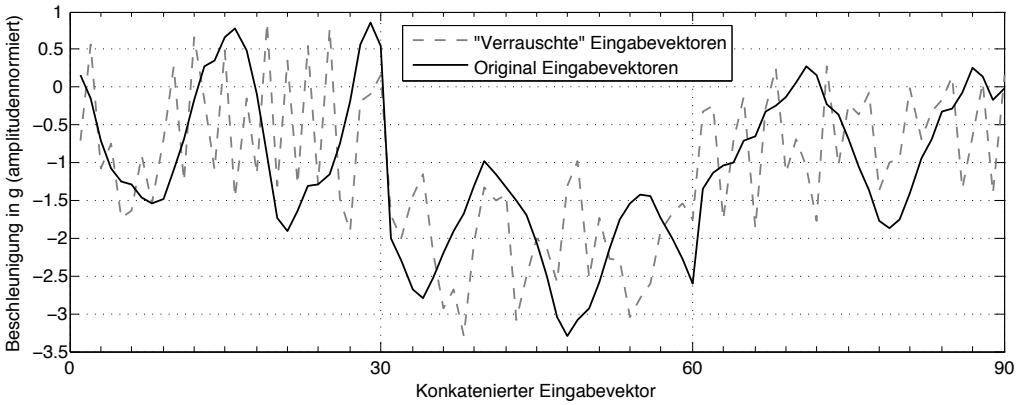

Abbildung 9. Die konkatenierten Eingabevektoren der X-, Y- und Z-Komponente, durch eine zufällige Sortierung verrauscht und Original

bevektor mit zahlreichen Sprungstellen führt, die den Konkatenierungsartefakten ähneln, führt zu keiner Verschlechterung in der Erkennungswahrscheinlichkeit. Diese bleibt davon unberührt. Durch die Eindeutigkeit des Ergebnisses ist die Vermutung, dass das Konkatenierungsartefakt einen Einfluss besitzt, falsifiziert. Die Translation der Komponentenvektoren zu 0 führt vielmehr zu einer Homogenisierung des Eingaberaumes, kleinere Winkeländerung zu Beginn der Geste werden so entfernt. Dem Eingaberaum für die SFA wird somit teilweise die Varianz des Eingangswinkel entzogen.

[5] http://www.mathworks.de/help/techdoc/ref/randperm.html Abruf: Februar 2012

4 Systemeinfluss

An dieser Stelle sollen verschiedene Eigenschaften des erzeugten Modells und deren Einfluss auf die Erkennung beleuchtet werden. Das Modell einer SFA steht als Ergebnis eines „One-Shot-Learnings“ zur Verfügung und kann nach diesem Lernprozess verwendet werden. Zur Anwendung der SFA auf einen neuen Eingabevektoren gehört die Whiteningmatix, die während des Trainings ermittelten Eigenvektoren, sowie Vektoren, die die Mittelwerte der Gestendaten beinhalten.

Weiterhin sollen in diesem Abschnitt verschiedene Punkte aufgeführt werden, die Teile des Erkennungsprozesses sind und Einfluss auf die Erkennungsleistung besitzen. Beginnend mit der Vorverarbeitung bis hin zur Klassifikation werden verschiedene Teilschritte betrachtet und deren Einfluss bestimmt. Jeder Teilschritt ist als Teil des Systems von Bedeutung und sollte bestimmt werden, um dem Ziel einer idealen Erkennung schrittweise näher zu kommen.

Als Erstes soll auf den Trainingszustand des Modells eingegangen werden.

4.1 Trainingszustand

Das für die Anwendung verwendete Modell wurde in einem „Cross Validation“ Verfahren erzeugt. Bei diesem wird ein gewisser Teil des Datensatz einer Trainingsmenge zugeordnet, während der verbleibende Teil als Testmenge verwendet wird. Um alle Datensätze des Datensatzes für die Modellerzeugung verwenden zu können, wird dieser Prozess wiederholt, wobei darauf geachtet wird, dass es keine Überschneidungen der Testmengen gibt. Abbildung 10 zeigt die Erkennungswahrscheinlichkeiten im Mittel für unterschiedliche Einteilungen der Cross Validation. Pro Segmentwahl wurden fünf Durchläufe gestartet, wobei die Initialindexierung zur Einteilung von Trainings- und Testmenge bei jedem Durchlauf per Zufall gewählt wurde. Zu der Bestimmung der Erkennungswahrscheinlichkeit wurde der gesamte Datensatz aus Testdaten und Trainingsdaten verwendet. Die Einteilungen wurde bei zwei gestartet, was bedeutet, dass der Datensatz geteilt wird (Verhältnis 1:1). Eine Hälfte wird zum Training des Modells benutzt, während die andere Hälfte zum Testen verwendet wird. Der gleiche Schritt erfolgt mit „vertauschten Rollen“, sodass alle Datensätze getestet werden. Diese Teilung wurde bis 200, was einem Trainings-Test-Verhältnis von 199:1 entspricht, durchgeführt. Es fällt auf, dass der Fehler bei einer Einteilung der Cross Validation von neun auf der Testmenge ein deutli-

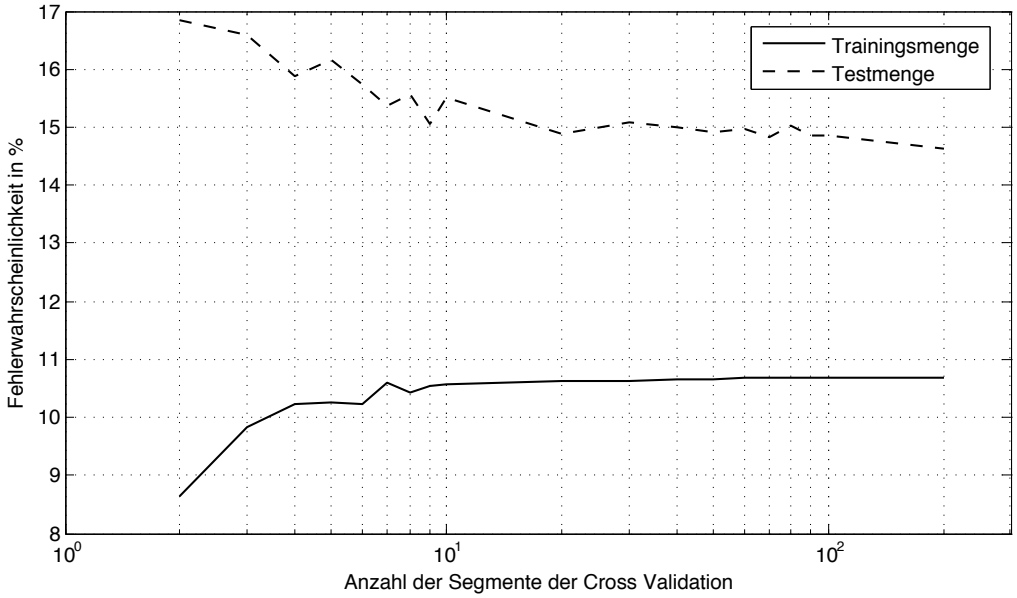

Abbildung 10. Fehlerquoten der Trainings- und Testmenge für unterschiedliche Cross Validation Einteilungen auf allen Datensätzen

ches lokales Minimum erfährt und bei feineren Einteilungen stagniert. Diese Berechnung wurde zuvor auf dem Trainingsdatensatz durchgeführt, wobei der Verlauf des Graphen bis auf eine Verschiebung übereinstimmt. Das lokale Minimum, welches auf den kombinierten Datensätzen bestimmt wurde, liegt bei neun, das lokale Minimum nur auf dem Trainingsdatensatz liegt bei sieben. Wird die Anzahl der Datensätze aus der jeweiligen Trainingsmenge durch die Anzahl der jeweiligen Testmenge dividiert, erhält man jeweils den Betrag 127. Durch Einflüsse in der Klassifkation, die später betrachtet werden, ist dieser Effekt zu erklären.

4.2 Klassifikation

Wie bereits im Abschnitt Feineinteilung beschrieben, erfolgt die Klassifikation durch den Gauss-Klassifikator sehr entschlossen. Diese Entschlossenheit entspringt der Tatsache, dass die Klassifikation alle zu klassifizierenden Datensätze während des Trainings kennengelernt hat. Sie sinkt auf dem Testset deutlich. Ursache hierfür ist, dass der Gauss-Klassifikator als diskriminativer Klassifikator mit vollständig klassifizierten Trainingsdaten trainiert wurde. Er versucht die Geste als Ganzes zu klassifizieren, indem er sie global betrachtet. Dieser Vorgang hat Vorteile, dass das Training relativ leicht durchzuführen ist

sowie Training und Anwendung mit geringem Rechenaufwand durchführbar sind. Sobald die Daten, die er klassifizieren soll, von den Trainingsdaten abweichen, hängt die Qualität der Erkennung von der Robustheit des Klassifikators ab. Ist der Klassifikator über- oder unterangepasst, erfolgt die Erkennung nur noch eingeschränkt. Im Falle des verwendeten Gauss-Klassifikators äußert sich die Entschlossenheit durch die Beträge in der inversen Kovarianzmatrix. Die Angepasstheit des Klassifikators lässt sich in Abbildung 10 erkennen, wenn der Verlauf der Fehlerwahrscheinlichkeit der Testmenge mit dem Verlauf der Mittelwerte der Hauptdiagonalen der inversen Kovarianzmatrix in Abbildung 11 verglichen wird. Die Mittelwerte der Hauptdiagonalen ergeben sich aus der Summe aller Hauptdiagonalen der berechneten $N \times N \times M$ Matrix, dividiert durch die Anzahl der Hauptdiagonalelemente, wobei M die Anzahl der Klassen darstellt. Es ist zu erkennen, dass es eine direkte Korrelation zwischen den Mittelwerten und der Fehlerwahrscheinlichkeit auf der Testmenge gibt. Die Anpassung des Klassifikators steht im Verhältnis zur Testmenge.

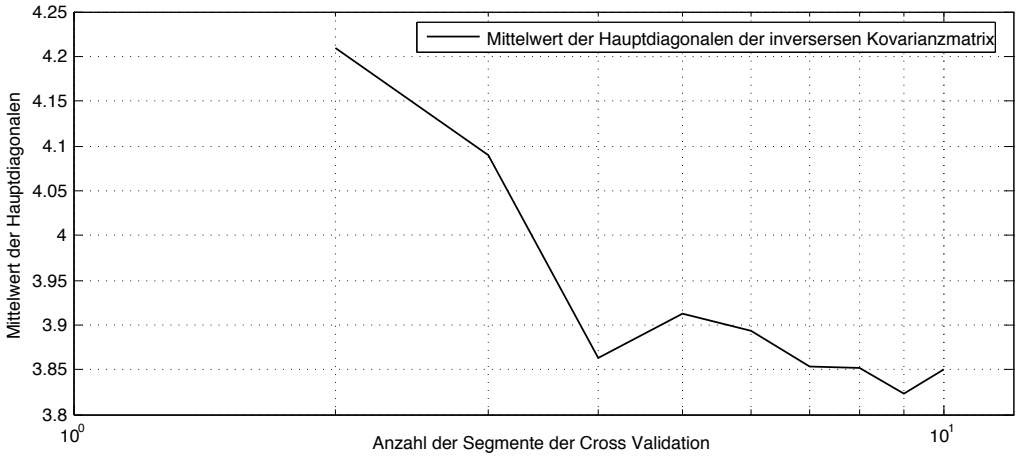

Abbildung 11. Mittelwerte der Hauptdiagonalen der inversen Kovarianzmatrizen für unterschiedliche Cross Validation Einteilungen

Wenn die Beträge der Kovarinzmatrix händisch reduziert werden, tritt ein interessanter Effekt auf. Ab einem gewissen Punkt verliert die Kreisgeste massiv an Erkennungswahrscheinlichkeit während alle anderen Gesten dazugewinnen. Wird der Betrag erhöht, steigt die Kreisgeste in ihrer Erkennungswahrscheinlichkeit wieder auf ihr ursprüngliches Niveau, darüber hinaus treten nur noch kleine Verschiebung in der Erkennung auf. Die einzelnen Beträge, der zu der Geste gehörenden Matrix, zeigen, dass der Mittelwert der

Kreisgeste und der X-Geste durchschnittlich 25% geringer ausfallen als die Mittelwerte der anderen Gesten. Da die X-Geste nicht unter diesem Verfall der Erkennungswahrscheinlichkeit leidet, kann der Betrag in der Kovarianzmatrix nicht alleinig Ursache für dieses Verhalten sein. Durch die Reduktion der Beträge in der Kovarianzmatrix wird die Hyperfläche, die durch die Matrix beschrieben wird, verkleinert. Die Position und Lage der Hyperfläche wird vermutlich diesen Einfluss begründen.

Als Alternative für globale Klassifikatoren bieten sich lokale, generative Klassifikatoren an. Diese sind in der Verarbeitung oftmals sehr aufwändig, besitzen aber den Vorteil lokale Besonderheiten der Eingabevektoren zu beachten. Einen interessanten Ansatz verfolgen Lasserre et al. [24], in dem sie generative und diskriminative Ansätze miteinander kombinieren.

4.2.1 Der „perfekte“ Datensatz

Um die Eigenschaften des Gauss-Klassifikators näher zu bestimmen und um gegebenenfalls weitere Einflüsse und Eigenschaften der SFA zu identifizieren, wurde aus den aufgenommenen Datensätzen der „perfekte“ Datensatz generiert. Wie bereits gezeigt, verhält sich der Gauss-Klassifikator überaus entschlossen in seiner Urteilsbildung. Erkennungswahrscheinlichkeiten, egal ob benutzerabhängig oder benutzerunabhängig, erzielen nicht selten Erkennungswahrscheinlichkeiten von über 90%.

Dieses kann auf verschiedene Einflüsse hinweisen. Die SFA konnte die Essenz der Bewegung so gut selektieren, dass die zu klassifizierenden Vektoren eine so hohe Erkennungswahrscheinlichkeit erlauben. Welcher Teilschritt diesbezüglich am einflussreichsten ist, gilt es noch zu bestimmen. Der Klassifikator besitzt immer einen elementaren Einfluss auf die Erkennung, der beste Klassifikator kann aus unbrauchbaren Daten keine Information ziehen. Leider gilt das auch für verwertbare Informationen, die mit einem unbrauchbaren Klassifikator betrachtet werden.

Was passiert, wenn die SFA ausschließlich gutmütige Datensätze zum Training erhält? Was, wenn ein quasi perfekter Datensatz verwendet wird? Wie wirkt sich das auf die Erkennung des Testdatensatzes aus?

Um diese Fragen zu beleuchten, wurden zwei Teilmengen des Trainingsdatensatzes gebildet. Eine Menge besteht ausschließlich aus den positiven Datensätzen, die zweite

nur aus Datensätzen, die mit mindestens 99% Wahrscheinlichkeit erkannt wurden. Der Grundgedanke des ersten Teildatensatz ist, den Einfluss der negativen Elemente in dem Trainingsdatensatz zu bestimmen. Der zweite Teildatensatz, bestehend aus den quasi perfekten Datensätzen, soll zwei Sachverhalte offenbaren: Wenn das Modell nur mit idealen Datensätzen trainiert wurde, wie verhält es sich in Bezug auf den Testdatensatz, der weit entfernt von dem makellosen Teildatensatz ist? Wie verhält sich die Erkennung *an sich*: beschreibt der perfekte Teildatensatz wirklich eine Menge von perfekt aufeinander abgestimmten Datensätzen oder ist *perfekt* nur ein Attribut für schwächer ausgeprägte Mängel?

Zuerst soll der Fall betrachtet werden, dass nur die positiven Datensätze des Trainingsdatensatzes für das Training verwendet werden. In diesem Fall ergibt sich Konfusionsmatrix 12. Auf der Testmenge, die durch die Cross Validation mit dem Segmentindex 10 entstand, ergibt sich eine Erkennungswahrscheinlichkeit von 93,5%, auf der benutzen Trainingsmenge eine Erkennungswahrscheinlichkeit von 98,2%. Interessant ist die Verteilung der Fehlerkennungen. Diese stimmt bis auf kleinere Abweichungen in ihrer Tendenz mit der Verteilung der Fehlerkennungen des vollständigen Trainingsdatensatzes überein. Die Hauptdiagonalen in der inversen Kovarianzmatrix weisen über alle Klassen fast die doppelten Beträge auf.

	G1	G2	G3	G4	G5	G6
G1	113	0	0	0	9	1
G2	1	124	0	1	4	7
G3	1	1	136	0	1	0
G4	0	0	1	134	1	0
G5	11	3	1	0	127	4
G6	1	3	0	1	1	123

Tabelle 12. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 ausschließlich auf den positiven Datensätzen.

Das Training mit der quasi perfekten Trainingsdatensatzmenge erzielt eine Erkennungswahrscheinlichkeit von 98,5% auf der Testmenge. Auf der Trainingsmenge ist die Erkennungswahrscheinlich makellos. In diesem Fall sind die Beträge der Hauptdiagonalen der inversen Kovarianzmatrix über dreimal so groß wie bei dem Training mit allen Datensätzen. Auf dem Testset beträgt die Erkennungswahrscheinlich 56,7%, im Fall, dass

nur die positiven Datensätze zum Training verwendet werden, liegt die Erkennungswahrscheinlichkeit bei 57,5%. Wenn der gesamte Trainingsdatensatz, also positive wie auch negative Datensätze, verwendet wird, ist die Erkennungswahrscheinlichkeit mit 56,3% nur knapp geringer als die Wahrscheinlichkeiten mit ausschließlich positiven Datensätzen. Dieser Sachverhalt deutet an, dass sich zwei Faktoren gegenseitig kompensieren. Durch die gesteigerte Qualität der Trainingsdatensätze steigt die Güte der Erkennung, fehlerhafte Einflüsse werden großteils nicht mehr in die Betrachtung mit einbezogen, dafür wird der Gauss-Klassifikator immer (über)angepasster. Die perfekte Trainingsdatensatzmenge erzeugt Konfusionsmatrix 13. Obwohl nur Datensätze für das Training verwendet wurden, die eine Erkennungswahrscheinlichkeit von über 99% besitzen, gibt es 10 Fehlerkennungen. Das ist eine Auswirkung des überangepassten Gauss-Klassifikators, der bei dieser Trainingsmenge bereits kleinste Abweichungen bestraft.

	G1	G2	G3	G4	G5	G6
G1	85	0	0	0	2	0
G2	0	107	0	1	0	1
G3	0	0	121	0	0	0
G4	0	0	0	122	0	0
G5	0	2	0	3	98	1
G6	0	0	0	0	0	108

Tabelle 13. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 ausschließlich auf den positiven Datensätzen.

Aber was passiert, wenn man das Training mit den negativen Datensätzen durchführt? Die Menge der negativen Datensätze besteht zwar zu Teilen aus defekten Gesten, aber einige sollten doch die Natur der Geste besitzen. Da 82 Datensätze negativ ausfallen, ist diese Menge allein zu gering für ein Training. Durch Bootstrapping wird die Anzahl der Trainingsdatensätze erhöht. Bootstrapping bezeichnet einen Prozess, bei dem eine Eingangsmenge durch modifizierte Kopien ergänzt wird. Diese Kopien können unter anderem verrauscht, gedreht, gezerrt oder gestaucht sein. Für das Training wurde die vierfache Menge an modifizierten Kopien hinzugefügt, die Anwendung erfolgte auf der Menge der negativen Datensätze. In diesem Fall ergibt sich eine Erkennungswahrscheinlichkeit von 40,7%. Wird das Testset auf das so generierte Modell angewendet, so ergibt sich eine Erkennungswahrscheinlichkeit von 11,7%! Wenn davon ausgegangen werden kann, dass alle Gesten ungefähr mit gleicher Häufigkeit in der Fehlermenge vorkommen, liegt die Erkennungsleistung $\approx 5\%$ geringer als zufälliges Raten.

4.2.2 Unterabtastung in der Vorverarbeitung

Im Abschnitt **SFA** wird beschrieben, dass während der Vorverarbeitung eine Längennormierung durchgeführt wird. Die Reduktion des Eingabevektors erfolgt über die Mittelwertbildung von N Elemente. Durch diesen Umstand wird das Abtasttheorem von Nyquist verletzt und der neue Vektor entspricht einem unterabgetasteten Vektor. Dieser Umstand entsteht, da höherfrequente Anteile enthalten sein können. Eine Unterabtastung kann zum *Aliasing* führen [25], die als Artefakte auftreten und durch den Einfluss unterdrückte Anteile im Spektrum entstehen. Um einen potenziellen Einfluss auf die Erkennungsleistung bestimmen zu können, wurde der während der Vorverarbeitung längennormierte Vektor vor der Reduktion einer Tiefpassfilterung zugeführt. Durch einen Tiefpassfilter werden hochfrequente Anteile in einem Signal gedämpft, während niederfrequente Anteile (fast) ungedämpft übernommen werden. Abbildung 12 zeigt einen Datensatz vor und nach der Tiefpassfilterung.

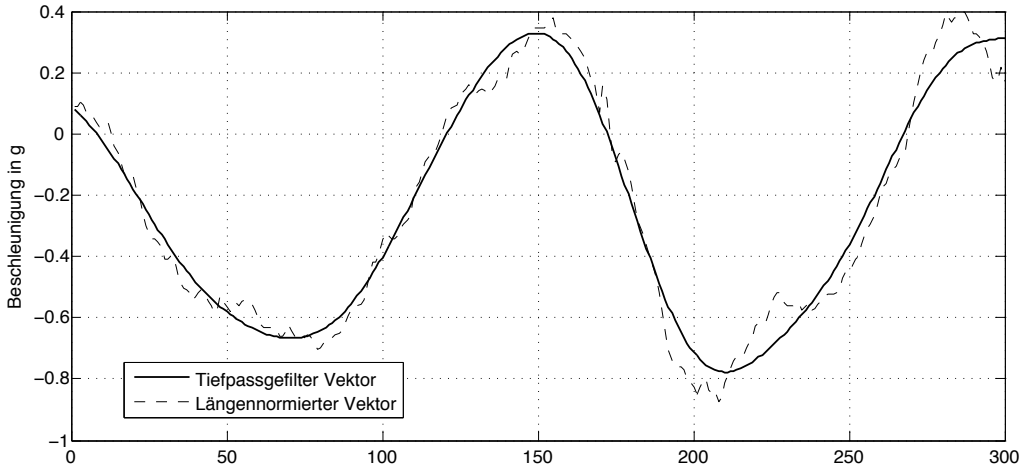

Abbildung 12. Datensatz vor und nach einer Tiefpassfilterung mittels Butterworth Filter dritten Grades

Als Tiefpassfilter wurden Tests mit Butterworth und Chebyshev Filtern durchgeführt. Da die Wahl des Dämpfungsgrades und der Grenzfrequenz entscheidend für das Filterdesign ist, wurden zuerst Versuche durchgeführt um diese zu bestimmen. Beide Werte müssen zweckmäßig gewählt werden, um die unerwünschten Frequenzen zu blockieren, gleichzeitig aber die niederfrequenten Anteile so ungedämpft passieren zu lassen, dass das gefilterte Signal dem Original noch gerecht wird und nicht nur aus der Filterfunktion besteht. Als Dämpfungsgrad hat sich die dritte Ordnung (visuell) als geeignet herausgestellt,

als Grenzfrequenz der Betrag 0.03. Dieser Betrag bedeutet nicht, dass die Grenzfrequenz $\approx$33Hz beträgt! Der MATLAB Befehl *butter*[6] bedingt eine Angabe zwischen 0 und 1 als Grenzfrequenz, wobei der Betrag 1 der Nyquistfrequenz entspricht.

Eine Anwendung der SFA, nachdem der Trainingsdatensatz in der Vorverarbeitung tiefpassgefiltert wurde, führte nur zu kleinen Veränderungen in der Erkennungswahrscheinlichkeit. Auf der Trainingsmenge erhöhte sich die Erkennungswahrscheinlichkeit um 0,2%, auf der Testmenge um 0,5%. Diese Veränderungen sind zu klein, um von einen Einfluss von Aliasing zu sprechen. Vielmehr handelt es sich wahrscheinlich um einen statistischen Zufall, da bereits kleine Änderungen in dem Dämpfungsfaktor oder der Grenzfrequenz die Erkennungswahrscheinlichkeit unverändert oder sogar sehr leicht reduziert beeinflussen.
Dieses Verhalten lässt folgende Vermutungen entstehen:

- **Effektlosigkeit der Unterabtastung**
  Da die SFA die langsamsten Prozesse bestimmt, entfällt auf Artefakte der Unterabtastung kein Gewicht. Diese können in ihrer Frequenz bereits so hoch sein, dass sie keine Rolle mehr spielen. Weiterhin kann es sein, dass die Unterabtastung nicht zum Vorschein kommt, weil ihr Auftreten periodisch in Erscheinung tritt und sie somit nicht als Information gewertet wird.

- **Unzureichende Filterung**
  Obwohl mit verschiedenen Faktoren und Kombinationen experimentiert wurde, kann nicht ausgeschlossen werden, dass eine Kombination aus Dämpfungsgrad und Grenzfrequenz existiert, die einen deutlich positiven Effekt auf die Erkennungswahrscheinlichkeit ausgeübt hätte. Da das Betragsfenster für die Grenzfrequenz nach oben und unten gut erkennbare Grenzen besitzt, wurde dieser Fall als sehr unwahrscheinlich eingestuft. Wird die Grenzfrequenz zu niedrig gewählt, bestimmen Artefakte das Signal und repräsentieren nicht mehr die Bewegung: die Erkennungswahrscheinlichkeit bricht stark ein. Wird die Grenzfrequenz zu hoch gewählt, unterscheidet sich das Signal nur in feinen Nuancen vom ungefilterten Signal, keine Änderung in der Erkennung tritt ein.

Neben der Tiefpassfilterung wurden auch Versuche mit einer anderen Form der Unterabtastung durchgeführt. Wie bereits beschrieben, erfolgt die Unterabtastung durch die

[6] http://www.mathworks.de/help/toolbox/signal/ref/butter.html

Mittelwertbestimmung von m aufeinanderfolgenden Elementen, die ein reduziertes Element darstellen. Als Alternativform zur der Mittelwertbestimmung wurde unter anderem eine kubische Spline-Interpolation mittels *interp1*[7] auf dem zu reduzierenden Vektor durchgeführt. Diese und verschiedene andere Interpolationsmethoden, die der *interp1* Methode als Parameter mitgegeben wurden, führten zu keiner Veränderung in der Erkennungswahrscheinlichkeit.

4.3 Expansionsartefakte in $avg1$

Wie im Abschnitt **SFA** in Gleichung 8 beschrieben ist, wird von jeweiligen expandierten Vektor e der Mittelwert e_0 abgezogen. Der Mittelwert e_0 besteht jedoch nur aus Beträgen die extrem nahe 0 sind (kleiner $1,0\text{e}{-16}$) oder Beträgen die gegen 1 streben. Interessant ist die Verteilung der Beträge nahe 1, da diese der Verteilung der Expansion oder aus einem Expansionsartefakt bestehen, wie in Abbildung 13 zu sehen ist. Der

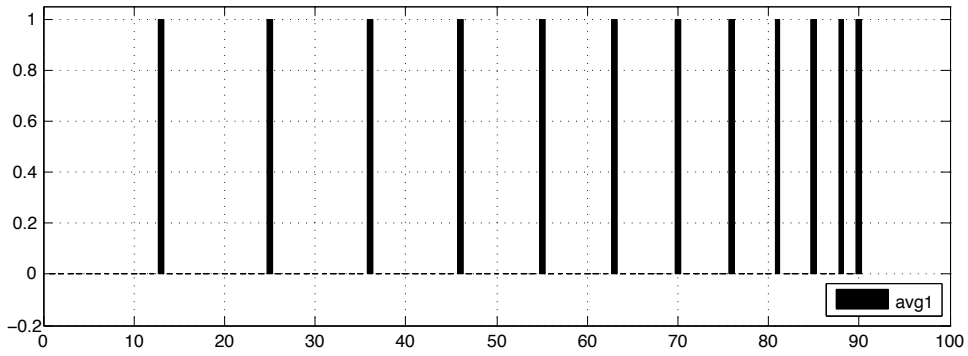

Abbildung 13. Expansionsartefakte in dem Vektor $avg1$

Ursprung dieses Artefaktes konnte nicht bestimmt werden, da die Erstellung und Weitergabe durch zahlreiche Pakete erfolgt und die Debuggingmöglichkeiten von MATLAB beschränkt sind. Durch die homogenen Werte, die auf alle Vektoren angewendet werden, kann die Subtraktion des Vektors e_0 während der Anwendung entfallen. Die Erkennungswahrscheinlichkeit wird nicht tangiert.

4.4 Gaussdimensionen

Wie im Gleichung 9 dargestellt, erfolgt eine Reduktion der Gausseingangsdimensionen von N auf M Elementen, die anschließend zur Klassifikation ausgewertet werden. M ist durch $k-1$ gegeben. Faktisch werden aber die restlichen Informationen ignoriert, die auf

[7] http://www.mathworks.de/help/techdoc/ref/interp1.html

die Erkennung der Gesten relevant sein könnten.

Um die Erkennungswahrscheinlichkeit für unterschiedliche Eingangdimensionen zu bestimmen, wurde M von k entkoppelt. Da die SFA im aktuellen Stand pro Datensatz einen expandierten Vektor mit 90 Elementen anbietet, wurden Tests mit M auf Werte von 1 bis 12 beschränkt. Für die Tests wurde ein Modell mit dem vollständigen Trainingsdatensatz erstellt, welcher im Anschluss den Testdatensatz klassifiziert hat. Das Training sowie die Anwendung erfolgten mit dem gewählten M.

Zur Auswertung wurde die Erkennungewahrscheinlichkeit für $M = 1$ nicht dargestellt, diese betrug auf dem Trainingsdatensatz nur 61% und auf dem Testdatensatz 33,2%. Wie in Abbildung 14 zu sehen ist, reduziert sich die Erkennungswahrscheinlichkeit gleichermaßen für Trainings- und Testdatensatz für kleine und für hohe M. Für den Testdatensatz gibt es lokale Minima für $M = 4$ und $M = 6$ während der Trainingsdatensatz für M größer 4 nur geringfügig schwankt. Die Wahl von $M = k - 1$ kann, wenn auch nur in diesem Fall, bestätigt werden. Kleine M reduzieren die Differenzierbarkeit für den Klassifikator (der Klassifikator ist unterangepasst) während große M zu einer Überanpassung führen, welche im benutzerunabhängigen Fall die Fehlerwahrscheinlichkeit steigen lässt.

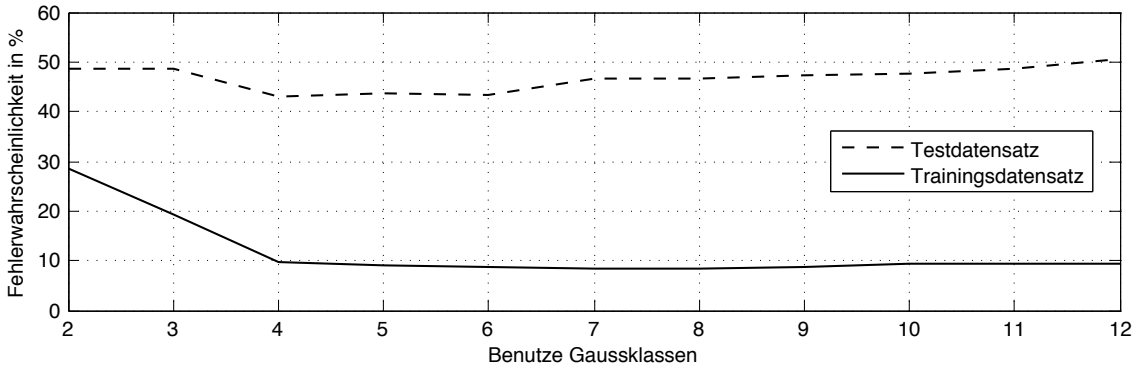

Abbildung 14. Fehlerwahrscheinlichkeit des Trainings- und Testdatensatzes für unterschiedliche Gaussdimensionen

Es ist beeindruckend, dass von den 90 Dimensionen, die die SFA für jeden Datensatz liefert, bereits 4 ausreichen um ein gutes Klassifikationsergebniss zu erhalten. Da die Ausgabe der SFA nach Relevanz *sortiert* erfolgt, tragen die ersten Vektoren den Großteil der Informationen, während der Informationsgehalt abnimmt. Wenn statt der ersten M

Elemente des Klassifikators durch eine Verschiebung ein vom Betrag kleinerer Teil der Ausgabe verwendet wird, reduziert sich die Erkennungswahrscheinlichkeit deutlich. Werden sogar nur die M letzten Elemente verwendet, sinkt die Erkennungswahrscheinlichkeit auf dem Trainingsdatensatz auf 38,6% und auf dem Testdatensatz auf 29,6%.

5 Benutzereinfluss

Bei allen Untersuchungen ist der Einfluss des Menschen nicht außer Acht zu lassen. Das Wissen, wie eine Geste in die Luft zu „zeichnen“ ist, bedeutet nicht immer, dass das auch so passiert. Während der Aufnahme des Testsets wurde genau beobachtet, welche Abweichungen die einzelnen Probanden hatten. Immer wieder kam es zu Aufnahmen, die so weit von den Gesten abwichen, dass diese auch von außen betrachtet nicht erkannt werden konnten. Teilweise wurden auch Aspekte einer Geste unbeabsichtigt verändert, wie zum Beispiel in Abbildung 15 dargestellt. Durch das Fehlen der beiden diskreten Momente, an den Ecken des *X*, die charakteristisch für eine X-Geste sind, wurde die Geste (b) als *Epsilon* klassifiziert. Neben dieser Abweichung gab es eine Vielzahl von Variationen

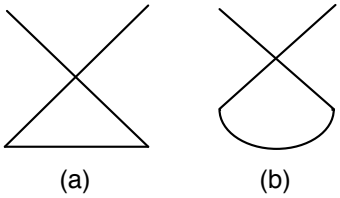

Abbildung 15. (a) stellt eine korrekt durchgeführte Geste vom Typ X dar, (b) eine inkorrekte Durchführung.

in der Ausführung von Gesten. Die Kreisgeste z.B. war oft interessant zu beobachten, weil sie zwar leicht umzusetzen ist, gleichzeitig jedoch ein hohes Fehlerpotenzial eröffnet. Zwischen Halbkreisen und Doppelkreise waren alle Varianten zu beobachten. Sogar bei dem *Z*, welches eine sehr robuste Geste darstellt, gab es Ausrutscher: so kamen unter anderem „runde“ *Z* (ohne Spitzen, also als ein großes gespielteltes *S* oder ein *Doppel-Z*, welches zwei mal unter einander gezeichnet wurde, vor. Die Geste *Herz* wurde oft nur als eiförmiger Kreis gezeichnet.

Zusammenfassend lässt sich festhalten, dass „die perfekte Gestenerkennung“ mit allen Variationen, die Menschen erzeugen können, umgehen und diese interpretieren muss. Eine Leistung, die wohl nie erbracht werden kann. Da während der Beobachtungen aber auch korrekt durchgeführte Gesten nicht richtig erkannt wurden, gibt es bis dahin noch einige Aspekte zu beachten.

Im Folgenden werden drei Einflüsse des Menschen dargestellt und potenzielle Lösungen, wie mit diesen umgegangen werden kann, vorgestellt.

5.1 Ruhebetrachtung

Während der Aufnahme des Testdatensatzes ist aufgefallen, das einige Gesten durch eine Pause vor oder nach der Durchführung der eigentlichen Geste begleitet wurden, in welcher das Gerät nicht bewegt, die Aufzeichnung jedoch schon begonnen wurde. Diese Pause kann als Ruheposition betrachtet werden, in der das Gerät bis auf kleinere Schwankungen durch den Menschen den Ruhevektor aufnimmt, mit dem das Gerät gehalten wird. Abbildung 16 zeigt einen solchen Datensatz. Bis zum 150. Element besteht dieser Datensatz aus der Ruhelage und ab dem 150. Element beginnt die eigentliche Geste. Dieser Datensatz wird als negativ erkannt, durch eine manuelle Entfernung der Anfangsruhe traten zwei Änderungen ein: Zum einen wurde der Datensatz positiv erkannt, was als lokale Verbesserung zu werten ist, und zum anderen erhöhte sich die Erkennung auch global auf drei weiteren Gestentypen. Dieses kann zurückgeführt werden auf das Training, da die Stille in der Geste nicht mehr als Merkmal dieser Geste gewertet wurde und andere Datensätze anderer Gesten dieses Merkmal teilen.

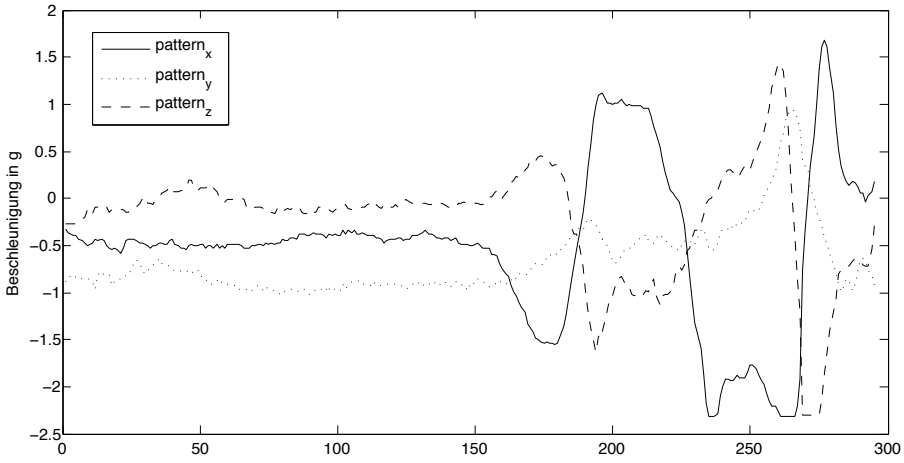

Abbildung 16. Rohdaten des Datensatzes 431

5.2 Automatische Segmenttrennung

Die automatische Erkennung des Anfangs und des Endes einer Geste wird als Segmentierung bezeichnet. Da in diesem Fall jeder Datensatz a priori nur eine Geste beinhaltet, war nur eine Segmenttrennung von Interesse. Der Teil des Datensatzes, der die eigentliche Geste beinhaltet, ist das Gestensegment. Der oder die Teile, die die Ruhe beinhalten, werden als Ruhesegment bezeichnet. Dieses kann insbesondere dazu benutzt werden, die

Lage des Gerätes über eine Bestimmung des g-Vektors zu ermitteln.

Für die Erkennung des Gestensegmentes gibt es verschiedene Arbeiten, die sich damit beschäftigt haben. Zahlreiche Arbeiten benutzen eine manuelle Segmentierung, auf diese soll hier nicht eingegangen werden. Prekopcsák [12] zum Beispiel benutzt zum automatischen Segmentieren den Betrag der Eingangsvektoren, um schnelle Änderungen als Gestenstart zu bestimmen.

Im Folgenden wird ein recht einfaches Konzept zur Gestensegmentierung umgesetzt, ähnlich wie es auch in [26] beschrieben wird. In diesem wird zuerst eine Amplitudennormierung der Eingabevektoren v_w durchgeführt, wobei $w \in X, Y, Z$. Im Anschluss wird die potenzielle Ruhe am Anfang und am Ende des Datensatzes bestimmt. Hierzu wurde der Betrag des ersten (k_1) und letzten (k_N) Elementes pro Beschleunigungsvektor genommen und als Referenzwert η_1 und η_N gespeichert. Um Schwankungen und langsame Bewegungen vor und nach der eigentlichen Geste zu unterdrücken, wurde der Referenzwert adaptiv gestaltet. Die Referenz für die Beschleunigungswerte des Folgeelementes k_{n+1} ergibt sich durch:

$$\eta = \frac{(2\eta_1 + k_n)}{3} \tag{20}$$

Dadurch werden kleine Schwankungen des Gerätes auch über einen längeren Zeitraum als Ruhe erkannt. Der Schwellwert α beschreibt die Differenz, die ein Folgeelement besitzen muss, um den Beginn des Gestensegmentes zu markieren:

$$k_{n+1} > (\eta + \alpha) \text{ oder } k_{n+1} < (\eta + \alpha) \tag{21}$$

Für die Beseitigung für Ruhe am Ende einer Geste wird nur der Startpunkt und die Laufrichtung umgekehrt. Die Wahl fiel auf die Betrachtung der Komponenten, weil der Betragsvektor aller Komponenten die Summe kleiner Änderungen beinhaltet und somit eher einen vermeintlichen Gestenstart kennzeichnet.

Diese brachiale Methode führt für unterschiedliche α Beträge zu einer effektiven Ruheauslöschung. Es wurde noch ein weiterer Effekt beobachtet: für relativ hohe α erzielt diese Methode erstaunlich gute Ergebnisse, obwohl nicht nur Ruhe entfernt wird, ganze Signalflanken werden ausgelöscht. Trotz dieser Auslöschungen steigt bis zu einem gewissen Punkt die Erkennungswahrscheinlichkeit auf dem Testdatensatz an, was dafür spricht,

dass die Signalflanken eines Datensatzes für die SFA nur von niederwertigem Rang sind. Die Homogenität der Signalform führt zu einer Reduktion auf die wesentlichen Elemente einer Geste. Abbildung 17 stellt die Fehlerwahrscheinlichkeit für unterschiedliche α dar.

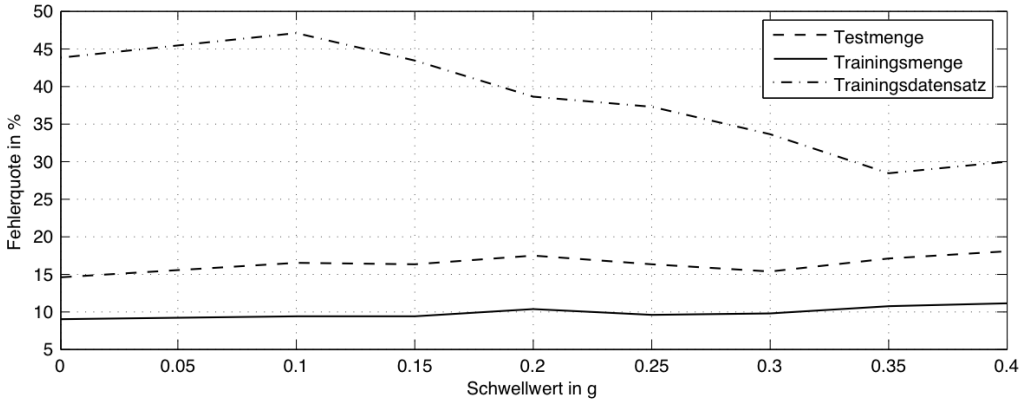

Abbildung 17. Fehlerwahrscheinlichkeit der Test- und Trainingsmenge des Trainingsdatensatzes sowie die Fehlerwahrscheinlichkeit des Testdatensatzes für unterschiedliche α

Gut zu erkennen ist, dass auf dem Trainingsdatensatz, also dem benutzerabhängigem Fall, die Fehlerquote leicht schwankt. Auf dem Testdatensatz, also dem benutzerunabhängigem Fall, erzielt ein α von 0,35g ein Minimum in der Fehlerquote. Für diesen Fall steigt die Erkennungswahrscheinlichkeit auf dem Testset von 56,2% um 15,4% auf 71,6%!
Auf dem Trainingsdatensatz reduziert sich die Erkennungswahrscheinlichkeit leicht auf 89,3% auf der Trainingsmenge und 82,9% auf der Testmenge.

Die durchschnittliche Länge eines Datensatzes im Trainingsdatensatz reduziert sich von 153 Elementen auf 142 Elemente. Die Differenz zwischen den Original- und den reduzierten Datensätzen zeigt, dass nur eine kleine Gruppe von Datensätzen einen Betrag über dieser Längendifferenz besitzt und eine noch kleinere Gruppe Beträge, die sehr deutlich abweichen. Diese Datensätze sind es, die von diesem Verfahren besonders profitieren.

Solange keine automatische Gestensegmentierung durchgeführt wird und der Mensch Start und Ende definiert, scheint der Fall der Gestenruhe einen der intensivsten Einflüsse für den aktuellen Prozess darzustellen. Die Ruhe vor und nach der Geste bietet keinen Mehrwert für das Verfahren, beeinflusst es aber durch unterschiedliche Längen, die von der SFA als „Information“ gewertet werden. Konfusionsmatrix 14 zeigt sehr deutlich den positiven Einfluss der Segmenttrennung, die Hauptdiagonale ist sehr deutlich erkennbar

und im Vergleich zu der Original-Konfusionsmatrix 7 gibt es auch keine dominanten Fehlerkennung.

	G1	G2	G3	G4	G5	G6
G1	33	4	0	3	6	0
G2	2	29	1	1	4	1
G3	4	5	28	3	1	1
G4	1	7	3	31	6	3
G5	2	1	0	0	32	2
G6	3	2	1	0	3	24

Tabelle 14. Die von MATLAB bestimmte Konfusionsmatrix, der Gesten G1 bis G6 auf dem Testset

5.3 Winkelkompensation

Bei einem Winkelausgleich in einem Datensatz handelt es sich um eine Rotation des gesamten Datensatzes um die Winkel α, β und γ. Je nach Lage des Gerätes kann jeder dieser Winkel unterschiedlich starke Ausprägungen besitzen. Um eine Unabhängigkeit von der Haltung des Gerätes zu erzielen und somit die benutzerunabhängigen Erkennungsraten zu steigern, oder zumindest an die benutzerabhängigen Erkennungsraten zu steigern, wurde versucht, mit einem Winkelausgleich die Eingangsvektoren zu justieren.

Um die Lage des Gerätes, seine Orientierung in einem 3D-Raum, zu beschreiben, gibt es verschiedene Möglichkeiten. Weit verbreitet ist eine Beschreibung mittels Eulerwinkel, durch Rotationsmatrizen oder mittels Quaternionen.

Eulerwinkel wurden von Leonhard Euler (1707 – 1783) zur Beschreibung starrer Körper im Raum eingeführt. Ihr großer Vorteil ist, dass sie sich durch die Angabe von drei Winkeln ausdrücken lassen. Eulerwinkel werden jedoch von zwei Problemen begleitet. Zum einen fehlt ihnen die Eindeutigkeit, so gibt es zwölf Möglichkeiten eine Drehung in einem 3D-Raum auszudrücken und zum anderen „leiden“ sie unter einer potenzielle *kardanischen Blockade*. Diese kann auftreten, wenn zwei Achsen zeitweise aufeinander liegen. In diesem Fall geht eine Dimension, ein Freiheitsgrad, verloren. Um insbesondere von letzterem Problem Abstand zu gewinnen, werden Eulerwinkel in dieser Arbeit nicht weiter betrachtet.

Rotationen können als Kombination von Einzelrotationen durch Rotationsmatrizen um die drei Koordinatenachsen ausgedrückt werden [27]. Rotationsmatrizen drücken Roationen in 3x3 Matrizen aus, wobei die Matrixmultiplikationen nicht kommultativ sind. Gleichung 22 stellt eine allgemeine Rotationsmatrix dar.

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{22}$$

Eine Rotation um die x-Achse ist definiert als:

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{pmatrix} \tag{23}$$

Um die y-Achse:

$$R_y = \begin{pmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{pmatrix} \tag{24}$$

Um die z-Achse:

$$R_z = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{25}$$

Nicht jede 3x3 Matrix beschreibt eine Rotation, hierzu muss sie orthogonal sein und eine Determinante von **1** besitzen. Da jede Rotationsmatrix durch neun Werte beschrieben wird, ist die Arbeit mit erhöhtem Speicher- und Rechenaufwand verbunden und es ist schwer, zwischen Werten zu interpolieren.

5.4 Winkelkompensation mittels Quaternion

Ein Quaternion, entworfen 1843 von Hamilton, ist eine Erweiterung der komplexen Zahlen auf drei imaginäre Einheiten und kann durch eine vierdimensionale Hyperfläche (Abbildung 18) dargestellt werden. Ziel von Hamilton war die Division von Vektoren im dreidimensionalen Raum[28]. Die Rotation erfolgt hierbei direkt um die gewünschte Achse. Ein Quaternion kann jedoch keine Skalierung oder Translation abbilden, hierzu

sei auf die Biquaternionen verwiesen. Jeder Quaternion ist ein Quadrupel und lässt sich hierbei in der Form

$$q = (w, xi, yj, zk) \tag{26}$$

darstellen, wobei

$$i^2 = j^2 = k^2 = -1 \tag{27}$$

mit

$$\begin{aligned} ij &= k, \quad ji = -k \\ jk &= i, \quad kj = -i \\ ki &= j, \quad ik = -j \end{aligned} \tag{28}$$

Die Multiplikation von zwei Quaternionen ist nicht kommutativ und definiert als:

$$\begin{aligned} q_1 \cdot q_2 = \quad & (w_1w_2 - x_1x_2 - y_1y_2 - z_1z_2) \\ + \quad & (y_1z_2 - y_2z_1 + w_1x_2 + w_2x_1) \cdot i \\ + \quad & (z_1x_2 - z_2x_1 + w_1y_2 + w_2y_1) \cdot j \\ + \quad & (x_1y_2 - x_2y_1 + w_1z_2 + w_2z_1) \cdot k \end{aligned} \tag{29}$$

Der Betrag eines Quaternion q ist definiert als:

$$|q| = \sqrt{w^2 + x^2 + y^2 + z^2} \tag{30}$$

Der konjugierte Quaternion ist definiert als:

$$q^* = (w, -xi, -yj, -zk) \tag{31}$$

Die Division von Quaternionen wird mittels Multiplikation durchgeführt.

$$\frac{q_1}{q_2} := q_1 q_2^{-1} = q_1 \frac{q_2^*}{|q_2|^2} \tag{32}$$

Eine Rotation um den Winkel α eines normierten Vektors wird durch das Quaterion:

$$q = \cos\frac{\alpha}{2} + i \cdot \sin\frac{\alpha}{2} + j \cdot \sin\frac{\alpha}{2} + k \cdot \sin\frac{\alpha}{2} \tag{33}$$

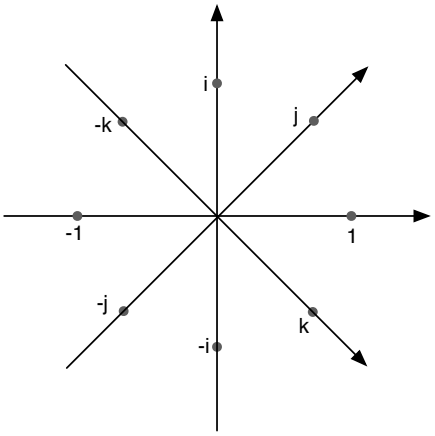

Abbildung 18. Graphische Darstellung der Quaternion-Hyperfläche

Ziel dieses Winkelausgleichs ist es, die Eingabevektoren einer Geste an einen Normalvektor anzupassen, um somit die Haltung des Gerätes im Raum aus den Berechnungen zu tilgen. Für diese Rechnungen wurde die Quaternion Toolbox für MATLAB [29] verwendet. Ein Punkt $p = (x_p, y_p, z_p)$ wird durch das Quaternion

$$q_p = (0, x_p, y_p, z_p) \tag{34}$$

beschrieben, eine Rotation um die Y-Achse um den Winkel β wird durch den Quaternion

$$q_r = (\cos\frac{\beta}{2}, 0, \sin\frac{\beta}{2}, 0) \tag{35}$$

beschrieben. Die Rotation des Punktes q_p um q_r ergibt sich durch:

$$q_p\prime = q_r \cdot q_p \cdot {q_r}^* \tag{36}$$

Ein normalisiertes Quaternion kann auch als Matrix dargestellt werden, sowie auch ein Quaternion als Matrix dargestellt werden kann

$$M_r = \begin{pmatrix} 1-2(y^2+z^2) & 2(xy-zw) & 2(xz+yw) \\ 2(xy+zw) & 1-2(x^2+z^2) & 2(yz-xw) \\ 2(xz-yw) & 2(yz+xw) & 1-2(x^2+y^2) \end{pmatrix} \tag{37}$$

$$q = \begin{pmatrix} \frac{\sqrt{1+r_{11}+r_{22}+r_{33}}}{2} \\ \frac{r_{32}-r_{23}}{4w} \\ \frac{r_{13}-r_{31}}{4w} \\ \frac{r_{21}-r_{12}}{4w} \end{pmatrix} \tag{38}$$

Zur Rotation eines Datensatzes wird ein Ansatz aus der Satellitennavigation benutzt, um die Orientierung eines Satelliten, der um die Erde kreist, zu bestimmen. Sei q_N der Normpunkt, mit dem alle Datensätze starten müssen.

$$q_N = (0, -0.02i, -0.7j, -0.7k) \tag{39}$$

q_1 sei das erste Element aus einem Datensatz, so wird das Quaternion q_r, welches die relative Rotation der beiden Koordinatensysteme beschreibt, q_N bestimmt durch $q_N = q_r q_1 q_r^*$. Das Quaternion $q = q_N q_1^*$ erfasst die Rotation von q_1 zu q_N, so ist $q = q_r q_r^*$. Um Gleichung 36 zu erfüllen, wird zuerst jedes Element in einem zu bestimmenden Datensatz auf den Betrag **1** normiert. Im Anschluss wurde der Initialwinkel von q_1 zu q_N bestimmt und jedes Element um q_r rotiert.

Nach der Modellerzeugung der mit Quaternionen rotierten Datensätze ergibt sich Konfusionsmatrix 15. Die Erkennungswahrscheinlichkeit auf der Trainingsmenge sinkt von 91,1% auf 89,3% und auf der Testmenge von 85,4% auf 81,4%.

	G1	G2	G3	G4	G5	G6
G1	113	8	8	1	14	10
G2	7	111	9	1	4	5
G3	6	19	115	1	2	5
G4	2	3	9	135	1	2
G5	6	4	2	2	128	4
G6	9	7	1	2	12	124

Tabelle 15. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach dem Training des Modells mit rotierten Datensätzen

Auf dem Testdatensatz sinkt die Erkennungswahrscheinlichkeit von 56,2% auf 53%. Insbesondere Geste G2 und G6 brechen in ihrer Erkennung ein.

Durch die positiven Ergebnisse, die mit der Translation zu 0 erzielt wurden, wurde eine Translation nach der Quaternionenrotation durchgeführt. Der positive Effekt konnte lei-

	G1	G2	G3	G4	G5	G6
G1	21	11	3	1	1	6
G2	4	17	2	2	4	9
G3	4	12	23	3	3	2
G4	4	4	3	30	4	0
G5	6	2	1	0	30	4
G6	6	2	1	2	10	10

Tabelle 16. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testdatensatz nach einer Translation zur Idealstartposition

der nicht vollständig übertragen werden. Während die Erkennungswahrscheinlichkeit auf der Trainingsmenge 90,1% und auf der Testmenge 83,2% leicht höher als bei dem reinen Quaternionausgleich sind, reduziert sich die Erkennungswahrscheinlichkeit auf dem Testdatensatz um fast 1% auf nur 52,2% im Vergleich zu dem reinen Quaternionausgleich.

	G1	G2	G3	G4	G5	G6
G1	115	10	2	1	7	4
G2	14	115	12	6	4	6
G3	4	16	120	1	0	5
G4	2	2	3	132	1	3
G5	3	7	5	2	134	6
G6	5	2	2	0	15	126

Tabelle 17. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach dem Training des Modells mit rotierten Datensätzen und anschließender Translation zu 0

Wie im Abschnitt **g-Vektor** festgestellt wurde, ist der Mittelwert über die gesamte Geste deutlich stabiler als Bezugswert für die Lage des Gerätes ist, als der Betrag des ersten Elementes. Hierzu wurde zuerst der Mittelwert von jeder Beschleunigungskomponente genommen und mit diesem das Quaternion q_N überschrieben mit

$$q_N = (0, ix_{mean}, jy_{mean}, kz_{mean}) \tag{40}$$

Durch diese veränderte Winkelwahl wird die Gerätelage besser berücksichtigt. Auf dem Trainingsdatensatz erhöht sich die Erkennungswahrscheinlichkeit der Trainingsmenge von 91,1% auf 91,7% leicht, auf der Testmenge steigt die Erkennungswahrscheinlichkeit

von 85,4% auf 87,2%. Die Konfusionsmatrix 18 zeigt deutlich die verbesserten Erkennungen in Vergleich zu Konfusionsmatrix 15.

	G1	G2	G3	G4	G5	G6
G1	119	8	2	0	11	5
G2	5	128	7	1	5	5
G3	4	9	127	0	2	1
G4	2	1	6	137	0	3
G5	2	2	2	4	135	4
G6	11	4	0	0	8	132

Tabelle 18. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach dem Training des Modells mit rotierten Datensätzen, dessen Ursprungslage durch den Mittelwert bestimmt wurde

Auch auf dem Testdatensatz erhöht sich die Erkennungsleistung von 56,2% auf 57,9%. Die Ergebnisse lassen den Schluss zu, dass eine Rotation des zu klassifizierenden Datensatzes vom Betrag des Mittelwertes deutlich stabiler in der Erkennung ist, als eine Rotation vom Betrag des ersten Elementes.

5.5 Test des Quaternionausgleichs

Die Erkennungswahrscheinlichkeiten mit dem Quaternion sind teilweise schwächer als die anderen Modifikationen, die durchgeführt wurden. Die Anwendung von Quaternionen auf dem Datensatz erfolgt nur sekundär aus Gründen der Verbesserung der Erkennung, sondern primär um die Erkennung unabhängig von dem Winkel, mit dem das Gerät gehalten wird, zu verbessern. Um diesen Aspekt überprüfen zu können, wurde ein sehr kleiner Datensatz im Liegen aufgenommen. Der Gedanke hierzu war, die Funktionsfähigkeit der Quaternionenrotation zu überprüfen. Der Datensatz besteht aus 62 Gesten, die von zwei Personen aufgenommen wurden.

Während der Aufnahme hat es sich als große Herausforderung aufgezeigt, das Gerät in einer liegenden Position mit nur einer Hand zu benutzen. Diese Position ist mit einem Gerät vom Format eines iPhones weder angenehm noch ergonomisch. Den Daumen nicht zum Halten des Gerätes benutzen zu können, kam erschwerend hinzu. Dass durch diese Einschränkung in der Bewegung die Gestenbewegung nur eingeschränkt durchgeführt werden konnte, soll an dieser Stelle nicht unerwähnt bleiben. Von den zwei aufgenommenen Personen ist eine Person im Trainingsdatensatz enthalten, die andere nicht. Erstere

hatte für die *Z*- und *Epsilon*-Geste eine Erkennungswahrscheinlichkeit von 100%, für die anderen betrug die Wahrscheinlichkeit 0%. Die zweite Person erhielt für die *X*-Geste eine Erkennungswahrscheinlichkeit von 100%, für die restlichen Gesten eine Wahrscheinlichkeit, die ebenfalls nahe 0% lag. Durch den Fall, dass Gesten perfekt erkannt wurden, andere gar nicht, wird ersichtlich, dass die SFA an sich invariant zu Winkeln oder Einflüssem dieser Rotation ist, solange die Natur der Bewegung eingehalten wird. Da verschiedene Gesten nicht erkannt wurden, scheint bei diesen die Bewegungsnatur so weit von der erlernten Natur entfernt zu sein, dass eine Erkennung nicht möglich war. Bemerkenswert ist Konfusionsmatrix 19. Diese entstand durch das Anwenden des liegend aufgenommenen Datensatzes auf ein aus dem Trainingsdatensatz vollständig trainiertes Modells. Die während der Aufnahme festgestellte perfekte Erkennung der *X*-Geste von Person zwei und die Erkennung der *Z*- und *Epsilon*-Geste lassen sich in der Konfusionsmatrix nicht wiederfinden!

	G1	G2	G3	G4	G5	G6
G1	0	0	0	0	0	0
G2	0	3	2	0	0	1
G3	0	3	4	2	0	0
G4	0	0	0	0	0	0
G5	11	2	5	10	10	5
G6	0	0	0	0	0	4

Tabelle 19. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem liegend aufgenommenen Datensatz

Die Ursache für dieses Verhalten liegt in der Klassifikation. Das Modell des Klassifikators auf dem Gerät ist mit einer Cross Validation Einteilung von 10 erstellt, während die Klassifikation des Datensatzes in MATLAB mit dem gesamten Trainingsdatensatz erfolgt ist. Wie im Abschnitt **Klassifikation** gezeigt wurde, ist das Modell bereits übertrainiert, was bei diesen Datensätzen besonders auffällig wurde. Weiterhin ist auffällig, dass es zwei Nullzeilen gibt, keine Geste wurde dem *Kreis* oder dem *Epsilon* zugeordnet. Da während der Aufnahme fünf *Epsilon* erfolgreich erkannt wurden, kann ein weiterer Effekt, neben der Überangepasstheit, nicht ausgeschlossen werden.

Die Erkennungswahrscheinlichkeit des liegend aufgenommenen Datensatzes beträgt bei einer Anwendung auf ein vollständig trainiertes Modell 33,8%. Wird das Modell aus

dem Trainingsdatensatz mit einem Winkelausgleich mittels Quaternionen erzeugt, dessen g-Vektor aus dem ersten Element des Datensatzes erzeugt wird und der Testdatensatz ebenfall so eingelesen, erhöht sich die Erkennungswahrscheinlichkeit auf 51,6% und ergibt Konfusionsmatrix 20. Gut zu erkennen ist, dass die *Kreisgeste* (G1) massiv von der Rotation profitiert. Wird stattdessen der g-Vektor nicht vom ersten Element, sondern durch alle Elemente des zu klassifizierenden Datensatzes bestimmt, sinkt die Erkennungswahrscheinlichkeit auf 48,4%.

	G1	G2	G3	G4	G5	G6
G1	9	6	1	0	1	1
G2	2	1	0	4	2	2
G3	0	1	7	0	0	1
G4	0	0	2	6	0	0
G5	0	0	1	1	5	2
G6	0	0	0	1	2	4

Tabelle 20. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem liegend aufgenommenen Datensatz nach der Rotation mittels Quaternion

Durch die positiven Ergebnisse der Kombination des Quaternionausgleichs mit anschließender Translation zu 0 wurde auch dieser Datensatz mit dieser Kombination untersucht. Es zeigt sich, dass die Erkennungswahrscheinlichkeit 46,8% beträgt und sich die Konfusionsmatrix 21 ergibt, die der Konfusionsmatrix 20 wie erwartet recht ähnlich ist.

	G1	G2	G3	G4	G5	G6
G1	9	0	0	1	1	1
G2	2	2	2	5	3	0
G3	0	2	7	1	0	0
G4	0	0	0	2	0	0
G5	0	3	2	2	4	4
G6	0	1	0	1	2	5

Tabelle 21. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem liegend aufgenommenen Datensatz nach der Rotation mittels Quaternion und einer Translation zu 0

Zusammenfassend kann man sagen, dass eine Geräterotation bei einigen Gesten wie der Kreisgeste zu Verbesserungen in der Erkennung führt. Die SFA an sich ist in der Lage,

Gesten auch in ungewöhnlichen Posen zu erkennen, wenn die Bewegung in ihrer Natur korrekt durchgeführt wird, unabhängig davon, in welchem Winkel sich das Gerät befindet oder ob die Geste an sich rotiert ist. Die liegende Aufnahme eines Testdatensatzes ist gewiss eine der extremen Möglichkeiten, aber sie zeigt sehr deutlich die beschriebenen Eigenschaften auf. Dass auch die durch Quaternionen rotierten Datensätze keine höhere Erkennungswahrscheinlichkeit erreichen, liegt vermutlich in der Art der Aufnahme. Liegend ist nicht nur die Haltung des Gerätes sondern auch die Beweglichkeit des Armes eingeschränkt.

5.6 Bewegungsabruch und Ausklingen

Als Gegensatz zur *Ruhe* wurde ein weiteres Verhalten festgestellt, welches die Charakteristik einer Geste verändert: der Bewegungsabbruch. Dieser tritt auf, wenn eine Person noch während der Durchführung der Geste diese, durch das Lösen des Daumens, beendet. Hierdurch endet eine Geste nicht in annähernder Startposition, sondern mit den zu dem Abbruchzeitpunkt herrschenden Beschleunigungen. Diese können mitunter bei schnell durchgeführten Gesten hohe Amplituden erreichen, weil sich zum Beispiel das Gerät gerade in einer Phase der Beschleunigung befand. Während der Aufzeichnung des Testsets wurde die Beobachtung gemacht, dass das vorzeitige Beenden der Geste in aller Regel zu einem negativen Ergebnis führt.

Um diesem Abbruch entgegen zu wirken und um zu bestimmen, ob sich dieser Einfluss reduzieren lässt, wurde von jedem Datensatz das letzte Element zum Ausklingen gebracht.
Ausklingen beschreibt in diesem Fall eine lineare Rückführung zum Ordinatenursprung. Hierzu wurden zuerst die X-, Y- und Z-Vektoren des Datensatzes im Betrag ihres Initialwertes wie in der **Translation zu 0** verschoben. Im Anschluss wurde das letzte Element e von v_x, v_y oder v_z mit dem höchsten Betrag der Amplitude gesucht. Als Ausklingdauer d wurde 20 als Multiplikator m für die gefundene Maximalamplitude gewählt.

$$d = |e_{max}| \cdot m \tag{41}$$

Mit der bestimmten Ausklingdauer können jetzt die Beträge s_k für $k \in x, y, z$ für v_k bestimmt werden, mit der der jeweilige Vektor linear zurück geführt wird.

$$s_k = \frac{m_k}{d} \tag{42}$$

Der Faktor m wurde empirisch bestimmt. Während für einen Multiplikator von fünf nur geringfügige Änderungen eingetreten sind, hatte das Trainingsset mit der Trainings- und Testmenge sowie das Testset für einen Betrag von 20 jeweils ein Maximum in der Erkennungswahrscheinlichkeit. Für größere Beträge sind die Erkennungswahrscheinlichkeiten auf Test- und Trainingsset wieder gesunken.

Auf dem Trainingsset ergibt sich nach dem Ausklingen folgende Konfusionsmatrix:

	G1	G2	G3	G4	G5	G6
G1	130	7	2	0	12	2
G2	5	127	2	8	7	9
G3	0	8	130	1	2	2
G4	2	1	6	130	1	0
G5	6	6	2	3	137	6
G6	0	3	2	0	2	131

Tabelle 22. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 nach dem Ausklingen

Die Erkennungswahrscheinlichkeit auf der Trainingsmenge steigt von 91,1% auf 93,5% und auf der Testmenge von 85,4% auf 88%. Auf der Testmenge steigt die Erkennungswahrscheinlichkeit von 56,2% auf 67,6% mit der Konfusionsmatrix 23.

	G1	G2	G3	G4	G5	G6
G1	32	4	1	2	5	1
G2	1	32	5	1	6	8
G3	0	0	25	3	0	1
G4	2	9	1	30	0	0
G5	7	3	1	2	28	1
G6	3	0	0	0	13	20

Tabelle 23. Die von MATLAB bestimmte Konfusionsmatrix der Gesten G1 bis G6 auf dem Testset nach dem Ausklingen

Insbesondere das Testset profitiert mit fast 9% stark von dem Ausklingen, während auf dem Trainingsset noch jeweils 2% Verbesserung erzielt werden. Durch diese Modifikation der Eingangsvektoren wurden die Eingabevektoren homogenisiert und somit der Einfluss der potenziellen Bewegungsabbrüche reduziert. Leider sind die Ergebnisse dieser Methode nicht isoliert betrachtbar, da auf dem Eingangsvektor vor dem Ausklingen

eine Translation zu 0 durchgeführt werden muss.

Die Differenz betrachtend fällt auf, dass auf dem Trainingsset nur minimale Änderungen in der Erkennung eintreten. Die Erkennungswahrscheinlichkeit auf der Testmenge steigt um 0,1% und auf der Trainingsmenge sinkt diese um 0,2%. Dieses sind quasi keine Änderungen zu der Translation zu 0. Auf dem Testset jedoch erzielt die Ausklingfunktion eine Steigerung von fast 5% im Vergleich zu der reinen Translation. Dieses kann verschiedene Ursachen haben. Durch das Ausklingen mit vorhergehender Translation zu 0 werden die Datensätze im benutzerunabhängigen Fall stark in ihren Initial- und Endamplituden normiert und dem Trainingsdatensatz angepasst, was zu einer verbesserten Erkennung führt. Weiterhin ist es schlicht möglich, dass im Testdatensatz einfach mehr Datensätze enthalten sind, die auf ein Ausklingen ansprechen, die „abgebrochen“ wurden.

5.7 Methodenkombination

In den vorherigen Abschnitten wurden verschiedene Methoden vorgestellt, die die Erkennungswahrscheinlichkeit teilweise deutlich verbessert haben. Neben der Verbesserung der Erkennungsqualität führen diese teilweise auch zu einer Varianzreduktion und in drei Fällen sogar zu einer Invarianz. An dieser Stelle sollen Kombinationen aus den drei vielversprechnenden Ansätzen der Rotationsinvarianz, der Gestensegmentierung und der Ruheauslöschung gebildet werden. Alle drei Ansätze führen eine Invarianz herbei:

- Rotationsinvarianz: durch Winkelausgleich mit Quaternionen
- Ruheinvarianz: durch Gestensegmentierung mittels Ruheauslöschung
- Einflussreduktion des Bewegungsabbruchs: mittels Ausklingen

Hierzu sind verschiedene Kombinationen umgesetzt worden. Als erste Kombination wurden alle drei Methoden verkettet. Um dieses zu erreichen, wurde als erster Schritt der g-Vektor über den gesamten Eingabevektor dazu benutzt, den Gerätewinkel zu der Normhaltung q_N auszugleichen. Im Anschluss wurde vom rotierten Vektor die Ruhe entfernt. Abgeschlossen wurde die Kombination durch eine weitere Homogenisierung mittels Ausklingen, welches eine Translation zu 0 beinhaltet, um auch den Effekt des Bewegungsabruches zu tilgen.

Diese Kombination wurde so gewählt, um potenzielle Ruhe eines Datensatzes zur Bestimmung des Gerätewinkels nutzen zu können. Die Erkennungswahrscheinlichkeit reduziert sich durch diese Kombination auf dem Trainingsdatensatz auf 75,8% und auf dem

Testdatensatz auf 55,5%. Die einfache Kombination der einzelnen Verfahren, die einzeln mitunter sehr gute Ergebnisse erzielt haben, reduziert die Erkennungswahrscheinlichkeit auf dem Trainingsdatensatz um 15,3% und um 0,7% auf dem Testdatensatz. Bei der Kombination der drei Invarianz erzeugenden Methoden beträgt die benutzerunabhängige Erkennungswahrscheinlichkeit nur noch $\approx 55\%$. Mit dieser Kombination wurde eine maximale Invarianz zu Lasten einer optimalen Erkennungswahrscheinlichkeit getestet.

Die Aufzeichnung des kleinen Testsets, welches im Liegen aufgenommen wurde, hat gezeigt, dass starke Abweichungen im Gerätewinkel aus ergonomischen Gründen eher unwahrscheinlich sind. Da die Natur einer Geste entscheidender ist als der Winkel, wurde als zweite Kombination der Winkelausgleich entfernt und nur die Gestensegmentierung mit anschließendem Ausklingen getestet. In diesem Fall beträgt die Erkennungswahrscheinlichkeit auf dem Trainingsdatensatz 90,5% und auf dem Testdatensatz 69,6%. Auch wenn sich bei dieser Kombination die Erkennungswahrscheinlichkeit auf dem Trainingsdatensatz leicht reduziert, so erhält der Testdatensatz ein Plus von 13,4%. Die Unabhängigkeit vom Gerätewinkel geht verloren, die Erkennungsleistung in der benutzerunabhängigen Erkennung steigt deutlich.

Da die Winkelkompensation mittels Quaternion und das Ausklingen die Lage des Eingangsvektors beeinflussen, wurde die Kombination der Winkelkompensation mit anschließender Gestensegmentierung durchgeführt. Die Auslöschung der Ruhe vor und nach der eigentlichen Geste verändert die Lage des Vektors nicht. Für diesen Fall beträgt die Erkennungswahrscheinlichkeit auf dem Trainingsdatensatz 87,7% und auf dem Testdatensatz 59,9%. Der Mehrwert des Ausklingens kommt in dieser Kombination kaum zur Geltung. Die Rotation des Datensatzes macht diesen zwar invariant zur Lage im Raum, es werden aber Informationen zur Erkennung reduziert.

Es ergibt sich das Problem, dass einzelne Methoden mitunter sehr gute Ergebnisse in der Verbesserung der Erkennungsqualität führen, sich diese Methoden kombiniert mitunter aber auch negativ beeinflussen. Zur näheren Betrachtung der Kombinationsmöglichkeiten muss eine andere, formalere Betrachtung der Methoden durchgeführt werden.

5.7.1 Filterbetrachtung

In der Elekrotechnik und Nachrichtentechnik sind Filter zu den wichtigsten Operationen in der Signalverarbeitung [25]. Filter dienen dazu, eine Selektion im Bild- oder Frequenz-

bereich eines Signals vorzunehmen. Im Abschnitt *Unterabtastung* wurde ein Butterworth Filter für die Tiefpassfilterung angewendet. Durch ihren Ursprung in der Signalverarbeitung werden sie im Allgemeinen global auf ein Signal angewendet [30]. Lokale oder global-spezifische Anwendungen wie zum Beispiel die Rotation eines Datensatzes mittels Quaternion lassen sich nicht ohne weiteres beschreiben und formalisieren.

5.7.2 Operatorbetrachtung

In Anlehnung an die Bildverarbeitung [31] wird in diesem Abschnitt jede Methode als Filteroperator betrachtet. Hierzu wird zwischen den Operatoren die *lokal*[8] und *global* operieren unterschieden. Sei v der Eingangsvektor, so ist $\bar{v}$ der mit einem Operator T modifizierte Eingangsvektor.

$$\bar{v} = T(v) \tag{43}$$

- **Lokale Operatoren:**
  Verändern oder entfernen Elemente aus den Eingabevektoren nach einer oder mehrerer Regeln, ohne die anderen Elemente zu verändern. Werden Elemente gelöscht, werden die verbleibenden Elemente lückenlos aufgerückt, um keine Artefakte entstehen zu lassen. Ein Problem der lokalen Operatoren in dieser Arbeit ist, dass diese nicht auf eine feste Menge von Elementen fixiert sind, sondern ihren Einflussbereich selbst bestimmen. Unter den positiven Methoden fallen die diese Kategorie:
  - Ausklingen (T_A)
  - Ruheauslöschung (T_S)
- **Globale Operatoren:**
  Diese Operatoren verändern den gesamten Eingabevektor ohne lokale Eigenschaften zu beachten. Durch diese Veränderung arbeiten alle weiteren globalen Operatoren nur noch auf einem potenziell modifiziertem Vektor. Folgende Modelleigenschaften und Methoden, die die Erkennungsleistung positiv beeinflusst haben, gehören zu den globalen Operatoren:
  - Normierung (Länge und Amplitude)
  - Offset zu 0 (T_0)

[8] Die kleinste Form des lokalen Operators ist der Punktoperator, dieser wird in dieser Arbeit nicht beachtet.

- (Ausklingen) (T_A)
- Winkelkompensation mittels Quaternion (T_Q)

Das Ausklingen ist zwar ein lokaler Operator, da er einen potenziellen Bewegungsabbruch kompensiert, dies erfordert jedoch ein vorheriges Anwendung eines Offset zu 0 Ausgleiches. Somit ist das Ausklingen ein lokaler Operator, der aber erst nach einem globalen Operator ausgeführt werden kann. Die Normierung ist zwar auch ein globaler Operator, da diese für die Erkennung essenziell ist und der Unterabtastung kein negativer Einfluss nachgewiesen werden konnte, wurde die Normierung nicht weiter untersucht.

Im Folgenden wird die Kombination der Gestensegmentierung mit anschließendem Ausklingen weiter betrachtet. Für diese Kombination wird der Eingangsvektor v zuerst in einer Ruhebetrachtung mit T_S segmentiert und im Anschluss mit T_A zum Ausklingen gebracht .

$$\bar{v} = T_S(T_A(v)) \tag{44}$$

Beide Operatoren (T_S und T_A) beeinflussen als lokale Veränderung das Flankensegment der Geste, welches den Übergang von der Ruhelage zur eigentlichen Geste darstellt. Während T_S als Schwellwertoperator fungiert und die Flanken entfernt, fügt T_A dem Gestensegment wieder eine Flanke zum Ausklingen hinzu. Dennoch darf T_A nicht als Inverse zu T_S verstanden werden. Beide Operatoren dienen dem Homogenisierungsprozess. Der Einfluss der Operatoren untereinander soll nun dargestellt werden.

5.7.3 Operatoreinfluss

Um den Einfluss der vier zu betrachtenden Operatoren untereinander zu beschreiben, werden diese als Einflusstabelle abgebildet. Ein o symbolisiert Neutralität, ein $-$ oder $+$ stellt einen geringen Einfluss in negative oder positiver Art dar, $--$ oder $++$ einen hohen Einfluss. Für diese Beurteilung wird die Steigerung in der Erkennungswahrscheinlichkeit der einzelnen Operatoren in Relation der Operatoren untereinander auf dem Testdatensatz genommen. Durch diese Wahl wird der Einfluss auf den wichtigen Punkt der benutzerunabhängigen Gestenerkennung gefördert.

* Der Ausklingoperator T_A führt vorher einen Ausgleich mittels T_0 durch. Die Tabelle 24 stellt die Einflüsse der Operatoren dar. Der Operator der jeweiligen Zeile wird in

	T_A	T_S	T_0	T_Q
T_A	/	o	++	-
T_S	+	/	o	–
T_0	/*	+	/	o
T_Q	-	–	–	/

Tabelle 24. Überblick über den Einfluss der Operatoren untereinander

Abhängigkeit zum Operator, der in der Spalte angegeben ist, bewertet.

Bei dem Vorgehen wurde der Eingangsvektor nicht erhalten und der Operator T_2 wurde auf der Ausgabe von Operator T_1. Dieses ist zum Beispiel bei T_Q tabu, da jeder andere Operator die Eigenschaften und somit die Qualität von T_Q verändern würde. T_Q darf nur auf den Originaleingangsvektor angewendet werden.

Als weiterer Punkt, der bei der Anwendung der Operatoren beachtet werden muss, ist der Anwendungszeitpunkt. Wenn zwei Operatoren angewendet werden sollen, muss die Entscheidung getroffen werden, ob der Operator T_2 auf der Ausgabe von Operator T_1 angewandt werden soll oder ob beide Operatoren auf dem Eingangsvektor angewandt werden. Im Anschluss muss im zweiten Fall noch eine Zusammenführung der Ergebnisse durchgeführt werden. Hierzu muss jedoch der Einfluss isoliert werden. Alle vier vorgestellten Operatoren besitzen einen isolierbaren Einfluss. Beim *Ausklingen* (T_A) besteht der Einfluss aus den beiden Faktoren der Gestentranslation t_0, um die Geste zum Nullpunkt zu verschieben und der Ausklingdauer t_d. Für die Gestentranslation zu 0 wird ebenfalls der Wert t_0 verwendet. Der Rotationsoperator T_Q besitzt als isolierbaren Einfluss den Initialwinkels ω_0, der den bestimmten g-Betrag beinhaltet. Dieser Betrag muss immer vor der Modifikation aller anderer Operatoren bestimmt werden. Der Ruheoperator T_S benötigt zur Beschreibung die beiden Werte s_0 und s_1. Die Länge der Ruhe zu Beginn einer Geste wird durch s_0 angegeben, die Länge der Ruhe am Ende der Geste durch s_1.

6 Fazit

Diese Arbeit hat gezeigt, dass es Methoden gibt, die die Erkennungswahrscheinlichkeit im benutzerabhängigen und benutzerunabhängigen Fall durch Hilfe von Invarianzen steigern können, sowie im benutzerunabhängigen Fall die Erkennungsleistung stabilisiert wird. Tabelle 25 zeigt die Erkennungswahrscheinlichkeiten für die unterschiedlichen Methoden im Überblick. Die „klassische“ Signalanalyse hat sich als wenig aussagekräftig herausgestellt, weil die *Slow Feature Analysis* durch ihr Vorgehen in der Lage ist, unterschiedliche Datensätze auf ihre Essenz herunterzubrechen. Beobachtungen der Personen beim Aufzeichnen der Datensätze haben Ansätze zur Varianzreduktion geliefert. Durch einen Perspektivwechsel mittels FFT hat sich herausgestellt, dass die beiden Betrachtungsformen des Bild- und Frequenzbereiches jeweils einige Informationen betonen und andere verschweigen.

Methode	Trainingsmenge	Testmenge	Testset
Original	91,1%	85,4%	56,2%
Ausgleich zur Startposition	91,9%	85,5%	60,7%
Ausgleich zu 0	93,4%	88,2%	62,7%
Segmenttrennung	89,3%	82,9%	71,6%
Ausklingen	93,5%	88%	67,6%
Quaternion	89,3%	81,4%	53%
1 Quaternion mit g	91,7%	87,2%	57,9%

Tabelle 25. Überblick über die Erkennungswahrscheinlichkeiten der unterschiedlichen Methoden

Es wurde gezeigt, das die Qualität des gesamten Prozesses maßgeblich von dem Trainingsdatensatz und der Wahl des Klassfikators abhängt. Insbesondere der für den Prozess gewählte Gauss-Klassifkator reagiert schnell mit einer Über- oder Unteranpassung. Die Gradwanderung zwischen Über- und Unteranpassung hängt neben der Anzahl der Datensätze, die zum Training verwendet werden, auch von den Datensätzen an sich ab. Eine „zu gute“ Trainingsmenge führt den Klassifikator schnell in eine Überanpassung.

Für drei Fälle wurden Invarianzen erschaffen. Durch die Gestensegmentierung mittels Ruheauslöschung und dem Ausklingen werden die intensivsten Einflüsse des Menschen auf die Erkennung entfernt. Die Datensatzrotation mittels Quaternionen erlaubt eine Unabhängigkeit vom Gerätewinkel. Durch diese Methoden kann die Geste des Menschen im

Datensatz isoliert und der Erkennung zugeführt werden, sodass der Mensch weniger bei der Anwendung beachten muss.

In dieser Arbeit sind Elemente des Erkennungs- und Klassifikationsprozesses in den Fokus gerückt, die an dieser Stelle nicht weiter betrachtet werden können. Tests mit einem anderen Klassifkationsverfahren drängen sich auf. Das Potenzial, die spektralen Eigenschaften mit in die Erkennung aufzunehmen, gilt es weiter zu untersuchen. Die Korrelation der Spektren in Bezug auf positive und negative Erkennungen fallen mit in diesen Bereich. Weiterhin sollten die Methoden, die im ersten Ansatz als Operator ausgedrückt wurden, stärker formalisiert werden. Eine Beschreibung der Verkettung dieser Operatoren und deren Einfluss muss bestimmt werden, um diese berechenbar und vorhersagbar zu machen.

Literatur

[1] Kristine Hein. *Gestenerkennung mit der SFA - Klassifizierung von beschleunigungsbasierten 3D-Gesten des Wii-Controllers*. FH Köln (Cologne University of Applied Sciences), Master Thesis, 2010, 2010.

[2] Lee W. Campell, David A. Becker, Ali Azarbayejani, Aaron F. Bobick, and Alex Pentland. *Invariant features for 3-D gesture recognition*. M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 379, 1996.

[3] Warner Bros. Entertainment Inc. *Harry Potter Spells*. http://itunes.apple.com/de/app/harry-potter-spells/id337402021?mt=8, 2010.

[4] Stephane Perrin, Alvaro Cassinelli, and Masatoshi Ishikawa. *Gesture Recognition Using Laser-Based Tracking System*. Ishikawa Hashimoto Laboratory, University of Tokyo, 2004.

[5] Michael Hoffman, Paul Varcholik, and Joseph LaViola. *Breaking the Status Quo: Improving 3D Gesture Recognition with Spatially Convenient Input Devices*. www.cs.ucf.edu jjlpubslaviola336_Paper.pdf, University of Central Florida, 2008.

[6] Dean Rubine. *Specifying Gestures by Example*. InSIGGRAPH'91:Proceedings of the 18th annual conference on Computer graphics and interactive techniques, pages 329–337, New York, NY, USA, 1991.

[7] Yoav Freund. *Boosting a weak learning algorithm by majority*. AT&T Bell Laboratories, New Jersey, 1995.

[8] Jia Sheng. *A Study of AdaBoost in 3D Gesture Recognition*. Department of Computer Science, University of Toronto, 2003.

[9] Marco Klingmann. *Accelerometer-Based Gesture Recognition with the iPhone*. Goldsmiths University of London, 2009.

[10] M.B. Holte and T.B. Moeslund. *VIEW INVARIANT GESTURE RECOGNITION USING 3D MOTION PRIMITIVES*. Computer Vision and Media Technology Lab, Aalborg University, Denmark, 2008.

[11] Jiahui Wu, Gang Pan, Daqing Zhang, Guande Qi, and Shijian Li. *Gesture Recognition with a 3-D Accelerometer*. Department of Computer Science, Zhejiang University, Hangzhou, 310027, China, 2009.

[12] Zoltán Prekopcsák. *Accelerometer Based Real-Time Gesture Recognition*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.148.5590, 2008.

[13] Ross Cutler and Matthew Turk. View-based interpretation of real-time optical flow for gesture recognition. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 416–421, 1998.

[14] Laurenz Wiskott. *Learning Invariance Manifolds*. http://itb.biologie.hu-berlin.de/ wiskott/Abstracts/Wis98a.html, Proc. of the 5th Joint Symposium on Neural Computation, San Diego, CA, pp. 196-203, (1998).

[15] Pietro Berkes. *Pattern Recognition with Slow Feature Analysis*. http://cogprints.org/4104/, Institute for Theoretical Biology, Humboldt-University Berlin, 2005.

[16] Welf Walter. *Slow feature analysis, Universität Ulm*. http://www.informatik.uni-ulm.de/ni/Lehre/SS05/HauptseminarMustererkennung/ausarbeitungen/Walter.pdf, 2005.

[17] Pietro Berkes. *sfa-tk : Slow Feature Analysis Toolkit for Matlab (v.1.0.1)*. http://itb.biologie.hu-berlin.de/ berkes/software/sfa-tk/sfa-tk.shtml, 2003.

[18] Thomas Deselaers. *Image Retrieval, Object Recognition, and Discriminative Models*. Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen, 2008.

[19] MEMS. *www.st.com/stonline/books/pdf/docs/15094.pdf*. 2010, Datum des Abrufes: 06. Mai 2011.

[20] Cheng Yang. *Music Database Retrieval Based on Spectral Similarity*. Department of Computer Science, Stanford University, 2001.

[21] Mathieu Lagrange and Martin Raspaud. *Spectral similarity metrics for sound source formation based on the common variation cue*. Journal Multimedia Tools and Applications archive, Volume 48, Issue 1, May 2010, 2009.

[22] Dan Ellis. *Dynamic Time Warp (DTW) in Matlab*. http://www.ee.columbia.edu/ dpwe/resources/matlab/dtw/, Abruf im Februar 2012, 2003.

[23] Giorgio Tomasi, Thomas Skov, and Frans van den Berg. *Dynamic Time Warping (DTW) and Correlation Optimized Warping (COW)*. http://www.models.life.ku.dk/DTW_COW, Abruf im Februar 2012, 2006.

[24] Julia A. Lasserre, Christopher M. Bishop, and Thomas P. Minka. Principled hybrids of generative and discriminative models. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1*, pages 87–94, Washington, DC, USA, 2006. IEEE Computer Society.

[25] W. Strampp and E. V. Vorozhtsov. *Mathematische Methoden der Signalverarbeitung*. Oldenbourg Wissenschaftsverlag GmbH, 2004.

[26] Frank G. Hofmann, Peter Heyer, and Günter Hommel. Velocity profile based recognition of dynamic gestures with discrete hidden markov models. In *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, pages 81–95, London, UK, 1998. Springer-Verlag.

[27] Klaus D. Tönnies and Heinz U. Lemke. *3D-Computergrafische Darstellungen*. H.U.: Handbuch der Informatik Band 9.2. Oldenbourg Verlag, 1994.

[28] I.N. Bronstein, K. A. Semendjajew, G. Musiol, and H. Mühlig. *Taschenbuch der Mathematik*. Verlag Harri Deutsch, Frankfurt am Main, 7 edition, 2008.

[29] Steve Sandwine and Nicolas Le Bihan. *Quaternion toolbox for Matlab*. Département Images et Signal, GIPSA-Lab in Grenoble, France, 2005.

[30] Martin Werner. *Nachrichtentechnik*. Vieweg Verlag, 4. edition, 2003.

[31] Wilhelm Burger and Mark James Burge. *Digitale Bildverarbeitung : Eine Einführung mit Java und ImageJ*. Springer-Verlag, Berlin, Heidelberg, 2. überarbeitete auflage edition, 2005.