Template-basierte Klassifikation planarer Gesten

Dissertation
zur Erlangung des akademischen Grades
Dr. rer. nat.

vorgelegt im Monat März 2014 an der
Technischen Universität Dresden
Fakultät Informatik

eingereicht von
Dipl.-Inf. Michael Schmidt
geboren am 25. Oktober 1980 in Meerane

Gutachter:

Prof. Dr. rer. nat. habil. Gerhard Weber
Technische Universität Dresden
Fakultät Informatik, Institut für Angewandte Informatik
Professur Mensch-Computer Interaktion
01062 Dresden

Prof. Dr.-Ing. Bernhard Jung
TU Bergakademie Freiberg
Institut für Informatik
Professur Virtuelle Realität und Multimedia
09596 Freiberg

Tag der Verteidigung: **25. April 2014**

Dresden im Juni 2014

Abstract

Pervasion of mobile devices led to a growing interest in touch-based interactions. However, multi-touch input is still restricted to direct manipulations. In current applications, gestural commands - if used at all - are only exploiting single-touch. The underlying motive for the work at hand is the conviction that a realization of advanced interaction techniques requires handy tools for supporting their interpretation. Barriers for own implementations of procedures are dismantled by providing proof of concept regarding manifold interactions, therefore, making benefits calculable to developers. Within this thesis, a recognition routine for planar, symbolic gestures is developed that can be trained by specifications of templates and does not imply restrictions to the versatility of input. To provide a flexible tool, the interpretation of a gesture is independent of its natural variances, i.e., translation, scale, rotation, and speed. Additionally, the essential number of specified templates per class is required to be small and classifications are subject to real-time criteria common in the context of typical user interactions. The gesture recognizer is based on the integration of a nearest neighbor approach into a Bayesian classification method. Gestures are split into meaningful, elementary tokens to retrieve a set of local features that are merged by a sensor fusion process to form a global maximum-likelihood representation. Flexibility and high accuracy of the approach is empirically proven in thorough tests. Retaining all requirements, the method is extended to support the prediction of partially entered gestures. Besides more efficient input, the possible specification of direct manipulation interactions by templates is beneficial. Suitability for practical use of all provided concepts is demonstrated on the basis of two applications developed for this purpose and providing versatile options of multi-finger input. In addition to a trainable recognizer for domain-independent sketches, a multi-touch text input system is created and tested with users. It is established that multi-touch input is utilized in sketching if it is available as an alternative. Furthermore, a constructed multi-touch gesture alphabet allows for more efficient text input in comparison to its single-touch pendant. The concepts presented in this work can be of equal benefit to UI designers, usability experts, and developers of feedforward-mechanisms for dynamic training methods of gestural interactions. Likewise, a decomposition of input into tokens and its interpretation by a maximum-likelihood matching with templates is transferable to other application areas as the offline recognition of symbols.

Kurzfassung

Obwohl berührungsbasierte Interaktionen mit dem Aufkommen mobiler Geräte zunehmend Verbreitung fanden, beschränken sich Multi-Touch Eingaben größtenteils auf direkte Manipulationen. Im Bereich gestischer Kommandos finden, wenn überhaupt, nur Single-Touch Symbole Anwendung. Der vorliegenden Arbeit liegt der Gedanke zugrunde, dass die Umsetzung von Interaktionstechniken mit der Verfügbarkeit einfach zu handhabender Werkzeuge für deren Interpretation zusammenhängt. Auch kann die Hürde, eigene Techniken zu implementieren, verringert werden, wenn vielfältige Interaktionen erprobt sind und ihr Nutzen für Anwendungsentwickler abschätzbar wird. In der verfassten Dissertation wird ein Erkenner für planare, symbolische Gesten entwickelt, der über die Angabe von Templates trainiert werden kann und keine Beschränkung der Vielfalt von Eingaben auf berührungsempfindlichen Oberflächen voraussetzt. Um eine möglichst flexible Einsetzbarkeit zu gewährleisten, soll die Interpretation einer Geste unabhängig von natürlichen Varianzen - ihrer Translation, Skalierung, Rotation und Geschwindigkeit - und unter wenig spezifizierten Templates pro Klasse möglich sein. Weiterhin sind für Nutzerinteraktionen im Anwendungskontext übliche Echtzeit-Kriterien einzuhalten. Der vorgestellte Gestenerkenner basiert auf der Integration eines Nächste-Nachbar-Verfahrens in einen Ansatz der Bayes'schen Klassifikation. Gesten werden in elementare, bedeutungstragende Einheiten zerlegt, aus deren lokalen Merkmalen mittels eines Sensor-Fusion Prozesses eine Maximum-Likelihood-Repräsentation abgeleitet wird. Die Flexibilität und hohe Genauigkeit des statistischen Verfahrens wird in ausführlichen Tests nachgewiesen. Unter gleichbleibenden Anforderungen wird eine Erweiterung vorgestellt, die eine Prädiktion von Gesten bei partiellen Eingaben ermöglicht. Deren Nutzen liegt - neben effizienteren Eingaben - in der nachgewiesenen Möglichkeit, per Templates spezifizierte direkte Manipulationen zu interpretieren. Zur Demonstration der Praxistauglichkeit der präsentierten Konzepte werden exemplarisch zwei Anwendungen entwickelt und mit Nutzern getestet, die eine vielseitige Verwendung von Mehr-Finger-Eingaben vorsehen. Neben einem Erkenner trainierbarer, domänenunabhängiger Skizzen wird ein System für die Texteingabe mit den Fingern bereitgestellt. Anhand von Nutzerstudien wird gezeigt, dass Multi-Touch beim Skizzieren verwendet wird, wenn es als Alternative zur Verfügung steht und die Verwendung eines Multi-Touch Gestenalphabetes im Vergleich zur Texteingabe per Single-Touch effizienteres Schreiben zulässt. Von den vorgestellten Konzepten können UI-Designer, Usability-Experten und Entwickler von Feedforward-Mechanismen zum dynamischen Lehren gestischer Eingaben gleichermaßen profitieren. Die Zerlegung einer Eingabe in Token und ihre Interpretation anhand der Zuordnung zu spezifizierten Templates lässt sich weiterhin auf benachbarte Gebiete, etwa die Offline-Erkennung von Symbolen, übertragen.

Danksagung

Ich möchte mich bei meinem Betreuer Gerhard Weber bedanken, der nicht nur durch fachlichen Rat und dem nötigen sanften Druck einen großen Anteil an der erfolgreichen Fertigstellung meiner Dissertation hatte. Auch die Zusammenarbeit auf menschlicher Ebene, den immer fairen Umgang und das Bereiten des sehr angenehmen Arbeitsumfeldes weiß ich sehr zu schätzen. Ganz besonderer Dank gilt auch meiner besten Freundin, Frau und ausgezeichneten Lektorin Adriane für die kreativen gestalterischen Ideen und die sprachliche Aufwertung meiner Arbeit. Vor allem aber danke ich dir für die stete Motivation, den Halt und dafür, die wertvollste Bereicherung in meinem Leben zu sein. Meiner Familie danke ich für das konstante Vertrauen und den Rückhalt sowie die vielen Möglichkeiten, die mir immer offen standen. Speziell möchte ich meinen beiden Schwestern - Nora und Sandra - dafür danken, dass sie die Menschen sind, die sie sind. Sandra, du hattest außerdem großen positiven Einfluss auf meine Englisch-Kenntnisse und die sprachliche Qualität der Veröffentlichungen. Nicht vergessen möchte ich meine derzeitigen und ehemaligen Kollegen - Antje Elsner, Claudia Loitsch, Denise Prescher, Jens Bornschein, Jens Voegler, Kerstin Baldauf, Limin Zheng, Martin Spindler, Mei Miao, Michael Kraus, Thorsten Völkel und Ursula Weber - die zum oben genannten angenehmen Arbeitsumfeld gehören und nicht nur willkommene Ablenkung durch interessante Gespräche boten, sondern auch durch fachliche Diskussionen, Ratschläge und vor allem durch die nicht selbstverständliche Unterstützung und Entlastung eine große Hilfe waren. Außerdem danke ich den von mir betreuten Studenten, allen voran Anja Fibich, die durch ihre Entwicklungsarbeit die Umsetzung der interessanteren Ideen erst ermöglichten. Zu guter Letzt gilt mein Dank auch den vielen engagierten Personen, die ihre Zeit zur Verfügung stellten und als Probanden an den Studien teilgenommen haben.

Michael Schmidt
Dresden, 30. Juni 2014

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1

Einleitung und Motivation

Berührungsempfindliche Oberflächen fanden ihre Verwendung bei der Interaktion zwischen Mensch und Maschine schon vor der Entwicklung des Computers, Touchscreens existieren seit den 1960er Jahren [25]. Der Einzug von Gesten auf berührungssensitiven Flächen in die Mensch-Computer Interaktion kann ebenfalls in die frühen 1960er Jahre zurückverfolgt werden [138]. Anfang der 1990er Jahre gab es bereits Gesten-basierte Betriebssysteme für Tablets, wie das PenPoint OS [30], welches neben direkten Manipulationen (Drag & Drop) auch die Interaktion mittels Gesten unterstützte. Mittlerweile existiert eine Vielzahl einzelner Anwendungen, die gestische Eingaben anbieten. Die Software zur Gestenerkennung 'xstroke' [220] stellt eine Methode für die Texteingabe unter dem 'X Window System' zur Verfügung. Mittels 'StrokeIt'[1] werden mehr als 80 verschiedene Mausgesten für die Verknüpfung mit Aktionen unter Windows-Systemen bereitgestellt. Gestische Steuerungen per Maus finden sich ebenso in Webbrowsern[2], aber auch in Anwendungen für Flugverkehrskontrollen [33]. Im Allgemeinen vorteilhaft an Gesten ist dabei die Angabe eines Kommandos und dessen Parameter in einer Eingabe sowie die Lernförderlichkeit durch eine die Syntax kommunizierende Form [219].

Die oben genannten Systeme sind für die Bedienung per Maus- oder Stifteingabe bzw. einem Finger ausgelegt. Konzepte zur Interaktion mittels Multi-Touch finden sich seit den frühen 1980er Jahren [25]. Den Durchbruch alternativer Bedienungen zum klassischen WIMP Paradigma in ubiquitäre Systeme begünstigten allerdings wesentliche Entwicklungen der letzten Jahre. Jeff Han [74] entwarf mit vergleichsweise günsti-

[1] `http://www.tcbmi.com/strokeit/`

[2] Nativ in Opera: `http://www.opera.com/browser/tutorials/gestures/`
'FireGestures' für Firefox: `http://www.xuldev.org/firegestures`

gen Mitteln einen Multi-Touch-fähigen Touchscreen und erzeugte ein reges Interesse in Wissenschaft und Wirtschaft. Die Einführung des 'iPhone' im Jahr 2007 erschloss einen Massenmarkt und der im gleichen Jahr erschienene 'Microsoft PixelSense' (ehemals 'MS Surface') trug ebenfalls zu einer weiten Verbreitung des Bedienparadigmas bei.

Allerdings beschränken sich derzeitige Interaktionen meist auf direkte Manipulationen, die Verschiebe-, Rotations- oder Zooming-Operationen steuern. Selten findet eine Interpretation der Eingabe abseits dieser Abbildung von Punkten der Anwendung auf sich bewegende Kontakte über der berührungsempfindlichen Oberfläche statt. Von der Möglichkeit, auch Sequenzen von Berührungen mit mehreren Fingern als geschlossene Eingabe zuzulassen, wird nicht Gebrauch gemacht. Komplexere Formen der Multi-Stroke Stifteingabe finden sich nur im Bereich der Interpretation von Handschrift und in Anwendungen zum Skizzieren.

Auch wenn symbolische Single-Touch Gesten in diversen Applikationen in Gebrauch sind, so ist dem Autor kein Gestenset unter Verwendung von Multi-Touch bekannt. Ebenso kritisieren Webel et al. [207], dass derzeitige Forschung im Bereich der gestischen Interaktion auf eine sehr begrenzte Auswahl dieser manipulierenden Interaktionen beschränkt ist. Freeman et al. [59] sehen den Grund fehlender komplexerer Multi-Touch Interaktionen in kommerziellen Anwendungen im zu hohen notwendigen Lernaufwand für den Anwender. Die Bereitstellung komplexer, gestischer Eingaben ist zudem an die Implementierung zugeschnittener Erkenner und einen dementsprechend höheren Entwicklungsaufwand gebunden. Neue Konzepte für Interaktionen können in Betracht gezogen, untersucht und getestet werden, wenn eine solche Komponente zur Verfügung steht.

Die vorliegende Arbeit soll diese Lücke schließen und einen universellen Klassifizierer für symbolische Gesten auf berührungsempfindlichen Oberflächen herleiten. Der Klassifizierer soll per Spezifikation der Templates gewünschter Gesten trainierbar sein und eine größtmögliche Vielfalt an Gesten unterstützen. Ein auf diese Weise trainierbarer Gestenerkenner erlaubt die Anpassung der Gesten an besondere Anforderungen und bietet demzufolge Vorteile für Personengruppen mit Einschränkungen. Beispielsweise wird in [90, 134] demonstriert, dass Touchscreens mobiler Geräte durch angepasste Gestensets für Blinde zugänglich gemacht werden können. Zudem wird Nutzern auch die eigene Spezifikation präferierter Gesten [123] ermöglicht. Eine Untersuchung in [218] zeigte zwar, dass Nutzer bei der eigenständigen Auswahl von Gesten für verschiedene Editieroperationen in 68% der Fälle konsistent die gebräuchlichsten Varianten bevorzugten, im Umkehrschluss bedeutet das allerdings, dass für 1/3 der Nutzer diese üblichen Gesten nicht die erste Wahl sind. Abseits der für Designer ohne Implementierungsaufwand durchführbaren Definition von Gestensets werden des Weiteren vergleichende Evaluationen [216, 143] durch Usability-Experten und das Prototyping sowie die Weiterentwicklung gestischer Interfaces erleichtert.

Neben der Vergrößerung der Vielfalt und damit des unterscheidbaren Repertoires, können unter Verwendung von Multi-Touch über Gesten mehr Informationen übertragen und damit Interaktionszeiten verkürzt werden. Texteingabesysteme auf Basis von Gesten [84, 215, 32] könnten davon profitieren. Es existieren noch eine Vielzahl weiterer Anwendungen, für die eine leistungsfähige Gestenerkennung potenziell Vorteile bringt. So sind auch die Erkennung von Skizzen [111, 81] oder mathematischen Symbolen [206] verwandte Gebiete, auf welche die vorgestellten Ansätze ebenfalls übertragbar sind.

Ein weiterer berücksichtigter Aspekt ist das Lernen von Gesten durch den Nutzer. Obwohl in einem kleinen Test mit 13 Nutzern gezeigt wurde [219], dass Gesten auch nach einer Woche noch im Gedächtnis behalten werden, ist nach [59] der initiale Lernaufwand eine Hürde. Auch da in [218] durch Nutzer selbstgewählte Gesten für verschiedene Operationen mehrdeutig spezifiziert wurden, sind Konzepte zum Unterstützen des Lernens, für Erinnerungshilfen und für Feedback- oder Feedforward-Systeme notwendig. Das entwickelte Verfahren zur Klassifikation von Gesten soll demzufolge auch partielle Eingaben frühest möglich einer Interpretation zuweisen können, um die Grundlagen für Realisierungen derartiger Rückmeldungen oder Hilfestellungen anzubieten.

Der Beitrag dieser Arbeit liegt neben der Herleitung eines Verfahrens, welches die oben genannten Einsatzmöglichkeiten bietet, in der ausführlichen Systematisierung und Einordnung der Konzepte der Gestenklassifikation und in der Literatur beschriebener Ansätze. Bestehende Taxonomien für Gesten werden untersucht und eigene Begriffsformalisierungen für die Bedürfnisse der Arbeit abgeleitet. Durch eine getrennte Betrachtung der nicht zwangsläufig unabhängigen Schritte in der Gestenklassifikation und die klare Aufteilung der Merkmalsgewinnung in einen Segmentierungs- und einen Extraktionsprozess, können bestehende und vorgestellte Methoden besser eingeordnet werden. Um die Nützlichkeit und die Anwendbarkeit der umgesetzten Verfahren einschätzen zu können, werden Untersuchungen durch umfangreiche Testszenarien begleitet, der sogenannte 'Proof of Concept' aber auch durch die Erschließung eigener Anwendungsgebiete bekräftigt.

1.1 Anforderungen und Ziele

Ziel der Arbeit ist die Entwicklung eines Systems zur Erkennung von Multi-Touch Gesten, welches um selbst definierte Gesten erweiterbar ist. Die Erkennung soll unabhängig von der Art der Eingabe sein und auf den Trajektorie-Daten basieren. Derzeit sind Single-Touch Eingaben durch Stift oder Finger gebräuchlich, während Multi-Touch Interaktionen für direkte Manipulationen genutzt werden und symbolische Eingaben mit (sequenziellem) Multi-Touch keine Verwendung finden. Dies wird auch darauf zurückgeführt, dass Softwareentwicklern kein Werkzeug in die Hand gegeben ist, derartige Gesten mit wenig Aufwand zu klassifizieren. Ein solches Werkzeug soll das Ergebnis der vorliegenden Arbeit sein.

Um in praktischen Anwendungen einsetzbar zu sein, sind folgende Anforderungen zu erfüllen:

- **Robustheit**: Die Erkennung sollte invariant gegenüber Translation, Skalierung, Rotation und Geschwindigkeit sein, ohne vom Nutzer das Zeichnen idealisierter Gesten zu erwarten. Natürliche Varianzen bei der menschlichen Eingabe sollen die Erkennung ebensowenig beeinflussen wie Trainingseffekte, etwa kleinere und/oder schnellere Ausführungen.

- **Natürlichkeit**: Bei angenommenen bekanntem Start und Ende der Geste sollen keine Einschränkungen für die Form der Eingabe planarer Gesten bestehen.

- **Skalierbarkeit**: Das Gestenset soll vom Nutzer selbständig mittels Training von wenigen Templates erweiterbar sein. Auch für den Anwendungsentwickler hat diese Art der Erweiterbarkeit Vorteile, wenn keine Einzelbehandlungen für jede weitere Geste implementiert werden müssen (wie etwa im Sparsh-UI Framework[3]).

- **Performance**: Die Erkennung soll in Echtzeit geschehen, das heißt für die Anforderung bei UI Interaktionen sollen keine bemerkbaren Verzögerungen entstehen. Miller benennt in [136] 'calculated guesses' der maximalen Antwortzeit auf Steuerbefehle mit 100 ms.

Der Argumentation in [99], nach der neben Skalierung und Rotation auch die bildliche Spiegelung und entgegengesetzt gerichtete Eingaben Variationen von Gesten darstellen, wird hier nicht gefolgt. Die Form, der zeitliche Verlauf in ihrer Erzeugung und Spiegelungen werden als Kriterien angesehen, die Gesten voneinander unterscheiden. Nach [122] trägt allerdings die Größe eines Symbols im Vergleich zur Form zur von Nutzern empfundenen Ähnlichkeit mit einem anderen wenig (logarithmisch) bei. Auch in [8] wird die Beobachtung gemacht, dass bei Eingaben von Gesten aus dem Gedächtnis die Skalierung eine hohe Standardabweichung besitzt und ein Gestenerkenner demzufolge robust bezüglich dieser Variationen sein sollte. Die Position und die Orientierung einer Geste variieren mit der Position des Nutzers zur sensitiven Oberfläche und sollen daher bei der Klassifikation hier ebenfalls keine Bedeutung tragen. Unter dem Aspekt der Merkmalsgewinnung beschreibt Kozlay [104, S. 1] neben additivem Rauschen auch die Beeinflussung der Daten durch zufällige Translation unbekannter Verteilung und der zufälligen Rotation jeweils mit begrenzten Magnituden. Daher stellen entsprechende Invarianzen auch hier ein Kriterium der Robustheit dar. Werden solche Unterscheidungen dennoch gewünscht, können diesbezüglich Parameter zurückgegeben und von einer Anwendung interpretiert werden. Konzepte, um Anfang oder Ende einer Geste festzulegen - etwa Modi, Zeitsteuerung, Restriktionen der Eingabe - oder zu erkennen - etwa durch

[3] `http://code.google.com/p/sparsh-ui/`

Koartikulation[4], Interpretation der Bedienzeiten - sind nicht Gegenstand dieser Arbeit und werden daher nicht weiter untersucht.

1.2 Aufbau der Arbeit

Zunächst werden bestehende Taxonomien zu Gesten untersucht, Begrifflichkeiten festgelegt und erweitert, um die Arten der Eingabe zu systematisieren und das Gebiet abzugrenzen. Danach wird anhand des Standes der Technik einzelner Komponenten eine allgemeine Architektur der Gestenerkennung im Detail spezifiziert.

Unter Einbezug verwandter Gebiete - etwa die Erkennung von Symbolen und Skizzen - werden in der Literatur verfügbare Verfahren für die Klassifikation Trajektoriebasierter Eingaben analysiert und kategorisiert sowie eine Systematik der für die Repräsentation von Gesten geeigneten Merkmale geschaffen.

Nach den Erkenntnissen aus dem Stand der Technik werden Methoden der Nächste-Nachbar-Klassifikation unter - nicht auf Interpretationen basierenden oder durch Quantisierung verrauschten - lokalen Merkmalen als vielversprechend angesehen und näher untersucht. Der Autor systematisiert bestehende Verfahren unter dem generischen Konzept des Dynamic Time Warpings anhand verwendeter Shape-Signaturen und unterzieht diese - unter Einbezug eigener Methoden und verschiedener Parametrisierungen - einem ausführlichen Vergleich. Die Ergebnisse dieser Tests an drei unähnlichen Gestensets zeigen geeignete Klassifikationsmethoden für Single-Touch Eingaben auf.

Die Herleitung und Entwicklung eines vielseitigen Gestenklassifizierers stellt den Kern der Arbeit dar. Unter Ausnutzung des Zusammenhangs zwischen der statistischen Klassifikation und speziellen Nächste-Nachbar-Methoden wird ein kombinierter Ansatz für die Interpretation (sequenzieller) Multi-Touch Eingaben hergeleitet. Dazu wird die Verknüpfung lokaler Informationen multipler Berührungen zur Repräsentation der globalen Struktur als ein Problem der Sensor-Fusion angesehen und mittels eines Maximum-Likelihood-Matchings in den Ansatz der Bayes'schen Klassifikation integriert. Weiterhin werden die Möglichkeiten der Parameterwahl untersucht und die zugrunde gelegten Annahmen kritisch betrachtet. Aufgrund mangelnder Verfügbarkeit komplexer Gestensets werden Tests an konstruierten Multi-Touch Gesten durchgeführt, welche die Praxistauglichkeit des Verfahrens empirisch belegen.

Die Klassifikationsmethode wird anschließend um Funktionalitäten ergänzt, so dass unter gleichbleibenden Anforderungen eine Prädiktion möglich wird. Die Erkennung von Gesten unter partiellen Eingaben und die frühestmögliche Zuweisung einer möglichst genauen Interpretation lässt auch die Umsetzung per Templates spezifizierter direkter Manipulationen zu. Abermals weisen Tests an zwei Gestensets die Anwendbarkeit des Ansatzes nach.

[4] Die nach einer Gesten geplante Bewegung kann nach [214] schon die aktuelle Ausführung gegen Ende der Geste beeinflussen.

Den Abschluss der Arbeit bildet die Vorstellung zweier Anwendungen, anhand derer ein exemplarischer Proof of Concept für die brauchbare Verwendung (sequenzieller) Multi-Touch Eingaben und ihrer praxistauglichen Klassifikation erbracht werden. Um die Flexibilität des entwickelten Gestenerkenners zu demonstrieren, wird zum einen ein Erkenner für Skizzen vorgestellt, der trainierbare Primitive unter beliebigen, vom Nutzer festgelegten Ausführungen interpretiert. Die im Vergleich zu bestehenden Anwendungen vielseitigeren Eingaben werden mit wettbewerbsfähigen Genauigkeiten klassifiziert. Eine Nutzerstudie zeigt, dass Multi-Touch beim Skizzieren verwendet wird, wenn es als Alternative zur Verfügung steht. Zum anderen wird ein Texteingabe-System vorgestellt, welches unter Verwendung eines Multi-Touch Gestenalphabets im Vergleich zur Texteingabe per Single-Touch effizienteres Schreiben zulässt (Nachweis in zwei Nutzerstudien).

2

Vorbetrachtungen und Definitionen

Im vorliegenden Kapitel sollen Erläuterungen und Abgrenzungen für die in dieser Arbeit verwendeten, grundlegenden Begriffe erfolgen. Dafür werden vorangegangene Arbeiten aufgegriffen, aber auch eigene Definitionen eingeführt, um ein Verständnis für die Einordnung der Thematik zu schaffen.

2.1 Taxonomien von Gesten

Eine Arbeit, die sich das Erkennen von Gesten zum Ziel setzt, sollte auch festlegen, was genau unter diesem Begriff zu verstehen ist. Das Ziel dieses Abschnittes ist zum einen die Abgrenzung des Gestenbegriffes und seine Einordnung in den Kontext der Mensch-Computer Interaktion (MCI). Zum anderen soll anhand dieser Einordnung der Fokus der vorliegenden Arbeit aufgezeigt werden. Man sollte meinen, die Definition einer Geste sei kein schwieriges Unterfangen. Im natürlichen Sprachgebrauch haben die meisten Menschen eine Vorstellung davon, was mit 'Geste' gemeint ist. Üblicherweise wird man eine Geste als eine bedeutungstragende Bewegungen des menschlichen Körpers zum Zwecke der (zwischenmenschlichen) Kommunikation verstehen. Bewegungen des Gesichts werden dabei unter dem Begriff der Mimik gesondert betrachtet. In dem Gebiet der Mensch-Computer Interaktion ist der Begriff der Geste zudem für Kommandos per Stift-, oder Fingereingaben auf einem berührungsempfindlichen Display gebräuchlich. Auch Mausgesten, etwa zur Browsersteuerung, haben sich im Sprachgebrauch etabliert.

In der Literatur findet sich eine Definition des Begriffs 'Geste' meist nur indirekt über eine Typisierung. McNeill [135, S. 12-18] betrachtet Gesten aus der linguistischen Sicht mit Einschränkung auf koverbale Bewegungen der Hände und Arme und Unterscheidung in:

- **Ikonisch**: Diese bildhafte Form der Gesten steht in enger Beziehung zum semantischen Inhalt der Sprache und veranschaulicht ein mentales Modell (z.B. Nachahmung von Aktionen, Bewegungen).
- **Metaphorisch**: Ebenso wie die ikonischen Gesten dienen metaphorische der bildhaften Darstellung, repräsentieren aber eher abstrakte Ideen statt konkreter Ereignisse oder Objekte (z.B. Umgreifen eines imaginären Objektes um 'das Ganze' auszudrücken).
- **Deiktisch**: Deiktische Gesten beinhalten das Zeigen realer Objekte oder Ereignisse, aber auch das abstrakte Konzept des Zeigens.
- **Takt** (engl. beats): Diese Geste dient der rhythmischen Untermalung des Gesprochenen in einer zwei-phasigen Bewegung (z.B. Wippen mit der Hand).
- **Kohesiv**: Im Gegensatz zum Takt signalisieren diese Gesten Kontinuität und tragen die Bedeutung im wiederholten Ausführen.

Nach dieser Definitionen existiert eine Geste nicht nur als Bewegung, sondern steht immer im Kontext zum (evtl. unbewussten) Zweck der Kommunikation. Eine gleiche Ausführung bedeutet nicht zwangsläufig die gleiche Geste oder den gleichen Gestentyp. Außerdem werden Gesten, da sie keine Linearisierung der von ihnen getragenen, komplexen Information über die Zeit darstellen, zu Sprachen, auch Zeichensprachen, die einer Syntax unterliegen, abgegrenzt [135, S. 19].

Die Sicht auf Gesten im Kontext der Mensch-Computer Interaktion unterscheidet sich nicht wesentlich von der aus linguistischer oder psychologischer Perspektive. Wexelblat [210] stellt verschiedene in der Literatur existierende Taxonomien von Gesten gegenüber. Er trägt fünf Kategorien (ikonisch, metaphorisch, Takt, symbolisch, deiktisch) zusammen, die unter verschiedenen Begriffen mehr oder weniger konsistent abgegrenzt in der Literatur Verwendung finden. Zu den bereits genannten kommt die Kategorie der symbolischen Gesten hinzu, welche gemeinhin bekannte Gesten zusammenfassen, denen eine feste Bedeutung zugewiesen ist. Die kohäsiven Gesten werden nicht getrennt betrachtet. Es werden unter den Takt-Gesten alle zusammengefasst, welche den Rhythmus der Sprache unterlegen. Allerdings bemängelt Wexelblat die eingeschränkte Anwendbarkeit der Taxonomien, da sie eine wenig scharfe Trennung und keine Regeln für eine Einteilung liefern. Ebenso wenig werde deutlich, was nicht unter den Begriff der Geste zu fallen hat, was sich auf eine dahingehend fehlende Definition reduzieren lässt.

Auch Wilson und Bobick [214, S. 1] beschreiben eine Geste nur allgemein als: 'a motion that has special status in a domain or context'. In [201, S. 313] wird zwischen visuell bedeutsamen Gesten und abstrakteren Arten ohne Bezug von Ausführung und Semantik unterschieden.

Dass eine engere Abgrenzung notwendig ist, lässt sich leicht im Gebiet der Virtual Reality (VR) demonstrieren. Wenn ein Erzähler zur Veranschaulichung das Betätigen einer Türklinke imitiert, so wird dies gemeinhin als Geste (ikonisch) eingestuft. Die Betätigung einer Türklinke selbst dagegen nicht. Wie ist nun damit umzugehen, wenn Interface Designer die Metapher einer Tür in einer VR Umgebung einsetzen und ein Nutzer nun exakt diese, die virtuelle Türklinke manipulierende Bewegung ausführt? Aus Sicht des Systems kommuniziert der Nutzer über diese Bewegung, aus Nutzersicht kann diese Grenze aber leicht verschwimmen. Eine Möglichkeit ist, Gesten abstrakter nach ihrem Anwendungsgebiet einzuteilen. Nach Edwards [51, S. 13] werden folgende Kategorien empfohlen:

- **Natürliche Gesten**: Hierunter fallen Gesten, die meist zwischenmenschliche Kommunikation begleiten und im Affekt ausgeführt werden.
- **Synthetische Gesten**: Diese Gesten werden in der Mensch-Computer Interaktion benutzt und sind dementsprechend auch für die spezielle Anwendung und die leichtere Erkennung angepasst.
- **Virtual Reality Interaktion**: Diese Gesten versuchen Handlungen in der realen Welt nachzubilden und stellen die mit realen Objekten assoziierten Bewegungen nach.

Eine aktuellere Arbeit, in der die mehrdimensionale Kategorisierung aufgegriffen und versucht wird, eine einheitliche Taxonomie Gesten-basierter Interaktion unter anderem aus bereits existierenden Studien abzuleiten, findet sich in [92]. Die in Abbildung 2.1 dargestellte Taxonomie unterteilt Gesten-basierte Interaktionen zunächst nach ihrer Anwendungsdomäne und des Weiteren nach möglichen Instrumenten, der Art des Feedbacks und letztlich auch dem Typ der Geste [92, S. 1-8]. Die Instrumente - oder aktivierenden Technologien - berücksichtigen die Sichten der kommunizierenden und der empfangenden Instanz.

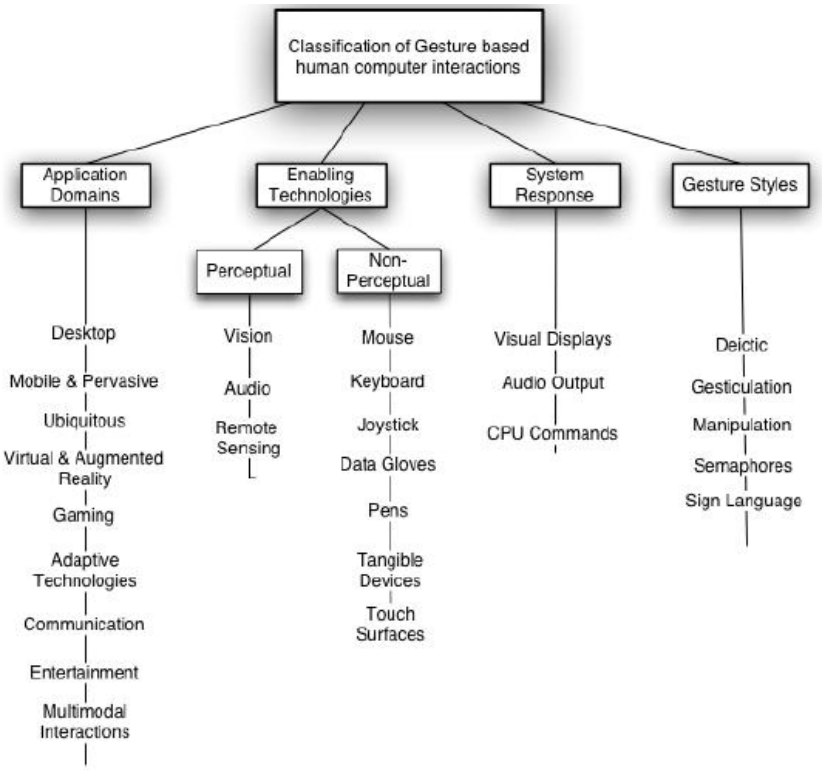


Abbildung 2.1: Eine Taxonomie von Gesten nach [92], bei der mehrdimensional nach Anwendungsdomäne, aktivierender Technologie, Antwort des Systems und den Gestenarten kategorisiert wird.

Die Typisierung der Gesten (siehe Abbildung 2.1) geschieht unter dem Nutzungskontext der MCI und unterscheidet, neben Mischformen, nach:

- **Deiktische** Gesten dienen zum Identifizieren oder der Vermittlung der Position eines Objektes und können implizit in anderen Gesten enthalten sein.
- **Manipulative** Gesten liegen bei einer engen Kopplung der Bewegung zu einem manipulierten Objekt vor und wenn die Bewegung, wie in [171] behandelt, Parameter für die Manipulation beisteuert.
- **Semaphorische** Gesten können durch ein festgelegtes Set statischer Positionen oder dynamischer Bewegungen repräsentiert werden. Darunter fallen beispielsweise Browser-Gesten oder Graffiti [128], aber auch Handzeichen (OK-Symbol).
- Unter **Gestikulation** fallen koverbale Gesten, die im Kontext der Sprache analysiert werden müssen. Somit werden auch die in [135] unter ikonisch bzw. metaphorisch eingeordneten Gesten zur Gestikulation.
- Im Gegensatz zu [135] wird **Zeichensprache** als spezielle Form der Gesten gesehen. Obwohl sie sich auch an Symbolen orientiert, wird sie aufgrund ihrer Besonderheiten nicht der Kategorie der semaphorischen Gesten zugeordnet.

Die obige Einteilung legt implizit eine Systemsicht fest. Der Zweck und Aufwand der Interpretation des Zielsystems einer Geste definieren die Einordnung. Auch McNeill

ordnet gleich ausgeführte Bewegungen anhand ihrer Bedeutung unterschiedlichen Kategorien von Gesten zu. Hier erscheint allerdings der Zweck vordergründiger. Gesten dienen als Sprache, Sprachbegleitung, Zeigehandlung, Parametrisierung direkter Manipulation oder direkt als Symbol. Diese Sichtweise soll hier aufgegriffen werden. Eine strengere Unterscheidung der semaphorischen Gesten in kontextabhängige ikonische und eine feste Bedeutung tragende symbolische Gesten in Anlehnung an [210] ist nicht vorgesehen. Obwohl diese zusätzliche Betrachtung dem Verfasser dieser Arbeit für andere Anwendungsgebiete sinnvoll erscheint wird sie an dieser Stelle nicht aufgegriffen und im weiteren Verlauf der Begriff 'symbolische Gesten' stellvertretend für diese Kategorien verwendet.

Interessanterweise nimmt mit der Natürlichkeit für den Nutzer der Interpretationsaufwand für einen Beobachter zu. Manipulative oder deiktische Gesten stellen einen Ortsbezug her und benötigen nahezu kein Kontextwissen. Symbolische Gesten können über Wörterbücher identifiziert werden und Zeichensprache benötigt eine zusätzliche Grammatik. Gestikulation sind dagegen nur im Kontext von Sprache zu interpretieren. Eine weitere mögliche Einordnung könnte demnach nach der für den Nutzer inhärenten Natürlichkeit und für das interpretierende System dem Grad des nötigen Kontextwissens sein.

Die Einführung der manipulativen Gesten ist dem Kontext der MCI geschuldet. Es werden in [92] nur diejenigen Manipulationen als Gesten betrachtet, bei denen ein Kommando ausgelöst wird. Eine Drag & Drop Operation mit der Maus würde demnach keine Geste sein. Allerdings wird die Auswahl eines Objektes gefolgt von seiner Verschiebeposition über Klickoperationen als Geste angesehen. Aus Nutzersicht ist eine solche Unterscheidung nicht zwingend offensichtlich. Der Systementwickler hingegen wird komplexe Interpretationen eher als Gestenerkennung auffassen, als das direkte Mapping von speziellen Punkten eines Anwendungsobjektes auf die Positionierungsdaten des Eingabegerätes. Klarer wird die Abgrenzung, wenn man das von Shneiderman [192, S. 64] mit folgenden Eigenschaften definierte User Interface Paradigma der direkten Manipulation heranzieht:

- Das interessierende Objekt ist fortwährend präsent.
- Aktionen werden durch physikalische Vorgänge (Verschieben, Selektieren, Drücken von Schaltflächen) statt einer komplexen Syntax definiert.
- Sofortige, inkrementelle Rückmeldung über die Auswirkung von umkehrbaren Aktionen sind am betreffenden Objekt sichtbar.

Auch wenn der Begriff der Geste für direkte Manipulationen geläufig ist, werden sie hier nicht darunter gefasst. Gesten sind im Gegensatz dazu atomar und werden abschließend interpretiert. *Eine Geste ist somit eine Körperbewegung*[5]*, die in ihrer Gesamtheit Informationen kodiert, die für ein beobachtendes System interpretierbar sind.*

[5] Im Kontext der MCI sind häufig Bewegungen des Armes oder der Hand gemeint [150].

2.2 Planare Gesten

Planare Gesten stellen eine Untermenge der Gesten dar. Im Allgemeinen geschieht dies unter der aktivierenden Technologie berührungsempfindlicher Oberflächen (mit Fingern, Stylus, o.a.). Da sie nicht natürlich sind, werden an dieser Stelle planare Pendants zu Gestikulationen oder sprachunterstützenden Gesten ausgegrenzt. Planare Gesten können demzufolge deiktisch, symbolisch oder manipulativ sein. Prinzipiell können diese Gesten auch als zweidimensionale Projektionen von Gesten verstanden werden, bei denen nur diskrete 'Berührungspunkte' beobachtet werden bzw. von Interesse sind.[6] Aus Nutzersicht verschiedentliche Bewegungen oder aktivierende Technologien (Stift-, Fingereingabe) können aus Systemsicht gleiche Gestenkategorien bedeuten. Dennoch wird das Bewusstsein des Nutzers gegenüber einer solchen Projektion vorausgesetzt und die Geste letztendlich über diese definiert. Im Gebrauch sind planare Gesten bisher üblicherweise als symbolische Kommandos ergänzend zu Point & Click Interaktionen oder direkten Manipulationen.

In Anlehnung an [92] definiert sich eine planare Geste hier über die Trajektorien haptischer Interaktion, die als Zeigehandlung, symbolisches Kommando oder Parameter einer Manipulation interpretiert werden. Direkte Manipulationen, wie etwa in [110], werden entgegen der geläufigen Begrifflichkeit weiterhin nicht als Gesten aufgefasst.

In Abbildung 2.2 sind beispielhaft typische Vertreter dieser Gestenformen zusammengestellt.

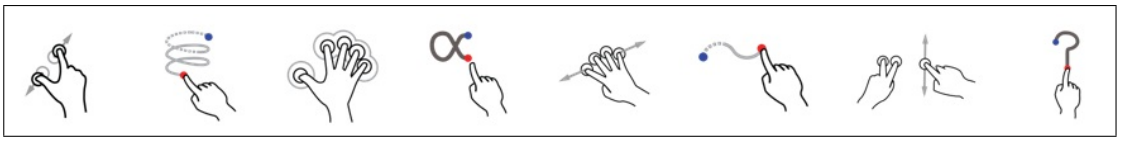


Abbildung 2.2: Beispiele planarer Gesten aus der 'Open Source Gesture Library' [64].

Angelehnt an die vorhergehende Diskussion werden in dieser Arbeit planare symbolische, manipulative und deiktische Gesten voneinander abgegrenzt. Die vorliegende Arbeit beschäftigt sich mit der Klassifikation von planaren symbolischen Gesten, also all jenen Gesten, deren Form bzw. ganzheitlicher Verlauf die zu interpretierenden Informationen trägt. Deiktische Gesten, bei denen nicht die Form der Bewegung bzw. deren ganzheitlicher Verlauf interpretiert wird (z.B. die Lasso-Geste[7] zur Selektion) oder manipulative Gesten werden nicht weiter betrachtet.

Im Zusammenhang mit planaren Gesten sollen die unter [183] veröffentlichten Begriffsdefinitionen herangezogen werden. Danach besteht eine Geste[8] aus Strokes. In der Literatur wird der Begriff Stroke meist als die von einem Kontakt auf der berührungsempfindlichen Fläche erzeugte Trajektorie (Pfad zwischen Kontaktaufnahme und Auf-

[6] So werden in [106] räumliche Gesten erkannt, deren Pfade sich auf einer Ebene bewegen. Für das zur Erkennung verwendete Verfahren ist die Herkunft solcher planaren Gesten nicht relevant.

[7] Die Zeigehandlung umschreibt in diesem Fall einen Bereich und nicht nur eine diskrete Position.

[8] Im folgenden werden planare Gesten und der Spezialfall der planaren symbolischen Gesten als Gesten bezeichnet. Im Kontext sollte deutlich werden, welche Form gemeint ist.

heben des Kontaktes) gesehen. Sobald Gesten mit mehreren gleichzeitigen Kontakten - etwa durch Mehrfinger-Berührungen - vorgesehen werden, bietet sich eine Erweiterung des Begriffes an. Hier wird eine Taxonomie der Geste anhand ihrer im Folgenden aufgeführten Bestandteile aus Sicht des interpretierenden Systems zu Grunde gelegt:

- **Trajektorie**: Eine Datenreihe von zeitgestempelten Positionsdaten, die den Verlauf einer Berührung von seinem Anfang bis zum Beenden des Kontaktes beschreiben.
- **Stroke**: Bezeichnet einen Abschnitt der Gesten, innerhalb dessen zu jedem Zeitpunkt wenigstens ein Kontakt die sensitive Fläche berührt. Er wird begrenzt durch berührungsfreie Phasen.
- **Single-Touch**: Eine Geste benötigt nur einen dauerhaften Kontakt mit der berührungsempfindlichen Fläche, wie er bei der Eingabe mit Stift, etwa beim Skizzieren oder der Handschrift vorliegt. Die Geste wird demnach durch genau eine Trajektorie beschrieben.
- **Multi-Stroke**: Eine Geste enthält mehrere Strokes, die keinen gleichzeitigen Kontakt besitzen. Eine solche Geste kann auch als Sequenz von Single-Touch Gesten gesehen werden. Gesten mit mehreren Strokes brauchen eine gesonderte Beginn- und Ende-Erkennung, da diese Information nicht über Beginn und Ende des Kontaktes impliziert wird.
- **Multi-Touch**: In der Ausführung der Geste sind mehrere gleichzeitige Berührungspunkte vorgesehen. Die Hardware muss zeitgleiche Berührungen detektieren und lokalisieren können. Da dem Wissen des Autors nach in der Literatur derzeit keine solchen Gesten beschrieben wurden, die aus mehreren Strokes bestehen, wird hier ebenso der geläufige Gebrauch des Begriffes übernommen, welcher eine Single-Stroke Eingabe vorsieht.
- **Sequenzieller Multi-Touch**: Gesten, die sowohl aus mehreren Strokes bestehen, als auch innerhalb wenigstens eines Strokes mindestens zwei gleichzeitige Berührungen aufweisen, werden hier als sequenzielle Multi-Touch Gesten bezeichnet. Diese Gestenform kann als aus den anderen Arten komponiert gesehen werden und nutzt potenziell alle Freiheiten einer Eingabe auf berührungssensitiven Oberflächen.
- **Token**: Token bezeichnen elementare Bestandteile einer Geste, in die eine Geste zum Zwecke der Erkennung zerlegt wird. Im einfachsten Fall sind das die Trajektorien, aber ebenso sind feinere Segmente, aus denen dann Merkmale gewonnen werden, möglich.

Die obige Taxonomie wird in Abbildung 2.3 noch einmal illustriert.

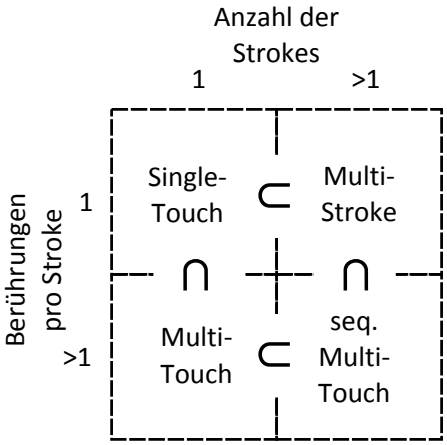


Abbildung 2.3: Veranschaulichung der Gesten-Taxonomie nach Anzahl der Strokes und gleichzeitigen Berührungspunkten wie in [183] veröffentlicht. Die Teilmengenbeziehungen verdeutlichen, in welcher Gestenform andere im Sinne von Teilgesten oder Komplexität implizit enthalten sind.

Die Taxonomie aus Abbildung 2.3 wird in dieser Arbeit zugrunde gelegt, um die möglichst vielseitige Erkennung beliebiger Gestentypen anhand nutzerdefinierter Beispiele anzustreben. Eine Taxonomie der Ausführung von Multi-Touch Gesten nach der Art der Berührungen und des Verlaufes findet sich in [59] und ist in Abbildung 2.4 wiedergegeben.

Registration Pose	*Single Finger*	Initial touch with a single finger
	Multi-Finger	Initial touch with multiple fingers
	Single Shape	Initial touch with a single hand shape ('blob') (e.g., a palm down)
	Multi-Shape	Initial touch with multiple hand shapes (typically bimanual)
Continuation Pose	*Static*	Hand pose remains the same after registration: no relative movement
	Dynamic	Hand pose changes after registration (e.g., new fingers come in contact with the surface)
Movement	*No path*	Hand stays in place
	Path	Hand moves along a surface path

Abbildung 2.4: Die Taxonomie planarer Multi-Touch Gesten anhand ihrer Eigenschaften aus Sicht des Nutzers nach [59].

Die Taxonomie orientiert sich an realistischen Gesten, mit Merkmalen, die auch zuverlässig von verfügbarer Hardware erfasst werden können. Es ist nicht vorgesehen, dass Gesten aus mehreren nachfolgenden Strokes bestehen und das Verlassen des Kontaktes

zur Oberfläche als Merkmal aufweisen. Dennoch ist diese Taxonomie für die Nutzersicht gut geeignet, da sie die Ausführung der Geste beschreibt. Unter dem Aspekt der Klassifikation sollen Gesten frei gewählt werden können, ohne dass entscheidend ist, ob die Eingabe zweihändig oder mit mehreren Fingern geschieht. Für die hiesigen Zwecke ist somit die Systemsicht und die Taxonomie auf Basis von Trajektorien geeigneter.

3

Architektur der Gestenerkennung

In diesem Kapitel sollen aus der Literatur zusammengetragene Methoden und Konzepte der Gestenerkennung aufgeführt sowie verschiedene Verfahren eingeordnet werden. Zunächst wird versucht, eine allgemeine Einteilung der Aufgaben einer Gestenerkennung anhand eines Modells und der Kategorisierung nach Teilbereichen vorzunehmen. Die Gestenerkennung als Aufgabe der Mensch-Computer Interaktion lässt sich als Problem der Mustererkennung und somit als ein Teilgebiet des maschinellen Lernens auffassen. Etwas spezieller wird sie hier als ein Vorgang verstanden, bei dem aus Sensor- bzw. Rohdaten Merkmale gewonnen und gelernten, in beliebiger Form hinterlegten Klassen zugeordnet werden. In Bezug auf die Taxonomie im vorherigen Kapitel werden die von einem Nutzer ausgeführten Bewegungen mittels einer aktivierenden Technologie registriert. Je nach Sensorik findet eine Vorverarbeitung der Daten statt in denen anschließend interessierende Objekte über der Zeit verfolgt werden. Aus den Informationen der interessierenden Objekte und ihrer Bewegung werden Merkmale gewonnen, welche die Eingabe möglichst kompakt repräsentieren. Dazu werden aus den Daten Segmente - in dieser Arbeit aufgrund der Entsprechung kleinster bedeutungstragender Einheiten - Token genannt - bestimmt, die geeignete Merkmale tragen. Eine Klassifikation nimmt dieses Merkmalsset entgegen und liefert eine Interpretation in Form eines Labels oder eines Klassennamens zurück. Zusätzlich zur Zuordnung einer Klasse können Parameter wie ein statistisches Maß für die Zuverlässigkeit der Interpretation und die Eingabe beschreibende Größen zurückgegeben werden. In Abbildung 3.1 ist die verfeinerte Architektur eines Erkennungssystems, angelehnt an das eben beschriebene Konzept der Gestenklassifikation, dargestellt.

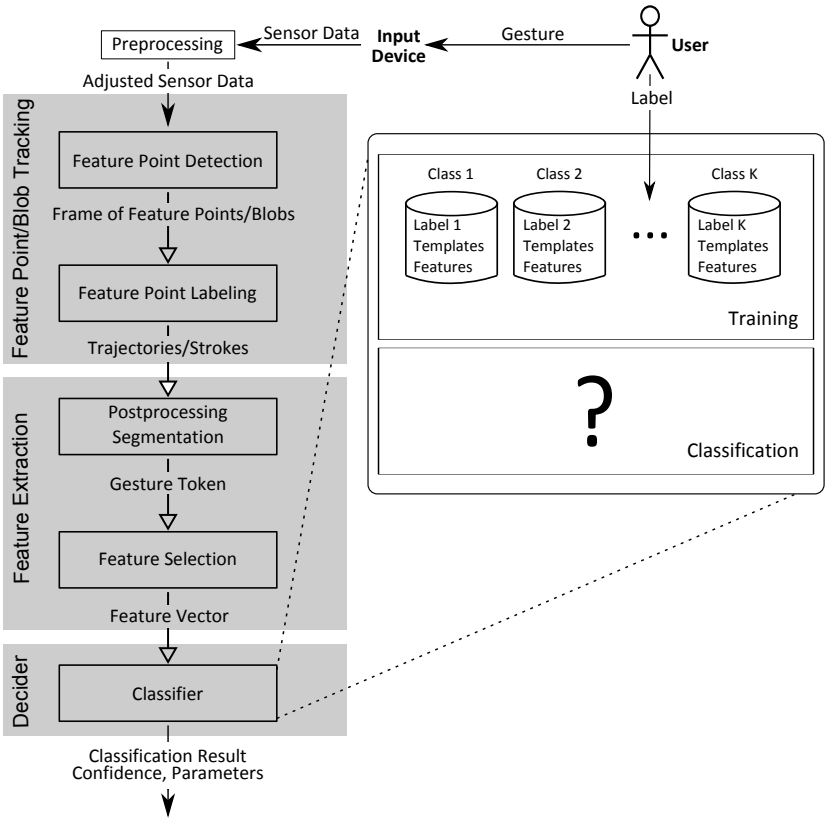


Abbildung 3.1: Architektur eines Erkennungssystems für Gesten nach in [183] veröffentlichter Version. Die Anfang- und Ende-Erkennung einer Geste wird nicht betrachtet und kann, je nach Umsetzung, im Klassifizierer oder in der Merkmalsextraktion gesehen werden.

Die Elemente eines Gestenerkenners werden hinsichtlich der genannten Aspekte in drei Hauptbestandteile gegliedert, da sich verschiedene Varianten in der Literatur beschriebener Verfahren oftmals nur in einem dieser Bereiche unterscheiden. Generell sind die Prozeduren der Bestandteile aber nicht so klar zu trennen. Auch müssen nicht immer alle Phasen durchlaufen werden. Die Detektion von 'Feature Points' oder 'Points/Regions of Interest' etwa soll ermöglichen, gleiche Regionen veränderlicher, lokaler Merkmale in subsequenten Daten wiederzuerkennen. Dies ist selbst wieder ein Vorgang der Mustererkennung und extrahiert zudem oft Merkmale, welche in der eigentlichen Klassifikatorkomponente Verwendung finden. Auch kann der Schritt des Trackings entfallen, wenn nur globale oder statische Merkmale für die Klassifikation benötigt werden beziehungsweise vorhanden sind. Dennoch lassen sich anhand der strikten Aufteilung in der schematischen Darstellung die wesentlichen Vorgänge besser beschreiben und einordnen. Details zur Gewinnung der Sensordaten werden im nächsten Kapitel zur technischen Realisierung von Gesten-basierten Eingabegeräten kurz beschrieben. Es verbleibt, aus diesen Daten nach diversen Vorverarbeitungsschritten Merkmale zu extrahieren (engl. Feature Extraction) und sie zu klassifizieren (nach etwa [169] auch Decider).

In den nachfolgenden Kapiteln wird unter Eingliederung von Literaturbeispielen näher auf diese Komponenten eingegangen. Auch Problemstellungen der Spracherkennung und des Bildverstehens lassen sich dieser Einteilung unterwerfen. Zu jenen Aufgaben, welche der (Online-)Gestenerkennung am nächsten kommen und mit denen eine Schnittmenge besteht, gehören die Erkennung von (mathematischen) Symbolen, diskreter Buchstaben und der (Hand-)Schrift[9] ebenso wie das Erkennen von Skizzen in den entsprechenden Online-Varianten. Im State of the Art werden generell Trajektorie-basierte Verfahren einbezogen, wenn die Klassifikationsmethoden übertragbar sind und eine Unterscheidung erst im Anwendungskontext sinnvoll ist. Ebenso werden Spezialfälle der Erkennung von Gesten im Raum betrachtet, bei denen - aufwendiger zu verfolgende - planare Trajektorien interessierender Punkte für die Klassifikation genutzt werden. In [18] etwa werden Trajektorien 2-dimensionaler Handbewegungen im 3-dimensionalen Raum aus Kamerabildern gewonnen. Diese lassen sich ebenso als Berührungsverläufe eines Kontaktes auf einem berührungssensitiven Gerät auffassen.

3.1 Hardware - Touchscreen Technologien

An dieser Stelle wird ein kurzer und grober Überblick über die gebräuchlichen Technologien zur Erfassung von (Multi-)Touch Interaktionen gegeben. Hauptsächliches Augenmerk sind die Realisierungen von Touchscreens. Nähere Informationen zu den Vor- und Nachteilen der einzelnen Technologien finden sich beispielsweise in [185, 196]. Nach [75, S. 20] lassen sich die Technologien einteilen in basierend auf Membranen, Kapazitätsänderungen, Oberflächen-Schallwellen, Infrarot und Piezoelektrik[10]. Die Variante der Membrane ist dabei in der aktuelleren Literatur eher unter dem Begriff der 'resistiven' Technologie geläufig, da Berührungen anhand von Änderungen eines elektrischen Widerstandes erkannt werden. Hier werden außerdem Verfahren, die auf Reflexionen infraroten Lichts basieren, unter optische Methoden gefasst.

3.1.1 Resistive Technologien

Bei resistiver Sensorik werden zwei leitfähige Schichten (bei Touchscreens meist Indiumzinnoxid, da durchsichtig) durch isolierende Elemente so getrennt, dass sie durch leichten Druck verbunden werden können und ein durch einen Controller gesteuerter, an den Schichten angelegter, wechselseitiger Stromfluss mittels Widerstandsmessungen die Erfassung der horizontalen und vertikalen Position der Druck erzeugenden Berührung ermöglicht [185]. In Abbildung 3.2 ist dieser Prozess schematisch dargestellt.

[9] Tatsächlich existieren auch einige Gestenalphabete [215, 84, 102, 32].

[10] Kristalle an den Ecken eines Touchscreens lassen die Umsetzung von Druck in elektrische Ströme und die aus den vier verschiedenen Messungen abgeleitete Berechnung des Druckpunktes zu. [75, S. 22]

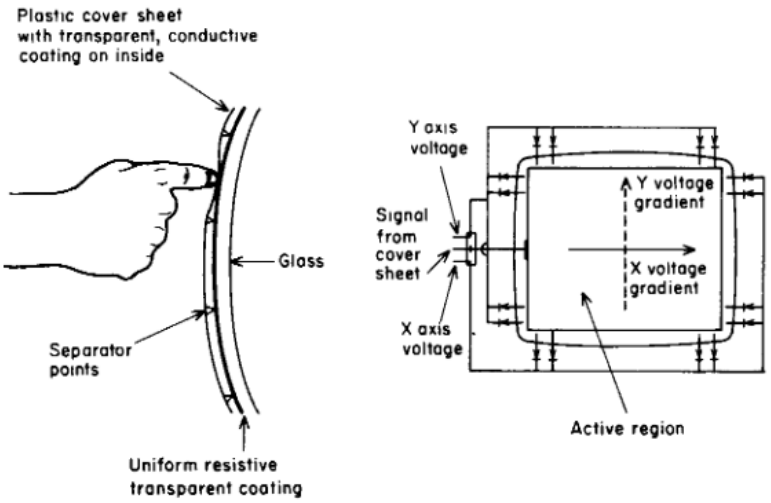


Abbildung 3.2: Illustration resistiver Touchscreens nach [153].

Die Technik gehört zu den älteren Methoden, berührungssensitive Oberflächen umzusetzen und fand beispielsweise schon beim ersten 'Smartphone' [24] und dem ersten PDA von Palm Einsatz (siehe Abbildung 3.3).

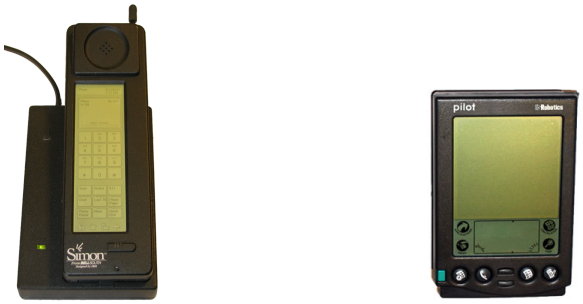


Abbildung 3.3: Das erste 'Smartphone' IBM Simon [212] von 1993 (links) und der Palm Pilot [211] (rechts) lassen berührungsbasierende Interaktionen über einen resistiven Touchscreen zu.

Vorteile der Technik sind die vergleichsweise geringen Kosten und die Unabhängigkeit bezüglich berührender Objekte, nachteilig ist die Anfälligkeit der Sensorik gegenüber Zerkratzen der Oberfläche [48].

3.1.2 Kapazitive Technologien

Kapazitive Techniken nutzen ein elektrisches Feld, welches sich bei Berührung entlädt. Die Berührung muss dabei nicht zwingend erfolgen, da auch eine Annäherung schon zu einer messbaren Entladung führen kann. Oberflächen-kapazitive Systeme verwenden eine leitende Beschichtung und ein an den Ecken der berührungssensitiven Oberfläche erzeugtes elektrisches Feld, dessen Beeinflussung in Form einer Entladung durch etwa einen berührenden Finger ebenda gemessen wird [185]. Das ebenfalls präzise und für ein kleines Bauvolumen geeignete projiziert-kapazitive Verfahren verwendet ein elektrisches Feld zwischen einem Raster aus jeweils zeilenförmig angeordneten Sende- und

Empfangsantennen, an denen eine Spannung angelegt bzw. deren Abfluss gemessen wird [196]. Abbildung 3.4 links stellt diesen Vorgang schematisch dar. Dieses Verfahren findet mittlerweile in einem Großteil handelsüblicher mobiler Geräte Anwendung. In [46] wird durch eine kapazitive Kopplung der berührenden Personen ein eindeutiges elektrisches Signal erzeugt, welches es ermöglicht, die Berührung der entsprechenden Person zuzuordnen (siehe Abbildung 3.4 rechts).

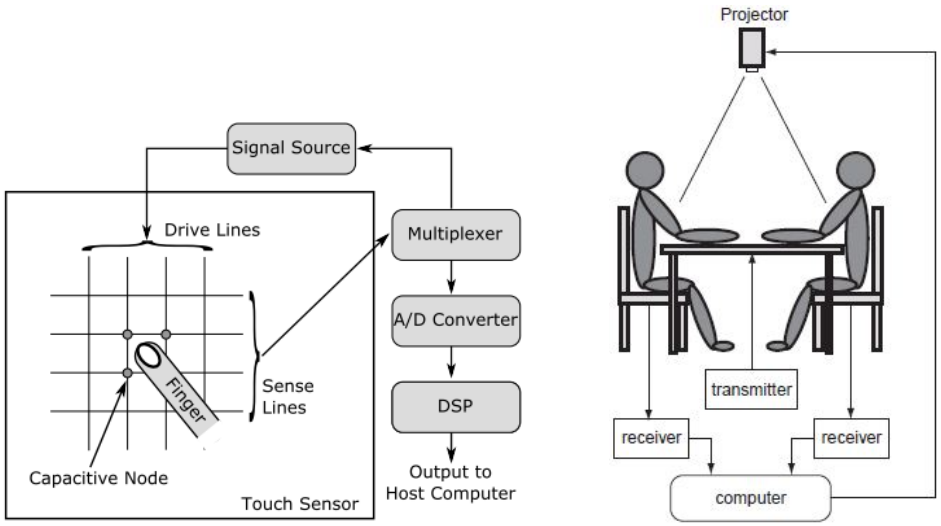


Abbildung 3.4: Links ist die projiziert-kapazitive Sensorik nach [185] schematisch dargestellt, rechts diejenige von 'Diamond Touch' [46], einem Verfahren, um über eindeutige elektrische Signale die kapazitive Touchscreen-Technologie so zu erweitern, dass eine Benutzer-Identifikation möglich ist.

In Abbildung 3.5 wird eine weitere Anwendung kapazitiver Sensorik gezeigt, welche taktile Ausgabe und berührungsbasierte Eingabe in einem Touchscreen für Blinde kombiniert [180]. In der konkreten Veröffentlichung wurde die Technik verwendet um blinden Nutzern die Formen und Ausführungen von Multi-Touch Gesten über taktile Eindrücke zu vermitteln. Die Sensorik diente dabei der Detektion aktueller Handpositionen und Fingerkonfigurationen, um die Ausgabe dynamisch zu steuern und um - aus Gründen der Ergonomie - die Skalierung der Gesten an die Abmessung der Hand anzupassen.

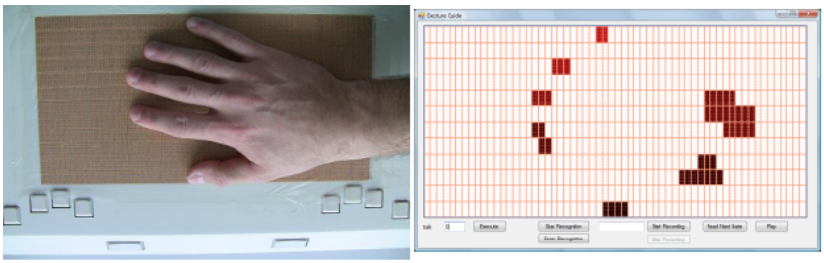


Abbildung 3.5: Ein Touchscreen für Blinde, der auch für die Vermittlung gestischer Interaktionstechniken Verwendung fand [180]. Rechts daneben sind die erfassten Sensordaten einer aufgelegten Hand visualisiert.

Je nach Scanning-Methode der Sensorik können bei projiziert kapazitiven Verfahren sogenannte Geister-Effekte auftreten [12]. Dies beschreibt die Ungewissheit über die genauen Positionen bei mehr als einer Berührung, wenn x- und y- Koordinaten unabhängig voneinander in jeder Abtastung erfasst werden (siehe Abbildung 3.6) Solche Effekte sind nicht nur auf kapazitive Ansätze beschränkt, sondern können auch bei Methoden unter Verwendung von Lichtschranken oder Surface Acoustic Waves eine Rolle spielen. In der vorliegenden Arbeit wird immer von 'echtem' Multi-Touch ausgegangen, bei dem neben der Anzahl der Berührungen auch immer deren genaue Positionen bekannt sind.

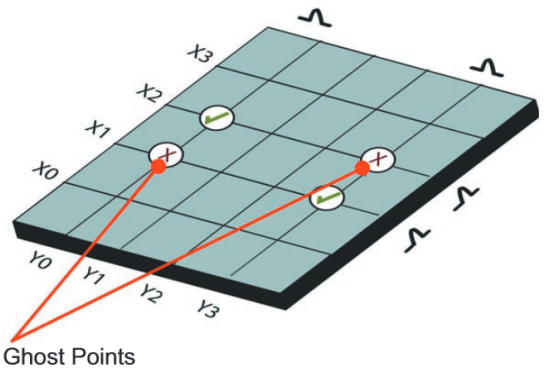


Abbildung 3.6: Illustration des Geister-Effektes in [12]. Zwei Berührungen kann durch die entkoppelte Auswertung der Koordinaten keine der beiden möglichen Interpretationen zu den jeweiligen Positionen der Kontaktpunkte eindeutig zugeordnet werden.

Weitere, vielfältige Realisierungen, um durch kapazitive Verfahren für verschiedene Oberflächen Berührungssensitivität zu erreichen, finden sich neben interessanten möglichen Anwendungen auch in [174].

3.1.3 Optische Technologien

Bei den optischen Verfahren werden Berührungen durch erzeugte Reflexionen oder Unterbrechungen von (meist infrarotem) Licht erfasst. Die Technologien Frustrated Total Internal Reflection (FTIR) und Diffused Illumination (DI) realisieren Touchscreens, indem das anzuzeigende Bild mittels eines Beamers auf eine Projektsionsfläche (z.B. Acryl) gebracht wird. Bei DI wird die Projektionsfläche zusätzlich mit infrarotem Licht ausgeleuchtet, dessen Reflexionen aufgrund von Berührungen (beispielsweise mit dem Finger) mit Infrarotkameras erfasst und dadurch lokalisiert werden. Ein ähnliches Konzept wird bei dem durch Jeff Han im Jahr 2005 vorgestellten FTIR [74] eingesetzt (Abbildung 3.7 links). Im Unterschied zu DI wird auf der Projektionsschicht eine Platte aufgebracht, in die seitlich mittels LED Infrarotlicht unter der Bedingung einer totalen Reflexion abgegeben wird [196]. Die Änderung des Brechungsindex bei Berührung bewirkt die Aufhebung der Bedingung für die Totalreflexion und eine Reflexion am Berührungspunkt durch die Projektionsschicht. Diese wird, wie bei DI, von Infrarotkameras detektiert.

Andere optische Verfahren nutzen spezielle Linsen, die eine Art Infrarot-Teppich auf einer Oberfläche erzeugen oder alternativ je eine Reihe Infrarot-Leuchtdioden für die x- und y-Richtung gepaart mit entsprechend gegenüberliegenden Detektoren, um Berührungen ebenfalls durch Lichtunterbrechung zu erkennen [196]. Das 'Sensor Frame' [132] nutzte das Lichtschranken-Konzept (siehe Abbildung 3.7 rechts) der ersteren Methode bereits in den 1980er Jahren.

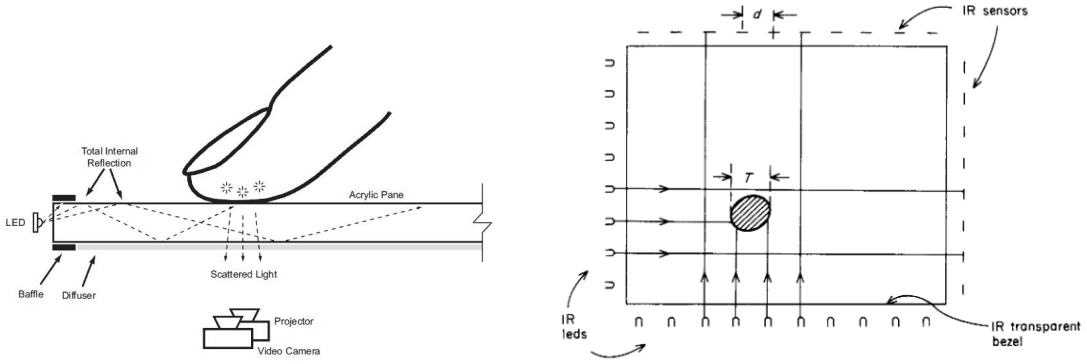


Abbildung 3.7: Illustration der Techniken Frustrated Total Internal Reflection (links, aus [74]) und den ebenfalls auf infrarotem Licht basierenden Lichtschranken (rechts, aus [153]).

Der 'Microsoft PixelSense' (ehemals 'MS Surface') nutzte in seiner ursprünglichen Version DI, der Nachfolger die in Abbildung 3.8 dargestelle Technik. Die unter dem Namen 'ThinSight' [79] veröffentlichte Methode, basiert auf einer Matrix aus Emmissionsquellen und optisch von diesen isolierten Detektoren. Emittiertes infrarotes Licht durchleuchtet dabei ein LCD Panel und wird von vorhandenen Kontakten reflektiert.

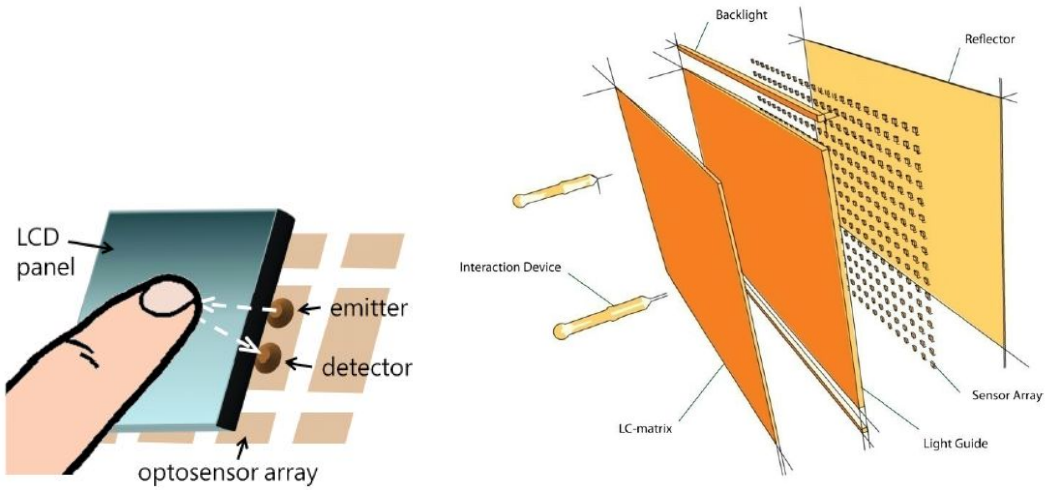


Abbildung 3.8: Bei 'ThinSight' [79] (links) dienen zur Bilddarstellung verwendete Dioden gleichzeitig als Emitter. Die Sensor Matrix in 'Mighty Trace' [105] (rechts) detektiert aktiv von Objekten gesendetes oder mittels FTIR reflektiertes infrarotes Licht. Eine Unterscheidung beider Quellen wird durch zeitliches Multiplexing ermöglicht.

In [105, S. 74] wird unter 'MightyTrace' (siehe Abbildung 3.8) eine Technik beschrieben, bei der ebenfalls die Durchlässigkeit von LCD Panels gegenüber infraroten Lichts ausgenutzt wird. Allerdings dient die Matrix aus Infrarot Sensoren hinter dem Display nur zur Detektion von Objekten (Tangible User Interfaces), welche aktiv infrarotes Licht senden. Mittels zusätzlicher Schicht für FTIR wird dennoch die passive Detektion von Berührungen möglich. Ein weiteres ähnliches Konzept findet sich auch in LED Matrizen, welche wechselseitig zur Ausgabe von Bildern und der Lichtreflexionsmessung (Photodiode) eingesetzt werden [196]. Damit können in Bereichen, in denen eine Anzeige stattfindet, Berührungen durch Reflexionen des Lichts der darstellenden Dioden erkannt werden.

Optische Verfahren, die auf Verwendung von Kameras basieren, sind bezüglich der Detektion multipler Kontakte skalierbar. Zudem bieten sie die Möglichkeit, weitere Parameter wie Abstand oder den Neigungswinkel des berührenden Fingers [100] zu liefern.

3.1.4 Induktive Technologien

Die durch Änderungen eines magnetischen Feldes induzierten Ströme können ebenfalls zur Umsetzung berührungssensitiver Oberflächen genutzt werden. So kann nach [105, S. 20] über das Magnetfeld von in einer Matrix angeordneten Drahtspulen Energie von speziellen Stiften aufgenommen werden, wobei eine grobe Position durch die Entladung der beteiligten Spulen feststellbar ist. Für eine präzise Positionsbestimmung kann die Induktion des nun vom Stift erzeugten Magnetfeldes durch Einschränkung der aktiven Spulen der Sensorik genutzt werden. Weitere Vorteile dieser von Wacom entwickelten Technik werden ebenda benannt und liegen in der möglichen Übermittlung zusätzlicher Daten, wie ID und Druckstärke des Stiftes. In Abbildung 3.9 ist die Funktionsweise des auf Induktion basierenden 'Stylators' [47] aus dem Jahr 1957 dargestellt, dem nach [105, S. 51] erstem Gerät, welches einen Stylus auf einem Tablet detektieren konnte.

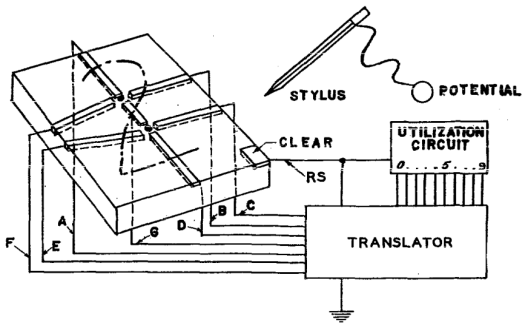


Abbildung 3.9: Schematische Darstellung der Funktionsweise des 'Stylators' [47], bei dem mit Hilfe eines Stiftes ein Stromfluss in bestimmten Zonen induziert wird.

Der 'Stylator' nutzt sieben, um zwei Punkte angeordnete elektrische Leiter, in die beim Überfahren mit dem Stylus Energie induziert wird, wobei die Reihenfolge der Aktivierungen direkt interpretiert und für die Übersetzung in Ziffern genutzt wird [47].

3.1.5 Akustische Technologien

Die resistente - und damit im Outdoor-Bereich, etwa für Bankautomaten eingesetzte - Technik der Surface Acoustic Waves (SAW) verwendet gerichtete Ultraschallwellen. Ähnlich dem Infrarot-Gitter werden mittels piezoelektrischer Signalgeber/-Nehmer jeweils für die x- und y-Achse Schallwellen erzeugt und detektiert. Berührungspositionen lassen sich durch den Grad der durch einen Kontakt beeinflussten Absorption erkennen [185].

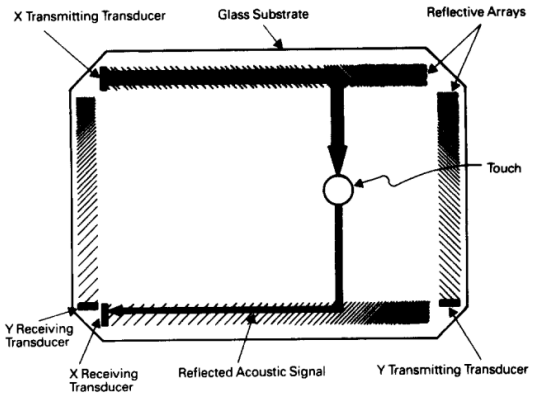


Abbildung 3.10: Schaubild der SAW-Technologie aus [133].

Ebenfalls durch Schallwellen werden in [4] einfache Gesten, die auf Oberflächen eingegeben werden, direkt erkannt, indem durch Mikrofone am Handgelenk aufgenommene Geräusche des Bewegungsapparates interpretiert werden. Mittels Mikrofonen können auch diskrete Berührungen (Taps) auf Glasflächen detektiert, lokalisiert und anhand des aktivierenden Objektes (Knöchel, Fingerspitze) und ihrer Intensität unterschieden werden [148].

3.2 Feature Point Tracking

Aus den statischen Informationen der verschiedenen Frames den Verlauf einer oder mehrerer Berührungen eines sensitiven Eingabesystems zu konstruieren, ist ein essenzieller Verarbeitungsschritt, um Gesten zu erkennen. Die Berührungspunkte in aufeinanderfolgenden Frames müssen identifiziert und die Trajektorien der Berührungspunkte verknüpft werden. Diese 'Feature Points' werden im Falle der Gesteneingabe auf berührungsempfindlichen Flächen auch als Blobs und deren Verfolgung dementsprechend als Blob Tracking bezeichnet. Die Bewegungsmuster der Feature Points stellen dann die Kontaktverläufe auf dem Eingabegerät dar. Jede Trajektorie beginnt mit dem Eintreten des berührenden Objektes in den von der Sensorik erfassbaren Bereich und endet mit dem Verlassen desselben.

Vor dem Vorgang des Feature Point Trackings werden wesentliche Vorverarbeitungsschritte über den Sensor Daten wie etwa Rauschunterdrückung, Signalverstärkung,

Quantisierung und Clusterung vorgenommen, um Daten in einem für die nachfolgende Verarbeitung geeigneten Format zur Verfügung zu stellen. Es wird angenommen, dass eine m x n Matrix S mit den Werten $s_{xy}, 1 \leq x \leq m$ und $1 \leq y \leq n$ das Resultat der Vorverarbeitungsschritte ist. Die Matrix kann beispielsweise ein Graustufenbild oder ein Raster mit Sensorwerten der Berührungsintensitäten auf einer Oberfläche sein. Getaktet durch die Abtastrate des Eingabegerätes erhält man eine Sequenz von Frames S_i, $(1 \leq i \leq s)$ mit derart aufbereiteten Sensorwerten.

Das Feature Point Tracking verfolgt im Zeitverlauf die Points/Regions of Interest beziehungsweise Feature Points innerhalb der Frame-Sequenz. Zwei Arbeitsschritte fallen dafür an: Die Feature Point Detection und deren Zuweisung zu einem Label. Im Bereich Bildverstehen ist diese Aufgabe, auch unter partiellen Informationen, als (Motion) Correspondence-Problem [35] bekannt.

3.2.1 Feature Point Detection

Die eigentliche Aufgabe der Feature Point Detection ist es, Regionen zu identifizieren, welche interessante Merkmale enthalten. Diese Detektion ist zum einen stark von der Art der Datenquelle und zum anderen vom Verfahren der späteren Merkmalsextraktion abhängig. Werden nur statische Merkmale benötigt, so ist nur eine Region, S selbst, relevant. Andernfalls - zum Beispiel bei der Detektion von Fingern, um eine Handgeste in einem Kamerabild zu erkennen - sind mehrere Bereiche festzulegen, aus denen lokale Merkmale extrahiert werden sollen.

Unzählige Beispiele für das Auffinden von Merkmalen finden sich im Bereich der Bildverarbeitung als Vorverarbeitungsschritte für die Objekterkennung, -verfolgung oder Bildanalyse. In [167] wird ein Überblick über Ansätze zur Detektierung von Eckpunkten als hervorstehende Merkmale in Bildern gegeben. Mit dem Problem der Auswahl der Glättungsstufe, um möglichst viele Merkmale, etwa Knotenpunkte, in einem Bild zu erkennen, beschäftigt sich Lindeberg [121]. Ein Verfahren zur Facial Feature Point Detection wird in [203] vorgestellt. Die Aufgabe besteht darin, verschiedene Merkmals-Regionen menschlicher Gesichter (Augen-, Mundwinkel, Position von Kinn, etc.) in Bildern zu identifizieren. Ebenda findet sich noch ein Überblick weiterer Ansätze zur Lösung dieses Problems. Darauf folgend kann beispielsweise eine Gesichtserkennung oder die Verfolgung dieser Merkmale zum Zwecke der Erkennung von Gestik vorgenommen werden. Um Handgesten aus Bewegtbildern zu erkennen, ist zunächst die Detektion der Hand vor dem Hintergrund jeweils eines Bildes nötig. Dies kann beispielsweise über die Analyse der Farbe [229, 70] oder der Frequenzspektren [101] geschehen. Des Weiteren können aus Kontur-Informationen genauere Positionierungen der Handelemente erkannt werden [70]. Ebenfalls mit Hilfe von Kantendetektion in Bildern kann die Fingerspitze identifiziert werden, um Video-basierte Schrifterkennung von mit dem Zeigefinger geschriebenem Text vorzunehmen [224]. Gleichsam wäre es auch möglich, die Information aus der Verfolgung des Fingers zur Gestenerkennung einzusetzen.

Obwohl die Aufgabe der Feature Point Detection in vielen Problemstellungen, auch der Gestenerkennung, einen aufwendigen Vorgang darstellt, werden in dieser Arbeit dementsprechend vorverarbeitete Daten vorausgesetzt. Durch die im Bereich der planaren Gesten eingesetzten Hardware wird das Problem meist vermieden oder zumindest sehr vereinfacht.

3.2.2 Feature Point Labeling

Nimmt man an, dass die Feature Points innerhalb einer gegebenen Frame-Sequenz durch den vorhergehenden Schritt lokalisiert wurden, so besteht die Aufgabe des Labelns darin, gleichartige Feature Points in unterschiedlichen Frames mit einer gleichen Markierung zu versehen, welche sich von denjenigen sämtlicher anderer Feature Points unterscheidet. Wird bei der Feature Point Detection im allgemeinen Fall also ein Frame mit Informationen zu Positions- und Ausdehnungsdaten seiner interessanten Regionen angereichert, kommt in diesem Arbeitsschritt zu jedem dieser Feature Points ein Label hinzu. Das wesentliche Resultat des Vorgangs ist die Trajektorie der Feature Points über die Zeit.

Die Trennung dieser beiden Vorgänge in der Umsetzung ist nicht immer nötig und sinnvoll, wenn, wie in [168], kein a-priori-Wissen über die zu verfolgenden Objekte vorhanden ist. Dann werden Feature Points aus vergangenen Frames in nachfolgenden gesucht und das Labeling ensteht quasi als Beigabe zur Detektion. Im allgemeinen Fall dient ein Ähnlichkeitsmaß für Feature Points dazu, diese in nachfolgenden Frames einander zuordnen zu können.

Im Unterschied zum allgemeineren Correspondence-Problem gelten im Bereich des Trackings von Berührungspunkten (Blobs) üblicherweise einfachere Bedingungen. Zwar bieten Kontaktpunkte wenig mehr Eigenschaften zu ihrer Unterscheidung als den Verlauf ihrer Bewegung und können sich durch unterschiedliche Druckstärken von Frame zu Frame in ihrer Form ändern. Verschmelzungen sowie fehlende Daten sind bei heutiger Hardware und Auflösung aber nicht zu erwarten. Außerdem kann von einer stark begrenzten Zahl zu verfolgender Objekte ausgegangen werden. Dadurch ist es nicht nötig, Blobs anhand ihrer zweidimensionalen Projektion zu unterscheiden.

An dieser Stelle soll nicht weiter auf komplexere Lösungsansätze des allgemeineren Problems, wie in [188] oder [35], eingegangen werden. Im Bereich der Erkennung planarer Gesten reduziert sich das Problem des Trackings bedingt durch die eingesetzte Hardware und Treiber abermals sehr stark. Das Windows 7 Touch Lib Interface etwa liefert bereits getrackte Berührungsdaten. Die minimale Summe aller Abstände zwischen zugeordneten Feature Points nachfolgender Frames ist andernfalls ein hinreichendes Kriterium für ein robustes Tracking und wurde auch in dieser Arbeit, wenn nötig, eingesetzt. Diese Aufgabe lässt sich auf ein klassisches Matching-Problem zurückführen und kann in $\mathcal{O}(N^3)$ bei maximal N Datenpunkten je Frame mit dem Ungarischen Algorithmus [108] gelöst werden.

3.3 Merkmalsextraktion

Obwohl bereits in [116] aufgegriffen, soll die Sichtweise auf die Mustererkennung als ein Problem der 'extraction of the significant features from a background of irrelevant detail' aus [187] an dieser Stelle zur Einleitung dienen. Ebenso merken auch Duda et al. [50] an, dass ein allmächtiger Feature Extractor die Komponente des Klassifikators überflüssig macht und umgekehrt. Die Funktion der Merkmalsextraktion wird dabei als von der Anwendungs-Domäne abhängige Datenreduktion auf bestimmte Merkmale oder Eigenschaften, deren Werte dem Klassifizierer zum Treffen einer Entscheidung übergeben werden, beschrieben. Ist ein Problem der Musterklassifikation gegeben, so ist die Merkmalsextraktion vom Klassifizierer nicht unabhängig. Zwar lassen sich Klassifizierer im Einzelfall bei einem gegebenen Set von Merkmalen durchaus austauschen, generell ist allerdings eine geeignete Wahl der Merkmale sinnvoll.

Die Vorgänge der Merkmalsextraktion und der Entscheidungsfindung werden in dieser Arbeit dennoch als getrennt aufgefasst. Innerhalb der Merkmalsextraktion können allerdings neue Klassifikationsprobleme zu lösen sein. Um das Herangehen zu verdeutlichen, soll zunächst eine Kategorisierung von Merkmalsarten vorgenommen werden, die in - aus Sicht des Autors - ähnlichen Problemstellungen, wie der einer Gestenklassifikation, Verwendung finden. Ziel ist es, Zusammenhänge zwischen vermeintlich verschiedenen Bereichen zu verdeutlichen und eine Fülle möglicher Klassifikationskriterien auch für die Anwendung auf Gesten aufzuzeigen. Diese Arbeit hat nicht das Ziel, alle diese Ansätze vorzustellen, allerdings empfiehlt sich ein umfassender Überblick als Inspirationsquelle auch für weiterführende Studien.

Die Prozedur der Merkmalsextraktion selbst beinhaltet zwei wesentliche Verarbeitungsvorgänge. Zunächst werden die Tracking-Daten weiter segmentiert[11], danach werden aus diesen Segmenten oder Merkmalsobjekten Merkmale extrahiert. Die Merkmale liegen dann als ein reellwertiger n-dimensionaler Vektor $\vec{x}$, nach [141, S. 837] 'Pattern', aus verschiedenen Elementen vor. Auch wenn in der Literatur diese Trennung nicht vorgenommen wird, macht es diese Einteilung einfacher, die Merkmale zu kategorisieren. Außerdem kann die Datenstruktur abgegrenzt werden, die dem Klassifizierer bereitgestellt wird.

Aus Gründen der Anschaulichkeit und der vorgenommenen Ausweitung des Segment-Begriffes werden die beiden Schritte nachfolgend in umgekehrter Reihenfolge behandelt. Zunächst werden die Arten von Merkmalen vorgestellt, die aus verschiedenen Typen von Segmenten gewonnenen werden können. Anschließend wird noch einmal konkret auf die für diese Arbeit relevanten Methoden eingegangen, mit denen eine weitere Zerlegung von Trajektorie-Daten möglich ist.

[11] Der Schritt der Segmentierung kann selbst wieder die Lösung von Klassifikationsaufgaben beanspruchen.

3.3.1 Merkmalsselektion

An dieser Stelle werden die verschiedenste Merkmale systematisiert, welche in der Erkennung planarer Gesten Einsatz finden können. Abgrenzungen und Überschneidungen zu visueller Gestenerkennung sollen ebenfalls verdeutlicht werden. Da die Aufgaben in der Bildanalyse häufig ähnlich der hier betrachteten Aufgabe gelöst werden und sich die Merkmale übertragen lassen, wird dieser Bereich ebenfalls einbezogen. Außerdem wird eine Trennung in Merkmale vorgenommen, welche aus Feature Points selbst und aus deren Trajektorien gewonnen werden. Die Strukturierung der Konzepte orientiert sich weiterhin an der in Abbildung 3.1 gezeigten Architektur eines Gestenklassifizierers. Es wird im Folgenden davon ausgegangen, dass ein in irgendeiner Form geartetes Tracking-Verfahren die Feature Points in subsequenten Frames detektiert und markiert (gelabelt) hat.

Willems et al. [213] liefern einen umfassenden Überblick diverser Merkmale, die für die Gestenerkennung in Frage kommen. Die Autoren tragen ein Set aus 48 globalen und lokalen Merkmalen für die Erkennung von Multi-Stroke Symbolen und Gesten zusammen. Die globalen Merkmale umfassen dabei, jeweils über der gesamten Eingabe berechnet, beispielsweise die Fläche der konvexen Hülle, den Abstand vom Schwerpunkt zum Mittelpunkt des umschließenden Rechtecks, die Anzahl von Krümmungen oder Liniensegmenten, Ausdehnungsverhältnisse, aber auch die Verbundenheit, Geschwindigkeit und Beschleunigung in der Eingabe sowie Auf- und Absetzen des Stiftes. Die lokalen Merkmale umfassen Mittelwerte und Standardabweichungen des gleichen Sets über die einzelnen Trajektorien. Eine weitere Studie, in der eine Fülle an Merkmalen zusammengetragen wird, findet sich in [44]. Die Autoren erstellen ein Set aus 49 Merkmalen für den Zweck der Online-Symbolerkennung. Sie unterteilen den Prozess der Merkmalsfindung in theoretisch basiert, konstruktiv oder selektiv [44, S. 122] und wählen für ihre Zwecke den heuristischen konstruktiven Prozess. Das Merkmalsset umfasst dynamische Merkmale, welche die Art der Eingabe widerspiegeln und visuelle Merkmale, die davon unbeeinträchtigt sind. Ersteres Set enthält Merkmale einer Trajektorie wie die Positionen der Start- und Endpunkte, initiale Winkel und den Winkel zwischen Start und Endpunkt, Geschlossenheit von Kurven, aber auch die Anzahl der abwärts gerichteten Striche in der Eingabe. Visuelle Merkmale sind beispielsweise die Ausrichtung des umschließenden Rechtecks, die Länge von Trajektorien, deren durchschnittliche Richtung oder kumulierte Krümmungen und verschiedene Momente.

In der Bildanalyse werden zur Vorverarbeitung häufig Verfahren zur Kantendetektion und Konturverfolgung eingesetzt. Generell lassen sich die Datenreihen aus Konturen als verfolgte Feature Points, wie sie Online entstehen (Zeitreihen/Trajektorien), auffassen. Aus diesen Daten werden verschiedene nach [116] als lokal bezeichnete Merkmale (Eck-, Knoten-, Endpunkte, Linien, Verzweigungen, Winkel, Krümmungen, Wendepunkte) ermittelt. Gemessene Winkel, Längen, Häufigkeiten werden von Rubine [169, S. 35-37] als geometrische Merkmale bezeichnet. In [206, S. 7-9] hingegen werden Häufig-

keiten (beispielsweise des Auftretens von Schnitt-, Scheitelpunkten) zu geometrischen Merkmalen zusammengefasst, während Winkel eine eigene Kategorie (Direktionale) stellen. Weiterhin ordnen Watt und Xie [206] diskretisierte Verhältnisse von Breite und Höhe umschließender Rechtecke und sogenannte Zonen zu einer Kategorie der globalen Merkmale. Zonen (auch Regionen) sind ein häufig eingesetztes Mittel (z.B. Buchstabenerkennung in [82, 118, 61]), um Verläufe zu quantisieren. Eine Kontur oder Trajektorie wird dabei durch die von ihr durchschrittenen Bereiche einer festgelegten räumlichen Einteilung repräsentiert. Rubine [169] führt für die Zonen ebenfalls eine eigene Kategorie ein. Diskretisierungen/Quantisierungen und relative Werte werden hier hingegen nicht gesondert eingeordnet. Werden Positionen durch Symbole repräsentiert, so bleiben es Positionsangaben. Watt und Xie etwa [206] nutzen für ihre Symbolerkennung die Bereiche, in welche die Anfangs- und Endpunkt eines Striches fallen, als Merkmale. Relative Längen verbleiben in der Merkmals-Kategorie der Distanzen.

Hier soll die Einteilung dieser Merkmale in geometrische (wie Positionen, Distanzen und Direktionen; von Repräsentanten aber auch kumuliert), sowie numerische (Zählungen) erfolgen. Zu den Zählungen werden auch relative Häufigkeiten gerechnet, wie die in [206] vorgeschlagene Ermittlung der Verteilung einer Trajektorie auf bestimmte Zonen als Form eines Dichtemaßes. Die Merkmalsobjekte/Segmente (etwa Linien oder Wendepunkte) werden allerdings hier dem Segmentierungsschritt zugeordnet. Eine Systematisierung der Merkmale und der Merkmalsextraktion findet sich in Abbildung 3.11.

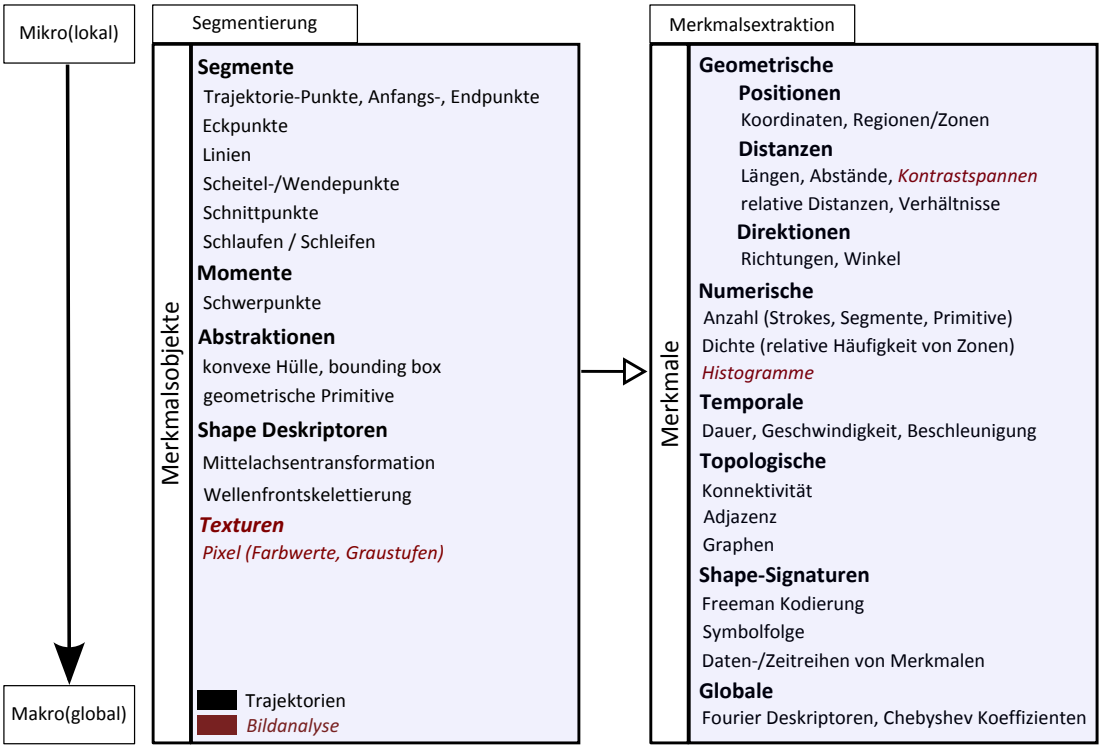


Abbildung 3.11: Taxonomie der Merkmale anhand der Verarbeitungsschritte der Merkmalsextraktion. In den einzelnen Kategorien der Merkmalsobjekte sind prominente Vertreter der Merkmale angegeben.

Die in [116] verwendete Kategorie der Momente (z.B. bieten Mittelpunkte geometrischer Primitive verschiedene Invarianzen) wird hier als mögliche Ausgabe der Segmentierung übernommen. Merkmale selbst können dann deren Positionen direkt, Abstände zur Kontur o.Ä. sein. Diese werden beispielsweise in [18] zum Zweck der Gestenerkennung aus 2D Trajektorien der Hände neben weiteren geometrischen, numerischen und temporalen Merkmalen genutzt. Numerische Merkmale sind dort die Anzahl der Punkte, bei denen die Änderung des Winkelverlaufes einen Schwellwert von 45° überschreitet. Bei den temporalen Merkmalen werden Geschwindigkeiten und Beschleunigungen erfasst.

Die Gruppe der topologischen Merkmale enthält Konnektivitäten, Adjazenzen und Graph-Strukturen. In [116] werden zusätzlich dazu auch Zählungen und Positionen von markanten Punkten (End-, Eckpunkte) oder auch Richtungen markanter Formen gezählt. Der Argumentation, dass dies Elemente darstellten, die von Menschen zur Beschreibung verwendet werden würden, wird hier nicht gefolgt. Schleifen, Haken, Konturen, konvexe Hüllen und andere Formen werden als Merkmalsobjekte aus dem Segmentierungsschritt betrachtet und deren Richtungen (auch wenn diese als z.B. links/rechts vorliegen), Positionen und Häufigkeiten werden den entsprechenden Merkmalstypen zugeordnet.

Weitere Merkmale werden in [206] als 'Ink-Related' bezeichnet und entstammen der Metapher der elektronischen Tinte. Sie enthalten dort die Zahl der Strokes oder deren Dichte, welche hier unter Zählungen fallen. In [213] werden Druckstärke und Anzahl/Verhältnis des Auf- und Absetzens betrachtet. Letzteres sind ebenfalls numerische Merkmale. Die Druckstärke, obwohl hier nicht aufgegriffen, kann allerdings - ebenso wie Neigungswinkel des Eingabegerätes (Stift, Finger) - dieser Gruppe ('Ink-Related') zugeordnet werden.

Eine Merkmalsgruppe, die in [116] und [169] definiert wird, soll auch hier übernommen werden. Beide Arbeiten sehen unter verschiedenen Begriffen eine Merkmalskategorie vor, die Datenreihen von Punkten der Eingabe [169] oder auch Positionen von Primitiven [116] (durch Fenster-basierte Scan-Methoden geordnet) darstellt. Grund für diese Einteilung ist, dass bestimmte Methoden der Klassifikation auf einem direkten Vergleich oder einer Distanzberechnung zwischen derart vorbereiteten Daten basieren. In [222, S. 1] wird die Merkmalskategorie der 'Templates' nur für die einfachste Form solcher Zeitreihen vorgesehen: 'Templates are the simplest features to compute; they are simply the input data in its raw form'. Gesten werden ebenda vereinfacht als in dieser Form oder durch eine Beschreibung spezifiziert gesehen.

An dieser Stelle wird die Annahme getroffen, dass derartige 'Online'-Daten aus bereits beschriebenen Merkmalen bestehen, die unter temporalen Informationen (Zeitreihen aus Sequenz der Samples, Verlauf der Trajektorie oder künstlich aus Konturkodierungen bzw. Fenster-basierten Verfahren) geordnet wurden.

Diese spezielle Gruppe soll unter dem aus dem Gebiet der konturbasierten Shape-Klassifikation entlehnten Begriff der Shape-Signaturen (siehe beispielsweise [228]) zusammengefasst werden. Shape-Signaturen[12] repräsentieren dort 2-dimensionale Bilder in Form einer 1-dimensionalen Abbildung [52, S. 162]. Hier wird im einfachsten Fall eine Punktefolge herangezogen, es können aber auch Abstände dieser Punkte zu ihrem Schwerpunkt, Winkelfolgen (absolut/relativ) und als Spezialfall die Freeman-Kodierung [60] und andere Sequenzen gleichartiger Daten, welche die Form der Trajektorie/Kontur repräsentieren, gewählt werden. Als ein weiterer Ansatz aus der Bildanalyse soll die Transformation solcher Datenreihen in den Frequenzbereich durch Bestimmung von Fourier-Deskriptoren [13] (in [169] ebenfalls als globale Merkmale in Betracht gezogen) noch als mögliches Feature erwähnt werden. Den globalen Merkmalen werden dementsprechend auch Chebyshev-Koeffizienten zugeordnet. Diese werden beispielsweise in [96] verwendet, um Trajektorien von zu erkennenden Symbolen als Funktionen zu approximieren. Weitergehende Betrachtungen wären für die hier gesetzten Ziele nicht zweckmäßig und würden den Rahmen sprengen. Insbesondere die in Abbildung 3.11 aufgeführten Bild-basierten Merkmale wie Texturen (Farbwerte/Graustufen), Histogramme oder Kontrastspannen sind aufgrund der hier betrachteten Trajektorie-basierten Ausgangsdaten nicht weiter von Relevanz. Ihr Einbezug dient zugleich der Vervollständigung der Systematik und der Abgrenzung.

3.3.2 Segmentierung

Ohne weiteren Segmentierungsschritt werden die Merkmale direkt aus den Feature Points und den zugehörigen Trajektorie-Daten abgeleitet. Verfahren, welche die Shape-Signaturen direkt aus den Trajektorien extrahieren, finden sich in [217, 40, 119], jeweils zur Erkennung von Single-Touch Gesten. In [193] werden zur Erkennung mathematischer Symbole markante Punkte der nach Freeman [60] kodierten Trajektorie vorselektiert, indem wie in [207] (für Gesten) aufeinanderfolgende gleiche Merkmale der Signatur vernachlässigt werden. Bhuyan et al. [18] abstrahieren Trajektorien durch Reduktion der Datenpunkte, bis der Interpolationsfehler einen Schwellwert überschreitet. Eine weitere Möglichkeit, markante Datenpunkte zu erhalten, ist der Ramer-Douglas-Peucker-Algorithmus, wie er beispielsweise in [160] zur Polygon-Approximation von Kurven vorgestellt wurde.

Demgegenüber stehen komplexere Zerlegungen[14], wie sie häufig im Bereich der Er-

[12] Im Gebiet der konturbasierten Klassifikation von Formen werden Shape-Signaturen genutzt, um sogenannte Shape-Deskriptoren (etwa durch Fourier-Transformation) zu erhalten.

[13] Fourier-Deskriptoren werden durch diskrete Fourier-Transformation (Frequenzbereich) einer Datenreihe der Repräsentation der Kontur einer Form gewonnen. Die Koeffizienten dieser Fourier-Transformation (Deskriptoren) können zum Vergleich verschiedener Formen und damit auch zur Klassifikation genutzt werden. Die Konturen bezeichnen dabei im Allgemeinen zwar geschlossene Kurven, es sind aber auch Vergleiche unter fehlenden Daten möglich [120]. Oft findet für die Konturrepräsentation eine Sequenz der komplexen Darstellung abgetasteter, geordneter Punkte der Kontur Anwendung.

[14] Da bekannte Start- und Endpunkte bei der Eingabe vorausgesetzt werden, wird auf das in der Erkennung von Skizzen zusätzliche Problem der Gruppierung in dieser Arbeit nicht eingegangen.

kennung von Skizzen vorgenommen werden. In [190] wird eine Trajektorie unter anderem anhand von Geschwindigkeitsänderungen in Linien und Bezier-Kurven zerlegt. Calhoun et al. [27] sowie Chen und Xie [34] nutzen die relative Geschwindigkeit und Krümmungen als Zerlegungskriterien. Krümmungen werden auch in [223] als Kriterium für eine Segmentierung genutzt. Hse et al. [81] hingegen segmentieren Trajektorien mittels dynamischer Programmierung in Linien und Bögen. Konvexe Hüllen werden von Apte et al. [9] und Fonseca et al. [58] vorberechnet, um geometrische Merkmale zu selektieren.

Eine an Impulse an das zentrale Nervensystem angelehnte modellbasierte Segmentierung elektronischer Handschrift wird in [154] mittels Wendepunkten der Geschwindigkeit vorgenommen. Li et al. [117] greifen den Ansatz auf und nutzen Krümmungen und Wendepunkte für die Zerlegung und Rekonstruktion des Textes mit dem Ziel der Datenreduktion. Eine modellbasierte Zerlegung von Single-Touch Gesten findet sich in [28]. Die Zerlegung an Krümmungen, Linien und Ecken wird für die Prognose von Eingabezeiten genutzt, kann allerdings für die Anwendung der Gestenerkennung übertragen werden.

3.4 Klassifikation Trajektorie-basierter Eingaben

Fukunaga [63] fasst die Mustererkennung - und damit auch den speziellen Fall der Gestenerkennung - als einen Prozess der Entscheidungsfindung (engl. decision-making) auf. Ziel der Entscheidung ist die Zuweisung eines durch einen mehrdimensionalen Merkmalsvektor repräsentierten Objektes zu einer Klasse oder Kategorie mittels Festlegung einer Diskriminanzfunktion. Durch die Diskriminanzfunktion werden Entscheidungsgrenzen in einem mehrdimensionalen Raum festgelegt. Ein Klassifizierer (auch Kategorisierer) wertet diese Funktionen aus, um anhand ihrer Werte die Entscheidung über die Zuweisung des Objektes zu einer Klasse zu treffen. Auch wenn diese Trennung in der Literatur häufig nicht so scharf ist, ist in dieser Arbeit für einen solchen Vorgang der engere Begriff der Musterklassifikation (engl. pattern classification) - entsprechend der in [172, S. 437] allgemein formulierten 'Überprüfung, ob ein Objekt zu einer Kategorie gehört' - gemeint. Übertragen auf das vorliegende Problem, stellen Kategorien Klassen von Gesten dar während das Objekt eine eingegebene Geste ist. Das Ergebnis der Überprüfung ist das Label oder der Index der Klasse, zu der die Eingabe gehört. Objekte sind dabei durch einen n-dimensionalen Merkmals-Vektor (siehe Kapitel 3.3) charakterisiert. Über den Trainingsdaten ist ein Modell zu bilden, welches die Label von Testinstanzen anhand ihrer Merkmale vorhersagt. Fehlt dieses Label in den Trainingsbeispielen, so können Methoden der sogenannten unüberwachten Klassifikation (Clustering) Kategorien innerhalb der empirischen Daten durch Klasseneinteilungen des Raums suchen [63, S.3-7]. In dieser Arbeit findet entsprechend eine überwachte Klassifikation als aus empirischen Daten abgeleitetes Mapping von Eingabe (Merkmalsvektor) auf eine diskrete Ausgabe (Klassen) Verwendung.

Zunächst sei ein Überblick existierender Verfahren zur Klassifikation planarer Gesten gegeben. Abermals werden auch verwandte Gebiete betrachtet und Methoden einbezogen, die sich auf den Verwendungszweck übertragen lassen. Zudem ist eine scharfe Trennung der Gebiete nicht immer sinnvoll. Beispielsweise werden sowohl in der Gestenerkennung [222, 18] als auch der Symbolerkennung [193, 206] häufig Single-Stroke Ziffern klassifiziert. Eine Unterscheidung zwischen Gesten und Skizzen wird ebenso oft (z.B. in [77]) nicht getroffen. Der Fokus liegt allerdings auf den Online-Methoden. Bild-basierte Verfahren der Symbolerkennung wie in [91, 147] werden nicht betrachtet, auch wenn sie temporale Merkmale einbeziehen. In ähnlicher Form wurde der State of the Art in den folgenden Abschnitten unter [184] und in Bezug auf Skizzen in [182] veröffentlicht.

Es existieren verschiedene Sichten auf das Gebiet der Musterklassifikation, die teilweise in den obigen Betrachtungen bereits eingeflossen sind. Yang und Xu [222] teilen Klassifikationsmethoden für die Betrachtung von Vor- und Nachteilen in die Kategorien 'template-matching', 'dictionary lookup', 'statistical matching', 'linguistic matching', 'neural network' und 'ad hoc methods'. Im Nachfolgenden werden die Klassifikationsmethoden in der Literatur vertretener Verfahren grob in basierend auf Modellen, Beschreibungssprachen, Regeln, Statistik, neuronalen Netzen und Nächste-Nachbar-Suche eingeteilt. Die Methoden des Template-Matchings oder der Wörterbuchsuche werden dabei zu den Nächste-Nachbar-Verfahren geordnet. Linguistische Ansätze finden sich unter den Beschreibungssprachen, während Ad-hoc Methoden als spezielle Form der Regel-basierten Klassifikation gesehen werden. Eine weitere, wenn auch selten umgesetzte, Kategorie wird mit den Modellen eingeführt. Ohne Anspruch auf erschöpfende Behandlung zeigte sich diese Einteilung für Klassifikationen Trajektorie-basierter Eingaben als geeignet, die in der Literatur vertretenen Verfahren einzuordnen.

3.4.1 Modelle und Beschreibungssprachen

In [154] wird ein Erzeugungsmodell für Handschrift genutzt, um diese zu segmentieren. Die Autoren geben an, dass ebenso die Überprüfung von Unterschriften oder die Erkennung von Handschrift möglich wäre. Genauso vorstellbar ist die Anwendung des 'Human Performance Models' für Gesten aus Stifteingaben aus [28] für den Zweck der Gestenerkennung. Ein Kalman-Filter und ein physikalisches Modell zur Bewegung einer beschwerten Hand wird in [178] zur Erkennung von Gesten eingesetzt. Zwar werden prinzipiell räumliche Gesten unterstützt, aber das Verfahren beschränkt sich vorerst auf planare Freihandgesten, die über Schwellwerte der Ausführungsgeschwindigkeit segmentiert werden.

Beschreibungssprachen (engl. description languages) oder Grammatiken finden sich häufig im Bereich der Erkennung von Skizzen wieder. Sie erlauben das Zerlegen von Eingaben in primitive Objekte und das Formalisieren ihrer Beziehungen in einer für den Menschen verständlichen Form. So werden Grammatiken zur Beschreibung domänenspezifischer, komplexerer Skizzen aus Primitiven etwa in [72, 87, 38, 21] eingesetzt.

Costagliola et al. [38] generieren dabei Parser für die Klassifikation direkt aus den vom Nutzer spezifizierten Grammatiken. Parser-Generatoren werden ebenfalls in [21] eingesetzt, um domänenspezifische Erkenner für Skizzen zu erzeugen. Auf lexikalischer Ebene wird ein statistischer Klassifizierer eingesetzt. Über Grammatiken werden dann Kompositionen aus diesen Token (z.B. Linien, Rechtecke, Pfeile) unter Berücksichtigung von Kontextinformationen und Auflösung von Mehrdeutigkeiten als skizzierte Diagrammbestandteile interpretiert.

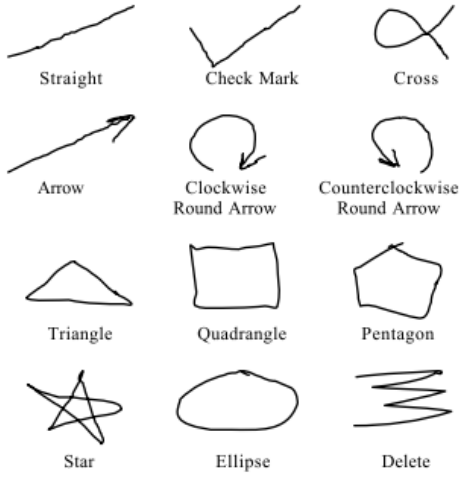
Eine semiotische Beschreibungssprache für Gesten wurde in [89] entwickelt. Eine weitere 'Gesture Definition Language' findet sich in [97]. Sie ähnelt der früheren prozeduralen 'Gesture Description Language' aus [124], welche Gesten in Form von Tests auf Primitive und deren Beziehungen beschreibt. Ein mathematisches Kalkül wird in [69] zur Erkennung von Gesten verwendet. Ebenfalls für die Beschreibung von Gesten werden in [99, 220] reguläre Ausdrücke spezifiziert. Ersterer Ansatz kodiert die Shape-Signatur von Gesten in Sequenzen von 12 möglichen Richtungsänderungen ähnlich der Freeman-Codierung. Der letztere Ansatz quantisiert den Ortsverlauf der Trajektorie aus traversierten Zellen eines Rasters.

3.4.2 Regel-basierte Klassifikation

Zu den Regel-basierten Ansätzen werden hier intuitive Tests und Ad-hoc Methoden ebenso gefasst, wie formalere Entscheidungsbäume, semantische Netze oder logische Systeme. Im Gegensatz zu den Beschreibungssprachen wird bei den hierarchischen Tests auf Eigenschaften oder Ablehnungskriterien nicht auf eine vom Menschen lesbare Form Wert gelegt. Entscheidungsbäume klassifizieren ein Objekt anhand einer Sequenz von Abfragen, die jeweils eine Verzweigung im Baum repräsentieren. Diese Methode kann beliebige Entscheidungsgrenzen in beliebiger Näherung beschreiben [50]. Außerdem eignen sich Regel-basierte Ansätze auch für die Klassifizierung nach nominalen Merkmalen.

In Abbildung 3.12 ist ein exemplarisches Gestenset samt unterscheidender Regeln aus [146] wiedergegeben. Ou et al. [146] nutzen für die Klassifizierung von Gesten, welche mit dem Stift über einem Video-Stream eingegeben werden, ein gleitendes Fenster, um Stellen, deren Anstiegsänderung eine Schwelle überschreitet, zu identifizieren. Per Kriterien wie der Anzahl dieser Richtungsänderungen und Kurven oder der Geschlossenheit einer Geste werden über je eine einfache Regel 12 Gesten klassifiziert. Die Art der Regeln lässt Invarianzen bezüglich Geschwindigkeit, Translation, Rotation und Skalierung zu, aber das Regelset ist stark vom Gestenset abhängig. Zusätzliche Gesten könnten neue Kriterien für jede bereits aufgenommene Gestenklasse bedeuten.

Einige Verfahren in der Erkennung von Skizzen verfolgen ebenso Regel-basierte Ansätze. In [226] werden Ad-hoc Tests auf räumliche und zeitliche Eigenschaften für die Erkennung von Gesten und Formen vorgenommen. Alvarado und Davis [2] erkennen einfache Formen wie Linien oder Kreise durch Regeln und wenden diese hierarchisch an, um komplexere mechanische Skizzen zu erkennen. Ähnlich nutzen Sezgin et al. [190] Tests



Open Gestures	
Straight Line	Has no vertexes or curves
Check Mark	Has only one vertex
Cross	Has no more than two vertexes, and a segment of curve
Arrow	Begins with a straight line and has three vertexes in the latter part
Round Arrow	Begins with a curve and has three vertexes in the latter part
Delete	Has more than three vertexes
Closed Gestures	
Triangle	Has three vertexes
Quadrangle	Has four vertexes
Pentagon	Has five vertexes and the average slope changes at vertexes is lower than a threshold
Star	Has five vertexes and the average slope changes at vertexes is higher than a threshold
Ellipse	Has fewer than two vertexes

Abbildung 3.12: Planares Gestenset aus Stift-basierten Eingaben und zugehörige Regeln zur Klassifikation aus [146].

auf Linien und Kurven und einfache, daraus kombinierte Figuren. In [189] werden Tests auf kreuzende Linien und weitere Formen ergänzt. Hammond und Davis [71] greifen den Ansatz von [190] auf, um zusätzlich verschiedene Pfeile zu erkennen. Hierarchische Tests auf geometrische Eigenschaften (Fläche/Länge-Verhältnis) konvexer Hüllen von Skizzen werden in Apte et al. [9] genutzt. Der Ansatz wird von Fonseca et al. [58] aufgegriffen und durch händische Fuzzy-Logik Regeln für Perzentile klassenspezifischer Sets geometrischer Merkmale ergänzt. Fuzzy-Logik wird auch in [34] zur Erkennung geometrischer Primitive verwendet. Heuristische Ad-hoc Regeln werden in [83, 226] für die Erkennung von einfachen Gesten, Schriftsymbolen und Formen genutzt. Zeleznik et al. [226] erweitern dabei die Arbeit in [83] und nutzen nach dem Segmentierungsansatz aus [27] die heuristischen Ad-hoc Regeln über flächige und zeitliche Eigenschaften, um Trajektorien in Gesten, Schrift und Primitive zu unterscheiden. Weitere heuristische, Regel-basierte Ansätze der Skizzenerkennung werden in [223, 149] verfolgt und in [27] werden Symbole durch semantische Netze erkannt.

In [110] werden Entscheidungsbäume trainiert, um Multi-Touch Eingaben zu erkennen. Allerdings werden nur direkte Manipulationen in Form verschiedener Taps, Pinch-, Rotations- und Wisch-Gesten unterstützt. Die Form der Trajektorien wird dabei nicht berücksichtigt und das Verfahren erreicht beim Vergleich von lediglich zwei Klassen eine Genauigkeit von 90%, bei vier Klassen nur noch 80%.

Im Bereich der Schrifterkennung wird ein interessanter Ansatz in [104] verfolgt. Eine hierarchische Überprüfung auf das Vorhandensein bestimmter Merkmale teilt über eine binäre Entscheidungsbaumstruktur den Suchraum sukzessive ein. Kann eine Entscheidung nicht binär getroffen werden, werden jedoch beide Pfade weiter verfolgt.

3.4.3 Statistische Verfahren

Klassifikation als Separierung eines hochdimensionalen Raums kann durch verschiedene Herangehensweisen umgesetzt werden. Anderson [5, S. 207] beschreibt Klassifikation aus statistischer Sicht als ein Problem von 'statistical decision functions'. Dabei werden Hypothesen - also Annahmen, dass eine Beobachtung eine bestimmte Verteilung (welche eine Klasse repräsentiert) hat - getestet. Die Klassifikation besteht aus der Wahl der besten dieser Hypothesen, während die anderen abgelehnt werden. Dementsprechend gestaltet sich das Problem aus dieser Sicht als Schätzung der bedingten Wahrscheinlichkeitsdichtefunktionen[15] (engl. probability density function, kurz: PDF) der Klassen und der Ableitung einer Diskriminanzfunktion unter einem Optimierungskriterium.

Nach [37, S. 274] wurde der erste Algorithmus zur Mustererkennung bereits 1936 von Fisher in [57] vorgestellt. Der Algorithmus nutzt eine statistische Diskriminanzfunktion und beruht auf der Annahme, dass die n-dimensionalen Merkmalsvektoren einer von zwei Normalverteilungen $N(m_1, \Sigma_1)$, $N(m_2, \Sigma_2)$ mit den Mittelwertvektoren m_1, m_2 und den Kovarianzmatrizen Σ_1 und Σ_2 entstammen. Der Wert der Diskriminanzfunktion $M = \log(f_2(X)/f_1(X))$ aus den jeweiligen PDF zweier Klassen entscheidet, welcher Klasse ein Objekt mit dem Merkmalsvektor X zugewiesen wird [204, S. 480].

Nach [63, S. 3] ist der theoretisch beste Klassifizierer, im Falle gegebener Wahrscheinlichkeitsdichtefunktionen der Zufallsvektoren, der Bayes'sche Klassifizierer, da er die Wahrscheinlichkeit einer Fehlklassifikation minimiert. Er basiert ebenfalls auf statistischen Diskriminanzfunktionen und findet häufig unter vereinfachenden Annahmen Anwendung. Weitere Klassifikationsmethoden in diesem Abschnitt sind Hidden Markov Modelle, (dynamische) Bayes'sche Netze oder Kalman-Filter.

In [170] werden planare Gesten durch lineare Gauß'sche Diskriminanzanalyse und ein Set von 13 mutmaßlich Gauß-verteilten Merkmalen erkannt. Das Verfahren setzt die Bayes'sche Klassifikation unter Annahme gleicher Verteilungen und Kovarianzmatrizen der verschiedenen Klassen um. Das verwendete Merkmalsset unterstützt keine Invarianzen bezüglich Rotation, Skalierung und Geschwindigkeit. Eine Erweiterung für die Erkennung von Multi-Touch Gesten durch hierarchische Single-Touch Erkennung wird in [169] vorgeschlagen. Allerdings werden die verschiedenen Trajektorien einer Multi-Touch Geste anhand ihrer Positionen und Startzeiten geordnet, was zu Mehrdeutigkeiten führen kann. Eine Alternative, basierend auf globalen Merkmalen, vermeidet zwar das Sortierungsproblem, hat aber schlechtere Klassifikationsraten zur Folge. Hussain et al. [82] verwenden Merkmale der Auftrittshäufigkeit von Punkten in Zonen (Dichte), um handgeschriebener Großbuchstaben mit einem Bayes'schen Klassifizierer zu erkennen.

Hidden Markov Modelle (HMM) gehören wohl zu den am häufigsten eingesetzten Klassifikationsmethoden für Zeitreihen und sind dementsprechend sehr gut erforscht. Sie basieren ebenso auf dem Lernen von Verteilungen und eignen sich nach [228], wie auch Nächste-Nachbar-Methoden, sehr gut für die Anwendung auf Shape-Signaturen.

[15] Es werden hier nur kontinuierliche Verteilungen betrachtet.

Eine Einführung in die Methodiken findet sich in [159]. Yang und Xu [222] nutzen in der Spracherkennung verbreitete Bakis-Modelle [11] (auch Links-Rechts Modell) sowie Zeitreihen von Fourier-Koeffizienten, um neun Maus-Gesten (Ziffern) zu erkennen. Die Autoren geben an, dass sich durch Verknüpfung der Modelle auch eine kontinuierliche Gestenerkennung, also eine Gestendetektion im Eingabestrom, umsetzen ließe. Ebensolche Gestendetektion in Trajektorien kontinuierlicher Handbewegungen wird in [45] über ein Modifizieren des Viterbi-Algorithmus[16] für die Erkennung von acht Ziffern realisiert. Die Trajektorien werden dafür in Sequenzen 16 möglicher Richtungen codiert, die wiederum Bakis-Modelle trainieren. Quantisierte Richtungsänderungen (16 mögliche Werte) in Verbindung mit diskreten HMM werden auch in [207] eingesetzt, um nicht näher spezifizierte Single-Touch Gesten zu erkennen. Die Zustandszahl der HMM wird durch die Segmentierung der Gesten abgeschätzt, allerdings nachfolgend manuell justiert. Die Ablehnung von Eingaben wird über ein ergodisches (vollständig vernetztes) HMM umgesetzt, welches die Modelle einer jeden Gestenklasse vereint und dessen Ausgabe als Schwellwert für die Entscheidung einer Ablehnung dient.

In [118] werden Zonen zur Merkmalsgewinnung aus handgeschriebenen Ziffern genutzt, welche dann mit einem HMM klassifiziert werden. Links-Rechts Modelle werden des Weiteren in [198] für die Klassifikation von Skizzen einfacher geometrischer Symbole und in [114] für koreanische Buchstaben, die aus Linien komponiert sind, verwendet. Multi-Stroke Skizzen unter jeder vom Nutzer präferierten Eingabefolge und damit bekannter Ordnung der Trajektorien werden in [189] mit Hilfe von Hidden Markov Modellen trainiert. Diskrete Ausgabesymbole sind dabei die in der Vorsegmentierung nach [190] ermittelten Primitive. Es können auch komplexere Szenen erkannt werden, indem mögliche Interaktionsverläufe als Graph modelliert werden, dessen Kantengewichtungen den Wahrscheinlichkeiten für partielle Symbolemissionen in einem möglichen zuordenbaren HMM entsprechen. Anschließend wird eine Kürzeste-Wege-Suche (dynamische Programmierung) im Graphen zur Ermittlung der bestmöglichen (im Sinne von maximaler Auftrittswahrscheinlichkeit) Segmentierung der Szene durchgeführt.

Ein weiterer Ansatz zur Erkennung von Gesten mittels HMM ist in [42] aufgeführt. Er erlaubt die Erkennung von Multi-Touch Gesten und laut den Autoren auch recht feine Unterscheidungen. Allerdings werden keine Multi-Stroke Gesten erkannt. Die Struktur der zugrunde liegenden HMM und konkrete Details der Klassifikation werden zudem nicht näher beschrieben. In [29] kommen ebenfalls HMM zum Einsatz, wobei auch die Anzahl der Zustände der HMM aus den Trainingsdaten gewonnen werden. Shape-Signaturen werden hier sowohl aus quantisierten Direktionen (acht absolute Richtungen) als auch Positionen gebildet. Jeweils zwei HMM werden pro Gestenklasse trainiert, eines für quantisierte Winkel, eines für die Positionen. Selbstorganisierende Karten (engl. self organizing maps), eine Form neuronaler Netze, werden für die Rasterung der Koordinaten eingesetzt und bestimmen die Zustandszahl des zugehörigen

[16] Das Viterbi-Verfahren wird im Kontext der Hidden Markov Modelle eingesetzt, um wahrscheinlichste Zustandssequenzen einer Eingabe zu finden.

HMM. Das HMM für die Direktionen erhält seine Zustandszahl ebenso über die Quantisierung und Gruppierung gleicher Subsequenzen.

Bayes'sche Netze werden in [3] auf Skizzen angewandt. Zusätzliche Kontextinformationen fließen dabei in die Interpretation der Eingaben. 'GestureLab' [21], integriert in 'Cider' [87], verwendet als Standard-Erkennungsroutine die statistische Klassifikation mit Support Vector Machines[17] (SVM), um einfache, skizzierte Primitive bzw. Gesten zu erkennen. Es werden aus Eigenschaften der digitalen Tinte, wie Position, Zeit, Druck und Richtungen, ein Merkmalsvektor aus Länge der Geste, initialer Winkel und maximaler Krümmung extrahiert (Standard-Merkmalsset aus [170]), um die Klassenzuweisung durch SVM zu ermitteln. Allerdings warnt die Dokumentation der Software [20]: 'Additionally, typical training times for the default recognizer algorithms are in the order of hours (or even days) for problems of five-way classification or more with a reasonably large number of samples per class (e.g. 100).'

3.4.4 Nächste-Nachbar-Klassifikation

Neben obenstehenden, statistisch abgeleiteten Diskriminanzfunktionen lassen sich direkter Diskriminanzfunktionen[18] bestimmen, welche Entscheidungsgrenzen implizit festlegen. Eine solche Methode ist die Nächste-Nachbar-Klassifikation [63, S.3-7]. Dafür wird unter einem Distanzmaß das ähnlichste Muster (Template) aus einer Menge die verschiedenen Klassen repräsentierenden Beispiel-Instanzen gesucht. Das Objekt wird dann der Klasse dieses nächsten Nachbarn zugeordnet. Verfahren der Nächste-Nachbar-Klassifikation von Online-Eingaben bestimmen üblicherweise eine Shape-Signatur (Zeitreihe), normalisieren diese bezüglich verschiedener Varianzen und definieren ein lokales Distanzmaß für die Elemente der Zeitreihe sowie ein globales Distanzmaß für die Kumulation der lokalen Informationen.

Ein einfaches Nächstes-Nachbar-Verfahren ist es, den geometrischen (euklidischen) Abstand zwischen zwei Punktsequenzen als Ähnlichkeitskriterium zu nutzen. In [217] wird ein Single-Touch Klassifizierer vorgestellt, der Trajektorien einheitlich abtasted, bezüglich Position (Translation des Schwerpunktes auf den Koordinatenursprung), Skalierung und Rotation normalisiert und die Eingabe der Gestenklasse zuordnet, aus der ein nächster Nachbar über die kleinste Summe euklidischer Distanzen gefunden wurde. Die Normalisierung bezüglich der Rotation erfolgt dabei in einem ersten Schritt nach dem 'indikativen Winkel' zwischen Startpunkt und Schwerpunkt der Geste und darauffolgend über iterative Adaption auf die zu vergleichende Template-Geste. Die Skalierung einer Geste passt deren Ausdehnung einem Einheitsquadrat an. Der Ansatz liefert sehr gute Ergebnisse und die geforderten Invarianzen. Aufgrund der ungleichförmigen Skalierung, die wiederum der adaptiven Rotation geschuldet ist, werden allerdings eindimensionale Gesten (beispielsweise Linien) verzerrt. Zudem sind Unterscheidungen, zum

[17] Eine Einführung zu SVM wird beispielsweise in [37] anhand der Klassifikation von Ziffern gegeben.
[18] Die unterschiedlichen Sichten können durchaus zu gleichen Diskriminanzfunktionen bzw. Entscheidungsgrenzen führen.

Beispiel zwischen Kreis und Ellipse, aus den gleichen Gründen nicht möglich. Unterstützung für Multi-Stroke Gesten wurde in [6] durch Verbindung der aufeinanderfolgenden, einzelnen Trajektorien zu einer Single-Touch Geste und nachfolgender Klassifikation nach dem gleichen Verfahren untersucht. Der Ansatz hat den Nachteil, dass Mehrdeutigkeiten zu Single-Touch Gesten gleicher Form unvermeidbar sind [7]. Die Erkennung von Multi-Touch Eingaben ist auch unter diesen Anpassungen nicht möglich.

Eine ähnliche Vorverarbeitung wird in [119] angewendet, wobei keine Skalierung vorgenommen wird. Statt der Summe euklidischer Distanzen wird die Kosinus-Distanz zwischen Gesten berechnet, die invariant gegenüber Skalierung und Translation ist. Die Rotationsinvarianz wird über die Minimierung des Rotationsversatzes zwischen Template und Eingabe erreicht. Im Gegensatz zum iterativen Vorgehen in [217] wird dieser allerdings analytisch bestimmt, über einen Ansatz, der, obwohl nicht explizit erwähnt, in der sogenannten 'Prokrustes-Analyse' [161] zur Bestimmung der Ähnlichkeit von Formen genutzt wird. Die Autoren berichten einen erheblichen Performance-Gewinn[19] und ähnliche (beim Testset aus [217]) oder signifikant bessere (bei Verwendung eines eigenen Sets mit lateinischen Buchstaben) Klassifikationsraten. Eine weitere Variante, um iterative Suchen für die beste Rotation zu vermeiden, ist in [78] aufgeführt. Der Klassifizierer vergleicht Zeitreihen aus euklidischen Distanzen zum Schwerpunkt der Geste. Die z-normalisierten Abstände der Punkte einer Trajektorie zu ihrem Schwerpunkt und damit auch die Gestenerkennung sind invariant gegenüber Rotation, Skalierung und Translation.

Eine Variante, Shape-Signaturen aus Direktionsdaten zu gewinnen, wird in [103] verfolgt. Verläufe der Single-Touch Eingabe werden auf acht mögliche absolute Richtungen quantisiert. Die entstandene Sequenz aus Richtungen wird dann (parametrisiert) von eventuellem Rauschen (gleiche Subsequenzen kurzer Länge) befreit, bevor nachfolgend sämtliche Subsequenzen gleicher Richtungen zu jeweils einem Repräsentanten verschmolzen werden. Einfache planare Gesten können so durch direkten Abgleich der erzeugten Symbolfolge mit abgelegten Templates erkannt werden. Allerdings ist das Verfahren durch die Art der Rauschunterdrückung eher ungeeignet für Gesten, die aus Bögen bestehen. Multi-Stroke Gesten werden ähnlich dem Ansatz aus [6] ermöglicht, führen aber ebenso zu Mehrdeutigkeiten. Shape-Signaturen aus Sequenzen nominaler Merkmale werden auch in [39] eingesetzt. Vergleiche mit Templates werden allerdings über die sogenannte Levenshtein-Distanz [115] durchgeführt. Sie definiert den Abstand zweier Zeichenketten über die Anzahl Elementaroperationen (Löschen, Einfügen, Tauschen), die nötig sind, um eine Sequenz in die jeweils andere zu überführen. Dadurch kann mit verrauschten Daten besser umgegangen werden und eine Zuordnung zu einem Template fällt nicht binär über Un-/Gleichheit aus, sondern liefert ein Maß für die Ähnlichkeit bzw. Distanz. Coyette et al. [39] quantisieren die Koordinaten einer Trajektorie mittels eines Rasters und erzeugen eine Shape-Signatur als einen Verlauf der

[19] Allerdings nutzen die Autoren eine auf ein im Vergleich um ein Viertel reduzierte Abtastrate.

Direktionen über 8-Nachbarschaften der Zellen nach der Freeman-Kodierung [60] (auch Chain-Code). Ist die Abtast-Rate der Eingabe zu gering, wird die Trajektorie mit dem Bresenham-Verfahren [23] interpoliert, um den Kurvenverlauf durch Nachbarschaften adjazenter Zellen repräsentieren zu können. Subsequenzen gleicher Symbole (Ziffern) werden zu jeweils einem zusammengefasst. Die Klasse des Templates, welches mit minimalen Editierschritten aus der Eingabe (Levenshtein-Distanz) erzeugt werden kann, wird zurückgegeben. Symbole für das Absetzen des Stiftes verbunden mit gewichtetem Einfluss auf die Levenshtein-Distanz ermöglichen Multi-Stroke Skizzen[20].

Simistira et al. [193] nutzen ebenso den Freeman-Code, um handgeschriebene mathematische Symbole zu erkennen. In der Freeman-Codierung einer Trajektorie werden gleichermaßen zu [39] nur diejenigen Punkte belassen, die sich in der Sequenz zu einem vorhergehenden unterscheiden. Von diesen 'dominanten Punkten' werden zusätzlich zu ihrem Freeman-Kode (8-Nachbarschaft) noch die Distanz zum Vorgänger und damit der Länge des von ihnen repräsentierten Segmentes in der Shape-Signatur erfasst. Distanzen zwischen Templates und Eingabe werden über eine summierte punktweise lokale Distanz ermittelt. Multi-Stroke Eingaben sind möglich, indem bei einem Vergleich zwischen Template und Eingabe diejenige kombinatorische Zuordnung der Trajektorien gewählt wird, bei der die Distanz minimal ist.

Caridakis et al. [29] nutzen zusätzlich zum HMM-basierten Ansatz außerdem die Levenshtein-Distanz. In ihrer Arbeit wird diese Distanz zwischen den Shape-Signaturen quantisierter Direktionen und Positionen genutzt, um den Median - die Sequenz mit geringste Summe der Levenshtein-Distanzen zu allen anderen Sequenzen der Klasse - zu ermitteln.

Vergleiche zwischen Zeitreihen unter Beachtung nicht-linearer temporaler Verzerrungen zueinander können mit Dynamic Time Warping (DTW) durchgeführt werden. Das Verfahren ist, ebenso wie die HMM, in der Spracherkennung weit verbreitet. Distanzen zwischen Zeitreihen werden mit dieser Methode unter Minimierung der zeitlichen Verzerrungen berechnet. Dazu wird eine lokale Distanz zwischen Datenpunkten der Zeitreihe festgelegt und - unter gewissen Nebenbedingungen - die Zuordnung der Datenpunkte und damit die Kumulation der lokalen Distanzen optimiert. Levenshtein-Distanzen können als Spezialfall gesehen werden, bei dem Zeitreihen Symbolsequenzen sind, die lokale Distanz der Preis für einen Editierschritt ist und Elemente unter Strafzuschlag ausgelassen werden können. Ebenso können Nebenbedingungen das DTW so einschränken, dass Datenpunkte nur bijektiv zugeordnet werden. Somit ist beispielsweise die euklidische Norm als Spezialfall möglich, aber auch andere bereits vorgestellte Verfahren lassen sich als Spezialfall von DTW sehen.

Die Erkennung von Single-Touch[21] Buchstaben durch DTW wird in [36] untersucht.

[20] Die Arbeit befasst sich mit der Erkennung von Skizzen, es wird aber kein Unterschied zu Gesten gemacht.

[21] Multi-Stroke Eingaben werden abermals durch Aneinanderhängen der einzelnen Trajektorien und Single-Touch Klassifikation unterstützt.

Die Zeitreihen bestehen aus Folgen drei-dimensionaler Merkmalsvektoren mit jeweils dem absoluten Abstand der x- und y-Koordinaten der Trajektorie zu deren Startpunkt und dem Neigungswinkel am betreffenden Punkt. Während beim allgemeinen DTW mehrere Punkte einer Zeitreihe einem einzelnen der jeweils anderen zugeordnet werden können, erlauben die Autoren das nur bijektiv. Punkte können nur ausgelassen werden, wobei dies durch Erhöhung der Distanz bestraft wird. Sequenzen aus den Neigungswinkeln der Trajektorie von Formkonturen werden von Niblack und Yin [142] verwendet. Obwohl es nicht explizit erwähnt wird, werden Distanzen zwischen diesen Shape-Signaturen über einen DTW Ansatz berechnet. In [1] werden Formen mittels DTW Distanzen miteinander verglichen. Die Shape-Signaturen werden durch Folgen der Konvexitäts-/Konkavitätseigenschaften der Kontur in verschiedenen Skalierungsstufen gewonnen. Ein interessanter Ansatz, bei dem Repräsentanten aus den Templates (unterschiedlich erzeugter) planarer Gesten über deren Verteilung um eine 'Principal Curve' (Hauptkurve) bestimmt werden, wird in [214] untersucht. Temporale Informationen werden zunächst vernachlässigt, aber später einbezogen, um Mehrdeutigkeiten beispielsweise durch Selbstschneidungen von Trajektorien aufzulösen. Die Templates einer Klasse werden entlang ihres Prototypen in eine Sequenz von 'Fuzzy States' segmentiert. Die Zugehörigkeit einer Eingabe zu einem solchen Prototypen wird durch Ermittlung desjenigen mit der geringsten Distanz unter Anwendung dynamischer Programmierung bestimmt. Watt und Xie [206] erkennen mathematische Symbole unter Anwendung von DTW. In einem Vorverarbeitungsschritt werden die Merkmale Anzahl der Trajektorien[22], Startposition, Seitenverhältnis des umschließenden Rechtecks und die diskretisierten Winkel zwischen Start- und Endpunkt sowie am Anfang und am Ende der Eingabe genutzt, um die Zahl der zu vergleichenden Templates einzugrenzen. Dazu dürfen die entsprechenden Merkmale der Eingabe nicht über bestimmte Schwellwerte hinaus abweichen. Anschließend wird mittels 'Elastic Matching' unter den verbleibenden Templates dasjenige mit der minimalen Distanz gesucht.

3.4.5 Neuronale Netze

Neuronale Netze können beliebige Diskriminanzfunktionen lernen [50, S. 9]. Einen Überblick zu neuronalen Netzen liefern [22, 50]. Ebenso wie manche Formen des Template Matchings, lassen sich auch neuronale Netze aus dem Bayes'schen Ansatz der statistischen Klassifikation unter bestimmten Voraussetzungen generieren [22, S. 77].

Crossan und Brewster [41] entwickelten ein System, mit dem sich Formen und Gesten per haptischem und Audio-Feedback blinden oder sehbehinderten Nutzern vermitteln lassen. Um die Qualität der von den Nutzern gelernten Formen zu überprüfen, werden deren Eingaben klassifiziert. Die Autoren trainieren für die Unterscheidung zwischen 12 verschiedenen Klassen 3-schichtige neuronale Netze mit jeweils einer Zeitreihe aus den

[22] Durch das verwendete Eingabegerät wurden auch die Trajektorien der Bewegung aufgenommen und verwendet, bei denen kein Kontakt bestand.

Winkelverläufen der gleichmäßig abgetasteten (37 Punkte) Trajektorie einer idealisierten Eingabe.

Yuan et al. [224] erkennen Single-Touch Gesten mit Hilfe von Elman Netzen [53], mit denen es möglich ist, temporale Informationen einfließen zu lassen. Da die Eingabe auch für Menschen mit eingeschränkten motorischen Fähigkeiten möglich sein soll, werden Kontakte unterschiedlicher Art (z.B. Handballen) auf dem Eingabegerät zugelassen. Als Folge daraus werden Merkmale in Form verschiedener Momente des Blobs, seiner Orientierung und dessen Trajektorie sowie deren über die Zeit kumulierten Orientierungen extrahiert. Wie in [169] werden Merkmale von Direktionen über ihren Sinus und Kosinus erfasst, um bei Radianten oder Gradmaßen inhärente Diskontinuitäten zu vermeiden. Für jeweils eine der 10 unterscheidbaren Gesten werden 25 Templates trainiert.

Nachfolgend wird auf für die vorliegende Arbeit relevante Konzepte aus dem Stand der Technik hinsichtlich der Klassifikation von Gesten näher eingegangen. Im Bezug zur Abbildung 3.1 zu Beginn des Kapitels 3 wird dabei von überwachtem Lernen mit vollständigen, gelabelten Trainingsdaten ausgegangen. Eine Klassifikation ist wesentlich abhängig von den zuvor beschriebenen Prozessen der Zerlegung von Tracking-Daten in kleinste bedeutungstragende Einheiten und der Gewinnung von Merkmalen aus derartigen Token. Der Verfasser dieser Arbeit verfolgt einen hierarchischen Ansatz, in dem Token Trajektorien von Gesten darstellen, die durch lokale Merkmale repräsentiert werden, um sie mittels eines Nächste-Nachbar-Verfahrens vergleichen zu können. Weitere Merkmale bilden die strukturellen Zusammenhänge der Token ab, um sie in einem statistischen Ansatz zu kombinieren. Methodisch wird daher speziell auf die statistische sowie Nächste-Nachbar-Klassifikation Bezug genommen und versucht, die Vorteile aus beiden Bereichen zu verbinden.

4

Universelle Klassifikation planarer Gesten

In diesem Kapitel werden Methoden näher untersucht, welche nach Ansicht des Autors geeignet sind, das eingangs gestellte Problem zu lösen. Die Auswahl erfolgt anhand der Anforderungen zum Training, der Echtzeitfähigkeit und der Erwartungen an die Klassifikationsgenauigkeit. Aufbauend auf den Voruntersuchungen wird ein eigener Klassifikator entwickelt. Das Problem der Erkennung sequenzieller Multi-Touch Gesten wird in die Klassifikation einzelner Trajektorien und deren hierarchische Verwendung in einem statistischen Ansatz zerlegt. Im Detail werden Trajektorien mittels Nächste-Nachbar-Methoden klassifiziert und deren Komposition mit einer Bayes'schen Strategie erkannt. Diese Aufteilung erlaubt es, globale Merkmale zu vermeiden, welche schneller zu Mehrdeutigkeiten führen können, wenn das Gestenset anwächst. Da die Spezifizierung neuer Gesten allein durch Templates möglich sein soll, lässt sich nicht einkalkulieren, welcher Art die Gesten sind, die Verwendung finden. Globale Merkmale, wie sie in Regel-basierten Ansätzen häufig genutzt werden, eignen sich daher nur bedingt. Zusätzlich kann das Set der Merkmale, auch durch den konstruktiven Ansatz der Erstellung, kleiner gehalten werden, als es häufig in der Literatur, etwa in [169, 213, 44], der Fall ist. Auch in [213] findet sich die Erkenntnis, dass Ansätze, welche auf globalen Merkmalen basieren, unzureichend in ihrer Unterscheidungsfähigkeit von Multi-Stroke Eingaben sind, während hingegen unter Einbezug lokaler Merkmale auf Trajektorie-Ebene signifikante Verbesserungen möglich sind.

Insgesamt versprechen nach Recherche der Literatur die Klassifikation auf Basis statistischer Verfahren oder die Nächste-Nachbar-Suche den größten Erfolg. Es finden sich zudem in [213] Hinweise, dass die Nächste-Nachbar-Klassifikation über DTW eine sehr gute Strategie darstellt. Der auf einem Bewegungsmodell basierende Ansatz zur

Gestenerkennung aus [178] wurde ebenda mit dem statistischen Klassifizierer in [169] verglichen, der bei dem kleinen Gestenset von vier verschiedenen Gesten bessere Resultate (96,75% gegenüber 99,25%) erzielt. Allerdings wird betont, dass ein geeigneteres zugrunde liegendes Modell den Ansatz wesentlich verbessern würde. Ebenso wird in [208] festgestellt, dass statistische Klassifizierer, wie der von Rubine [169] oder SVM - neben teuren Ansätzen mit neuronalen Netzen - besser als andere angewandte Methoden für die Erkennung von Skizzen geeignet sind. Für die Klassifizierung mittels Shape-Signaturen sind nach [228] Strategien, die Verteilungen lernen (speziell HMM) sowie Nächste-Nachbar-Methoden gut geeignet. Neuronale Netze werden hier nicht weiter betrachtet, da die Anzahl der benötigten Trainingsdaten vergleichsweise hoch ist. Bei eigenen Experimenten erwiesen sich HMM im Vergleich zu den getesteten Nächste-Nachbar-Methoden als wenig vielversprechend. Letztere kommen im Allgemeinen mit einer geringeren Trainingsmenge aus und erzielten bei den Tests bessere Klassifikationsraten. Zudem haben sowohl HMM als auch neuronale Netze den Nachteil, dass ihre Struktur kaum aus Trainingsdaten abgeleitet werden kann und manuelle Anpassungen nötig sind. In den meisten in der Literatur verwendeten Verfahren zur Gestenerkennung wird beispielsweise die Art des Modells der HMM (oft nach Bakis) festgelegt. Auch die Anzahl der benötigten Zustände kann zwar mittels der Trainingsdaten abgeschätzt werden, wird aber häufig entsprechend feinjustiert und festgelegt. Für eine flexible Erkennung frei durch Templates spezifizierbarer Gesten ist das ein Nachteil. Quantisierungsfehler durch die Einteilung einer Geste in diskrete Abschnitte lassen des Weiteren Einbußen in der Genauigkeit erwarten.

Die hier verfolgte Idee basiert zunächst auf der Erkennung der Bestandteile (Token) einer Geste über Template-Vergleiche und der anschließenden Bewertung der Wahrscheinlichkeit ihrer Verteilung und Ausrichtung innerhalb der Geste. Der Vorteil, lokale Merkmalssets mit geringerer Größe verwenden zu können, schafft das Problem, die Token einer Eingabe denen eines Gestentemplates zuzuordnen. Hierauf wird bei der Beschreibung der hierarchischen Klassifikation Bezug genommen. Zunächst sollen im Folgenden Nächste-Nachbar-Methoden zur Erkennung der Bestandteile einer Geste näher untersucht werden. Diese Token sind dabei die Trajektorien und die Prozeduren können demzufolge bei Single-Touch Gesten Anwendung finden. Ein Großteil dieses Kapitels zur nicht-parametrischen Nächste-Nachbar-Klassifikation von Single-Touch Eingaben ist unter [184] veröffentlicht.

4.1 Nächste-Nachbar-Klassifikation von Single-Touch Gesten

Template-basierte Methoden der Online-Klassifikation haben die Teilprobleme der Normalisierung, der Merkmalsextraktion einer Shape-Signatur sowie der Definition eines lokalen Distanzmaßes für die Elemente der Zeitreihe und eines globalen Distanzmaßes

für deren Kumulation zu lösen; siehe etwa [142, 36, 1, 119, 217, 78] und Kapitel 3.4.4.

Xi et al. [221] kommen nach ihrer Recherche und eigenen Tests zu dem Schluss, dass DTW-Strategien bezüglich der Genauigkeit in der Klassifikation von Zeitreihen anderen Methoden überlegen sind. Auch in [213] wird DTW als ein - in dieser Version vergleichsweiser teurer - akkurater Referenz-Klassifizierer verwendet, um die Genauigkeit anderer Strategien bei der Erkennung ikonischer Gesten zu evaluieren. Beim Vergleich ihres in der Literatur erfolgreichen und populären Verfahrens mit einem einfachen DTW-Ansatz kommen Wobbrock et al. [217] zu ähnlichen Ergebnissen, Kritikpunkt ist abermals die Geschwindigkeit des Verfahrens. Tatsächlich lassen sich die meisten ähnlicher Nächste-Nachbar-Methoden als Spezialfälle eines DTW-Ansatzes auffassen. Das Verfahren wird im Folgenden genauer erklärt. Danach werden in der Literatur verfügbare Klassifikatoren eingeordnet und systematisiert, um abschließend einen detaillierten Vergleich gebräuchlicher und eigener Ansätze zu ermöglichen.

4.1.1 Dynamic Time Warping

Im Prinzip berechnet Dynamic Time Warping mit Hilfe des Ansatzes der dynamischen Programmierung eine (gegenüber zeitlicher Verzerrungen invariante [65]) Distanz zwischen nicht zwingend gleichlangen Zeitreihen. Die Datenpunkte der hier betrachteten Zeitreihen sind Merkmalsvektoren oder als Spezialfall Skalare mit extrahierten Merkmalen der Geste. Die Zeitreihe kann dabei durch die Sample-Rate vorgegeben oder künstlich, beispielsweise durch Codierung des Verlaufs, erzeugt werden. Für die allgemeine Beschreibung seien im Folgenden zwei derartige Zeitreihen als Sequenzen von Vektoren gegeben:

$$X = \vec{x_1}, \vec{x_2}, ..., \vec{x_m}$$

$$Y = \vec{y_1}, \vec{y_2}, ..., \vec{y_n}$$

Um die Distanz beider Verläufe unter Berücksichtigung etwaiger (nichtlinearer) zeitlicher Verzerrungen zu ermitteln, wird entlang des zeitlichen Verlaufes beider Reihen ein Mapping mit nicht notwendigerweise eineindeutigen paarweisen Zuordnungen $(\vec{x_i}, \vec{y_j})$ oder kurz (i,j), $1 \leq i \leq m$, $1 \leq j \leq n$ unter Nebenbedingungen entwickelt. Hier sei die Notation aus [173, S. 43,44] verwendet, in der das Mapping F (die Warping-Funktion) als eine Sequenz von Index-Paaren[23] gegeben ist:

$$F = c(1), c(2), ..., c(K) \; mit \; c(k) = (i(k), j(k)),$$

$$1 \leq k \geq K, \; 1 \leq i(k) \leq m, \; 1 \leq j(k) \leq n$$

Unter einem lokalen Distanzmaß[24] $d(i,j)$ und einer Gewichtung $w : k \rightarrow \mathbb{R}$ ist

[23] Ohne zeitliche Verzerrung wäre diese $i(k) = j(k), \; \forall k$ und $m = n$

[24] Häufig und so auch in [173] wird die absolute Distanz $d(i,j) = d(\vec{x_i} - \vec{y_j})$ genutzt. Auch (quadrierte) euklidische Distanzen kommen oft zur Anwendung [221].

eine die zeitlichen Verzerrungen minimierende DTW-Distanz zweier Zeitreihen für eine Warping-Funktion nach [173] folgendermaßen gegeben:

$$d(X,Y) = \min_{F} \frac{\sum_{k=1}^{K} d(c(k)) \cdot w(k)}{\sum_{k=1}^{K} w(k)} \tag{4.1}$$

Die Realisierung der Warping-Funktion F ist der eigentliche Kern des DTW und wird meist durch folgende Nebenbedingungen eingeschränkt [173]:

$$Monotonie: \quad i(k-1) \leq i(k)),\ j(k-1) \leq j(k) \tag{4.2}$$

$$Stetigkeit: \ i(k) - i(k-1) \leq 1,\ j(k) - j(k-1) \leq 1 \tag{4.3}$$

$$Begrenzung: \ i(1) = 1,\ j(1) = 1,\ i(K) = m,\ j(K) = n \tag{4.4}$$

Die Monotonie-Bedingung sorgt dafür, dass die Reihenfolge der Signale bzw. auftretenden Merkmalsvektoren unangetastet bleibt. Die Stetigkeits- sowie die Randbedingung garantieren, dass keine Zeitsprünge stattfinden bzw. alle Signale in beiden Zeitreihen einbezogen werden.

Abbildung 4.1 illustriert das Matching zweier Gesten bei punktweiser (lokaler) euklidischer Distanz unter Minimierung der zeitlichen Verzerrungen nach den obigen Bedingungen.

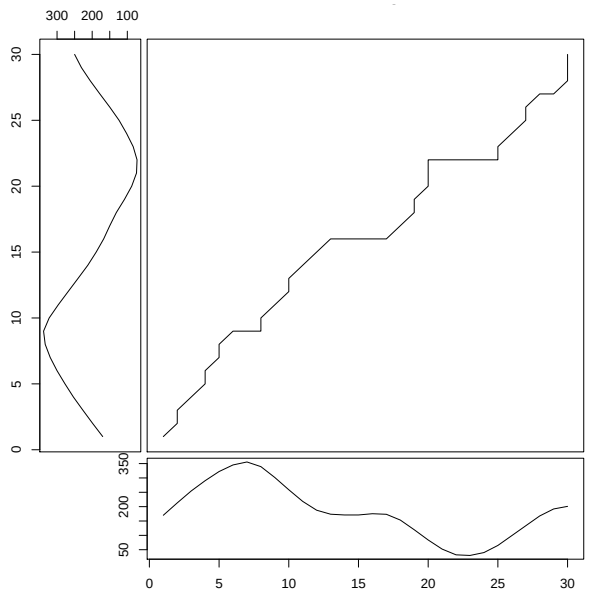


Abbildung 4.1: Der Warping-Pfad veranschaulicht die paarweise Zuordnung der Punkte beider Zeitreihen. Horizontale oder vertikale Abschnitte implizieren die Zuordnung mehrerer Punkte einer Zeitreihe zu einem Punkt der anderen. Diagonale Abschnitte stehen für bijektive (eineindeutige) Zuordnungen in aufsteigender Index-Folge.

Ohne weitere Restriktionen darf demnach der sogenannte Warping-Pfad F, der durch (i,j) geht, dies von (i-1,j),(i,j-1) oder (i-1,j-1) aus. Lokale Beschränkungen werden als Schrittmuster für den Warping-Pfad umgesetzt. Werden keine Gewichtungen verwendet,

berechnen sich die minimalen Kosten C(i,j) der Warping-Funktion F (die Distanz der Zeitreihen) bis zu einem Index-Paar (i,j) unter diesen Bedingungen nach:

$$C(i,j) = \min(C(i,j-1), C(i-1,j-1), C(i-1,j)) + d(i,j) \tag{4.5}$$

Die minimalen Kosten des spezifizierten Mappings ergeben sich durch Berechnung von C(m,n). Die Suche nach den minimalen Pfadkosten und der somit die Distanz minimierenden Zuordnung zwischen den Zeitreihen kann über die Technik der dynamischen Programmierung iterativ und effizient in $\mathcal{O}(mn)$ berechnet werden. Mit Hilfe unterer Schranken [94] oder den Suchraum einschränkenden Nebenbedingungen sind sehr performante Umsetzungen möglich. In [55] (dort unter 'Nonlinear Elastic Matching') wird nachgewiesen, dass im Allgemeinen die DTW-Distanz keine Metrik ist, aber unter bestimmten Annahmen eine abgeschwächte Form der Metrik angenommen werden kann. Tatsächlich wurde in empirischen Tests für symmetrische Formen des DTW und einem großen Datensatz aus dem Bereich der Spracherkennung keine Verletzung der Dreiecksungleichung festgestellt [31]. Unter diesen Gesichtspunkten und unter Verwendung von Indexing-Techniken [94] gibt es großes Potenzial für weitere Steigerungen der Effizienz. Solche Ansätze sollen hier nur erwähnt werden, aber keine weitere Betrachtung finden, da im Bereich der Gesten nicht mit sehr großen Datensätzen zu rechnen ist.

Hauptsächlich im Kontext der Spracherkennung wurden modifizierte und zusätzliche Nebenbedingungen untersucht [173, 158]. So sind lokale und globale Beschränkungen möglich, um zu starke Entzerrungen beim Zeitreihenvergleich zu vermeiden, die etwa dazu führen, dass Wörter als gleich angesehen werden, welche sich nur durch verschiedentlich gedehnte Sprechweise unterscheiden. Falls die Gestenerkennung beispielsweise nicht vollständig invariant gegenüber Geschwindigkeitsänderungen während der Eingabe sein soll, lässt sich das Argument übertragen. Spezielle Schrittmuster (engl. step pattern) und Fensterfunktionen (engl. warping windows) begrenzen die Art und den Umfang der gegenseitigen zeitlichen Entzerrung der Signale. Aber auch lockernde Nebenbedingungen, wie mögliches Auslassen von Beobachtungen [115, 36] oder das Zulassen partieller Übereinstimmungen [65], können sinnvoll sein.

In [139] werden verschiedene Ansätze lokaler Nebenbedingungen für die Erkennung gesprochener Wörter gegenübergestellt und auf ihre Wirkung untersucht. Die Autoren kommen dabei zu dem Schluss, dass die Unterschiede in der Qualität der Erkennung zwischen ihren untersuchten Varianten entweder gering sind oder im komplexesten Fall zur Verschlechterung tendieren. Globale Einschränkungen des Warping-Pfades verbessern zwar die Performance, gingen aber auf Kosten der Genauigkeit. Die Einschränkung des Suchraumes lässt allerdings nicht nur performantere Anwendungen zu. Es gibt auch Untersuchungen, die belegen, dass die Genauigkeit einer Klassifikation dadurch verbessert werden kann [163]. In [94] wird dies mit der Verhinderung pathologischer Entzerrungen begründet, welche einen kleinen Teil einer Zeitreihe einem zu großen der anderen zuordnen.

Xi et al. [221] stellen fest, dass tatsächlich, im Gegensatz zur Intuition, nicht durch eine Fensterfunktion beschränkte Warping-Pfade eines Zeitreihenvergleiches die Fehlerrate erhöhen.[25] Die beste Größe des Fensters nach [173] ist in den Tests der Autoren eher klein (nicht größer als 10%) und nimmt mit der Größe der Trainingsdaten ab. Weiterhin wird in [221] bemerkt, dass zu strenge Einschränkungen, wie für den Spezialfall des nur diagonal zugelassenen Warping-Pfades unter (quadrierten) euklidischen lokalen Distanzen, die Genauigkeit üblicherweise ebenfalls verschlechtern. Obwohl es Fälle gibt, in denen die euklidische Norm der Differenz zweier Zeitreihen unbeschränktes DTW übertreffen kann[26], wird die Überlegenheit des letzteren Verfahrens auch in [205] belegt. In [76] wurde zur Verifikation von Unterschriften eine optimale Breite des Fensters nach [173] von ebenfalls 10% empirisch ermittelt. Keogh und Ratanamahatana [95] geben diese Begrenzung als die in der Literatur übliche Wahl an. In eigenen Klassifikationstest zeigen die Autoren, dass der Spezialfall des diagonalen Warping-Pfades der Klassifikation unter größerem Fenster unterlegen ist, die optimale Fensterbreite aber stark problemabhängig scheint. Bei einem 2-Klassen-Problem zur Erkennung von Betriebsstörungen in Atomkraftwerken anhand der Daten zweier Sensoren zeigte sich die höchste Genauigkeit ab etwa 3% der Länge der Datenreihen als Fensterbreite und fiel mit größerem Fenster nicht mehr ab. Für ein Problem zur Erkennung von vier Wörtern in merikanischer Zeichensprache anhand der rechtshändischen Bewegung entlang der x-Achse allerdings stellte sich eine Fenstergröße von etwa 15-18% als optimal heraus.

Als letzter Aspekt sei noch die Normalisierung der DTW-Distanz bezüglich zeitlicher Entzerrungen betrachtet. Je mehr Verzerrung korrigiert wird, desto länger wird der Warping-Pfad (siehe Abbildung 4.1) und um so mehr Datenpaare aus beiden Zeitreihen gehen in die Distanzberechnung ein. Um diesen Effekt zu korrigieren, geht in Gleichung 4.1 ein Normalisierungsterm ein. Eine solche Normalisierung ist allerdings nicht für jedes Schrittmuster verfügbar, ohne die quadratische Laufzeit des DTW-Verfahrens zu verlassen. Der in Abbildung 4.1 gezeigte Pfad beispielsweise wurde unter dem in Gleichung 4.5 implizierten Schrittmuster ohne Gewichtungen (bzw. mit $w(k) = 1, \forall k$) erzeugt. Der Normalisierungsfaktor ist in diesem Fall nicht konstant und ohne Wissen der Pfadlänge nicht bekannt [65], kann demnach also nicht im Voraus bestimmt werden. Würden Diagonal-Schritte mit $w(k) = 2$ gewichtet, so können die DTW-Distanzen mit dem Term $\sum_{k=1}^{K} w(k) = n + m$ für Gleichung 4.1 gewählt werden, um Distanzen zu einem Wertebereich von [0..1] zu normalisieren.

Im Folgenden werden konkrete Realisierungen verschiedener DTW-Ansätze gegenübergestellt. Dabei werden in der Literatur verfügbare und hier als Spezialfälle von DTW aufgefasste Nächste-Nachbar-Klassifikatoren untersucht und systematisiert. Eine Einordnung erfolgt anhand der gewählten Merkmale und zugehörigen lokalen Distanzfunktionen. Neben derart definierten Shape-Signaturen werden auch weitere, aus

[25] Diese Erkenntnis wird teilweise durch [205] gestützt, da für verschiedene Testdaten die Resultate beider Methoden nah beieinander liegen.

[26] Interessanterweise werden vergleichbare Resultate mit der Manhattan-Distanz erreicht.

vom Autor gewählten Merkmalen konstruierte, einbezogen. In einem Vergleich werden die für die Klassifikation von Single-Touch Eingaben geeignetsten Ansätze ermittelt und die Möglichkeit der Verbesserung der Klassifikationsgenauigkeit durch DTW untersucht. Außerdem wird der Einfluss von Fensterfunktionen und Schrittmustern näher betrachtet.

4.1.2 Shape-Signaturen

Im Trivialfall stellen Trajektorien, also Sequenzen zeitgestempelter Koordinaten, Shape-Signaturen dar, wie beispielsweise in [217, 119] verwendet. Quantisierte Koordinaten [39] oder Zellen eines traversierten Rasters [220] sind Abstraktionen dieser Informationen. Um allerdings verschiedene Arten von Invarianzen zu erhalten, müssen Shape-Signaturen aus Positionierungen vorverarbeitet werden.

Caridakis et al. [29] nutzen Verläufe quantisierter Richtungen und Positionen. Chain-Codes bzw. Freeman-Codierungen in Form von Sequenzen relativer oder absoluter Winkel sind sehr gebräuchliche Shape-Signaturen in der Online-Klassifikation von Formen und Trajektorie-basierten Eingaben [99, 142, 45, 36, 230, 207]. Häufig werden in den Signaturen gleiche, aufeinanderfolgende Beobachtungen verworfen [103, 207] oder nur markante Punkte der Trajektorie einbezogen [193]. Diese Art der Codierungen haben den Vorteil, inhärent invariant gegenüber Translation und Skalierung sowie, falls relative Messungen Verwendung finden, Rotation zu sein.

Punktweise Distanzen zum Startpunkt [36] oder Schwerpunkt [78] sind weitere Repräsentationen, welche invariant gegenüber Rotation, Translation und, falls sie in normalisierter Form verwendet werden, Skalierung sind. In [36] werden dafür, zusätzlich zum Winkel am betrachteten Punkt, absolute Abstände jeweils der x- und y-Koordinate verwendet. Es wird ein hoher, empirisch bestimmter Gewichtungsfaktor auf das direktionale Merkmal bei der Berechnung der lokalen Distanz als Linearkombination der einzelnen Merkmals-Distanzen angewendet, was auf dessen bessere Eignung hindeutet. Diese Vermutung wird auch durch die ebenfalls guten Ergebnisse der allein auf der Kosinus-Distanz basierenden Klassifizierung in [119] gestützt.

Abbildung 4.2 gibt einen Überblick zu den verschiedenen in der Literatur verfügbaren oder eigenen Strategien, wie Shape-Signaturen aus Sequenzen von Positionen, Winkeln oder Distanzen konstruiert werden können. In der ersten Reihe (1-3) der Abbildung finden sich euklidische Distanzen der Punkte einer Trajektorie vom Startpunkt (1) ähnlich dem Ansatz von [36] (dort werden Manhattan-Distanzen genutzt), dem globalen Schwerpunkt (2) [78] oder einem inkrementellen Schwerpunkt[27] (3).

[27] Als inkrementeller Schwerpunkt bezeichnet wird hier ein Schwerpunkt aller Punkte einer Trajektorie von derem Startpunkt bis zum aktuell beobachteten Punkt, dessen Merkmal gewählt wird. Es wird erwartet, dass diese Version eines Bezugspunktes nicht so robust ist wie der eigentliche Schwerpunkt. Allerdings kann damit eine Shape-Signatur inkrementell zur Eingabe und somit auch für partielle Eingaben berechnet werden.

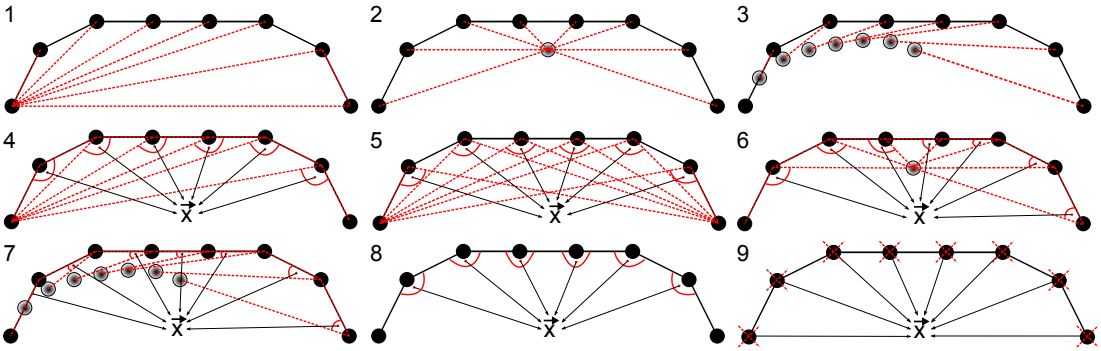


Abbildung 4.2: Merkmale, die für die Erzeugung von Shape-Signaturen zur nachfolgenden Klassifizierung mittels DTW geeignet scheinen. Schwarze Punkte verdeutlichen den Verlauf einer Trajektorie von links nach rechts. Graue Punkte stehen für den Schwerpunkt der (Teil-)Eingabe. Die gewählten Merkmale werden durch gestrichelte Linien (Distanzen in 1-3) und Bögen (Winkel in 4-8) oder Kreuze (Positionen in 9) markiert.

Verschiedene, gegenüber Skalierung und Rotation invariante Winkel werden durch die vom Autor definierten Merkmale 4 bis 7 dargestellt. Die Merkmale 4 und 5 repräsentieren den eingeschlossenen Winkel zwischen Startpunkt, Punkt der Trajektorie und seinem Nachfolger (4) beziehungsweise dem Endpunkt der Trajektorie (5). Die Merkmale 6 und 7 definieren an jedem Punkt der Trajektorie den Winkel zwischen dem Vorgänger und dem globalen (6) oder inkrementellen (7) Schwerpunkt. Merkmal 8 repräsentiert typische Chain-Codes [103, 45], wobei hier absolute Winkel[28] genutzt werden. Merkmal 9 schließlich steht für die Punkte der Trajektorie selbst [217, 119].

Jedem Merkmal aus Abbildung 4.2 ist im Folgenden eine Metrik zugeordnet, die als lokale Distanz (siehe Gleichung 4.1) verwendet wird. Für die Abstände (Merkmale 1-3) wird die absolute Differenz $|x_i - y_j|$ gewählt. Beim direkten Vergleich zwischen Punkten der Trajektorien (Merkmal 9) wird die Manhattan-(9a) bzw. die euklidische (9b) oder quadrierte euklidische Distanz (9c) sowie die Kosinus-Distanz (9d) angewendet. Distanzen zwischen Winkelmaßen (Rad) werden über eine spezielle Metrik (Gleichung 4.6) berechnet und zu [0..1] normalisiert (im Folgenden Angular-Distanz genannt).

$$d(i,j) = \begin{cases} \frac{1}{\pi}|x_i - y_j|, & |x_i - y_j| < \pi \\ \frac{1}{\pi}(2\pi - |x_i - y_j|), & otherwise \end{cases} \tag{4.6}$$

Für die Merkmale, die keine Winkel repräsentieren, wird ebenfalls eine Normalisierung vorgenommen. Im nächsten Abschnitt werden die angewandten Normalisierungs- und Vorverarbeitungsschritte konkreter dargelegt.

4.1.3 Vorverarbeitung und Normalisierung

Eine Normalisierung bezüglich der Eingabegeschwindigkeit wird durch Abtastung (Resampling) der Trajektorie auf einheitlich 64 Beobachtungen ermöglicht. Diese Anzahl hat sich in [217] bewährt und stellt damit auch eine Vergleichbarkeit mit den dortigen Ergebnissen her. Des Weiteren ist die Anzahl ausreichend, um die Eingabe gut zu

[28] Relative Winkel sind in ihrem Verlauf zu chaotisch und haben sich als nicht zweckmäßig erwiesen.

repräsentieren. Häufig sind weniger Beobachtungen vorhanden und die Zahl der Beobachtungen in einer Eingabe wird durch lineare Interpolation erhöht. Hinderlich ist die Normalisierung durch Abtastung nur für Anwendungsfälle, in denen die zeitlichen Informationen eine Rolle spielen. So wird in [76] berichtet, dass die Verifikation von Unterschriften insofern durch Resampling beeinträchtigt wird, als dass die häufig längere Eingabezeit als ein Indiz für Fälschungen eliminiert wird. Ansonsten ist hinsichtlich der Genauigkeit kein signifikanter Unterschied beim Vergleich von Zeitreihen unterschiedlicher Längen gegenüber dem von auf gleiche Längen interpolierten zu erwarten [162]. Zu erwähnen ist, dass nach gleichabständiger Abtastung für Schrittmuster, welche einen nur von der Länge der Zeitreihen abhängigen Normalisierungsfaktor erlauben, eine solche Normalisierung nicht mehr relevant ist und sie hier auch nicht vorgenommen wird.

Die Shape-Signaturen aus Abbildung 4.2 weisen unter den verwendeten Metriken verschiedene Invarianzen bezüglich Variationen der Eingabe auf. Tabelle 4.1 gibt einen Überblick über die neben dem Resampling nötigen Vorverarbeitungsschritte für jedes Merkmal, um alle gewünschten Invarianzen zu erhalten. Die Bezeichnungen der Merkmale werden im Folgenden auch für die aus ihnen gewonnenen, normalisierten Shape-Signaturen im Verbund mit den zugeordneten lokalen Distanzen verwendet.

Tabelle 4.1: Charakteristiken der Merkmale (erste Spalte) bei Verwendung für Shape-Signaturen im Rahmen der DTW-Klassifikation. Die Tabelle weist jedem in Abbildung 4.2 repräsentierten Merkmal seinen Typ, das verwendete Distanzmaß und die nötigen Vorverarbeitungsschritte zum Erhalt der vorausgesetzten Invarianzen zu.

Shape-Signatur	Lokales Merkmal	Lokale Distanz	Vorverarbeitung
1	Distanz	Absolut	Skalierung
2	Distanz	Absolut	Skalierung
3	Distanz	Absolut	Skalierung
4	Winkel	Angular	-
5	Winkel	Angular	-
6	Winkel	Angular	-
7	Winkel	Angular	-
8	Winkel	Angular	Rotation
9a	Punkt	Manhattan	Transl., Skal., Rot.
9b	Punkt	euklidische	Transl., Skal., Rot.
$1 (9b)	Punkt	euklidische	Transl., Skal., Rot.
Prokrustes (9c)	Punkt	quad. euklid.	Transl., Skal., Rot.
Protractor (9d)	Punkte	Kosinus	Rotation

Distanzmaße (Merkmale 1-3) sind invariant gegenüber Rotation, Translation, aber nicht gegenüber der Skalierung. Bei Division durch die größte Distanz innerhalb der Zeitreihe werden alle enthaltenen Beobachtungen in den Wertebereich [0..1] normalisiert. Eine weitere mögliche Normalisierung, die diese Skalierung bewirkt, ist die Division durch die Gesamtlänge des Pfades. Vorabtests zeigten allerdings, dass eine gleichförmige Skalierung der Trajektorien vor der Merkmalsextraktion eine wesentlich höhere (mindestens 15%-ige) Genauigkeit der Klassifizierung erzielt. Die Winkel (Merkmale 4-7) sind invariant gegenüber Translation, Skalierung und Rotation. Ausnahme bildet Merkmal 8. Da hier absolute Winkel gemessen werden, ist eine Invarianz nur gegenüber

Translation und Skalierung gegeben. Im Fall der direkten Distanzen zwischen Punkten der Trajektorien - wie für die Merkmale 9a, 9b und 9c - wird eine Normalisierung bezüglich aller drei geforderten Invarianzen benötigt. Für die Kosinus-Distanz (9d) hingegen reicht die Normalisierung bezüglich der Rotation.

Eine notwendige Normalisierung bezüglich Translation, Skalierung und Rotation wird, unter Ausnahme des $1-Klassifizierers, nach der entsprechenden Methode im Verfahren der Prokrustes-Analyse [49] vorgenommen. Tatsächlich entspricht das Vorgehen bei Merkmal 9c diesem Verfahren, insofern beim DTW der Warping-Pfad nur durch die Diagonale zugelassen wird. Die Umsetzung der Skalierung ist in Algorithmus 4.1 in modifizierter Form in Pseudocode wiedergegeben. Gleichzeitig wird die Translation vorgenommen, sodass der Schwerpunkt der Trajektorie im Koordinatenursprung liegt. Es wird davon ausgegangen, dass ein Resampling bereits stattgefunden hat.

Algorithm 4.1 ProcrustesScaleUniformly(T,s)

Require: INPUT: T - Trajektorie des eingegebenen Token
Require: INPUT: s - Skalierungsvariable, Ziel der Skalierung
▷ n ist die Anzahl der Samples in der Trajektorie T
n ← $|T|$
mean ← GetMean(T[i])
▷ Berechnung des quadratischen Mittels der Distanzen der Punkte einer Trajektorie zu deren Schwerpunkt
distance ← 0
for all $i = 1$ to n **do**
    distance ← distance + SquaredEuclideanDistance(T[i],mean)
    T[i] ← T[i]-mean
end for
distance ← $\sqrt{distance}$
scale ← $distance \cdot \sqrt{1/s}$
for all $i = 1$ to n **do**
    T[i] ← T[i]/scale
end for
return T

Die Skalierungsvariable in Algorithmus 4.1 ist die Zielgröße und entscheidet, wie groß das quadratische Mittel der Distanzen aller Punkte einer Trajektorie zu deren Schwerpunkt nach der Skalierung sein soll. Für den Vergleich zweier Shape-Signaturen ist nur die Skalierung auf die gleiche Größe von Bedeutung. Im Kapitel 4.2 wird der Wert benutzt, um die Summe quadrierter euklidischer Distanzen zweier Shape-Signaturen zu normalisieren (ein Wert von 1 normiert diese zu [0..1], ein Wert von 0,5 zu [0..2]).

Sind zwei Trajektorien auf die gleiche Größe skaliert, können sie bezüglich ihrer Rotation ebenfalls nach der entsprechenden Methode aus der Prokrustes-Analyse normalisiert werden. Der Pseudocode (nach [161]), um zwei gleichlange Trajektorien bestmöglich (im Sinne geringster quadrierter euklidischer Distanzen) zueinander bezüglich ihrer Orientierung auszurichten, findet sich in Algorithmus 4.2.

Algorithm 4.2 ProcrustesRotateToMatch(I,G)

Require: INPUT: I - Trajektorie des eingegebenen Token
Require: INPUT: G - Trajektorie des Tokens eines Templates
▷ n ist die Anzahl der Samples jeweils in den Trajektorien T und G
n ← $|T|$
numerator ← 0
denominator ← 0
for all $i = 1$ to n **do**
numerator ← numerator + (G[i].X · I[i].Y - G[i].Y · I[i].X)
denominator ← denominator + (G[i].X · I[i].X + G[i].Y · I[i].Y)
end for
▷ α enthält den Versatz hinsichtlich der Rotation der beiden Trajektorien I und G zueinander
α ← ATAN(numerator, denominator)
α ← ROTATE(I, $-\alpha$)
return I

Auch die beiden Verfahren aus [217, 119] können als Spezialfälle des DTW unter Merkmal 9 gesehen werden. Protractor [119] vereint dabei die Punkte der Trajektorie zu einem Punkt im Raum und nutzt die lokale (und gleichermaßen globale) Kosinus-Distanz zwischen zwei derart definierten Punkten als Vergleichsmaß (lokale oder globale Nebenbedingungen sind dabei nicht möglich). Der $1-Klassifizierer [217] ließe demgegenüber gelockerte lokale oder globale Nebenbedingungen in der Distanzberechnung zu. Das Verfahren ist allerdings sehr ähnlich zur Prokrustes-Analyse und unterscheidet sich in einer weniger formalen Vorverarbeitung. Ebenso entspricht DTW unter Merkmal 9b und nur zugelassenem diagonalen Warping-Pfad der Prokrustes-Analyse mit dem Unterschied, dass euklidische statt quadrierte euklidische lokale Distanzen verwendet werden. Es zeigte sich, dass dies eine Verbesserung zur originalen, statistisch hergeleiteten Prokrustes-Analyse darstellt. Aus diesen Gründen werden die drei in Tabelle 4.1 benannten Verfahren als Referenzen in die nachfolgenden Vergleiche aufgenommen, ohne durch spezielle Parametrisierungen verändert zu werden.

4.1.4 Vergleich der Verfahren

Die nachfolgenden Tests gliedern sich in drei Abschnitte: Zunächst wird anhand von Tests mit dem Gestenset aus [217] eine Auswahl an Shape-Signaturen und lokalen Schrittmustern begründet. Das Gestenset ist für Nächste-Nachbar-Verfahren aufgrund nur wenig verrauschter Daten gut geeignet und soll daher zur Referenz einer Minimalanforderung dienen. Anschließend erfolgt ein Vergleich der Methoden dieses reduzierten Sets anhand vielversprechender Schrittmuster und gebräuchlicher Fensterfunktionen. Hierfür werden zwei schwieriger zu klassifizierende Gestensets - die frei verfügbare 'Intuidoc-Loustic Gestures DataBase' [164] und Single-Touch Gesten des vom Autor entwickelten Gestenalphabets [179] - herangezogen.

Begrenzung der Auswahl an Methoden

In einem ersten Vergleich werden die vorgestellten Shape-Signaturen unter verschiedenen Schrittmustern und sonst freiem DTW hinsichtlich ihrer Klassifikationsgenauigkeit gegenübergestellt. Das komplett beschränkende Schrittmuster, welches nur Warping-Pfade in der Diagonalen zulässt und auch als globale Beschränkung gesehen werden kann, wird ebenfalls einbezogen. Ebenso werden die als Spezialfälle zum Merkmal 9 anzusehenden Verfahren aus [119] (Protractor), [217] ($1) sowie die Prokrustes-Analyse betrachtet. Für den Vergleich werden Templates von fünf Testpersonen aus dem Set von [217] verwendet. Jeweils ein Template pro Gestenklasse wurde zufällig (auch unter verschiedenen Ausführungsgeschwindigkeiten) ausgewählt und dieses Template-Set gegen fünf abermals zufällig gewählte der verbleibenden Instanzen getestet. Diese Prozedur wurde jeweils fünf mal pro Set eines Nutzers wiederholt und die Ergebnisse über die Wiederholungen und die verschiedenen Nutzer gemittelt. Die vielversprechendsten Verfahren werden für weiterführende Untersuchungen ausgewählt.

Um den Einfluss verschiedener Schrittmuster zu untersuchen, wurde das 'dtw' Paket [66] der Statistik Software 'R' [156] verwendet. Das Paket unterstützt neben den gebräuchlichen Schrittmustern 'symmetric1', 'symmetric2', 'asymmetric' und dem restriktivsten Fall nur eines möglichen diagonalen Warping-Pfades 'rigid' noch acht Schrittmuster der Taxonomie nach Sakoe und Chiba [173], 14 nach Rabiner und Myers [140] sowie 56 nach Rabiner und Juang [157] und eines nach Mori et al. [137].

Werden die Versionen mit geglätteten Gewichten nach Rabiner und Juang (28), Sakoe und Chiba (3), Rabiner und Meyers (4) sowie Duplikate[29] außer Acht gelassen, verbleiben 30 verschiedene Schrittmuster.

In der vorliegenden Arbeit werden nur symmetrische Schrittmuster in Betracht gezogen, da nicht erwartet wird, dass sich Templates und eingegebene Gesten systematisch unterscheiden. Es wird zudem davon ausgegangen dass Gesten wenig rauschbehaftet sind, also beispielsweise bis auf am Anfang und am Ende kein Zittern während der Eingabe auftritt. Insofern wird diesbezüglich der Argumentation in [173] gefolgt wird, dass jeder Teil der Eingabe von gleicher Relevanz ist. Dennoch werden unstetige Schrittmuster einbezogen, um die Annahme zu prüfen. Konkret lassen die Schrittmuster der Typen Rabiner-Myers 'typeIIa', 'typeIIb', 'typeIId' (letzteres dabei normalisierbar) Unstetigkeiten im Warping-Pfad zu, beachten demnach also nicht zwingend jeden Punkt der Eingabe.

Die untersuchte Auswahl des aufgrund obiger Ausführungen eingeschränkten Sets verfügbarer Schrittmuster wird in Abbildung 4.3 gezeigt.

[29] Bei identischen Schrittmustern nach verschiedenen Taxonomien wurde jeweils jene Variante mit erster Nennung in der Literatur gewählt. Ein weiteres Duplikat im Paket ist durch die gleiche Implementierung der geglätteten Version des 'typeIbs' nach Rabiner und Myers gegenüber der ungeglätteten Variante 'type1b' enthalten.

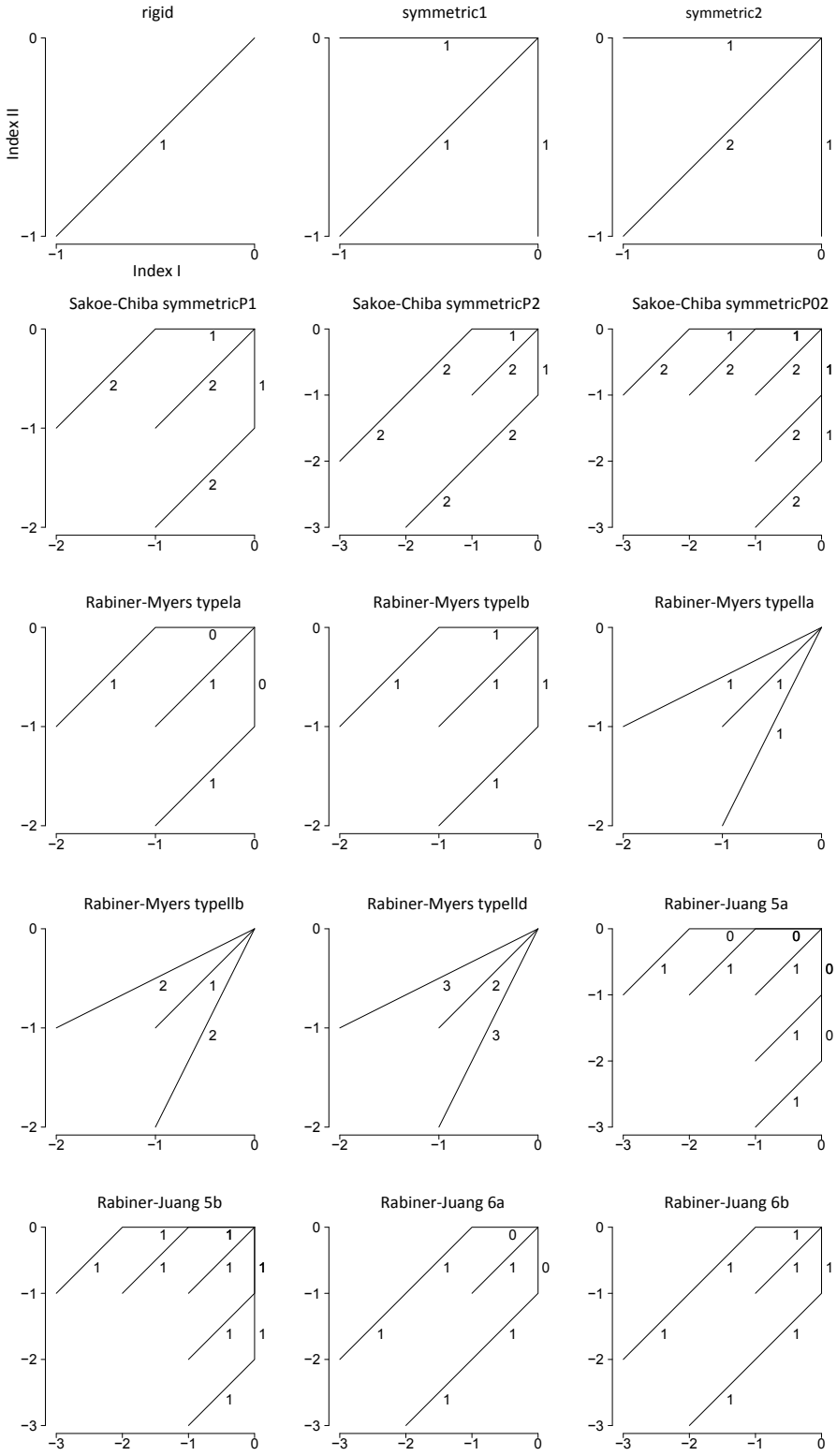


Abbildung 4.3: Eine Auswahl in der Literatur verfügbarer Schrittmuster, die für Tests herangezogen wurden. Aus den bekannten Schrittmustern wurden Duplikate unter verschiedenen Namen außer Acht gelassen. Die Auswahl beschränkt sich auf symmetrische Schrittmuster und auf Varianten ohne Glättung der Gewichte.

Die Ergebnisse des Vergleiches zunächst zwischen ‘rigid’ und ‘symmetric2’ sind in Abbildung 4.4 vorgestellt. Die Daten sollen dazu dienen, anhand zweier Extreme abzuschätzen, welche Shape-Signaturen sich für eine ausführlichere Betrachtung anbieten.

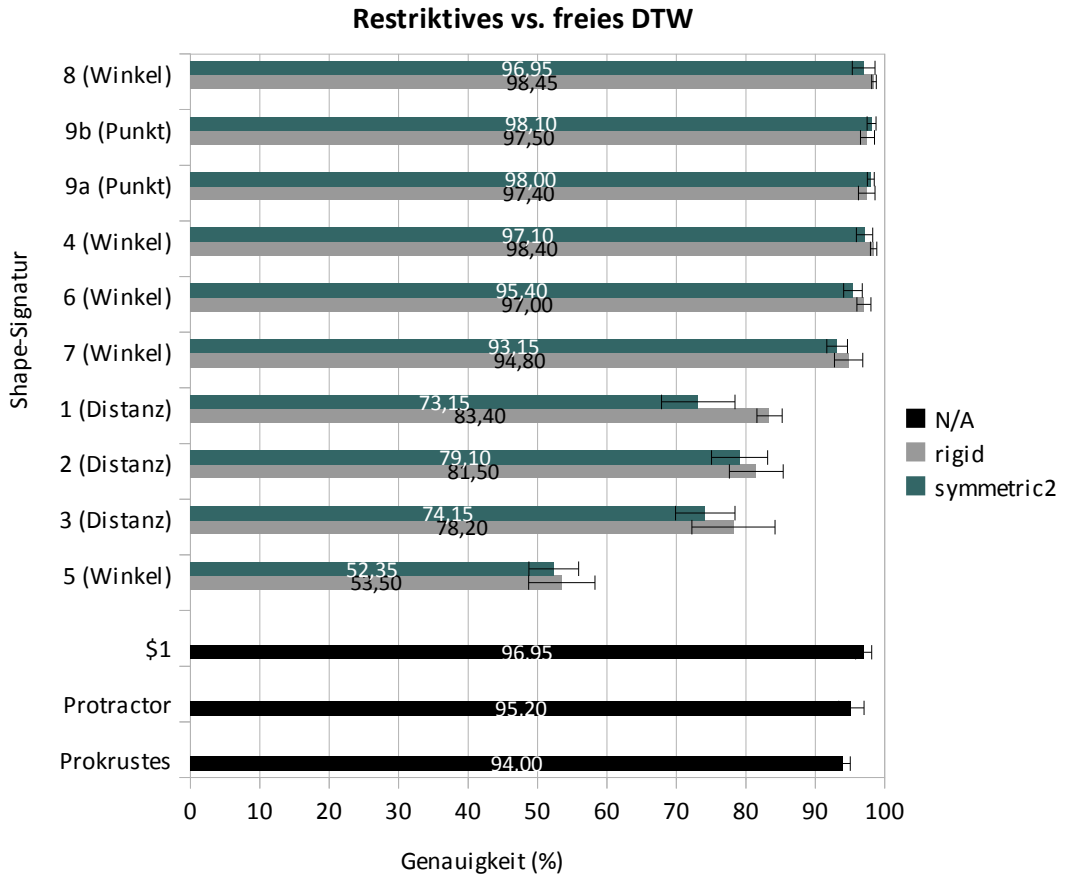


Abbildung 4.4: Vergleich der Nächste-Nachbar-Klassifikationsmethoden nach den vorgestellten Methoden und Shape-Signaturen und unter zwei verschiedenen Schrittmustern für das DTW.

Es ist zu erkennen, dass die Verfahren Protractor, $1 und Prokrustes zu denen von DTW unter Signaturen aus den Merkmalen 4, 6, 7, 8 und 9 vergleichbare aber insgesamt schlechtere Resultate erzielen. Dabei sind die Unterschiede zwischen beschränktem und unbeschränktem DTW gering und am ausgeprägtesten unter Merkmal 1. Dies geht mit der Mutmaßung aus [221] einher, dass sich Fensterfunktionen positiv auswirken. Allerdings treten Verbesserungen durch freies DTW nur unter Merkmal 9 (Positionen) auf, in allen anderen Fällen ist der direkte, punktweise Vergleich ohne Warping erfolgreicher. Die ebenfalls in [221] beobachtete Verschlechterung der Genauigkeit für zu strenge Spezialfälle bei diagonalem Warping-Pfad deutet sich allerdings nicht bei der Verwendung von Shape-Signaturen aus Distanzen und Winkeln an. Da die Signaturen aus Distanzen (neben Merkmal 5) schon bei dem einfachen Gestenset im Durchschnitt die schlechtesten Resultate liefern, werden diese Fälle im nachfolgenden, vollständigen Vergleich der Schrittmuster nicht mehr einbezogen. Ein χ^2-Test mit Yates-Korrektur brachte den Nachweis ($\chi^2 = 90,91$; df=1; $p \approx 0$), dass der Abstand zwischen dem in Abbildung 4.4

aufgeführten schlechtesten Ergebnis der weiterhin betrachteten Methoden (Signatur 7) und dem besten Ergebnis der nicht mehr in einen Vergleich einbezogenen (Signatur 1) Methoden signifikant ist.

Eine Gegenüberstellung der verbleibenden Shape-Signaturen sowie detaillierte Ergebnisse erreichter Klassifikationsgüten bei ausgewählten Schrittmustern werden in Tabelle 4.2 gezeigt. Der Spezialfall für 'rigid' wird in Durchschnittsberechnungen nicht einbezogen, jedoch nachfolgend zum Vergleich mit aufgeführt.

Tabelle 4.2: Ergebnisse der Klassifikation verschiedener Shape-Signaturen unter dem Einfluss gebräuchlicher Schrittmuster. Fett gedruckte Werte zeigen für jede Shape-Signatur das geeignetste Schrittmuster, also jenes, unter dem die höchste Genauigkeit erreicht wird, an. Eingerahmte Werte ordnen jedem Schrittmuster die vielversprechendste Shape-Signatur zu. Alle Werte sind Prozentangaben. Durchschnittswerte (⌀) und Standardabweichungen (s) über Zeilen und Spalten sind ebenfalls angegeben.

	9b	**8**	**4**	**9a**	**6**	**7**	⌀	s
rigid	97,50	98,45	98,40	97,40	97	94,80	97,26	1,22
symmetric 1	98,25	97,20	97,90	98,00	97,25	94,85	97,24	1,24
symmetric2	98,10	96,95	97,10	98,00	95,40	93,15	96,45	1,89
symmetricP1	98,30	98,55	***99,70***	98,20	97,70	97,55	98,33	0,77
symmetricP2	***98,40***	***99,50***	***99,70***	98,25	97,95	98,95	***98,79***	0,71
symmetricP05	98,20	98,30	99,65	98,00	97,35	95,70	97,87	1,30
typeIa	98,25	98,80	***99,70***	98,10	97,70	96,30	98,14	1,14
typeIb	98,25	98,60	***99,70***	98,15	97,85	98,35	98,48	0,64
typeIIa	97,85	98,40	***99,70***	97,45	97,85	99,15	98,40	0,87
typeIIb	98,25	98,70	***99,70***	98,00	97,80	99,20	98,61	0,74
typeIId	98,05	98,60	***99,70***	97,80	97,80	***99,25***	98,53	0,80
rabinerJuang5a	97,85	98,15	99,65	97,80	96,85	89,20	96,58	3,73
rabinerJuang5b	98,30	98,40	99,65	98,10	97,65	97,30	98,23	0,81
rabinerJuang6a	98,25	***99,50***	99,65	98,10	***98,00***	98,35	98,64	0,73
rabinerJuang6b	98,30	99,45	***99,70***	***98,30***	***98,00***	99,15	***98,82***	0,71
⌀	98,19	98,51	***99,37***	98,02	97,51	96,89		
s	0,17	0,75	0,81	0,22	0,69	2,90		

In Tabelle 4.2 zeigt sich, dass für die neue Shape-Signatur aus Merkmal 4 die Klassifikationsgenauigkeit, ausnehmend der Schrittmuster 'symmetric1' und 'symmetric2', relativ unbeeinflusst hoch und durchgängig sowie im Gesamtdurchschnitt über den verschiedenen Schrittmustern die beste ist. Die einfachen und häufig verwendeten Schrittmuster 'symmetric1' und 'symmetric2' erzielen nur für Distanzen zwischen Positionen (9a und 9b) gute Ergebnisse und auch bei diesen sind andere Muster die bessere Wahl[30]. Die Schrittmuster 'symmetricP2' und 'rabinerJuang6b' weisen, mit einer Ausnahme für 'typeIId' bei Signatur 7, die besten Ergebnisse über die verschiedenen Signaturen auf. Signatur 7 hat dabei allerdings auch die größte Varianz der Klassifikationsgenauigkeit über die Schrittmuster. Tabelle 4.3 untersucht die Signifikanz (einfaktorielle ANOVA bei $\alpha = 0,05$) der Einflüsse unterschiedlicher Schrittmuster.

[30] Die nicht normalisierbare und unstetige Variante des 'symmetric1'-Musters, 'typeIa', zeigte durchgängig unter jeder Signatur bessere Ergebnisse.

Tabelle 4.3: Zusammenfassung des Einflusses der Schrittmuster für verschiedene Shape-Signaturen. Signifikante Einflüsse sind durch Hervorhebung der p-Werte angegeben. Weiterhin sind für jede Signatur die im Durchschnitt besten und schlechtesten erzielten Klassifikationsgenauigkeiten aufgeführt.

Signatur	F	p	Min (%)	Max (%)
4	18,12	$\ll \mathbf{0.01}$	97,10	99,70
8	3,68	$\ll \mathbf{0.01}$	96,95	99,50
7	48,86	$\ll \mathbf{0.01}$	89,20	99,25
9b	0,35	0,98	97,85	98,40
9a	0,54	0,89	97,45	98,30
6	2,40	**0,01**	95,40	98,00

Demnach beeinflussen Schrittmuster bei den klassischen Shape-Signaturen (euklidische oder Manhattan-Distanz zwischen Punkten) die Genauigkeit nicht signifikant. Eine tendenzielle Beeinflussung liegt ebenso wenig vor. Erwartungsgemäß wird die Shape-Signatur nach Merkmal 4 ebenso wenig beeinflusst ($F = 0,066; p \approx 1$), wenn die Schrittmuster 'symmetric1' und 'symmetric2' nicht mit einbezogen werden.

Anhand der obigen Ergebnisse wird eine Einschränkung der weiter zu untersuchenden Methoden getroffen. Da das Interesse bei robuster Klassifikation liegt, werden Verfahren, die anhand des einfachen Gestensets schon schlecht abschneiden oder dabei von anderen Verfahren konsistent überboten werden, als ungeeignet angesehen und nicht weiter untersucht.

Bei den Shape-Signaturen erzielte 9a konsistent schlechtere Ergebnisse gegenüber der Version mit euklidischer Distanz (9b) und wird daher nachfolgend nicht weiter betrachtet. In die engere Auswahl der Schrittmuster werden jene gefasst, welche bei einer festgelegten Signatur die besten Ergebnisse lieferten: 'rabinerJuang6a', 'rabinerJuang6b', 'symmetricP2', 'typeIId'. Ausgeschlossen werden weiterhin Schrittmuster, welche von anderen subsumiert werden, das heißt konsistent schlechtere Ergebnisse für jede Signatur liefern. So ist 'rabinerJuang5b' konsistent besser als die typgleichen (unter anderen Gewichten) Muster 'symmetricP05' und 'rabinerJuang5a'. Die typgleichen Muster 'symmetricP1', 'typeIa' und 'typeIb' lieferten konsistent schlechtere Ergebnisse als 'symmetricP2'; 'typeIb' wird aber als bester Vertreter dieses Typs aufgenommen. Bei den ebenfalls typgleichen Mustern 'typeIIa', 'typeIIb', 'typeIId' wurde letztere, normalisierbare Variante bereits aufgenommen und aufgrund der Ähnlichkeit in den Ergebnissen wird nur diese Variante beibehalten. Die Muster 'symmetric1' und 'symmetric2' werden aufgrund ihrer schlechten Ergebnisse (alle anderen aufgenommenen Muster, außer 'typeIId', sind konsistent besser) nicht weiter untersucht.

Im Nachfolgenden werden die vielversprechendsten fünf Shape-Signaturen, in Kombination mit sechs ausgewählten Mustern, sowie die Version des diagonalen Warping-Pfades 'rigid' und drei globalen Fensterfunktionen in abschließenden Tests untersucht.

Lokale und Globale Beschränkungen

Die in den finalen Tests verwendeten Schrittmuster sind in Abbildung 4.5 noch einmal zusammengestellt.

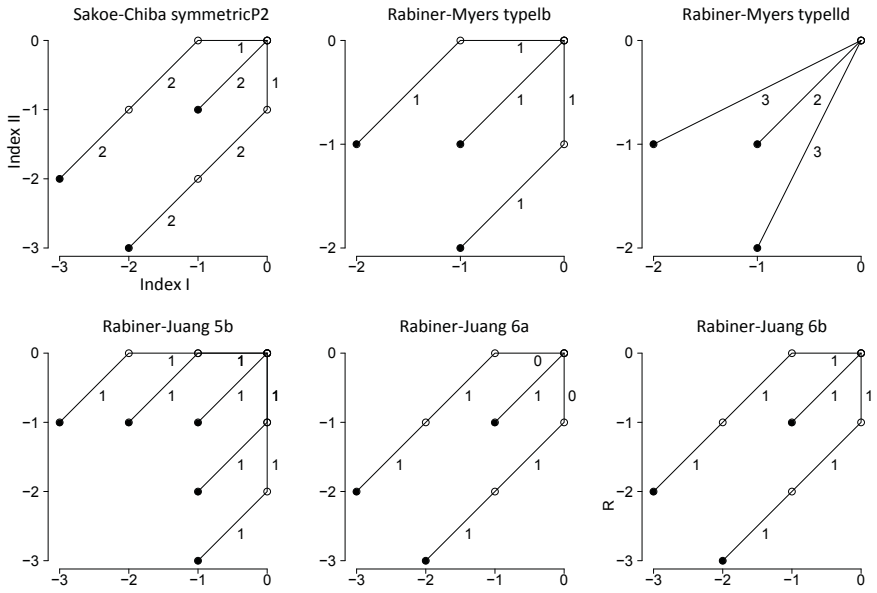


Abbildung 4.5: Das reduzierte Set aus Schrittmustern, welche für weiterführende Tests herangezogen wurden.

Globale Nebenbedingungen werden üblicherweise durch a-priori Wissen definiert und beschränken den Grad der erlaubten Korrekturen von Verzerrungen durch Begrenzung des Warping-Pfades [65]. Bei den globalen Fensterfunktionen sind drei Typen in der Literatur gebräuchlich. In [173] wird der Abstand zwischen zugeordneten Index-Paaren durch einen festen Parameter begrenzt. Dieses Fenster verhindert, dass zu lange Abschnitte zu stark entzerrt werden und entspricht bei gleich langen Zeitreihen einer anderen gebräuchlichen Fensterfunktion: dem 'Slanted Band' (siehe [65]). Das 'Slanted Band' wiederum ist im Gegensatz dazu auch definiert, wenn sich die Längen der Zeitreihen stark unterscheiden. Eine weitere Fensterfunktion, das sogenannte 'Itakura-Parallelogramm' [85], lässt stärkere Entzerrungen im mittleren Bereich der Zeitreihen zu, während die Randbedingungen an den Enden strenger sind. Die laut [94] zudem am häufigsten verwendeten Fensterfunktionen nach [85] und [173] sind in Abbildung 4.6 dargestellt.

Das 'Itakura-Parallelogramm' wird nach Angaben in [65] durch lokale Einschränkungen wie dem 'Rabiner-Juang'-Schrittmuster vom Typ IV erzeugt, welches ebenfalls dem 'Rabiner-Myers'-Muster vom typeIIIc (Abbildung 4.6 rechts) entspricht. Allerdings wird es hier im Gegensatz zu dieser Argumentation als globale Fensterfunktion aufgefasst, innerhalb dessen andere Schrittmuster Verwendung finden können.

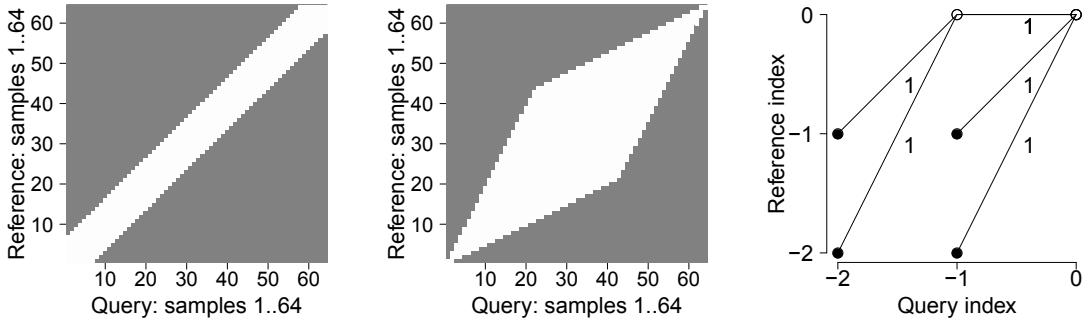


Abbildung 4.6: Links ist die Fensterfunktionen 'Sakoe-Chiba-Band' [173] mit einer Fensterbreite von 6 dargestellt. In der Mitte wird ein 'Itakura-Parallelogramm' [85] als globales Fenster gezeigt. Rechts ist ein asymmetrisches lokales Schrittmuster aufgeführt, mit dem alternativ ebenfalls ein 'Itakura-Parallelogramm' erzeugt werden kann.

Die untersuchten Fensterfunktionen neben dem Fall ohne globale Beschränkungen sind das 'Sakoe-Chiba-Band' und das 'Itakura-Parallelogramm'. Nach Xi et al. [221] und [76] liegt die optimale Breite des 'Sakoe-Chiba'-Fensters bei etwa 10% der Länge der Zeitreihen. Mangels näherer Informationen zur Ableitung der Fenstergröße aus den Daten wird dieser Wert ($\lfloor (N \cdot 10\%) \rfloor = 6$) auch hier verwendet.

Da das bisherige Gestenset genutzt wurde, um Nebenbedingungen einzuschränken und eine Vorauswahl an Methoden zu treffen, werden, um nach [205] eine Überanpassung (parameter bias) zu vermeiden, die abschließenden Tests anhand zweier weiterer Single-Stroke Datensets durchgeführt. Der Argumentation in [95] folgend ist eines davon - die 'Intuidoc-Loustic Gestures DataBase' (ILGDB) [164] - zudem ein frei verfügbares und bereits in Veröffentlichungen verwendetes Gestenset, was Vergleichbarkeit erlaubt und Verzerrungen durch die Auswahl der Daten (data bias) vermeidet. In Abbildung 4.7 wird dieses Gestenset dargestellt.

Abbildung 4.7: Illustrationen für die Gesten des ILGDB-Sets aus [164].

Das ILGDB-Set wurde mit Hilfe dreier Nutzergruppen angelegt. Zwei Gruppen definierten dabei jeweils komplett oder partiell eigene Gesten für vorgegebene Interaktionen. Da die Spezifikationen somit heterogen sind, ist mit diesen Sets kein nutzerübergreifender Vergleich möglich. Eine feste Auswahl von 21 dieser Gesten war der Gruppe 3

vorgegeben, die dazu Templates ablegte. Jede Gruppe durchlief fünf Phasen, wobei in vier Phasen jeweils mittels Fragebogen oder anhand von aufzurufenden Interaktionen die Gesten abgefragt wurden. In einer Initialisierungsphase waren zuvor drei Templates pro Gestenklasse zu spezifizieren. Das aus dieser Phase verfügbare[31] Set von Gruppe 3 (elf Nutzer) wird im Folgenden verwendet.

Ausgewählte Beispielinstanzen aus dem ILGDB-Gestenset der Gruppe 3 in der Initialisierungsphase finden sich in Abbildung 4.8.

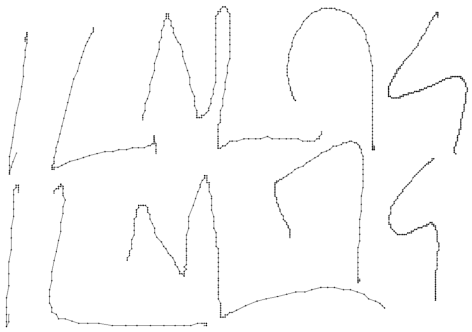


Abbildung 4.8: Konkrete, von Nutzern eingegebene Instanzen (zu Darstellungszwecken gleichförmig skaliert) einer kleinen Auswahl an Gesten des ILGDB Gestensets.

Die verwendeten Gesten aus der ILG-Datenbank zeichnen sich durch unsichere Eingaben zu Beginn und Ende einer Geste aus (siehe Abbildung 4.8). Klassifizierer, denen die Verwendung globaler Merkmale zugrunde liegt, eignen sich besser für derartig verrauschte Daten. Klassifikatoren, die auf Vergleichen von Shape-Signaturen basieren, sind dafür weniger geeignet, da sie lokale Informationen stärker berücksichtigen. Es wird an dieser Stelle dennoch keine Vorverarbeitung der Gesten vorgenommen, bei denen Anfang und Ende geglättet werden. In [164] wurden bei Trennung nach Nutzern für die nutzerabhängigen Gestensets (Templates der Initialisierungsphase, Testgesten der vier anderen Phasen) der Gruppe 3 mit dem HBF49 Set [44] aus 49 Merkmalen mittels eines Nächste-Nachbar-Klassifizierers und einem SVM-Ansatz 90,57% respektive 90,8% Klassifikationsgenauigkeit ermittelt. Gleichzeitig war die Genauigkeit für dieses Set aus vorgegebenen Gesten schlechter als für das der Gruppen 1 und 2 mit jeweils etwa 93,5%.

Das zweite verwendete Set besteht aus Buchstaben des im Kapitel 6.2 entwickelten Gestenalphabetes (MTIS, vorgestellt in [179]). Für dieses Set wurden die von Nutzern in den Tests im Kapitel 6.2 ohne ihr Wissen spezifizierten Gesten verwendet. Sie wurden in einem Trainings-Durchlauf angelegt, bei dem nur ein symbolischer Hinweis die Ausführung vorgab. Für jeden Nutzer sowohl aus der Single-Touch als auch der Multi-Touch Gruppe wurden jeweils beide der pro Klasse zu spezifizierenden Buchstaben hinzugefügt. Es wurde dabei die in Abbildung 4.9 gezeigte Auswahl von Single-Touch Symbolen gewählt. Ausgewählte Beispielinstanzen der verwendeten Gesten des MTIS-Sets sind in Abbildung 4.10 dargestellt.

[31] `www.irisa.fr/intuidoc/ILGDB.html`

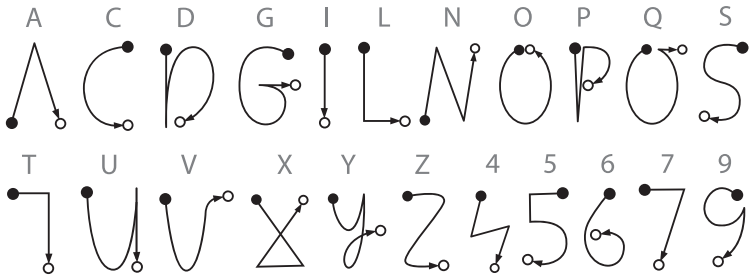


Abbildung 4.9: Illustrationen der ausgewählten Single-Touch Gesten aus dem Gestenalphabet, welches im 'Multi-Touch Text Input System' (siehe Kapitel 6.2) Verwendung findet.

Die in Abbildung 4.10 beispielhaft aus den 22 Klassen gewählten Instanzen der Gesten aus dem MTIS-Set weisen keine so starken Krümmungen am Anfang oder am Ende einer Eingabe auf. Allerdings liegt - wie ebenfalls im Set der ILGDB - eine ausgeprägte Variabilität in den Eingaben der Nutzer vor. Die eingegebenen Gesten mussten während der Akquirierung erfolgreich gegen vordefinierte Templates klassifiziert werden, so dass zumindest eine Plausibilitätsprüfung stattfand und zu große Abweichungen auch im nutzerunabhängigen Set nicht vorkommen sollten.

Abbildung 4.10: Zu Darstellungszwecken gleichförmig skalierte Instanzen einer Auswahl nutzerspezifizierter Gesten aus dem MTIS-Gestenset.

Die nachfolgenden Tests wurden jeweils für jedes Verfahren (wobei jede Parameterwahl als getrennte Methode betrachtet wird) unter den gleichen fünf Testläufen mit denselben Testsets durchgeführt. Insgesamt wurden demnach zehn Testläufe (jeweils fünf im ILGDB- und im MTIS-Set) über die zufällige und nutzerübergreifende Auswahl von Templates und Testinstanzen generiert.

Für die Untersuchungen anhand des MTIS-Sets wurden für jeden dieser Testläufe aus den spezifizierten Gesten fünf zufällige Templates und 15 Testinstanzen pro Klasse gewählt. Die Testläufe anhand des ILGDB-Sets entstanden gleichermaßen, mit dem Unterschied, dass zwar ebenfalls nutzerübergreifend fünf Templates, aber aufgrund der geringeren Set-Größe nur 10 Testinstanzen pro Klasse gewählt wurden.

Im nächsten Abschnitt werden die Ergebnisse der unter den obigen Settings anhand der vorgestellten Gestensets durchgeführten Tests beschrieben. Zunächst werden die Ergebnisse bezüglich des ILGDB-Sets, nachfolgend die des MTIS-Sets, vorgestellt.

Testresultate ILGDB

Durchschnittliche Genauigkeiten der Tests anhand des ILGDB-Sets in Abhängigkeit von Shape-Signatur, Schrittmuster und dem verwendeten globalen Fenster sind in Tabelle 4.4 gegeben.

Tabelle 4.4: Ergebnisse für das ILGDB-Set in Abhängigkeit der gewählten Shape-Signatur und der lokalen (Schrittmuster) als auch globalen (Fensterfunktion) Parameter. Alle Werte sind Prozentangaben. Fett gedruckte Werte markieren Ausnahmen, in denen die Wahl des 'Sakoe-Chiba'-Fensters bei entsprechend gleichen Shape-Signaturen und lokalen Beschränkungen **nicht** die beste Wahl darstellte.

Fensterfunktion	Shape-Signatur	symmetricP2	rabinerJuang5b	rabinerJuang6a	rabinerJuang6b	typelb	typelld	⊘
N/A	9b	78,00	78,57	77,71	78,29	78,57	78,48	78,27
	8	92,76	92,10	92,48	92,67	92,19	92,57	92,46
	4	**79,43**	77,71	**79,24**	79,81	79,24	78,86	79,05
	6	78,19	75,62	**78,48**	**79,52**	77,43	76,57	77,63
	7	73,52	72,57	73,43	74,38	73,43	75,62	73,83
Itakura	9b	78,00	78,57	77,71	78,29	78,57	78,48	78,27
	8	92,76	92,00	92,48	92,67	92,19	92,57	92,44
	4	**79,43**	77,90	**79,24**	79,81	79,24	78,86	79,08
	6	78,19	75,81	**78,48**	**79,52**	77,43	76,57	77,67
	7	73,52	72,86	73,43	74,38	73,43	75,62	73,87
Sakoe-Chiba	9b	78,00	78,57	77,71	78,29	78,57	78,48	78,27
	8	93,33	93,14	93,14	93,24	93,05	93,05	93,16
	4	**79,33**	79,71	**79,14**	79,90	79,81	80,29	79,70
	6	78,29	76,48	**78,38**	**79,33**	77,62	76,57	77,78
	7	74,00	73,43	73,43	74,67	74,67	75,81	74,33
	⊘	80,45	79,67	80,3	80,98	80,36	80,56	80,39

Die Ergebnisse des Tests für das ILGDB-Set in Tabelle 4.4 zeigen, dass die Wahl des 'Sakoe-Chiba'-Fensters (mit Breite 10%) gegenüber der des 'Itakura-Parallelogramms' oder freiem DTW bis auf wenige Ausnahmen höhere Genauigkeiten zur Folge hat. Dieser Aspekt wird in Tabelle 4.5 anhand durchschnittlicher Genauigkeiten für jedes Schrittmuster unter den verschiedenen Fenstern noch einmal verdeutlicht.

Tabelle 4.5: Über alle Shape-Signaturen gemittelte Werte für jedes Schrittmuster unter jeweils einer spezifizierten Fensterfunktion. Maximale Werte (in Prozent) jeder Zeile sind hervorgehoben.

	N/A	Itakura	Sakoe-Chiba
symmetricP2	80,38	80,38	**80,59**
rabinerJuang5b	79,31	79,43	**80,27**
rabinerJuang6a	80,27	80,27	**80,36**
rabinerJuang6b	80,93	80,93	**81,09**
typelb	80,17	80,17	**80,74**
typelld	80,42	80,42	**80,84**
⊘	80,25	80,27	80,65

Unabhängig von der Wahl eines Schrittmusters sind die Ergebnisse unter dem 'Sakoe-Chiba'-Fenster - trotz nur geringer Unterschiede - im Durchschnitt am besten. Die Annahme, dass eine größere Beschränkung eine höhere Genauigkeit bewirkt, kann demnach bestätigt werden. Eine weitere Betrachtung der Ergebnisse für die beiden verbleibenden Fensterfunktionen ist nicht zweckmäßig. In Tabelle 4.6 sind die Ergebnisse unter dem 'Sakoe-Chiba'-Fenster daher noch einmal denen unter dem Schrittmuster 'rigid' gegenübergestellt.

Tabelle 4.6: Für das ILGDB-Set sind die Klassifikationsergebnisse unter der Fensterfunktion von 'Sakoe-Chiba' in Abhängigkeit der verwendeten Shape-Signaturen und der untersuchten Schrittmuster den Resultaten unter dem Schrittmuster 'rigid' gegenübergestellt. Die Werte in Prozentangaben sind zeilen- und spaltenweise jeweils nach ihrem Durchschnitt (ohne Einbezug der Werte für 'rigid') absteigend sortiert. Eingerahmte Werte zeigen die unter einer Shape-Signatur beste Wahl eines Schrittmusters an, fettgedruckte hingegen markieren die beste Zuordnung einer Shape-Signatur zu einem Schrittmuster.

Shape-Signatur	rabinerJuang6b	typeId	typeIb	symmetricP2	rabinerJuang6a	rabinerJuang5b	⊘	rigid
8	***93,24***	***93,05***	***93,05***	***93,33***	***93,14***	***93,14***	***93,16***	***90,19***
4	79,90	80,29	79,81	79,33	79,14	79,71	79,70	76,29
9b	78,29	78,48	78,57	78,00	77,71	78,57	78,27	74,86
6	79,33	76,57	77,62	78,29	78,38	76,48	77,78	75,24
7	74,67	75,81	74,67	74,00	73,43	73,43	74,34	68,1
⊘	81,09	80,84	80,74	80,59	80,36	80,27	80,65	76,93

Auch hinsichtlich der strengsten Beschränkung ('rigid') - entsprechend eines 'Sakoe-Chiba'-Fensters mit Breite 0 - unter der keine zeitliche Entzerrung stattfindet, bestätigt sich die in der Literatur vorgefundene Beobachtung, dass eine solche Wahl wiederum zu einer Verschlechterung führt. Anhand der in Tabelle 4.6 dargestellten Resultate kann abgelesen werden, dass unter keiner Shape-Signatur die Verringerung der Breite des globalen Fensters auf 0 einen Zugewinn verspricht. Bei den Shape-Signaturen selbst fällt auf, dass absolute Winkel (8) und Winkel zwischen Startpunkt, Trajektorien-Punkt und dessen Nachfolger (4) abermals und konsistent die besten Genauigkeiten - wenn auch in umgekehrter Platzierung - liefern. Signatur 8 ist dabei in diesem Fall die eindeutig beste Wahl. Für die anderen drei untersuchten Signaturen zeigte sich die gleiche Reihenfolge wie beim ersten Test am $1-Gestenset.

Hinsichtlich der für das ILGDB-Set am besten geeigneten Schrittmuster kann keine derart eindeutige Aussage getroffen werden (siehe eingerahmte Werte in Tabelle 4.6). Das Muster 'rabinerJuang6b' erzielt allerdings abermals durchschnittlich die höchsten Genauigkeiten, obwohl das insgesamt beste Ergebnis unter dem 'symmetricP2'-Muster erzeugt wird. Die ähnlich hohen Ergebnisse für 'typeIId' zeigen auf, dass unstetige Schrittmuster für verrauschte Daten besser geeignet sind als für idealisierte Gesten.

Bisher wurden die Methoden Prokrustes, Protractor und $1 nicht betrachtet. Sie

können - wie die Anwendung des Schrittmusters 'rigid' - ebenfalls als strenger Spezialfall des DTW gesehen werden, bei dem nur ein diagonaler Warping-Pfad zugelassen ist und nur die Shape-Signatur das Verfahren spezifiziert. Die durchschnittlichen Klassifikationsgenauigkeiten dieser Verfahren sind in Tabelle 4.7 aufgeführt. Zum Vergleich werden die Ergebnisse der unter der 'rigid'-Beschränkung getesteten Verfahren ebenfalls gelistet.

Tabelle 4.7: Durchschnittliche Resultate hinsichtlich der Klassifikation anhand des ILGDB-Sets für die Methoden Prokrustes, Protractor, $1 und der unter 'rigid'-Beschränkung getesteten Shape-Signaturen. Die Methoden innerhalb jeweils einer der beiden Gruppen wurden absteigend bezüglich der erzielten Genauigkeiten sortiert.

Shape-Signatur	Genauigkeit (%)
Protractor	83,81
$1	74,29
Prokrustes	72,48
8	90,19
4	76,29
6	75,24
9b	74,86
7	68,10

In Tabelle 4.7 kann abgelesen werden, dass das auf Kosinus-Distanzen basierende Verfahren Protractor vergleichsweise gute Resultate erzielt und etwa zwischen denen der Signaturen 8 und 4 angesiedelt ist. Die Reihenfolge der Verfahren Prokrustes und $1 entspricht der im ersten Test; das $1-Verfahren schneidet auch hier besser ab. Es bestätigt sich allerdings abermals, dass das Verfahren durch die analytische Vorverarbeitung des Prokrustes-Verfahrens verbessert werden kann. Diese Methode entspricht der Shape-Signatur 9b, die auch als Prokrustes-Analyse unter lokaler euklidischer Distanz - anstatt quadrierter euklidischer Distanz - gesehen werden kann.

Den Abschluss der Betrachtung des Tests bildet eine Gegenüberstellung der Verfahren, jeweils unter der für sie geeignetsten Parameterwahl. In Tabelle 4.8 ist daher zunächst die Zuordnung der Parameter zu den Methoden angegeben, mit denen für das ILGDB-Set die besten Resultate erreicht werden konnten.

Tabelle 4.8: Zu jeder Shape-Signatur ist die für das ILGDB-Set beobachtete, geeignetste Wahl der lokalen und globalen Parameter für die Klassifikation per DTW angegeben.

Shape-Signatur	Schrittmuster	Fenster
8	symmetricP2	Sakoe-Chiba
4	typeId	Sakoe-Chiba
6	rabinerJuang6b	frei, Itakura
9b	rabinerJuang5b, typeIb	frei, Itakura, Sakoe-Chiba
7	typeId	Sakoe-Chiba

Die den in Tabelle 4.8 gegebenen Spezifikationen zugeordneten Testergebnisse sind für eine bessere Anschaulichkeit in Abbildung 4.11 dargestellt.

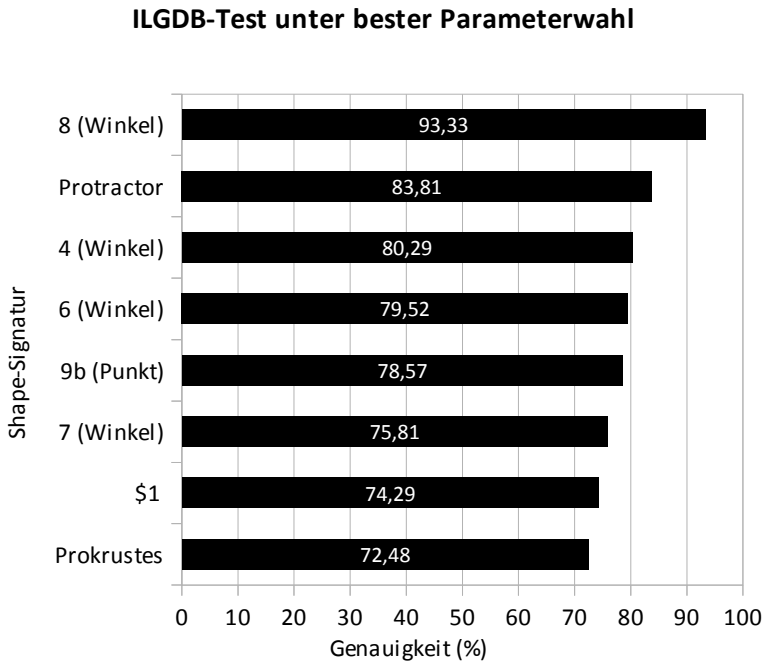


Abbildung 4.11: Unter geeignetsten Parametern erzielte Ergebnisse für das ILGDB-Set.

Testresultate MTIS

Die durchschnittlichen Genauigkeiten der Klassifikation unter dem MTIS-Set sind in Tabelle 4.9 gegeben. Die Angaben sind abermals der gewählten Shape-Signatur, dem lokalen Schrittmuster und dem verwendeten globalen Fenster zugeordnet. Bei Betrachtung der Resultate erweist sich auch für das MTIS-Set das 'Sakoe-Chiba'-Fenster als die beste Wahl. Die jeweils höchsten Ergebnisse bei Festlegung einer Shape-Signatur (fett gedruckte Werte) oder einem lokalen Schrittmuster (eingerahmte Werte) finden sich immer unter dieser globalen Beschränkung. Die Klassifikation mit Shape-Signatur 8 ist erneut die beste Alternative mit der sowohl im Durchschnitt als auch häufigsten besten Genauigkeit. Die insgesamt höchste Klassifikationsrate wird ebenfalls unter dieser Signatur und dem Schrittmuster 'rabinerJuang6b' erzielt. Das Schrittmuster 'rabinerJuang6b' stellt dabei, bis auf zwei Ausnahmen, generell die geeignetste lokale Beschränkung dar. Unter Signatur 4 wurden die zweithöchsten Klassifikationsraten erzielt. Die zudem im Vergleich zu 9b durchschnittlich besseren Ergebnisse auch von Signatur 6 (und dem iterativ berechenbaren Pendant 7) bestätigen weiterhin die gute Eignung von Winkel-Merkmalen zur Klassifikation von Gesten.

Tabelle 4.9: Resultate (in %) hinsichtlich Klassifikationstests mit dem MTIS-Set in Abhängigkeit der gewählten Shape-Signatur und der lokalen (Schrittmuster) als auch globalen (Fensterfunktion) Parameter. Eingerahmte Werte markieren die für eine Shape-Signatur beste Wahl eines Schrittmusters, fett gedruckte Werte hingegen weisen einem Schrittmuster die Shape-Signatur mit der höchsten erzielten Genauigkeit zu.

Fensterfunktion	Shape-Signatur	symmetricP2	rabinerJuang5b	rabinerJuang6a	rabinerJuang6b	typeIb	typeId	⊘
N/A	**9b**	94,18	94,06	93,88	94,18	94,06	93,94	94,05
	8	96,06	93,15	95,76	96,24	94,42	94,00	94,94
	4	95,76	94,61	95,52	95,88	95,33	94,97	95,34
	6	94,97	93,03	94,48	95,09	94,24	94,36	94,36
	7	93,94	90,97	93,70	94,12	92,18	93,33	93,04
Itakura	**9b**	94,18	94,06	93,88	94,18	94,06	93,94	94,05
	8	96,06	93,33	95,76	96,24	94,42	94,00	94,97
	4	95,76	94,97	95,52	95,88	95,33	94,97	95,40
	6	94,97	93,52	94,48	95,09	94,24	94,36	94,44
	7	93,94	90,97	93,70	94,12	92,18	93,33	93,04
Sakoe-Chiba	**9b**	94,18	94,06	93,88	94,18	94,06	93,94	94,05
	8	***96,55***	95,27	***96,42***	***96,79***	95,82	***95,94***	***96,13***
	4	95,76	***96,12***	95,58	95,94	***96,36***	95,70	95,91
	6	94,97	94,42	94,55	95,09	94,85	94,73	94,77
	7	94,67	94,18	94,36	94,73	94,30	95,03	94,55
	⊘	95,06	93,78	94,76	95,18	94,39	94,44	94,60

Die Ergebnisse der Verfahren unter diagonalem Warping-Pfad werden zum Vergleich in Tabelle 4.10 präsentiert.

Tabelle 4.10: Anhand von Tests mit dem MTIS-Set erreichte Klassifikationsgenauigkeiten für die Methoden Prokrustes, Protractor, $1 und der Shape-Signaturen unter 'rigid'-Beschränkung.

Shape-Signatur	Genauigkeit (%)
Prokrustes	92,00
$1	90,12
Protractor	84,30
8	96,79
4	95,09
6	93,52
9b	92,97
7	92,30

Die 'rigid'-Verfahren zeigen eine ähnliche Ordnung wie die entsprechenden DTW-Varianten. Signatur 8, gefolgt von Signatur 4, weisen die höchsten Klassifikationsraten auf. Die Verfahren Prokrustes, Protractor und $1 sind im dritten Test die schlechtesten 'rigid'-Methoden. Interessanterweise liegt dabei im Gegensatz zu den vorherigen beiden Tests die Prokrustes-Analyse vor dem $1-Klassifizierer und Protractor weist die niedrigs-

te Genauigkeit auf. Ein weiterer bemerkenswerter Aspekt ist, dass die Shape-Signatur 8 bei Verwendung des 'rigid'-Musters wie auch mit dem 'rabinerJuang6b'-Muster und der Shakoe-Chiba-Begrenzung die unter allen Verfahren besten Resultate erzielt.

Eine Gegenüberstellung aller Verfahren und der für sie geeignetsten Parameterwahl bei Klassifikation des MTIS-Sets ist in Tabelle 4.11 gegeben.

Tabelle 4.11: In Tests mit dem MTIS-Set nachgewiesene beste Parametrisierungen für die Klassifikation mittels DTW unter gegebenen Shape-Signaturen. Alle Angaben beziehen sich auf die Verwendung des 'Sakoe-Chiba'-Fensters (im Fall 'rigid' mit der Fensterbreite 0).

Shape-Signatur	Schrittmuster
9b	symmetricP2, rabinerJuang6b
8	rigid, rabinerJuang6b
4	typelb
6	rabinerJuang6b
7	typelId

In Abbildung 4.12 werden abschließend abermals alle Ergebnisse unter dieser bestmöglichen Parameterwahl illustriert. Auffallend ist, dass die Resultate der Methoden zwar näher beieinander liegen, die wesentlichen bisherigen Beobachtungen im Vergleich der Verfahren dennoch konsistent sind.

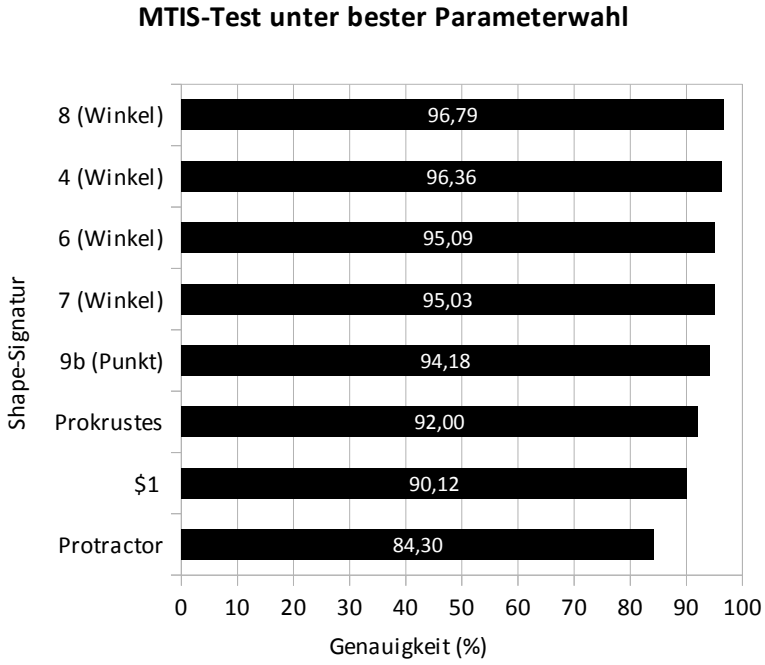


Abbildung 4.12: Entsprechend der besten Parameterwahl erzielte Resultate bei Klassifikation des MTIS-Sets.

4.1.5 Zusammenfassung und Diskussion

Anhand der drei Gestensets mit verschiedenen Graden von idealisierten Gesten abweichenden Spezifikationen wurde ein breites Spektrum an DTW-Variationen getestet. Obwohl Heterogenitäten in den Testergebnissen auftreten, lassen sich konsistente Schlüsse ziehen. Insgesamt sind Winkelmaße besser als andere Merkmale geeignet, die Form einer Single-Touch Eingabe zu repräsentieren. Dieser Effekt scheint sich zu verstärken, je größer die Varianz in den Daten in Form der Abweichung von idealisierteren Gesten ist. In den entsprechenden beiden Tests konnten unter der Shape-Signatur 8 (absolute Winkel) die besten Resultate erzielt werden. Diese Signatur zeigte sich außerdem am robustesten gegenüber unpräzisen Eingaben am Anfang und Ende einer Geste im ILGDB-Set. Die genauso effizient berechenbare Signatur aus dem neuem Merkmal 4 wies ebenso durchgängig gute und im Fall des $1-Gestensets sogar bessere Ergebnisse auf. Im Gegensatz zu den absoluten Winkeln liegt ein weiterer Vorteil darin, dass die erwünschten Invarianzen gegenüber Variationen in der Eingabe aufgrund ihrer Inhärenz elegant erlangt werden. In der Vorverarbeitung werden neben dem (bei Anwendung von DTW optionalen) Resampling keine weiteren Schritte benötigt. Ebenso ist die Berechnung der Signatur iterativ über den Fortschritt der Geste und damit auch für partielle Eingaben möglich, während für Signatur 8 zur Bestimmung der benötigten Rotation und der Skalierung die komplette Eingabe bekannt sein muss. Weiterhin zeigte sich die Shape-Signatur 4 für die Gestensets MTIS und $1 robust gegenüber den verwendeten Schrittmustern. Beide Verfahren sind den gängigen Methoden Prokrustes, Protractor und $1-Klassifizierer, welche auf jegliche zeitliche Entzerrung verzichten, überlegen. Unter Vorzug von Shape-Signatur 8 sind demnach beide Varianten auch in ihren Umsetzungen mit linearer Komplexität die bessere Wahl und auch aus Performance-Gründen besteht keine Notwendigkeit, auf die höhere Klassifikationsgenauigkeit zu verzichten.

Wie für jede der anderen Signaturen konnten allerdings durchgängig Verbesserungen unter Anwendung der DTW-Methoden erreicht werden. Dabei erwies sich in allen Fällen bis auf einen (im ILGDB-Test) eine nicht zu strenge Fensterfunktion (das 'Sakoe-Chiba'-Fenster mit 10% Breite) um den Warping-Pfad als die beste Option.

Bei den Schrittmustern konnte nachgewiesen werden, dass übliche, einfache Varianten wie 'symmetric1' oder 'symmetric2' wenig geeignet sind. Sie verbessern nur Methoden, die auf lokalen Distanzen zwischen Punkten basieren und auch für diese Fälle gibt es bessere Optionen. In allen drei Tests wurden die besten Ergebnisse mit dem Muster 'rabinerJuang6b' erzielt. Das Muster 'symmetricP2' brachte ebenfalls sehr gute Ergebnisse und kann als Alternative empfohlen werden, wenn das Kriterium der Normalisierbarkeit wichtig ist. Für verrauschte Daten können auch unstetige Schrittmuster, etwa 'typeIId', geeignet sein. Ohne Vorwissen sollte allerdings auf die beiden erstgenannten Varianten zurückgegriffen werden, die sich konsistent zuverlässig zeigten.

Eine weitere Erkenntnis ist, dass eine Modifikation der Prokrustes-Analyse hinsichtlich der Verwendung von euklidischen anstatt quadrierten euklidischen lokalen Di-

stanzen eine zuverlässige Verbesserung der Klassifikationsraten bewirkt. Die analytische Bestimmung der besten Skalierung und Rotation für den Vergleich zweier Shape-Signaturen erwies sich in allen benötigten Normalisierungsschritten als hilfreich, um zusätzliche Invarianzen zu erhalten. Im Vergleich zur diesbezüglichen ungleichförmigen Skalierung aufgrund der verwendeten iterativen Anpassung der Rotation im $1-Klassifizierer ist die analytische Variante durchgängig besser und zu bevorzugen.

Für die Klassifikation von Single-Touch Eingaben ist die Auswahl nach der besten Genauigkeit unter den vorgestellten Methoden hinreichend. Falls auf eine Normalisierung der Eingaben verzichtet werden soll, bietet sich die Shape-Signatur 4 an, andernfalls ist die Signatur 8 nach den Ergebnissen der durchgeführten Tests die beste Variante. Jede der Methoden kann durch die Verwendung von DTW-Strategien mit skalierbaren zeitlichen Kosten[32] weiter verbessert werden. Gründe für eine Verwendung von Signaturen, die auf Distanzen zwischen Punkten basieren finden sich nur, falls die Berechnung von Winkeln als zu teuer erachtet wird. Ist allerdings eine Normalisierung bezüglich der Rotation gewünscht, ist dieses Argument und die Ablehnung der Verwendung der Shape-Signatur 4 nicht mehr plausibel. Im folgenden Kapitel wird dennoch die Prokrustes-Analyse aufgegriffen. Das elegante Verfahren zeigte zumindest brauchbare Ergebnisse und eignet sich aufgrund seiner statistischen Herleitung für die Integration in einen Bayes'schen Klassifizierer. Es soll an dieser Stelle noch Erwähnung finden, dass eine Abweichung von der Verwendung dieses Ansatzes für Single-Touch Gesten über eine einfache Fallunterscheidung umsetzbar ist und auch empfohlen wird. In dieser Arbeit werden derartige Modifikationen jedoch nicht weiter betrachtet.

4.2 Hierarchische Klassifikation sequenzieller Multi-Touch Eingaben

An dieser Stelle wird ein probabilistischer Klassifizierer für Multi-Touch Gesten spezifiziert. Hierfür werden die zugrunde liegenden Mechanismen der Bayes'schen Klassifikation erklärt und eine spezielle Lösung abgeleitet. Das Konzept knüpft an das vorherige Kapitel an und basiert auf der Zerlegung der Eingaben in Token. Da keine feinere Segmentierung vorgenommen wird, sind Token hier die Trajektorien der jeweiligen Kontakte. Aus diesen werden lokale Merkmale bestimmt, deren Auftrittswahrscheinlichkeiten für die Erkennung der Geste genutzt werden. Die Methodik kann als Prozess einer Sensor-Fusion [50] gesehen werden und ist auf einer breiteren Ebene anwendbar, obwohl hier nur der Anwendungsfall der Gestenerkennung betrachtet wird. Der Inhalt dieses Kapitels wurde in ähnlicher Fassung in [183] veröffentlicht.

[32] Für Abschätzungen bezüglich der Geschwindigkeit von DTW siehe auch [163].

4.2.1 Bayes'sche Klassifikation

Der verfolgte Ansatz nutzt die Regeln der Bayes'schen Klassifikation, einem Spezialfall der Bayes'schen Entscheidungsfindung. Der Klassifizierer wird durch die Angabe einer Diskriminanzfunktion für jede Klasse repräsentiert. Diese Funktion hängt von der Verteilung der Merkmale ab und liefert für einen Merkmalsvektor einen Wert zurück, anhand dessen eine Entscheidung bzw. eine Zuweisung zur Klasse festgelegt wird. Selbst unter bekannten Verteilungen der Merkmale sind die Methoden der Klassifikation praktisch durch ihre Komplexität eingeschränkt. Üblicherweise werden deshalb Annahmen bezüglich der Verteilungen und damit den separierenden Diskriminanzfunktionen getroffen [63, S. 4].

Die relevanten Konzepte der Bayes'schen Entscheidung werden werden an dieser Stelle entsprechend den Ausführungen in [177, S. 1-2, 7-8] erläutert, um die theoretische Basis des hier verfolgten Ansatzes zu legen. Ein Objekt besitzt demnach zwei Parameter, ein beobachtbares Merkmal x und einen versteckten Parameter k, den Zustand des Objektes. Über der Menge $X \times K$ der paarweisen Kombinationen aus den möglichen Beobachtungen $x \in X$ und möglichen Zuständen $k \in K$ sei eine Wahrscheinlichkeitsverteilung p: $X \times K \to \mathbb{R}$ gegeben, so dass p(x,k) die Verbundwahrscheinlichkeit, dass sich das Objekt im Zustand k befindet und die Beobachtung x gemacht wird, angibt. Sei nun D eine Menge möglicher Entscheidungen und W: $K \times D \to \mathbb{R}$ eine Straffunktion (engl. penalty function), bei der W(k,d), $k \in K$, $d \in D$ die sich ergebende Strafe angibt, wenn ein Objekt sich im Zustand k befindet und die Entscheidung d getroffen wird. Des Weiteren sei q: $X \to D$ eine Funktion, welche jedem beobachtbaren Merkmal[33] $x \in X$ die Entscheidung $q(x) \in D$ zuordnet. Unter der Entscheidungsstrategie q kann nun der Erwartungswert der Straffunktion, das Risiko R(q), angegeben werden. Die Bayes'sche Aufgabe der statistischen Entscheidung besteht nun darin, unter den gegebenen Mengen X, K, D und den gegebenen Funktionen p: $X \times K \to \mathbb{R}$, W: $K \times D \to \mathbb{R}$ eine Strategie q: $X \to D$ zu finden, die das Bayes'sche Risiko

$$R(q) = \sum_{x \in X} \sum_{k \in K} p(x,k) \underbrace{W(k, q(x))}_{= \begin{cases} 0, & q(x) = k \\ 1, & otherwise \end{cases}} \tag{4.7}$$

minimiert. Die Strategie q, welche R(q) minimiert, wird Bayes'sche Strategie genannt. In der Bayes'schen Klassifikation werden über die Straffunktion W die Kosten für eine Fehlklassifikation mit 1 und für eine korrekte Klassifikation mit 0 belegt, wodurch nach der Zahl der Fehlentscheidungen minimiert wird.[34]

[33] Hier zu Anschauungszwecken diskret, auch wenn stetige Variablen verwendet werden.

[34] Eine andere Wahl der Strafen setzt Kontextwissen des Anwendungsgebietes voraus, welches hier durch die der Arbeit zugrunde liegenden Problemstellung nicht gegeben ist.

Die bezüglich obiger Voraussetzungen optimale Strategie $q : X \to K$, Bayes'sche Strategie genannt, ist in Gleichung 4.8 gegeben.

$$q(x) = \arg\max_{k \in K} g_k(x) = \arg\max_{k \in K} p(k \mid x) \tag{4.8}$$

Die a-posteriori-Wahrscheinlichkeit $p(k \mid x)$ dient als Diskriminanzfunktion $g_k(x)$. Das bedeutet die Wahl jeweils des k's mit der höchsten a-posteriori-Wahrscheinlichkeit unter der Beobachtung x minimiert in diesem Fall das Bayes'sche Risiko beziehungsweise die Wahrscheinlichkeit einer Fehlklassifikation. Dieses Vorgehen findet sich in der Literatur auch als Maximum-a-posteriori (MAP)-Klassifizierung.

Übertragen auf die Gestenklassifikation ist der Zustand k hier die unbekannte Klasse (das Label oder der Index) und die Beobachtung x entspricht dem Vektor $\vec{x}$ der extrahierten Merkmale aus möglichen Beobachtungen X. Die Menge der möglichen Entscheidungen D ist demnach gleich der Menge K möglicher Klassen. Eine Trainingsinstanz ist durch ein Tupel $(\vec{x}, k)$ aus Merkmalsvektor und Klassenlabel repräsentiert. Die Gleichung 4.8 legt mit der a-posteriori-Wahrscheinlichkeit des Zustandes k ein Entscheidungskriterium für die Klassifikation von Objekten, in dieser Betrachtung Gesten, anhand der Beobachtung x fest. Die bedingte Wahrscheinlichkeit $p(k \mid \vec{x}) = p(k \mid x_1, ..., x_n)$ als Kriterium ist intuitiv naheliegend und dennoch ist eine Klassifikation nach ihm nicht ohne Weiteres möglich. Die Informationen über die a-posteriori-Wahrscheinlichkeiten sind in konkreten Problemstellungen normalerweise nicht vorhanden. Unter Anwendung des Bayes'schen Theorems[35] [14] und Ignorierung des für einen Merkmalsvektor konstanten Normierungsfaktors $p(x_1, ..., x_n)$ lässt sich die Entscheidungsregel

$$q(\vec{x}) = \arg\max_{k \in K} p(k)p(\vec{x} \mid k) = \arg\max_{k \in K} p(k)p(x_1, ..., x_n \mid k) \tag{4.9}$$

ableiten. Für die hier vorausgesetzte kontinuierliche Verteilung der Merkmale lässt sich analog ein Kriterium unter Benutzung ihrer Wahrscheinlichkeitsdichtefunktion ableiten. Sei $f_k(\vec{x})$ eine Wahrscheinlichkeitsdichtefunktion (engl. probability density function, kurz PDF) der k-ten Klasse, so lässt sich die Gleichung 4.9 ebenfalls nach dem Bayes'schen Theorem wie folgt ableiten [50]:

$$q(\vec{x}) = \arg\max_{k \in K} \frac{likelihood}{evidence} p(k) = \arg\max_{k \in K} \frac{f_k(\vec{x})}{\sum_{l \in K} f_l(\vec{x})p(l)} p(k) \tag{4.10}$$

$$= \arg\max_{k \in K} p(k) f_k(\vec{x}) \tag{4.11}$$

Das Wissen über a-priori-Wahrscheinlichkeiten der Klassen und bedingte Auftrittswahrscheinlichkeiten der Merkmale kann durch überwachtes Lernen ermittelt bzw. ge-

[35] $p(a \mid b) = \frac{p(b \mid a)p(a)}{p(b)}$

schätzt werden. Bei der Klassifikation von Gesten bedeutet das, Templates von Gesten zusammen mit der Zuordnung zur Klasse einzugeben. Dennoch sind vereinfachende Annahmen nötig und bei in der Literatur beschriebenen Anwendungen üblich. Eine gängige Praxis ist die Annahme, dass Beobachtungen einer Klasse k Instanzen multivariat Gauß-verteilter Zufallsvektoren $\vec{x}_k \sim \mathcal{N}(\vec{\mu}_k, \Sigma_k)$ sind. Diese Annahme wird auch hier getroffen und die zur bedingten Wahrscheinlichkeitsverteilung gehörige PDF $f_k(\vec{x}_k)$ ist in Gleichung 4.12 gegeben.

$$f_k(\vec{x}_k) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma_k|^{\frac{1}{2}}} e^{(-\frac{1}{2}(\vec{x}_k-\vec{\mu}_k)^T\Sigma_k^{-1}(\vec{x}_k-\vec{\mu}_k))} \tag{4.12}$$

Der Ausdruck $(\vec{x}_k-\vec{\mu}_k)^T\Sigma_k^{-1}(\vec{x}_k-\vec{\mu}_k)$ ist als Mahalanobis-Distanz [129] bekannt und soll der Einfachheit halber zu $d_k^2(\vec{x}_k)$ abgekürzt werden. Die Anwendung der Bayes'schen Regel auf Gleichung 4.8 liefert die Maximum-a-posteriori-Entscheidungsregel aus Gleichung 4.11, welche unter der in Gleichung 4.12 gegebenen PDF und weiteren monotonen Transformationen (Logarithmus), nach [50] zu

$$q(\vec{x}) = \arg\max_{k\in K}(\ln(f_k(\vec{x})) + \ln(p(k))) \tag{4.13}$$

$$= \arg\max_{k\in K}(-\ln|\Sigma_k| - d_k^2(\vec{x}) + \ln(p(k))) \tag{4.14}$$

reduziert werden kann. Unter der - bedingt durch mangelndes Kontextwissen über die Anwendung des Klassifikators - im weiteren Verlauf genutzten Annahme gleicher a-priori-Auftrittswahrscheinlichkeiten der Gestenklassen kann der Term $\ln(p(k))$ weggelassen werden, was zu einer Maximum-Likelihood (ML)-Entscheidungsregel führt. Diese Annahme führt nur zu kleinen Modifikationen und es sollte an den entsprechenden Stellen klar werden, wie die Klassifikation je nach deren Gültigkeit anzupassen ist.

4.2.2 Herleitung des Verfahrens

Im hier verfolgten konstruktiven Ansatz wird ein (lokales) Merkmalsset extrahiert, welches für jeden Token (jeweils eine Trajektorie) dessen Form und relative Position und Ausdehnung sowie sein zeitliches Auftreten innerhalb einer Geste repräsentiert. Eine Single-Touch Geste wird dabei nur über die Form der einzelnen Trajektorie klassifiziert, während die zusätzlichen Merkmale das Erkennen von Multi-Stroke, Multi-Touch und sequenziellen Multi-Touch Gesten ermöglichen. Die Nutzung lokaler Merkmale erlaubt komplexe Gesten, wohingegen ein globales Merkmalsset ungleich größer sein müsste, um Mehrdeutigkeiten zu vermeiden. Der Nachteil ist, dass die Token einer Eingabe, denen eines Templates zum Zweck eines Vergleichs zugeordnet werden müssen. Dieses Problem kann, abhängig vom berührungssensitiven Eingabegerät, technisch gelöst werden, wenn die Quellen der Kontaktpunkte (zum Beispiel die Finger) eindeutig zugeordnet werden können. Unter solchen, sehr speziellen Voraussetzungen kann das Verfahren wesentlich vereinfacht werden. Die vorgestellte Lösung ist unabhängig von Eingabege-

räten und zeigte sich trotz ihrer Einfachheit als für den Zweck passend. Im Folgenden wird die Herleitung des Klassifizierers beschrieben, beginnend mit der Anpassung der Bayes'schen Klassifikation bezüglich der Zerlegung der Gesten in Token. Danach wird das Merkmalsset vorgestellt, welches zum einen die strukturellen und zeitlichen Relationen der Token zueinander widerspiegelt und zum anderen die Konturen eines jeden Tokens einbezieht. Nach der Herleitung des prinzipiellen Vorgehens werden Methoden zur Schätzung der Parameter für die angenommene, zugrunde liegende Verteilung der Merkmale vorgestellt. Den Abschluss bildet eine Evaluation des Gestenklassifizierers anhand eines eigens erstellten Gestensets.

Segmentierung in Token

Im Allgemeinen besteht eine Geste aus einem oder mehreren Token, die in beliebiger zeitlicher Relation, von parallel zu sequenziell, auftreten können. Werden zeitliche und strukturelle Relationen vernachlässigt, besteht eine Geste aus T Klassen von Token. Damit lässt sich die Beobachtung einer Geste in T Untermengen $x_v \in x$, $1 \leq v \leq T$ einteilen. Unter der 'naiven' Annahme, dass das Auftreten der Token bedingt (unter gegebener Gestenklasse) voneinander unabhängig und ihre Merkmale unabhängig voneinander sind, kann ein Sensor-Fusion Prozess bezüglich der Kombination unabhängiger Informationen über diese Token formuliert werden:

$$p(\underbrace{t_{k1}, ..., t_{kT}}_{k} \mid \underbrace{x_1, ..., x_T}_{x}) = \prod_{v=1}^{T} p(t_{kv} \mid x_v) \tag{4.15}$$

Bei gegebenen T gleich großen Merkmalsvektoren für jeden Token einer Gestenklasse k und Anwendung der gleichen Manipulationen wie in Abschnitt 4.2.1 ergibt sich die MAP-Regel:

$$q(\vec{x}) = \arg\max_{k \in K} \prod_{v=1}^{T} \frac{f_{kv}(\vec{x}_v)}{\sum_{l \in K} \sum_{u=1}^{T} f_{lu}(\vec{x}_v) p(t_{lu})} p(t_{kv}) \tag{4.16}$$

Abermals werden monotone Transformationen angewendet und die Entscheidungsregel $q(\vec{x})$ wird zu:

$$q(\vec{x}) = \arg\max_{k \in K} \sum_{v=1}^{T} (-\ln |\Sigma_{kv}| - d_{kv}^2(\vec{x}_v) + \ln(p(t_{kv}))) \tag{4.17}$$

Wie zuvor, kann der Ausdruck $\ln(p(t_{kv}))$ aufgrund angenommen gleicher a-priori-Wahrscheinlichkeiten vernachlässigt werden. Jede Gestenklasse besteht aus einer individuellen Menge von Token. Demnach werden für eine Klassifikation nur diejenigen Klassen in Betracht gezogen, bei denen die Anzahl der Token der in der aktuellen Eingabe entspricht. Es wird $p(k \mid x) = 0$ für jede Klasse k festgelegt, deren Tokenzahl unterschiedlich zur Eingabe ist. Ohne auf manuelles Eingreifen des Nutzers, einer Beschränkung der Eingabe oder ein spezielles Eingabegerät zur eindeutigen Identifikation

der Token in der Eingabe angewiesen zu sein, bleibt die Frage nach deren Zuordnung zur jeweiligen PDF. Dazu wird eine Maximum-Likelihood-Zuordnung gewählt, das heißt eine Zuordnung, die das Auftreten der Eingabe bestmöglich erklärt.

$$\sum_{v=1}^{T} G(t_v, f_{kh(v)}) \rightarrow \max_h \tag{4.18}$$

$$\sum_{v=1}^{T} (-\ln|\Sigma_{kh(v)}| - d^2_{kh(v)}(\vec{x}_v)) \rightarrow \max_h \tag{4.19}$$

In Gleichung 4.18 ist h ein bijektives Matching der Nummer einer PDF zu einem Token v und G ist der Nutzen eines Matchings, hier die log-Likelihood welche sich ergibt, wenn das Argument x_v der PDF, die durch h zugewiesen wurde, übergeben wird. Die Aufgabe der Maximierung von Gleichung 4.18 ist ein klassisches Zuordnungsproblem und kann zum Beispiel mit der 'Ungarischen Methode' [108] gelöst werden. Es gilt nun die in Gleichung 4.20 vorgeschlagene Entscheidungsregel:

$$q(\vec{x}) = \arg\max_{k \in K} \max_h \sum_{v=1}^{T} (-\ln|\Sigma_{kh(v)}| - d^2_{kh(v)}(\vec{x}_v)) \tag{4.20}$$

Das Merkmalsset

Die Merkmale der Token werden nach zwei Kategorien erstellt: Zum einen nach der Form ihrer Trajektorie, zum anderen nach ihrer zeitlichen und strukturellen Platzierung innerhalb der Geste.

Um Token anhand ihrer Positionierung innerhalb der Geste zu unterscheiden, wurde ein Merkmalsset entwickelt, welches die zur Geste relativen strukturellen und zeitlichen Eigenschaften gut repräsentiert. Die Merkmale sind angelehnt an die vorangehende Arbeit des Verfassers in [181] und - bis auf das zeitliche Merkmal - in Abbildung 4.13 dargestellt. Jedes Merkmal ist lokal auf einen Token bezogen und steht im relativen Verhältnis zur gesamten oder der um den betreffenden Token reduzierten Geste. Die zeitliche Einordnung eines Tokens innerhalb einer Geste wird über die Differenz seiner Startzeit (Aufnahme des zur Trajektorie gehörigen Kontaktes) zum Startzeitpunkt (erster Kontakt) der Geste im Verhältnis zur halben Gesamtdauer der Gesteneingabe festgehalten. Auf diese Weise werden die Invarianz gegenüber der Ausführungsgeschwindigkeit sowie Werte in [0..2] erzeugt.

Merkmale 1, 2 und 3 in Abbildung 4.13 sind normalisierte Distanzen zwischen dem Schwerpunkt des verbleibenden Teils (ohne aktuell betrachteten Token) der Geste und dem Startpunkt, Endpunkt sowie dem Schwerpunkt des betrachteten Tokens. Um Invarianzen gegenüber Skalierung zu erhalten, wird die Geste bezüglich ihrer Größe vorher gleichförmig normalisiert, indem die Distanz aller Punkte zum Schwerpunkt der Geste auf 1 gemittelt wird. Die Geste wird dazu verschoben, so dass ihr Schwerpunkt auf dem Koordinatenursprung liegt. Danach wird jede Koordinate eines jeden Punktes durch

den Mittelwert der quadrierten euklidischen Distanzen der Punkte zum Schwerpunkt geteilt.

Die Merkmale 4-7 sind Winkelmaße und berücksichtigen Start- und Endpunkt eines Tokens und die Schwerpunkte (graue Punkte in den ersten beiden Bildern) des verbleibenden Teils der Geste, sowie der kompletten Geste. Im dritten Bild (rechts) steht der graue Punkt für den Durchschnitt der Startpunkte aller Token außer dem aktuell betrachteten, für den das Merkmal ermittelt wird. Die Informationen, die durch die meisten Winkelmerkmale und die Distanzmaße extrahiert werden, sind nicht frei von Redundanz. Beide Merkmalsarten enthalten Informationen bezüglich der Rotation und Entfernung des Token. Alle Winkel werden wie in [169, 224] über ihren Sinus und Kosinus erfasst, um bei Radianten oder Gradmaßen inhärente Diskontinuitäten zu vermeiden. Daher ergibt sich eine Gesamtzahl von 12 Merkmalen (Distanzen, Winkel, Zeit).

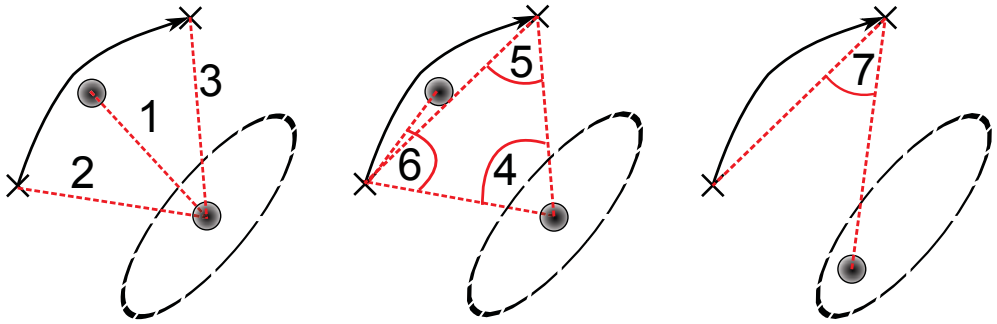


Abbildung 4.13: Ein Merkmalsset, welches die strukturelle Relation eines Tokens innerhalb einer Geste beschreibt. Pfeile stellen den Token dar, für den das Merkmal gewählt wird, gestrichelte Ellipsen symbolisieren den verbleibenden Teil der Geste mit einer beliebigen Anzahl weiterer Token.

Die Merkmale sind so gewählt, dass sie die verschiedenen Freiheitsgrade, die ein Token neben seiner Form innerhalb einer Geste haben kann, abdecken. Diese Eigenschaften sind seine relative Distanz, Größe, Neigung und Rotation. Weiterhin sollen die Merkmale möglichst eine Gauß-artige Verteilung aufweisen, um die Annahmen für den gewählten Bayes'schen Ansatz zu unterstützen. Jedes Merkmal bekommt das gleiche Gewicht, indem es auf das gleiche (im Falle der Distanzen gleich erwartete) Intervall möglicher Minimal- und Maximalwerte skaliert wird.

Die Form eines Tokens wird durch eine Shape-Signatur repräsentiert. Es wird angenommen, dass die Form bedingt unabhängig seiner zeitlichen und strukturellen Relationen zur Geste ist. In Anlehnung an Kapitel 4.1 wird die Ähnlichkeit von Token durch eine Distanz ihrer Shape-Signaturen ermittelt. Unter Kombination mit dem statistischen Ansatz lassen sich im Allgemeinen mit Wahrscheinlichkeiten gewichtete normalisierte Distanzen, etwa durch DTW berechnete, für die Klassifikation nutzen. Hier wird davon Gebrauch gemacht, dass manche Distanzen eng verwandt mit Wahrscheinlichkeitsdichtefunktionen sind. Die entsprechende Wahl der Shape-Signatur und des Distanzmaßes - genauer gesagt, der quadrierten euklidischen Distanz zwischen norma-

lisierten Punkten zweier Trajektorien (eines Templates und der aktuellen Eingabe) - kann als Mahalanobis-Distanz einer standardnormalverteilten PDF aufgefasst werden.

Gleichung 4.15 wird nun insoweit angepasst, als dass $x_v = (y_v, z_v) \in x, 1 \leq v \leq T$ und y_v die Merkmale aus Abbildung 4.13 sowie den zeitlichen Versatz enthält sowie z_v die Shape-Signatur repräsentiert:

$$p(k \mid x) = \prod_{v=1}^{T} p(t_{kv} \mid y_v, z_v) \tag{4.21}$$

Die bedingte Unabhängigkeit von y_v und z_v unter t_v führt zu:

$$p(t_v \mid y_v, z_v) = \frac{p(t_v \mid y_v,)p(t_v \mid z_v,)p(z_v)}{p(t_v)p(z_v \mid y_v)} \tag{4.22}$$

$$\propto f_v(y_v)g_v(z_v)p(t_v)\frac{p(z_v)}{p(z_v \mid y_v)} \tag{4.23}$$

In Gleichung 4.23 wird der Normalisierungsfaktor ignoriert und $g_v(z_v)$ ist die vorausgesetzte PDF der Shape-Signatur. Zum Zwecke der Klassifikation ist der Ausdruck $\frac{p(z_v)}{p(z_v \mid y_v)}$ nicht von Bedeutung, ebenso wie $p(t_v)$ bei angenommenen gleichen a-priori-Auftrittswahrscheinlichkeiten. Die Anpassung der Entscheidungsregel in Gleichung 4.20 bezüglich dieser Modifikationen und die Einführung der gewohnten Notationen für Merkmalsvektoren führt zu:

$$q(\vec{y}, \vec{z}) = \arg\max_{k \in K} \max_{h} \sum_{v=1}^{T} r(\vec{y}, \vec{z}, k, h) \tag{4.24}$$

mit

$$r(\vec{y}, \vec{z}, k, h) = -\ln|\Sigma_{kh(v)}| - d^2_{kh(v)}(\vec{y}_v) + \ln(g_{kh(v)}(\vec{z}_v)) \tag{4.25}$$

Im Allgemeinen kann das fehlende Wissen über $p(y_v \mid t_v)$ ebenfalls mit Hilfe der Diskriminanzanalyse kompensiert werden. Die Anzahl der Merkmale ist allerdings sehr hoch und die Schätzung der Parameter demzufolge bei kleinen Stichprobengrößen aussichtslos. Daher wird ein nicht-parametrisches Nächste-Nachbar-Verfahren gewählt und seine Implikationen im statistischen Kontext untersucht. Das gewählte Verfahren ist die Prokrustes-Analyse [68], welche Resampling, Translation (Schwerpunkt zum Koordinatenursprung), gleichförmige Skalierung und Rotation einer Sequenz sogenannter 'landmark points' (zeitgestempelte x,y-Koordinaten) in einem Normalisierungsschritt durchführt und über die quadrierte euklidische Distanz zweier solcher Sequenzen deren Ähnlichkeit feststellt. Die in [217] verwendete Methode verfolgt die gleiche Strategie. Allerdings wird dort eine ungleichförmige Skalierung verwendet, um die iterative Korrektur der Rotationsvarianz vorzunehmen. Obwohl im Allgemeinen gute Resultate damit erzielt werden, sind nahezu eindimensionale Eingaben unter dem Ansatz pro-

blematisch. Weiterhin nutzt die Prokrustes-Analyse aus, dass der Versatz bezüglich der Rotation zweier solcher Shape-Signaturen analytisch bestimmt werden kann. Aus Sicht der statistischen Klassifikation impliziert die Verwendung der Methode die Standardnormalverteilung der 'landmark points'. In Algorithmus 4.3 sind die Schritte der Prokrustes-Analyse wiedergegeben (siehe auch Ausführungen in Kapitel 4.1).

Algorithm 4.3 ProcrustesShapeDistance(I,G)

Require: INPUT: I - Trajektorie des eingegebenen Token
Require: INPUT: G - Trajektorie des Tokens eines Templates
▷ Resampling der Trajektorien zu gleichen Längen
$i \leftarrow$ Resample(I)
$t \leftarrow$ Resample(G)
$i \leftarrow$ TranslateToOrigin(I)
$t \leftarrow$ TranslateToOrigin(G)
▷ Skalierung anhand des quadratischen Mittels der Distanzen der Punkte einer Trajektorie zu deren Schwerpunkt
$i \leftarrow$ ScaleUniformly(I)
$t \leftarrow$ ScaleUniformly(G)
▷ Korrektur des Unterschieds der Orientierung zweier Trajektorien anhand der Minimierung der Summe quadrierter euklidischer Distanzen der Punktepaare
$i \leftarrow$ RotateToMatch(I,G)
▷ Der Rückgabewert liegt aufgrund der Skalierung in [0..2]
return SquaredEuclideanDistance(I, G)

Die Skalierung in Algorithmus 4.3 wird leicht modifiziert und per Division durch die mit $\sqrt{2}$ gewichtete Summe der quadrierten euklidischen Abstände der Punkte zum Schwerpunkt vorgenommen, was die durchschnittliche Entfernung der Punkte zu ihrem Schwerpunkt auf 1/2 normalisiert. Sei eine Trajektorie I und ihr Schwerpunkt durch $(\bar{x}, \bar{y})$ gegeben, so berechnet sich der Skalierungsfaktor, der die Abweichung der Punkte vom Schwerpunkt normalisiert, nach Gleichung 4.26:

$$sf = \frac{1}{\sqrt{2 \cdot \sum_{p=1}^{n} (I_x^{(p)} - \bar{x})^2 + (I_y^{(p)} - \bar{y})^2}} \tag{4.26}$$

Die Skalierung ist unabhängig der Translation und ohne Beschränkung der Allgemeinheit gilt nach Verschiebung des Schwerpunktes auf den Koordinatenursprung:

$$\sum_{p=1}^{n} ||sf \cdot I^{(p)}||_2^2 = \sum_{p=1}^{n} ((sf \cdot I_x^{(p)})^2 + (sf \cdot I_y^{(p)})^2) = \sum_{p=1}^{n} ||sf \cdot G^{(p)}||_2^2 = \frac{1}{2} \tag{4.27}$$

Nachdem I und G skaliert wurden, kann nach [161] der Winkel ermittelt werden, um den die Eingabe rotiert werden muss, um die Summe quadratischer Distanzen zwischen

den Punkten aus I und G zu minimieren (Methode der kleinsten Quadrate):

$$\alpha = \tan^{-1}\left(\frac{\sum_{p=1}^{n} G_x^{(p)} I_y^{(p)} - G_y^{(p)} I_x^{(p)}}{\sum_{p=1}^{n} G_x^{(p)} I_x^{(p)} + G_y^{(p)} I_y^{(p)}}\right) \tag{4.28}$$

Zwei gleichlange Trajektorien einer Eingabe I und eines Templates G können nun verglichen werden, indem beide zum Koordinatenursprung verschoben, nachfolgend durch sf skaliert werden und die Eingabe durch $-\alpha$ um ihren Schwerpunkt bezüglich Minimierung des Abstands bestmöglich rotiert wird. Dieser Prozess wird hier als Merkmalsextraktion und die auf diese Weise normalisierten Shape-Signaturen als Merkmale in $\vec{z}_v$ aufgefasst. Die Distanz zwischen derart normalisierten Shape-Signaturen wird durch die Summe der quadrierten euklidischen Abstände berechnet:

$$\hat{d}_G^2(I) = \sum_{p=1}^{n} ((I_x^{(p)} - G_x^{(p)})^2 + (I_y^{(p)} - G_y^{(p)})^2) = 1 - 2\sum_{p=1}^{n} (I_x^{(p)} G_x^{(p)} + I_y^{(p)} G_y^{(p)}) \tag{4.29}$$

Unter angenommener gleicher Gauß-Verteilung ist die Differenz der beiden Shape-Signaturen mit dem Nullvektor als Erwartungswert ebenfalls Gauß-verteilt. In Gleichung 4.29 entspricht damit $\hat{d}_G^2(I)$ bis auf konstante Faktoren der log-Likelihood der PDF der Differenz von Eingabe und Template unter Standardnormalverteilung ebenso wie dem Wert der Mahalanobis-Distanz.

Aufgrund des gewählten Skalierungsfaktors gilt für die euklidische Norm der Differenz beider Shape-Signaturen[36] bwz. dem euklidischen Abstand (siehe Gleichung 4.27) unter Ausnutzung der Dreiecksungleichung:

$$0 \leq \sqrt{\hat{d}_G^2(I)} = ||I - G||_2 \leq ||I||_2 + ||G||_2 = 2 \cdot \sqrt{\frac{1}{2}} \tag{4.30}$$

$$0 \leq \hat{d}_G^2(I) \leq 2 \tag{4.31}$$

Die quadrierte euklidische Distanz in Gleichung 4.29 ist demnach zu [0..2] normalisiert. Ohne weitere Parameter führt die Prozedur zu gleichen Gewichten wie bei den strukturellen Merkmalen. Die Integration der Shape-Signaturen in die Entscheidungsregel aus Gleichung 4.24 wird letztendlich auf die beschriebene Weise vorgenommen.

Parameter Schätzung

Bisher wurde das Problem der Parameterschätzung für Σ_{kv} und μ_{kv} innerhalb der PDF eines Tokens nicht betrachtet. Für die Form eines Tokens wird implizit die Annahme einer Standardnormalverteilung getroffen. Da strukturelle und zeitliche Eigenschaften

[36] Der Einfachheit halber werden, ohne eine neue Notation einzuführen, die Signaturen I und G an dieser Stelle als Vektoren aufgefasst, in denen alle Punkte in ihrer Reihenfolge jeweils durch abwechselnd ihre Abszisse und Ordinate abgelegt sind.

des Tokens über ein Set von 12 Merkmalen erfasst werden, führen Maximum-Likelihood-Schätzungen der Kovarianzen unvermeidbar zu singulären Matrizen, wenn weniger als 13 Templates für die Schätzung herangezogen werden ('small sample size problem') [63, S. 39]. Die Schätzmethode sollte demnach robust gegenüber kleinen Stichprobengrößen sein, auch da die Spezifizierung einer größeren Zahl an Templates wenig nutzerfreundlich ist. Ein weiteres Problem bei der Parameterschätzung ist die Gruppierung der Token in den Templates zu eigenen Klassen. Die Information, welche Token eines Gestentemplates denen in den anderen Templates der gleichen Gestenklasse entsprechen, ist nicht verfügbar, ohne dass der Nutzer oder das Eingabegerät diese mitteilen. Eine Trainingsroutine, die die Identifizierung der Token vom Nutzer verlangt, wäre unnötig umständlich. Im Allgemeinen ist davon auszugehen, dass die Token der Templates keine explizite Ordnung haben, nach denen eine Gruppierung der Token, die ein Nutzer für gleich ansieht, innerhalb der Gesten einer Klasse erfolgen kann. Während der Klassifikation wird dieses Problem durch eine Maximum-Likelihood-Zuordnung unter einer Diskriminanzfunktion gelöst. Mittels dieser Funktion kann auch ein Clustering der Token (visualisiert in Abbildung 4.14) erfolgen.

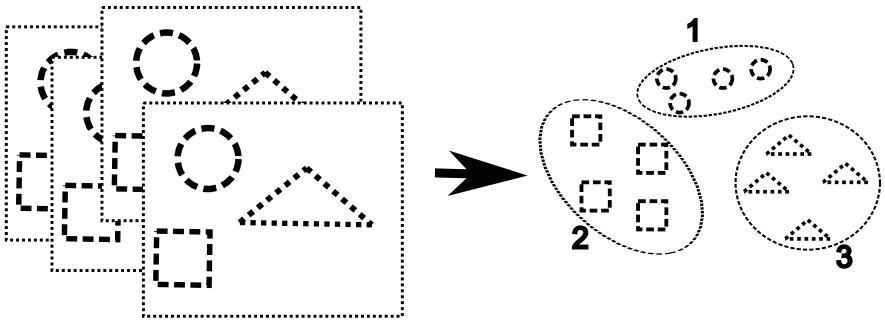


Abbildung 4.14: Die Gruppierung aller gleichen Token einer Klasse (hier als gleiche geometrische Symbole dargestellt) kann durch Clustering mit Hilfe der PDF als Ähnlichkeitsfunktion erreicht werden.

Ein solches Clustering nutzt die Entscheidungsregel, um die Ähnlichkeit der Token innerhalb ihrer Gruppen zu maximieren. Allerdings würden dafür selbst wieder Parameterschätzungen oder -annahmen nötig, unter denen ein Bootstrapping-Mechanismus realisiert wird.[37] Jedem Token eines Templates kann dann allerdings eine PDF eindeutig zugeordnet werden, unter der ein Vergleich mit einem Token in der Eingabe möglich ist. Dennoch wäre auch unter einer gefundenen Gruppierung für eine gute Parameterschätzung eine hohe Anzahl an Templates nötig. Aus diesem Grund wird hier eine Parameterschätzung bemüht, die ohne Gruppierung der Token auskommt, also eine gemeinsame Schätzung über alle Token wenigstens einer Klasse vornimmt. Zusätzlich wird davon Gebrauch gemacht, dass Differenzen multivariater Gauß-verteilter Variablen wieder Gauß-verteilt sind, mit dem Nullvektor als Erwartungswert und summierten Kovarianz-Matrizen. Eine durch einen Nutzer eingegebene Geste kann als i.i.d.

[37] Unter einer Ähnlichkeitsfunktion kann die 'optimale' Gruppierung der Token innerhalb einer Gestenklasse kombinatorisch in $\mathcal{O}(T!^G)$ bei T Token und G Templates pro Klasse gelöst werden.

Stichprobe der gleichen Grundgesamtheit gesehen werden, aus der die Templates der entsprechenden Klasse stammen. Die Klassifikation beruht daher auf Differenzen zwischen den Token der Eingabe und denen eines Templates. Es wird maximal ein Schätzer einer Kovarianzmatrix[38] pro Gestenklasse benutzt.

Die Wahl der Kovarianzmatrix hat direkten Einfluss auf die Diskriminanzfunktion innerhalb der Entscheidungsregel (siehe Gleichung 4.14 in Abschnitt 4.2.1). Im allgemeinen Fall der quadratischen Diskriminanzanalyse (QDA) kann eine hyperquadratische Entscheidungsgrenze modelliert werden, allerdings unter quadratischem Zeitaufwand (abhängig von der Größe des Merkmalssets) [50]. Der Aufwand der Bestimmung der Determinante und der Inversen der Kovarianzmatrix kann nach dem Training einmalig geschehen, aber die Berechnung der Mahalanobis-Distanz hat quadratischen Aufwand. Unter Diagonalmatrizen oder gleichen Kovarianzmatrizen für alle Klassen kann die Diskriminanzfunktion als Linearkombination der Merkmale ausgedrückt werden (Lineare Diskriminanzanalyse, LDA). Die Entscheidungsgrenzen sind allerdings dann nur noch als Hyperebenen modellierbar [50].

Im Fall der multivariaten Normalverteilung ist die Stichprobengröße für eine Entscheidung, ob eine LDA oder QDA gewählt wird, nach Wahl und Kronmal [204] entscheidend. Demnach werden bei kleinen Stichproben für große Dimensionen (Merkmalszahl n>6) bessere Ergebnisse durch die lineare Diskriminanzanalyse erreicht, während für große Unterschiede in den Kovarianzen und große Dimensionen QDA (unter genügend großen Stichproben) besser geeignet ist. Das Problem, genügend große Stichproben für die Schätzung der mit n^2 von der Merkmalszahl abhängigen Werte in der Kovarianzmatrix zu erhalten, ist auch unter dem in [15] eingeführtem Begriff 'curse of dimensionality' bekannt. Es reicht demnach nicht aus, die Anzahl der Samples linear zur Anzahl der Merkmale zu skalieren, wenn die Güte der Schätzung gleich bleiben soll. Cortes et al. [37, S. 274] merken an, dass die Schätzung von $\omega(n^2)$ Parametern mit weniger als $10n^2$ Daten nicht zuverlässig ist. Bei Anwendungen in der Template-basierten Gestenerkennung kann man allerdings von einer eher geringen Stichprobenzahl (Templates pro Gestenklasse) ausgehen. Der in [170, S. 334] vorgeschlagene Ansatz, die Merkmale zu reduzieren, wenn zu wenige Samples in der Stichprobe sind, ist aufgrund des konstruierten Merkmalssets nicht anwendbar.

Der hier vorgestellte Klassifikator soll möglichst mit kleinen Stichprobengrößen auskommen, da aus Nutzersicht das Anlegen großer Template-Datenbanken wenig gebrauchstauglich ist. Daher finden zwei Ansätze Anwendung, die ebenfalls für kleine Stichproben geeignet sind: zum einen gemeinsame Kovarianzmatrizen, um eine größere Anzahl einzubeziehen und damit den Rang der Matrix zu erhöhen (möglichst soweit, dass Invertierbarkeit gegeben ist); zum anderen die Verwendung robusterer Schätzmethoden. Von den möglichen Schätzern der Kovarianzen werden sechs Methoden im Folgenden untersucht. Bis auf die vorgestellte Variante 2 führen alle zu LDA, wobei

[38] Der konstante Faktor 2, welcher sich aus der Differenz der zwei gleichen Gauß-Verteilungen ergibt, kann vernachlässigt werden.

weitere QDA-Methoden leicht mit den gleichen Mitteln erreicht werden können, wenn keine klassenübergreifenden Schätzer gewählt werden. Es ist anzumerken, dass nach Enis und Geisser [54, S. 404] die Entscheidung aufgrund der unter geschätzten Parametern gewonnenen Diskriminanzfunktion nicht mehr mit der Bayes'schen Strategie (Minimierung des Fehlklassifikations-Risikos) zu begründen ist. Im Allgemeinen ist den Autoren nach eine Sample-Diskriminanzfunktion, die das Bayes'sche Risiko minimiert, außerdem nicht linear[39], auch wenn es die aus der PDF abgeleitete Diskriminanzfunktion ist. So sei es eher intuitiv und durch die Erfahrung zu begründen, dass derart gewonnene Sample-Diskriminanzfunktionen durchaus erfolgreich angewendet werden.

Im Folgenden seien $|C|$ die Anzahl der Klassen, $|S_k|$ die Menge der Samples für eine Klasse k, $|T_k|$ die Anzahl der Token in dieser Klasse und $\vec{y}_{ksv}$ der Merkmalsvektor des v-ten Tokens des s-ten Samples der k-ten Klasse.

Konventionelle Schätzer

Variante 1: Die einfachste Form geht von einer multivariaten Standardnormalverteilung mit Einheitsmatrix $\Sigma_k = E_{|F|}$ in Größe der Merkmalszahl $|F|$ aus. Diese Methode ignoriert mögliche Korrelationen zwischen den Merkmalen und gewichtet sie gleich. Die Mahalanobis-Distzanz wird zur quadrierten euklidischen Norm. Im Widerspruch zum hier gewählten konstruktiven Ansatz der Merkmalsextraktion wird dadurch die Unabhängigkeit der Merkmale impliziert ('naiver' Bayes Klassifizierer). In der Praxis führt diese Annahme nach [172, S. 594] und [166] dennoch häufig zu guten Resultaten, auch wenn die Annahme der Unabhängigkeit der Merkmale nicht wahr ist.

Variante 2: Um mögliche Korrelationen zwischen den Merkmalen zu berücksichtigen, ist die gemeinsame Kovarianzmatrix pro Gestenklasse über alle Templates und alle Token eine bessere Wahl. Für die Schätzung der Kovarianzmatrix Σ_k ist die Stichproben-Kovarianzmatrix

$$\overline{\Sigma}_k = \frac{1}{|S_k| \cdot |T_k|} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} (\vec{y}_{ksv} - \hat{\vec{\mu}}_k)(\vec{y}_{ksv} - \hat{\vec{\mu}}_k)^{\mathsf{T}} \tag{4.32}$$

bekanntermaßen nicht erwartungstreu. Üblicherweise werden Bias-korrigierte Varianten für die Matrix der Kovarianzen und Varianzen (siehe auch [63, S. 21]) verwendet. Der Bias-korrigierte Maximum-Likelihood-Schätzer ist:

$$\Sigma_k = \frac{1}{|S_k| \cdot |T_k| - 1} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} (\vec{y}_{ksv} - \hat{\vec{\mu}}_k)(\vec{y}_{ksv} - \hat{\vec{\mu}}_k)^{\mathsf{T}} \tag{4.33}$$

mit

$$\hat{\vec{\mu}}_k = \frac{1}{|S_k| \cdot |T_k|} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} \vec{y}_{ksv} \tag{4.34}$$

[39] Wenn die Sample-Größe zunimmt, strebt sie allerdings gegen eine lineare Form.

Variante 3: Für kleine Stichprobengrößen pro Gestenklasse ist zu erwarten, dass Variante 2 keine robusten Schätzer liefert. Daher wird auch ein gemeinsamer Schätzer über alle Klassen, Templates und Token untersucht:

$$\Sigma_k = \Sigma = \frac{1}{\sum_{k=1}^{|C|} |S_k| \cdot |T_k| - 1} \sum_{k=1}^{|C|} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} (\vec{y}_{ksv} - \hat{\vec{\mu}})(\vec{y}_{ksv} - \hat{\vec{\mu}})^\mathsf{T} \tag{4.35}$$

mit

$$\hat{\vec{\mu}} = \frac{1}{\sum_{k=1}^{|C|} |S_k| \cdot |T_k|} \sum_{k=1}^{|C|} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} \vec{y}_{ksv} \tag{4.36}$$

Variante 4: Ein weiterer - unter Gültigkeit der impliziten Annahme, dass jede Klasse die gleiche Kovarianzmatrix besitzt, konsistenter [200, S. 2161] - Schätzer der gemeinsamen Kovarianzmatrix kann über die Stichproben der verschiedenen Klassen berechnet werden. Ähnlich zur Variante 3 wird ein sogenannter 'vereinigter' (engl. pooled) Schätzer berechnet, indem der Durchschnitt der Kovarianzmatrizen aus jeder Klasse gebildet wird:

$$\Sigma = \frac{1}{S} \sum_{k=1}^{K} \Sigma_k \tag{4.37}$$

$$\Sigma_k = \Sigma = \frac{1}{|C|} \sum_{k=1}^{|C|} \frac{1}{|S_k| \cdot |T_k| - 1} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} (\vec{y}_{ksv} - \hat{\vec{\mu}}_k)(\vec{y}_{ksv} - \hat{\vec{\mu}}_k)^\mathsf{T} \tag{4.38}$$

Shrinkage-Estimation:
Maximum-Likelihood-Schätzer sind zwar asymptotisch optimal, aber wie ihre Bias-korrigierten Pendants statistisch ineffizient [176]. Verbesserte Schätzer, die zusätzlich robuster, immer positiv definit (garantierte Invertierbarkeit, auch bei kleinen Stichprobengrößen) und gut konditioniert (numerisch invertierbar) sind, können durch die Methode des 'Schrumpfens' (engl. shrinkage) gefunden werden [176]. Die Idee besteht darin, einen Kompromiss zwischen Bias und Varianz einzugehen. Der Bias entspricht dabei einem systematischen Fehler durch abgeschwächte Modellannahmen, während die Varianz demgegenüber den Fehler durch Überanpassung angibt, also wie stark ein Schätzer die Stichprobe abbildet und damit weniger aussagekräftig gegenüber dem wirklichen zugrunde liegenden Modell gestaltet ist. In [113, S. 12] wird vorgeschlagen, einen gewichteten Durchschnitt eines verzerrten (engl. biased) und unverzerrten (engl. unbiased) Schätzers zu bilden. Im Prinzip wird ein konventioneller Schätzer U (behaftet mit hoher Varianz aufgrund großer Anzahl Parameter, aber unverzerrt; etwa aus Gleichung 4.33) in Richtung eines robusteren (varianzärmeren, aber möglicherweise mit systematischem Fehler bzw. Verzerrung behafteten; etwa Gleichung 4.37) Ziels T 'geschrumpft', indem beide zu einem gewichteten Mittel U^* kombiniert werden [176]:

$$U^* = \lambda T + (1 - \lambda) U \tag{4.39}$$

Hoffbeck und Landgrebe beschreiben in [80] ebenfalls einen solchen kombinierten Schätzer aus gemeinsamer Kovarianzmatrix und Stichproben-Kovarianzmatrix. Sie vergleichen ihren Schätzer mit dem Ansatz der regularisierten Diskriminanzanalyse (engl. regularized discriminant analysis) und der Stichproben-Kovarianzmatrix bzw. der gemeinsamen Kovarianzmatrix anhand einer Maximum-Likelihood-Klassifikation von multivariat normalverteilten Daten. Bei der regularisierten Diskriminanzanalyse[40] werden ebenfalls Schätzer kombiniert: zusätzlich zur gemeinsamen Kovarianzmatrix und der Stichproben-Kovarianzmatrix fließt die Identitätsmatrix ein. Die Bezeichnung drückt die Sichtweise aus, eine Abwägung zwischen linearer und quadratischer Diskriminanzanalyse durch die Kombination der unter verschiedenen Annahmen erlangten Schätzer der Kovarianzmatrizen zu finden. In ihren Tests [80] mit 15 zufällig gezogenen Trainings-Samples deutet sich an, dass die Stichproben-Kovarianzmatrix nur sinnvoll ist, wenn sich die Kovarianzmatrizen der einzelnen Klassen stark unterscheiden. Dennoch sind die robusteren Schätzer zuverlässig besser. Bei gleichen Kovarianzmatrizen eignet sich die gemeinsame Kovarianzmatrix, allerdings zeichnet sich auch hier gerade bei höheren Dimensionen ein Vorsprung der robusten Schätzer ab. In [200] wird dieser Schätzer für die MAP-Schätzung unter Gauß'schen Verteilungen für die Klassifikation von Gesichtsausdrücken getestet. Auch hier schneidet der kombinierte Schätzer wesentlich besser ab als die gemeinsame oder Stichproben-Kovarianzmatrix.

Obwohl mehrere Methoden (z.B. Vergleichsprüfung/'cross validation') existieren, um ein, in der Praxis gut funktionierendes Intensitätsmaß λ zu bestimmen, kann dieses interessanterweise auch analytisch hergeleitet werden, ohne Annahmen einer Verteilung zugrunde zu legen [176].[41] Das Intensitätsmaß wird dabei so gewählt, dass der erwartete quadratische Fehler zwischen dem Parameter und seinem Schätzer minimiert wird. Der interessierte Leser kann die Herleitungen in den angegebenen Quellen nachlesen.

An dieser Stelle soll zielführend der Formalismus wiedergegeben werden, um die benötigten Anpassungen zu verdeutlichen. Das Verfahren basiert auf den Arbeiten von [176] und [145]. In [197, S. 27] sind die grundsätzliche Prozedur zur Gewinnung des Schätzers sowie weitere Hinweise und Modifikationen gegenüber dem in [176] beschriebenen Ansatz zusammengefasst dargestellt.[42] Aus der Zusammenführung der Informationen aus diesen Quellen wurden die untenstehenden Formeln abgeleitet. Der Schätzer wird erhalten, indem separat die Korrelationen in Richtung Null und die Varianzen in Richtung ihres Medians 'geschrumpft' werden [197].

[40] Friedman [62, S. 168] führt diesen Begriff ein und definiert auch die Wahl der Regulierungs-Parameter an dem Ziel, Fehlklassifikationen zu minimieren.

[41] Im Gegensatz zu Methoden, welche die Parameter über Diskriminanzanalyse bestimmen [62, 80].

[42] Die Methode ist auch in der Statistik Software R [156] im Paket 'corpcor' implementiert. Siehe zudem auch [175] für weitere Hinweise.

Die konkrete Umsetzung der oben beschriebenen Methode des 'Schrumpfens' ist in Gleichung 4.40 wiedergegeben.

$$s^*_{i,j} = \begin{cases} \lambda^*_2 s_{median} + (1-\lambda^*_2)s_i, & i = j \\ r^*_{i,j}\sqrt{s^*_i s^*_j}, & i \neq j \end{cases} \tag{4.40}$$

$$r^*_{i,j} = (1-\lambda^*_1)r_{i,j} \tag{4.41}$$

$$\lambda^*_1 = min(1, \frac{\sum_{i\neq j} \hat{Var}(r_{i,j})}{\sum_{i\neq j} {r_{i,j}}^2} \tag{4.42}$$

$$\lambda^*_2 = min(1, \frac{\sum_{i=1}^{m} \hat{Var}(s_i)}{\sum_{i=1}^{m} (s_i - s_{median})^2}) \tag{4.43}$$

$$\hat{Var}(s_{i,j}) = \hat{Cov}(s_{i,j}, s_{i,j}) \tag{4.44}$$

Unter gegebenem unverzerrten bzw. erwartungstreuen Schätzer der Kovarianzmatrix mit den Einträgen $s_{i,j}$ für die Merkmale i und j und den verzerrten Pendants $\overline{w_{i,j}}$ ist $s^*_{i,j}$ der neue 'geschrumpfte' Schätzer für die Merkmale i und j. Die Variable $s_i = s_{i,i}$ ist die Varianz des i-ten Merkmals, s_{median} der Median der Varianzen aller Merkmale und $r_{i,j}$ die Korrelation des i-ten und j-ten Merkmals. Die Varianz der Kovarianzen der Merkmale i und j ist mit $\hat{Var}(s_{i,j}) = \hat{Cov}(s_{i,j}, s_{i,j})$ angegeben.

Die Methode des 'Schrumpfens' wurde nachfolgend auf die Variante 2 (resultierende Variante 5) und Variante 3 (resultierende Variante 6) angewandt.

Variante 5:

$$m = |S_k| \cdot |T_k| \tag{4.45}$$

$$s^*_{i,j} = \Sigma_{k,ij} = \frac{1}{m-1} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} \underbrace{(\vec{y}_{ksv,i} - \hat{\vec{\mu}}_{k,i})(\vec{y}_{ksv,j} - \hat{\vec{\mu}}_{k,j})}_{w_{ksv,ij}} \tag{4.46}$$

$$\overline{w_{k,ij}} = \frac{1}{m} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} w_{ksv,ij} \tag{4.47}$$

$$r_{i,j} = \frac{\overline{w_{k,ij}}}{\sqrt{\overline{w_{k,ii}}\,\overline{w_{k,jj}}}} \tag{4.48}$$

$$\hat{Cov}(s_{i,j}, s_{a,b}) = \frac{m}{(m-1)^3} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} (w_{ksv,ij} - \overline{w_{k,ij}})(w_{ksv,ab} - \overline{w_{k,ab}}) \tag{4.49}$$

Variante 6:

$$m = \sum_{k=1}^{|C|} |S_k| \cdot |T_k| \tag{4.50}$$

$$s^*_{i,j} = \Sigma_{ij} = \frac{1}{m-1} \sum_{k=1}^{|C|} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} \underbrace{(\vec{y}_{ksv,i} - \hat{\vec{\mu}_i})(\vec{y}_{ksv,j} - \hat{\vec{\mu}_j})}_{w_{ksv,ij}} \tag{4.51}$$

$$\overline{w_{ij}} = \frac{1}{m} \sum_{k=1}^{|C|} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} w_{ksv,ij} \tag{4.52}$$

$$r_{i,j} = \frac{\overline{w_{ij}}}{\sqrt{\overline{w_{ii}} \overline{w_{jj}}}} \tag{4.53}$$

$$\hat{Cov}(s_{i,j}, s_{a,b}) = \frac{m}{(m-1)^3} \sum_{k=1}^{|C|} \sum_{s=1}^{|S_k|} \sum_{v=1}^{|T_k|} (w_{ksv,ij} - \overline{w_{ij}})(w_{ksv,ab} - \overline{w_{ab}}) \tag{4.54}$$

Die Schätzmethoden für die verwendeten Parameter schließen die Herleitung des Verfahrens ab. Im folgenden Abschnitt werden die Architektur und die resultierende Prozedur noch einmal zusammenfassend dargestellt.

Gesamt-Architektur des Gestenerkenners

Abbildung 4.15 illustriert den Ablauf des Erkennungsprozesses, dessen Merkmalsextraktion und einzelne Abschnitte der Klassifikation in den vorangegangenen Abschnitten vorgestellt wurden. Durch den Nutzer, zusammen mit ihren Labels, spezifizierte Gestentemplates werden genutzt, um das System zu trainieren, indem die beschriebenen Verfahren zur Extraktion der Merkmale und Schätzung der Parameter Anwendung finden. Die Klassifikation vereint die bereits beschriebenen Verfahren zur Nächste-Nachbar-Suche und der Bayes'schen Entscheidung. Die Zuweisung zu einer Interpretation beruht auf einem Vergleich der eingegebenen Gesten mit jedem Template bezüglich struktureller und zeitlicher Merkmale sowie der Gestalt. Die relativen zeitlichen und strukturellen Eigenschaften werden mittels der zu Beginn vorgestellten Diskriminanzanalyse einbezogen. Wie bereits gezeigt, wird die Form eines Tokens, obwohl durch Distanzberechnung, mit dem gleichen Werkzeug unter implizierten Annahmen behandelt. Die Kombination beider Maße wird durch die in Gleichung 4.24 gezeigte Formel vorgenommen. Durch die separate Betrachtung und den paarweisen Vergleich der Token unter Auswertung der jeweiligen PDF wird eine Matrix berechnet, welche (log-)Likelihoods der Zuordnungen der jeweiligen Token aus Template und Eingabe enthält. Die kumulierte Likelihood der Übereinstimmung einer Geste mit dem Template wird durch Lösung des durch die Matrix formulierten Zuordnungsproblems (oder Finden eines perfekten Matchings mit maximalem Gewicht bei Betrachtung als bipartiten Graphen) ermittelt.

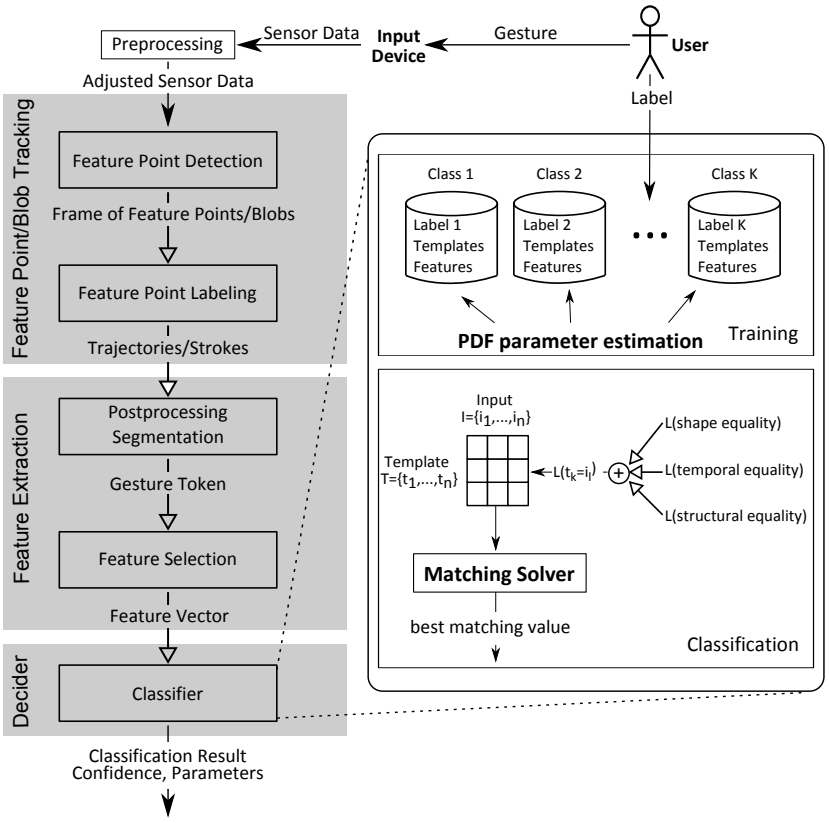


Abbildung 4.15: Die Architektur des Klassifizierers im Kontext des allgemeinen Systems der Gestenerkennung.

Entscheidend für einen paarweisen Vergleich der Token einer Eingabe mit denen aller Templates innerhalb einer Klasse ist die einmal mehr modifizierte Entscheidungsregel:

$$q(\vec{y},\vec{z}) = \arg\max_{k\in K}\max_{s\in S_k}\max_{h}\sum_{v=1}^{T} r(\vec{y},\vec{z},k,h) - \ln|\Sigma_k| \tag{4.55}$$

mit:

$$r(\vec{y},\vec{z},k,h) = -((\vec{y}_{h(v)} - \vec{y}_{ksv})^T\Sigma_k^{-1}(\vec{y}_{h(v)} - \vec{y}_{ksv})) - ||\vec{z}_{h(v)} - \vec{z}_{ksv}||_2^2 \tag{4.56}$$

Die Klassifikation einer Eingabe erfolgt durch einen Vergleich mit jedem Template jeder Klasse unter jeweiliger Berechnung der log-Likelihoods der bestmöglichen Zuordnung. Zurückgegeben wird der Index oder das Label der Klasse des Templates, für welches die Maximum-Likelihood-Zuordnung der Token den größten Wert ergab. Der Ablauf ist in Algorithmus 4.4 dargestellt.

Die eigentliche Prozedur der Klassifikation benötigt keine aufwendige Programmierung und reduziert sich größtenteils auf die Schätzung der Kovarianzen, der Berechnung von Mahalanobis-Distanzen und der Lösung des Zuordnungsproblems. Der Ausdruck $-\ln det(Cov)$ wird nur benötigt, wenn die Schätzer der Kovarianzen für verschiedene Klassen unterschiedlich sind (Variante 2). Wird die Einheitsmatrix als Schätzer verwen-

Algorithm 4.4 CompareGestures(I, T, Cov, n)

INPUT: T - Template-Geste
INPUT: I - Eingabe-Geste
INPUT: Cov - Kovariance Matrix
INPUT: n - Anzahl der Token
for all $i = 1$ to n **do**
  $\vec{y}_T^{(i)} \leftarrow$ ExtractStructuralFeatures(Token(T,i))
  $\vec{z}_T^{(i)} \leftarrow$ ExtractShapeSignature(Token(T,i))
  $\vec{y}_I^{(i)} \leftarrow$ ExtractStructuralFeatures(Token(I,i))
  $\vec{z}_I^{(i)} \leftarrow$ ExtractShapeSignature(Token(I,i))
end for
for all $i = 1$ to n **do**
  for all $j = 1$ to n **do**
    md $\leftarrow$ MahalanobisDistance($\vec{y}_T^{(i)}$,$\vec{y}_I^{(j)}$, Cov)
    sd $\leftarrow$ SquaredEuclideanDistance($\vec{z}_T^{(i)}$,$\vec{z}_I^{(j)}$)
    mm[i,j] $\leftarrow -md - sd - \ln det(Cov)$
  end for
end for
result $\leftarrow$ MaximumMatchingLikelihood(mm)

det, so reduziert sich die Mahalanobis-Distanz auf die Berechnung euklidischer Distanzen und sowohl Training als auch Klassifikation werden vereinfacht (auch zugunsten der Laufzeit). Das Verfahren hat in Abhängigkeit der maximalen Anzahl aller Templates über alle Klassen G bei maximal T Token pro Template durch die konstante Abtastung der Trajektorien und das konstante Merkmalsset eine maximale Laufzeitkomplexität von $\mathcal{O}(G \cdot T^3)$. Bei adaptiver Anpassung der Resampling-Rate R an die Länge der eingegebenen Trajektorien ist eine realistischere Abschätzung dieser Komplexität durch $\mathcal{O}(G \cdot (T^3 + T^2 \cdot R))$ gegeben. Die Berechnung der Distanzen zwischen den Token ist für praktische Anwendung ausreichend effizient möglich und zeigt gute Resultate bezüglich der Erkennungsrate. Nähere Untersuchungen des letzten Aspekts finden sich im nachfolgenden Abschnitt.

4.2.3 Evaluation der Methode und Resultate

Die Evaluation des Gestenklassifizierers erfordert ein geeignetes Gestenset. Jeder Gestentyp in der Taxonomie aus Abbildung 2.3 kommt dafür in Frage, aber sequenzielle Multi-Touch Gesten stellen die größte Herausforderung dar. Dem Wissen des Autors nach ist kein derartiges Gestenset in Gebrauch (auch kein dafür geeigneter Klassifizierer). Daher wurde ein Gestenset konstruiert. Für den Anwendungsfall wäre ein Gestenset geeignet, welches intuitiv gelernt werden kann und einfach zu benutzen ist. Das Ziel an dieser Stelle hingegen ist der Test das Klassifizierers. Das gewählte Set besteht aus diesem Grund aus einer kleinen Auswahl an Primitiven, die sich leicht in verschiedenen Aspekten, wie zeitlicher oder räumlicher Positionierung und ihrer Form, unterscheiden.

Im Set wurden nur sequenzielle Multi-Touch Gesten verwendet, die aus drei Token bestehen. Eine verschiedentliche Anzahl an Token würde die Klassifikation nur erleichtern. Prinzipiell können die Gesten beliebig komplex sein und auch der zeitliche Versatz der Token ist beliebig wählbar. Allerdings würde ein zu komplexes Gestenset nur den Nutzer in seiner Fähigkeit, diese Gesten einzugeben, testen. Dennoch sollten alle Gesten ohne langwieriges Training ausführbar sein. In Abbildung 4.16 ist das letztendlich gewählte Gestenset dargestellt.

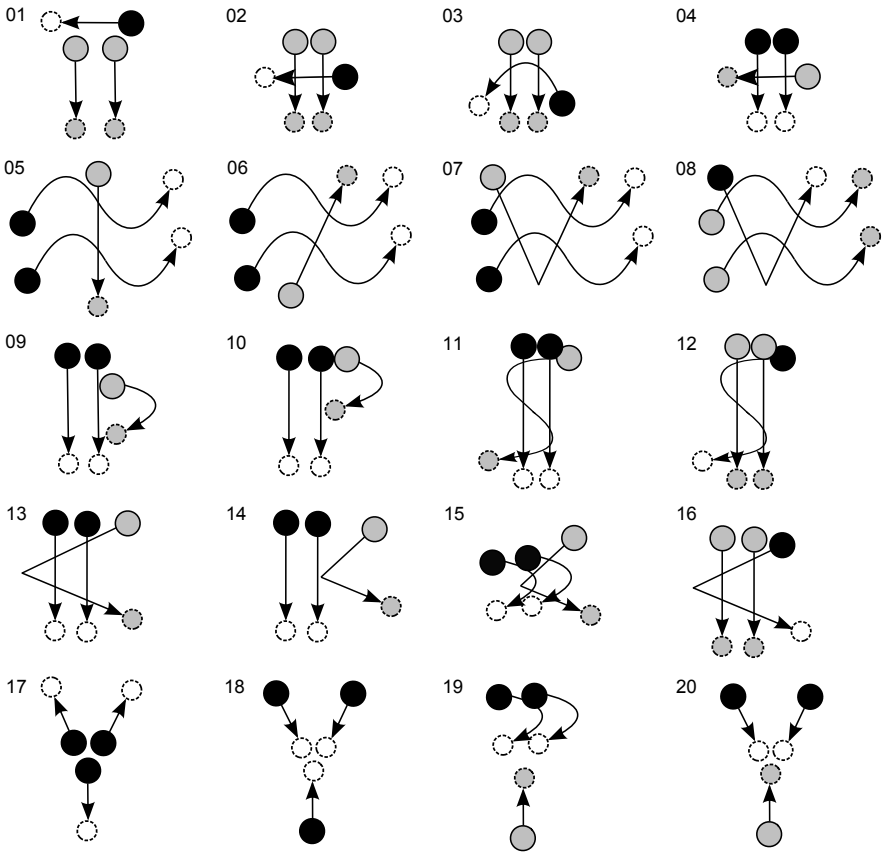


Abbildung 4.16: Das für die Evaluation gewählte Gestenset. Größere Punkte zeigen den Beginn einer Trajektorie (einer einzelnen Berührung), Pfeile den Verlauf und gestrichelte, kleinere Punkte symbolisieren ihr Ende. Verschiedene Strokes werden unterschiedlich farbig dargestellt. Schwarze Elemente gehören zum ersten Stroke und graue zum zweiten.

Das Set basiert auf den Gesten, die in [181] gezeigt wurden, und ist an mathematische Symbole angelehnt.[43] Insgesamt sechs Testpersonen spezifizierten[44] jeweils ein Set aus zehn Gesten pro Typ. Die Piktogramme aus Abbildung 4.16 dienten als Anleitung und die Teilnehmer erhielten eine kurze Beschreibung, wie diese zu deuten sind.

[43] Zu beachten ist, dass die Gestenerkennung invariant gegenüber Rotation ist und daher die Gesten einander ähnlicher sind, als sie auf den ersten Blick erscheinen mögen.

[44] Dazu wurde ein HP touchsmart tx2 Notebook genutzt. Die Eingabe der Gesten erfolgte mit den Fingern.

Im eigentlichen Test wurde jedes Set der Probanden pro Gestenklasse zufällig in gleich große Templates und Testeingaben geteilt. Für jedes Set wurde jede Testgeste gegen alle Templates klassifiziert. Diese Prozedur aus zufälliger Einteilung und Klassifikation wurde fünfmal wiederholt und die durchschnittliche Klassifikationsrate und Laufzeit ermittelt. Die Resultate sind in Tabelle 4.12 für alle Schätzer der Kovarianzmatrizen angegeben.

Tabelle 4.12: Vergleich der Klassifikation unter verschiedenen Kovarianzschätzern.

	Genauigkeit (in %)					
Testperson	**Var. 1**	**Var. 2**	**Var. 3**	**Var. 4**	**Var. 5**	**Var. 6**
1	99,39	86,06	98,59	99,39	97,78	99,60
2	99,40	92,00	99,20	99,80	99,20	99,60
3	98,60	87,40	98,00	98,00	99,00	99,40
4	100,00	94,80	99,20	99,20	99,80	100,00
5	99,60	91,60	99,80	100,00	99,00	100,00
6	99,60	97,80	99,80	99,40	99,80	99,40
⌀(%)	99,43	91,62	99,10	99,30	99,10	99,67
⌀(ms)	20,47	22,10	22,06	21,87	21,74	21,98

Die Werte in Tabelle 4.12 wurden für die Gestensets jeder Testperson (erste Spalte) ermittelt, indem wiederholt fünf zufällig gewählte Testgesten pro Klasse gegen die verbleibenden Templates klassifiziert wurden.[45]

Zeitmessungen sind gemittelte Werte für den Vergleich einer einzelnen Eingabe mit allen Templates. Unter diesen Bedingungen würde die Eingabe einer Geste bei einem Set aus 20 Gestenklassen mit jeweils fünf spezifizierten Templates in weniger als 25ms klassifiziert werden. Die Zeit für das Lernen der Parameter wurde im Voraus investiert und ist nicht in den Messungen enthalten, so dass die Laufzeiten der Klassifikation vergleichbar bleiben. Im Anwendungsfall muss das Training nur einmalig für ein neu spezifiziertes oder verändertes Template-Set ausgeführt werden.

Im Vergleich der Ansätze schneiden die auf euklidischen Distanzen basierende Methode und der 'geschrumpfte' Schätzer gemeinsamer Kovarianzen besser ab und liefern praktikable Resultate. Da das unterschiedliche Verhalten in der Laufzeit nicht relevant ist, kann Variante 6 als überlegen angesehen werden. Alle Methoden (außer Variante 2) sollten allerdings in der Praxis brauchbar sein. Variante 1 ist die am leichtesten zu implementierende Methode, welche zusätzlich nicht von genügend großen Trainingsdaten abhängig ist.[46] Die Prozedur der 'Schrumpfung' verbessert die Schätzer der Variante 2 zu vergleichbaren Resultaten wie bei der Variante mit gemeinsamer Kovarianzmatrix (Variante 3). Obwohl der vereinigte Schätzer (Variante 4) leicht bessere Resultate hervorbringt, ist Variante 5 die bessere Option, wenn mit kleinen Stichprobengrößen

[45] Während der Evaluation stellte sich heraus, dass die Eingabe eines Templates der Klasse 2 im Gestenset von Testperson 1 in falscher Reihenfolge der Strokes erfolgte. Das Template wurde nachträglich vom Set entfernt; für diesen Nutzer wurden nur 4 Testgesten aus dieser Instanz pro Testlauf genutzt.

[46] Diese Methode kann auch mit nur einem spezifizierten Template pro Gestenklasse Verwendung finden und diese Anzahl reicht in vielen Fällen aus.

gearbeitet wird und nicht auf die impliziten Annahmen einer Einheitsmatrix oder eines gemeinsamen Schätzers vertraut werden soll.

Missinterpretationen traten mit geringer Häufigkeit auf. Dennoch ist es interessant zu wissen, wann und warum die Klassifikation fehlschlägt. In Tabelle 4.13 sind detaillierte Informationen zu fehlinterpretierten Gesten unter Klassifikation mit den für praktische Anwendungen relevantesten Schätzern aus Variante 1 und Variante 6 gegeben.

Tabelle 4.13: Wahrheitsmatrix (im Engl. gebräuchlich 'confusion matrix') der auf die minimalen Fälle mit nicht-trivialen Klassifikationsergebnissen reduzierten Gesten. Verwechslungen sind für jede Testperson und jeweils zwei der verwendeten Schätzmethoden gegeben. In jeder Zelle informieren die Einträge über Fehlinterpretationen (Spalte) für eine eingegebene Geste (Zeile) durch die Angabe der Nummer des durch eine Testperson spezifizierten Gestensets für jedes Auftreten einer Fehlklassifikation. Kursiv geschrieben Zahlen gehören zum Test mit Einheitsmatrix als Kovarianzschätzer (Variante 1), fett geschriebene Zahlen zur Methode der 'Schrumpfung' aus Variante 6.

T.-Nr.	**2**	**3**	**14**	**16**	**18**	**19**	**20**
2		*2,2,2*/**2,2**					
3	*5,5*/-			*3,3*/-			
12	*3*/**3**						
15			*3,3,3*/**3,3**				
18							*1,6,6*/**1,6,6,6**
20					-/**1**	*1,1,3*/-	

Die Zeilenköpfe in Tabelle 4.13 stehen für eine tatsächlich eingegebene Geste, die Spaltenköpfe jeweils für eine mögliche Fehlinterpretation. Ausgefüllte Zellen weisen einer Eingabe aufgetretene Fehlinterpretationen zu. Die Zahlen in der Zelle stehen dabei für ein Gestenset (eines Nutzers), wiederholte Nennungen repräsentieren die Anzahl der Fehlinterpretationen für dieses Set. Sind die Zahlen kursiv geschrieben, stehen sie für Fälle, die unter Variante 1 auftraten. Fett geschriebene Werte gehören dementsprechend zum Test unter der Variante 6. Beispielsweise wurde die Geste 15 im Set von Testperson 3 dreimal unter Variante 1 und zweimal unter Variante 6 als Geste 14 fehlinterpretiert.

Insgesamt traten Fehlinterpretationen für sechs Gestenklassen in Variante 1 und fünf Klassen in Variante 6 auf. Unter Variante 1 erzielte Geste 3 die schlechteste Rate der richtig Positiven (Sensitivität: $\frac{146}{150} = 97,33\%$), aber die Gesten 2, 15, 18 und 20 lieferten ähnliche Raten. Die schlechteste Rate der richtig Negativen wurde für Variante 1 durch die Gesten 3, 14 und 20 (jeweils mit Spezifität: $\frac{2842}{2845} = 99,89\%$) erreicht. Unter Variante 6 trat die schlechteste Rate der richtig Positiven für die Geste 18 (Sensitivität: $\frac{146}{150} = 97,33\%$) auf und die schlechteste Rate der richtig Negativen verursachte Geste 20 (Spezifität: $\frac{2841}{2845} = 99,86\%$). Für die Gestenklassen 18 und 20 trat die einzige gegenseitige Verwechslung innerhalb des Gestensets von Testperson 1 auf. Bei dennoch guten Resultaten kann der kritischste Fall hier folglich für die Gestenklasse 20 gesehen werden, während es unter Variante 1 die Gestenklasse 3 ist. Bemerkenswert ist, dass Fehlinterpretationen vom Gestenset der Nutzer abhängig zu sein scheinen und weder gleich über die Nutzer, noch über die Gestenklassen verteilt sind.

Eine Überprüfung der Verteilungsannahme der lokalen Merkmale aus Abschnitt 4.2.2 soll exemplarisch anhand der (über alle Nutzer) gesammelten Testdaten von Geste 10 des unter Abbildung 4.16 gegebenen Gestensets vorgenommen werden. Für andere Gesten zeichnete sich ein ähnliches Bild. In Abbildung 4.17 sind QQ-Plots für Normalverteilungen sowie Histogramme der Distanzmerkmal-Verteilung innerhalb der durch die Benutzer akquirierten Gestendaten gegeben.

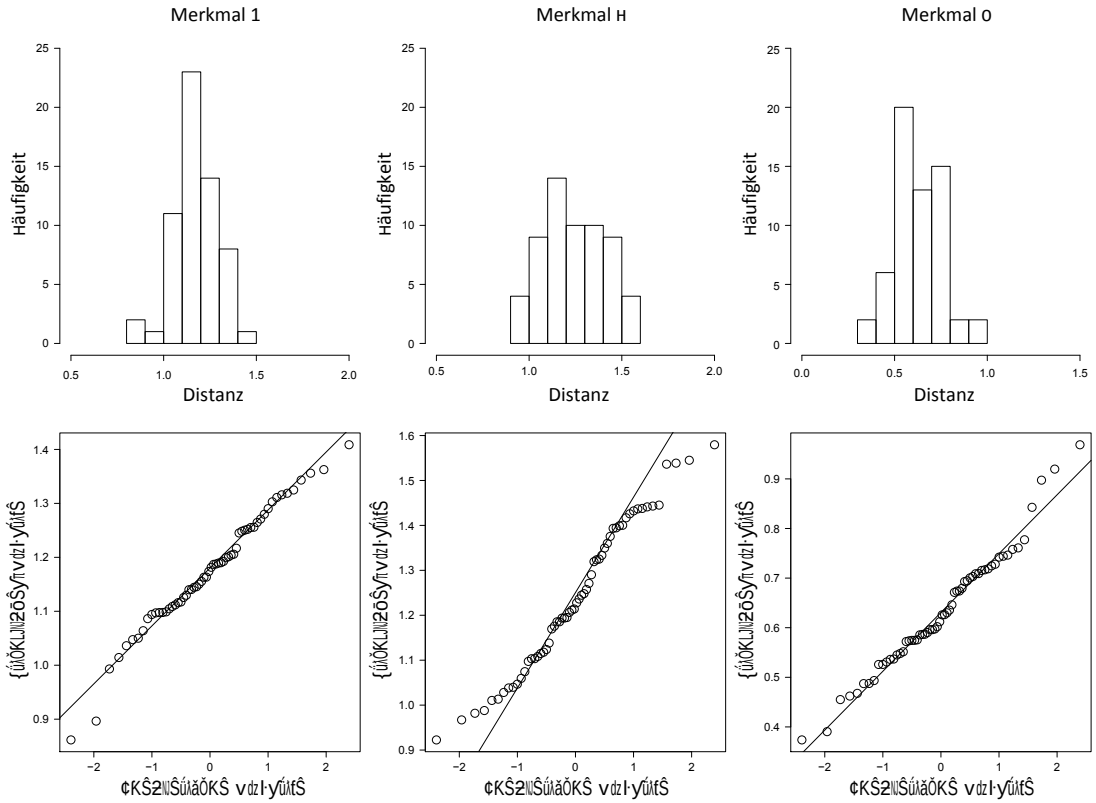


Abbildung 4.17: Histogramme (oben) und QQ-Plots (unten) für die Verteilung der Merkmale 1-3 (von links nach rechts) aus Abbildung 4.13 des zeitlich letzten Tokens der Geste 10.

Die Merkmale 1 und 3 können anhand der 60 Testdaten mit Hilfe der Darstellungen in Abbildung 4.16 als annähernd normalverteilt angesehen werden. Dies wird auch durch Shapiro-Wilk Tests auf Normalverteilung bestätigt ($p = 0,48$ für Merkmal 1 und $p = 0,32$ für Merkmal 3). Für Merkmal 2 zeigt der Shapiro-Wilk Test zwar ebenfalls kein signifikantes Ergebnis zu Gunsten einer Ablehnung der Normalverteilungsannahme ($p = 0,16$), der entsprechende QQ-Plot weist aber ziemlich starke Abweichungen an den Rändern der Verteilung auf. Eine Betrachtung der Testperson-spezifischen Daten mittels des gleichen Testverfahrens ($p = \{0,17; 0,87; 0,42; 0,37; 0,83; 0,80\}$) deutet an, dass die Verteilungen näher an eine Normalverteilung herankommen, aber zwischen den Benutzern stärker variieren.

Trotz der Eignung des eher strengen Shapiro-Wilk Tests[47] auch für sehr kleine Stich-

[47] In einem Vergleich verschiedener Testverfahren anhand zufallsgenerierter Daten in [186] zeigte sich der Shapiro-Wilk Test durchschnittlich zuverlässig bezüglich der Wahrscheinlichkeit, die Null-

proben und keiner Ablehnung der Normalverteilungsannahme, ist für eine bessere visuelle Abschätzung eine größere Datenmenge je Teilnehmer nötig. Dennoch ist es plausibel, dass sich die unterschiedlichen Eingabestile in ähnlichen Verteilungen mit abweichenden Parametern widerspiegeln und eine Zusammenlegung der Daten bezüglich eines Merkmals weiter von einer univariaten Normalverteilungen abweicht als die einzelnen Datensets.

In Abbildung 4.18 sind sowohl Histogramm und QQ-Plot für das zeitliche Merkmal ($p = 0,53$ unter Shapiro-Wilk Test) als auch, repräsentativ für die Winkel-Merkmale 4 und 5, zirkulare Histogramme sowie ebenfalls QQ-Plots angegeben. Als Datenbasis dienten abermals exemplarisch alle gesammelten Templates der Geste 10.

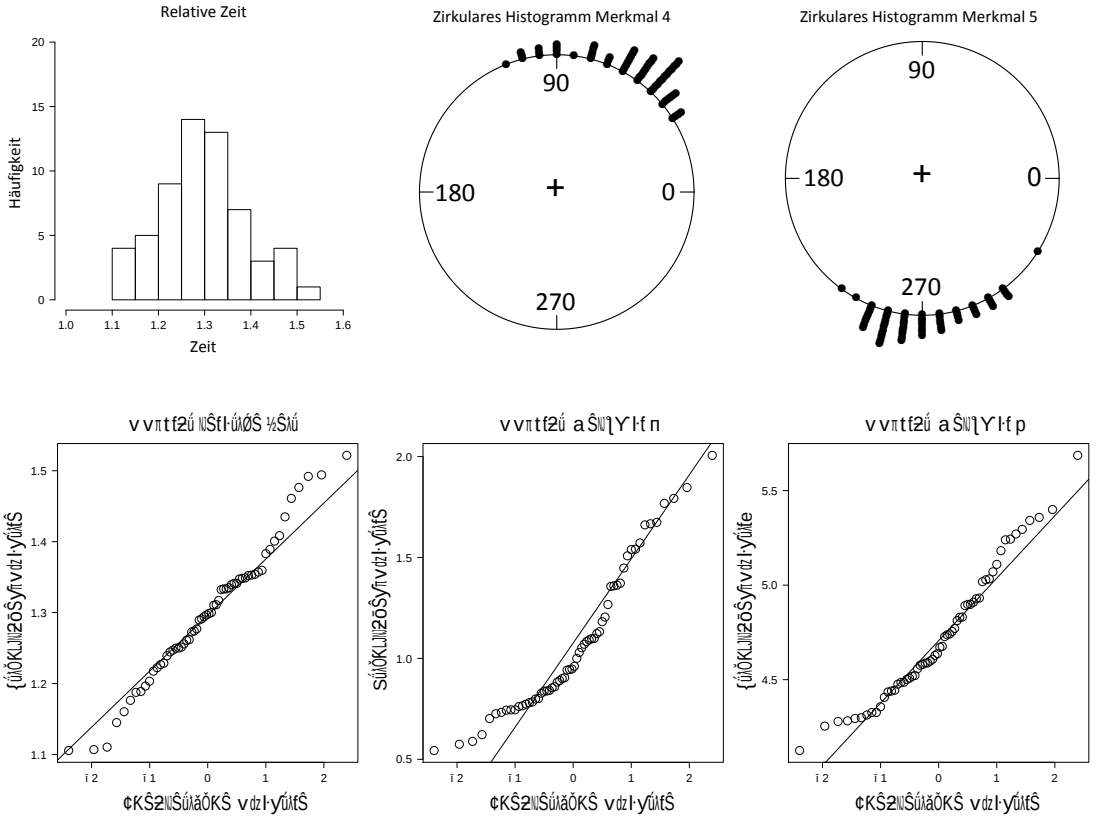


Abbildung 4.18: Histogramme (oben) und QQ-Plots (unten) für die Verteilung des zeitlichen Merkmals (links) sowie der Merkmale 4 und 5 (mittig und rechts) für den zeitlich letzten Token der Geste 10.

Die Datenpunkte in den Histogrammen der zirkularen Daten für Merkmal 4 stammen nach einem Kolmogorov-Smirnov Anpassungstest ($p = 0,19$) mit hoher Wahrscheinlichkeit aus einer von Mises-Verteilung ($\mu = 1,07, \kappa = 8,26$), einer approximativen Beschreibung der zirkularen Normalverteilung. Für Merkmal 5 lehnt der gleiche Test ($p = 0,53$) die Verteilungsannahme ($\mu = -1,56, \kappa = 8,90$) ebenfalls nicht ab. Nach den bisherigen Untersuchungen kann davon ausgegangen werden, dass die ge-

Hypothese abzulehnen, wenn die Alternativ-Hypothese wahr ist (hohe Teststärke, d.h. die Wahrscheinlichkeit, keinen Fehler vom Typ 2 zu begehen).

wählten Merkmale der Annahme einer näherungsweisen Normalverteilung entsprechen, die sich allerdings zwischen den Nutzern unterscheidet.

Es sei an dieser Stelle noch erwähnt, dass die Normalverteilung einzelner Merkmale eine mehrdimensionale Normalverteilung nur zur Folge hat, wenn diese unabhängig sind (beispielsweise im Falle der Korrektheit der Annahme einer multivariaten Standardnormalverteilung). Streng genommen müsste eine Überprüfung auf multivariate Normalverteilung stattfinden. An dieser Stelle soll allerdings nicht tiefer in derartige Tests eingestiegen werden, da das Anliegen die Untersuchung der zugrunde gelegten Vermutung zur Merkmalsselektion war. Die im Klassifikationsverfahren (je nach Schätzer) vorausgesetzten multivariaten Normalverteilungen oder die Robustheit gegenüber deren Verletzungen werden ausreichend empirisch durch die Klassifikationstests belegt.

Insgesamt scheint der Klassifikator robust gegenüber Verletzungen der Annahme eines möglichst normalverteilten Merkmalssets zu sein. Um mehr Informationen diesbezüglich zu erfahren, wurde in einem zweiten Test in einer zufälligen Auswahl jeweils ein Template pro Gestenklasse aus jedem Set als Trainingsinstanz gewählt. Alle verbleibenden Gesten aller Nutzer fungierten als Testinstanzen. Für jeden Durchlauf standen demnach sechs (eines pro Nutzer) Templates pro Klasse für das Training zur Verfügung. Die Resultate über die verbleibenden 5395 Testinstanzen über fünf Läufe verschlechterten sich leicht, sind aber dennoch vielversprechend. Für Variante 1 wurde eine Genauigkeit von 98,37% erreicht. Abermals erzielte Variante 6 ähnliche, aber leicht bessere Resultate (98,65%).

Als letzter Aspekt sei im Folgenden noch die Möglichkeit der Reduktion des Merkmalssets untersucht. Zwar ist die Zahl der verwendeten Merkmale nicht groß, hinsichtlich der Skalierbarkeit ist die Betrachtung dennoch sinnvoll. Derartige Ansätze können zudem auch die Performance eines Klassifikationsverfahrens verbessern. Da durch das kleine Merkmalsset nur eine unwesentliche Ersparnis in der Laufzeit zu erwarten ist, soll diese Möglichkeit nur im Hinblick auf dessen Ergänzungen erwähnt werden. An dieser Stelle erfolgt demnach nur eine kleine Untersuchung bezüglich des Einflusses auf die Robustheit des Verfahrens.

4.2.4 Merkmalsreduktion

Der konstruktive Ansatz der Merkmalsgewinnung lässt Korrelationen in den Merkmalen vermuten und die Verbesserung der Klassifikationsrate bei Verwendung eines Parameterschätzers für Kovarianzen deutet ebenfalls darauf hin. Diese Korrelationen können mittels der Hauptkomponentenanalyse[48] (engl. principal component analysis, PCA) auch für die Reduktion des Merkmalssets durch Merkmalstransformation genutzt werden, um den 'curse of dimensionality' abzuschwächen. Die insgesamt 12 verwendeten Merkmale, die neben der Shape-Signatur zur Klassifikation herangezogen werden, sind in ihrer Anzahl noch als gering anzusehen. Allerdings kann eine solche Merkmalsre-

[48] Für eine tiefere Betrachtung siehe auch [88].

duktion bei korrelierten Daten auch die Robustheit der Klassifikation erhöhen [144]. Auch im Hinblick auf mögliche Erweiterungen des Merkmalssets soll an dieser Stelle die Zweckmäßigkeit der Anwendung einer PCA untersucht werden.

In Abbildung 4.19 sind, abermals für alle akquirierten Templates der Geste 10, die Streudiagramme der Merkmale des zeitlich letzten Tokens (aufgrund seiner eindeutig möglichen Identifikation durch die zeitliche Relation) aufgeführt.

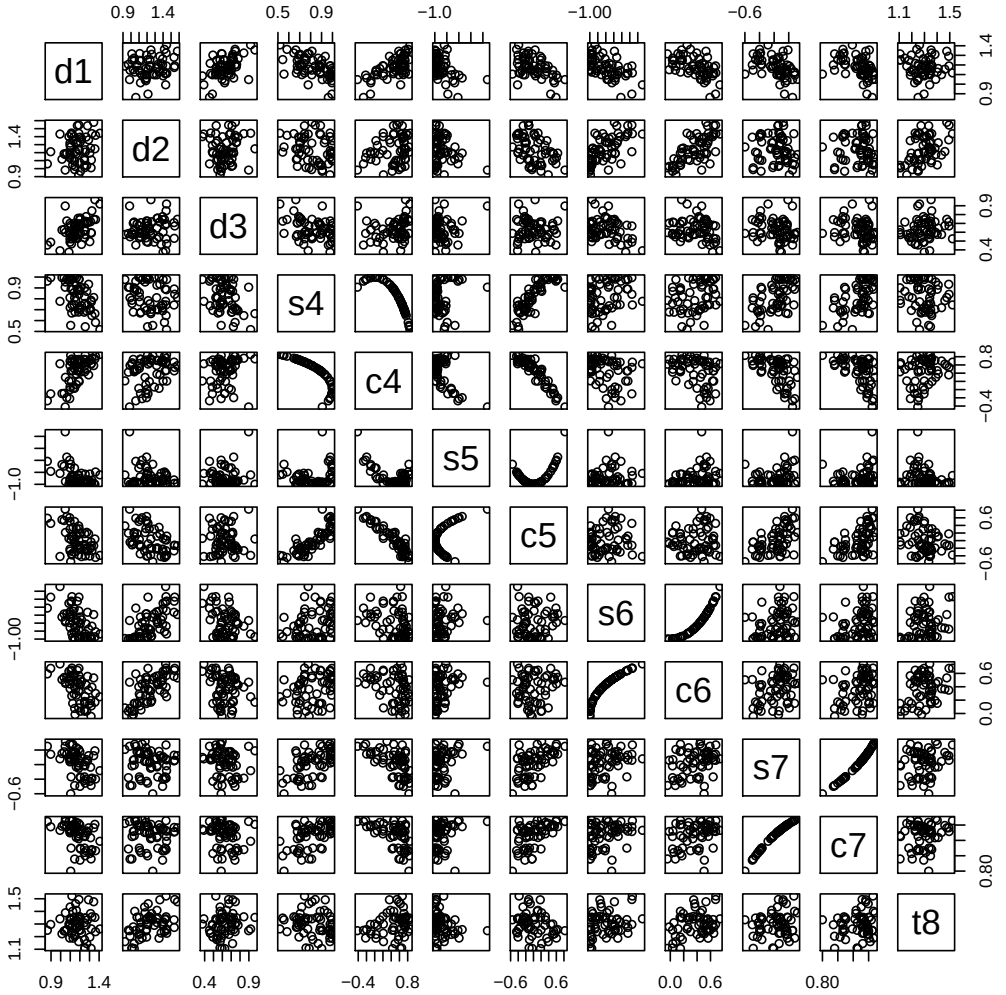


Abbildung 4.19: Matrix der Streudiagramme der Merkmale für den letzten Token der Geste 10 aus den Daten aller Nutzer. Die Merkmale sind die in Abschnitt 4.2.2 beschriebenen drei Distanzmaße (d1-d3), jeweils der Sinus (s) und der Kosinus (c) der vier Winkelmaße (4-7) und der zeitliche Offset (t8) des Tokens.

Neben den erwarteten Korrelationen[49] sind Zusammenhänge auch zwischen den Merkmalen 1 und 6 sowie 4 und 5 angedeutet. Durch die PCA werden dekorrelierte Linearkombinationen der Merkmale erzeugt, die (unter Gültigkeit der Annahme multivariater Normalverteilung der Merkmale) statistisch unabhängig voneinander sind [98].

[49] Für den Sinus und den Kosinus von gleichen Winkel-Merkmalen sind starke Korrelationen zu erwarten.

Hier erfolgte die Implementierung nach [195] mittels des Accord.NET Frameworks[50]. Dabei werden die Komponenten (Linearkombinationen der Merkmale) der durch Eigenwertzerlegung der Kovarianzmatrix (Schätzer aus Variante 6) erfolgten PCA genutzt, welche die höchsten Eigenwerte λ_i aufweisen und demzufolge den Großteil der Varianz im neuen Merkmalsraum erklären, wie in Gleichung 4.57 aus [144] wiedergegeben.

$$R = \frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \tag{4.57}$$

Die zwölf originalen Merkmale können unter Verwendung von $R \geq 99\%$ auf 10 und unter $R \geq 95\%$ auf sechs Komponenten reduziert werden. Unter der Annahme ursprünglicher multivariater Normalverteilung der Merkmale kann die Unabhängigkeit der Komponenten zudem durch die Verwendung euklidischer Distanzen im Klassifikator genutzt werden. Nachteilig an diesem Ansatz ist nach [151], dass er nur Varianzen in den Daten berücksichtigt, aber keine Information über die Zuordnung zu den Klassen. Eine kleine Auswahl alternativer Herangehensweisen, die auch den Beitrag zur Klassenzuweisung einbeziehen, wird ebenda vorgestellt. Verschiedene Methoden zur Festlegung der Anzahl der Komponenten finden sich auch in [88].

Unter abermals zufällig gewählten Templates und Testsets wurden bei der Verwendung aller Komponenten jeweils 599 von 600 = 99,83% der Gesten sowohl für Variante 6 als auch Variante 1 korrekt klassifiziert. Die gleichen Resultate wurden auch erreicht, wenn die 'geschrumpfte' Kovarianzmatrix für die Merkmale über jeweils eine Klasse (Variante 5) und alle Merkmale (Variante 6) für die Reduktion auf $R \geq 95\%$ bzw. $R \geq 99\%$ der Varianz genutzt wurde. Die Klassifikation über dem reduzierten Merkmalsset erfolgte mittels der Einheitsmatrix. Im vorliegenden Fall kann demzufolge die PCA als eine Möglichkeit gesehen werden, einen kleinen Performance Gewinn zu erzielen. Ob und inwieweit auch die Klassifikation verbessert werden kann, sollte an einem Gestenset getestet werden, bei dem die Standardprozedur nicht schon eine so hohe Klassifikationsgenauigkeit erreicht.

4.2.5 Zusammenfassung und Diskussion

Im zurückliegenden Kapitel wurde das Schema für eine Klassifikation beliebiger planarer Gesten präsentiert. Es wurde empirisch bewiesen, dass die Methode für einen flexiblen Template-basierten Multi-Touch Gestenerkenner geeignet ist. Auch die Anforderungen an die Laufzeit konnten für ein realistisches Gestenset eingehalten werden. Die Formalisierung und Herleitung des Bayes'schen Entscheidungsprozesses wurden demonstriert. Mehrere Schätzmethoden für Parameter wurden untersucht, bei denen sich die Annahme einer Einheitsmatrix als Kovarianzmatrix als ausreichend gut erwiesen hat. Ausgeklügeltere Schätzmethoden über 'Schrumpfung' können die Resultate noch verbessern und in der Praxis ist eine Kombination beider Methoden - abhängig von der

[50] C. R. Souza, "The Accord.NET Framework", Apr 2012; `http://accord.googlecode.com`

Größe des Template-Sets - vermutlich die beste Wahl. Das Ziel war die Entwicklung eines Klassifizierers, der in Echtzeit und nur durch Spezifikation per Templates alle Arten planarer Multi-Touch Gesten erkennen kann. Die möglichen Anwendungen bleiben aber nicht auf Gesten beschränkt.

Die Zerlegung einer Gesteneingabe in Token und die Extraktion lokaler Merkmale ist die Besonderheit des vorgestellten Verfahrens und erlaubt eine Unterscheidung ähnlicher Gesten. Das dadurch entstandene Problem, paarweise Zuordnungen der Token aus Eingabe und Template zu finden, wird durch ein Maximum-Likelihood-Matching gelöst. Dieser Ansatz kann auch als ein auf breitere Anwendungen übertragbarer Sensor-Fusion Prozess gesehen werden, bei dem verschiedene Sensoren Werte liefern, aber die Identifikation der Sensoren nicht bekannt ist.

Der Ansatz ist naheliegend und lässt auch komplexere Segmentierungen zu. In [81] beispielsweise wird die Segmentierung eingegebener Skizzen in zwei Arten von Primitiven vorgenommen. Dabei wird mittels dynamischer Programmierung nach der Fehlerrate der Klassifikation optimiert. Denkbar wäre auch hier die weitere Zerlegung der Trajektorien. Eine weitere Eigenschaft des Segmentierungsprozesses ist die Abhängigkeit der Klassifikation von einer Ordnung der Token. Der Autor dieser Arbeit ist der Ansicht, dass die Information über den relativen Zeitpunkt eines Teils der Eingabe essenziell bei der Behandlung von Gesten ist. Soll die Methode aber beispielsweise auf Skizzen angewendet werden, so kann es sinnvoll sein, die Reihenfolge der Token zu ignorieren. In diesem Fall wäre nur das zeitliche Merkmal zu vernachlässigen. Sind hingegen nur bestimmte, verschiedene Variationen diesbezüglich zuzulassen, stellt das Anlegen mehrerer Gestenklassen, die die verschiedenen Varianten repräsentieren und deren erfolgreiche Erkennung an die gleiche Aktion geknüpft ist, eine Möglichkeit dar.

Ein bisher nicht diskutierter Aspekt ist die Ablehnung einer Entscheidung. In vielen Fällen ist es vorteilhafter, bei genügender Unsicherheit keine Aktionen auszuführen. Die Bayes'sche Entscheidung selbst kann über die höchste a-posteriori-Wahrscheinlichkeit abgelehnt werden, wenn diese kleiner als $1-\epsilon$ für einen festgelegten Grenzwert ϵ ist [177, S. 9,10]. Die vorgestellten Modifikationen ändern an diesem Konzept nichts. Ablehnungen können über einen Grenzwert definiert werden, bei dem das Risiko einer Fehlentscheidung nach empirisch ermittelten Informationen zu groß ist. Ein solcher Grenzwert sollte allerdings im Anwendungskontext und eventuell klassenspezifisch festgelegt werden.

Selbst definierte Gesten erlauben bessere Individualisierbarkeit und kann Nutzern mit speziellen Bedürfnissen entgegenkommen. Allerdings steht die Entwicklung geeigneter Gestensets und die einer Software, welche solche Optionen der Eingabe sinnvoll umsetzt, noch aus. Mögliche Anwendungen werden im Kapitel 6 vorgestellt. Zunächst wird allerdings das vorgestellte Verfahren mit dem Ziel erweitert, auch partielle Eingaben zu klassifizieren.

5

Autovervollständigung Planarer Gesten

In diesem Kapitel soll die Problemstellung bedacht werden, wie eine Erkennung von Gesten schon im Verlauf ihrer Eingabe möglich ist. Im besten Fall steht begleitend während der Eingabe eine kontinuierlich adaptierende Interpretation zur Verfügung, anhand derer eine Abwägung nach der Wahrscheinlichkeit einer beabsichtigten Geste abgeleitet werden kann. Neben der Möglichkeit, Multi-Touch einzusetzen und damit in kürzeren Gesten mehr Information zu codieren, kann die Interaktionszeit auch bedeutend verkürzt werden, indem der Nutzer die Eingabe beenden kann, sobald genug Information für deren Erkennung vorhanden ist. Der Einsatz dieser vorausschauenden Erkennung ermöglicht ebenso die Auflösung der Grenzen zwischen gestischer Interaktion und direkter Manipulation. Aber auch den Nutzer bei der Eingabe unterstützende Hilfestellungen oder dynamische Lernhilfen für Gesten werden möglich.

Freeman et al. [59] stellen fest, dass die Notwendigkeit des Lernens komplexer physischer Eingaben eine Barriere für Nutzer darstellt. Dies wird als Grund dafür gesehen, dass sich Hersteller kommerzieller Systeme scheuen, Multi-Touch Interaktionen zu implementieren, die über die einfachen direkten Manipulationen nach Shneiderman [192] hinausgehen. Arbeiten, welche sich mit partiellen Eingaben von Gesten beschäftigen, haben demzufolge oft das Erlernen von Gesten oder die Unterstützung bei der Eingabe zum Ziel. Die eigentliche Methodik hinter der Erkennung basiert meist auf praktikabler Anpassung bestehender Verfahren, um das Konzept der Hilfestellung zu demonstrieren bzw. dessen Nützlichkeit nachweisen zu können.

5.1 Stand der Technik

In [169] soll der als 'Eager Recognition' bezeichnete Ansatz einen Übergang einer Geste zu einer Parameter-vermittelnden, direkten Manipulation ermöglichen. Für diesen Zweck werden pro Klasse zwei Gestensets gebildet, eines für vollständige Gesten (sowie Teilgesten, die 'vollständig genug' sind) und eines für unvollständige (Sub- bzw. Teil-) Gesten. Die Einteilung erfolgt, indem die Teilgesten in absteigender Länge klassifiziert und ab der ersten Fehlklassifikation in das Set der unvollständigen Gesten eingefügt werden. Bei korrekter Klassifikation wird das Set der vollständigen Gesten, ebenfalls jeweils für die entsprechende Klasse, ergänzt. In einem Korrekturschritt werden Gesten aus dem Set der vollständigen Gesten in ein Set der unvollständigen verschoben, wenn ihre Mahalanobis-Distanz des Merkmalsvektors zum Mittelwert des Sets einen dynamisch ermittelten Schwellwert überschreitet. Aus diesen Daten wird unter Feinjustierung der Parameter per Kreuzvalidierung ein binärer Klassifizierer trainiert, der entscheidet, ob von einer aktuellen Eingabe genug gesehen wurde (wenn sie dem Set der vollständigen Gesten zugeordnet wird), um sie dem Standard-Klassifizierer zu übergeben. Die Klassifikation für Single-Touch Gesten erfolgt dabei zu jedem Abtastzeitpunkt der Eingabe [170].

Die Methode der Gestenklassifikation sowie der -vorhersage aus [169] wird beispielsweise in [77] benutzt, um handgezeichnete Skizzen von ER-Diagrammen (4 einfache geometrische Symbole für Entitäten, Relationen und Attribute) zu erkennen. Anhand des Ortes der Eingabe entscheidet die Anwendung, ob ein Klassifizierer für Skizzen und Gesten (dabei wird keine Unterscheidung zwischen Skizzen und Gesten vorgenommen) oder für gestische Befehle (Kringel bzw. Kreuz für Verschieben oder Löschen) gewählt wird.

Eine ähnliche Methode wird in [202] zur Erkennung von Skizzen umgesetzt. Anhand ihrer Strokes zerlegte Skizzen werden den Trainingsdaten hinzugefügt, was die Erkennung auf partielle Eingaben mit komplettierten Strokes beschränkt. Die Trainingsdaten werden mit Hilfe visueller Merkmale nach [147] geclustert. Bleibt ein Cluster dabei heterogen, enthält also Skizzen verschiedener Klassen, wird ein mittels überwachtem Lernen trainierter SVM-Klassifizierer hinzugezogen um die Klassen innerhalb eines Clusters zu unterscheiden.

Für ihr Feedback-System 'Octopocus' zur Unterstützung von Single-Touch Gesteneingaben mit Vorschlägen für mögliche weitere Verläufe entfernen Bau und Mackay [13] jeweils vom Anfang eines Templates je Klasse ein Stück der Länge der aktuellen Gesteneingabe und fügen stattdessen die Nutzereingabe an. Danach wird die so modifizierte Geste dem Erkenner unter Verwendung des originalen Template-Sets übergeben. Die Erkennung arbeitet basierend auf der Arbeit in [169] mit Mahalanobis-Distanzen. Die Autoren geben an, dass alternativ auch mit einem Distanzmaß zwischen Shape-Signaturen aus Winkelverläufen klassifiziert werden kann, gehen auf diesen Ansatz allerdings nicht weiter ein. Der Abstand zu einem Template einer Klasse dient als Fehler-

maß der Eingabe und wird in dem System visuell repräsentiert (Strichstärke), um nach ihrer Wahrscheinlichkeit gewichtete weiterführende Gesteneingaben anzuzeigen (siehe Abbildung 5.1 links).

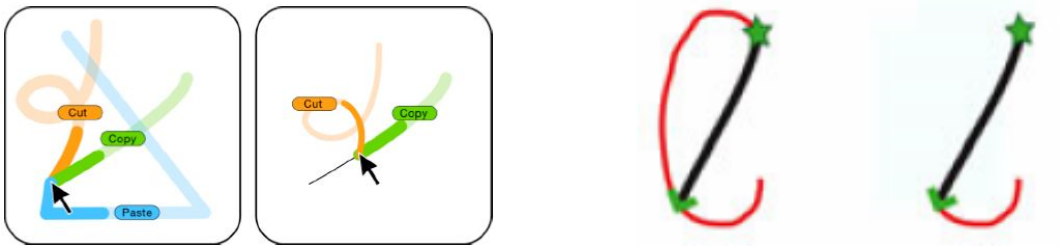


Abbildung 5.1: Auf der linken Seite ist die grafische Präsentation dreier Kommandos durch Gesten mittels 'Octopocus' [13] dargestellt. Wird eine der Gesten, in dem Fall 'copy', weiter verfolgt, werden die Darstellungen unwahrscheinlicher Prognosen dünner. Die rechte Seite zeigt bei zwei möglichen Präsentationen (vollständige prognostizierte Geste gegenüber nur der erwarteten zukünftigen Eingabe) mittels 'SimpleFlow' [16] die Auswirkungen einer fehlerhaften Interpretation als ein 'C' beim Versuch, ein Dreieck (Startpunkt ist der Stern) einzugeben.

Die Templates der Gesten werden in der Größe standardisiert, womit eine größenunabhängige Erkennung nicht möglich ist. Ebenso wird aufgrund des verwendeten Klassifikationsansatzes auf Rotationsinvarianz verzichtet.

Ein zu 'Octopocus' ähnlicher Ansatz für die Unterstützung von Single-Touch Gesteneingaben wird in [16] vorgestellt. Allerdings wird die Hilfestellung hier auf die Anzeige der wahrscheinlichsten angestrebten Eingabe beschränkt, um Nutzer nicht zu überfordern (siehe Abbildung 5.1 rechts). Auch das Vorgehen für die frühzeitige Erkennung der Gesten ist sehr ähnlich zu dem von [13]. Die Prädiktion von Gesten geschieht allerdings, indem Templates auf die Bounding-Box der aktuellen Geste skaliert werden. Danach wird ebenfalls ein Abschnitt der Länge der angefangenen Geste in den skalierten Templates durch die begonnene Geste ersetzt und die entstehende Geste zur Klassifikation gegen die ursprünglichen Templates verwendet. In [86] wird für die Umsetzung einer schnellen gestischen Texteingabe das Auftreten ungünstiger Skalierungen auf partielle Eingaben und damit verbundener Verzerrungen der Templates abgeschwächt, indem die geeignetere Dimension der Bounding-Box für die Skalierung herangezogen wird. Diese Dimension ist diejenige, die während einer Eingabe als erstes die volle Ausdehnung erreicht. Skaliert wird dann jeweils gleichförmig bezüglich der aktuellen Größe in dieser Dimension.

Eine präzisere Abschätzung der Größe einer unvollständig eingegebenen Geste wird in [8] versucht. Dazu wird eine skalierungsunabhängige Repräsentation aus quantisierten absoluten Winkelverläufen genutzt. Sequenzen ähnlicher Winkel werden zusammengefasst, so wie beispielsweise im Ansatz in [193]. Damit wird eine Unabhängigkeit von der Geschwindigkeit der Eingabe erreicht, was unter üblicherweise gleichabständigem Resampling für einen Vergleich der Shape-Signaturen das Wissen der Länger der kom-

pletten Eingabe benötigen würde. Verglichen wird eine Eingabe über den paarweisen Abstand ihrer Shape-Signatur zum Präfix (Sequenz gleicher Anzahl Winkel) eines Templates. Die Differenz definiert sich über den Anteil der Winkel-Paare, deren Differenz einen vorgegebenen Schwellwert ($\pi/4$) überschreitet. Ist dieser Anteil kleiner 10%, wird anhand der mittleren Länge der von den betrachteten Winkel-Paaren repräsentierten Segmente[51] ein Skalierungsfaktor für die Eingabe berechnet. Tests zeigten, dass die Skalierung um diesen Faktor die eigentliche Größe der Geste im Durchschnitt um etwa 1/3 überschätzt. Durch die gewählte Shape-Signatur ist eine Rotationsinvarianz nicht gegeben.

In [137] wird der DTW Ansatz genutzt, um partielle Eingaben planarer Gesten im drei-dimensionalen Raum zu erkennen. Dazu wird die Möglichkeit der zeitlichen Invarianz aufgegeben und der Vergleich mit dem Template nur anteilig der benötigten Zeit der aktuellen Eingabe vorgenommen. Weiterhin wird ein Gesten-Netzwerk erstellt, mit welchem ähnlich dem Präfixbaum in [164, 86] die Modellierung gemeinsamer Teilgesten und die formale Darstellung des Potenzials der Prädiktion möglich ist. Es kann damit ein Zeitpunkt in der Eingabe gefunden werden, ab welchem eine Klassifikation sinnvoll ist. In [137] wird anhand der im Netzwerk definierten Primitive eine vorläufige Vorhersage der weiteren Bewegung aus der Mittelung der im Graphen auf die Eingabe folgenden Trajektorien vorgenommen. Allerdings ist sowohl in [86] als auch in [137] für die Darstellung der gemeinsamen Teilsequenzen von Gesten eine manuelle Analyse des Gestensets nötig. Die Erkennung ist den Autoren nach meist schon nach weniger als der Hälfte der Eingabe möglich. In [93] wird das Konzept aus [137] erweitert, um Eingaben behandeln zu können, die sich in der Ausführungszeit stark zu den (per SOM quantisierten) Templates unterscheiden. Allerdings werden dazu sämtliche Teilsequenzen eines Templates per euklidischer Distanz mit der Eingabe verglichen, um das ähnlichste Segment auszuwählen, sobald es sich zum zweitähnlichsten um ein Mindestmaß unterscheidet.

Die Klassifikation mittels DTW kann ebenfalls durch Lockerung der Nebenbedingungen so angepasst werden, dass auch partielle Übereinstimmungen zugelassen oder begünstigt werden [65].

In [214] werden 'fuzzy state'-Sequenzen aus der Segmentierung eines Prototypen, welcher aus der Verteilung flächig projizierter Trajektorien der Templates einer Klasse um eine 'Principal Curve' erzeugt wird, gewonnen. Zeitliche Informationen werden dadurch zunächst vernachlässigt, aber in einem Korrekturschritt zur Aufhebung von Mehrdeutigkeiten (etwa bei Überschneidungen) einbezogen. Über die Schätzung der Parameter der jeweils einem Zustand zugeordneten Gauß-verteilten Segmente der Templates kann für die Punkte einer Eingabe die Likelihood ihrer Zugehörigkeit zu einzelnen Zuständen ermittelt werden. Durch dynamische Programmierung werden komplette Eingaben somit dem ähnlichsten Prototypen zugeordnet. Der Ansatz erlaubt laut den Autoren auch die Erkennung partieller Gesten und die Detektion von Gesten in

[51] Da nicht abgeschätzt werden kann, ob das letzte Segment eine nicht abgeschlossene gerade Linie ist, wird dieses nicht mit in die Prozedur einbezogen.

fortlaufenden Eingaben. Allerdings wurden nur zwei Klassen pro Test genutzt und die Segmentierung der Testgesten zumindest teilweise manuell unterstützt.

Weitere mögliche Ansätze lassen sich aus anderen Anwendungsgebieten übertragen. So kann die Klassifikation per HMM durch Verknüpfung der Modelle [222] oder Modifikation des Viterbi-Verfahrens [45] erweitert werden, um Gesten im kontinuierlichen Eingabestrom zu erkennen. Derartiges 'Gesture Spotting' ließe sich übertragen, um Teilgesten in Templates ausfindig zu machen.

An dieser Stelle soll, aufbauend auf der Klassifikation des vorangehenden Kapitels, eine Methode entwickelt werden, mit der eine Erkennung von Gesten schon während der Eingabe vorgenommen werden kann. Die Anforderungen der Invarianzen bezüglich Translation, Skalierung, Rotation und Geschwindigkeit sollen dabei weiterhin aufrecht erhalten werden. Die vorgestellten Verfahren lassen sich damit nicht übertragen, zumal auch eine Erkennung sequenzieller Multi-Touch Gesten nicht unterstützt wird. Die Methoden in [169, 8, 202] sind durch die gewählten Merkmale (absolute Winkel) inhärent variant gegenüber Rotation. Der Ansatz aus [202] verwendet zudem (unter gewisser Toleranz) positionale Merkmale nach [147]. Bennett et al. [16] skalieren Templates auf das aktuelle, umschließende Rechteck der unvollständigen Eingabe, was die Gesten verzerrt und eine Erweiterung für Rotationsinvarianz erschwert. Bau und Mackay [13] erwarten bezüglich der Größe standardisierte Eingaben und geben zudem die Invarianz gegenüber Rotation durch die Verwendung des Klassifizierers von Rubine [169] auf. Die Abschätzung des Umfangs der Teilgeste geschieht in [137] über die Eingabezeit, die somit für die gleiche Geste in etwa gleich sein sollte. Die Lösung über erschöpfende Suche in [93] wird als unpraktikabel angesehen.

5.2 Eigener Ansatz

Das größte Problem in der frühzeitigen Erkennung einer Geste ist die Abschätzung des Umfangs der eingegebenen Teilgeste. Naheliegende Lösungen, wie das Heranziehen verstrichener Zeit oder der Länge der Trajektorie, haben die Aufgabe wenigstens einer der gewünschten Invarianzen zur Folge. Zusätzlich werden mehrere Klassifikationen innerhalb einer Gesteneingabe notwendig, womit eine Nächste-Nachbar-Suche unter Einbezug aller möglicher Teilgesten der Templates zu aufwendig ist.

Der hier vorgeschlagene Ansatz beruht auf der Festlegung markanter Punkte in den Templates. Eine Erkennung wird zwar zu jedem Abtastzeitpunkt (hier sind auch Timeouts möglich), aber nur für die Templates vorgenommen, deren markante Punkte dem letzten Punkt der Eingabe am ähnlichsten sind. Somit kann die Last hinsichtlich der Echtzeitanforderung skaliert werden. Die markanten Punkte werden durch Merkmalsvektoren repräsentiert, die unabhängig vom Ort, der Skalierung und der Orientierung einer Geste sind.

Wurde die aktuelle Eingabe einem Template zugeordnet, so kann über dem zuge-

ordneten markanten Punkt auch deren Anteil im Template abgeschätzt werden. Dies wiederum ermöglicht die Prognose des weiteren Verlaufs sowie Hilfestellungen oder Trainingssysteme nach den Methoden in [13, 180, 59, 16].

Folgende Schritte sind nötig:

- Finden markanter Punkte in den Templates in der Trainingsphase.
- Festlegung einer Repräsentation markanter Punkte anhand von Merkmalen, die die Invarianzen unterstützen und welche unter sequenzieller, schrittweiser Adaption berechnet werden können.
- Speicherung der markanten Punkte in einer geeigneten Datenstruktur, welche Nächste-Nachbar-Suchen zum Auffinden ähnlichster Punkte unterstützt.
- Finden der zum letzten Punkt einer aktuellen Eingabe ähnlichsten markanten Punkte in den Templates sowie Klassifikation gegen die durch diese markanten Punkte repräsentierten und begrenzten Teilgesten.

Das Finden markanter Punkte wird über den Ramer-Douglas-Peucker-Algorithmus (RDP) [160, 43] vorgenommen. Er approximiert Kurven durch Auslassung der Punkte, die am wenigsten zur Form beitragen. In seiner ursprünglichen Version werden schrittweise Punkte zu einer aktuellen Approximation hinzugefügt und das rekursive Verfahren terminiert, sobald ein Punkt hinzugefügt würde, dessen Abstand zum verbleibenden Streckenzug einen anzugebenden Schwellwert unterschreitet.

In den Templates gefundene markante Punkte werden durch einen Merkmalsvektor repräsentiert dessen Merkmale in Abbildung 5.2 illustriert sind. Die Repräsentation eines markanten Punktes ist angelehnt an den in Untersuchungen im Kapitel 4.1 für geeignet befundenen Merkmalen aus Abbildung 4.2.

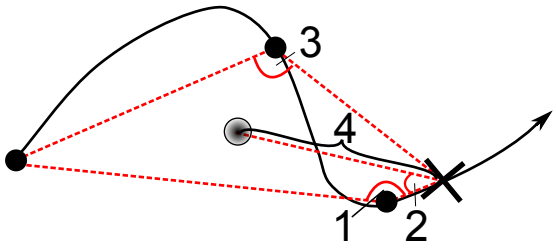


Abbildung 5.2: Das Merkmalsset aus Winkel- und Distanzmaßen - unter Einbezug interessierender Punkte auf der Trajektorie (schwarz) oder einem inkrementell berechneten Schwerpunkt (grau) - welches über dem markanten Punkt (Kreuz) berechnet und zur Ähnlichkeitssuche herangezogen wird.

Die in Abbildung 5.2 gezeigten Merkmale enthalten den vom Startpunkt, dem markanten Punkt in der Trajektorie vorhergehenden und dem markanten Punkt selbst eingeschlossenen Winkel (1), den vom Vorgänger in der Trajektorie, markanten Punkt und inkrementellen Schwerpunkt eingeschlossenen Winkel (2), den Winkel zwischen Startpunkt, dem Punkt auf halber Länge zum markanten Punkt (Median) und dem markanten Punkt (3) sowie den relativen Abstand des inkrementellen Schwerpunkts

zum markanten Punkt im Verhältnis zur Länge der Trajektorie bis zu diesem Punkt (4). Nicht in Abbildung 5.2 eingezeichnet ist das letzte Merkmal, welches die Kosinus-Distanz zwischen den beiden am Median geteilten Abschnitten der Trajektorie bis zum markanten Punkt ist und einen Indikator für die Kontinuität im Verlauf darstellt.

Im Fall mehrerer Trajektorien innerhalb einer durch den markanten Punkt begrenzten Teilgeste werden die strukturellen und temporalen Merkmale aus Kapitel 4.2.2 ergänzt, mit dem einzigen Unterschied, dass diese über den möglicherweise partiellen Trajektorien berechnet werden. Alle Merkmale lassen sich in Laufzeit maximal linear zur Länge der Trajektorie und in diesem Fall inkrementell berechnen.

Hier wird das RDP-Verfahren für die Bestimmung markanter Punkte in den Templates insofern angepasst (siehe Algorithmus 5.1), als dass eine Maximalzahl verbleibender markanter Punkte definiert wird. Wird ein markanter Punkt gefunden, so wird die verbleibende Zahl anteilig der Sample-Punkte der Trajektorie vor und nach diesem markanten Punkt aufgeteilt. Der erste Punkt einer Trajektorie wird am Ende des Verfahrens aus der Liste markanter Punkte für einen Token eines Templates entfernt.

Algorithm 5.1 ModifiedRamerDouglasPeucker(T,n,f,l)

Require: INPUT: T - einzelne Trajektorie einer Geste
Require: INPUT: n - Maximalzahl zu findender markanter Punkte
▷ Definiert die im aktuellen Rekursionsschritt relevante Sequenz der Trajektorie
Require: INPUT: f - Index des ersten Punktes
Require: INPUT: l - Index des letzten Punktes
▷ Am Ende der Rekursion enthält das Punkte-Set, unter Ignorierung der Duplikate und des ersten Punktes, alle interessanten, markanten Punkte
StoreToPointSet(T(f))
StoreToPointSet(T(l))
▷ Falls weitere markante Punte eingefügt werden, wähle denjenigen mit maximaler Lot-Distanz zum Liniensegment, welches durch die Punkte mit Index l und f definiert wird
if $n > 0$ & $l - f > 0$ **then**
  for all $i = f + 1$ to $l - 1$ **do**
    distance ← PerpendicularDistance(T(i),T(l),T(f))
    if distance>maxdistance **then**
      landmark ← i
      maxdistance ← distance
    end if
  end for
  StoreToPointSet(T(landmark))
  ▷ Verteile die verbleibenden zu suchenden markanten Punkte zu gleichen Teilen
  left ← $n \cdot (landmark - f)/(l - f)$
  right ← n-left
  ModifiedRamerDouglasPeucker((T,left,f,landmark))
  ModifiedRamerDouglasPeucker((T,right,landmark,right))
end if

Für jeden markanten Punkt einer Trajektorie einer Geste werden in jeder eventuell parallel vorhandenen Trajektorie zu dessen Abtast-Zeitpunkt[52] ebenfalls der Merkmalsvektor bestimmt. Die Kombination der Merkmalsvektoren aller Trajektorien zum Zeitpunkt eines markanten Punktes stellt einen Punkt im Raum dar. Für jede mögliche Kombination wird ein solcher Punkt in einen für diese Anzahl Trajektorien gewählten kd-Baum [17] samt Referenz auf das entsprechend der Merkmale umsortierte Template abgelegt. Diese Datenstruktur unterstützt für die Anfrage eines Punktes die effiziente Suche nach nächsten Nachbarn einer gegebenen Anzahl oder in einem festgelegten Radius. In Abbildung 5.3 ist der Vorgang veranschaulicht.

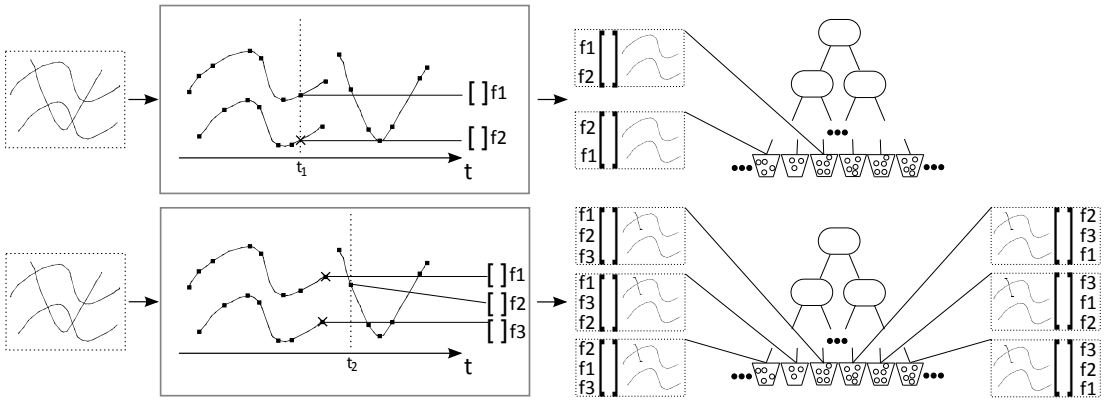


Abbildung 5.3: Der Prozess der Merkmalsgewinnung markanter Punkte, die stellvertretend für partielle Gesten stehen. Für jeden markanten Punkt (schwarz) werden zusätzlich Abtastpunkte (Kreuz) in den zeitlich parallelen (oben) oder bereits abgeschlossenen (unten) Trajektorien ermittelt. Die kombinierten Merkmale der Punkte werden für die anschließende Verwaltung zur effizienten Suche ähnlicher Teilgesten einer geeigneten Datenstruktur übergeben.

Während einer Eingabe werden nun zu jedem Zeitpunkt einer versuchten Klassifikation in dem kd-Baum für die derzeitige Anzahl an Trajektorien in der Eingabe die nächsten Nachbarn zum Merkmalsvektor des letzten Punktes der Eingabe gesucht. Die Kandidaten in Form der durch die nächsten Nachbarn referenzierten partiellen Templates werden für die Klassifikation herangezogen. Das Ergebnis der Klassifikation stellt die derzeit als am wahrscheinlichsten angesehene Eingabe dar.

5.3 Evaluation

Die Evaluation des Verfahrens erfolgte durch Klassifikationstests partieller Gesten mit Hilfe zweier Gestensets. Das Verfahren wurde zunächst anhand des in Kapitel 4.2.3 vorgestellten Multi-Touch Gestensets jeweils nutzerabhängig für alle Templates eines jeden Nutzers getestet.[53] Es sei angemerkt, dass dieses Set aufgrund seiner inhärent identischen Präfixe in den Gesten[54] für praktische Anwendung zur Prädiktion einer Eingabe

[52] Im Falle bereits beendeter Trajektorien wird deren letzter Punkt gewählt.

[53] Das Set enthält - bis auf eine Ausnahme - jeweils zehn Instanzen für jede der 20 Klassen pro Nutzer, womit insgesamt 1199 Gesten spezifiziert sind.

[54] So ist jeweils - bei Berücksichtigung der Pause zwischen den Strokes - mehr als die erste Hälfte der Eingabe der Gesten $\{1,2\},\{5,6,7\},\{4,9,10,11,13,14\},\{15,19\}$ und - unter rotationsinvarianter

ungeeignet ist. Für die Analyse und eine erste Abschätzung der Performance des Verfahrens ist dieser systematische 'Konstruktionsfehler' des nicht für diesen Zweck entwickelten Gestensets dennoch hilfreich. Für ein realistischeres Szenario wurde zusätzlich ein Testset konstruiert, welches sowohl Single- als auch Multi-Touch Gesten enthält.[55] Dazu wurde jeweils eine Multi-Touch Geste aus den Gruppen gleicher Präfixe des ersten Sets gewählt. Neben den dadurch enthaltenen Drei-Finger-Pinch-Gesten wurden durch den Autor definierte Zwei-Finger-Varianten hinzugefügt, um das Potenzial für die Erkennung direkter Manipulationen durch den vorgestellten Ansatz zu untersuchen. Mangels Verfügbarkeit anderer bekannter Multi-Touch Gestensets wurden diese Gesten durch eine Auswahl der in [179] entwickelten und im Kapitel 6.2 näher vorgestellten Gesten (alle Templates aller Nutzer) ergänzt. Vier weitere Single-Stroke Gesten wurden dem Set aus [217] (Templates der Geschwindigkeit 'medium' aller Nutzer) entnommen. In Abbildung 5.4 sind die beiden verwendeten Gestensets dargestellt.

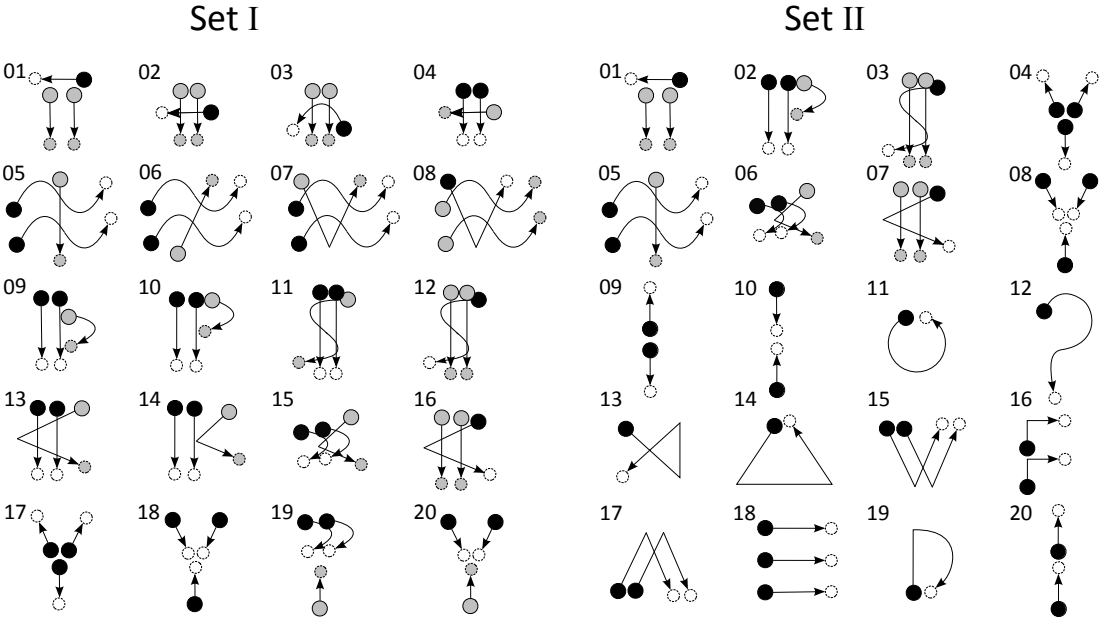


Abbildung 5.4: Die zwei Gestensets, welche für Testzwecke herangezogen wurden. Auf der linken Seite sind die Ausführungen des originalen Multi-Touch Gestensets dargestellt, dessen Gesten viele gemeinsame Präfixe aufweisen. Auf der rechten Seite ist eine modifizierte und realistischere Version abgebildet, welche Gesten des Sets aus [217], Multi-Touch Buchstaben des Gestenalphabets aus [179] und Zwei-Finger-Pinch-Gesten enthält. Ausgefüllte Kreise stehen für den Anfang der Trajektorie einer Berührung, Pfeile für die Richtung der Bewegung und gestrichelte, kleinere Kreise symbolisieren ihr Ende. Schwarz gefärbte Startmarkierungen gehören zum ersten Stroke, grau gefärbte zum zweiten.

Aus dem ersten Set wurden für jeden der sechs Nutzer, aus dem zweiten Set nutzerunabhängig Templates und Testfälle gewählt. Aufgrund der Konstruktion des Sets II sind Trennungen in nutzerabhängige Tests nicht möglich und die nutzerübergreifende Varianz in den Ausführungen der Gesten mutmaßlich höher. Die komplette, folgende

Erkennung - {8,16} komplett gleich.

[55] Für die verschiedenen Klassen des Sets existieren daher verschieden viele Spezifikationen, insgesamt sind 1046 Instanzen enthalten.

Prozedur wurde jeweils fünf mal für jedes dieser Testsets durchgeführt, wobei für das erste Gestenset die Ergebnisse über die sechs Testsets der Nutzer gemittelt wurden: Für jede der 20 Gesten wurden fünf bzw. in einer Ausnahme vier[56], zufällig gewählte Templates entnommen. In jedem dieser Templates wurden die zehn markantesten Punkte mit dem RDP-Verfahren gesucht und die Merkmalsvektoren samt korrespondierender Teilgesten in kd-Bäumen abgelegt. Aus den - je nach Nutzer - 99 bwz. 100 gewählten Templates des ersten Sets entstanden so in der Vorverarbeitung etwa 7300 Teilgesten.[57] Die Klassifikation erfolgte gegen jeweils fünf zufällig gewählte Gesten pro Klasse aus dem restlichen Set. Diese wurden in Teilgesten zwischen 10% und 100% (in 10% Schritten) ihres zeitlichen Fortschritts zerlegt, um eine kontinuierliche Gesteneingabe zu simulieren (siehe Abbildung 5.5). Bei einer Klassifikation wurde die Suche nach den nächsten Nachbarn in den kd-Bäumen[58] auf 10 Gesten (das entspricht etwa 0,14% der möglichen Teilgesten) begrenzt. Die Eingabe wurde dann gegen diese zehn gefundenen Teilgesten klassifiziert.

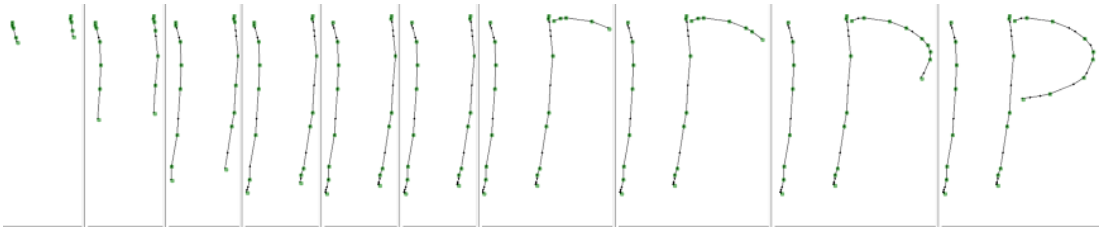


Abbildung 5.5: Zerlegung einer Geste aus der Klasse 10 des ersten Sets nach 10%-Schritten ihrer Gesamtdauer.

Die Ergebnisse der beiden Tests unter dem beschriebenen Verfahren werden nachfolgend präsentiert.

[56] Da für die Klasse 2 des ersten Sets für Teilnehmer 1 ein Template fehlt, wurde in diesem Fall nur mit vier Templates trainiert.

[57] Davon enthielten etwa 190 einen, 1570 zwei und 5540 drei Strokes.

[58] Es wird nur jeweils in dem kd-Baum gesucht, welcher Instanzen von Gesten mit der gleichen Zahl von Token verwaltet wie sie die aktuelle Eingabe aufweist.

5.4 Resultate

An dieser Stelle werden zunächst die Resultate für das erste Gestenset bereitgestellt. Tabelle 5.1 führt die Genauigkeiten für die Klassifikation partieller Gesten in Abhängigkeit der Länge der Eingaben auf. Die Angaben sind zeilenweise nach den durchschnittlichen Erkennungsraten für einzelne Gestenklassen sortiert.

Tabelle 5.1: Die Ergebnisse bezüglich der Genauigkeit von Prädiktionen partieller Gesten des Sets I. Die Werte wurden zeilenweise nach ihrem Durchschnitt - der über alle Längen partieller Gesten gemittelten Genauigkeit - sortiert. Jede Spalte enthält Werte die für Gesten-Eingaben einer festen, relativen Länge (in Prozent) ermittelt wurden.

Geste	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	⊘
20	0,80	0,91	0,89	0,89	0,92	0,92	0,92	0,97	0,98	0,99	0,92
17	0,51	0,79	0,94	0,97	0,99	1	1	1	1	1	0,92
18	0,41	0,77	0,97	1	1	1	1	1	0,99	0,98	0,91
12	0,64	0,68	1	0,96	0,95	0,94	0,73	1	1	1	0,89
8	0,95	0,64	0,80	0,84	0,84	0,82	0,99	1	1	0,99	0,89
3	0,62	0,80	0,85	0,82	0,83	0,91	0,79	0,93	0,94	0,99	0,85
15	0,48	0,73	0,74	0,78	0,83	0,81	0,95	1	1	1	0,83
19	0,51	0,66	0,85	0,84	0,87	0,85	0,85	0,87	1	1	0,83
16	0,59	0,53	0,78	0,85	0,85	0,87	0,73	1	1	1	0,82
1	0,54	0,52	0,55	0,57	0,57	0,44	0,71	0,92	1	1	0,68
2	0,36	0,49	0,44	0,42	0,42	0,52	0,78	0,96	0,99	0,99	0,64
11	0,17	0,23	0,35	0,35	0,59	0,87	0,87	0,87	0,98	0,98	0,63
7	0,27	0,29	0,46	0,48	0,49	0,53	0,80	0,88	0,97	0,97	0,61
13	0,34	0,33	0,39	0,39	0,39	0,65	0,84	0,86	0,89	0,98	0,61
6	0,37	0,30	0,45	0,47	0,52	0,55	0,57	0,88	0,99	0,95	0,61
14	0,18	0,32	0,33	0,26	0,27	0,60	0,97	1	1	1	0,59
9	0,27	0,28	0,23	0,37	0,37	0,53	0,95	0,96	0,96	0,97	0,59
10	0,22	0,20	0,29	0,23	0,23	0,39	0,93	0,96	0,98	0,99	0,54
5	0,31	0,37	0,36	0,40	0,41	0,42	0,45	0,78	0,93	0,96	0,54
4	0,24	0,22	0,29	0,37	0,40	0,42	0,63	0,85	0,97	0,98	0,54
⊘	0,44	0,50	0,60	0,61	0,64	0,70	0,82	0,93	0,98	0,99	0,72

Die Genauigkeit der Vorhersage ist am besten für die Gesten 20 und 17. Letztere wird dabei ab 30% der Eingabe in mindestens 94% der Fälle und ab 60% der Eingabe fehlerfrei erkannt. Für die Geste 18 werden ebenfalls hohe Genauigkeiten erreicht, was eine Eignung des Ansatzes auch für die Erkennung direkter Manipulationen (beispielsweise Zooming-Operationen) zeigt. Außerdem zu erkennen ist - neben der erwarteten, mit dem Fortschritt einer Geste steigenden Genauigkeit - eine starke Verbesserung dieser Rate ab der Eingabe des letzten Drittels einer Geste. Diese ist auf die eingangs erwähnte Eigenschaft des Gestensets zurückzuführen, da für 13 Gesten erst nach mehr als der Hälfte der Eingabe, Informationen zu deren Unterscheidbarkeit vorliegen.

Für eine differenziertere Betrachtung der an der Erkennung beteiligten Prozesse werden in Abbildung 5.6 die Ergebnisse der Nächste-Nachbar-Suche allein durch die Merkmale am aktuellen Punkt der Eingabe präsentiert. Die Abbildung zeigt auf, mit welchem Anteil die korrekte Gestenklasse bereits nächster Nachbar ist bzw. unter den nächsten beiden, respektive drei und zehn nächsten Nachbarn zu finden ist.

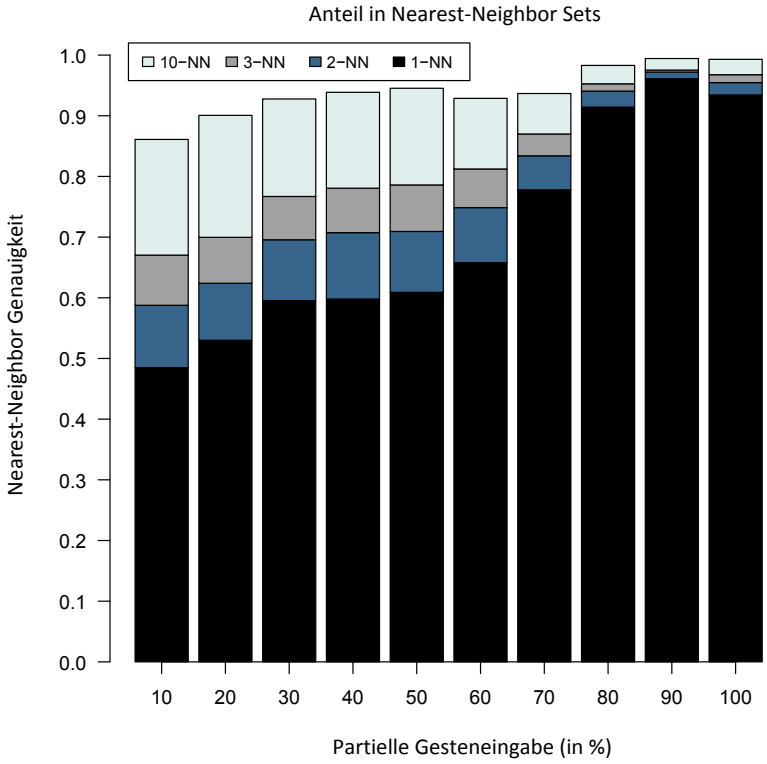


Abbildung 5.6: Für verschiedene Grade fortschreitender Eingabe, der Anteil mit welchem sich die korrekte Gestenklasse unter den nächsten eins bis drei und zehn Kandidaten der Nachbarsuche im kd-Baum befindet.

Es zeigt sich, dass mit wenig mehr als 10% der eingegebenen Geste in über 50% der Fälle der nächste Nachbar bereits die korrekte Klassifikation repräsentiert. Gleichzeitig lässt sich die obere Schranke der durchschnittlichen Erkennungsrate bei der Einschränkung des Suchraumes auf die 10 nächsten Nachbarn mit 98% ab etwa 80% der eingegebenen Geste und mindestens 90% ab 20% eingegebener Geste ablesen.

Demgegenüber wird an den unter einer Nächste-Nachbar-Suche mit Radius zehn erreichten und in Tabelle 5.1 aufgeführten Klassifikationsraten deutlich, dass das Potenzial der Nächste-Nachbar-Suche nicht ausgeschöpft wird. Allerdings verbessert sich das Verhältnis mit wachsendem Fortschritt der Eingabe. Es ist naheliegend, dass bei sehr kurzen Eingaben die Form eines Tokens keinen positiven Beitrag zur Klassifikation leisten kann und, bis zu einem Mindestmaß an Fortschritt in der Eingabe, die Erkennung allein auf Basis des nächsten Nachbarn das Potenzial auf Verbesserung der Ergebnisse birgt.

Um auch den Einfluss des Gestensets besser abschätzen zu können, werden in Abbildung 5.7 die Ergebnisse zur Vorhersagegenauigkeit unter dem Suchradius mit zehn nächsten Nachbarn noch einmal veranschaulicht. Die Genauigkeit ist zusätzlich für die dreizehn Gesten mit identischem Präfix und die verbleibenden sieben, sowie für die Gesten mit bestem und schlechtestem Resultat getrennt aufgeführt.

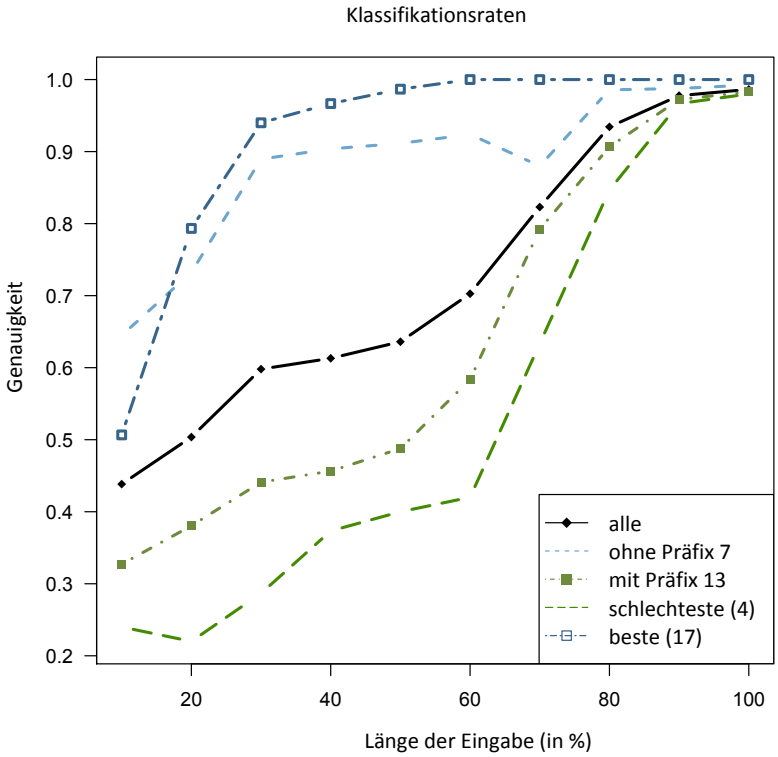


Abbildung 5.7: Die durchschnittliche Genauigkeit der Vorhersage beabsichtigter Gesten des Testsets I bei unterschiedlich fortgeschrittenen Eingaben und einer Suche innerhalb der zehn nächsten Nachbarn.

Im Vergleich zur Nächste-Nachbar-Suche wird im Durchschnitt die korrekte Geste in über 50% der Fälle ab einer Eingabe von mehr als 20% vorhergesagt. Eine korrekte Auswahl unter den zwei nächsten Nachbarn hätte demnach bereits eine höhere Genauigkeit zur Folge. So enthält dieses Set die korrekte Wahl im Durchschnitt über alle Teilgesten in 94% der Fälle gegenüber durchschnittlicher korrekter Klassifikation von wenig mehr als 72% bei Heranziehen der bis zu 20 nächsten Nachbarn (siehe auch Abbildung 5.9). Wie erwartet zeigt sich, dass der Klassifizierer aus einer Nachbarschaft unter größerem Suchradius umso besser selektiert je mehr von einer Geste eingegeben wurde. Vorhersagen beendeter Gesten (100%) werden mit nur leicht schlechterer Genauigkeit getroffen als bei einer Klassifikation des Gestenerkenners gegen alle kompletten Gestentemplates. Allerdings zeigt die Abbildung 5.7 auf, dass dieses Ergebnis maßgeblich durch die Gesten mit identischen Präfixen beeinflusst wird. Die Raten der Prädiktion steigen bei den sieben Gesten ohne identische Präfixe schon bei 30% der Eingabe auf akzeptable Werte. Ab diesem Umfang einer Eingabe werden sie in nie weniger als 88% der Fälle korrekt vorausgesagt. Weitere Verbesserungen durch zusätzliche Informationen über den Verlauf der Geste verbessern diese Rate dementsprechend nur noch leicht. Der Knick im Verlauf der Raten zur Prädiktion bei etwas mehr als 60% der Eingabe deutet ebenfalls darauf hin, dass zu kurze Token die Klassifikation negativ beeinflussen.

Die erzielten Genauigkeiten zur Prädiktion partieller Gesten des Sets II sind in Tabelle 5.2 gegeben.

Tabelle 5.2: Resultate für die Prädiktions-Genauigkeit für partielle Gesten des Sets II. Die Werte sind spaltenweise abhängig der relativen Länge einer Eingabe aufgeführt. Abermals erfolgte eine zeilenweise Sortierung der Werte nach der durchschnittlichen Genauigkeit über alle Längen (letzte Spalte).

Geste	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	⌀
9	0,76	1	1	1	1	1	1	1	1	1	0,98
2	0,92	0,88	0,92	1	1	1	1	1	1	1	0,97
10	0,80	0,96	0,96	1	1	1	1	1	1	1	0,97
8	0,56	0,80	1	1	1	1	1	1	1	1	0,94
6	0,64	0,84	0,96	0,92	0,92	0,92	1	1	1	1	0,92
19	0,60	1	0,92	0,56	1	1	1	1	1	1	0,91
4	0,44	0,92	0,96	0,96	0,96	0,96	0,96	0,96	0,96	0,96	0,9
18	0,36	0,8	0,84	0,92	0,96	1	1	1	1	1	0,89
16	0,64	0,96	0,92	0,96	0,96	1	1	0,88	0,68	0,88	0,89
5	0,44	0,64	0,84	1	0,92	0,92	0,92	0,96	1	1	0,86
15	0,64	0,56	0,56	0,72	0,84	1	1	1	1	1	0,83
11	0	0,4	0,96	1	1	1	1	1	0,96	0,88	0,82
3	0,76	0,48	1	0,76	0,76	0,72	0,68	1	1	1	0,82
17	0,68	0,88	0,84	0,64	0,48	0,68	1	0,92	0,96	0,96	0,80
7	0,48	0,56	0,88	0,80	0,76	0,76	0,76	1	1	1	0,80
20	0,56	0,64	0,68	0,80	0,84	0,96	0,92	0,88	0,88	0,60	0,78
12	0,20	0,40	0,76	0,80	0,84	0,80	0,80	0,88	0,96	1	0,74
1	0,72	0,52	0,44	0,44	0,44	0,36	0,88	0,96	1	1	0,68
13	0,24	0,24	0,16	0,28	0,44	0,48	0,64	0,88	1	0,92	0,53
14	0,12	0,24	0,36	0,40	0,40	0,40	0,36	0,48	0,96	0,88	0,46
⌀	0,53	0,69	0,80	0,80	0,83	0,85	0,9	0,94	0,97	0,95	0,82

Die durchschnittliche Klassifikationsgüte unterschreitet demnach 80% ab 30% der Eingabe und 90% ab 70% der Eingabe nicht mehr. Für einen praktischen Anwendungsfall verspricht dies eine mögliche Verkürzung der Eingabe auf weniger als 50% bei immer noch praktikablen Erkennungsraten. Insgesamt wird eine durchschnittliche - und erwartungsgemäß gegenüber dem ersten Set höhere - Genauigkeit von 82% erreicht. Die Rate der korrekten Prädiktionen ist für das erste Set jedoch höher, wenn die Eingabe einer Geste fast vollständig ist. Die an direkten Manipulationen angelehnten (Pinch-)Gesten 9 und 10 werden ab 30% bzw. 40% der Eingabe sicher klassifiziert. Die drei-Finger Versionen 4 und 8 werden ebenfalls schon ab wenig verfügbaren Daten verlässlich erkannt. Bei partieller Eingabe am schlechtesten erkannt werden die Gesten 13 und 14, welche, bedingt durch die rotationsinvariante Klassifikation einen gemeinsamen Präfix aufweisen und dementsprechend erst im letzten Teil der Eingabe voneinander unterscheidbar werden.

Um abermals die an der Klassifikation beteiligten Prozesse besser bewerten zu können, werden in Abbildung 5.8 die durchschnittlichen Prädiktions-Raten den Trefferquoten für die Nächste-Nachbar-Suchen gegenübergestellt.

Es bestätigt sich die Beobachtung, dass für kurze Eingaben die Klassifikation allein auf Basis des nächsten Nachbarn geeigneter ist. Erst wenn genügend Informationen

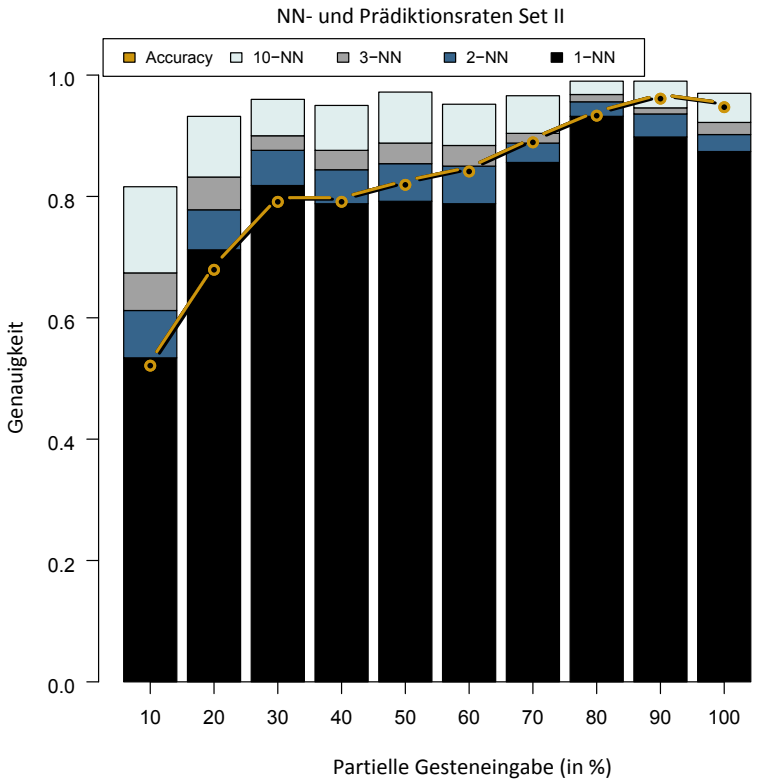


Abbildung 5.8: Für das Gestenset II ist die Rate, mit der die korrekte Gestenklasse bereits der nächste Nachbar ist oder sich unter den nächsten zwei, drei und zehn Kandidaten der Suche im kd-Baum befindet, der durchschnittlichen Genauigkeit der Klassifikation partieller Gesten gegen die zehn nächsten Nachbarn der Suche gegenübergestellt.

vorhanden sind, um die Form eines Tokens zu bewerten und mit den in den Templates abgelegten Kandidaten zu vergleichen, lohnt sich der der Einsatz des hierarchischen Gestenerkenners. Im konkreten Fall ist bei Eingaben unter 30% Länge die Nächste-Nachbar-Suche ohne anschließende Klassifikation die genauere Methode. Mit zunehmender Eingabelänge findet eine bessere Selektion im Set statt, aber erst gegen Ende einer Eingabe wird eine Größere Menge an Kandidaten zweckmäßig.

Parametrisierungen

Um mehr Einblick in die Auswirkungen der Parameterwahl zu geben, sind in Abbildungen 5.9 die über alle Testeingaben und alle Längen gemittelten Quoten des Vorhandenseins der richtigen Gestenklasse in den Nächste-Nachbar-Sets der Größe eins bis zwanzig dargestellt. Die Abbildung enthält zudem demgegenüber zusammengefasste, tatsächliche Ergebnisse der Güte für drei verschiedene Belegungen des Umfanges der Nächste-Nachbar-Suche.

Wie in Abbildung 5.9 zu sehen, umfassen die nächsten Nachbarn bereits eine gute Auswahl an Kandidaten. Eine Ausweitung des Suchradius auf eine 20-Nachbarschaft verspricht etwas Zugewinn und scheint im vorliegenden Fall ein geeigneter Kompromiss

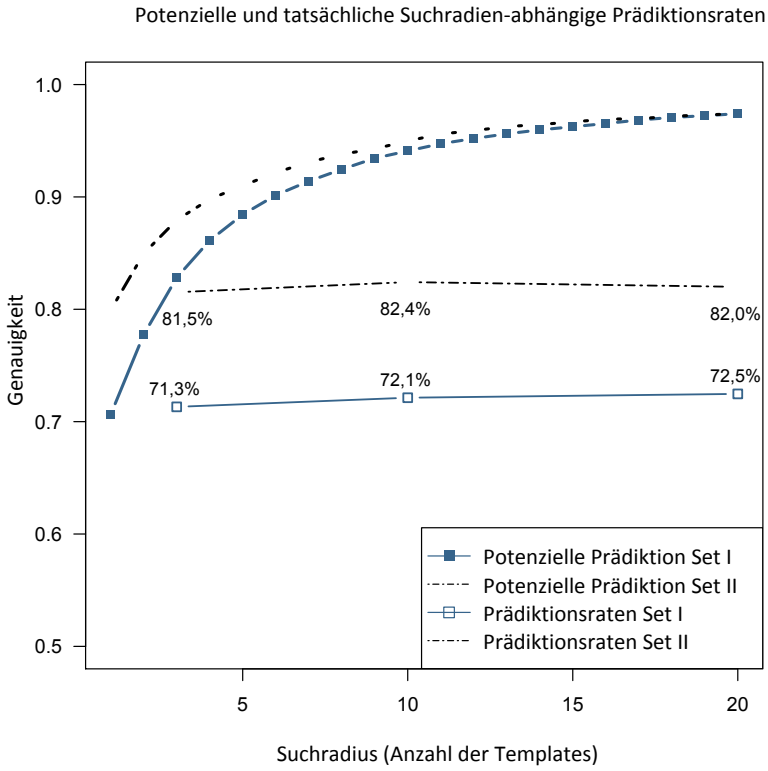


Abbildung 5.9: Die mögliche durchschnittliche Klassifikationsrate über alle Teilgesten, wenn aus dem Set der nächsten Nachbarn (eins bis zwanzig) immer korrekt gewählt würde wird der tatsächlichen, über allen Teilgesten gemittelten Genauigkeit der Klassifikation bei der Beschränkung der Nächste-Nachbar-Suche auf drei, zehn und zwanzig Elemente gegenübergestellt.

zwischen Performance und Genauigkeit zu sein. Tatsächlich deutet sich durch die in Abbildung 5.9 ebenfalls gezeigten Genauigkeiten für Klassifkationen unter verschiedenen Suchradien an, dass für beide Gestensets, zumindest bei moderater Ausweitung der Suche, keine Verbesserungen zu erwarten sind.

Ein weiterer Parameter ist die Anzahl der Landmark-Points, die pro Gestentemplate herangezogen werden. Im Durchschnitt enthalten die Token der Gesten des Sets I knapp 24 Abtastpunkte[59]. Die Wahl, die Suche des RDP-Algorithmus auf zehn markante Punkte zu begrenzen, ist eher einer Abschätzung der möglichst sinnvollen Anzahl an Klassifikationen während einer Eingabe geschuldet. Werden alle Punkte eines Token als markante Punkte gewählt, so erhöht sich die Treffergenauigkeit der Nächste-Nachbar-Suche bei sonst gleichem Testsetting für das erste Set um 0,59%. Eine größere Anzahl an markanten Punkten verschlechtert demnach die Suche nicht, scheint aber unter Betrachtung des anderweitigen Verbesserungspotenzials nicht die geeignetste Stellschraube. Für das zweite Set verbessert diese Anpassung hingegen die Nächste-Nachbar-Suche von 80% auf 83% und die Klassifikationsgenauigkeit von 82% auf durchschnittlich 85%.

[59] Die Spanne über die Sets der Nutzer liegt etwa zwischen 20 und 30 Punkten.

Die Klassifikation einer partiellen Geste aus Set I benötigte im Durchschnitt[60] 97,73ms, womit bei einer ermittelten Eingabedauer von durchschnittlich[61] 1,41s für eine Geste aus dem Testset mehr als zehn Klassifikationen vorgenommen werden können.

Eine weitere mögliche Modifikation ist die Zuordnung der Trajektorien einer Eingabe zu denen der durch die Nächste-Nachbar-Suche ermittelte Auswahl an Templates nur anhand der markanten Punkte. Auch wenn diese Methode der im Kapitel 4 vorgestellten Maximum-Likelihood-Zuordnung unterlegen ist, kann der Gewinn für die Laufzeit des Verfahrens in eine breitere Suche investiert werden. Beim Test brachte der Verzicht auf ein Matching der Token im Klassifikationsschritt jedoch nur einen geringen Geschwindigkeitsvorteil, zeigte sich allerdings in stärkeren Einbußen der Genauigkeit. Gegenüber der Zuordnung der Token einer Eingabe zu denen eines Templates allein durch die im Merkmalsvektor des nächsten Nachbarn implizierte Reihenfolge wurde mit dem klassischen Matching eine durchschnittlich etwa um 14 Prozentpunkte höhere Genauigkeit erzielt. Es ist anzunehmen, dass dies auf eine bessere Aussortierung falscher Kandidaten in der Menge der nächsten Nachbarn zurückzuführen ist.

5.5 Diskussion und Ausblick

Der vorgestellte Ansatz erlaubt die zuverlässige Vorhersage beabsichtigter Gesten während einer Eingabe. Die zu erkennenden Gesten werden dabei nur durch Templates spezifiziert. Prognosen anhand partieller Gesten beliebigen Typs werden entsprechend den Anforderungen unabhängig ihrer Orientierung, Platzierung oder Skalierung schon ab wenig fortgeschrittener Eingabe möglich. Damit werden auch per Templates definierbare direkte Manipulationen unterstützt. Je nach Anwendungsgebiet ist ein Ablehnen einer Entscheidung auf Kosten der erreichbaren Genauigkeit sinnvoll. Dies kann über einen Suchradius per Schwellwert anstatt der Anzahl nächster Nachbarn für die Auswahl geeigneter Kandidaten unter den Templates geschehen. Das Verfahren kann bezüglich der Anforderungen an die Echtzeit skaliert werden. Die Einbeziehung aller markanten Punkte der Templates und die damit einhergehende Vergrößerung des Template-Sets partieller Gesten verspricht eine höhere Genauigkeit. Leichte Verbesserungen des Verfahrens waren auch unter Verwendung der PCA für die Klassifikation partieller Gesten zu beobachten. Das größte Potenzial liegt allerdings in der Spezifikation präfixfreier Gestensets. Hier können ebenfalls, je nach Anwendung, absolute Merkmale ergänzt werden, so dass die Möglichkeiten gemeinsamer Präfixe der Gesten reduziert werden. Weitere Verbesserungen versprechen der Einbezug vergangener Beobachtungen in die Interpretation der aktuellen Prognose sowie die Klassifikation nur durch den nächsten Nachbarn der Ähnlichkeitssuche anhand der markanten Punkte in Abhängigkeit des Fortschritts

[60] Testumgebung: AMD Phenom II X4 945 Prozessor (3.01 Ghz)

[61] Für die zufällig gewählten Templates wurden die Durchschnittswerte pro Nutzerset berechnet und nochmals über die Nutzer gemittelt. Die minimale über die Testgesten gemittelte Eingabedauer für einen Nutzer lag bei 1,29s und die maximale Dauer bei 1,58s.

der Geste. Die Abhängigkeit der Interpretationen vom Anwendungskontext ist außerdem ein interessanter Ansatz für weiterführende Untersuchungen. Vorstellbare Anwendungen sind Werkzeuge, die dynamische Feedforward-Mechanismen beim Skizzieren sowie bei der gestischen Eingabe von Text bieten oder Trainingskonzepte für das Erlernen von Gesten ähnlich dem Ansatz in [180].

6

Anwendungen und Proof of Concept

In diesem Kapitel sollen mögliche Anwendungen des universellen Gestenklassifizierers vorgestellt werden. Da die Möglichkeiten von Multi-Touch Gesten bisher nicht ausgeschöpft werden, wurden de facto Probleme gelöst, welche so bisher nicht existierten. Mathematik-Editoren auf Basis handschriftlicher Eingaben [194, 225] können von den Konzepten der Erkennung ebenso profitieren wie vom direkten Einsatz des Klassifikators, um fließende Übergänge zwischen den Interaktionstechniken oder Eingaben per Multi-Touch zu erlauben. In Kombination mit Methoden der Autovervollständigung können Lernwerkzeuge zur Vermittlung von Gesten oder Formen ähnlich dem haptischen Ansatz von Crossan und Brewster [41] auch für blinde und sehbehinderte Nutzer entwickelt werden. Um jedoch nicht nur einen spekulativen Ausblick bieten zu können, wurden zwei Anwendungen entwickelt, welche die Einsatzmöglichkeiten des Gestenerkenners demonstrieren und gleichzeitig Anregungen zu weiteren Ideen liefern sollen. Die erste vorgestellte Anwendung basiert auf der Übertragung der vorgestellten Konzepte auf die Erkennung von Skizzen. Gleichzeitig werden die Unterschiede von Gesten und Skizzen verdeutlicht und Anpassungen diskutiert, um den Gestenerkenner im Bereich der Skizzen einzusetzen. Es soll die Eingabe von Multi-Touch bei der Erstellung per Finger gezeichneter Skizzen erlaubt werden. Außerdem ist die Anwendung um selbstdefinierte Skizzen erweiterbar und lässt somit die domänenunabhängige Verwendung zu. Die zweite vorgestellte Anwendung ist ein System für die Texteingabe, ebenfalls unter Ausnutzung von Multi-Touch. Hier steht die Idee im Vordergrund, visuell an lateinische Buchstaben angelehnte Zeichen - durch die Verwendung mehrerer Finger gleichzeitig - schneller eingeben zu können. Es wird zudem untersucht, ob Nutzer Intuitivität oder Effizienz bei der Texteingabe bevorzugen. Dazu werden die Möglichkeiten der Template-

basierten Erkennung ausgenutzt, um unterschiedliche Symbole zu spezifizieren und in Nutzertests vergleichen zu können.

6.1 Erkennung selbst definierbarer Skizzen

An dieser Stelle werden die Konzepte und Umsetzung eines Klassifikators für Multi-Touch Skizzen vorgestellt. Der Gestenklassifizierer wurde diesbezüglich angepasst und in eine Anwendung zum Erstellen domänenunabhängiger Skizzen integriert. Die Anwendung verschönert (engl. beautification) mit einer praktikablen Erkennungsrate beliebige, selbst definierbare Skizzen und ist vielseitig einsetzbar. Das vorliegende Kapitel wurde in leicht modifizierter Version in [182] veröffentlicht.

6.1.1 Gesten versus Skizzen

Die Erkennung erstellter Skizzen erlaubt deren Interpretation, Nachbearbeitung, Suche und Aufbereitung für eine bessere Darstellung [81]. Die Anwendungsgebiete sind seit deren Aufkommen ('Sketchpad' [199] wurde 1963 vorgestellt) breit gefächert und umfassen u.a. die Erstellung von UML [71, 38] und anderen Diagrammen [226, 87, 209], UI Entwürfen [111, 40, 26], technischen Skizzen [109] oder 3D Modellen [10].

Die Klassifikation von Gesten ist ein zur Interpretation von Skizzen eng verwandtes Gebiet (siehe auch Klassifikationsmethoden in Abschnitt 3.4). Ein Unterschied zwischen den Klassifikationen beider Trajektorie-basierter Eingaben ist, dass Skizzen eher unter Multi-Stroke Eingaben entstehen [73], während diese Eingabetechnik bei planaren Gesten (noch) nicht verbreitet ist. Ein weiteres Problem bei der Erkennung von Skizzen ist, dass deren bildhafte Darstellung relevant ist und nicht die Reihenfolge der Entstehung. Die zeitlichen Relationen in der Ausführung sind allerdings ein Unterscheidungsmerkmal für Gesten. Auch können mehrere Objekte skizziert werden, so dass eine Gruppierung der Trajektorien unter unbekannter Reihenfolge und eventuell unvollständig verbundenen Strichen notwendig wird [9]. Daher ist in einer Vorverarbeitung die korrekte Gruppierung [209, 189] neben der eventuellen Segmentierung [34, 223, 27, 190, 81] oder Verbindung von Trajektorien notwendig. Aus diesem Grund werden die Möglichkeiten der Eingabe häufig beschränkt. Üblich sind die Verwendung von Timeouts oder zusätzliches Betätigen von Schaltflächen, um die Vervollständigung einer Eingabe zu signalisieren [73]. Timeouts, um die Zugehörigkeit der Teileingaben zu einer Skizze zu definieren, werden beispielsweise in [9] genutzt. In [149] werden derartige Einschränkungen und der daraus resultierende Trainingsaufwand allerdings als Gründe dafür benannt, dass Erkenner für Skizzen es noch nicht in Mainstream-Software geschafft haben.

Ansätze, die solche Restriktionen vermeiden, interpretieren die räumliche [209] oder zusätzlich zeitliche [71] Nähe von Trajektorien für deren Gruppierung. Sezgin und Davis [189] berücksichtigen bei der Erkennung verschiedene, in Nutzerstudien als konsistent beobachtete, Möglichkeiten der Reihenfolge einer Eingabe.

Weitergehende Segmentierung der Strokes wird in [27] anhand der Krümmungen und Geschwindigkeitsänderungen vorgenommen. Ein Segmentierungspunkt wird detektiert, sobald die Ausführungsgeschwindigkeit unter einen Schwellwert fällt. Ein ähnlicher Ansatz, in dem ebenfalls Schwellwerte für Geschwindigkeiten und Richtungsänderungen verwendet werden, findet sich in [190]. Chen und Xie [34] beziehen Geschwindigkeit, Beschleunigung und Richtungsänderungen ein, um Wendepunkte zu erkennen.

Die Erkennung von Primitiven muss nicht zwangsläufig von der Vorverarbeitung zur Gruppierung und Segmentierung entkoppelt sein. Während Calhoun et al. [27] die Segmente der Strokes anhand von Regeln in Linien und Bögen einteilen, wird in [81] direkt eine optimale Zerlegung von Multi-Stroke Symbolen in die gleichen Primitive mittels dynamischer Programmierung gesucht. Allerdings basiert dieser Ansatz auf bereits segmentierten Templates für alle Primitive und unterstützt - wie die meisten Verfahren - ebenfalls kein Verknüpfen[62] von Strokes zu einem Primitiv. Zusammenfügen von Strokes wird in [223] vorgenommen. Der ebenda verfolgte Ansatz segmentiert Strokes anhand ihrer Krümmungen sobald eine Eingabe abgeschlossen ist. Können die Segmente keinen Primitiven zugeordnet werden, wird die Segmentierung rekursiv auf die Segmente angewendet. Abschließend wird getestet, ob dass Verknüpfen von Primitiven zu besseren Interpretationen führt.

Trotz der im Allgemeinen zusätzlich nötigen Verarbeitungsschritte bei der Interpretation von Skizzen wird in der Literatur häufig kein Unterschied zwischen Gesten und Skizzen gemacht. Auch Coyette et al. [39] stellen fest, dass Skizzen-Anwendungen zumeist auf einem Erkennungsverfahren für Gesten und primitive Formen basieren. Der Gestenerkenner von Rubine [170] wird beispielsweise in [77, 111, 40] direkt eingesetzt, um sowohl Gesten als auch Primitive zu erkennen. Weiterhin wird er üblicherweise beim Vergleich verschiedener Verfahren zur Erkennung von Skizzen aufgeführt [72, 223, 149, 109], obwohl Paulson und Hammond [149] die mangelnde Vielfalt im Skizzieren bei der Anwendung solcher Methoden kritisieren. Dennoch wird das Merkmalsset aus [170] auch in [21] (ebenfalls für Primitive und Gesten) eingesetzt, wenn auch unter einem anderen statistischen Klassifikator (SVM statt Bayes'sche Klassifikation). Weitere Arbeiten, die die gleichen Techniken für die Erkennung von Gesten und Skizzen anwenden, finden sich in [38, 226, 223, 58].

Verfahren, die zur Erkennung von Skizzen eingesetzt werden und eher an Methoden der Gestenerkennung angelehnt sind [169, 40], vermeiden explizite Segmentierungen und Gruppierungen werden implizit durch den Nutzer vorgenommen. Die Richtung und Reihenfolge der Eingabe ist durch die Verwendung der Online-Informationen vorgegeben. Interessanterweise stellen Sezgin und Davis [189] fest, dass sich Stile im Skizzieren zwar für verschiedene Nutzer unterscheiden, aber pro Nutzer relativ konsistent sind. Daher kann dieser Nachteil dadurch kompensiert werden, dass, im Gegensatz zu den meisten, zwar auf intuitiveren und fassbareren Regeln basierenden Skizzenerkennern

[62] Damit darf eine Eingabe maximal so viele Strokes enthalten, wie die Anzahl der im zugeordneten Template vorhandenen Primitive.

[9, 58, 190, 226, 34, 27, 223], Gesten-basierte Ansätze das leicht durch Nutzer durchzuführende Training per Templates unterstützen [170, 21, 40]. Die Spezifikation von Primitiven[63], auch in mehreren Varianten, erlaubt ohne aufwendige Erkennungstechniken die Individualisierung nach eigenen Stilen der Nutzer.

Zusätzliche Vorteile unterstützen die Anwendung Gesten-basierter Ansätze. Durch [58, 209] wird bestärkt, dass trainierbare, statistische Klassifizierer Regel-basierten Ansätzen im Sinne der Erkennungsgenauigkeit überlegen sind. Wenyin et al. [209] vergleichen dabei Regel-basierte Methoden mit SVM und künstlichen neuronalen Netzen in ihrer Leistungsfähigkeit, einfache Primitive zu erkennen. Dabei schneiden SVM und neuronale Netze bezüglich der Genauigkeit besser ab. In [58] wird ein Vergleich trainierbarer Methoden - Nächste-Nachbar-Klassifikation, Induktive Entscheidungsbäume und das Naïve Bayes Verfahren - vorgenommen, wobei der letztere, statistische Ansatz am besten abschneidet. Allerdings werden, obwohl weniger Trainingsdaten als in den anderen Methoden, immer noch eine unpraktische Anzahl von mehr als 40 Templates pro Klasse gebraucht, um eine Erkennungsrate von über 90% zu erreichen.

Letztendlich ist ein weiterer Vorteil der Skizzenerkennung durch eine Methode der Gestenerkennung, dass die Kombination von Gesten und Skizzen in einem User Interface aus einem Guss und jeweils durch Spezifikation der Templates realisiert werden kann. Häufig werden beide Techniken der Eingabe kombiniert, um natürlichere und flüssigere Bedienungen zu ermöglichen. In der hier vorgestellten Variante lässt die Option, Skizzen mit mehreren Fingern einzugeben, eine engere Verknüpfung gestischer Interaktion und des Skizzierens zu. Die Unterscheidung der flexibleren Eingabe durch unterschiedliche Anzahl benutzter Finger in verschiedenen Template-Sets erlaubt einen flüssigen Wechsel zwischen den verschiedenen Eingabemodalitäten (Gesten, Skizzen, direkte Manipulation). Die andernfalls notwendigen Modi-Schaltflächen, Timeouts oder Wechsel des Eingabegerätes sind dennoch weiterhin anwendbar.

6.1.2 Realisierung der Skizzenerkennung

Der in Kapitel 4 vorgestellte, trainierbare Gestenerkenner wurde durch den Autor in die zu diesem Zweck entwickelte Software 'SkApp'[64] integriert. Sie erlaubt Nutzern, beliebige, idealisierte Skizzen zu konstruieren und diese dann mit gestischen Kommandos, die die Skizze widerspiegeln können, aber nicht müssen, zu verknüpfen.

Abbildung 6.1 zeigt Skizzen aus zwei Anwendungsdomänen - GUI-Design und UML - die stellvertretend für den Proof of Concept der Multi-Touch Skizzenerkennung gewählt wurden. Die Skizzen entstanden durch Eingabe mit den Fingern auf einem Android-Tablet vom Typ Motorola Xoom.

[63] Auch wenn in der Literatur die Unterscheidung in Primitive und komplexe Skizzen getroffen wird, sollen an dieser Stelle elementare, interpretierbare Bestandteile einer skizzierten Szene als Primitive bezeichnet werden.

[64] 'SkApp' ist eine Android-App, die innerhalb studentischer Komplexpraktika und einer Belegarbeit entwickelt wurde und das Erstellen von Skizzen auf einem Tablet zulässt.

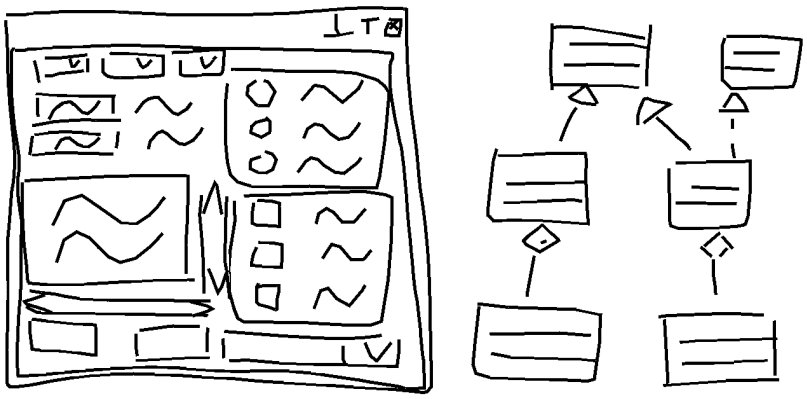


Abbildung 6.1: Eine Zusammenstellung nicht interpretierter Skizzen. Beliebige Teile können als idealisierte Primitive markiert und als Templates, verknüpft mit einer gestischen Eingabe, abgelegt werden. Die Eingaben werden unabhängig des Grades der Gleichzeitigkeit oder Reihenfolge ihrer Trajektorien klassifiziert. Eine Unterscheidung nach der Anzahl und der Eingabe-Richtung der Trajektorien findet statt, aber ein Primitiv kann mit mehreren diesbezüglichen Varianten einer Eingabe verknüpft werden.

Der Gestenerkenner wurde insofern angepasst, als dass das Merkmalsset um das zeitliche Merkmal reduziert wurde. Die Wellenlinien innerhalb des Rechtecks in den abgebildeten Skizzen können somit beispielsweise simultan mit zwei Fingern parallel oder sequenziell (oder mit beliebigem zeitlichen Versatz) skizziert werden. Im ursprünglichen Verfahren hätten Templates für jede gewählte Variante spezifiziert werden müssen. Zudem werden zu den Templates für jede Klasse zusätzliche Informationen abgelegt, wie bei einer Klassifizierung die genaue Darstellung der Skizze geschehen soll. Der Klassifizierer ist invariant gegenüber Translation, Skalierung und Rotation, aber eine Ausgabe sollte auch eine feste Orientierung oder Skalierung aufweisen können. Aus diesem Grund werden die Parameter Größe, Lage und Drehung der Eingabe in Relation zum ähnlichsten Template an die Anwendung zurückgegeben. Die interpretierte und idealisierte Skizze wird dann nur insoweit der Eingabe angepasst, wie es für die entsprechende Klasse definiert ist. Als eine weitere Anpassung wurde die Klassifikation von Taps (Antippen) per Regeln vorgenommen. Im ursprünglichen Gestenerkenner werden Token mittels Nächste-Nachbar-Vergleich klassifiziert. Taps werden nun davon ausgenommen, indem Trajektorien, die einen Schwellwert in ihrer Länge nicht überschreiten, automatisch als Tap klassifiziert werden. Beim Vergleich von Eingaben mit Templates haben zwei Trajektorien immer maximale Ähnlichkeit, wenn beide ein Tap sind und maximale Unähnlichkeit, wenn das nur für eine Trajektorie der Fall ist. Die Nächste-Nachbar-Klassifikation ist zwar invariant gegenüber gleichförmiger Skalierung, unterscheidet aber nach Ausdehnung in nur einer Dimension. Aus diesem Grund wird eine ungleichförmige Skalierung vorgenommen, sobald eine Eingabe mit einem Template verglichen wird, welches ein Primitiv repräsentiert, dessen Größe nur in einer Dimension skaliert (für Skizzen, die beispielsweise Scrollbars in einem UI-Kontext abbilden sollen). Außerdem werden die Parameter der Entscheidungsregel entsprechend der Annahme einer Standardnormalverteilung der Merkmale gewählt (Einheitsmatrix als Kovarianz-

schätzer), auch wenn das nicht das ganze Potenzial des Klassifizierers ausschöpft. Dies erlaubt direkte Vergleiche, auch wenn nur ein Template pro Klasse oder Templates in unterschiedlichen Variationen für das gleiche Primitiv spezifiziert wurden. Im letzteren Fall wären andernfalls verschiedene Klassen mit der gleichen verknüpften Aktion anzulegen, was das User Interface unnötig komplexer werden lässt.

Bedingt durch die auf Templates basierende Erkennung kann ein Nutzer auch Variationen oder der idealisierten Skizze unähnliche gestische Kommandos definieren. So ist es möglich, etwa für komplexe Skizzen (zusätzlich) 'Abkürzungen' in Form abstrakterer Symbole anzulegen. In der linken Skizze (GUI) in Abbildung 6.1 wurden solche Abstraktionen für die Schaltflächen 'Minimieren', 'Maximieren' und 'Schließen' eingesetzt. Im UML-Klassendiagramm (rechte Seite) wird ein einzelner Tap innerhalb des Kopfes des Pfeiles (Rhombus) benötigt, um die farbliche Unterscheidung zwischen Komposition und Aggregation zu symbolisieren.

Unterstützte Symbole

In einer ausführlichen Literaturstudie zur Klassifikation von Skizzen wurden verschiedene Objekte zusammengetragen, die üblicherweise erkannt werden. Meist wird nur ein recht begrenztes Set geometrischer Formen unterstützt. In manchen Arbeiten werden Segmentierungen in Linien und Bögen zum Zweck der idealisierteren Darstellung vorgenommen [81, 38, 27], während andere zusätzliche Formen wie Ellipsen, Drei- und Vierecke unterstützen [9, 209, 34], um die Interpretation der Objekte zu erlauben. Aus diesen Primitiven werden domänenspezifische, komplexere Skizzen oft über Grammatiken oder Regeln komponiert. Die in der Literatur behandelten Objekte lassen sich größtenteils in das folgende Set einordnen: Linien, Bögen, Ellipsen, Kreise, Dreiecke, Rechtecke, Rhomben, Pfeile und kreuzende Linien. Diese Eingaben können per Templates spezifiziert werden und der Erkenner ist in der Lage, größere Sets zu unterscheiden. Weitere Elemente, die durch manche Anwendungen unterstützt werden, sind: Spiralen, Schnörkel oder Wellenlinien, Gekritzel und Lasso-Gesten. Die auf Templates basierende Methode kann derartige Primitive nur bedingt behandeln. Jedes Element steht für eine eigene Gruppe Skizzen unterschiedlicher Ausführungen, die sich in der Anzahl Schlaufen, Krümmungen oder Kanten unterscheiden. Die Spezifikation multipler Templates kann die gängigen erwarteten Eingaben abdecken. Andernfalls müssen andere Methoden der Erkennung für die Identifikation dieser Elemente herangezogen werden. Ebenfalls durch andere Verfahren zu behandeln sind folgende, in einigen Tools verwendete Objekte: Polylinien, Polygone und Bezier-Kurven. Diese Elemente können durch Resampling oder Glätten - optional unterstützt von Methoden, die spezielle Segmentierungspunkte finden (wie in [190]) - aus einer Eingabe gewonnen werden. Der verwendete Ansatz, Muster zu erkennen, ist nicht für die Detektierung solcher generalisierter Primitive geeignet, ohne dass für spezielle Instanzen (etwa Rechtecke) Templates angelegt werden. Im Sketching-

Editor 'SkApp' sind daher Modi wählbar, mit denen Polylinien oder Freihand-Skizzen angelegt oder über Lasso-Gesten Selektionen vorgenommen werden können.

6.1.3 Evaluation

In der vorliegenden Arbeit wurde eine Nutzerstudie im Bereich der Erstellung von UML-Diagrammen und GUI-Entwürfen durchgeführt. Ziel war die Bewertung der Leistungsfähigkeit des Systems, aber auch der Nutzbarkeit der Eingabemethode für Skizzen. Weiterhin sollte das Verhalten der Nutzer bezüglich des Gebrauchs der erweiterten Möglichkeiten der Eingabe untersucht werden. Abbildung 6.2 zeigt die Anwendung 'SkApp' und interpretierte Skizzen, so wie sie in Abbildung 6.1 demonstriert wurden.

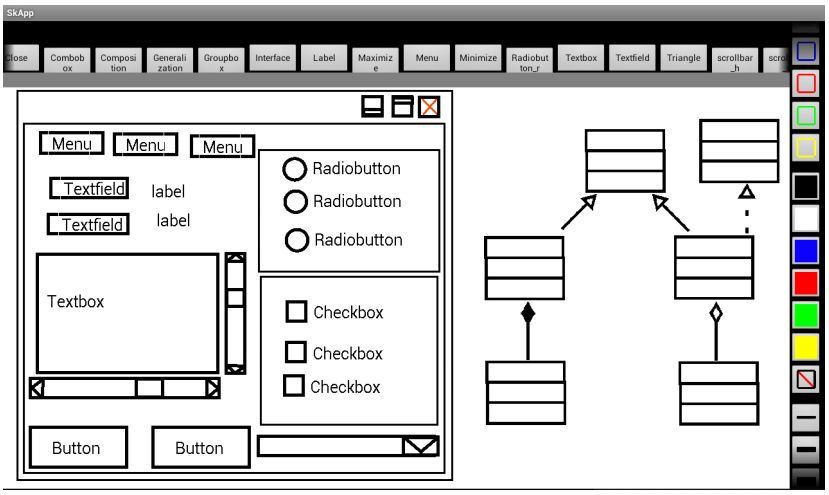


Abbildung 6.2: Interpretierte Skizzen im Bereich GUI und UML. Die Elemente der kompletten Szene wurden per Gesten im Modus für Skizzen eingegeben. Für jedes grafische Element wurden verschiedene Gesten im Voraus spezifiziert.

Die Anwendung enthält im oberen Bereich eine Reihe mit Schaltflächen, die jeweils ein einzelnes Primitiv repräsentieren. Ein langer Tap auf einer Schaltfläche zeigt eine Vorschau des Primitivs zusammen mit einer Geste, mit der es sich aufrufen lässt. Auf der rechten Seite sind Optionen für die Auswahl von Farben und Linientypen verfügbar. Im Vorfeld der Nutzerstudie wurden 19 Primitive (aus Abbildung 6.2) für UI-Elemente (Schalftfläche, Checkbox, Combobox, Groupbox, Label, Minimieren, Menü, Maximieren, Radiobutton, Schließen, horizontale Scrollbar, vertikale Scrollbar, Textbox, Textfeld) und Klassendiagramme (Aggregation, Klasse, Komposition, Generalisierung, Interface) zusammen mit ihren assoziierten Gesten spezifiziert. Die Primitive wurden innerhalb von 'SkApp' erstellt und Parameter, die bestimmen, wie sie auf Variationen in der Eingabe bezüglich Skalierung und Rotation ansprechen sollen, wurden konfiguriert. Alle UI-Elemente behalten ihre Orientierung (keine Rotation) und die meisten dieser Elemente skalieren nicht (Schaltflächen, Checkbox, Label, Maximieren, Menü, Minimieren, Radiobutton, Schließen), während manche in einer Dimension (Combobox, horizontale Scrollbar, vertikale Scrollbar, Textfeld) und wenige in beiden (Groupbox,

Textbox) skalieren. Alle Relationen eines Klassendiagramms dürfen frei skalieren und rotieren, während eine Klasse immer eine feste Größe und Orientierung hat.

Die im Vorfeld spezifizierten Gesten wurden unter der Annahme erstellt, dass eine generelle Top-Down und Links-Rechts Eingabe beim Skizzieren vorherrscht. Rechtecke können demnach mit einem Strich, der in der oberen linken Ecke beginnt und in zwei mögliche Richtungen verlaufen kann, gezeichnet werden. Eine sequenzielle Multi-Touch Alternative ist die Eingabe mit jeweils zwei parallelen Strichen von oben nach unten und links nach rechts. Da die Zeit beim Klassifizieren nicht berücksichtigt wird, können die einzelnen Striche aber auch in jedem beliebigen zeitlichen Versatz eingegeben werden. Alle spezifizierten Gesten korrespondieren in visueller Hinsicht mit den Abbildung 6.1 angedeuteten Variationen. Für jedes Primitiv wurde für jede Variation nur jeweils ein Template abgelegt. Insgesamt wurden zwei Sets mit 28 Gesten für die GUI-Elemente und 17 Gesten für die UML-Elemente vordefiniert.

Testumgebung

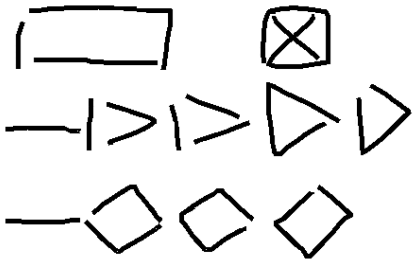
Für die Tests wurden 8 Probanden - 4 männliche, 4 weibliche, alle rechtshändig, ausgebildet in der angewandten Informatik und vertraut mit UML- und GUI-Elementen - gebeten, die Skizzen aus Abbildung 6.2 zu zeichnen. Die Tests wurden auf einem Android-Tablet des Typs Motorola Xoom und durch Eingaben per Finger ausgeführt. Alle Teilnehmer bekamen ein Merkblatt, auf welchem die Primitive und jeweils eine zugewiesen Geste zur Demonstration des Konzeptes aufgeführt waren. Aus den Abbildungen ging nicht hervor, in welcher Richtung sie eingegeben wurden und die Teilnehmer wurden angehalten, ihren eigenen Stil für die Eingabe anzuwenden. Es wurde darauf hingewiesen, dass Eingaben auch unter gleichzeitiger Benutzung mehrerer Finger möglich sind. Im Fall, dass ein nicht durch die Templates abgedeckter Stil der Eingabe versucht wurde, wurden die Teilnehmer instruiert, eine eigene Version der Geste mit dem entsprechenden Primitiv zu verknüpfen. Wurden solche zusätzlichen Spezifikationen nötig oder traten Fehlinterpretationen auf, musste die Eingabe wiederholt werden. Ein Test war mit dem Anfertigen der vollständigen Skizze abgeschlossen. Um im Voraus zu wissen, welches Primitiv als nächstes eingegeben wird, wurden die Teilnehmer gebeten, dieses jeweils anzukündigen (Thinking Aloud). Es wurden Daten bezüglich der verwendeten Stile, Fehlklassifikationen und hinzugefügten Gesten gesammelt.

6.1.4 Resultate

In Abbildung 6.3 sind die Resultate der Untersuchung bezüglich Fehlinterpretationen der Eingaben und notwendiger Spezifikationen zusätzlicher Templates aufgeführt.

Fehl-Interpretationen bei 208 + 88 eingegebenen Elementen	Button	Checkbox	Groupbox	Radiobutton	Textfield	Aggregation	Generalization	durch Teilnehmer
Button			8					5
Checkbox				8				4
Combobox					1			1
Groupbox	4							3
Radiobutton		2						2
Scrollbar	1							1
Aggregation							2	1
Generalization						5		2

(a) Fehlinterpretationen



(b) Zusätzliche Spezifikationen

Abbildung 6.3: Fehlinterpretationen (Tabelle) von Eingaben (Zeile) und die Anzahl der Teilnehmer für welche diese beobachtet wurden. Auf der rechten Seite sind Repräsentanten der zusätzlich von Nutzern während der Evaluation angelegten Templates aufgeführt.

Definitionen neuer Templates wurden für die Schaltfläche Schließen (ein Nutzer) und für Pfeilköpfe, die in zwei Strichen gezeichnet wurden (zwei Nutzer für das Dreieck und zwei für den Rhombus), nötig. Zwei Nutzer (chinesischer Herkunft) definierten alle Elemente, die ein Rechteck enthalten, neu, um dieses durch drei Striche eingeben zu können (siehe rechte Seite der Abbildung 6.3). Zusätzliche Definitionen wurden auch deshalb vorgenommen, weil sich eigene Eingaben von den vordefinierten Templates bezüglich der Richtung beim Zeichnen unterschieden. Zum Beispiel wurde das 'x' in der Schließen-Schaltfläche deshalb einmal neu spezifiziert. Die originalen Templates sahen nur Zeichnungen der Pfeilköpfe mit jeweils einem Startpunkt vor. Hier waren ebenfalls neue Templates nötig, für drei Nutzer allerdings schien der Startpunkt von der Orientierung der Eingabe abhängig. Ein Nutzer zeichnete die Striche der Scrollbar in zueinander entgegengesetzten Richtungen und den Schaft von Pfeilen in sich von der Pfeilspitze entfernender Richtung. Obwohl alle Nutzer anmerkten, dass sie die Eingabe mit einem Stift bevorzugen würden, um Verdeckungen durch den Finger zu vermeiden, wurde Multi-Touch sehr konsequent eingesetzt. Nur ein Nutzer machte nicht davon Gebrauch. Zwei Teilnehmer zeichneten die Scrollbars unter Verwendung von Multi-Touch, vier Teilnehmer die inneren Linien von Textboxen und Klassen, davon drei ebenso für Scrollbars. Ein Teilnehmer zeichnete nur die äußeren Rechtecke der Textbox und der Combobox mittels Multi-Touch.

Die Klassifikation erzielte insgesamt eine Erkennungsrate von 88,5% für die GUI-Elemente und 92% bei UML.[65] Die Elemente Schaltfläche und Groupbox wurden oft

[65] Anzumerken ist allerdings, dass, bedingt durch die vorgegebene Aufgabe, unterschiedlich viele

fehlgedeutet, da sie jeweils über ein Rechteck spezifiziert sind, welches sich im Verhältnis zwischen Höhe und Breite unterscheidet. Werden diese Verwechslungen ignoriert, so steigt die Genauigkeit auf 94,2% für das Set der GUI-Elemente.

6.1.5 Diskussion und Ausblick

Die Pilotstudie zeigte, dass die Klassifikationsmethode in der Lage ist, im Bereich des Skizzierens gebräuchliche Primitive zu erkennen und gleichzeitig Multi-Touch Eingaben zuzulassen. Der Nachteil der expliziten Segmentierung durch den Nutzer wird durch die Flexibilität in der Eingabe und die Möglichkeit, eine Vielzahl komponierter, komplexer Skizzen zu erkennen, kompensiert. Der trainierbare Ansatz erlaubt die Individualisierung bezüglich verschiedener Stile der Nutzer, aber auch die Anpassung an unterschiedliche Domänen, hier am Beispiel der Erstellung von UML-Diagrammen und GUI-Entwürfen aufgezeigt. Die Notwendigkeit, domänenabhängige Objekte in einer speziellen Beschreibungssprache anzulegen oder anderweitig komplexere Skizzen aus einer vordefinierten Auswahl an Primitiven zu komponieren, wird damit umgangen. Die nicht-visuelle Online-Klassifikation arbeitet im Gegensatz zu Regel-basierten Ansätzen nicht mit harten Schwellwerten und benötigt keine Behelfseingaben, um etwa die Verbindungen von Strichen herzustellen.

Obwohl verschiedene Nutzer unterschiedliche Stile beim Skizzieren verfolgten, waren diese jeweils konsistent. Das bestätigt die Beobachtung von Sezgin und Davis [189]. Auch die Resultate in [26] zeigen, dass sich Nutzer in ihren Vorstellungen, wie UI-Elemente zu skizzieren sind, unterscheiden. Das Verhalten zweier chinesische Teilnehmer legt kulturelle Unterschiede nahe, da beide, im Gegensatz zu den anderen Teilnehmern, gewohnt waren, alle Rechtecke mit drei Strichen zu zeichnen. Ein (Online-)Erkenner für Skizzen sollte demnach nicht nur an verschiedene visuelle Repräsentationen, sondern auch an die Art, diese zu erstellen, anpassbar sein. Derartige Anpassung können einfach über die Bereitstellung zusätzlicher Trainingsdaten umgesetzt werden. Der kritischste Aspekt ist allerdings, dass die Wahl des Anfangspunktes eines Striches offenbar von der Rotation mancher Skizzen, in diesem Fall der Pfeile, abhängig ist. Obwohl solche Varianten auch automatisch von Templates abgeleitet werden können, können sie im Fall komplexer Skizzen zu einer kombinatorischen Explosion führen. Eine bessere Alternative ist die Anpassung dahingehend, dass jeder Vergleich eines Tokens - für seine Form und Struktur - jeweils in beide mögliche Richtungen vorgenommen und anschließend diejenige mit geringerer Distanz gewählt wird. So werden unter Verdoppelung der Laufzeit weiterhin wenige Templates (nur noch für jede mögliche Segmentierung) benötigt, das Training bleibt bequem.

Die Klassifikationsmethode benötigt nur wenige Trainings-Templates und eines für jede einzugebende Variante war zweckmäßig.[66] Eine Verbesserung der Ergebnisse der

Testfälle je Element vorlagen. Des Weiteren wurden alle Eingaben so lange wiederholt, bis die Szene fertig gestellt war, was sich nachteilig auf die gemessene Genauigkeit ausgewirkt hat.

[66] Zum Vergleich, SILK [111] nutzt 15-20 Templates für jedes der vier unterstützten Primitive. Drei

Klassifikation kann erreicht werden, wenn mehr als ein Template pro Variante spezifiziert wird und die Definitionen vom Nutzer selbst angelegt werden. Weiterhin werden Klassifikationen unter Invarianz gegenüber Skalierung, Rotation und Geschwindigkeit gegenüber der Eingabe vorgenommen, diese Eigenschaften aber als Parameter zurückgegeben, so dass klassifizierte Skizzen (konfigurierbar) verschoben, skaliert und rotiert werden, um bestmöglich der Eingabe zu entsprechen. Eine Verbesserung der Erkennungsgenauigkeit wird allerdings erwartet, wenn die Normalisierungen zum Erzeugen dieser Invarianzen abhängig vom konfigurierten Verhalten der Primitive eingesetzt werden.

Für den direkten Vergleich der Klassifikationsgenauigkeit mit in der Literatur vorgestellten Methoden eignen sich wenige Arbeiten. Manche Autoren stellen keine Testergebnisse zur Verfügung [87, 109, 40], andere berichten informelle Ergebnisse, schätzen die Klassifikationsrate [27, 34] oder stellen nur ungenügende Informationen zur Verfügung [38]. Vergleiche dreier verschiedener Ansätze der Klassifikation finden sich in [209], allerdings nur für drei einfache geometrische Formen (Dreieck, Rechteck, Ellipse). Yu und Cai [223] berichten einen Anteil von 98% korrekter Segmentierungen und eine darauffolgende Erkennungsgenauigkeit von 70% für ein ungenau beschriebenes Set verschiedener Formen und der Akzeptanz einer Interpretation, sobald nicht näher spezifizierte wesentliche Merkmale erkannt wurden. Sezgin et al. [190] erreichten eine Rate korrekter Segmentierungen in Linien und Bögen von 96% für zehn verschiedene Figuren, geben aber keine Erkennungsrate des Regel-basierten Ansatzes an. Ein auf der Arbeit aufbauender Ansatz mit HMM erreichte eine Erkennungsrate von 96.5% für zehn komplexere Symbole aus verschiedenen Domänen, die für sechs Stile, jeweils mit zehn Templates angelernt wurden [189]. Apte et al. [9] berichten für ihren auf heuristischen Regeln basierenden Ansatz eine Genauigkeit von 98% für sechs verschiedene Formen. Die ähnliche Methode in [149] weist eine Genauigkeit von 98,5% für acht primitive und komplexere Formen auf. In [58] werden durch Fuzzy Logic 95,8% korrekte Klassifikationen für zwölf Primitive erreicht.

Ein zum vorliegenden Testaufbau vergleichbares Szenario, in dem GUI-Mock-Ups skizziert werden, findet sich in [112]. Die Autoren geben eine erreichte Erkennungsrate von 69% der Eingaben an. Die Klassifikation erfolgt unter Verwendung das Bayes'schen Klassifizierers aus [169] bei 15-20 Templates pro einfachem Primitiv und zusätzlichen Regeln, um zusammengesetzte Elemente zu erkennen. In [191] werden die Regeln so modifiziert, dass Abgrenzungen statistisch getroffen werden können. Es zeichnet sich ab, dass Resultate ähnlich denen des hiesigen Ansatzes möglich sind, wohl aber große Trainingsmengen nötig werden.

In der prototypischen Anwendung 'SkApp' werden Skizzen in einem speziellen Modus eingegeben, der aktiviert ist, solange zwei Finger der linken Hand die Oberfläche berühren. Auf diese Art wird dem Nutzer explizit die Gruppierung der Eingabe über-

weitere, in [58] verglichene Methoden, benötigen mehr als 40 Templates, um eine vergleichbare Genauigkeit bei der Klassifikation von 12 Primitiven zu erreichen.

lassen. Eine mögliche Alternative wäre es, nachträglich per Lasso-Geste eine Auswahl zu treffen, die als Skizze interpretiert werden soll. Das würde es erlauben, zunächst mehrere Elemente oder eine komplette Szene fertig zu stellen und den expliziten Befehl zur Interpretation von der Eingabe zu trennen. In der Literatur beschriebene Strategien oder kombinatorische Suchen zur Gruppierung können ebenfalls Anwendung finden. Weitere Merkmale einer skalierbaren Anwendung sollten die Vermeidung zu ähnlicher Definitionen für verschiedene Objekte und die Möglichkeit der manuellen Auswahl aus unterschiedlichen Interpretationen oder gar eine Abweisung der Eingabe sein. Letzteres kann über die Festlegung eines Schwellwertes gelöst werden.

6.2 Texteingabe mit einem Multi-Touch Gestenalphabet

Eine weitere mögliche Anwendung für Multi-Touch Eingaben sind sogenannte Gestenalphabete. Als eine Abstraktion zur Handschrift repräsentieren symbolische Gesten Buchstaben eines Alphabetes und somit eine Methode der platzsparenden Texteingabe (etwa auf mobilen Geräten). An dieser Stelle soll ein neuartiges Multi-Touch Gestenalphabet vorgestellt werden. In einem Vergleich mit einer Single-Touch Variante werden dessen Charakteristiken und die Nutzbarkeit ebenso untersucht, wie die Möglichkeiten der schnelleren Eingabe. Der Inhalt des vorliegenden Kapitels entstand in Zusammenarbeit mit Anja Fibich und wurde in [179] veröffentlicht.

6.2.1 Motivation und State of the Art

Die Eingabe von Text mittels eines Gestenalphabetes ist keine der effizientesten Optionen. Dennoch gibt es Anwendungsbereiche, in denen diese Technik Vorteile bietet. So kann eine Eingabe mit wenig 'focus of attention'[67] (FOA) umgesetzt werden. Bei Eingaben kurzer Phrasen während der Arbeit mit einer berührungssensitiven Oberfläche kann der Wechsel zu alternativen Techniken aufwendiger sein.

Für die Eingabe von Texten in mobilen Umgebungen verwendet die gebräuchlichste Methode 12 Hardware-Tasten und erreicht eine Eingabegeschwindigkeit von etwa 10wpm[68], wobei bei T9-Unterstützung 20wpm möglich sind [127]. Als Ersatz für QWERTZ Hardware-Tastaturen sind platzsparende virtuelle Tastaturen im Bereich der Smartphones ebenfalls gebräuchlich. Daneben existierende lautlose Eingabesysteme umfassen Handschrift und die symbolische Eingabe durch Gestenalphabete. Während die Handschrift für die meisten Nutzer natürlich ist, ist sie immer noch eine Herausforderung im Gebiet der Mustererkennung. Virtuelle Tastaturen haben den Vorteil, recht

[67] Der Ausdruck 'focus of attention' (FOA) wird in [127] eingeführt und beschreibt den Grad der Aufmerksamkeit, den eine Aufgabe vom Nutzer abverlangt. Beispielsweise verlangt blindes Schreiben eines eingeprägten Textes weniger FOA, als wenn dieser mit einer Eingabemethode verfasst wird, die zusätzlich die Beobachtung des Schreibbereiches verlangt [127].

[68] Die Einheit wpm (engl. words per minute, Wörter pro Minute) bezieht sich in der englischsprachigen Literatur auf Wörter der Länge von fünf Zeichen, in der deutschen Sprache sind Wörter durchschnittlich sechs Zeichen lang [127].

schnell zu sein - konservative Schätzungen mittels Modellen sagen wenigstens 28wpm voraus [227] - sind aber unnatürlich und anfällig für Parallaxe-Fehler.

Existierende Weiterentwicklungen der reinen Eingabe auf virtuellen Tastaturen verknüpfen diese mit gestischer Interaktion. In einigen Arbeiten [152, 130, 131] werden Symbole neu geordnet oder gruppiert, um Strukturen für schnelles, hierarchisches Anwählen durch gestische Eingaben zu erzeugen. In [155] werden Buchstaben in abstrakte, wiederkehrende, grafische Bestandteile zerlegt, um diese in hierarchischen Strukturen zu definieren, deren Navigation die Buchstaben produziert. Andere Methoden erlauben gestische Eingaben direkt auf virtuellen Tastaturen, um Buchstaben zu Wörtern zu verknüpfen. Obwohl auf virtuellen Tastaturen ausgeführte Gesten die Effizienz im Allgemeinen steigern können, gibt es wesentlichen Spielraum für Verbesserungen durch spezielle Tastaturlayouts [165]. In [19] werden derartige Eingaben sogar in beidhändiger Ausführung unterstützt. Durch die Interpretation der gezeichneten Formen auf Basis eines großen Wörterbuches verlangt der Ansatz in [107] kein genaues Treffen der Tasten. All diesen Ansätzen mangelt es an der Möglichkeit, blind (ohne Blickkontakt) zu schreiben. Im Allgemeinen verlangen Methoden, die die Form interpretieren - wie in [107] - dem Nutzer weniger Aufmerksamkeit ab, da sie erlauben, Präzision gegen Geschwindigkeit einzutauschen [127]. Die notwendigerweise genaue Ansteuerung bei Eingabe mit einer Bildschirmtastatur, die hierarchische Auswahl oder Handschrift benötigen allerdings die Fokussierung des Eingabebereiches. Blinde Eingaben hingegen würden es erlauben, unterstützenden Techniken wie der Autovervollständigung von Wörtern mehr Aufmerksamkeit zu widmen [125].

Reine Gesten stellen eine Alternative bereit, die auch den Eingabebereich effizient ausnutzt, indem Buchstaben übereinander geschrieben werden. In [67] wurde dieses Konzept unter dem Begriff 'heads-up writing' eingeführt und behauptet, dass dies, verglichen mit Handschrift, schonender für das Handgelenk ist. Ein weiterer Vorteil des Schreibens diskreter Symbole in einzelnen, voneinander unabhängigen Bewegungen ist die Möglichkeit, Text mit weniger Aufmerksamkeit für den Bildschirm oder ohne visuelles Feedback zu verfassen. Gestenalphabete liefern diese Form der Eingabe, indem Buchstaben in verschiedenen Abstraktionsgraden zu diskreten Symbolen überführt werden. Der Preis einer größeren Abstraktion in Form des Verzichts auf intuitivere Symbole kann im Gegenzug deren robustere Erkennung bewirken. 'EdgeWrite' [215] und 'Minimal Device Independent Text Input Method' (MDITIM) [84] ermöglichen eine robuste Erkennung, indem nur kurze Sequenzen gerader, gerichteter Striche verwendet werden. Dadurch sind beide Methoden unabhängig von präzisen Eingabegeräten und können auch etwa mittels Eye-Tracking genutzt werden. Etwas mehr Anlehnung an lateinische Buchstaben findet sich im 'Unistrokes'-Alphabet [67]. Das 'Graffiti'-Alphabet (im Palm OS) ahmt hingegen die meisten lateinischen Buchstaben nach und ist daher intuitiver zu erfassen. Der Nachfolger[69] 'Graffiti 2', obwohl anfällig für Fehler (19%, bei davon etwa

[69] 'Graffiti 2' enthält im Gegensatz zu den anderen vorgestellten Varianten (wenige) Multi-Stroke

12% aufgrund von Fehlklassifikationen), wird im Vergleich zur virtuellen Tastatur von Nutzern bevorzugt (unter Verwendung von Autovervollständigung der Wörter), da es als intuitiver und benutzbarer sowie weniger erschöpfend wahrgenommen wird [102]. Interessanterweise kamen diese Resultate trotz einer langsamen Eingaberate von 9 wpm - etwa zwei Drittel der mit der virtuellen Tastatur erreichten Geschwindigkeit - zustande.

Das an dieser Stelle neu eingeführte Gestenalphabet aus Multi-Touch Symbolen erlaubt vielfältigere Eingaben und die Anlehnung an lateinische Buchstaben bei - im Vergleich zu vergleichbaren Alphabeten - Verbesserung der Schreibgeschwindigkeit. Multi-Touch Symbole können in einem Zug eingegeben werden und benötigen, wie Single-Touch Gesten, weniger FOA. Durch Verwendung des trainierbaren Gestenerkenners können verschiedene Gestenalphabete direkt und fair miteinander verglichen werden. Die Invarianz der Erkennung gegenüber Größe und Geschwindigkeit der Eingabe erlaubt die Skalierung des Systems mit der Erfahrung seiner Nutzer. Dennoch ist ein nachgelagertes Training jederzeit möglich und bietet auch die Option der Individualisierung, indem eigene Symbole aufgenommen werden.

6.2.2 Umsetzung einer Multi-Touch Texteingabe

Das entwickelte Multi-Touch Gestenalphabet basiert auf den beiden 'Graffiti'-Versionen. Um die Schreibgeschwindigkeit zu erhöhen, wurden die Multi-Stroke Symbole entfernt und Multi-Touch Symbole für komplexere Buchstaben eingefügt. Der Abstraktionsgrad ist niedriger als bei 'Unistrokes' und die meisten Symbole sind an Großbuchstaben (Blockschrift) des lateinischen Alphabets angelehnt. Effizienz und Intuitivität werden gegeneinander abgewogen, indem Buchstaben abstrahiert werden, die beim normalen Schreiben mit einem Stift durch mehrere Striche entstehen oder die eine visuelle Darstellung haben, die schneller mittels Multi-Touch erzeugt werden kann. Für mehrere Buchstaben wurden verschiedene Varianten eingeführt, die für schnellere Eingaben oder je nach gewünschtem Schreibstil gewählt werden können. Ein zweites Referenzalphabet aus Single-Touch Symbolen dient dem Vergleich und unterscheidet sich nur in den Symbolen, die im ersten Alphabet mittels Multi-Touch umgesetzt wurden. Abbildung 6.4 veranschaulicht beide Alphabete.

Auf der linken Seite ist das Single-Touch Alphabet zu sehen. Es besteht aus den Buchstaben aus 'Graffiti' 1+2 ohne deren Multi-Stroke Symbole und enthält die Erweiterung um vereinfachte Alternativen für manche Buchstaben. Auf der rechten Seite ist das Multi-Touch Alphabet aufgeführt, bei dem ausgewählte Symbole durch Multi-Touch Versionen ersetzt wurden. Die verbleibenden Single-Touch Gesten werden von beiden Alphabeten für die gleichen Buchstaben geteilt.

Für die Klassifikation der Eingaben beider Alphabete wurde abermals der in Kapitel 4 vorgestellte Gestenerkenner verwendet. Eine Anpassung fand nur insoweit statt, als dass die Rotationsinvarianz auf einen Bereich von 20° begrenzt wurde, um Gesten ver-

Symbole.

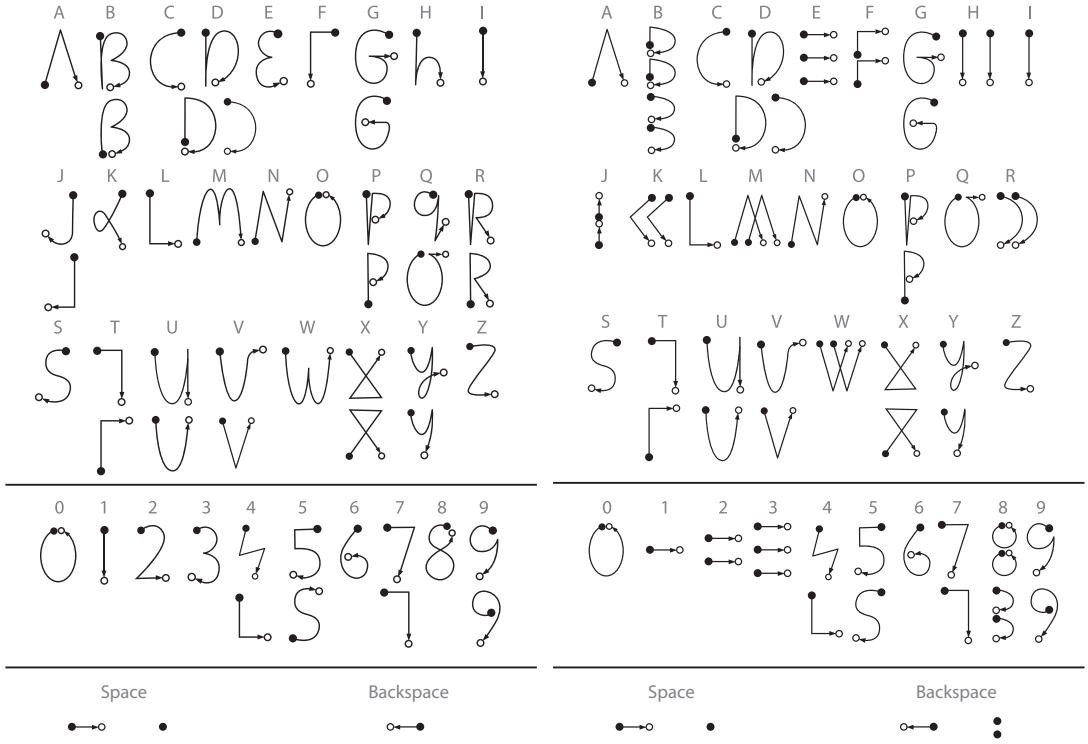


Abbildung 6.4: Illustration der Eingabe für Single-Touch (links) und Multi-Touch Alphabete (rechts). Die Symbole werden jeweils in einem Stroke eingegeben, können allerdings (nur auf der rechten Seite) aus den Trajektorien von bis zu drei gleichzeitigen Kontakten entstehen. Ein schwarzer Punkt veranschaulicht eine Berührung, ein Pfeil deren Bewegung und ein leerer Kreis das Beenden eines Kontaktes.

wenden zu können, die sich größtenteils nur hinsichtlich der Orientierung unterscheiden. Die Rückweisung von Eingaben wäre über eine einfache Definition eines Schwellwertes möglich, wurde aber nicht angewendet.

6.2.3 Evaluation

Eine erste Version des vom Autor entwickelten Multi-Touch Alphabetes wurde in einer Diplomarbeit [56] mit 20 Nutzern evaluiert. Das Alphabet wurde anhand von Aufgaben, in denen Text abgeschrieben werden sollte[70], mit 'Graffiti 2' verglichen ('Between Group Design'). In einem anschließenden Fragebogen befanden die Nutzer das Multi-Touch Alphabet insgesamt brauchbarer und im Detail lernförderlich. Weiterhin war die Eingabe der Multi-Touch Symbole während der Trainingsphase schneller und, bei Mittelung über die gleichen Test-Sätze, signifikant schneller in den Tests.[71] Jedoch enthält das 'Graffiti 2'-Alphabet Multi-Stroke Symbole für die Buchstaben 'I','K','T','X' und die Ziffer '4', deren korrekte Erkennung die Umsetzung eines Timeouts benötigt. Um bessere Vergleichbarkeit zu gewährleisten, wurden beide Alphabete modifiziert. Alle

[70] Derartige 'text-copy' Aufgaben werden gegenüber 'text-creation' Aufgaben bevorzugt, um die mentale Belastung und Fehler durch Vergessen oder falsche Schreibweise zu Minimieren [127].

[71] Insbesondere die Eingabe der Buchstaben 'E','H','M','W' erwies sich als signifikant schneller.

Symbole in beiden Alphabeten sind auf einzelne Strokes beschränkt. Im Multi-Touch Alphabet wurden die ehemals Single-Touch Symbole für die Buchstaben 'R' und 'K' durch abstraktere Multi-Touch Symbole ersetzt.[72]

Weitere Erkenntnisse sollten insbesondere hinsichtlich der folgenden Fragen gewonnen werden:

- Unterscheiden sich die Bewertungen der Nutzer zu beiden Alphabeten bezüglich Intuitivität, Fehlerrate und Zufriedenheit?
- Wird die Texteingabe per Gesten, im Vergleich zu konservativen Methoden, als nützlich wahrgenommen?
- Werden abstrakte Symbole bevorzugt oder eher zur Handschrift ähnliche?
- Ist es im Interesse der Nutzer, eigene Gesten zu spezifizieren und das Alphabet mitzugestalten?

Die zweite Evaluation wurde ebenfalls mit Hilfe einer in [56] entwickelten SMS-Anwendung durchgeführt. Die Applikation erlaubt es, Gestenalphabete anzlegen, zu lehren und Textbeispiele für Testzwecke zu präsentieren. Die verwendeten Phrasen orientieren sich an dem in [126] präsentierten Set und wurden in [56] ins Deutsche übertragen und modifiziert, um sie bezüglich der deutschen Buchstabenhäufigkeit anzupassen. Zu der so entstandenen Auswahl von 50 Phrasen fanden zudem noch ebenfalls in [56] erstellte Phrasen Verwendung, welche auch Ziffern enthalten.

[72] Vorabtests deuteten an, dass diese schwerer zu merken sind, sie wurden aber beibehalten, um zu testen, ob potenziell schneller einzugebende Abstraktionen oder intuitive Symbole bevorzugt werden.

Abermals in einem 'Between Group Design' wurden jeder der zwei Gruppen je ein Alphabet und die folgende Testroutine zugeteilt:

- Kurze Einführung zum Alphabet, der Notation und des Ablaufs der Evaluation.
- Training:
  1. Visuelle Abbildungen der Symbole werden vom Nutzer mit den Fingern nachgefahren.
  2. Merkhilfen in Form von Icons der einzugebenden Symole werden in einer zweiten Trainingsphase angezeigt. Die Zeichen werden in zufälliger Reihenfolge und jedes dabei je zweimal präsentiert. Eingegebene Gesten werden, wenn sie als erwartete Eingabe erkannt wurden, automatisch als zusätzliche Templates gespeichert, um die Klassifikation robuster gegenüber unterschiedlichen Nutzern zu gestalten.[73]
  3. In der dritten Trainingsphase werden die Symbole zu Buchstaben in randomisierter Reihenfolge und ohne visuelle Hilfe abgefragt. Nutzer sollen an dieser Stelle die von ihnen präferierte Variante eingeben, falls Alternativen möglich sind. Bei fünfmaliger Fehleingabe wird über die Präsentation von Icons eine Hilfestellung gegeben.
- Test: Die Nutzer werden gebeten, den vorgegebenen Text so schnell und genau wie möglich einzugeben. Korrekturen sind nur über die Rücknahmetaste oder Gesten erlaubt. Es ist möglich, über eine Schaltfläche eine Erinnerungshilfe mit einer Übersicht der Symbole aufzurufen. Insgesamt gibt jeder Teilnehmer 16 Phrasen mittels Gesten ein.
  1. Ein Pangramm[74] wird per virtueller Tastatur vom Nutzer eingegeben.
  2. Zehn zufällig gewählte Phrasen, bestehend nur aus Buchstaben, werden per Gesten eingegeben.
  3. Aus dem Set der 20 Phrasen, die jeweils Buchstaben und Zahlen enthalten, wird für fünf zufällig gewählte die Eingabe durch Gesten vom Nutzer abgefragt.
  4. Das Pangramm wird erneut eingegeben, diesmal mittels Gesten.
- Die Nutzerstudie wird mit einem Fragebogen über wahrgenommene Nutzbarkeit und subjektive Präferenzen abgeschlossen. In einem Freifeld können zusätzliche Kommentare angegeben werden.

[73] Eine solche Individualisierung (Adaptierung) ohne das Wissen des Nutzers hat auch Schwächen. Die Templates entsprachen nicht immer dem Stil des Nutzers. Obwohl in der Einführung darauf hingewiesen wurde, dass die Trainingsphase nicht zum Austesten genutzt werden soll, taten das manche Nutzer. Andere gestalteten die Eingaben sehr vorsichtig und akribisch, so dass die Templates auch hier nicht die spätere Eingabetechnik widerspiegelten.

[74] Die klein geschriebene Version des Satzes 'Franz jagt im komplett verwahrlosten Taxi quer durch Bayern'.

Ebenso wie in der ersten Evaluation [56], wurden Teilnehmer einer Gelegenheitsstichprobe einbezogen. Von den 12 Teilnehmern (7 weiblich, 5 männlich, Alter 19-34) waren neun Studenten oder ehemalige Studenten der Informatik oder ähnlichen Studiengängen. Die meisten gaben an, täglich etwa 1-5 SMS zu schreiben. Je ein Teilnehmer beantworte diese Frage mit 0, 6-10 und 11-20. Sechs Teilnehmer nutzten virtuelle Tastaturen beim Verfassen von SMS, vier die 12er und zwei die 'QWERTZ' Hardware-Tastatur.

6.2.4 Resultate

In Anbetracht der Eingabegeschwindigkeit (in cpm, wie in [127] definiert) beim Schreiben des Pangramms war die Multi-Touch Gruppe (M=36,93; SD = 4,63; N=6) signifikant schneller (Zweistichproben-t-Test; t=3,1; DF = 10; p = 0,01) als die Single-Touch Gruppe (M = 28,13; SD = 5,19; N=6).[75] In der verbleibenden Testphase zeigte der gleiche Test eine starke Tendenz zur schnelleren Eingabe bei Multi-Touch (t = 2,08; p = 0,06).

Die kumulierte Fehlerrate (aller Fehleingaben) während der dritten Trainingsphase (ohne Hilfe) belief sich auf 13,85% für Multi-Touch und auf 17,44% für Single-Touch. Allerdings zeigte sich keine Signifikanz (Zweistichproben-t-Test).

Im Fragebogen wurden Antworten bezüglich subjektiver Empfindungen der Fehlerrate, des Zeitdrucks, des Stresses und der Leichtigkeit der Erinnerung beim Schreiben mit Gesten ermittelt. Auf einer fünf-stufigen Likert Skala (1 am schlechtesten bis 5 am besten) wurden alle Items leicht höher für Multi-Touch und im Durchschnitt zwischen 3 und 4 beantwortet. Der höchste Unterschied (3,3 zu 4) zeigte sich im subjektiven Empfinden der Fehlerrate bei der Erkennung.

Im direkten Vergleich der Eingabemethoden durch Gestenalphabete mit der gewohnten Texteingabemethode zeigte sich eine Präferenz der letzteren. In beiden Gruppen erzielten die Antworten zur bevorzugten Technik, Fehlerrate, Stress und Konzentration durchschnittlich zwischen 3 und 3,83 Punkte auf der fünf-stufigen Likert Skala. Höhere Werte beschreiben dabei Tendenzen zur Bevorzugung der gewohnten Methode der Texteingabe. Unterschiede zwischen den Gruppen sind allerdings vernachlässigbar gering (0,16 oder weniger) und ohne Signifikanz. Eine Ausnahme trat für die Frage nach wahrgenommenem Spaß auf, bei der die durchschnittliche Bewertung innerhalb der Multi-Touch Gruppe mit 2 besser war als die im Vergleich erreichte Bewertung von 3 in der Single-Touch Gruppe.

In der letzten Frage des Interviews wurden alle Teilnehmer mit den Illustrationen beider Alphabete aus Abbildung 6.4 und einer kurzen Erläuterung der Unterschiede konfrontiert. Bis dahin wurde nicht mitgeteilt, dass eine vergleichende Evaluation stattfindet. Gefragt nach dem bevorzugten Alphabet, nur unter der gegebenen Auswahl,

[75] Zum Vergleich: die virtuelle Tastatur erreichte beim gleichen Pangramm eine durchschnittliche Geschwindigkeit von 96,69cpm.

wählten drei Teilnehmer (davon zwei in der Multi-Touch Gruppe) das Multi-Touch Alphabet und neun Teilnehmer die Single-Touch Variante.

Abbildung 6.5 zeigt die Verteilung der Antworten bei der Frage nach den bevorzugten Varianten alternativer Eingabemöglichkeiten.

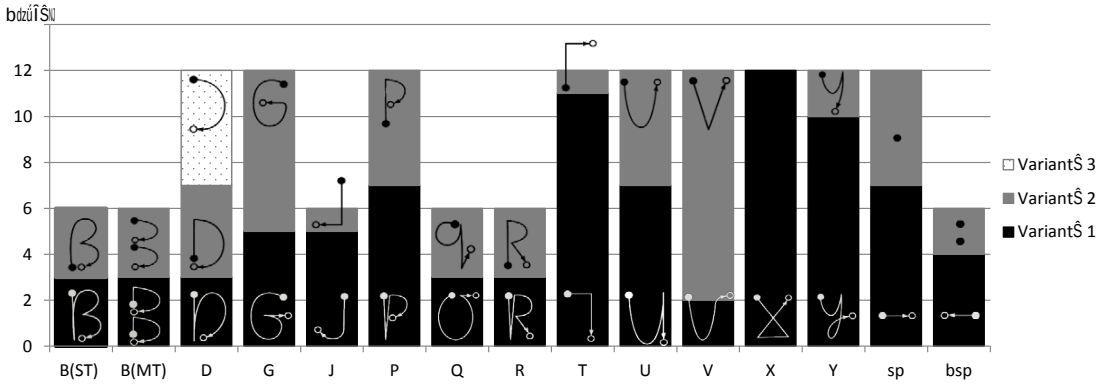


Abbildung 6.5: Anzahl der Nutzer, die eine der zur Verfügung gestellten Eingabemethoden für unterschiedliche Symbole bevorzugen. Anzumerken ist, dass nicht alle Symbole in beiden Alphabeten verfügbar sind.

Neben offensichtlichen Tendenzen zu speziellen Versionen von 'T', 'X' und angedeuteter Präferenzen bei 'J', 'V', 'Y' sind die meisten Symbole - einschließlich für Leerzeichen und Backspace - keiner allgemein bevorzugten Geste zugeordnet. Diese Erkenntnis wird durch die Logging-Daten bestätigt, die außerdem zusätzliche Informationen zu den Ziffern bereitstellen. So werden die '8' (Multi-Touch) und '9' in ihrer bildhaften Variante bevorzugt, während die '4' in einem Drittel der Fälle auch in ihrer abstrakten Version eingegeben wurde. Für Leerzeichen und Backspace ergaben sich ebenso die in Abbildung 6.5 angedeuteten Verhältnisse.

6.2.5 Diskussion und Ausblick

Die Resultate bestätigen, dass, neben zusätzlichen Optionen zur Eingabe, ein Geschwindigkeitsgewinn erreicht werden kann, wenn Multi-Touch einbezogen wird. Das Multi-Touch Alphabet selbst hat allerdings noch Potenzial zu weiterer Verbesserung. Trotz besserer, vergleichender Bewertung, würden es Nutzer nicht wählen, wenn sie nur die Wahl zwischen den vorgestellten Single-Touch und Multi-Touch Versionen bekommen. Gründe dafür bieten die Kommentare der Testteilnehmer. In drei Kommentaren aus der Multi-Touch Gruppe wurde die umständliche Eingabe für das 'R' kritisiert und in zwei Fällen die des 'K'. Für beide Buchstaben wurde den Logging-Daten zufolge auch die Erinnerungshilfe am häufigsten aufgerufen. Weiterer Missfallen wurde bezüglich der horizontalen Ausrichtung der Symbole für die Zahlen 1-3 und die generell häufigen Wechsel in der Eingaberichtung kommuniziert. Ein Teilnehmer bevorzugte eine Mischung aus beiden Alphabeten.

Die innerhalb des Fragebogens angegebene Verteilung der gewählten Stile in der Eingabe legt die Entwicklung eines Gestenalphabets nahe, welches wesentlich mehr Va-

riabilität in der Eingabe erlaubt, als aktuelle Alphabete zur Verfügung stellen. Eine wesentliche Schlussfolgerung ist der Bedarf nach vielfältigen Möglichkeiten der Individualisierung bei der Eingabe mittels Gestenalphabeten. Dieser Schluss wird durch die Antworten bestätigt, die auf die Frage nach dem Wunsch der Möglichkeit, eigene Gesten zu spezifizieren, gegeben wurden. Bis auf eine Ausnahme antworteten alle Teilnehmer in beiden Gruppen diesbezüglich positiv.

Die vorgestellten Konzepte bieten einen Ansatz, um beliebige Gestenalphabete durch kontinuierliche Modifikationen bestehender zu entwickeln. Die Gestenalphabete können fair miteinander verglichen werden, indem eine Klassifikation unabhängig von der konkreten Spezifikation möglich ist. Kriterien wie die Erlernbarkeit oder der Verlauf der Lernkurve können in einheitlichen Tests besser untersucht werden, als dies zur Zeit in der Literatur der Fall ist. Die Fehlerrate des Klassifizierers sollte für diese Zwecke angemessen sein und kann durch die Angabe zusätzlicher Templates weiter verbessert werden. Durch den Verzicht auf vordefinierte Templates kann außerdem eine Verzerrung des nachträglichen Trainierens in Richtung der ursprünglichen Spezifikationen vermieden werden.

Die Entwicklung eines benutzbaren, adaptierbaren Texteingabesystems benötigt allerdings noch Konzepte, um mehrdeutige Definitionen durch den Nutzer zu vermeiden und Erinnerungshilfen dynamisch aus den spezifizierten Templates abzuleiten. Außerdem sollten mittlerweile verbreitete Hilfemöglichkeiten und automatische Textkorrekturen zur Verfügung gestellt werden. Dennoch ist diese Art der Eingabe recht speziell und Anwendungen für Gestenalphabete fallen in eine Nische. Die Vermeidung des Wechsels zu anderen Eingabegeräten bei kurzen Phrasen, der reduzierte FOA oder die Unterstützung für spezielle Nutzergruppen sind Beispiele für einen sinnvollen Anwendungskontext. Der Autor stellt sich ein Texteingabesystem auf mobilen Geräten für blinde oder stark sehbehinderte Nutzer vor. Für diese Gruppe stellen Gestenalphabete auch unter dem Aspekt der erreichbaren Schreibgeschwindigkeit und der Möglichkeit selbst definierter Symbole eine aussichtsreiche Alternative dar. Möglicherweise lässt sich ähnlich den Konzepten aus [41, 180] ein (nicht notwendigerweise taktiler) Mechanismus zum Training von Gesten entwickeln, der neben dem Lernen auch als Erinnerungshilfe für blinde Nutzer anwendbar ist.

7

Fazit, kritische Reflexion und Ausblick

Die vorliegende Arbeit hatte zum Ziel, ein flexibles Werkzeug für die universelle Klassifikation planarer, symbolischer Gesten bereitzustellen. Der Aspekt der Vielseitigkeit wurde durch die Festlegung von Anforderungen bedacht, welche größtmögliche Freiheiten in der Eingabe bei allein durch Templates durchführbarem Training zulassen. Darüber hinaus gestellte Bedingungen an die Echtzeit-Fähigkeit des Verfahrens und die Erweiterung um Methoden, auch partielle Gesten zu erkennen, dienten ebenfalls der Sicherstellung eines breiten Anwendungsgebietes. Einen Einsatz der untersuchten Konzepte kann sich der Autor in den Bereichen der Mockup-Erstellung für UI-Designer, Nutzerstudien durch Usability-Experten und des Prototypings sowie der Aufwandsersparnis für Anwendungsentwickler vorstellen.

Bisher finden berührungsbasierte Interaktionen allerdings nur durch Point & Click, direkte Manipulationen, Multi-Stroke Skizzen und Single-Touch Gesten Anwendung. Falls auch zukünftig komplexere Mehr-Finger Eingaben keine Bedeutung erlangen, reduziert sich der wesentliche Beitrag der Dissertation auf die Taxonomisierung von Begrifflichkeiten im Kontext von Gesten, dem Vergleich in der Literatur verfügbarer Single-Touch Gestenerkenner und der Systematisierung zugrunde liegender Verfahren zur Merkmalsextraktion. Außerdem werden eigene, effiziente Methoden derartiger Gestenerkenner entwickelt, welche bessere Resultate liefern als aktuelle Klassifizierer.

Diese Arbeit basiert jedoch auf der Annahme, dass die Verfügbarkeit durchdachter und erprobter Interaktionstechniken die der ebensolche Eingaben interpretierenden Komponenten bedingt. Indizien für die Bestätigung der Annahme lieferten Tests mit zwei zur Demonstration der Zweckmäßigkeit komplexerer, gestischer Interaktionen entwickelten Anwendungen. Gleichzeitig sind die darin umgesetzten Konzepte, Multi-Touch

beim Skizzieren oder der symbolischen Texteingabe zu verwenden, ebenfalls neuartig. Der verfolgte Ansatz der Gestenerkennung zeigte sich zumindest im Rahmen des Proof of Concepts als robust, performant und mit hinreichender Genauigkeit. Mögliche Anpassungen wurden im Bezug zu beiden Anwendungen diskutiert. So wurden Invarianzen abgeschwächt, um eine größere Vielfalt der Gesten zuzulassen oder um der - im Gegensatz zu im Kontext von Skizzen üblichen - bildhaften Interpretation zu entsprechen. Diese Modifikationen lassen sich leicht übertragen, um auch eine Offline-Erkennung von Symbolen umzusetzen. Verfolgte Konturen können als Token gesehen werden und - unter Ignorierung der zeitlichen Relation - jeweils für beide mögliche Richtungen im Sensor-Fusion Prozess für die Maximum-Likelihood-Zuordnung zu einem Template herangezogen werden. Ebenso ist die Ergänzung absoluter Merkmale oder deren Betrachtung nur beim Vergleich mit gewählten Templates ohne weiteren Aufwand durchführbar. Falls die Unabhängigkeit der Erkennung von der Geschwindigkeit nicht erwünscht ist und gewollte Varianzen innerhalb einer Eingabe unterschieden werden sollen, kann die Methode des Resamplings gegen eine mit festem zeitlichen Abstand zwischen den Abtastpunkten ausgetauscht werden. Andernfalls werden nur die zeitlichen Relationen zwischen den Token als Diskriminierungsmerkmale beachtet.

Obwohl für die Erkennung von Single-Touch Gesten bessere Verfahren als bisher in der Literatur verfügbare vorgestellt wurden, wurde die Prokrustes-Analyse aufgegriffen und in den hierarchischen Klassifizierer integriert. Dies ist ausschließlich dem statistischen Ansatz geschuldet und der Verwendung einer der im Kapitel 4.1 als geeignet erwiesenen Methoden für die Klassifikation von Single-Touch Gesten steht nichts entgegen. Die durchschnittliche Genauigkeit des Verfahrens kann auf diese Art weiter verbessert werden. Ob auch die Erkennung von Multi-Touch Gesten oder partieller Eingaben von einer solchen Ersetzung profitiert, wurde an dieser Stelle nicht untersucht.

Als kritisch wird außerdem erachtet, dass alle Testergebnisse auf vom Autor erstellten Gesten basieren. Dies ist dem Mangel anderweitig existierender Anwendungen oder bestehender Gestensets geschuldet, welche (sequenzielle) Multi-Touch Gesten beinhalten, anhand denen ein Vergleich stattfinden hätte können. Der endgültige Nachweis der Praxistauglichkeit des vorgestellten Verfahrens kann demnach erst durch dessen Einsatz in weiteren Entwicklungen erbracht werden. Interessant wäre in diesem Zusammenhang die Beachtung des Anwendungskontextes für die Interpretation einer Eingabe. Ist eine konkrete Anwendung gegeben, können zum einen die entsprechenden a-priori-Wahrscheinlichkeiten für eine Geste gelernt und zum anderen kann ein Schwellwert für die Rückweisung als mehrdeutig angesehener Eingaben in Abhängigkeit der andernfalls gewählten Aktion spezifiziert werden. Die Ähnlichkeit zwischen Gesten sollte in einer um eigene Template-Spezifikationen erweiterbaren Anwendung auch für die Vermeidung zu ähnlicher Definitionen in unterschiedlichen Klassen oder für verschiedene Aktionen genutzt werden. Letztendlich ist ein Thema für eine eventuell aufgreifende Dissertation die Umsetzung der vorgestellten Konzepte innerhalb eines Systems der effizienten Tex-

teingabe für Blinde, welches die Prädiktion der Gesten sowohl für schneller abgeschlossene Eingaben als auch für Feedforward-Mechanismen innerhalb eines audio-haptischen Lerntools nutzt.

Literaturverzeichnis

[1] Adamek, T. und N. E. O'Connor: *A multiscale representation method for nonrigid shapes with a single closed contour*. IEEE Trans. Cir. and Sys. for Video Technol., 14(5):742–753, Mai 2004, ISSN 1051-8215. 41, 45

[2] Alvarado, C. und R. Davis: *Resolving ambiguities to create a natural computer-based sketching environment*. In: *Proceedings of IJCAI-2001*, S. 1365–1371, Aug. 2001. 34

[3] Alvarado, C. *et al.*: *A Framework for Multi-Domain Sketch Recognition*. In: *AAAI Spring Symposium on Sketch Understanding*, S. 1–8. AAAI Press, 2002. 38

[4] Amento, B., W. Hill und L. Terveen: *The sound of one hand: a wrist-mounted bio-acoustic fingertip gesture interface*. In: *CHI '02 extended abstracts on Human factors in computing systems*, CHI EA '02, S. 724–725, New York, NY, USA, 2002. ACM, ISBN 1-58113-454-1. 24

[5] Anderson, T. W.: *An Introduction to Multivariate Statistical Analysis*. Wiley, 3. Aufl., Sep. 2003, ISBN 0471360910. 36

[6] Anthony, L. und J. O. Wobbrock: *A lightweight multistroke recognizer for user interface prototypes*. In: *Proceedings of Graphics Interface 2010*, GI '10, S. 245–252, Toronto, Ont., Canada, Canada, 2010. Canadian Information Processing Society, ISBN 978-1-56881-712-5. 39

[7] Anthony, L. und J. O. Wobbrock: *$N Multistroke Recognizer Limitations*. `http://depts.washington.edu/aimgroup/proj/dollar/limits`, 2011. 39

[8] Appert, C. und O. Bau: *Scale detection for a priori gesture recognition*. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, S. 879–882, New York, NY, USA, 2010. ACM, ISBN 978-1-60558-929-9. 4, 100, 102

[9] Apte, A., V. Vo und T. D. Kimura: *Recognizing multistroke geometric shapes: an experimental evaluation*. In: *UIST '93: Proceedings of the 6th annual ACM symposium on User interface software and technology*, S. 121–128, New York, NY, USA, 1993. ACM, ISBN 0-89791-628-X. 32, 35, 117, 119, 121, 126

[10] Bae, S. H., R. Balakrishnan und K. Singh: *ILoveSketch: as-natural-as-possible sketching system for creating 3d curve models.* In: *Proceedings of the 21st annual ACM symposium on User interface software and technology*, UIST '08, S. 151–160, New York, NY, USA, 2008. ACM, ISBN 978-1-59593-975-3. 117

[11] Bakis, R.: *Continuous speech recognition via centisecond acoustic states.* The Journal of the Acoustical Society of America, 59(S1):S97–S97, 1976. 37

[12] Barrett, G. und R. Omote: *Projected-Capacitive Touch Technology.* Information Display, Society for Information Display, (26) 3:16–21, März 2010. 21

[13] Bau, O. und W. E. Mackay: *OctoPocus: a dynamic guide for learning gesture-based command sets.* In: *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, S. 37–46, New York, NY, USA, 2008. ACM, ISBN 978-1-59593-975-3. 99, 100, 102, 103

[14] Bayes, M. und M. Price: *An Essay towards Solving a Problem in the Doctrine of Chances. By the Late Rev. Mr. Bayes, F. R. S. Communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S.* Philosophical Transactions, 53:370–418, 1763. `http://rstl.royalsocietypublishing.org/content/53/370.short`, besucht: 30.06.2014. 72

[15] Bellman, R. und R. Corporation: *Dynamic programming.* Rand Corporation research study. Princeton University Press, 1957, ISBN 9780691079516. 81

[16] Bennett, M. *et al.*: *Simpleflow: enhancing gestural interaction with gesture prediction, abbreviation and autocompletion.* In: *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part I*, INTERACT'11, S. 591–608, Berlin, Heidelberg, 2011. Springer-Verlag, ISBN 978-3-642-23773-7. 100, 102, 103

[17] Bentley, J. L.: *K-d trees for semidynamic point sets.* In: *Proceedings of the sixth annual symposium on Computational geometry*, SCG '90, S. 187–197, New York, NY, USA, 1990. ACM, ISBN 0-89791-362-0. 105

[18] Bhuyan, M., D. Ghosh und P. Bora: *Feature Extraction from 2D Gesture Trajectory in Dynamic Hand Gesture Recognition.* In: *Cybernetics and Intelligent Systems, 2006 IEEE Conference on*, S. 1 –6, Juni 2006. 18, 30, 31, 33

[19] Bi, X. *et al.*: *Bimanual gesture keyboard.* In: *Proceedings of the 25th annual ACM symposium on User interface software and technology*, UIST '12, S. 137–146, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1580-7. 128

[20] Bickerstaffe, A. und V. Colquhoun: *GestureLab User and Developer Documentation.* School of Information Technology Monash University, Clayton, Victoria 3800 Australia, Sep. 2008. 38

[21] Bickerstaffe, A. *et al.*: *Developing Domain-Specific Gesture Recognizers for Smart Diagram Environments.* In: Liu, W., J. Lladós und J. M. Ogier (Hrsg.): *Graphics Recognition. Recent Advances and New Opportunities*, Kap. Sketching Interfaces and On-Line Processing, S. 145–156. Springer-Verlag, Berlin, Heidelberg, 2008, ISBN 978-3-540-88184-1. 33, 34, 38, 118, 119

[22] Bishop, C.: *Neural Networks for Pattern Recognition.* Clarendon Press, 1995, ISBN 9780198538646. 41

[23] Bresenham, J. E.: *Algorithm for computer control of a digital plotter.* IBM Syst. J., 4(1):25–30, März 1965, ISSN 0018-8670. 40

[24] Buxton, B.: *31.1: Invited Paper: A Touching Story: A Personal Perspective on the History of Touch Interfaces Past and Future.* SID Symposium Digest of Technical Papers, 41(1):444–448, 2010, ISSN 2168-0159. 19

[25] Buxton, W.: *Multitouch Systems That I Have Known And Loved.* `http://www.billbuxton.com/multitouchOverview.html`, 2010. 1

[26] Caetano, A. *et al.*: *JavaSketchIt: Issues in Sketching the Look of User Interfaces.* In: *AAAI Spring Symposium on Sketch Understanding*, S. 9–14. AAAI Press, Menlo Park, 2002. 117, 125

[27] Calhoun, C. *et al.*: *Recognizing Multi-Stroke Symbols.* In: *Proceedings of the AAAI Spring Symposium - Sketch Understanding*, S. 15–23. AAAI Press, 2002. 32, 35, 117, 118, 119, 121, 126

[28] Cao, X. und S. Zhai: *Modeling human performance of pen stroke gestures.* In: *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, S. 1495–1504, New York, NY, USA, 2007. ACM, ISBN 978-1-59593-593-9. 32, 33

[29] Caridakis, G. *et al.*: *SOMM: Self organizing Markov map for gesture recognition.* Pattern Recogn. Lett., 31:52–59, Jan. 2010, ISSN 0167-8655. 37, 40, 49

[30] Carr, R. K. und D. Shafer: *The Power of PenPoint.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st Aufl., 1991, ISBN 0201577631. 1

[31] Casacuberta, F., E. Vidal und H. Rulot: *On the metric properties of dynamic time warping.* Acoustics, Speech and Signal Processing, IEEE Transactions on, 35(11):1631–1633, 1987, ISSN 0096-3518. 47

[32] Castellucci, S. J. und I. S. MacKenzie: *Graffiti vs. unistrokes: an empirical comparison.* In: *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, S. 305–308, New York, NY, USA, 2008. ACM, ISBN 978-1-60558-011-1. 3, 18

[33] Chatty, S. und P. Lecoanet: *Pen computing for air traffic control.* In: *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*, CHI '96, S. 87–94, New York, NY, USA, 1996. ACM, ISBN 0-89791-777-4. 1

[34] Chen, C. L. P. und S. Xie: *Freehand drawing system using a fuzzy logic concept.* Computer-Aided Design, 28(2):77 – 89, 1996, ISSN 0010-4485. 32, 35, 117, 118, 119, 121, 126

[35] Chetverikov, D. und J. Verestói: *Feature point tracking for incomplete trajectories.* Computing, 62(4):321–338, 1999, ISSN 0010-485X. 25, 26

[36] Connell, S. D. und A. K. Jain: *Template-based online character recognition.* Pattern Recognition, 34(1):1 – 14, 2001, ISSN 0031-3203. 40, 45, 47, 49

[37] Cortes, C. und V. Vapnik: *Support-Vector Networks.* Machine Learning, 20:273–297, 1995, ISSN 0885-6125. 10.1023/A:1022627411411. 36, 38, 81

[38] Costagliola, G., V. Deufemia und M. Risi: *A trainable system for recognizing diagrammatic sketch languages.* In: *Visual Languages and Human-Centric Computing, 2005 IEEE Symposium on*, S. 281 – 283, Sep. 2005. 33, 34, 117, 118, 121, 126

[39] Coyette, A. *et al.*: *An Algorithm for Pen-Based Gesture Recognition Based on Levenshtein Distance.* Louvain School of Management, Universität catholique de Louvain, Working paper 07/05, 2007. 39, 40, 49, 118

[40] Coyette, A. *et al.*: *Trainable Sketch Recognizer for Graphical User Interface Design.* Human-Computer Interaction INTERACT 2007, S. 124–135, 2007. 31, 117, 118, 119, 126

[41] Crossan, A. und S. Brewster: *Multimodal Trajectory Playback for Teaching Shape Information and Trajectories to Visually Impaired Computer Users.* ACM Trans. Access. Comput., 1(2):12:1–12:34, Okt. 2008, ISSN 1936-7228. 41, 116, 135

[42] Damaraju, S. und A. Kerne: *A Gesture Learning and Recognition System for Multitouch Interaction Design.* Poster session presented at: The Future of Interactive Media Workshop on Media Arts, Science, and Technology (MAST), Jan. 2009. 37

[43] David H. Douglas und Thomas K. Peucker: *Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature.* Cartographica: The International Journal for Geographic Information and Geovisualization, 10(2):112–122, Dez. 1973. 103

[44] Delaye, A. und E. Anquetil: *HBF49 feature set: A first unified baseline for online symbol recognition.* Pattern Recognition, 46(1):117 – 130, 2013, ISSN 0031-3203. 28, 43, 61

[45] Deng, J. und H. Tsui: *An HMM-based approach for gesture segmentation and recognition.* In: *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, Bd. 3, S. 679–682 vol.3, 2000. 37, 49, 50, 102

[46] Dietz, P. und D. Leigh: *DiamondTouch: a multi-user touch technology.* In: *Proceedings of the 14th annual ACM symposium on User interface software and technology*, UIST '01, S. 219–226, New York, NY, USA, 2001. ACM, ISBN 1-58113-438-X. 20

[47] Dimond, T. L.: *Devices for reading handwritten characters.* In: *Papers and discussions presented at the December 9-13, 1957, eastern joint computer conference: Computers with deadlines to meet*, IRE-ACM-AIEE '57 (Eastern), S. 232–237, New York, NY, USA, 1958. ACM. 23

[48] Downs, R.: *Using resistive touch screens for human/machine interface.* Techn. Ber., Texas Instruments Incorporated, 2005. `http://focus.ti.com/lit/an/slyt209a/slyt209a.pdf`, besucht: 30.06.2014. 19

[49] Dryden, I. L. und K. Mardia: *Statistical shape analysis.* Wiley series in probability and statistics: Probability and statistics. J. Wiley, 1998, ISBN 0471958166. 52

[50] Duda, R. O., P. E. Hart und D. G. Stork: *Pattern Classification.* Wiley, New York, 2. Aufl., 2001. 27, 34, 41, 70, 72, 73, 81

[51] Edwards, A.: *Progress in sign language recognition.* In: Wachsmuth, I. und M. Fröhlich (Hrsg.): *Gesture and Sign Language in Human-Computer Interaction*, Bd. 1371 d. Reihe *Lecture Notes in Computer Science*, S. 13–21. Springer Berlin / Heidelberg, 1998. 10.1007/BFb0052985. 9

[52] El-ghazal, A., O. Basir und S. Belkasim: *Farthest point distance: A new shape signature for Fourier descriptors.* Signal Processing: Image Communication, 24(7):572 – 586, 2009, ISSN 0923-5965. 31

[53] Elman, J. L.: *Finding structure in time.* Cognitive Science, 14(2):179–211, 1990. 42

[54] Enis, P. und S. Geisser: *Optimal Predictive Linear Discriminants.* The Annals of Statistics, 2(2):pp. 403–410, 1974, ISSN 00905364. 82

[55] Fagin, R. und L. Stockmeyer: *Relaxing the Triangle Inequality in Pattern Matching.* International Journal of Computer Vision, 30:219–231, 1996. 47

[56] Fibich, A.: *Evaluation von Gestenalphabeten.* Diploma Thesis, Dresden University of Technology, 2012. 130, 131, 133

[57] Fisher, R. A.: *The Use of Multiple Measurements in Taxonomic Problems.* Annals of Eugenics, 7:179–188, 1936. `http://hdl.handle.net/2440/15227`, besucht: 30.06.2014. 36

[58] Fonseca, M. J., C. Pimentel und J. A. Jorge: *CALI: An Online Scribble Recognizer for Calligraphic Interfaces.* In: *Sketch Understanding, Papers from the 2002 AAAI Spring Symposium*, S. 51–58, 2002. 32, 35, 118, 119, 126

[59] Freeman, D. *et al.*: *ShadowGuides: visualizations for in-situ learning of multi-touch and whole-hand gestures.* In: *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, S. 165–172, New York, NY, USA, 2009. ACM, ISBN 978-1-60558-733-2. 2, 3, 14, 98, 103

[60] Freeman, H.: *On the encoding of arbitrary geometric configurations.* Institute of Radio Engineers, Transactions on Electronic Computers, EC-10:260–268, 1961. 31, 40

[61] Freitas, C. O. d. A. *et al.*: *Zoning and metaclasses for character recognition.* In: *Proceedings of the 2007 ACM symposium on Applied computing*, SAC '07, S. 632–636, New York, NY, USA, 2007. ACM, ISBN 1-59593-480-4. 29

[62] Friedman, J. H.: *Regularized Discriminant Analysis.* Journal of the American Statistical Association, 84(405):pp. 165–175, 1989, ISSN 01621459. 84

[63] Fukunaga, K.: *Introduction to statistical pattern recognition (2nd ed.).* Academic Press Professional, Inc., San Diego, CA, USA, 1990, ISBN 0-12-269851-7. 32, 36, 38, 71, 80, 82

[64] GestureWorks: *Open Source Gesture Library.* `http://gestureworks.com/icons-fonts`, 2011. 12

[65] Giorgino, T.: *Computing and Visualizing Dynamic Time Warping Alignments in R: The dtw Package.* Journal of Statistical Software, 31(7):1–24, Aug. 2009. 45, 47, 48, 59, 101

[66] Giorgino, T.: *Computing and Visualizing Dynamic Time Warping Alignments in R: The dtw Package.* Journal of Statistical Software, 31(7):1–24, Aug. 2009, ISSN 1548-7660. 54

[67] Goldberg, D. und C. Richardson: *Touch-typing with a stylus.* In: *Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*, CHI '93, S. 80–87, New York, NY, USA, 1993. ACM, ISBN 0-89791-575-5. 128

[68] Goodall, C.: *Procrustes Methods in the Statistical Analysis of Shape.* Journal of the Royal Statistical Society. Series B (Methodological), 53(2):pp. 285–339, 1991, ISSN 00359246. 77

[69] Görg, M. T., M. Cebulla und S. R. Garzon: *A Framework for Abstract Representation and Recognition of Gestures in Multi-touch Applications.* In: *Proceedings of the 2010 Third International Conference on Advances in Computer-Human Interactions*, ACHI '10, S. 143–147, Washington, DC, USA, 2010. IEEE Computer Society, ISBN 978-0-7695-3957-7. 34

[70] Hamada, Y., N. Shimada und Y. Shirai: *Hand shape estimation under complex backgrounds for sign language recognition.* In: *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, S. 589 – 594, Mai 2004. 25

[71] Hammond, T. und R. Davis: *Tahuti: A Geometrical Sketch Recognition System for UML Class Diagrams.* In: *Papers from the 2002 AAAI Spring Symposium on Sketch Understanding*, S. 59–68, Stanford, California, USA, 2002. AAAI Press. 35, 117

[72] Hammond, T. und R. Davis: *LADDER, a sketching language for user interface developers.* In: *ACM SIGGRAPH 2007 courses*, SIGGRAPH '07, New York, NY, USA, 2007. ACM. 33, 118

[73] Hammond, T. und B. Paulson: *Recognizing sketched multistroke primitives.* ACM Trans. Interact. Intell. Syst., 1(1):4:1–4:34, Okt. 2011, ISSN 2160-6455. 117

[74] Han, J. Y.: *Low-cost multi-touch sensing through frustrated total internal reflection.* In: *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, S. 115–118, New York, NY, USA, 2005. ACM, ISBN 1-59593-271-2. 1, 21, 22

[75] Hartson, H. und D. Hix: *Advances in Human-computer Interaction.* Nr. Bd. 3 in *Advances in Human-Computer Interaction.* Ablex, 1992, ISBN 9780893917517. 18

[76] Henniger, O. und S. Müller: *Effects of Time Normalization on the Accuracy of Dynamic Time Warping.* In: *Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007. First IEEE International Conference on*, S. 1–6, Sep. 2007. 48, 51, 60

[77] Henry, T. R., S. E. Hudson und G. L. Newell: *Integrating gesture and snapping into a user interface toolkit.* In: *Proceedings of the 3rd annual ACM SIGGRAPH symposium on User interface software and technology*, UIST '90, S. 112–122, New York, NY, USA, 1990. ACM, ISBN 0-89791-410-4. 33, 99, 118

[78] Herold, J. und T. F. Stahovich: *The 1¢ Recognizer: a fast, accurate, and easy-to-implement handwritten gesture recognition technique.* In: *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, SBIM '12, S.

39–46, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association, ISBN 978-3-905674-42-2. 39, 45, 49

[79] Hodges, S. *et al.*: *ThinSight: versatile multi-touch sensing for thin form-factor displays.* In: *Proceedings of the 20th annual ACM symposium on User interface software and technology*, UIST '07, S. 259–268, New York, NY, USA, 2007. ACM, ISBN 978-1-59593-679-0. 22

[80] Hoffbeck, J. und D. Landgrebe: *Covariance matrix estimation and classification with limited training data.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 18(7):763 –767, Juli 1996, ISSN 0162-8828. 84

[81] Hse, H., M. Shilman und A. R. Newton: *Robust sketched symbol fragmentation using templates.* In: *Proceedings of the 9th international conference on Intelligent user interfaces*, IUI '04, S. 156–160, New York, NY, USA, 2004. ACM, ISBN 1-58113-815-6. 3, 32, 97, 117, 118, 121

[82] Hussain, A. B. S., G. T. Toussaint und R. W. Donaldson: *Results Obtained Using a Simple Character Recognition Procedure on Munson's Handprinted Data.* Computers, IEEE Transactions on, C-21(2):201 –205, Feb. 1972, ISSN 0018-9340. 29, 36

[83] Igarashi, T. *et al.*: *Interactive beautification: a technique for rapid geometric design.* In: *Proceedings of the 10th annual ACM symposium on User interface software and technology*, UIST '97, S. 105–114, New York, NY, USA, 1997. ACM, ISBN 0-89791-881-9. 35

[84] Isokoski, P. und R. Raisamo: *Device independent text input: a rationale and an example.* In: *Proceedings of the working conference on Advanced visual interfaces*, AVI '00, S. 76–83, New York, NY, USA, 2000. ACM, ISBN 1-58113-252-2. 3, 18, 128

[85] Itakura, F.: *Minimum prediction residual principle applied to speech recognition.* Acoustics, Speech and Signal Processing, IEEE Transactions on, 23(1):67–72, 1975, ISSN 0096-3518. 59, 60

[86] Janke, G.: *Entwicklung eines Texteingabesystems für die kurzschriftliche Eingabe auf mobilen Geräten.* Diploma Thesis, Dresden University of Technology, 2013. 100, 101

[87] Jansen, A. R., K. Marriott und B. Meyer: *Cider: A Component-Based Toolkit for Creating Smart Diagram Environments.* In: Blackwell, A. F., K. Marriott und A. Shimojima (Hrsg.): *Diagrammatic Representation and Inference*, Bd. 2980 d. Reihe *Lecture Notes in Computer Science*, S. 415–419. Springer Berlin Heidelberg, 2004, ISBN 978-3-540-21268-3. 33, 38, 117, 126

[88] Jolliffe, I.: *Principal Component Analysis.* Springer Series in Statistics. Springer, 2002, ISBN 9780387954424. 94, 96

[89] Kammer, D. *et al.*: *Towards a formalization of multi-touch gestures.* In: *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, S. 49–58, New York, NY, USA, 2010. ACM, ISBN 978-1-4503-0399-6. 34

[90] Kane, S. K., J. P. Bigham und J. O. Wobbrock: *Slide rule: making mobile touch screens accessible to blind people using multi-touch interaction techniques.* In: *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility*, Assets '08, S. 73–80, New York, NY, USA, 2008. ACM, ISBN 978-1-59593-976-0. 2

[91] Kara, L. B. und T. F. Stahovich: *An image-based, trainable symbol recognizer for hand-drawn sketches.* Comput. Graph., 29(4):501–517, Aug. 2005, ISSN 0097-8493. 33

[92] Karam, M. und M. C. Schraefel: *A taxonomy of gestures in human computer interactions.* Transactions on Computer-Human Interactions, 2005. `http://eprints.ecs.soton.ac.uk/11149/1/GestureTaxonomyJuly21.pdf`, besucht: 30.06.2014. 9, 10, 11, 12

[93] Kawashima, M. *et al.*: *Adaptive template method for early recognition of gestures.* In: *Frontiers of Computer Vision (FCV), 2011 17th Korea-Japan Joint Workshop on*, S. 1–6, 2011. 101, 102

[94] Keogh, E.: *Exact indexing of dynamic time warping.* In: *Proceedings of the 28th international conference on Very Large Data Bases*, VLDB '02, S. 406–417. VLDB Endowment, 2002. 47, 59

[95] Keogh, E. und C. A. Ratanamahatana: *Exact Indexing of Dynamic Time Warping.* Knowl. Inf. Syst., 7(3):358–386, März 2005, ISSN 0219-1377. 48, 60

[96] Keshari, B. und S. M. Watt: *Online Mathematical Symbol Recognition using SVMs with Features from Functional Approximation.* In: *Electronic Proc. Mathematical User-Interfaces Workshop (MathUI 08)*, Birmingham, UK„ Juli 2008. 31

[97] Khandkar, S. H. und F. Maurer: *A domain specific language to define gestures for multi-touch applications.* In: *Proceedings of the 10th Workshop on Domain-Specific Modeling*, DSM '10, S. 2:1–2:6, New York, NY, USA, 2010. ACM, ISBN 978-1-4503-0549-5. 34

[98] Kim, D. und S. K. Kim: *Comparing patterns of component loadings: Principal Component Analysis (PCA) versus Independent Component Analysis (ICA) in analyzing multivariate non-normal data.* Behavior Research Methods, 44(4):1239–1243, 2012. 95

[99] Kim, J.: *On-Line Gesture Recognition by Feature Analysis.* Research Report. IBM Thomas J. Watson Research Center, 1987. 4, 34, 49

[100] Kim, J., J. G. Ahn und H. Ko: *Orientation Responsive Touch Interaction.* In: *Proceedings of the 13th International Conference on Human-Computer Interaction. Part II: Novel Interaction Methods and Techniques*, S. 461–469, Berlin, Heidelberg, 2009. Springer-Verlag, ISBN 978-3-642-02576-1. 23

[101] Kölsch, M. und M. Turk: *Robust hand detection.* In: *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, S. 614–619, Mai 2004. 25

[102] Költringer, T. und T. Grechenig: *Comparing the immediate usability of graffiti 2 and virtual keyboard.* In: *CHI '04 extended abstracts on Human factors in computing systems*, CHI EA '04, S. 1175–1178, New York, NY, USA, 2004. ACM, ISBN 1-58113-703-6. 18, 129

[103] Kostakos, V. und E. O'Neill: *A directional stroke recognition technique for mobile interaction in a pervasive computing world.* In: *People and Computers XVII, Proceedings of HCI 2003*, S. 197—206, 2003. `http://opus.bath.ac.uk/5531/`, besucht: 30.06.2014. 39, 49, 50

[104] Kozlay, D.: *Feature Extraction in an Optical Character Recognition Machine.* IEEE Trans. Comput., 20(9):1063–1067, 1971, ISSN 0018-9340. 4, 35

[105] Kraner, R. H.: *Tracking Technologies for Interactive Tabletop Surfaces.* Dissertation, ETH ZURICH, 2011. 22, 23

[106] Kratz, S. und M. Rohs: *A $3 gesture recognizer: simple gesture recognition for devices equipped with 3D acceleration sensors.* In: *Proceedings of the 15th international conference on Intelligent user interfaces*, IUI '10, S. 341–344, New York, NY, USA, 2010. ACM, ISBN 978-1-60558-515-4. 12

[107] Kristensson, P. O. und S. Zhai: *SHARK2: a large vocabulary shorthand writing system for pen-based computers.* In: *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, S. 43–52, New York, NY, USA, 2004. ACM, ISBN 1-58113-957-8. 128

[108] Kuhn, H. W.: *The Hungarian method for the assignment problem.* Naval Research Logistics Quarterly, 2:83–97, 1955. 26, 75

[109] Kurtoglu, T. und T. F. Stahovich: *Interpreting schematic sketches using physical reasoning.* In: *Proceedings of the AAAI Spring Symposium*, S. 78–85, Stanford, CA, 2002. AAAI Press. 117, 118, 126

[110] Lü, H. und Y. Li: *Gesture coder: a tool for programming multi-touch gestures by demonstration.* In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, S. 2875–2884, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1015-4. 12, 35

[111] Landay, J. A. und B. A. Myers: *Interactive sketching for the early stages of user interface design.* In: *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95, S. 43–50, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co., ISBN 0-201-84705-1. 3, 117, 118, 125

[112] Landay, J. A. und B. A. Myers: *Sketching Interfaces: Toward More Human Interface Design.* Computer, 34(3):56–64, März 2001, ISSN 0018-9162. 126

[113] Ledoit, O. und M. Wolf: *Improved estimation of the covariance matrix of stock returns with an application to portfolio selection.* Journal of Empirical Finance, 10(5):603 – 621, 2003, ISSN 0927-5398. 83

[114] Lee, J. J., J. Kim und J. H. Kim: *Data-driven design of HMM topology for online handwriting recognition.* International Journal of Pattern Recognition and Artificial Intelligence, 15(01):107–121, 2001. 37

[115] Levenshtein, V. I.: *Binary codes capable of correcting deletions, insertions, and reversals.* Soviet Physics Doklady, 10(8):707–710, 1966. 39, 47

[116] Levine, M.: *Feature extraction: A survey.* Proceedings of the IEEE, 57(8):1391 – 1407, Aug. 1969, ISSN 0018-9219. 27, 28, 30

[117] Li, X. *et al.*: *Segmentation and Reconstruction of On-line Handwritten Scripts*, 1997. 32

[118] Li, X., R. Plamondon und M. Parizeau: *Model-Based On-Line Handwritten Digit Recognition.* Pattern Recognition, International Conference on, 2:1134, 1998, ISSN 1051-4651. 29, 37

[119] Li, Y.: *Protractor: a fast and accurate gesture recognizer.* In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, S. 2169–2172, New York, NY, USA, 2010. ACM, ISBN 978-1-60558-929-9. 31, 39, 45, 49, 50, 53, 54

[120] Lin, C. C. und R. Chellappa: *Classification of partial 2-D shapes using Fourier descriptors.* IEEE Trans. Pattern Anal. Mach. Intell., 9:686–690, Sep. 1987, ISSN 0162-8828. 31

[121] Lindeberg, T.: *Feature Detection with Automatic Scale Selection.* Int. J. Comput. Vision, 30:79–116, Nov. 1998, ISSN 0920-5691. 25

[122] Long, Jr., A. C. *et al.*: *Visual similarity of pen gestures.* In: *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '00, S. 360–367, New York, NY, USA, 2000. ACM, ISBN 1-58113-216-6. 4

[123] Long Jr., A. C.: *Quill: A Gesture Design Tool for Pen-based User Interfaces.* Dissertation, University of California, Berkeley, 2001, ISBN 0-493-58448-X. 2

[124] Ma, C. und G. Dai: *Gesture language use in natural UI: pen-based sketching in conceptual design.* In: Z. Pan & J. Shi (Hrsg.): *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, Bd. 4756 d. Reihe *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, S. 122–128, Apr. 2003. 34

[125] MacKenzie, I. S., J. Chen und A. Oniszczak: *Unipad: single stroke text entry with language-based acceleration.* In: *Proceedings of the 4th Nordic conference on Human-computer interaction: changing roles*, NordiCHI '06, S. 78–85, New York, NY, USA, 2006. ACM, ISBN 1-59593-325-5. 128

[126] MacKenzie, I. S. und R. W. Soukoreff: *Phrase Sets for Evaluating Text Entry Techniques.* In: *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, S. 754–755, New York, NY, USA, 2003. ACM, ISBN 1-58113-637-4. 131

[127] MacKenzie, I. S. und K. Tanaka-Ishii: *Text Entry Systems: Mobility, Accessibility, Universality.* The Morgan Kaufmann Series in Interactive Technologies. Boston, 2007, ISBN 9780123735911. 127, 128, 130, 133

[128] MacKenzie, I. S. und S. X. Zhang: *The immediate usability of graffiti.* In: *Proceedings of the conference on Graphics interface '97*, S. 129–137, Toronto, Ont., Canada, Canada, 1997. Canadian Information Processing Society, ISBN 0-9695338-6-1. 10

[129] Mahalanobis, P. C.: *On the generalised distance in statistics.* In: *Proceedings National Institute of Science, India*, Bd. 2, S. 49–55, Apr. 1936. 73

[130] Mankoff, J. und G. D. Abowd: *Cirrin: a word-level unistroke keyboard for pen input.* In: *Proceedings of the 11th annual ACM symposium on User interface software and technology*, UIST '98, S. 213–214, New York, NY, USA, 1998. ACM, ISBN 1-58113-034-1. 128

[131] Martin, B.: *VirHKey: a VIRtual Hyperbolic KEYboard with gesture interaction and visual feedback for mobile devices.* In: *Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, MobileHCI '05, S. 99–106, New York, NY, USA, 2005. ACM, ISBN 1-59593-089-2. 128

[132] Mcavinney, P.: *The sensor frame - a gesture-based device for the manipulation of graphic objects.* 1986. 22

[133] McClelland, D.: *Developments in touchscreen technology.* Displays, 11(2):93 – 95, 1990, ISSN 0141-9382. 24

[134] McGookin, D., S. Brewster und W. Jiang: *Investigating touchscreen accessibility for people with visual impairments.* In: *Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges*, NordiCHI '08, S. 298–307, New York, NY, USA, 2008. ACM, ISBN 978-1-59593-704-9. 2

[135] Mcneill, D.: *Hand and Mind: What Gestures Reveal about Thought.* University Of Chicago Press, Aug. 1992, ISBN 0226561321. 8, 10

[136] Miller, R. B.: *Response time in man-computer conversational transactions.* In: *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, AFIPS '68 (Fall, part I), S. 267–277, New York, NY, USA, 1968. ACM. 4

[137] Mori, A. *et al.*: *Early Recognition and Prediction of Gestures.* In: *Proceedings of the 18th International Conference on Pattern Recognition - Volume 03*, ICPR '06, S. 560–563, Washington, DC, USA, 2006. IEEE Computer Society, ISBN 0-7695-2521-0. 54, 101, 102

[138] Myers, B. A.: *A brief history of human-computer interaction technology.* interactions, 5(2):44–54, März 1998, ISSN 1072-5520. 1

[139] Myers, C., L. Rabiner und A. Rosenberg: *Performance tradeoffs in dynamic time warping algorithms for isolated word recognition.* Acoustics, Speech and Signal Processing, IEEE Transactions on, 28(6):623 – 635, Dez. 1980, ISSN 0096-3518. 47

[140] Myers, C. S.: *A Comparative Study Of Several Dynamic Time Warping Algorithms For Speech Recognition.* Diplomarbeit, Massachusetts Institute of Technology, Cambridge, 1980. `http://dspace.mit.edu/bitstream/1721.1/27909/1/07888629.pdf`, besucht: 30.06.2014. 54

[141] Nagy, G.: *State of the Art in Pattern Recognition.* In: *Proceedings of IEEE*, Bd. 56, S. 836–857, 1968, ISBN 978-1-59593-975-3. 27

[142] Niblack, W. und J. Yin: *A pseudo-distance measure for 2D shapes based on turning angle.* In: *Proceedings of the 1995 International Conference on Image Processing (Vol. 3)-Volume 3 - Volume 3*, ICIP '95, S. 3352–, Washington, DC, USA, 1995. IEEE Computer Society, ISBN 0-8186-7310-9. 41, 45, 49

[143] Nielsen, M. *et al.*: *A Procedure for Developing Intuitive and Ergonomic Gesture Interfaces for HCI.* In: Camurri, A. und G. Volpe (Hrsg.): *Gesture-Based Communication in Human-Computer Interaction*, Bd. 2915 d. Reihe *Lecture Notes in Computer Science*, S. 409–420. Springer Berlin Heidelberg, 2004, ISBN 978-3-540-21072-6. 2

[144] Novakovic J., R. S.: *Classification Performance Using Principal Component Analysis and Different Value of the Ratio R.* International Journal of Computers Communications & Control, 6(2):317–327, 2011. 95, 96

[145] Opgen-Rhein, R. und K. Strimmer: *Accurate Ranking of Differentially Expressed Genes by a Distribution-Free Shrinkage Approach.* Statistical Applications in Genetics and Molecular Biology, 6(1), 2007. 84

[146] Ou, J. *et al.*: *Gestural communication over video stream: supporting multimodal interaction for remote collaborative physical tasks.* In: *Proceedings of the 5th international conference on Multimodal interfaces*, ICMI '03, S. 242–249, New York, NY, USA, 2003. ACM, ISBN 1-58113-621-8. 34, 35

[147] Ouyang, T. Y. und R. Davis: *A visual approach to sketched symbol recognition.* In: *Proceedings of the 21st international jont conference on Artifical intelligence*, IJCAI'09, S. 1463–1468, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc. 33, 99, 102

[148] Paradiso, J. A. *et al.*: *Passive acoustic knock tracking for interactive windows.* In: *CHI '02 extended abstracts on Human factors in computing systems*, CHI EA '02, S. 732–733, New York, NY, USA, 2002. ACM, ISBN 1-58113-454-1. 24

[149] Paulson, B. und T. Hammond: *PaleoSketch: accurate primitive sketch recognition and beautification.* In: *Proceedings of the 13th international conference on Intelligent user interfaces*, IUI '08, S. 1–10, New York, NY, USA, 2008. ACM, ISBN 978-1-59593-987-6. 35, 117, 118, 126

[150] Pavlovic, V. I., R. Sharma und T. S. Huang: *Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review.* IEEE Trans. Pattern Anal. Mach. Intell., 19(7):677–695, Juli 1997, ISSN 0162-8828. 11

[151] Pechenizkiy, M., A. Tsymbal und S. Puuronen: *PCA-based Feature Transformation for Classification: Issues in Medical Diagnostics.* In: *Proceedings of the 17th IEEE Symposium on Computer-Based Medical Systems*, CBMS '04, S. 535–, Washington, DC, USA, 2004. IEEE Computer Society, ISBN 0-7695-2104-5. 96

[152] Perlin, K.: *Quikwriting: continuous stylus-based text entry.* In: *Proceedings of the 11th annual ACM symposium on User interface software and technology*, UIST '98, S. 215–216, New York, NY, USA, 1998. ACM, ISBN 1-58113-034-1. 128

[153] Pickering, J.: *Touch-sensitive screens: the technologies and their application.* International Journal of Man-Machine Studies, 25(3):249 – 269, 1986, ISSN 0020-7373. 19, 22

[154] Plamondon, R.: *A model-based segmentation framework for computer processing of handwriting.* In: *Pattern Recognition, 1992. Vol.II. Conference B: Pattern Recognition Methodology and Systems, Proceedings., 11th IAPR International Conference on*, S. 303 –307, Aug. 1992. 32, 33

[155] Poirier, F. und M. Belatar: *Glyph 2: une saisie de texte avec deux appuis de touche par caractère - principes et comparaisons.* In: *Proceedings of the 18th International Conferenceof the Association Francophone d'Interaction Homme-Machine*, IHM '06, S. 159–162, New York, NY, USA, 2006. ACM, ISBN 1-59593-350-6. 128

[156] R Development Core Team: *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2011. `http://www.R-project.org`, besucht: 30.06.2014, ISBN 3-900051-07-0. 54, 84

[157] Rabiner, L. und B. H. Juang: *Fundamentals of Speech Recognition.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993, ISBN 0-13-015157-2. 54

[158] Rabiner, L., A. Rosenberg und S. Levinson: *Considerations in dynamic time warping algorithms for discrete word recognition.* Acoustics, Speech and Signal Processing, IEEE Transactions on, 26(6):575 – 582, Dez. 1978, ISSN 0096-3518. 47

[159] Rabiner, L. R.: *A tutorial on hidden markov models and selected applications in speech recognition.* In: *Proceedings of the IEEE*, S. 257–286, 1989. 37

[160] Ramer, U.: *An iterative procedure for the polygonal approximation of plane curves.* Computer Graphics and Image Processing, 1(3):244 – 256, 1972, ISSN 0146-664X. 31, 103

[161] Rangarajan, A., H. Chui und F. L. Bookstein: *The Softassign Procrustes Matching Algorithm.* In: *Proceedings of the 15th International Conference on Information Processing in Medical Imaging*, IPMI '97, S. 29–42, London, UK, UK, 1997. Springer-Verlag, ISBN 3-540-63046-5. 39, 52, 78

[162] Ratanamahatana, A. und E. Keogh: *Everything you know about dynamic time warping is wrong.* 3rd Workshop on Mining Temporal and Sequential Data, in conjunction with 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD-2004), Seattle, WA, 2004. 51

[163] Ratanamahatana, C. A. und E. J. Keogh: *Three Myths about Dynamic Time Warping Data Mining.* In: *SDM'05*, S. –1–1, 2005. 47, 70

[164] Renau-Ferrer, N. *et al.*: *The ILGDB database of realistic pen-based gestural commands.* In: *ICPR*, S. 3741–3744. IEEE, 2012, ISBN 978-1-4673-2216-4. 53, 60, 61, 101

[165] Rick, J.: *Performance optimizations of virtual keyboards for stroke-based text entry on a touch-based tabletop.* In: *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, S. 77–86, New York, NY, USA, 2010. ACM, ISBN 978-1-4503-0271-5. 128

[166] Rish, I.: *An empirical study of the naive Bayes classifier.* In: *IJCAI-01 workshop on "Empirical Methods in AI"*, 2005. `http://www.research.ibm.com/people/r/rish/papers/RC22230.pdf`, besucht: 30.06.2014. 82

[167] Rohr, K.: *Localization properties of direct corner detectors.* Journal of Mathematical Imaging and Vision, 4:139–150, 1994, ISSN 0924-9907. 25

[168] Roth, P. M., M. Donoser und H. Bischof: *On-line learning of unknown hand held objects via tracking.* In: *In Int. Conf. on Computer Vision Systems*, 2006. 26

[169] Rubine, D.: *The Automatic Recognition of Gestures.* Dissertation, Carnegie Mellon University, 1991. 17, 28, 29, 30, 31, 36, 42, 43, 44, 76, 99, 102, 118, 126

[170] Rubine, D.: *Specifying gestures by example.* SIGGRAPH Comput. Graph., 25(4):329–337, 1991, ISSN 0097-8930. 36, 38, 81, 99, 118, 119

[171] Rubine, D.: *Combining gestures and direct manipulation.* In: *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '92, S. 659–660, New York, NY, USA, 1992. ACM, ISBN 0-89791-513-5. 10

[172] Russel, S. und P. Norvig: *Künstliche Intelligenz. Ein moderner Ansatz.* Pearson Education Deutschland, München, 2. Aufl., 2004. 32, 82

[173] Sakoe, H. und S. Chiba: *Dynamic programming algorithm optimization for spoken word recognition.* Acoustics, Speech and Signal Processing, IEEE Transactions on, 26(1):43 – 49, Feb. 1978, ISSN 0096-3518. 45, 46, 47, 48, 54, 59, 60

[174] Sato, M., I. Poupyrev und C. Harrison: *Touché: enhancing touch interaction on humans, screens, liquids, and everyday objects.* In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, S. 483–492, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1015-4. 21

[175] Schäfer, J., R. Opgen-Rhein und K. Strimmer: *Efficient Estimation of Covariance and (Partial) Correlation.* `http://strimmerlab.org/software/corpcor/`, 2010. 84

[176] Schäfer, J. und K. Strimmer: *A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics.* ical Applications in Genetics and Molecular Biology, 4(1):1175–1189, Nov. 2005. 83, 84

[177] Schlesinger, M. I. und V. Hlavác: *Ten Lectures on Statistical and Structural Pattern Recognition.* Kluwer Academic Publishers, 2002. 71, 97

[178] Schmidt, G. S. und D. H. House: *Towards Model-Based Gesture Recognition.* In: *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000*, FG '00, S. 416–, Washington, DC, USA, 2000. IEEE Computer Society, ISBN 0-7695-0580-5. 33, 44

[179] Schmidt, M., A. Fibich und G. Weber: *MTIS: A Multi-Touch Text Input System.* In: Streitz, N. und C. Stephanidis (Hrsg.): *Distributed, Ambient, and Pervasive Interactions*, Bd. 8028 d. Reihe *Lecture Notes in Computer Science*, S. 62–71. Springer Berlin Heidelberg, 2013, ISBN 978-3-642-39350-1. 53, 61, 106, 127

[180] Schmidt, M. und G. Weber: *Multitouch Haptic Interaction.* In: *UAHCI '09: Proceedings of the 5th International on ConferenceUniversal Access in Human-Computer Interaction. Part II*, S. 574–582, Berlin, Heidelberg, 2009. Springer-Verlag, ISBN 978-3-642-02709-3. 20, 103, 115, 135

[181] Schmidt, M. und G. Weber: *Enhancing Single Touch Gesture Classifiers to Multitouch Support.* In: *ICCHP (2)*, S. 490–497, 2010. 75, 89

[182] Schmidt, M. und G. Weber: *Recognition of Multi-touch Drawn Sketches.* In: Kurosu, M. (Hrsg.): *Human-Computer Interaction. Interaction Modalities and Techniques*, Bd. 8007 d. Reihe *Lecture Notes in Computer Science*, S. 479–490. Springer Berlin Heidelberg, 2013, ISBN 978-3-642-39329-7. 33, 117

[183] Schmidt, M. und G. Weber: *Template based classification of multi-touch gestures.* Pattern Recognition, 46(9):2487 – 2496, 2013, ISSN 0031-3203. 12, 14, 17, 70

[184] Schmidt, M. und G. Weber: *Shapes, Symbols, Sketches, Characters and Gestures - Trajectory-Based Classification of Single-Stroke Input.* Transactions on Pattern Analysis and Machine Intelligence, 2014. eingereicht am 7. März 2014. 33, 44

[185] Schöning, J. *et al.*: *Multi-Touch Surfaces: A Technical Guide.* Techn. Ber., Technical University of Munich, Okt. 2008. 18, 19, 20, 24

[186] Seier, E.: *Comparison of Tests for Univariate Normality.* Techn. Ber., East Tennessee State University, USA, 2002. 92

[187] Selfridge, O. G.: *Pattern recognition and modern computers.* In: *Proceedings of the March 1-3, 1955, western joint computer conference*, AFIPS '55 (Western), S. 91–93, New York, NY, USA, 1955. ACM. 27

[188] Sethi, I. K. und R. Jain: *Finding trajectories of feature points in a monocular image sequence.* IEEE Trans. Pattern Anal. Mach. Intell., 9(1):56–73, 1987, ISSN 0162-8828. 26

[189] Sezgin, T. M. und R. Davis: *HMM-based efficient sketch recognition.* In: *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*, S. 281–283, New York, NY, USA, 2005. ACM, ISBN 1-58113-894-6. 35, 37, 117, 118, 125, 126

[190] Sezgin, T. M., T. Stahovich und R. Davis: *Sketch Based Interfaces: Early Processing for Sketch Understanding.* Workshop on Perceptive User Interfaces, Orlando FL, 2001. 32, 34, 35, 37, 117, 118, 119, 121, 126

[191] Shilman, M., H. Pasula und S. R. R. Newton: *Statistical visual language models for ink parsing.* In: *Proceedings of the AAAI Spring Symposium on Sketch Understanding*, S. 126–32, 2002. 126

[192] Shneiderman, B.: *Direct Manipulation: A Step Beyond Programming Languages.* Computer, 16(8):57 –69, Aug. 1983, ISSN 0018-9162. 11, 98

[193] Simistira, F., V. Katsouros und G. Carayannis: *A Template Matching Distance for Recognition of On-Line Mathematical Symbols.* In: Simistira, F., V. Katsouros und G. Carayannis (Hrsg.): *11th International Conference on Frontiers in Handwriting Recognition (ICFHR 2008)*, Quebec, Canada, 2008. 31, 33, 40, 49, 100

[194] Smithies, S., K. Novins und J. Arvo: *A handwriting-based equation editor.* In: *Proceedings of the 1999 conference on Graphics interface '99*, S. 84–91, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc., ISBN 1-55860-632-7. 116

[195] Souza, C. R.: *A Tutorial on Principal Component Analysis with the Accord.NET Framework.* Techn. Ber., Department of Computing, Federal University of Sao Carlos, 2012. 96

[196] Spath, D. *et al.*: *Multi-Touch Technologie, Hard-/Software und deren Anwendungsszenarien.* `http://wiki.iao.fraunhofer.de/images/studien/studie-multi-touch-fraunhofer-iao.pdf`, 2010. 18, 20, 21, 22, 23

[197] Strimmer, K.: *Comments on: Augmenting the bootstrap to analyze high dimensional genomic data.* TEST: An Official Journal of the Spanish Society of Statistics and Operations Research, 17(1):25–27, Mai 2008. 84

[198] Sun, Z., W. Jiang und J. Sun: *Adaptive online multi-stroke sketch recognition based on hidden markov model.* In: *Proceedings of the 4th international conference on Advances in Machine Learning and Cybernetics*, ICMLC'05, S. 948–957, Berlin, Heidelberg, 2006. Springer-Verlag, ISBN 3-540-33584-6, 978-3-540-33584-9. 37

[199] Sutherland, I. E.: *Sketch pad a man-machine graphical communication system.* In: *Proceedings of the SHARE design automation workshop*, DAC '64, S. 6.329–6.346, New York, NY, USA, 1964. ACM. 117

[200] Thomaz, C., R. Feitosa und D. Gillies: *Using Mixture Covariance Matrices to Improve Face and Facial Expression Recognitions.* Pattern Recognition Letters, 24:2159–2165, 2003. 83, 84

[201] Tian, F. *et al.*: *Research on User-Centered Design and Recognition Pen Gestures.* In: Nishita, T., Q. Peng und H. P. Seidel (Hrsg.): *Advances in Computer Graphics*, Bd. 4035 d. Reihe *Lecture Notes in Computer Science*, S. 312–323. Springer Berlin Heidelberg, 2006, ISBN 978-3-540-35638-7. 9

[202] Tirkaz, C., B. Yanikoglu und T. M. Sezgin: *Sketched symbol recognition with auto-completion.* Pattern Recognition, 45(11):3926 – 3937, 2012, ISSN 0031-3203. 99, 102

[203] Vukadinovic, D. und M. Pantic: *Fully Automatic Facial Feature Point Detection Using Gabor Feature Based Boosted Classifiers.* In: *In SMC'05*, S. 1692–1698, 2005. 25

[204] Wahl, P. W. und R. A. Kronmal: *Discriminant Functions when Covariances are Unequal and Sample Sizes are Moderate.* Biometrics, 33(3):pp. 479–484, 1977, ISSN 0006341X. 36, 81

[205] Wang, X. *et al.*: *Experimental comparison of representation methods and distance measures for time series data.* Data Mining and Knowledge Discovery, 26(2):275–309, 2013, ISSN 1384-5810. 48, 60

[206] Watt, S. M. und X. Xie: *Prototype Pruning by Feature Extraction in Handwritten Mathematical Symbol Recognition.* In: *Maple Conference 2005*, S. 423–437, Waterloo Canada, 2005. Maplesoft. 3, 28, 29, 30, 33, 41

[207] Webel, S., J. Keil und M. Zoellner: *Multi-touch gestural interaction in X3D using hidden Markov models.* In: *Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, VRST '08, S. 263–264, New York, NY, USA, 2008. ACM, ISBN 978-1-59593-951-7. 2, 31, 37, 49

[208] Wenyin, L.: *On-line graphics recognition: state-of-the-art.* In: *In 5th Int. Workshop on Graphics Recognition*, S. 291–304, 2003. 44

[209] Wenyin, L. *et al.*: *Smart Sketchpad - An On-line Graphics Recognition System.* In: *Proceedings of the Sixth International Conference on Document Analysis and Recognition*, ICDAR '01, S. 1050–, Washington, DC, USA, 2001. IEEE Computer Society, ISBN 0-7695-1263-1. 117, 119, 121, 126

[210] Wexelblat, A.: *Research Challenges in Gesture: Open Issues and Unsolved Problems.* In: *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, S. 1–11, London, UK, 1998. Springer-Verlag, ISBN 3-540-64424-5. 8, 11

[211] Wikimedia Commons: *Palm Pilot 5000.* `http://commons.wikimedia.org/wiki/File:PalmPilot5000.jpg`, Juli 2004. 19

[212] Wikimedia Commons: *IBM Simon Personal Communicator.* `http://commons.wikimedia.org/wiki/File:IBM_Simon_Personal_Communicator.png`, Juni 2012. 19

[213] Willems, D. *et al.*: *Iconic and multi-stroke gesture recognition.* Pattern Recognition, 42(12):3303 – 3312, 2009, ISSN 0031-3203. New Frontiers in Handwriting Recognition. 28, 30, 43, 45

[214] Wilson, A. *et al.*: *Using Configuration States for the Representation and Recognition of Gesture.* M.I.T. Media Lab Vision and Modeling Group technical report. Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, 1995. 5, 9, 41, 101

[215] Wobbrock, J. O.: *Edgewrite: A Versatile Design for Text Entry and Control.* Dissertation, Carnegie Mellon University, Pittsburgh, PA, USA, 2006, ISBN 978-0-542-97631-5. 3, 18, 128

[216] Wobbrock, J. O., M. R. Morris und A. D. Wilson: *User-defined gestures for surface computing.* In: *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, S. 1083–1092, New York, NY, USA, 2009. ACM, ISBN 978-1-60558-246-7. 2

[217] Wobbrock, J. O., A. D. Wilson und Y. Li: *Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes.* In: *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, S. 159–168, New York, NY, USA, 2007. ACM, ISBN 978-1-59593-679-2. 31, 38, 39, 45, 49, 50, 53, 54, 77, 106

[218] Wolf, C. G.: *CAN PEOPLE USE GESTURE COMMANDS?* SIGCHI Bull., 18(2):73–74, Okt. 1986, ISSN 0736-6906. 2, 3

[219] Wolf, C. G. und P. Morrel-Samuels: *The use of hand-drawn gestures for text editing.* Int. J. Man-Mach. Stud., 27(1):91–102, Juli 1987, ISSN 0020-7373. 1, 3

[220] Worth, C. D.: *xstroke: Full-screen Gesture Recognition for X.* In: *USENIX Annual Technical Conference, FREENIX Track*, S. 187–196. USENIX, 2003, ISBN 1-931971-11-0. 1, 34, 49

[221] Xi, X. *et al.*: *Fast time series classification using numerosity reduction.* In: *Proceedings of the 23rd international conference on Machine learning*, ICML '06, S. 1033–1040, New York, NY, USA, 2006. ACM, ISBN 1-59593-383-2. 45, 48, 56, 60

[222] Yang, J. und Y. Xu: *Hidden Markov Model for Gesture Recognition.* Techn. Ber. CMU-RI-TR-94-10, Robotics Institute, Pittsburgh, PA, Mai 1994. 30, 33, 37, 102

[223] Yu, B. und S. Cai: *A domain-independent system for sketch recognition.* In: *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, GRAPHITE '03, S. 141–146, New York, NY, USA, 2003. ACM, ISBN 1-58113-578-5. 32, 35, 117, 118, 119, 126

[224] Yuan, Y., Y. Liu und K. Barner: *Tactile gesture recognition for people with disabilities.* In: *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, Bd. 5, S. 461– 464, März 2005. 25, 42, 76

[225] Zeleznik, R. *et al.*: *Hands-on math: a page-based multi-touch and pen desktop for technical work and problem solving.* In: *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, S. 17–26, New York, NY, USA, 2010. ACM, ISBN 978-1-4503-0271-5. 116

[226] Zeleznik, R. C. *et al.*: *Lineogrammer: creating diagrams by drawing.* In: *Proceedings of the 21st annual ACM symposium on User interface software and technology*, UIST '08, S. 161–170, New York, NY, USA, 2008. ACM, ISBN 978-1-59593-975-3. 34, 35, 117, 118, 119

[227] Zhai, S., M. Hunter und B. A. Smith: *Performance optimization of virtual keyboards.* Human-Computer Interaction, S. 89–129, 2002. 128

[228] Zhang, D. und G. Lu: *A Comparative Study on Shape Retrieval Using Fourier Descriptors with Different Shape Signatures.* Journal of Visual Communication and Image Representation, 14(1):41–60, 2003. 31, 36, 44

[229] Zhu, X., J. Yang und A. Waibel: *Segmenting hands of arbitrary color.* In: *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, S. 446 –453, 2000. 25

[230] Özün, O. *et al.*: *Vision Based Single Stroke Character Recognition For Wearable Computing.* IEEE Intelligent Systems and Applications, 16:33–37, 2001. 49