Lehrstuhl für Mensch-Maschine-Kommunikation
Technische Universität München

Multimodale Erfassung mathematischer Formeln durch einstufig-probabilistische semantische Decodierung

Jörg Hunsinger

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. rer. nat. G. Wachutka
Prüfer der Dissertation:	1. Univ.-Prof. Dr. rer. nat. M. Lang, i. R.
	2. Univ.-Prof. Dr. rer. nat. J. Schlichter
	3. Univ.-Prof. Dr.-Ing. E. Steinbach

Die Dissertation wurde am 30.09.2002 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 26.02.2003 angenommen.

Vorwort

Die hier vorgestellte Arbeit ist das Ergebnis meiner Forschungstätigkeit als wissenschaftlicher Assistent am Lehrstuhl für Mensch-Maschine-Kommunikation der Technischen Universität München.

Mein ganz besonderer Dank gilt meinem Doktorvater PROF. MANFRED LANG, der stets mit sicherer Hand dafür sorgte, dass dieses Projekt mit angemessenem methodischen Spielraum innerhalb einer klaren wissenschaftlichen Zielsetzung gelingen konnte. Das vorbildliche Arbeitsumfeld an seinem Lehrstuhl wirkte ebenso motivierend wie seine aktive Unterstützung bei meiner Teilnahme an wissenschaftlichen Fachtagungen im In- und Ausland.

Ein herzliches Dankeschön geht an meine lieben Kollegen, insbesondere an MICHAEL GEIGER, MARC HOFMANN, RALF NIESCHULZ und MARTIN ZOBL für viele anregende fachübergreifende Gespräche. Für die stete Unterstützung in allen technischen Belangen danke ich HEINER HUNDHAMMER, PETER BRAND und DR. CLAUS VON RÜCKER.

Herrn PROF. GÜNTHER RUSKE und seinen Mitarbeitern THILO PFAU und ROBERT FALTLHAUSER sei Dank für Rat und Tat in allen Fragen der Sprachverarbeitung.

Allen beteiligten Diplomanden, Studienarbeitern, Praktikanten und Werkstudenten danke ich für ihr Engagement, an erster Stelle ROBERT LIEB für unentbehrliche Beiträge in mehreren Projektphasen. Danke auch an BJÖRN SCHULLER für seine Mithilfe zu Beginn des Vorhabens.

Frau CHRISTINE REISCHER sorgte als langjährige Lehrstuhlsekretärin nicht nur für organisatorische Wundertaten, sondern belebte durch ihren warmherzigen Umgang in umfassender Weise den Arbeitsalltag. Vielen Dank dafür.

Und noch einmal danke: an DIRK BRUNS und DR. GOLE B. GORRST für weitere Inspirationen.

Für ihren pausenlosen Beistand und die Sicherung meines Wohlbefindens ganz lieben Dank an meine Frau JULIA und meine beiden Kinder JANICIA und JUVAL.

München, im September 2002

J. Hunsinger

Zusammenfassung

Hauptgegenstand dieser Arbeit ist ein neuartiges Verfahren zur automatischen maschinellen Erfassung mathematischer Formeln mittels natürlicher Handschrift, Sprache und Stiftgestik.

Die Besonderheit des dabei verfolgten Ansatzes liegt in der Integration aller notwendigen Systemkomponenten in einem erwartungsgetriebenen, *einstufig-probabilistischen* Decodierungsverfahren, das Handschrift- und Spracheingaben in eine semantische Darstellung mathematischer Formeln transformiert. Auf der Grundlage eines vorhandenen sprachverstehenden Systems wurde damit auf allen involvierten Abstraktionsebenen von der semantischen über die syntaktische bis zur morphologischen und Musterebene eine stochastische Modellierung erreicht, die die Anwendung einer vom Eingabemedium weitgehend unabhängigen *Maximum-a-posteriori-(MAP-)Klassifikation* erlaubt.

Eine wichtige Neuerung auf dem Teilgebiet der *Online*-Erkennung handgeschriebener Formeln liegt in der statistischen Beschreibung der zweidimensionalen Symbolanordnung einschließlich der Schriftgradvariation im Rahmen einer *kontextfreien Grammatik*. Die *Formelstrukturanalyse* fügt sich dadurch nahtlos in das Gesamtverfahren ein, wodurch unter anderem die sonst problematische *Segmentierung* auf Symbolebene erheblich vereinfacht wird.

Die einheitliche syntaktisch-semantische Repräsentation erleichtert zudem eine flexible *Fusion* sprach-/handschriftlicher Eingabedaten und deren nachfolgende *Transformation* zu Visualisierungs- und Nachbearbeitungszwecken. Durch Rücktransformation in die verwendete semantische Darstellungsform wird die vorrangig unterstützte *natürliche* Interaktion durch die *konventionelle* Interaktion innerhalb einer graphikorientierten Textverarbeitungsumgebung ergänzt.

Für eine fehlerrobuste Segmentierung und *Annotation* umfangreichen Trainingsmaterials wurden applikationsunabhängige graphische Arbeitsumgebungen mit vorwiegend stiftgestischen Bedienmechanismen entwickelt. Diese schließen auch eine systematische Verwaltung der verschiedenen externen Datenbasen mit ein, die für eine spätere Nachbearbeitung des Trainingskorpus sowie zum selektiven Zugriff des semantischen Decoders auf das benötigte domänenspezifische Wissen benötigt werden.

Als praxistaugliche Anwendung wird ein *multimodaler* Formeleditor vorgestellt, der vorzugsweise durch sequentielle Handschrift- und Sprachinteraktion betrieben wird. Querbezüge zwischen diesen Haupteingabekanälen werden vorrangig durch intuitive Stiftgestik oder alternativ anhand konventioneller Interaktion hergestellt. Ein weiteres Systemmerkmal ist die automatische *Übersetzung* handgeschriebener und typographischer in gesprochene Formeln.

Im Ergebnis können mathematische Formeln mit dem aus der vorliegenden Arbeit hervorgegangenen System mit einer Gesamterkennungsleistung von deutlich über 90 % bei weitgehend intuitiver Bedienweise am Computer erfasst werden. Auf einem Standard-PC mit INTEL Pentium-III-Prozessor (450 MHz) wird abhängig vom Formelumfang ein- bis dreifache Echtzeitverarbeitung erreicht.

Inhaltsverzeichnis

1

Einleitung

1.1 Motivation und Zielsetzung

Der Begriff *Mensch-Maschine-Kommunikation* (MMK) umfasst nach seiner heutigen Auslegung jede erdenkliche Form des Informationsaustausches zwischen Menschen auf der einen und technischen Systemen auf der anderen Seite. Im engeren Sinne, insbesondere im Hinblick auf den Umgang mit Computern (z.B. im Büro, im Privatbereich oder im Kraftfahrzeug) ist gewöhnlich von Benutzern (Anwendern) bzw. von Bedienumgebungen (Anwendungen) die Rede. Eines der wichtigsten Forschungsziele im Bereich der MMK besteht in einer möglichst weitgehenden Ausnutzung der naturgegebenen menschlichen Kommunikationsgewohnheiten, in erster Linie durch Anpassung der Ein- und Ausgabeschnittstellen technischer Arbeitsumgebungen an natürliche Interaktionskanäle wie gesprochene Sprache, Handschrift, Gestik und Mimik. Dabei stehen sowohl eine intuitive Bedienbarkeit als auch die Adaption solcher im allgemeinen *multimodalen*[1] Anwendungen an den jeweiligen Benutzer im Vordergrund [LAN02].

Die vorliegende Arbeit konzentriert sich, was die Anwendungsseite betrifft, auf die maschinelle Erfassung mathematischer Formeln zu Textverarbeitungs- oder auch Kalkulationszwecken. Solche Aufgaben fallen typischerweise im wissenschaftlich-technischen oder redaktionellen Büroumfeld an und stellen dort vergleichsweise hohe Anforderungen an den Benutzer: Die im typographischen Formelsatz enthaltene zweidimensionale Anordnung mathematischer Symbole, welche zudem mehrheitlich nicht auf einer Standardtastatur zur Verfügung stehen, erfordert entweder den Einsatz graphikbasierter Formeleditoren (z.B. MICROSOFT Word Formeleditor, ADOBE FrameMaker FrameMath) mit zum Teil sehr unübersichtlichen mausorientierten Bedienpaletten, oder aber die Verwendung spezieller Textsatzsysteme (z.B. LATEX) mittels formalsyntaktischer Markensprachen. Letztere entbehren meist der erwünschten intuitiven Bedienweise, da vor allem eine unmittelbare WYSIWYG[2]-Visualisierung fehlt.

---

[1] Die alternative oder auch kombinierte Nutzung verschiedener Eingabekanäle wird auch als *multimodale* Interaktion bezeichnet, in Analogie zur *multimedialen* oder *Multimedia*-Ausgabe.

[2] „What You See Is What You Get“

Mittlerweile haben sich graphikorientierte Formeleditoren des erwähnten Typs als Standardwerkzeuge für die Erfassung mathematischer Formeln durchgesetzt, maßgeblich im Zuge einer fortschreitenden Verbreitung der entsprechenden Desktop-Publishing-(DTP-)Systeme bzw. Text- und Bildverarbeitungsumgebungen, in die solche Editoren eingebettet sind. Bei näherer Betrachtung weist die dort vorgesehene *konventionelle*[3] Interaktion mehrere entscheidende Nachteile in den folgenden drei Kategorien auf:

- **Zeitaufwand.** Als direkte Konsequenz einer schrittweisen graphischen Erstellung mathematischer Ausdrücke mittels Maus und Tastatur ist der Zeitbedarf für das Setzen einer vollständigen Formel beträchtlich.

- **Ergonomie.** Selbst vergleichsweise einfache Formelbestandteile erfordern bereits verhältnismäßig komplizierte Maus-Tastatur-Interaktionen seitens des Anwenders. Dies bedeutet zum einen, dass eine intensive Einarbeitung unerlässlich ist, um mit der Bedienung genügend vertraut zu sein. Zum anderen ist es in den meisten Fällen notwendig, eine zu erstellende mathematische Formel komplett im Gedächtnis zu haben, bevor die Eingabe beginnen kann.

- **Intuitivität.** In Abhängigkeit von der zugrunde liegenden formalen Beschreibungssprache ist der Benutzer in der Regel gezwungen, eine mathematische Formel hierarchisch gemäß ihrer funktionalen Struktur aufzubauen. Eine annähernd sequenzielle bzw. natürliche Eingabereihenfolge – wie sie dem Benutzer vom Schreiben oder Sprechen her vertraut ist – wird gewöhnlich nicht unterstützt.

Um diesen Schwachpunkten entgegenzuwirken, liegt es nahe, die Interaktion zwischen Anwender und System natürlicher und zugleich effizienter zu gestalten. Als natürliche Modalitäten kommen speziell für die Erfassung mathematischer Formeln in erster Linie Handschrift und zusätzlich gesprochene Sprache in Frage. Der Nutzen, den die Integration dieser Kommunikationsmittel in einem neuartigen Formeleditor verspricht, soll durch die im nächsten Absatz beschriebene Vorab-Benutzerstudie etwas näher beleuchtet werden. Der Fokus lag dabei auf dem Aspekt Zeitaufwand, wohingegen Fragen der Ergonomie und der Intuitivität erst in einer späteren Projektphase, unter Berücksichtigung der realen Implementierung, untersucht wurden (siehe Benutzertest, Kap. 6).

Voruntersuchung. Zur Ermittlung des durchschnittlichen Zeitaufwands, den die verschiedenen Eingabemodalitäten dem Anwender abverlangen, wurde zunächst eine quantitative Testreihe nach folgender Maßgabe durchgeführt [HUN00A]:

Insgesamt sieben Testpersonen – alle mit der Bedienung graphischer Bedienoberflächen (GUI) grundsätzlich vertraut – wurden aufgefordert, zehn deutlich verschiedene mathematische Formeln am Computer zu erfassen. Dies geschah abwechselnd mittels 1) natürlich gesprochener Sprache, 2) natürlicher Handschrift bzw. 3) eines konventionellen graphischen Formeleditors[4]. Um Lerneffekte

[3] Hier und im Folgenden umfasst der Begriff *konventionell* herkömmliche Interaktionsformen, die über den haptischen Kanal, also in der Regel über Maus und Tastatur kommunizieren, im Gegensatz zur *natürlichen* Interaktion etwa mittels Handschrift oder Sprache.

[4] MICROSOFT Formeleditor Version 3.0 (deutsch)

zu vermeiden, wurde die zeitliche Abfolge der einzugebenden Formeln und ebenso der zu verwendenden Modalität unter den Probanden zufällig variiert. Drei Personen hatten bereits mit dem verwendeten konventionellen Formeleditor gearbeitet, so dass die Auswirkung einer vorherigen Einarbeitung bezüglich der maus-/tastaturbasierten Eingabe ebenfalls ausgewertet werden konnte. Es sei angemerkt, dass das Ziel dieser Studie lediglich ein Vergleich zeitlicher Größenordnungen in Abhängigkeit vom Eingabemedium war, ohne einen Anspruch auf statistische Signifikanz zu erheben. Die Ergebnisse sind in Tab. 1.1 zusammengefasst.

	Sprache	Hand-schrift	Maus/ Tastatur (Experte)	Maus/ Tastatur (Anfänger)
Mittlere Eingabezeit	10 s	20 s	100 s	140 s
Datenrate (normiert)	14	7	1.4	1

Tab. 1.1: Erfassung mathematischer Formeln. Durchschnittlich benötigte Eingabezeiten und relative Datenrate (normiert auf kleinsten Wert) unter Verwendung unterschiedlicher Eingabemodalitäten.

Es liegt auf der Hand, dass die handschriftliche gegenüber der konventionellen Eingabe einen signifikanten Zeitvorteil liefert – im Vergleich mit geübten Anwendern ergibt sich etwa ein Faktor 5. Bemerkenswert ist hingegen der zusätzlich zu erwartende Zeitgewinn durch die Verwendung natürlicher Sprache von ca. 50 % gegenüber Handschrift.

Dabei muss betont werden, dass keinerlei Systemantwortzeiten, wie sie beim Einsatz von Sprach- bzw. Handschrifterkennungskomponenten zu erwarten sind, in dieser Voruntersuchung berücksichtigt wurden. Im Rahmen dieses Projektes wurde indessen die Implementierung echtzeitnaher Erkennungsmodule angestrebt, so dass die zu erwartende Zeitersparnis bei der Formeleingabe dennoch nennenswert sein sollte. Es erscheint demnach lohnend, das Potential dieser beiden schnellsten und intuitivsten Modalitäten für die Erfassung mathematischer Formeln auszuschöpfen, zufriedenstellende Robustheit und Erkennungsleistung vorausgesetzt. Dies gilt umso mehr, wenn deren Einsatz in sinnvoller Weise – etwa in Orientierung an der zwischenmenschlichen Kommunikation – kombiniert werden kann.

Ausgehend von den oben beschriebenen Betrachtungen zielt die vorliegende Arbeit darauf ab, ein Konzept für einen neuartigen Formeleditor zu entwerfen und durch die Erarbeitung geeigneter Verfahren in einen lauffähigen Prototypen umzusetzen. Im nächsten Absatz werden die angestrebten Systemfunktionen, im darauffolgenden der vorgesehene Umfang an mathematischen Formelelementen vorgestellt. Der im resultierenden Gesamtsystem enthaltene Eigenbeitrag ist in Kap. 1.2 von den verwendeten Vorarbeiten abgegrenzt.

Funktionsumfang. Mit dem anvisierten Formeleditor soll es auf möglichst intuitive Weise möglich sein, mathematische Formeln schrittweise oder durchgängig auf direktem Wege digital zu erfassen. Das bevorzugte Eingabemedium wird dabei natürliche Handschrift sein, zumal diese nach allgemeiner Ansicht für eine solche Anwendung ideal geeignet ist. Unterstützend sollen natürliche Spra-

che sowie stiftgestische Bedienmechanismen angeboten werden, um mathematische Formeln zu erfassen, weiterzuverarbeiten und ggf. zu korrigieren.

Um die verschiedenen Kombinationsmöglichkeiten natürlicher und konventioneller Modalitäten voll ausschöpfen zu können, ist eine interaktive Arbeitsumgebung unerlässlich. Dies bedeutet vor allem, dass ein Online-Erkennungsverfahren mit schritthaltender Verarbeitung und geeignetem visuellen Feedback zu implementieren ist.

Für den Endanwender spielt die Frage, innerhalb welcher Anwendungsumgebung die Erkennungsergebnisse visualisiert werden und ggf. zur Weiterverarbeitung zur Verfügung stehen, eine bedeutende Rolle. Bei der Auswahl einer zu diesem Zweck geeigneten Zielanwendung sind in unserem Fall folgende Anforderungen zu stellen:

- Eine Standard-Softwareplattform mit hohem Verbreitungsgrad ist gegenüber einer Speziallösung (ob extern oder selbstentwickelt) zu bevorzugen. Dies deckt sich mit der Prämisse, typische Anwendergruppen von der Bedienung konventioneller Formeleditoren hin zur Verwendung natürlicher Modalitäten zu motivieren: Diese neuartigen Bedienmechanismen sollen nicht als Ersatz, sondern als sinnvolle Ergänzung bereits etablierter Arbeitsumgebungen verstanden werden.

- Es sollte sich um eine möglichst zeitgemäße, graphikorientierte Anwendungsumgebung handeln. Im Sinne einer intuitiven Mensch-Computer-Interaktion ist die Zielgruppe des vorliegenden Projektes in erster Linie unter mehr oder weniger versierten DTP-Anwendern, nicht aber unter Anhängern klassischer Textsatzsysteme (typischerweise LaTeX) anzusiedeln. Zusätzlich eröffnet sich dadurch die Möglichkeit, allseits gebräuchliche konventionelle (Maus-/Tastatur-)Interaktionstechniken mit dem Einsatz natürlicher Handschrift und Sprache zu kombinieren und damit dem Anwender größtmögliche Freiheit in seiner Arbeitsweise zu gewähren.

- Eine weitgehend offene Architektur oder zumindest die Unterstützung eines universellen Datenaustauschformates ist wünschenswert. Unter dieser Voraussetzung ist eine nahtlose Einbindung in das zu erstellende Gesamtsystem, ggf. unter Einbeziehung entsprechender Datentransformationsmodule, technisch realisierbar.

- Um eine möglichst vielseitige Verwendung zu ermöglichen, sollten sowohl umfangreiche Satzfunktionen als auch Kalkulations- bzw. Auswertungsfunktionen für mathematische Formeln vorhanden sein.

Die Standard-Publishing-Software Adobe FrameMaker mit integriertem konventionellen Formeleditor (FrameMath) erfüllt diese Kriterien und wurde daher als Zielanwendung an das Gesamtsystem angebunden (Einzelheiten siehe Kap. 3.6 und Kap. 7.3).

Formeldomäne. In Anlehnung an verwandte Arbeiten (z.B. [Grb95], [Kos98] und [Win97b]) wurde versucht, einen möglichst ausgewogenen Gesamtumfang an mathematischen Standard- und Spezialoperatoren sowie die gängigen Bezeichner für alphanumerische Größen in das System einzubeziehen. Dabei wird kein Anspruch auf Vollständigkeit gestellt, sondern es soll demonstriert werden, dass das aus dieser Arbeit hervorgegangene Grundverfahren in der Lage ist, bei einem Mi-

nimum domänenspezifischen Expertenwissens eine Vielzahl sehr unterschiedlicher mathematischer Formelbestandteile zuverlässig zu modellieren und zu decodieren.

Im Einzelnen werden die folgenden Formelkomponenten in beliebiger (mathematisch sinnvoller) Kombination unterstützt:

- Lateinisches und griechisches Alphabet (Klein- und Großschreib- bzw. -sprechweise)
- Arabische Ziffern, natürliche und Dezimalzahlen, „unendlich“ (∞)
- Arithmetische und logische Grundoperatoren ($+ - \cdot / = < \leq \geq > \rightarrow !$)
- Standardoperatoren (Wurzel mit Wurzelpotenz; Einfach- und Mehrfachintegral; indizierte Summe und Produkt; Faltung; runde, eckige und geschweifte Klammerung; diskrete und kontinuierliche Funktion)
- Sonderfunktionen (trigonometrische incl. hyperbolische und Arcus-Funktionen; Logarithmen und Exponentialfunktion; Limes; Argument-, Minimum- und Maximumfunktion)
- Exponenten, Indizes und Binomialkoeffizienten

Aus Anhang A.3 und A.5 ist der zugehörige Umfang an Handschriftsymbolen und sprachlichen Formulierungen (laut Training, siehe Kap. 5.2) zu entnehmen. Die zum Training der Handschrift-Modelle herangezogenen Referenzformeln (siehe Anhang A.2) vermitteln einen guten Gesamteindruck möglicher realistischer Kombinationen innerhalb dieser Domäne.

1.2 Lösungsansatz

Um sowohl gesprochene als auch handgeschriebene mathematische Formeln maschinell interpretieren zu können, sind für beide Modalitäten geeignete Verfahren zur Signalverarbeitung, Mustererkennung und syntaktisch-semantischen Modellierung erforderlich. Diese sind darüber hinaus so miteinander zu kombinieren, dass in beiden Fällen ein Gesamtklassifikationsverfahren entsteht, das in der Lage ist, die jeweils transportierte Symbol- und Strukturinformation zuverlässig – unter möglichst weitreichender Kommunikation zwischen den beteiligten Prozessen – zu extrahieren und dabei auftretende Mehrdeutigkeiten aufzulösen.

Da die semantische Modellierung nicht von der betrachteten Eingabemodalität abhängig sein sollte, ist hierfür ein gemeinsamer, die hierarchische Struktur mathematischer Formeln berücksichtigender Formalismus sinnvoll. Die in den Vorarbeiten von MÜLLER & STAHL [MÜL97][STA97B] eingeführte Semantische Gliederung erwies sich in dieser Hinsicht als idealer Ausgangspunkt für die Repräsentation und probabilistische Parametrisierung der untersuchten Formeldomäne. Um weiterhin die Vorzüge eines auch von diesen Autoren verfolgten einstufig-erwartungsgetriebenen Decodierungsansatzes nutzen zu können, wurde dessen Grundprinzip nicht nur für das formelbezogene Sprachverstehen übernommen, sondern überdies durch entsprechende Erweiterung und Anpassung auf den Bereich der Handschriftverarbeitung übertragen. Auf der syntaktischen Ebene gelang dabei eine

bimodale Modellierung, die mithilfe strukturell identischer Datenstrukturen (Syntaktisches Netzwerk bestehend aus Syntaktischen Modulen) und -operationen (Netzwerkübergänge und -emissionen) die Reproduktion und probabilistische Bewertung aller zulässigen sprachlichen und handschriftlichen Formeleingaben erlaubt. Die bestehende Analogie zwischen den syntaktischen Stilmitteln Sprech- und Schreibreihenfolge bzw. Wort- und Schriftzeichenwahl wurde ergänzt durch eine zusätzliche Modellkomponente für die handschriftspezifischen Maßnahmen zur zweidimensionalen Schriftzeichenpositionierung und -skalierung.

Die Semantische Gliederung als medienneutrale Darstellung dient zudem als gemeinsame Schnittstelle sowohl für die Fusion von Teileingaben aus unterschiedlichen Modalitäten als auch für die Hin- und Rücktransformation zur Visualisierung und konventionellen Weiterbearbeitung, und schließlich für die Übersetzung handgeschriebener bzw. typographischer Ausdrücke in gesprochene Form, so dass sie per Sprachsynthese ausgegeben werden können.

1.3 Gliederung der Arbeit

In **Kapitel 2** wird der aktuelle Forschungsstand zur Formelerkennung und zur multimodalen MMK, soweit für die vorliegende Arbeit relevant, erörtert. Nach einer Kurzdarstellung des Gesamtsystems (**Kapitel 3**) folgt in **Kapitel 4** eine ausführliche Beschreibung der semantischen Decodierung als Hauptkomponente des erarbeiteten Verfahrens. Die zum Training der verschiedenen Wissensbasen erforderlichen Maßnahmen und die wesentlichen Eigenschaften der daraus hervorgegangenen Modelle sind Gegenstand von **Kapitel 5**.

Anhand eines Benutzertests wird die multimodale Formelerfassung mittels natürlicher Handschrift, Sprache und Stiftgestik in **Kapitel 6** qualitativ und quantitativ auf ihre Gebrauchstauglichkeit überprüft. Daran schließt eine Erläuterung der möglichen Interaktionsformen innerhalb des umgesetzten Bedienkonzepts sowie der Formelweiterverarbeitung zum Zwecke der Visualisierung, Nachbearbeitung und Sprachausgabe an (**Kapitel 7**). **Kapitel 8** enthält eine Zusammenfassung der Systemimplementierung.

Die durch Test und Evaluierung des Gesamtsystems erhaltenen Ergebnisse sind in **Kapitel 9** dargelegt und werden abschließend in **Kapitel 10** diskutiert, worauf noch ein Ausblick auf zukünftige Verbesserungs- und Erweiterungsmöglichkeiten folgt.

2

Stand der Forschung

2.1 Erkennung mathematischer Formeln

Einordnung. Das Fachgebiet der maschinellen Formelerkennung[5] ist als anspruchsvolle Spezialdisziplin innerhalb der beiden Forschungsbereiche Handschriftverarbeitung und Dokumentanalyse anzusiedeln. Bereits in den 60er Jahren wurden erste – mitunter richtungsweisende – Arbeiten veröffentlicht [AND68], einen regelrechten Aufschwung erlebt diese Thematik jedoch erst seit den vergangenen ca. 15 Jahren. [BLO97] liefert eine recht gute Gesamtschau auf bisher verfolgte Ansätze und versucht diese auch in eine gewisse Systematik einzuordnen. Eine Bestandsaufnahme auf aktuellerem Stand und mehr vergleichender Natur bietet [CHA00].

Grundsätzlich ist zunächst zwischen *Online*- und *Offline*-Verfahren zu unterscheiden, je nachdem, ob und in welchem Maße Informationen über das zeitliche Zustandekommen des zu untersuchenden Schriftbildes vorliegen ([BLO97], S. 3). Der erstere Fall liegt in erster Linie dann vor, wenn mathematische Formeln mittels eines geeigneten Eingabegerätes, etwa eines Digitalisiertabletts mit integriertem LCD-Display, unter Verwendung natürlicher Handschrift erfasst werden, wie es auch in der vorliegenden Arbeit ausschließlich geschieht. Die benötigten Online-Eingabedaten liegen dann in der Regel sogleich in digitalisierter Form und damit auf direktem Wege zur Verarbeitung vor. Möchte man hingegen beispielsweise Papiervorlagen, die handgeschriebene oder auch typographisch gesetzte Formeln enthalten, analysieren, so muss sich das Erkennungsverfahren – üblicherweise nach Einscannen der Vorlage in ein gängiges Bilddatenformat – notgedrungen auf die zeitfreie, skalare Schriftbildinformation beschränken. Die hierfür entwickelten Offline-Erkennungsmethoden schließen aber nicht die Hinzunahme gewisser Quasi-Online-Merkmale aus, wie sie durch nachträgliche Rekonstruktion der Schriftbildentstehung, meist durch Einsatz von Energieminimierungsansätzen, aus den Original-Offline-Daten gewonnen werden können [JÄG98]. Umgekehrt ist es mitunter durchaus sinnvoll, schriftbildbezogene (und damit offline-artige) Merkmale in Online-

[5] Auf den Einsatz gesprochener Sprache zur Formelerfassung soll hier nicht näher eingegangen werden – in der Literatur überwiegen naturgemäß Arbeiten, bei denen zweidimensionale Formelnotationen ausgewertet werden.

Erkennungstechnologien mit einzubeziehen, wenn dadurch Mehrdeutigkeiten der Online-Modellierung aufgelöst werden können [MAN94][WIN96].

Allgemein erfordert eine Auswertung zweidimensionaler Formelnotation die Bewältigung zweier miteinander konkurrierender Teilprobleme: *Schriftzeichenerkennung* und *Formelstrukturanalyse* (oder auch Symbolanordnungsanalyse).

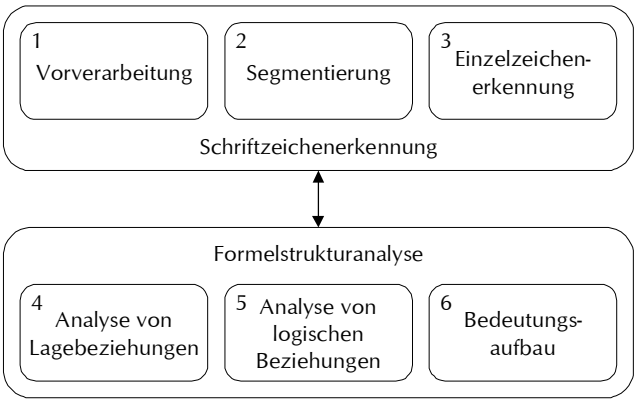


Abb. 2.1: Zusammensetzung der Formelerkennung aus sechs Teilprozessen nach [BLO95].

Wie in Abb. 2.1 schematisch dargestellt, lassen sich diese beiden Hauptaufgaben nach [BLO95] weiter in sechs Teilprozesse untergliedern. Die Autoren weisen darauf hin, dass diese Prozesse seriell oder auch parallel ablaufen können, wobei im letzteren Fall „frühere“ Prozesse kontextuelles Feedback von „späteren“ Prozessen erhalten. An dieser Stelle ist erkennbar, dass die signalnahen Vorgänge (1 bis 3) traditionell als „früh“ und ebenso die signalferneren Teilaufgaben (4 bis 6) als „spät“ eingestuft werden. Dies hat natürlich seine Ursache in der bis heute vorherrschenden signalgetriebenen Herangehensweise an Musterverarbeitungsaufgaben in praktisch allen Anwendungsbereichen. Es liegt ja, um bei den Formeln zu bleiben, durchaus nahe, sich erst dann mit dem mathematischen Bedeutungsgehalt zu befassen, wenn man eine Aussage über die enthaltenen mathematischen Symbole getroffen hat, die einen großen Teil der zugehörigen Information tragen.

Sieht man von der Vorverarbeitung (1) ab, die prinzipiell gesondert und ohne Zusatzwissen aus den übrigen Teilprozessen stattfindet, so kann man im Zusammenhang mit dem oben beschriebenen Grad der Parallelisierung die in der Literatur zu findenden Lösungswege in ein- und mehrstufige Verfahren unterteilen. Es überwiegen dabei deutlich (mindestens) zweistufige Ansätze, bei denen also – notwendigerweise signalgetrieben – zunächst eine Schriftzeichenerkennung über die gesamte zu analysierende Formel betrieben wird, bevor deren Ergebnis an eine nachgeschaltete Strukturanalysestufe weitergereicht und dort der mathematische Formelgehalt ermittelt wird. Da es wiederum diverse Kombinationsmöglichkeiten der genannten Teilprozesse in den beiden Hauptanalysephasen gibt, abgesehen vom Einsatz der unterschiedlichsten Verfahrensweisen innerhalb dieser Teilprozesse (statistisch, regelbasiert, graphentheoretisch, prozedural), sollen hier nur einige Arbeiten exemplarisch herausgegriffen werden, um die bestehenden Unterschiede zu beleuchten.

Aktuelle Verfahren. KOSMALA & RIGOLL [KOS98] verwenden diskrete *Hidden-Markov-Modelle* (HMM) zur simultanen Schriftzeichen-Segmentierung und -Erkennung über die gesamte Eingabe-

folge hinweg. Es gelang ihnen damit erstmals, die meist aufwendige und (aufgrund fehlenden Vorwissens) fehleranfällige Vorabsegmentierung bisheriger Arbeiten zu vermeiden und als integrierten Bestandteil der ersten Systemstufe zu implementieren. Auf die Einführung einer rekursiven 2D-Grammatik zur Formelstrukturmodellierung wurde dabei zunächst verzichtet, so dass sich erhebliche Einschränkungen der Schreibreihenfolge (zwecks Linearisierung der Eingabe) sowie eine Nichtunterstützung geschachtelter mathematischer Operatoren ergaben. Die zweite, nachgeschaltete Systemstufe zur Strukturanalyse wurde später [KOS99] durch eine kontextsensitive (nichtstochastische) *Graphgrammatik* ersetzt, was eine deutliche Flexibilisierung der Eingabe-Randbedingungen bewirkte. Das beschriebene Verfahren zählt – aufgrund der integrierten Segmentierung – zu den wenigen echt-zweistufigen (Bottom-Up-)Lösungen, während die meisten vergleichbaren Ansätze genau genommen als dreistufig einzuordnen sind.

Besondere Erwähnung verdient auch die dreistufige Bottom-Up-Strategie von WINKLER & LANG [WIN97A][WIN97B], bei der zwar eine externe, regelbasierte Schriftzeichensegmentierung vorgeschaltet ist, deren abschließende Beurteilung aber unter Verwendung eines *Symbolhypothesennetzes* in die nachfolgende HMM-Schriftzeichenerkennung verlagert wird, wodurch ein fließender Übergang zwischen den ersten beiden Analysestufen entsteht. Ein solcher *Soft-Decision Approach* findet zudem auch bei der ebenfalls datengetriebenen Strukturanalyse (dritte Systemstufe) mittels statistischer Bewertungsmaße Anwendung, so dass hier eines der ersten in weiten Teilen probabilistischen Gesamtverfahren entstand. Ein datenbasiertes Training der enthaltenen Wahrscheinlichkeiten ist allerdings nicht möglich, diese werden vielmehr aus fundierten Plausibilitätsbetrachtungen abgeschätzt.

HULL [HUL96] betreibt dagegen eine konsequent datenbasierte statistische Modellierung der wechselseitigen Symbolanordnung (vergleichbar der in der vorliegenden Arbeit vorgenommenen Offset-Modellierung) mithilfe einer trainierbaren *Kontextfreien Grammatik* (CFG), worin zweidimensionale Normalverteilungen für die Positionsanalyse herangezogen werden. Eine Strahlsuche auf der Grundlage der statistischen Hypothesenbewertung ermöglicht auch dort die Einschränkung des Suchraums auf handhabbare Größenordnungen.

Dem offensichtlichen Fehlerpotential aufgrund der allgemein vorgenommenen Trennung von Zeichenerkennung und Strukturanalyse (mehrstufiger Ansatz) begegnen MILLER & VIOLA [MIL98] durch Maßnahmen, die als *maintaining ambiguity* bezeichnet werden: Zur Durchführung der Strukturanalyse auf Basis einer speziellen geometrischen Grammatik werden Mehrdeutigkeiten der Symbolerkennung mitgeführt, um gegebenenfalls anhand des strukturellen Kontextes aufgelöst werden zu können. Auf vergleichbare Weise nutzen DIMITRIADIS & CORONADO [DIM95] einen Feedback-Mechanismus von der Strukturanalyse zur Erkennerstufe. Auch hier ist demnach ein Trend zur stärkeren Parallelisierung oder zumindest zur gegenseitigen Disambiguierung der beiden Hauptanalysestufen zu verzeichnen.

CHOU [CHO89] beschreibt – soweit aus der einschlägigen Fachliteratur ersichtlich – die erste konzeptuell einstufige Architektur eines Formelerkenners, indem er eine zweidimensionale stochastische CFG zum Top-Down-Parsing mathematischer Formeln bis hinab zur Schriftzeichenebene einsetzt. Obwohl darin lediglich eine Offline-Anwendung zur Analyse von (eingescannten) Typensatz-

Formeln vorgestellt wird (Pixel als Terminale der Grammatik), stellt diese Arbeit bis heute ein interessantes theoretisches Fundament auch für die Online-Auswertung handgeschriebener Eingabe dar.

Eine völlig fehlerfreie Funktionsweise handschriftbasierter Formeleditoren kann auch mittel- bis langfristig nicht erwartet werden. Dies hängt – abgesehen von der ohnehin kritischen 2D-Segmentierung – mit der praktisch unbegrenzten Vielfalt möglicher mathematischer Konstrukte (Baukastenprinzip) zusammen, die im Vergleich zur textuellen Handschrifteingabe nur wenige Anhaltspunkte für eine kontextuelle Disambiguierung liefern. Ein weiteres Indiz hierfür ist die gleichermaßen reduzierte Erkennungsleistung menschlicher Betrachter bei unsauber geschriebenen Formeln im Vergleich zu textueller Niederschrift ([SMI99], S. 84). Aus pragmatischer Sicht sind deshalb nähere Untersuchungen zu interaktiven Korrekturmöglichkeiten innerhalb solcher Systeme wünschenswert, denen sich erst in jüngerer Zeit einige Autoren zu widmen begannen. SMITHIES, NOVINS & ARVO [SMI99] schlagen hierzu ein aufgabenorientiertes Interaktionsschema vor, das durch entsprechende Visualisierungen in der Bedienoberfläche getragen wird: (a) Ausgleich einer fehlerhaften Schriftzeichensegmentierung mittels intuitiver Stiftgestik (ähnlich zur vorliegenden Arbeit), (b) Einzelzeichenkorrektur mit Hilfe lokaler Auswahlmenüs und (c) Strukturfehlerbehebung durch manuelle Drag-and-drop-Manipulation beliebiger Handschriftsegmente. MATSAKIS [MAT99] kombiniert benutzerinduzierte Maßnahmen wie die stiftbasierte Löschung von Teilausdrücken mit einer systemseitigen Rückweisung potentiell fehlerhafter Segmente. Zum erstgenannten Aspekt entwickelte KOSMALA [KOS00] ein erweitertes Korrekturschema, in dem ein weitgehend intuitives handschriftliches Entfernen, Ersetzen und Einfügen, sowie Undo-, Redo- und Refresh-Funktionalitäten enthalten sind.

Bewertung. Eine vergleichende Beurteilung der Leistungsfähigkeit bisher verfolgter Ansätze im Bereich der Formelverarbeitung steht noch aus und ließe sich derzeit nur unter großem Vorbehalt durchführen. Abgesehen davon, dass eine Vielzahl der veröffentlichten Verfahren nicht quantitativ ausgewertet wurden, hängt die angegebene Erkennungsleistung eines prototypischen handschriftbasierten Formeleditors von mehreren, meist stark variierenden Einflussfaktoren ab:

- Der angebotene Funktionsumfang unterscheidet sich hinsichtlich
  - erlaubter mathematischer Operatoren und Operanden,
  - deren Kombinierbarkeit und maximaler Schachtelungstiefe und
  - wählbarer Schreibweisen der verwendeten Schriftzeichen (auch im Zusammenhang mit schreiberabhängiger/-unabhängiger Modellierung).

- Es bestehen abweichende Einschränkungen der Schreibreihenfolge bezüglich einzelner Schriftzeichen und/oder Teilausdrücke (z.B. nachträgliche Ergänzung von Wurzelsymbolen oder Bruchstrichen bzw. Abfolge Zähler – Nenner – Bruchstrich).

- Im Einzelfall integrierte Autokorrekturmaßnahmen oder Rückweisungsmechanismen haben entsprechende Auswirkungen auf das Erkennungsergebnis.

- Der Zusammenhang zwischen Erkennungsrate und Speicher- bzw. Rechenaufwand sowie ggf. Trainingsaufwand der gewählten Implementierung muss berücksichtigt werden. Gerade im Hinblick auf eine spätere Anwendung im mobilen Bereich (stiftbasierte Handheld-PCs) sind gewis-

se Leistungseinbußen bei entsprechend sparsameren, möglichst echtzeitnahen Verfahren durchaus vertretbar.

- Die quantitative Evaluierung erfolgt je nach Autor auf der Basis sehr unterschiedlicher Testdaten, sowohl was den Umfang und die Komplexität der einzelnen Testformeln als auch die Handschriftqualität hinsichtlich ihrer Normabweichung betrifft. Zudem besteht ein großer Spielraum bei der Definition von Erkennungsleistung, etwa global (formelbezogen) oder lokal (auf Konstituenten bezogen, z.B. Unterscheidung von Struktur- und Symbolfehlern).

Die Situation wird zusätzlich dadurch erschwert, dass die genannten Kriterien oft nicht ausreichend und unmissverständlich dokumentiert sind. Die Schaffung einer universellen, standardisierten Testumgebung nach Art eines Benchmarks wäre offensichtlich von Vorteil, ist aufgrund der genannten Verschiedenheiten aber dementsprechend schwierig umzusetzen. Abschließend sei lediglich festgehalten, dass die Mehrheit der quantitativ analysierten aktuellen Verfahren mit Erkennungsraten von über 90 % bewertet werden.

Schlussfolgerungen. Aus der angeführten Literatur lassen sich drei Tendenzen ableiten, die als richtungsweisend für den in der vorliegenden Arbeit verfolgten Lösungsweg zu bezeichnen sind:

- Zunehmende Parallelisierung von Schriftzeichenerkennung und Strukturanalyse,
- verstärkter Einsatz trainierbarer statistischer Verfahrensweisen in beiden Bereichen und
- Integration flexibler interaktiver Korrekturmöglichkeiten.

Der einstufig-probabilistische Ansatz bietet ein Höchstmaß an Parallelisierung, indem bei inkrementeller Abarbeitung der Benutzereingabe Struktur- und Symbolinformation jeweils zyklisch miteinander verrechnet werden, bevor eine Entscheidung über die wahrscheinlichste Gesamtkonstellation möglicher Komponenten getroffen wird. Alle beteiligten Wissensbasen können aus authentischem Formelmaterial trainiert werden, dafür notwendiges Expertenwissen ist im Vorfeld global festgelegt. Die Formeleingabe erfolgt interaktiv bei fortlaufendem visuellen Feedback und ist multimodal (mittels Handschrift, Sprache und Stiftgestik bzw. konventionell) korrigierbar.

2.2 Multimodale Mensch-Maschine-Kommunikation

Der Wirkungsbereich aktueller Forschungsarbeiten zur multimodalen MMK ist ausgesprochen weit gefächert, so dass hier keine erschöpfende Darstellung laufender Aktivitäten gegeben werden soll und kann. Bereits bei der Zuordnung und Unterscheidung der verwandten Begriffe *multimodal* und *multimedial* gibt es bisher keine einheitliche Festlegung, wie eine neuere, ausführliche Studie zum Forschungsstand erneut bestätigt [BEN00]. Von den bestehenden, auch interdisziplinär differierenden Sichtweisen hierzu sei nur eine herausgegriffen, die von der EAGLES[6]-Initiative der Europäischen Kommission befürwortet wird:

[6] Expert Advisory Group on Language Engineering Standards (DG XIII, Linguistic Research and Engineering), siehe auch unter http://www.ilc.pi.cnr.it/EAGLES96/home.html und http://europa.eu.int/comm/index_de.htm .

- **Multimodale** Systeme umfassen die Repräsentation und Manipulation von Information aus unterschiedlichen menschlichen Kommunikationskanälen auf mehreren Abstraktionsebenen. Sie sind dadurch in der Lage, Bedeutungsinhalte aus Rohdaten zu gewinnen und umgekehrt menschlich wahrnehmbare Information aus einer symbolisch-abstrakten Darstellung zu generieren. Dies kann auch eine Informations-Transformation von einem Kanal zum anderen mit einschließen.

- **Multimediale** Systeme nutzen mehrere Geräte (wie Mikrophon, Lautsprecher, Touchscreen, Kamera) als Ein- und Ausgabemedien für vielfältige Datenformate. Eine automatische Abstraktion oder Transformation von Information findet nicht statt, die Codierung benötigter Meta-Information erfolgt gewöhnlich manuell.

Zur weiteren Abgrenzung möglicher Formen von Multimodalität existieren wiederum sehr gegensätzliche Klassifikationsschemata. Eine der aussagekräftigsten Taxonomien multimodaler Systeme zielt auf die *Integration* der vorhandenen Modalitäten ab und wurde von NIGAY & COUTAZ vorgeschlagen [NIG93]. Es gibt demnach zwei entscheidende „Freiheitsgrade“ für die multimodale Interaktion, nämlich die Art des Einsatzes (Interaktion) und der Verrechnung (Fusion) verschiedener Modalitäten. Abb. 2.2 zeigt die sich ergebenden vier Grundkategorien multimodaler Anwendungen, je nachdem ob Datenfusion erfolgt (kombiniert) oder nicht (unabhängig) bzw. ob die Modalitäten zeitlich getrennt (sequentiell) oder überlappend (parallel) einsetzbar sind. Der exklusive Fall stellt sozusagen die Minimalvariante eines multimodalen Systems dar, das wahlweise zwei (oder mehr) Interaktionskanäle unterstützt, ohne dass eine zeitliche oder inhaltliche Beziehung zwischen diesen entsteht. Je nach Anwendungsszenario treten die anderen drei Fälle mitunter in gemischter Form auf, wobei die synergistische Multimodalität als freieste Fassung (Datenfusion aus paralleler Interaktion) im Prinzip die alternierende und simultane Variante mit einschließt.

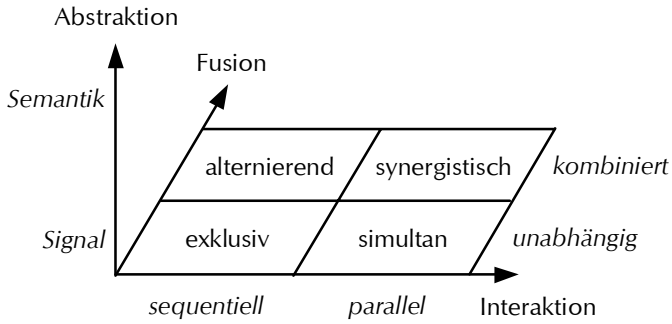


Abb. 2.2: Taxonomie multimodaler Systeme nach [Nig93] – „Multi-Feature System Design Space“

Ein weiteres wichtiges Integrationskriterium ist als zusätzliche Dimension angedeutet, und zwar die Abstraktionsebene, auf der gegebenenfalls eine Datenfusion betrieben wird. In der Literatur sind für die beiden in Abb. 2.2 angegebenen Extremfälle auch die Begriffe (*early*) *signal fusion* bzw. (*late*) *semantic fusion* üblich. Insbesondere bei stark gekoppelten und synchronen Modalitäten wie etwa Sprachsignal und Lippenbewegung in der audio-visuellen Sprachverarbeitung („speechreading“) ist eine signalnahe Kombination der Merkmalräume angebracht, um die vorhandene Informationsredundanz möglichst frühzeitig auszunutzen, siehe z.B. [ZHA00]. Bei schwacher Kopplung, beispielsweise zur kombinierten Auswertung von Sprache und (Stift-)Gestik, wird hingegen überwie-

gend auf der semantischen Ebene fusioniert. Das vielzitierte „Put-that-there"-Szenario von BOLT [BOL80] ist eines der frühesten, gleichwohl ein in der Praxis immer wiederkehrendes Anwendungsbeispiel für die semantisch-synergistische Integration zweier (hier komplementärer) Informationskanäle. OVIATT et al. geben in [OVI00] einen umfassenden Überblick zur neueren Entwicklung im Bereich der sprach- und stiftbasierten Multimodalität. Als Verfahren zur semantischen Datenfusion werden unter anderem *Multi-Agenten-* und auf *Typed Feature Structures* beruhende *Unifikations-Methoden* eingesetzt. VO & WAIBEL [VO97] definieren eine *Multimodale Grammatik* zur Modellierung multimodaler Eingabe und ein trainierbares *Multi-State Mutual Information Network* (MSMIN) für eine modalitätsübergreifende Maximum-a-posteriori-Klassifikation. In Anbetracht der Tatsache, dass den zu integrierenden Einzelkomponenten multimodaler Systeme meist statistische Erkennungstechnologien zugrunde liegen, ist eine ebenfalls statistische Herangehensweise an Fusionsprozesse besonders dann interessant, wenn Redundanz zwischen den beteiligten Eingabekanälen zur wechselseitigen Auflösung von Mehrdeutigkeiten (*mutual disambiguation*) ausgenutzt werden kann. Dem dadurch entsprechend stark anwachsenden Bedarf an multimodalem Trainingsmaterial begegnen WU et al. [WU99] mit beachtlichem Erfolg, indem sie die benötigten statistischen Parameter aufgrund theoretischer Überlegungen sehr zuverlässig abschätzen. ALTHOFF et al. [ALT02] verwenden einen adaptiven statistischen Ansatz unter Ausnutzung der Vorzüge *Evolutionärer Algorithmen*; es erfolgt eine *Genetische Multimodale Integration* prinzipiell beliebiger Eingangsdaten, indem unterschiedliche Fusionshypothesen durch Selektion und Vererbung sowie Mutation interagieren, bis ein geeignetes Resultat in Form einer konvergierten Population vorliegt.

Die obigen Ausführungen deuten bereits an, dass die Art und Weise der Integration verschiedener zu unterstützender Modalitäten, sowohl was die Frage der Abstraktionsebene als auch des Integrationsverfahrens betrifft, sehr stark von den Randbedingungen und Zielsetzungen der betrachteten Anwendungsdomäne abhängig ist. Unter den in Kap. 1.1 genannten Voraussetzungen stellt die Erfassung mathematischer Formeln einen Sonderfall dar, dem in der bisherigen Forschung vergleichsweise wenig Aufmerksamkeit geschenkt wurde: Die beiden Hauptmodalitäten natürliche Handschrift und Sprache (und zusätzlich die konventionelle Interaktion mittels Maus und Tastatur) sind hier – ungeachtet ihrer jeweiligen Schwächen und Stärken für spezielle Teilaufgaben – in ihren Ausdrucksmöglichkeiten weitgehend gleich mächtig. Dies gilt zwar ebenso für ein rein textbasiertes Anwendungsszenario wie etwa eine „intelligente" Schreibmaschine mit Sprach-, Handschrift- und konventioneller Eingabe, jedoch ergeben sich aus der zweidimensionalen Struktur mathematischer Ausdrücke deutlich andere Anforderungen, was sich unter anderem in der klaren Bevorzugung handschriftlicher Eingabe abzeichnet. Die Referenzierung handgeschriebener Teilausdrücke durch Stiftgestik ist ein weiterer Aspekt, der eine Abgrenzung zu ähnlichen Untersuchungen bewirkt.

Forschungsarbeiten anderer Gruppen, die sich mit einer hinreichend ähnlichen Aufgabenstellung zur Integration semantisch gleichwertiger Modalitäten befassen, konnten daher in der Literatur nicht gefunden werden. Von einigem Interesse ist in diesem Zusammenhang eine Usability-Untersuchung von OVIATT & OLSEN [OVI94], in der unter anderem das Benutzerverhalten in einer verwandten Domäne bewertet wurde: Bei der Durchführung einfacher wissenschaftlicher Berechnungen (Taschenrechnerfunktionen) unter freiem Einsatz natürlicher Handschrift und Sprache tendierten die Testpersonen in signifikanter Weise zu einem *kontrastiven* Einsatz dieser beiden Modalitäten. Ein Wechsel des Eingabemediums trat also am häufigsten dann auf, wenn während der Interaktion ein

entsprechender inhaltlicher oder funktionaler Wechsel zu verzeichnen war. Konkret handelte es sich um Übergänge von Original- zu Korrektureingabe, Daten- zu Befehlseingabe bzw. numerischer zu textueller Eingabe. Zusätzlich ergab sich ein auffällig geringer Anteil (< 0,5 %) simultaner Sprach-/Handschrifteingabe, wovon allerdings ein Großteil (85 %) auf Zahlenmaterial entfiel. Diese Ergebnisse stehen in guter Übereinstimmung mit dem hier erhobenen Benutzertest (siehe Kap. 6.5) und bestätigen damit einige Aspekte des alternierend-multimodalen Bedienkonzepts (Kap. 7).

3

Systemüberblick

3.1 Architektur

Die Grundarchitektur des aus dieser Arbeit hervorgegangenen multimodalen Formeleditors ist in Abb. 3.1 schematisch dargestellt. Der einstufig-probabilistische semantische Decoder stellt das Herzstück des Gesamtsystems dar, dessen Grundprinzip in Kap. 3.2 beschrieben wird. In Kap. 4 folgt eine ausführliche Darstellung seiner Funktionsweise.

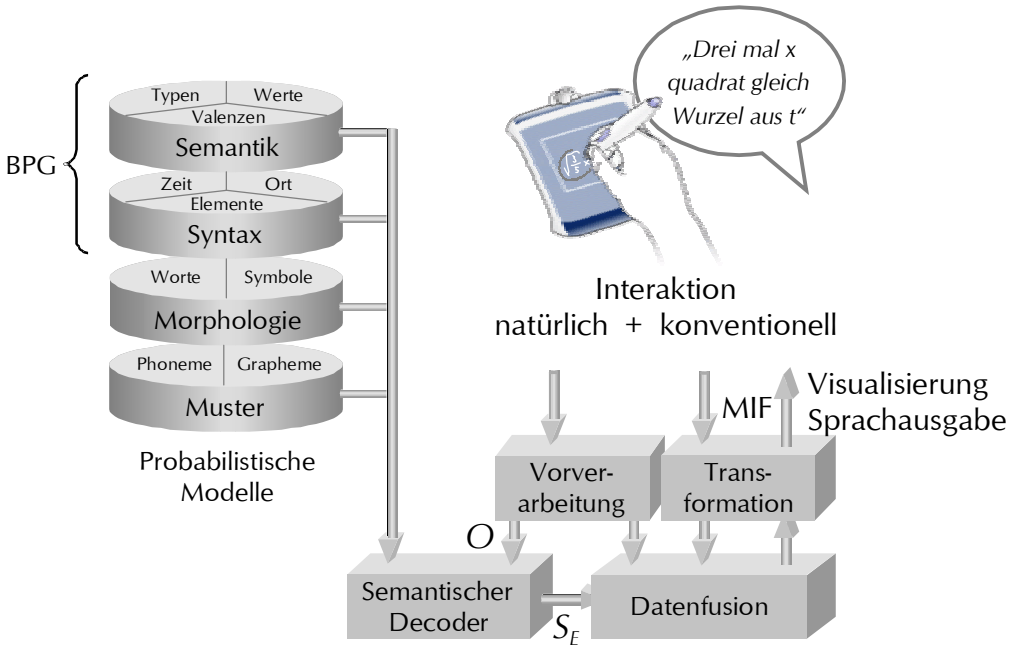


Abb. 3.1: Schematischer Aufbau des Gesamtsystems.

Der Decoder liefert den wahrscheinlichsten Bedeutungsinhalt S_E zur Beobachtungsfolge O (natürliche Interaktion) als Erkennungsergebnis. Dabei werden in Top-Down-Architektur alle Abstraktionsebenen von der semantischen bis zur Musterebene durchlaufen. Die zugehörigen probabilisti-

schen Modelle liefern alle für die Klassifikation benötigten statistischen Parameter, wobei Semantik und Syntax von einer kontextfreien *Bimodalen Probabilistischen Grammatik* (BPG) abgedeckt werden (siehe Kap. 4.1). Zusätzlich erfolgt eine Datenfusion zur Zusammenführung von Handschrift-, Sprach- und Stiftgestik-Interaktion sowie eine Datentransformation (MIF, siehe Kap. 3.6) zur Visualisierung und Weiterbearbeitung. Die konventionelle Interaktion ist mittels Datenrücktransformation in den Gesamtablauf integriert. Durch Umkehrung der Sprachdecodierung kann das Erkennungsergebnis aus Handschrift- und/oder konventioneller Eingabe bei Bedarf mittels Sprachsynthese akustisch wiedergegeben werden.

3.2 Decoder

Eine der wesentlichen Voraussetzungen für die erfolgreiche Decodierung handschriftlicher und natürlichsprachlicher Formulierungen ist eine möglichst realistische Modellierung des darin enthaltenen domänenspezifischen Wissens. Die dafür gewählte Repräsentationsform orientiert sich an der in [MÜL97] und [STA97B] verfolgten erwartungsgetriebenen Decodierungsstrategie und lässt sich als Schichtenmodell darstellen, in dem die beteiligten Abstraktionsebenen von der Signal- bzw. Musterebene über die morphologische und syntaktische bis zur semantischen Ebene übereinander aufgetragen sind (Abb. 3.1). Diese Ebenen werden während des Decodierungsvorgangs zyklisch in absteigender Reihenfolge (*top-down*) nach folgendem Schema durchlaufen:

(1) Expansion einer semantisch konsistenten (Teil-)Hypothese (Erwartung) im Suchraum,

(2) Projektion auf die syntaktische Ebene: Erzeugung zugehöriger (Teil-)Wortketten bzw. Schriftzeichenanordnungen,

(3) Projektion auf die lexikalisch-morphologische Ebene: Erzeugung zugehöriger Aussprachevarianten bzw. Schreibstile,

(4) Überprüfung der Übereinstimmung mit dem vorverarbeiteten Eingangssignal mittels Musterbewertung.

Bei jedem dieser vier Schritte findet eine probabilistische Bewertung aller neu erzeugten Hypothesen gemäß des probabilistischen Modells der jeweiligen Abstraktionsebene statt. Die Verwendung eines einheitlichen, modalitätsübergreifenden Decodierungsverfahrens für Handschrift- und Spracheingabe wurde mit Blick auf eine möglichst kompakte Realisierung der für die multimodale Interaktion erforderlichen Datenfusion angestrebt. Bezogen auf das oben genannte Durchlaufschema sind die zur Hypothesenexpansion und -projektion sowie zu deren probabilistischer Bewertung implementierten Mechanismen vom gerade verwendeten Eingabemedium unabhängig (abgesehen von einigen Erweiterungen für die 2D-Handschriftverarbeitung, die in der Sprachverarbeitung keine Entsprechung finden). Um ein übermäßiges Anwachsen des Suchraums zu verhindern, findet nach jedem Zyklus ein sogenanntes *Pruning* statt, bei dem alle diejenigen Hypothesen verworfen werden, deren Güte im Vergleich zur aktuell besten Hypothese einen festgelegten maximalen Abstand (Pruningdistanz) unterschreitet ([STA97B], S. 93 f.).

Der Übergang von (1) zu (2) bewirkt eine Projektion der (zeitfreien) semantischen Darstellung auf die Zeitachse, indem alle laut syntaktischem Modell erlaubten zeitlichen Abfolgen von Worten bzw. Schriftzeichen produziert werden. Die Abarbeitung des Eingangssignals in (4) erfolgt demgemäß chronologisch (im Echtzeitfall schritthaltend zur Eingabe) nach Vorgabe der in (2) erzeugten Wort- bzw. Schriftzeichen-Hypothesen. Sobald das Eingabesignal in dieser Weise vollständig ausgewertet wurde, wird die semantische Repräsentation der besten Gesamthypothese entsprechend einer *Maximum-a-posteriori*-Klassifikation (Kap. 4.2) als Decodierungsergebnis zurückgeliefert.

3.3 Probabilistische Modelle

Die aufeinander aufbauenden Modelle zur Beschreibung statistischer Bindungen in mathematischen Formeln (Abb. 3.1) stehen als externe, statische Wissensbasen zur Verfügung. Damit ist es je nach Bedarf möglich, bei Änderungen der gewünschten mathematischen Domäne oder der Schreib- bzw. Sprechgewohnheiten (z.B. bei Nutzerwechsel) diese Modelle auf einer oder mehreren Abstraktionsebenen zu erweitern, zu reduzieren oder auch auszutauschen, ohne Modifikationen an der Applikation selbst vornehmen zu müssen.

Um die zugehörigen Modellparameter mit geeigneten Werten zu belegen, werden die einzelnen Modelle mittels häufigkeitsbasierter Methoden auf der Grundlage von authentischem Formelmaterial trainiert (siehe Kap. 5). Für den prototypischen Formeleditor wurden hierzu Handschrift- und Spracheingaben von unterschiedlichen Testpersonen erhoben, vorverarbeitet und annotiert.

3.4 Natürliche Interaktion

Die allgemein bevorzugte und insbesondere zur intuitiven Formelerfassung prädestinierte Eingabemodalität ist die handschriftliche Notation, der dementsprechend auch als vorrangigem Interaktionsmodus die größte Aufmerksamkeit gewidmet wurde. Ausgehend von einer teilweise oder vollständig handgeschriebenen Formeleingabe sind Ergänzungen oder auch Korrekturen, sei es infolge systemseitiger Fehlerkennungen oder benutzerseitiger Fehleingaben, hauptsächlich durch den kombinierten Einsatz von Stiftgestik (zur Referenzierung eines bestehenden Teilausdrucks) und gesprochener Sprache (zur Ersetzung der referenzierten Formelbestandteile) vorgesehen. Der parallele Einsatz von Handschrift und Sprache findet nach den Ergebnissen eines eigens durchgeführten Benutzertests (Kap. 6) nur geringe Akzeptanz, höchstenfalls aber in Form eines „Mitsprechens“ einzelner Schriftzeichen, die einer erhöhten Verwechslungsgefahr unterliegen. (Bei vielen Schreibern ähneln sich z.B. die Symbole 'ω' und 'w', während sie in der gesprochenen Form 'Om@ga' bzw. 've:' nicht zu verwechseln sind.)

3.5 Konventionelle Interaktion

Die Anbindung an den in FrameMaker integrierten konventionellen Formeleditor dient einerseits der Visualisierung des Erkennungsergebnisses, andererseits bietet sich eine dortige Weiterbearbeitung bzw. nochmalige Fehlerbehebung mittels Maus und Tastatur an. Durch den Austausch von

Formeldaten zwischen handschriftlicher und konventioneller Bedienumgebung wird dieser Transfer geschaffen und zusätzlich die natürlichsprachliche Ausgabe konventionell eingegebener Formeln ermöglicht (siehe nächster Abschnitt).

3.6 Datenfusion und -transformation

Ziel der Datenfusion ist es, die semantischen Repräsentationen verschiedener Teileingaben aus Handschrift- und Sprachinteraktion so zu kombinieren, dass ein im Sinne des semantischen Modells konsistentes Gesamtergebnis resultiert. Im Falle der gesprochenen Korrektur eines handgeschriebenen Teilausdrucks besteht diese Aufgabe vorrangig in der Einfügung der zur Spracheingabe korrespondierenden in die handschriftbezogene Repräsentation, wobei der zu korrigierende Teilausdruck ersetzt werden soll. Die Identifikation des zu überschreibenden Teilbereiches der Datenstruktur erfolgt durch Auswertung der zugehörigen stiftgestischen Interaktion (siehe Kap. 7.2.3).

Die so gewonnene Gesamtrepräsentation einer Formel liegt als hierarchische *Semantische Gliederung* vor, deren Definition in Kap. 4.3 zu finden ist. Um sie in FrameMaker darstellen und editieren zu können, findet eine Datentransformation in das *Maker Interchange Format* (MIF) statt, das unter anderem eine spezielle Syntax-Spezifikation für die Codierung mathematischer Formeln in einer ebenfalls hierarchischen Datenstruktur umfasst [ADO97]. Die wechselseitige Wandlung (Kompilierung) von Formelinhalten zwischen diesen beiden Datenformaten ist in Kap. 7.3 näher beschrieben. Zusätzlich wird die Eigenschaft der semantischen Repräsentation *S* als *Intermedia*-Ebene genutzt, um eine maschinelle *Übersetzung* handgeschriebener bzw. konventionell gesetzter Formeln in gesprochene Form (auf dem Wege Handschriftanalyse $\xrightarrow{S}$ Sprachproduktion bzw. Formelsatz $\xrightarrow{MIF}$ Rücktransformation $\xrightarrow{S}$ Sprachproduktion) in Analogie zur klassischen Sprachübersetzung (Quellsprache $\rightarrow$ Zielsprache) zu demonstrieren (siehe Kap. 7.5).

4

Einstufig-probabilistische semantische Decodierung

4.1 Bimodale Probabilistische Grammatik

Um das in Kap. 3.2 skizzierte einstufige Klassifikationsverfahren sowohl auf Handschrift- als auch auf Spracheingabe einheitlich anwenden zu können, wird zunächst ein übergreifender Formalismus zur syntaktisch-semantischen Modellierung mathematischer Formeln benötigt. Während die semantische Repräsentation im Idealfall unabhängig von der Eingabemodalität sein sollte, müssen dabei Handschrift- und Sprachsyntax gleichermaßen durch geeignete Mechanismen berücksichtigt werden. Zu diesem Zweck wird zunächst formal eine *Bimodale Probabilistische Grammatik* (*BPG*) definiert:

$$BPG = \langle \Sigma, P, V, T \rangle; \quad \Sigma \Rightarrow T_1\, T_2 \cdots T_N : \quad \Sigma \xrightarrow{P_1} \cdots \rightarrow V_1\, V_2 \cdots V_M \rightarrow \cdots \xrightarrow{P_L} T_1\, T_2 \cdots T_N \qquad (4.1)$$

Das *Startsymbol* Σ repräsentiert den jeweils betrachteten Gesamtausdruck im Ausgangszustand und wird durch sukzessive Anwendung ($\rightarrow$) von *Ersetzungsregeln* oder *Produktionen P* in eine Folge von *Variablen V* und schließlich in eine Folge von *Terminalen T* überführt. Der gesamte Vorgang wird als *Ableitung* ($\Rightarrow$) des Gesamtausdrucks bezeichnet. Es werden ausschließlich solche Ersetzungsregeln zugelassen, deren Anwendung unabhängig vom linken und rechten Kontext der zu ersetzenden Variablen *V* erfolgt, so dass eine *kontextfreie* Phrasenstrukturgrammatik im Sinne von CHOMSKY [CHO59][GAZ85] vorliegt. Diese wird überdies als probabilistisch bezeichnet, da eine datenbasierte statistische Gewichtung der verschiedenen Ersetzungsregeln stattfindet, wie sie häufig zur Modellierung natürlicher Sprachen vorgenommen wird.

In der Domäne mathematischer Formeln besteht die Menge der auftretenden Variablen *V* hauptsächlich aus den unterschiedlichen erlaubten mathematischen Operatoren bzw. Funktionen, die natürlich gemäß entsprechender Ersetzungsregeln *P* in prinzipiell beliebig tief geschachtelter oder rekursiver Form auftreten können. Der bimodale Charakter dieser Beschreibung entsteht nun durch Einbeziehung der folgenden drei Kategorien von Terminalen *T*:

(1) gesprochene Worte (2) geschriebene Symbole (3) *Offsets*

Unter einem Offset ist hierbei die zweidimensionale, relative Ortsbeziehung zwischen handgeschriebenen Symbolen bzw. Symbolgruppen zu verstehen, die in direktem syntaktischem Zusammenhang stehen (wie z.B. Basisterm und Potenz eines Exponentialausdrucks). Dadurch werden sowohl Positionierung als auch Skalierung der betreffenden Schriftzeichen(-gruppen) zueinander modelliert. Der verfolgte Ansatz weist Parallelen zu sogenannten Graphgrammatiken auf, wie sie zur Beschreibung visueller Sprachen (etwa für die Erstellung von Flussdiagrammen) und in jüngerer Zeit auch vermehrt zur Modellierung handschriftlicher Formelnotation eingesetzt werden, siehe z.B. [GRB95].

Die praktische Implementierung dieser Grammatik innerhalb des Klassifikationsverfahrens ist in Kap. 4.3 und 4.4 genauer beschrieben. An dieser Stelle soll lediglich eine vereinfachte Darstellung der (Re-)Produktion eines handgeschriebenen mathematischen Ausdrucks am Beispiel von Abb. 4.1 gegeben werden:

Ableitung:

$$\Sigma \rightarrow V_{Bruch}\ V_{Offset(Bruch)} \rightarrow V_{Zähler}\ V_{Bruchop.}\ V_{Nenner}\ V_{Offset(Bruch)}$$
$$\rightarrow V_{nat.\,Zahl}\ V_{Bruchop.}\ V_{Quad.wurzel}\ V_{Offset(Quad.wurzel)}\ V_{Offset(Bruch)}$$
$$\rightarrow T_{'1'}\ T_{'\_'}\ T_{'\sqrt{\ }'}\ V_{Radikand}\ V_{Offset(Quad.wurzel)}\ V_{Offset(Bruch)}$$
$$\rightarrow T_{'1'}\ T_{'\_'}\ T_{'\sqrt{\ }'}\ V_{lat.\,Var.}\ V_{Offset(Quad.wurzel)}\ V_{Offset(Bruch)}$$
$$\rightarrow T_{'1'}\ T_{'\_'}\ T_{'\sqrt{\ }'}\ T_{'x'}\ T_{Offset(\sqrt{x})}\ T_{Offset(1/\sqrt{\ })}$$

Abb. 4.1: Ableitung eines Handschriftterms mit der BPG.

Die einzelnen Schriftzeichen werden über verschiedene Zwischenvariable (z.B. Bruch, Quad.-Wurzel, nat. Zahl) in der gewählten Schreibreihenfolge ($1 \rightarrow — \rightarrow \sqrt{\ } \rightarrow$ x) von links nach rechts als Terminale produziert. Jeweils nach vollständiger Produktion eines geschlossenen Teilausdrucks (z.B. $\sqrt{x}$) wird durch die zugehörigen Offset-Terminalen die zweidimensionale Anordnung der enthaltenen Schriftzeichen wiedergegeben.

Eine vollständige Angabe der BPG-Ersetzungsregeln und deren Bezug zur verwendeten syntaktisch-semantischen Repräsentation erfolgt in Kap. 4.4.3.

4.2 Maximum-a-posteriori-Klassifikation

Das einstufige sprachverstehende System [MÜL99], das als Ausgangspunkt für den hier vorgestellten einheitlichen Handschrift- und Sprachdecoder dient, vollführt eine *Maximum-a-posteriori-(MAP-)Klassifikation* durch Maximierung der Rückschlusswahrscheinlichkeit $P(S|O)$ von einer Beobachtungsfolge *O* (Observation) auf den zugehörigen Bedeutungsinhalt *S* (Semantik). Unter Anwendung der *BAYES'schen Regel* ist dies gleichbedeutend mit einer Maximierung der Größe $P(O|S)\,P(S)$, wobei $P(S)$ die *A-priori*-Wahrscheinlichkeit für das Auftreten des Bedeutungsinhal-

tes *S* angibt ([STA97B], S. 12-15). Damit erhält man als sogenannten *BAYES-Klassifikator* für den erkannten Bedeutungsinhalt S_E:

$$S_E = \underset{S}{\operatorname{argmax}}\, P(O|S)\,P(S) \tag{4.2}$$

Im Zuge der Übertragung dieses Ansatzes auf die Handschriftverarbeitung entstand ein verallgemeinertes Klassifikationsverfahren, das die einstufige semantische Decodierung handschriftlicher und sprachlicher Eingaben mit identischen Bewertungsmechanismen erlaubt. In diesem sogenannten *1STCLASS*[2]-Verfahren (*1 STage Compound LAnguage Syntactic-Semantic CLASSifier*) werden dementsprechend die folgenden beiden zusätzlichen Abstraktionsebenen zur Modellierung der bedingten Wahrscheinlichkeit $P(O|S)$ eingeführt:

- die Ebene des handschriftlichen oder sprachlichen *Ausdrucks* Ξ ('Xi' für eXpression) und
- die Ebene der *Phonem-* oder *Visemfolge*[7] Φ.

Die Modellierung von Ξ umfasst demnach einerseits mögliche Abfolgen von Schriftzeichen einschließlich ihrer (zweidimensionalen) Anordnung und andererseits mögliche Wortfolgen, und zwar jeweils im Zusammenhang mit einem gegebenen (mathematischen) Bedeutungsinhalt (s.u.). Diese Abstraktionsebene deckt also – über beide Eingabemodalitäten hinweg – den Bereich der *Syntax* ab. Die Elemente dieser syntaktischen Repräsentation, handgeschriebene Zeichen und gesprochene Worte, werden wiederum mittels der Darstellung Φ in ihre kleinsten interessierenden Bestandteile, nämlich Lauteinheiten oder Linienzüge[8], zerlegt, auf denen die letztendliche Modellierung der Beobachtungsfolge *O* mittels entsprechender Mustererkennungsverfahren (Kap. 4.6) basiert. Als Brücke zwischen syntaktischer und Signalebene ist hier auch von der *lexikalisch-morphologischen* Ebene die Rede (zur weiteren Erläuterung dieser beiden Begriffe siehe Kap. 4.5).

Unter der vereinfachenden Annahme, dass zwischen allen nunmehr zu betrachtenden Wahrscheinlichkeiten $P(O|\Phi)$, $P(\Phi|\Xi)$, $P(\Xi|S)$ und $P(S)$ statistische Unabhängigkeit gilt, ist damit eine Zerlegung der interessierenden Größe $P(O|S)$ möglich, die anderenfalls aufgrund der extrem großen Vielfalt möglicher Abbildungen $S \rightarrow O$ nicht mit vertretbarem Aufwand modellierbar wäre. Der tatsächlich verwendete MAP-Klassifikator lautet dann:

$$\boxed{S_E = \underset{S}{\operatorname{argmax}} \max_{\Xi} \max_{\Phi} \left[P(O|\Phi) \cdot P(\Phi|\Xi) \cdot P(\Xi|S) \cdot P(S) \right]} \tag{4.3}$$

---

(2) Um Verwirrung zu vermeiden: Die hochgestellte 2 soll hier *nicht* auf eine Fußnote verweisen, sondern den Ausdruck *CLASS* "quadrieren" (d.h. *1STCLASSCLASS*).

7 Der Begriff *Visem* wird gewöhnlich im Bereich der *Computer Vision* für die kleinsten zu modellierenden Einheiten verwendet. Da es sich bei dem hier beschriebenen Verfahren weniger um Bild- als um Graphikverarbeitung handelt, wäre *Graphem* eigentlich die bessere Wahl. Zugunsten der lautlichen Ähnlichkeit zwischen Φ ("Phi"), *Phonem* und *Visem* sei dieser Kunstgriff hier verziehen.

8 Zur Definition des Begriffes *Linienzug* siehe S. 39.

Die streng genommen erforderliche Summation über alle möglichen Ausdrücke Ξ und Phonem-/Visemfolgen Φ zu jedem Bedeutungsinhalt S wurde hier durch Maximumoperatoren ersetzt. Diese Maßnahme beruht auf der Annahme, dass nur die wahrscheinlichste Kombination von Ξ und Φ zu jedem S einen nennenswerten Anteil an der zugehörigen Produktionswahrscheinlichkeit $P(O|S)$ trägt. Tatsächlich trifft dies in hohem Maße auf die aus dem Training (Kap. 5) hervorgegangenen Wissensbasen zu. Das zu maximierende Produkt von Wahrscheinlichkeiten in Gl. 4.3 ist dann gleichbedeutend mit der Verbundwahrscheinlichkeit $P(O, \Phi, \Xi, S)$ dieser einen Kombination von Φ, Ξ und S zu gegebenem S (die Beobachtungsfolge O ist ja durch die zu bewertende Benutzereingabe festgelegt und damit konstant). Grundsätzlich sind alle derartigen Näherungen im hier beschriebenen Ansatz, auch die in den kommenden Kapiteln anzutreffenden, als vergleichsweise geringfügige Kompromisse zugunsten der Vereinfachung und Beschleunigung des Gesamtsystems zu verstehen. Ausschlaggebend für die Rechtfertigung dieser Vorgehensweise ist die Geschwindigkeit und Leistungsfähigkeit des resultierenden Gesamtverfahrens.

Der Zusammenhang zwischen Klassifikator (Gl. 4.3) und BPG (Gl. 4.1) wird deutlich, indem man die Ableitungswahrscheinlichkeit für einen syntaktischen Gesamtausdruck (Handschrift oder Sprache) mit der Verbundwahrscheinlichkeit für das Auftreten dieses Ausdrucks und des zugehörigen Bedeutungsinhaltes in Bezug setzt:

$$P(\Xi|S)\,P(S) = P(\Xi, S) = P(\Sigma \Rightarrow \Xi) = \prod_{i \in (\Sigma \Rightarrow \Xi)} P_i^{BPG} \tag{4.4}$$

Auf der rechten Seite ist die Ableitungswahrscheinlichkeit als Produkt über die statistischen Gewichte aller einzelnen Ersetzungsregeln P, die zur Ableitung $\Sigma \Rightarrow \Xi$ angewendet werden, angegeben. (Da mit dem Symbol P üblicherweise sowohl Wahrscheinlichkeiten als auch die genannten Produktionsregeln bezeichnet werden, steht P_i hier stellvertretend für das statistische Gewicht der i-ten angewandten Regel.) Die Modellierung der A-priori-Wahrscheinlichkeiten $P(S)$ (semantisches Modell) und der bedingten Wahrscheinlichkeiten $P(\Xi|S)$ (syntaktisches Modell) über geeignete probabilistische Parameter der zugehörigen Repräsentationen ist damit äquivalent zu den in Kap. 4.1 formal angeführten statistisch gewichteten Ersetzungsregeln und wird in den folgenden beiden Unterkapiteln 4.3 und 4.4 genauer beleuchtet. In Kap. 4.5 und 4.6 werden sodann die verbleibenden Größen $P(\Phi|\Xi)$ und $P(O|\Phi)$ zur Modellierung von morphologischer und Musterebene behandelt.

4.3 Semantische Ebene

4.3.1 Repräsentation

Die in Kap. 4.1 formal beschriebene Grammatik ist ein integrierter Bestandteil des einstufigen semantischen Decoders, der durch Anwendung eines verallgemeinerten Klassifikationsverfahrens Handschrift- und Spracheingaben in eine gemeinsame semantische Repräsentationsform übersetzt. Jede mathematische Formel wird dabei durch eine zugehörige *Semantische Gliederung S* dargestellt [MÜL97]. Die ursprüngliche Definition eines solchen Objektes wurde hinsichtlich ihres strukturellen Aufbaus weitgehend unverändert übernommen. Lediglich seine statistische Bewertung mittels

spezieller probabilistischer Parameter wurde in geeigneter Weise erweitert und an die Gegebenheiten der Domäne mathematischer Formeln angepasst.

Die Semantische Gliederung ist eine hierarchische Kombination von *N* verschiedenen *Semun*en (*sem*antic *un*its), die alle einem vorab festgelegten semantischen Inventar angehören. Jedes einzelne Semun ist durch seinen *Typ t*, seinen *Wert v* und seine von Typ und Wert abhängige semantische *Valenz* oder *Wertigkeit X*(*t*,*v*) (dies ist die Anzahl der jeweils angebundenen Nachfolgersemune) gekennzeichnet[9]:

$$S = \{s_n\},\ 1 \le n \le N;\quad s_n = (s_n.t,\ s_n.v,\ X(s_n.t,\ s_n.v)),\ X \ge 1 \tag{4.5}$$

In der hier beschriebenen Domäne repräsentiert jedes Semun einer Semantischen Gliederung einen bestimmten Operator[10] oder Operanden der betrachteten mathematischen Formel.

4.3.2 Parameter

Die semantische Repräsentation einer mathematischen Formel wird durch die folgenden Parameter probabilistisch bewertet:

- Die *Wurzelwahrscheinlichkeit* $\eta_0 = P(s_1.t)$ (4.6)
  für das Auftreten des Semuntyps *t* an der Wurzel einer Semantischen Gliederung (im Falle einer mathematischen Gleichung also etwa der dem Gleichsetzungsoperator zugeordnete Semuntyp),

- die *Wertwahrscheinlichkeiten* $\varepsilon_n = P(s_n.v \mid s_n.t)$ (4.7)
  für jeden existierenden Semunwert *v* zu einem gegebenen Semuntyp *t*, und

- die *Folgewahrscheinlichkeiten* $\eta_n = P(r(s_n) \mid s_n.t, s_n.v)$ (4.8)
  für das Auftreten einer *Nachfolgerschar* $r = \{r_{n1}, ..., r_{nX}\}$ von Semuntypen zum betrachteten Semun s_n in Abhängigkeit von dessen Typ *t* und Wert *v*.

Damit wird im Prinzip jeder denkbaren hierarchischen Kombination von sämtlichen zur Verfügung stehenden Semuntypen und -werten eine probabilistische Gesamtbewertung zugewiesen. De facto tritt jedoch in jeder realen Anwendungsdomäne, auch in der hier betrachteten, nur eine begrenzte Teilmenge solcher Kombinationen auf. Dieser Umstand führt dazu, dass in Gl. 4.6 - 4.8 eine Vielzahl theoretisch vorhandener Wahrscheinlichkeiten den Wert null erhalten und daher im semantischen Modell nicht berücksichtigt werden müssen. In [STA97B], S. 28, wurde bereits darauf hingewiesen, dass diese qualitative Aussage, ob also eine bestimmte Kombination von Semunen überhaupt sinnvoll ist oder nicht, zusammen mit der quantitativen Bewertung in Form der tatsächlich ermittelten Wahrscheinlichkeiten, die Bezeichnung der Gl. 4.6 - 4.8 als *stochastische Regeln* rechtfertigt. Der allgemeine Aufbau einer Semantischen Gliederung und ihre probabilistische Bewertung sind in Abb. 4.2 schematisch zusammengefasst.

---

[9] Die Schreibweise *a.b* identifiziert hier und im Weiteren eine Eigenschaft *b* eines Objektes *a*.

[10] Zur Bedeutung der zusätzlich auftretenden Meta-Semuntypen siehe Kap. 5.4.

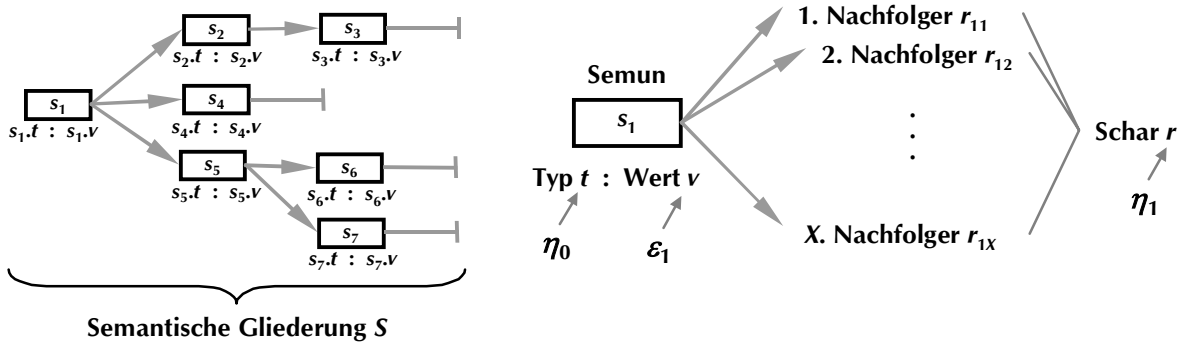


Abb. 4.2: Semantische Gliederung. Ausgehend vom Wurzelsemun s_1 verzweigt jedes X-wertige Semun zu seinen X Nachfolgersemunen, bis schließlich an jedem Teilast ein leerer Nachfolger (⊥-Symbol) auftritt. Die zugehörigen statistischen Parameter (siehe Gl. 4.6 - 4.8) beeinflussen wie angedeutet die Belegung der einzelnen Semun-Attribute *t*, *v* und *r*.

Es sei hier erwähnt, dass in der realen Implementierung anstelle der in Gl. 4.6 - 4.8 sowie aller weiter unten angegebenen Wahrscheinlichkeiten P durchgängig deren negativ logarithmierte Werte, sogenannte *Neglog-Scores Sc*, als Bewertungsmaße verwendet werden. Der allgemeine Zusammenhang zwischen diesen Größen lautet demnach:

$$Sc = -\ln P \tag{4.9}$$

Der Hauptvorteil dieser Bewertungsmaße Sc hinsichtlich der benötigten Rechenleistung besteht darin, dass man diese im Laufe der statistischen Gesamtbewertung aufaddiert, anstatt die entsprechenden Wahrscheinlichkeiten aufzumultiplizieren. Für das zur Klassifikation eingesetzte Suchverfahren lässt sich der resultierende Gesamthypothesen-Score im Sinne einer additiven Kostenfunktion darstellen, deren Minimierung die beste Gesamthypothese als Klassifikationsergebnis liefert (siehe Kap. 4.7).

Der Score-Anteil für die A-priori-Auftrittswahrscheinlichkeit $P(S)$ einer bestimmten Semantischen Gliederung S gemäß Gl. 4.3 errechnet sich demnach durch Summation der Einzelscores für die zugehörigen Wurzel-, Wert- und Folgewahrscheinlichkeiten über alle enthaltenen Semune $\{s_1,\ldots, s_N\}$:

$$Sc(S) = Sc(\eta_0) + \sum_{n=1}^{N} \left[Sc(\varepsilon_n) + Sc(\eta_n) \right] \tag{4.10}$$

4.4 Syntaktische Ebene

4.4.1 Repräsentation

In der syntaktischen Ebene wird jedem Semun einer gegebenen Semantischen Gliederung genau ein sogenanntes *Syntaktisches Modul* (SM) zugewiesen. Die syntaktischen Merkmale natürlicher Handschrift und Sprache werden darin mittels zweier gekoppelter stochastischer Prozesse modelliert:

(1) *Übergänge* zwischen je zwei *Knoten* eines SMs und

(2) *Emissionen* je eines syntaktischen *Elements e* (dies kann ein handgeschriebenes Symbol oder ein gesprochenes Wort sein) bzw. einer Folge von *Offset-Vektoren* $\vec{o}$ zwischen assoziierten Symbolen oder Symbolgruppen.

Durch den Übergangsprozess wird die bei der Eingabe verwendete Schreib- bzw. Sprechreihenfolge modelliert, wohingegen der Emissionsprozess Variationen in der Wahl von Worten, Schriftzeichen und der (relativen) Schriftzeichenanordnung Rechnung trägt. Die Gesamtheit der einer vollständigen Semantischen Gliederung zugehörigen Syntaktischen Module ist zu einem *Syntaktischen Netzwerk* (SN) wie folgt verknüpft:

$$\begin{aligned} SN &= \{SM_n\},\ 1 \le n \le N; \\ SM_n &= \{St_n, E_n, A_{n1}, \ldots, A_{nX}, B_{n1}, \ldots, B_{nY}, C_n\} \end{aligned} \tag{4.11}$$

Beginnend bei dem zum Wurzelsemun s_1 korrespondierenden Syntaktischen Modul SM_1 wird jedes einzelne Modul SM_n über seinen *Startknoten* St_n betreten und (nach Durchlaufen der übrigen Knoten) über seinen *Endknoten* E_n wieder verlassen. Alle seine Nachfolger-SMe sind über $X(s_n.t, s_n.v)$ individuelle *Nachfolgerknoten* A_{nk}, $1 \le k \le X$, an ihr gemeinsames Vorgänger-SM angebunden. Jeder dieser Nachfolgerknoten ist außerdem für die Emission der zugehörigen Folge von Offset-Vektoren zuständig. Diese Folge umfasst genau die handgeschriebenen Schriftzeichen(-gruppen), die dem an den betrachteten Nachfolgerknoten angebundenen Teil des Syntaktischen Netzwerkes zugeordnet sind. Zwischen diesen werden die benötigten (relativen) Offset-Vektoren jeweils paarweise berechnet [HUN00B]. Zudem existiert eine typ- und wertabhängige Anzahl $Y(s_n.t, s_n.v) \ge 0$ von (+)*Elementknoten* B_{nl}, $1 \le l \le Y$, aus denen jeweils ein (+)Element e^+ (Wort oder Schriftzeichen) der Eingabefolge emittiert wird. Optional kann zusätzlich der (–)*Elementknoten* C_n zur Emission eines (–)Elementes e^- betreten werden. Dieser Mechanismus dient speziell zur Modellierung von Füllworten bzw. umschreibenden sprachlichen Formulierungen, während bei der Handschrifteingabe gewöhnlich alle Schriftzeichen syntaktisch relevant sind und daher ausschließlich als (+)Elemente modelliert werden.

Auf diese Weise wird die Struktur der zugehörigen Semantischen Gliederung *S* eindeutig auf das Syntaktische Netzwerk *SN* abgebildet. Innerhalb des semantischen Decoders besteht der Zweck dieser Abbildung in einer Projektion der hierarchischen semantischen Repräsentation auf die Zeitachse, um dadurch jede zu erwartende syntaktische Realisierung einer erlaubten mathematischen Formel generieren bzw. reproduzieren zu können. Dabei entspricht jede mögliche Schreib- bzw. Sprechvariante einer gegebenen Formel genau einem geschlossenen Pfad durch das Syntaktische Netzwerk, einschließlich eines vollständigen Satzes aller zugehörigen Element- und Offsetemissionen. In Abb. 4.3 ist der Zusammenhang zwischen Eingabe, semantischer und syntaktischer Repräsentation an einem einfachen Beispiel illustriert.

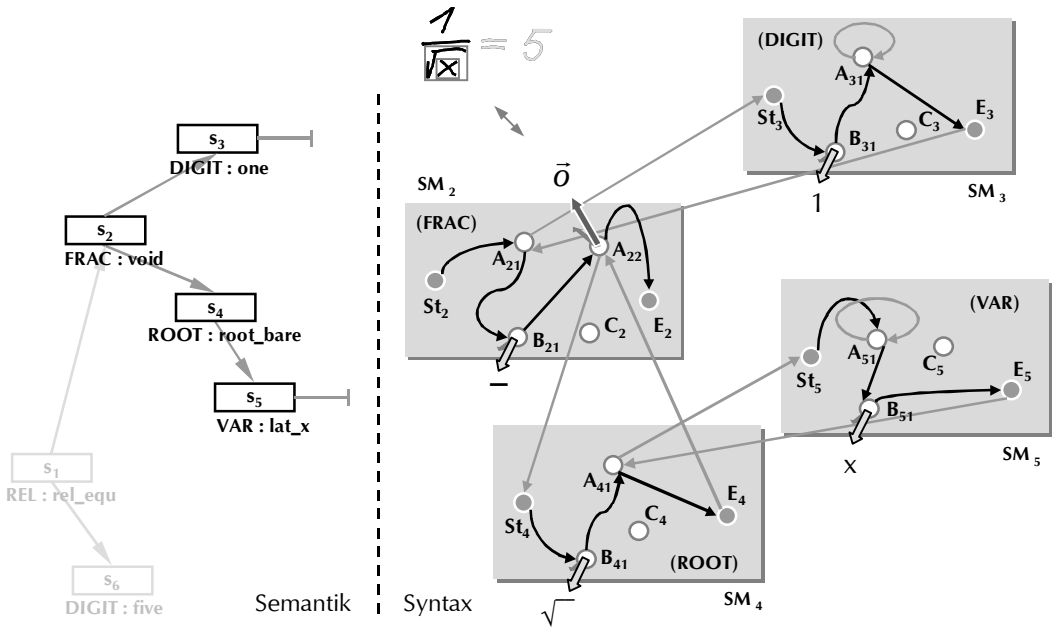


Abb. 4.3: Semantische Gliederung und Syntaktisches Netzwerk (vereinfachte Darstellung). In diesem Beispiel wird lediglich die linke Seite der handschriftlich eingegebenen Gleichung (oben) näher betrachtet. Die übrigen Bestandteile der Eingabe sowie der Semantischen Gliederung sind daher nur angedeutet.

Der angegebene Pfad durch das SN entspricht der gewählten Schreibreihenfolge ($1 \rightarrow — \rightarrow \sqrt{\ } \rightarrow$ x), die einzelnen Elementemissionen sind durch die entsprechenden Schriftzeichensymbole signalisiert. In dem zum Bruchoperator (Semuntyp FRAC) gehörenden Modul SM_2 findet die Emission eines Offset-Vektors $\vec{o}$ aus dem Nachfolgerknoten A_{22} statt, mit dem die relative Anordnung von Wurzelzeichen und Kleinbuchstabe x modelliert wird. Diese beiden Schriftzeichen wurden zuvor innerhalb des an A_{22} angebundenen Teilnetzwerks (SM_4 und SM_5) emittiert. Die restlichen zur Bewertung des hier untersuchten Bruches notwendigen Offsetemissionen (relative Anordnung von Bruchstrich, Zähler und Nenner) erfolgen nach der Rückkehr in den zum gesamten Bruch korrespondierenden Nachfolgerknoten A_{11} des Vorgänger-Moduls SM_1 (hier nicht dargestellt).

4.4.2 Parameter

Zur probabilistischen Beschreibung dieser syntaktischen Repräsentation müssen wiederum unterschiedliche Typen von Parametern ermittelt werden:

- Die typ- und wertspezifischen *Offsetwahrscheinlichkeiten*

$$\alpha_{nk} = P\left(A_{nk} \rightarrow \left\{\vec{o}\right\}_1^K \mid s_n.t, s_n.v\right), \quad 1 \le k \le X \tag{4.12}$$

  gewichten die Emission der Folge $\left\{\vec{o}\right\}_1^K$ von Offset-Vektoren aus dem jeweiligen Nachfolgerknoten A_{nk}. Die genaue Vorgehensweise bei der (rekursiven) Berechnung dieser Folge ist in Kap. 4.4.4 erläutert.

- Die *Elementwahrscheinlichkeiten*

$$\beta_{nl} = P\left(B_{nl} \to e^{+} \mid s_n.t, s_n.\nu\right), \quad 1 \le l \le Y \tag{4.13}$$

$$\gamma_n = P\left(C_n \to e^{-} \mid s_n.t\right) \tag{4.14}$$

tragen der Emission verschiedener zum jeweiligen Elementknoten vorhandener (+)Elemente (typ- und wertabhängig) bzw. (–)Elemente (typabhängig) Rechnung. Aus Konsistenzgründen ([STA97B], S. 36) wird γ_n=1 gesetzt, falls der Knoten C_n nicht betreten wurde (siehe auch Gl. 4.24).

- Unterschiedliche mögliche Pfade durch ein gegebenes Syntaktisches Modul SM_n werden durch eine Matrix von *Übergangswahrscheinlichkeiten*

$$\Delta_n = \left[\delta_n^{az} = P\left(a \to z \mid s_n.t, s_n.\nu, a\right)\right], \quad a, z \in SM_n \tag{4.15}$$

statistisch gewichtet, wobei $a \to z$ einen jeweils erlaubten Übergang vom Ausgangsknoten a zum Zielknoten z bezeichnet. Zusätzlich gilt die Vorschrift, dass ausgehend vom Startknoten St_n jeder einzelne Knoten – mit Ausnahme des optionalen (–)Elementknotens C_n – genau einmal durchlaufen wird, bevor der Endknoten E_n erreicht wird.

Abb. 4.4 zeigt den schematischen Aufbau eines Syntaktischen Moduls und die Zuordnung der beschriebenen Modellparameter im Überblick.

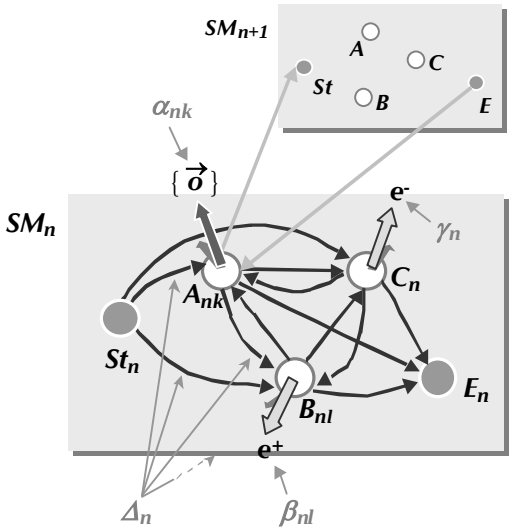


Abb. 4.4: Syntaktisches Modul. Der Nachfolgerknoten A_{nk} steht stellvertretend für eine typ- und wertabhängige Anzahl $X > 0$ von Nachfolgerknoten, $1 \le k \le X$. Ebenso typ- und wertabhängig ist die Anzahl $Y \ge 0$ der (+)Elementknoten B_{nl}, $1 \le l \le Y$. Die Bedeutung der ebenfalls dargestellten probabilistischen Modellparameter α, β, γ, und Δ und ihre Wirkung auf Übergänge und Emissionen sind oben näher beschrieben.

4.4.3 Bezug zur Grammatik

Nachdem nun sämtliche zur syntaktisch-semantischen Modellierung benötigten Datenstrukturen und probabilistischen Parameter definiert sind, lassen sich die in Kap. 4.1 erwähnten Ersetzungsregeln und deren statistische Bewertung unter Beachtung von Gl. 4.4 im Einzelnen aufstellen:

(a) $\Sigma \xrightarrow{\eta_0} s_1\, A_{01}$;

(b) $s_n \xrightarrow{\varepsilon_n \cdot \eta_n \cdot \prod_{az \in Pfad(SM_n)} \delta_n^{az}} \underbrace{B_{n1} \cdots B_{nY}\,(C_n)\, s_{r_{n1}}\, A_{n1} \cdots s_{r_{nX}}\, A_{nX}}_{\text{Permutation gemäß } Pfad(SM_n)} \quad \vee \quad s_n \rightarrow '\varepsilon'$

(c) $B_{nl} \xrightarrow{\beta_{nl}} e^+; \quad C_n \xrightarrow{\gamma_n} e^-; \quad A_{nk} \xrightarrow{\alpha_{nk}} \{\vec{o}\} \quad \vee \quad A_{nk} \rightarrow '\varepsilon'$

Zu (a): Das Startsymbol wird grundsätzlich durch je eine Variable für das Wurzelsemun s_1 und für den ihm übergeordneten (Pseudo-)Nachfolgerknoten[11] A_{01} ersetzt. Diese Ersetzung ist mit der vom Semuntyp abhängigen Wurzelwahrscheinlichkeit (Gl. 4.6) gewichtet.

Zu (b): Sodann erfolgt die rekursive Ersetzung jeder neu erzeugten Semunvariablen s_n durch eine Folge von Variablen, die alle Element- und Nachfolgerknoten (*C*-Knoten optional) des zugehörigen Syntaktischen Moduls SM_n sowie die Nachfolgersemune selbst repräsentieren. Die statistische Gewichtung ergibt sich als das Produkt aus den zugehörigen semantischen Wert- und Folgewahrscheinlichkeiten sowie den syntaktischen Übergangswahrscheinlichkeiten entlang des gewählten Pfades durch SM_n. Nach diesem Pfad richtet sich auch die Anordnung der genannten Variablen. Das auch in (a) anzutreffende feste Abfolgeschema $s_x\, A_{yz}$ ist dafür zuständig, die Offsetproduktionen laut (c) im Anschluss an die zugehörigen Elementproduktionen zu erzwingen.
Alternativ wird eine Semunvariable durch das leere ε-Symbol ersetzt, falls der Semuntyp 'leer' vorliegt (Ende eines Teilastes der Semantischen Gliederung).

Zu (c): Zur Produktion der verschiedenen Arten von Terminalen dienen Ersetzungen der Variablen für die jeweiligen Elementknoten durch zugehörige Elemente (Worte oder Schriftzeichen, Gewichtung mit Elementwahrscheinlichkeiten) bzw. für die Nachfolgerknoten durch zugehörige Folgen von Offset-Vektoren (Gewichtung mit Offsetwahrscheinlichkeiten). Alternativ zum letzteren Fall wird wiederum ein leeres ε-Symbol produziert, falls entweder keine Handschrift vorliegt oder das zugehörige Nachfolgersemun vom Typ 'leer' ist.

Abweichend von der in [STA97B], S. 31 f. gegebenen (nicht-probabilistischen) Darstellung ist in den hier vorgestellten Ersetzungsregeln das semantische Wissen einschließlich seiner statistischen Bewertung voll integriert. Im Gegensatz zu klassischen kontextfreien Grammatiken liefert die BPG

[11] Da ein Nachfolgerknoten laut syntaktischem Modell für die Offsetproduktion zum jeweils angebundenen Nachfolger-SM zuständig ist (Gl. 4.12 und 4.23), wird aus Konsistenzgründen ein zusätzlicher Knoten A_{01} definiert, der auf das Wurzelsemun verweist. Diese Vorgehensweise ist vergleichbar mit der bei Baum-Datenstrukturen üblichen Verwendung eines Pseudoeintrags *head*, der auf die eigentliche Wurzel eines Baumes zeigt [SED92].

damit eine im Rahmen des gewählten Ansatzes vollständige syntaktisch-semantische Beschreibung der betrachteten Anwendungsdomäne.

Die obigen Ausführungen dienen lediglich zur Einordnung der hier verwendeten Modellierung als kontextfreie Grammatik gemäß der in der Literatur üblichen Form. In der realen Implementierung des semantischen Decoders werden indessen als Datenstrukturen die Semantische Gliederung und das Syntaktische Netzwerk in der weiter oben definierten Fassung eingesetzt. Deren Nutzung zur Akkumulierung der probabilistischen Hypothesenbewertung innerhalb des einstufigen Suchverfahrens ist in Kap. 4.7 dargelegt.

4.4.4 Offsetberechnung

Bei der handschriftlichen Eingabe mathematischer Formeln trägt die zweidimensionale Anordnung der Schriftzeichen wesentliche syntaktische Information, die hingegen bei der Spracheingabe durch entsprechendes Zusatzvokabular (z.B. 'hoch', 'Index') oder auch mittels der Sprechreihenfolge (z.B. Zähler vor Nenner) transportiert werden muss.

Im Gegensatz zu den diskreten Phänomenen der Wort- bzw. Schriftzeichenwahl und der Sprech- bzw. Schreibreihenfolge handelt es sich bei der Symbolanordnung (zu der im übrigen hier auch die relative Schriftzeichenskalierung als weiteres syntaktisches Ausdrucksmittel gezählt wird) um einen kontinuierlichen Vorgang. Im Rahmen dieser Arbeit wurde entschieden, den zugehörigen Beitrag zur Formelklassifikation – wie in Kap. 4.4.1 und 4.4.2 beschrieben – in das syntaktische Modell zu integrieren. Dies hat den Vorteil, dass die vorgesehene kontextfreie Modellierung sich sehr gut in den bisherigen Formalismus einer kontextfreien probabilistischen Grammatik einfügen lässt. Andererseits wäre auch eine Kombination dieser Ortsmodellierung mit der eigentlichen Mustererkennungsaufgabe, also der Handschrifterkennung auf Schriftzeichenebene, ebenso denkbar gewesen.

Strukturkomponenten. Betrachtet man die Offsetemission aus einem beliebigen Nachfolgerknoten A_{nk} des Syntaktischen Moduls SM_n (vgl. Abb. 4.4), so dient diese zur Repräsentation der zweidimensionalen Struktur des zugehörigen mathematischen Teilausdrucks, dessen handschriftliche Bestandteile im nachfolgenden Syntaktischen Modul (SM_{n+1} für k=1) und dem darunter liegenden Teilnetzwerk zuvor emittiert wurden. (Die Offsetemission *nach* der Rückkehr in den Nachfolgerknoten A_{nk} stellt also sicher, dass alle dafür benötigten Schriftzeichen innerhalb der jeweiligen Formelhypothese bereits produziert worden sind.) Die Offsetberechnung findet nun *paarweise* zwischen je zwei *Strukturkomponenten* des betreffenden Teilausdrucks statt. Die Menge dieser Strukturkomponenten besteht per Definition aus allen Schriftzeichen, die aus den Elementknoten des genannten Nachfolgermoduls emittiert wurden, zuzüglich aller handgeschriebenen Teilausdrücke – jeweils als Einheit betrachtet – zu je einem Nachfolgerast dieses Moduls. Im Beispiel aus Abb. 4.3 bedeutet dies, dass etwa zur strukturellen Bewertung des Bruches (Semuntyp FRAC) mit einer Elementemission (Bruchstrich) und zwei Nachfolgertermen (Zähler und Nenner) insgesamt drei mögliche Paarungen dieser drei Strukturkomponenten zu berücksichtigen sind, also Bruchstrich-Zähler, Bruchstrich-Nenner und Zähler-Nenner. Von diesen dreien weist wiederum lediglich der Nenner ($\sqrt{x}$) eine innere Struktur auf, die bereits durch die zugehörige Offsetemission aus dem Nachfolgerknoten A_{22} bewertet wurde.

Um nun eine zuverlässige Modellierung der zweidimensionalen Formelstruktur zu erhalten, müssen die eingegebenen Handschriftdaten hinsichtlich der Symbolpositionen und -schriftgrößen einer geeigneten Vorverarbeitung unterzogen werden. Unter Einbeziehung notwendigen Expertenwissens sind sodann zugehörige Bewertungsmaße zu ermitteln, aus denen man die in Gl. 4.12 angegebenen Wahrscheinlichkeiten abschätzen kann.

Positionierungsmerkmale. Als Ausgangsbasis für die Analyse der relativen Symbolpositionen werden in Übereinstimmung mit den meisten verwandten Arbeiten umschreibende Rechtecke für jedes handgeschriebene Symbol verwendet. Hier ist zu beachten, dass die Zuordnung der verschiedenen Linienzüge[8] zu den jeweiligen Symbolen (Segmentierung) an dieser Stelle als gegeben vorausgesetzt werden kann. Der Grund besteht in der Tatsache, dass das hier verwendete Klassifikationsverfahren dem Grundprinzip einer erwartungsgetriebenen (Top-Down-)Decodierung folgt. Die tatsächliche oder besser gesagt die als richtig erkannte Segmentierung auf Symbolebene ergibt sich innerhalb dieses Ansatzes als impliziter Bestandteil des Erkennungsergebnisses, d.h. Erkennung und Segmentierung erfolgen simultan [HUN01A]. Für jedes zu untersuchende Symbol *i* wird zunächst ein *symbolspezifischer* Mittelpunkt $(\bar{x}_i, \bar{y}_i)$ seines umschreibenden Rechtecks $(x_i^{\min}, y_i^{\min}, x_i^{\max}, y_i^{\max})$ berechnet:

$$\bar{x}_i = x_i^{\min} + \lambda_x \left(x_i^{\max} - x_i^{\min} \right); \quad \bar{y}_i = y_i^{\min} + \lambda_y \left(y_i^{\max} - y_i^{\min} \right) \tag{4.16}$$

$\lambda_x, \lambda_y \in [0,1]$ sind auf Expertenwissen beruhende Gewichtungsfaktoren, mit denen durch die Symbolgeometrie verursachte Unterschiede in der Schriftzeichenpositionierung ausgeglichen werden. Dabei finden unter anderem gängige Schreibkonventionen, etwa hinsichtlich des vertikalen Versatzes unterschiedlicher Klein- und Großbuchstaben auf einer (gedachten) Grundlinie, Beachtung (vgl. [WIN97B], S. 133 f. + S. 153 f.). Als Positionierungsmerkmale für ein Symbolpaar *ij* werden die Differenzen dieser Symbolmittelpunkte nach Normierung auf die Gesamtabmessungen $(\Delta x_S, \Delta y_S)$ des zum betreffenden Semun gehörenden mathematischen Teilausdrucks (siehe Abb. 4.5) herangezogen.

$$\Delta_{ij}\,\tilde{x} = \frac{\bar{x}_i - \bar{x}_j}{x_S^{\max} - x_S^{\min}} = \frac{\Delta_{ij}\,\bar{x}}{\Delta x_S}; \quad \Delta_{ij}\,\tilde{y} = \frac{\bar{y}_i - \bar{y}_j}{y_S^{\max} - y_S^{\min}} = \frac{\Delta_{ij}\,\bar{y}}{\Delta y_S} \tag{4.17}$$

Für eine kontextfreie Modellierung ist es von entscheidendem Vorteil, dass die so gewonnenen Merkmale immer im Wertebereich $\Delta\tilde{x}, \Delta\tilde{y} \in [-1,1]$ liegen, so dass beispielsweise die innere Struktur von Zähler und Nenner keinen deutlichen Einfluss auf die Strukturbewertung eines Bruches liefert.

Da die Offsetberechnung und -emission prinzipiell nach der Rückkehr in den betreffenden Nachfolgerknoten aus dem dort angebundenen Nachfolgermodul erfolgt, ist eine Abarbeitung der Formelstruktur hinsichtlich der Schriftzeichenanordnung von den Blättern bis hin zur Wurzel der jeweiligen Semantischen Gliederung sichergestellt. Mit anderen Worten werden also ungeachtet der Schachtelungstiefe enthaltener mathematischer Operatoren zunächst die innersten Konstituenten einer betrachteten Formel bearbeitet, deren Strukturkomponenten dementsprechend ausschließlich

aus Einzelsymbolen[12] bestehen. Jede dieser innersten Konstituenten tritt wiederum als gesonderte Strukturkomponente bei der anschließenden Bearbeitung der ihr übergeordneten mathematischen Funktion auf, für die im Sinne von Gl. 4.16 ein nunmehr *komponentenspezifischer* Mittelpunkt $(\bar{x}_S, \bar{y}_S)$ berechnet werden muss. Die dabei erfolgende schrittweise Verschmelzung der Positionierungsmerkmale (und analog der Skalierungsmerkmale, s.u.) von einzelnen Schriftzeichen hin zu stetig anwachsenden Schriftzeichengruppen wird als *Komponentenfusion* bezeichnet und auf die folgende Weise betrieben:

$$\bar{x}_S = \sum_{i=1}^{n_S} \varepsilon_{x,i}\ \bar{x}_i \Bigg/ \sum_{i=1}^{n_S} \varepsilon_{x,i}\,;\ \ \bar{y}_S = \sum_{i=1}^{n_S} \varepsilon_{y,i}\ \bar{y}_i \Bigg/ \sum_{i=1}^{n_S} \varepsilon_{y,i} \tag{4.18}$$

Die binären Gewichtungsfaktoren $\varepsilon_{x(y),i}$ enthalten typ- und wertspezifisches Expertenwissen, um Beiträge der verschiedenen Strukturkomponenten i ($1 \le i \le n_S$) bezüglich der horizontalen bzw. vertikalen Mittelpunktberechnung zu berücksichtigen ($\varepsilon_{x(y)} = 1$) oder zu unterdrücken ($\varepsilon_{x(y)} = 0$). So wird beispielsweise bei der Komponentenfusion eines Exponentialausdrucks (Abb. 4.5) der Beitrag des Exponenten in beiden Achsrichtungen vernachlässigt, da erfahrungsgemäß für die Positionierung eines solchen Terms innerhalb übergeordneter mathematischer Funktionen der Basisausdruck ausschlaggebend ist ([LIE00], S. 27).

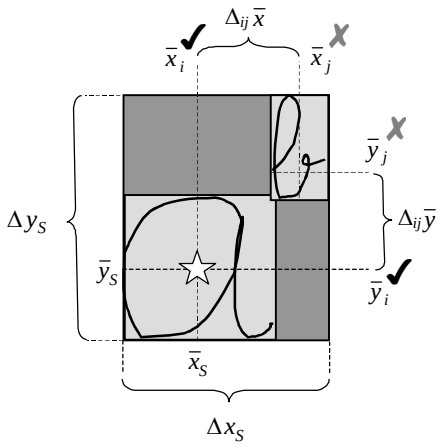


Abb. 4.5: Offsetberechnung (Positionsanteile). In diesem Beispiel wird die relative Position zweier einzelner Schriftzeichen betrachtet, die zusammen einen Potenzterm darstellen (Semuntyp POW). Basierend auf den zugehörigen umschreibenden Rechtecken $(x^{\min}, y^{\min}, x^{\max}, y^{\max})$ werden zunächst symbolspezifische Mittelpunkte $(\bar{x}, \bar{y})$ berechnet. Durch Differenzbildung und Normierung auf die Gesamtabmessungen des zum übergeordneten Semun (POW) korrespondierenden Teilausdrucks $(\Delta x_S, \Delta y_S)$ ergeben sich für beide Komponenten $(\Delta\tilde{x}, \Delta\tilde{y})$ Werte im Bereich [-1, 1] (siehe Gl. 4.17). Der komponentenspezifische Mittelpunkt $(\bar{x}_S, \bar{y}_S)$ gemäß Gl. 4.18 ist durch ☆ gekennzeichnet, ✔ bzw. ✘ bezeichnen die Komponenten, die zu dessen Berechnung berücksichtigt bzw. unterdrückt werden. Der Semunmittelpunkt wird zur rekursiven Offsetberechnung im übergeordneten Syntaktischen Modul weiterverarbeitet (Gl. 4.24).

[12] Abgesehen von den zugehörigen mathematischen Sonderzeichen sind dies im Normalfall Einzelziffern und/oder lateinische bzw. griechische Klein- und Großbuchstaben.

Die geschilderte Verfahrensweise entspricht einer rekursiven Formelstrukturanalyse von der Feinstruktur zur Grobstruktur und wird daher als *Inside-Out-Analyse* bezeichnet. Es ist zu beachten, dass bei der rekursiven Merkmalbildung nach Gl. 4.17 für jedes Komponentenpaar *ij* bei bereits fusionierten (also mehrere Schriftzeichen umfassenden) Strukturkomponenten *i*(*j*) die entsprechenden Mittelwertanteile $\bar{x}_{i(j)}, \bar{y}_{i(j)}$ durch die bei der zugehörigen Komponentenfusion ermittelten Werte $\bar{x}_S, \bar{y}_S$ (Gl. 4.18) ersetzt werden müssen. Dies gilt in gleicher Weise für die Komponentenfusion selbst, wenn die betreffenden zu fusionierenden Strukturkomponenten in Gl. 4.18 bereits aus vorherigen Komponentenfusionen hervorgegangen sind.

Skalierungsmerkmale. Die im vorigen Absatz behandelte Schriftzeichenpositionierung trägt ohne Zweifel einen gewichtigen Anteil der strukturellen Information bei der handschriftlichen mathematischen Notation. Eine zuverlässige und detaillierte Formelstrukturanalyse kann jedoch nur erfolgen, wenn zusätzlich syntaktisch relevante Variationen des verwendeten Schriftgrades, also die relative Skalierung korrelierter Schriftzeichen und -gruppen, in Betracht gezogen werden. Insbesondere geschachtelte Teilausdrücke wie etwa x_2^y werden gewöhnlich unter kombinierter Verwendung beider Stilmittel – Schriftzeichenpositionierung und -skalierung – niedergeschrieben. Im genannten Beispiel würde in der Regel durch fortschreitende Verkleinerung der Schriftgröße von x über 2 hin zu y die Eigenschaft des Teilausdrucks 2^y als Index des Teilausdrucks x betont. In komplexeren Fällen spielt das Zusammenwirken von Positionierungs- und Skalierungsmaßnahmen durch die damit verbundene Auflösung von Mehrdeutigkeiten eine noch größere Rolle.

Zur Berechnung der gewünschten Skalierungsmerkmale werden wiederum alle Paarungen der dem betrachteten Syntaktischen Modul zugeordneten Strukturkomponenten herangezogen. Für diese werden nach folgender Vorschrift logarithmierte Schriftgrößenverhältnisse ermittelt:

$$\Delta_{ij}\,\tilde{g} = \log_5 \frac{g_i}{g_j}\Bigg|_{g_i, g_j > 0} \quad ; \quad g_{i(j)} = \lambda_g \left| y_{i(j)}^{\max} - y_{i(j)}^{\min} \right| \tag{4.19}$$

Der Wert 5 als Basis des Logarithmus ist hier vorweggreifend als Resultat einer datenbasierten Optimierung der Merkmalgewinnung angegeben. Eine beobachtete Variation des in realistischen mathematischen Formeln vorkommenden Schriftgrades um einen Faktor 5 (z.B. im Bereich [10 pt ... 50 pt]) entspricht damit einem Wertebereich der Skalierungsmerkmale von $\Delta_{ij}\,\tilde{g} \in [-1, 1]$. Tatsächlich treten im hier erhobenen Trainingskorpus handgeschriebener Formeln etwas geringere Variationen der Schriftgröße auf, so dass die insgesamt erhaltenen Werte von Positionierungs- und Skalierungsmerkmalen ($\Delta_{ij}\,\tilde{x}, \Delta_{ij}\,\tilde{y}, \Delta_{ij}\,\tilde{g}$) ungefähr im gleichen Gesamtintervall streuen (siehe auch Abb. 4.6). Die Logarithmierung zur Basis 5 stellt also letztendlich sicher, dass Positionierungs- und Skalierungsmaßnahmen bei der Formelniederschrift einen zueinander ausgewogenen Einfluss auf die strukturelle Formelklassifikation liefern (siehe Gl. 4.23).

Wie üblich dient auch hier Expertenwissen zur Festlegung der symbolspezifischen Gewichtungsfaktoren $\lambda_g \in [0,1]$, durch welche die morphologisch bedingten Unterschiede in der vertikalen Ausdehnung verschiedener Schriftzeichen (bei jeweils gleichem Schriftgrad) berücksichtigt werden (zum Vergleich siehe auch [WIN97B], S. 134 f.). Der Sonderfall $\lambda_g = 0$ bewirkt den Ausschluss bestimmter Sonderzeichen von Skalierungsbetrachtungen. So tragen etwa die Abmessungen eines

handgeschriebenen Minuszeichens nur geringe und aufgrund hoher Schwankungen sehr unzuverlässige Information über den zugrunde liegenden Schriftgrad. Noch deutlicher ist dies beim Bruchstrich der Fall, dessen vertikale Ausdehnung unabhängig von der gerade verwendeten Schriftgröße möglichst niedrig (im Idealfall gleich null) sein sollte, während seine horizontale Ausdehnung vom Kontext, d.h. von Zähler und Nenner, nicht aber direkt von der Schriftgröße abhängt. (Anmerkung: Tatsächlich werden im hier vorgestellten System Minuszeichen und Bruchstrich aus syntaktischer Sicht als ein und dasselbe Schriftzeichen behandelt, was ja morphologisch durchaus gerechtfertigt ist. Die unterschiedliche Bedeutung dieser beiden Symbole schlägt sich konsequenterweise lediglich im semantischen Modell nieder, siehe Kap. 5.3.2).

Entsprechend zur Vorgehensweise bei der Verarbeitung der Positionierungsmerkmale (S. 31 f.) ist auch für die rekursive Analyse der Skalierungsmerkmale eine stufenweise Komponentenfusion notwendig, aus der in diesem Fall eine komponentenspezifische mittlere Schriftgröße g_S hervorgeht:

$$g_S = \sum_{i=1}^{n_S} \varepsilon_{g,i}\, g_i \Bigg/ \sum_{i=1}^{n_S} \varepsilon_{g,i} \tag{4.20}$$

Die binären Gewichtungsfaktoren $\varepsilon_{g,i}$ dienen wiederum zur selektiven Unterdrückung ($\varepsilon_g = 0$) irrelevanter Strukturkomponenten hinsichtlich dieser Schriftgrößenmittelung. Beispielsweise wird bei Exponentialausdrücken auch hier der Exponent unberücksichtigt gelassen, da üblicherweise der Basisausdruck für die Skalierung innerhalb übergeordneter mathematischer Funktionen ausschlaggebend ist. Bei der rekursiven Merkmalbildung sind nunmehr in Gl. 4.19 und 4.20 die Schriftgrößen $g_{i(j)}$ für bereits fusionierte Strukturkomponenten $i(j)$ durch die zugehörigen Fusionswerte g_S zu ersetzen, analog zur Positionierungsanalyse (Gl. 4.17 und 4.18).

Parametrisierung. Zur probabilistischen Bewertung der Symbolanordnung innerhalb des syntaktischen Modells werden die soeben definierten Positionierungs- und Skalierungsmerkmale zunächst für jedes Strukturkomponentenpaar ij zu dreidimensionalen Offset-Vektoren $\vec{o}_{ij}$ zusammengefasst:

$$\vec{o}_{ij} = \begin{Vmatrix} o_x \\ o_y \\ o_g \end{Vmatrix}_{ij} = \begin{Vmatrix} \Delta_{ij}\, \tilde{x} \\ \Delta_{ij}\, \tilde{y} \\ \Delta_{ij}\, \tilde{g} \end{Vmatrix} = \begin{Vmatrix} \Delta_{ij}\, \bar{x} \,/\, \Delta x_S \\ \Delta_{ij}\, \bar{y} \,/\, \Delta y_S \\ \Delta_{ij} \log_5 g \end{Vmatrix} \tag{4.21}$$

Sodann muss unter Berücksichtigung der zu erwartenden Merkmalverteilung ein geeignetes Bewertungsmaß gefunden werden, um die in Gl. 4.12 formal definierten Offsetwahrscheinlichkeiten abschätzen zu können. Hierzu betrachte man die in Abb. 4.6 paarweise dargestellte Verteilung der drei Offsetkomponenten, wie sie sich aus dem verwendeten Handschrift-Trainingskorpus (Kap. 5.2) ergeben. Die angegebenen Semuntypen PROD, POW, SUM und FRAC (für Multiplikation, Exponentiation, Addition und Bruch) stellen mathematische Operatoren dar, die bei der Handschriftanalyse häufig miteinander konkurrieren, so dass eine gute Trennbarkeit der zugehörigen Punktwolken wünschenswert ist. Es versteht sich von selbst, dass dies um so entscheidender ist, je weniger syntaktische Zusatzinformation in Form spezieller Schriftzeichen vorhanden ist: Die Unterscheidung der beiden Ausdrücke (x^y) und (xy) hängt ausschließlich von der Offsetmodellierung ab, während (x+y) von (x·y) in erster Linie anhand der Schriftzeicheninformation zu trennen ist. Abb. 4.6 weist

unter diesem Aspekt weitgehend gut trennbare Klassen auf, zu deren Modellierung aus folgenden Gründen für statistisch unabhängige Normalverteilungen entschieden wurde:

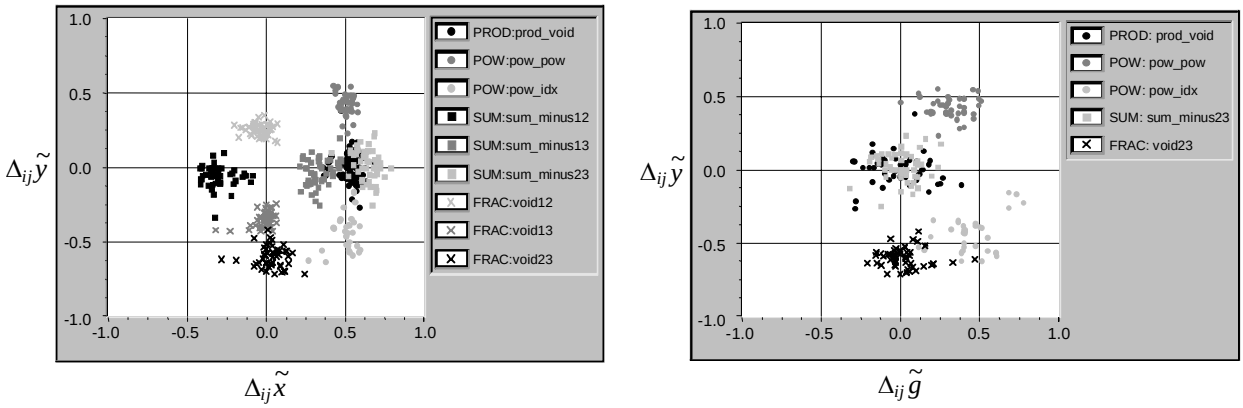


Abb. 4.6: Paarweise Verteilung der Offset-Komponenten für ausgewählte Semuntypen und -werte. Die Nummernsuffixe (12, 13, 23) dienen zur Unterscheidung der entsprechenden Strukturkomponentenpaare bei Semunwerten mit mehr als zwei Strukturkomponenten (hier $n_S = 3$).

- Die Merkmalverteilungen lassen keine deutlichen statistischen Kopplungen zwischen den jeweils drei Offsetkomponenten ($\Delta_{ij}\,\tilde{x}, \Delta_{ij}\,\tilde{y}, \Delta_{ij}\,\tilde{g}$) erkennen. In der Tat handelt es sich bei den syntaktischen Stilmitteln der horizontalen und vertikalen Positionierung sowie der Schriftzeichenskalierung um weitgehend voneinander unabhängige Maßnahmen.

- Eine Berücksichtigung der vorhandenen statistischen Abhängigkeiten in Form der zugehörigen Kovarianzen würde unverhältnismäßig umfangreiche Mengen an Trainingsmaterial pro Schreiber erfordern.

- Unter Ausnutzung der hier verfolgten schritthaltenden Verarbeitung wird ein möglichst echtzeitnahes Erkennungsverfahren angestrebt. Die zur statistisch abhängigen Modellierung erforderliche zusätzliche Rechenleistung wäre daher nur bei entsprechend niedrigem Kosten-Nutzen-Verhältnis, also einer signifikant gesteigerten Gesamterkennungsleistung zu rechtfertigen.

Für jeden unterstützten Semunwert müssen somit – je nach Anzahl n_S der Strukturkomponenten – insgesamt maximal $6 \cdot n_S ! \,/\, [2! \cdot (n_S - 2)!] = 3 \cdot (n_S^2 - n_S)$ Modellparameter (entsprechend je drei Mittelwerten und drei Standardabweichungen pro Strukturkomponentenpaar) aus dem zu erhebenden Datenmaterial abgeschätzt werden.

Bewertungsmaß. Auf Basis der oben angeführten Modellierung der drei Offsetkomponenten mittels statistisch unabhängiger Normalverteilungen sind nun die Offsetwahrscheinlichkeiten α nach Gl. 4.12 zu ermitteln. Gl. 4.9 legt es nahe, als Score-Beiträge einer einzelnen Offset-Emission $\vec{o}_{ij}$ (bezüglich eines Strukturkomponentenpaares *ij*) die negativierten Exponenten der zugehörigen Normalverteilungen heranzuziehen. Diese Maßnahme ist gleichbedeutend mit der Verwendung der quadratischen euklidischen Abstände als Bewertungsmaße, normiert auf die Streuungen der jeweiligen Merkmale. Die *x*-, *y*- und *g*-Anteile des so zu berechnenden Offset-Scores lauten dann:

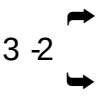
$$[Sc_x]_{ij} = \left(\frac{o_x - \mu_x}{\sigma_x}\right)_{ij}^2; \quad [Sc_y]_{ij} = \left(\frac{o_y - \mu_y}{\sigma_y}\right)_{ij}^2; \quad [Sc_g]_{ij} = \left(\frac{o_g - \mu_g}{\sigma_g}\right)_{ij}^2 \tag{4.22}$$

μ bzw. σ bezeichnen wie üblich Mittelwert bzw. Streuung. Als Gesamt-Offset-Score für die aus dem Nachfolgerknoten A_{nk} emittierte Folge von Offset-Vektoren $\vec{o}$ ergibt sich:

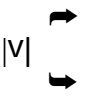
$$-\ln \alpha_{nk} = Sc(\alpha_{nk}) = Sc(A_{nk}) \rightarrow Sc(\vec{o}) = (2n_{o_{x,y}} + n_{o_g})^{-\lambda_f} \cdot \sum_{\forall ij \in r_{nk}} \left[Sc_x + Sc_y + Sc_g\right]_{ij} \tag{4.23}$$

Dabei wird über die einzelnen Offset-Scores zu allen Strukturkomponentenpaaren *ij* summiert, die dem an den Nachfolgerknoten A_{nk} angebundenen Nachfolgersemun r_{nk} (siehe Gl. 4.8) angehören. Das sogenannte *Formelstrukturgewicht* λ_f ist ein freier Parameter, dessen Wert empirisch ermittelt wurde. Seine Bezeichnung trägt der Tatsache Rechnung, dass eine Erhöhung von λ_f mit einer statistischen Aufwertung strukturell komplexerer Formelhypothesen verbunden ist. Für $\lambda_f = 1$ ergäbe sich zunächst die naheliegende Regelung, das Gesamtbewertungsmaß auf die Anzahl $n_{\vec{o}} = (2n_{o_{x,y}} + n_{o_g})$ der Einzelbewertungen, also der insgesamt berücksichtigten Paarungen *ij* von Strukturkomponenten, zu normieren. Dies würde dennoch eine Benachteiligung von Hypothesen mit mehr gegenüber solchen mit weniger Strukturkomponenten bedeuten, da jede weitere Einzelbewertung aufgrund des zusätzlichen Streuungsanteils der zugehörigen Merkmalverteilung den Gesamtscore negativ beeinflusst.

Man betrachte dazu die folgenden beiden Beispiele:

(1) 

(1a) (1b)

(2) 

(2a) (2b)

Die in Beispiel (1) links dargestellte Linienzugfolge[8] führt typischerweise zu den beiden konkurrierenden Hypothesen (1a) und (1b). Diese unterscheiden sich in der Anzahl $n_{\vec{o}}$ der vorzunehmenden Einzelbewertungen, wobei diese im Fall (1a) gleich $n_{\vec{o}} = 3$ ist (je eine Bewertung für o_x, o_y und o_g bei einem Strukturkomponentenpaar) und im Fall (1b) $n_{\vec{o}} = 7$ (drei Strukturkomponenten ergeben je drei Paarungen für o_x und o_y sowie eine Paarung für o_g, da das Minuszeichen von Skalierungsbetrachtungen ausgeschlossen ist – siehe S. 32). Da nun bei Hypothese (1b) die Berücksichtigung aller kombinatorisch möglichen Paarungen aus {3, -, 2} für o_x und o_y eine gewisse Redundanz beinhaltet, wird der Gesamtscore für diese Hypothese in der Regel ungünstiger ausfallen als für Hypothese (1a). (Die redundante Modellierung ist gleichwohl gerechtfertigt, da beispielsweise ein geringer vertikaler Abstand zwischen Minuend und Subtrahend bei etwas nach oben verschobenem Minuszeichen die Abwertung von Hypothese (1b) gegenüber weiteren Alternativen dämpft.)

Beispiel (2) stellt einen – allerdings relativ häufigen – Extremfall dar, bei dem in Fall (2a) $n_{\vec{o}} = 0$ Einzelbewertungen auftreten (nur eine Strukturkomponente vorhanden) und im Fall (2) $n_{\vec{o}} = 6$ (drei Strukturkomponenten, also je drei Paarungen für o_x und o_y sowie keine Paarung für o_g, da die Betragstriche ausscheiden – siehe S. 32). Besonders hier ist eine Aufwertung der strukturell komplexeren Hypothese (2b) erforderlich, damit diese sich gegen Hypothese (2a) durchsetzen kann, bei der überhaupt kein Offset-Score anfällt.

An dieser Stelle wird bereits deutlich, dass die Konkurrenz zwischen jeweils naheliegenden Hypothesen in einem Wechselspiel zwischen der Bewertung von Schriftzeichenanordnung (struktureller Anteil) und Schriftzeichenidentität (symbolischer Anteil) besteht. Der Parameter λ_f wurde deswegen nach datenbasierten Überlegungen so gewählt, dass einerseits die sich wechselweise akkumulierenden Scores beider Anteile einen vergleichbaren Einfluss auf die Gesamtbewertung erhalten und andererseits – als ausschlaggebendes Kriterium – die Gesamterkennungsleistung optimiert wird. Näheres hierzu findet sich in [LIE00], als Resultat ergibt sich $\lambda_f = 1{,}25$.

Der Score-Anteil für das Auftreten der syntaktischen Realisierung Ξ einer gegebenen Semantischen Gliederung *S* (vgl. Gl. 4.3) entspricht nach Gl. 4.12 - 4.15 der Summe aller Einzelscores von Offset-, Element- und Übergangswahrscheinlichkeiten entlang des zugehörigen Pfades durch das gesamte Syntaktische Netzwerk *SN*:

$$Sc(\Xi \mid S) = Sc(\alpha_{01}) + \sum_{n=1}^{N} \left(\sum_{k=1}^{X} Sc(\alpha_{nk}) \right) + \left(\sum_{l=1}^{Y} Sc(\beta_{nl}) \right) + Sc(\gamma_n) + \sum_{az \in Pfad(SM_n)} Sc(\delta_n^{az}) \qquad (4.24)$$

4.5 Morphologische Ebene

Um die aus dem syntaktischen Modell erhaltenen Schriftzeichenanordnungen und Wortketten einer signalnahen Musterbewertung zuführen zu können, müssen deren Elemente, also Schriftzeichen- und Worthypothesen, anhand eines morphologischen Modells als Abfolgen von Linienzügen[8] bzw. Lauteinheiten dargestellt werden.

Da das syntaktische Modell keine linguistisch korrekte Berücksichtigung von sprachlichen Morphemen im Sinne von bedeutungsunterscheidenden Einheiten (etwa dem Wortteil 'un-') beinhaltet [KUN00], während diese bei der handschriftlichen Formelnotation mit Graphemen bzw. Linienzügen gleichzusetzen sind, wird der Begriff der Morphologie hier synonym zum graphemischen bzw. phonetischen Aufbau von Schriftzeichen und Worten angewandt. Er dient also zur Bezeichnung der Wissensbasis, die die Verbindung zwischen syntaktischer und Musterebene herstellt.

Die bedingte Wahrscheinlichkeit $P(\Phi \mid \Xi)$ aus Gl. 4.3 gibt danach die Gesamtwahrscheinlichkeit für die Zuordnung einer Linienzug- bzw. Lautfolge zu einer gegebenen Schriftzeichenanordnung bzw. Wortkette an. Die probabilistische Modellierung dieser Zuordnung erlaubt konsequenterweise eine flexible Verwaltung von Schreib- bzw. Aussprachevarianten, die anhand des ohnehin erhobenen authentischen Trainingsmaterials statistisch gewichtet werden können. Tatsächlich wurden jedoch im Rahmen dieser Arbeit einfache *Lexika* (siehe Anhang A.3 und A.5) eingesetzt, in denen lediglich die Standardaussprachevariante jedes aufgetretenen Wortes als Phonemfolge (*Transkription*) bzw. die Schreibweise jedes Schriftzeichens als Linienzugsequenz hinterlegt ist. Für die Handschrift werden gegebenenfalls pro Schriftzeichen mehrere Schreibvarianten mit unterschiedlicher Linienzuganzahl verwaltet, auf die bei der Musterbewertung dann selektiv zugegriffen werden kann.

$P(\Phi|\Xi)$ entartet damit zu den Werten 1 oder 0, je nachdem ob die betrachtete Aussprache- bzw. Schreibvariante im Lexikon enthalten ist oder nicht. Der zugehörige Anteil am Gesamtklassifikator (Gl. 4.3) wird daher im Folgenden nicht weiter berücksichtigt.

4.6 Musterebene

Die in Kap. 3.2 angedeutete erwartungsgetriebene Top-Down-Decodierung erstreckt sich von der semantischen Abstraktionsebene bis hin zur Musterebene, die hier mit der Repräsentation des vorverarbeiteten Handschrift- bzw. Sprachsignals gleichbedeutend ist. Insofern enthält das Gesamtverfahren einen verbleibenden signalgetriebenen Bottom-Up-Anteil, der in der Vorverarbeitung und Merkmalextraktion der Eingangssignale mittels geeigneter Methoden zur Gewinnung der Beobachtungsfolge *O* besteht. Diese Methoden ergeben sich zum einen aus den Anforderungen der eingesetzten Mustererkennungs- bzw. -bewertungsverfahren, zum anderen sollten sie natürlich zu einer möglichst gleichbleibend guten Trennbarkeit der zu unterscheidenden Musterklassen beitragen.

4.6.1 Handschrift

Vorverarbeitung und Merkmalextraktion. Die Handschriftdaten (Trajektorien der x/y-Stiftposition) werden mit 60 Hz abgetastet, woraufhin eine *längenäquidistante Wiederabtastung* nach [WIN97B], S. 40, die Dynamik des Schreibvorgangs weitgehend eliminiert. Diese Maßnahme ist dadurch begründet, dass eine unterschiedlich stark variierende Schreibgeschwindigkeit zwar einen gewissen Einfluss auf das resultierende Schriftbild haben mag, dieses Schriftbild aber – im Gegensatz zur gesprochenen Sprache – entscheidend für die Lesbarkeit bzw. Klassifizierung ist. Daher wird eine annähernd konstante Dichte von Abtastpunkten entlang der jeweiligen Schreibkurve angestrebt, um eine dementsprechend gleichmäßige Verteilung der informationstragenden Merkmale innerhalb der Schriftprobe zu gewährleisten. Zum Ausgleich von Quantisierungsfehlern sowie von Störinformation im Originalsignal (z.B. Zittern durch Hand und/oder Eingabemedium) findet anschließend eine Tiefpassfilterung statt (siehe auch [GRO97], S.10).

Hinsichtlich der Klassifikation einzelner Schriftzeichen, auf die sich die hier angesiedelte Musterbewertung im Rahmen des einstufigen Ansatzes beschränkt, soll ferner der Einfluss der Schriftzeichenskalierung unterdrückt werden, die ja bereits als wichtiges syntaktisches Merkmal innerhalb der Offsetkalkulation (Kap. 4.4.4, S. 32 ff.) im Detail berücksichtigt wird. Aus diesem Grund findet stets eine zentrierte Normierung der Abtastkoordinaten auf die Abmessungen des umgebenden Rechtecks zum jeweiligen Schriftzeichen statt. Bei der Vorverarbeitung des Trainingsmaterials ist die hierfür erforderliche Zuordnung bestimmter Linienzüge[8] zu einem Schriftzeichen durch vorherige Annotation bekannt (siehe Kap. 5.3.2), während sie bei der späteren Decodierung unbekannter Eingabedaten im Sinne der Bewertung einer aktuell vorgegebenen Schriftzeichenhypothese (erwartungsgetrieben) vorliegt (Kap. 4.7).

Neben den normierten Abtastkoordinaten werden zusätzlich deren erste und zweite (diskrete) Ableitung als Merkmale verwendet, um die darin enthaltene Zusatzinformation über den Verlauf der Schreibrichtung und der lokalen Krümmung der Schreibkurve nutzen zu können [MAN95]. Diese Maßnahme verbessert insbesondere die Trennung morphologisch verwandter Schriftzeichen wie

z.B. '(' und 'C'. Aus Stetigkeitsgründen und zu Normierungszwecken (s.u.) dienen dazu je zwei Komponenten bestehend aus Sinus und Kosinus des Steigungs- bzw. Krümmungswinkels ([GRO97], S. 12). Da im Allgemeinen mehrere Linienzüge einem einzelnen Schriftzeichen zuzuordnen sind, wird ein weiteres, binäres Merkmal für den Stiftstatus hinzugenommen, welches die Zustände 'aufgesetzt' (+½) und 'angehoben' (-½) unterscheidet. Die so ermittelten insgesamt sieben Merkmalvektorkomponenten sind bezüglich der späteren Abstandsberechnung (Gl. 4.27) laut ihrer obigen Definition automatisch zueinander gleich gewichtet, indem sie den gemeinsamen Wertebereich [-1,1] etwa gleichmäßig abdecken. Der resultierende Merkmalvektor für den *n*-ten Abtastpunkt lautet dann:

$$\vec{O}_n^T = \left(\frac{\hat{x}_n - \bar{x}}{\Delta x}, \frac{\hat{y}_n - \bar{y}}{\Delta y}, \sin\delta_n^{(1)}, \cos\delta_n^{(1)}, \sin\delta_n^{(2)}, \cos\delta_n^{(2)}, p_n \right) \tag{4.25}$$

$(\hat{x}_n, \hat{y}_n)$ bezeichnen die Abtastkoordinaten nach Wiederabtastung und Tiefpassfilterung, $(\Delta x, \Delta y)$ Breite und Höhe des umgebenden Rechtecks und $(\bar{x}, \bar{y})$ seinen arithmetischen Mittelpunkt (nicht zu verwechseln mit den symbolspezifischen Mittelpunkten aus Gl. 4.16 und 4.17). Die momentanen Schreibrichtungs- bzw. Krümmungswinkel $\delta_n^{(1)}, \delta_n^{(2)}$ werden durch Differenzbildung aus vorhergehendem (*n*-1) und nachfolgendem (*n*+1) Abtastpunkt berechnet, p_n = ±½ (für *p*en oder *p*ressure) gibt den aktuellen Stiftstatus an.

Musterbewertung. Die Musterbewertung auf Schriftzeichenbasis erfolgt mittels eines Standardalgorithmus, der durch *Dynamic Time Warping* (DTW) den kleinstmöglichen Abstand zwischen Beobachtungsfolge *O* und *Referenzmuster R* ermittelt. Zur Implementierung wurde auf die frei verfügbaren Standardbibliotheken *Recognition Primitives Library* (RPL) und *Signal Processing Library* (SPL) von INTEL zurückgegriffen [INT98]. Dabei findet über eine Verzerrungsfunktion eine nichtlineare Zuordnung zwischen je zwei Abtastpunkten von Beobachtungs- und Referenzmuster statt, deren euklidische Abstände im 7-dimensionalen Merkmalraum (Gl. 4.25) schrittweise aufsummiert werden. Der minimale Gesamtabstand ergibt sich durch Berechnung der akkumulierten Distanzmatrix ***D*** als deren letztes Element *D*(*N*, *M*), und zwar nach folgender Vorschrift [RUS97]:

$$D(1,1) = d(1,1); \quad D(n,m) = \min \begin{cases} D(n-1,m) & + & d(n,m) \\ D(n-1,m-1) & + & 2d(n,m), \\ D(n,m-1) & + & d(n,m) \end{cases} \quad \begin{matrix} 1 \le m \le M \\ 1 \le n \le N \end{matrix} \tag{4.26}$$

$$D_{DTW}(O,R) = D(N,M) \tag{4.27}$$

$d(n,m) = |\vec{O}_n - \vec{R}_m|$ gibt den euklidischen Abstand zwischen dem *n*-ten Merkmalvektor des Beobachtungsmusters und dem *m*-ten Merkmalvektor des Referenzmusters an. Das in Gl. 4.26 zugrunde gelegte Wegediagramm schränkt die Verzerrungsfunktion auf horizontale, vertikale oder diagonale Teilstücke der Schrittweite 1 ein. Mit anderen Worten muss für den jeweils nächsten aufzusummierenden Teilabstand im Beobachtungsmuster und/oder im Referenzmuster zum nächstfolgenden Merkmalvektor übergegangen werden. Wie üblich gilt als weitere Randbedingung, dass die Abstandsberechnung beim ersten (1, 1) bzw. letzten (*M*, *N*) Merkmalvektorpaar beginnt bzw. endet.

Die zu einer handgeschriebenen Formel korrespondierende Beobachtungsfolge *O*, bestehend aus *J* Merkmalvektoren gemäß Gl. 4.25, weist – im Gegensatz zur Spracheingabe – bereits natürliche Segmentierungsgrenzen auf, die sinnvollerweise als Zusatzinformation bei der Schriftzeichenbewertung berücksichtigt werden. Es handelt sich dabei um alle Übergänge von einem zum jeweils zeitlich nachfolgenden *Linienzug* $L_i \rightarrow L_{i+1}$, die durch das Anheben und darauffolgende Wiederaufsetzen des Schreibstiftes (Stiftstatus-Merkmal p = -½ bzw. ½) gekennzeichnet sind:

$$O = \left(\vec{O}_1, \cdots, \vec{O}_j, \cdots, \vec{O}_J\right) = \left(L_1, \ldots, L_i, \ldots, L_I\right); \quad L_i = \left(\vec{O}_{j_1(i)}, \cdots, \vec{O}_{j_2(i)}\right) \tag{4.28}$$

Jeder einzelne Linienzug L_i besteht demnach aus einer Merkmalvektorfolge, die beim Abtastpunkt $j_1(i)$ beginnt (Stift aufgesetzt) und beim Abtastpunkt $j_2(i)$ endet (Stift angehoben). Beim Niederschreiben mathematischer Formeln wird im Allgemeinen eine weitgehend durchgängige Blockschriftnotation verwendet, so dass die im Laufe der Symbolbewertung vorzunehmende Segmentierung nach Einzelschriftzeichen auf Linienzugebene betrieben werden kann. Dies bedeutet, dass jedem zu bewertenden Schriftzeichen eine bestimmte Linienzuggruppe, bestehend aus einer Anzahl Δ_i von Linienzügen, eindeutig zuzuordnen ist. Überdies handelt es sich dabei grundsätzlich um eine zeitlich zusammenhängende Abfolge von Linienzügen: Die dem zugrunde liegende Forderung, dass jedes Schriftzeichen vollständig niedergeschrieben wird, bevor der Schreiber das nächste Zeichen einzugeben beginnt, haben alle in der Literatur zu findenden vergleichbaren Systeme gemeinsam [GRB95]. Um in bestimmten Fällen dennoch eine zeitliche Fragmentierung eines Einzelschriftzeichens zuzulassen, enthält das hier vorgestellte System einen Sondermechanismus, der auf S. 52 vorgestellt wird.

Obige Bemerkungen treffen ebenso auf die durchaus vorhandenen Kursivschriftanteile zu, die hauptsächlich bei der Notation von Funktionsnamen (z.B. 'sin' oder 'ln') zu beobachten sind. In Anbetracht der relativ überschaubaren Zahl der hierbei zu unterscheidenden Fälle werden solche feststehenden Buchstabenkombinationen bereits auf der Schriftzeichenebene als Ganzes zu speziellen Sonderzeichen zusammengefasst, die dementsprechend je nach Schreibweise aus einem oder mehreren aufeinanderfolgenden Linienzügen aufgebaut sind (siehe Kap. 5.3.2). Eine Modellierung solcher Schriftzüge unter Berücksichtigung ihrer inneren Struktur wie etwa in [WIN97A] (im Beispiel als Buchstabenfolge 's' 'i' 'n' oder 'l' 'n') hätte einerseits eine – hier nicht nutzbringende – Segmentierung auf Merkmalvektorebene und andererseits eine nachträgliche Zuordnung dieser Folgen zur entsprechenden mathematischen Funktion erfordert. Ungeachtet dessen hat der Benutzer auf diese Weise die freie Wahl, in solchen Fällen eine Blockschrift-, Kursiv- oder auch gemischte Notationsform anzuwenden: Seine jeweiligen Schreibgewohnheiten spiegeln sich letztendlich in den zugehörigen Referenzmustern der Musterbewertungsschicht (siehe Kap. 5.3.2 und 5.7.2) wider, so dass also etwa auch die blockschriftliche Notation 's' 'i' 'n' (sollte sie nicht zufällig – je nach Kontext – als Produkt der Variablen *s, i,* und *n* gedacht sein) dem Sonderzeichen 'sin' zugeordnet wird.

Definieren wir also die Linienzuggruppe der Länge $\delta_i > 0$ als Folge der δ_i nächsten Linienzüge ab Linienzugposition *i*:

$$L_i^{(\delta_i)} = \left(L_i, \ldots, L_{i+\delta_i-1}\right), \quad \delta_i > 0 \tag{4.29}$$

Dann wird nach folgender Vorschrift der Score für die Bewertung eines vorgegebenen Elements *e* an der aktuellen Linienzugposition *i* berechnet:

$$Sc_i(e) = \min_{1 \le \delta_i \le \delta_i^{\max}(e)} Sc_i^{(\delta_i)}(e) = \min_{1 \le \delta_i \le \delta_i^{\max}(e)} \min_{R \in \{R_e^{(\delta_i)}\}} D_{DTW}\{L_i^{(\delta_i)}, R\}$$
$$\Delta_i(e) = \operatorname*{argmin}_{\delta_i} Sc_i^{(\delta_i)}(e) \tag{4.30}$$

Es werden also nacheinander alle in Frage kommenden Linienzuggruppen $L_i^{(\delta_i)}, 1 \le \delta_i \le \delta_i^{\max}(e)$, hinsichtlich ihrer Übereinstimmung mit der gegebenen Elementklasse *e* überprüft. Dies geschieht durch Ermittlung des jeweils geringsten Abstands (Gl. 4.27) im Vergleich mit allen zugehörigen Referenzmustern $R_e^{(\delta_i)}$ der DTW-Wissensbasis (siehe Kap. 5.7.2) bei übereinstimmender Linienzuganzahl. Die maximale in Betracht zu ziehende Linienzuganzahl $\delta_i^{\max}(e)$ ist durch die vorhandenen Referenzmuster vorgegeben und liegt im Normalfall (bei Standardschriftzeichen) im Bereich von 1 bis 5. Die Musterbewertung liefert sodann den insgesamt besten Score sowie die zugehörige (und damit abgearbeitete) Linienzuganzahl Δ_i als Bewertungsergebnis zurück.

Damit lautet der Gesamtscore für die Übereinstimmung der Beobachtungsfolge *O* mit dem syntaktischen Gesamtausdruck Ξ :

$$Sc(O|\Xi) = \sum_{h=1}^{H} Sc_{i(h)}(e_h);$$
$$i(h) = 1 + \sum_{n=1}^{h-1} \Delta_i(e_n), \quad \Delta_i(e_H) = I - i(H) \tag{4.31}$$

Es erfolgt eine Summation über die Einzelmusterbewertungen nach Gl. 4.30 zu allen Elementen aus Ξ gemäß ihrer zeitlichen Abfolge (e_1, …, e_H), so dass sich nebenläufig als *Alignment* die Zuordnung der Linienzugendpositionen $i(h)$ zu den einzelnen Elementen e_h des Hypothesen-Gesamtausdrucks ergibt. Als abschließende Randbedingung gilt, dass für das zeitlich zuletzt emittierte Element e_H (zuletzt geschriebenes Schriftzeichen gemäß Hypothese) die noch verbleibenden Linienzüge der Beobachtungsfolge verarbeitet werden müssen.

4.6.2 Sprache

Vorverarbeitung und Merkmalextraktion. Hier wurde auf allgemein bewährte Verfahren zurückgegriffen [RUS94], indem das durch Abtastung mit 16 kHz und 16-Bit-Linearquantisierung digitalisierte Sprachsignal in 10ms-Kurzzeit-*Lautheitsspektren* überführt und durch gehörgerechte Frequenzabtastung nach der *Mel*-Skala [ZWI99] zunächst auf 20 spektrale Parameter pro Zeitschritt reduziert wird. Zusammen mit deren ersten und zweiten Ableitungen und einigen zusätzlichen Kenngrößen erhält man daraufhin 66-dimensionale Merkmalvektoren, die durch *Lineare Diskriminanzanalyse* (LDA) schließlich auf 30 Komponenten reduziert werden [RUS98]. Nähere Einzelheiten zur Vorgehensweise beschreibt [PFA00] auf Seite 36 f.

Musterbewertung. Im Rahmen der Anpassung des einstufigen Decodierungsverfahrens an die Domäne mathematischer Formeln wurden als Musterbewertungsschicht für die sprachliche Eingabe neue phonembasierte *Hidden-Markov-Modelle* (HMM) der Sprachverarbeitungsgruppe um G. RUSKE am Lehrstuhl für Mensch-Maschine-Kommunikation der TU München in das Gesamtverfahren integriert [LIE98]. Anhand eines phonetischen Lexikons (Kap. 4.5) werden für jedes mögliche Element *e* die ihm zugeordneten Phonemmodelle (kontextunabhängige Monophon-HMMs mit jeweils 3 bis 4 Zuständen) zu Wortmodellen verkettet (vgl. [PFA00], S. 125).

Die Beobachtungsfolge *O* (vorverarbeitetes Sprachsignal) besteht wiederum aus einer zeitlichen Abfolge von Merkmalvektoren $\vec{O}_i$:

$$O = \left(\vec{O}_1, \cdots, \vec{O}_i, \cdots, \vec{O}_I\right) \tag{4.32}$$

Der Laufindex *i*, der gleichermaßen bei der Handschriftverarbeitung für die Verwaltung der Linienzüge (*nicht* der Merkmalvektoren) Verwendung findet (Gl. 4.28), soll hier Folgendes verdeutlichen: Das verallgemeinerte Suchverfahren (Kap. 4.7) betreibt bei der inkrementellen Abarbeitung der Benutzereingabe eine schritthaltende Segmentierung (Zuordnung geeigneter Signalabschnitte zu den einzelnen Elementen), die bei Spracheingabe – in Abwesenheit einer natürlichen Vorsegmentierung – direkt auf der Skala der Beobachtungsfolge *O* beruht. Eröffnung, schrittweise Bewertung und Beendigung von Elementhypothesen auf der Musterbewertungsebene (siehe Struktogramm, Abb. 4.7, Phase 4) als signalbezogene Teilschritte des globalen Suchzyklus erfolgen damit je nach Eingabemodalität auf dem Raster der Zeitfensterung (Sprache) bzw. der Linienzugstruktur (Handschrift).

Analog zu den in Gl. 4.29 definierten Linienzuggruppen fassen wir nun Teilgruppen aufeinanderfolgender Merkmalvektoren (Anzahl δ_i, ab Position *i*) wie folgt zusammen:

$$O_i^{(\delta_i)} = \left(\vec{O}_i, \ldots, \vec{O}_{i+\delta_i-1}\right), \quad \delta_i > 0 \tag{4.33}$$

Als zugehöriger akkumulierter HMM-Score (Bewertung der δ_i nächsten Merkmalvektoren ab Position *i* bezüglich Element *e*) sei gegeben:

$$Sc_i^{(\delta_i)}(e) = Sc_{HMM}\left(O_i^{(\delta_i)}, e\right) \tag{4.34}$$

Der Gesamtscore für die Musterbewertung der Beobachtungsfolge *O* bei gegebenem syntaktischem Gesamtausdruck Ξ ergibt sich sodann durch Minimierung der Summe aus allen Einzelscores (zu den nacheinander emittierten Elementen $e_1, \ldots, e_H$) unter Variation der Elementgrenzen $i(h)$.

$$\begin{gathered} Sc(O|\Xi) = \min_{i(1)\ldots i(H)} \sum_{h=1}^{H} Sc_{i(h)}^{(\delta_i)}(e_h); \\ \delta_i = i(h+1) - i(h) \end{gathered} \tag{4.35}$$

4.7 Suchverfahren

Die Auffindung der wahrscheinlichsten Semantischen Gliederung zu einer gegebenen Handschrift- oder Spracheingabe erfordert die Bewältigung eines Suchproblems, bei dem aus einem Wald möglicher Bäume $\{S\}$ der beste Baum S_E gefunden werden muss. Dies geschieht unter Variation von S durch Optimierung des Durchlaufpfades im zugehörigen Syntaktischen Netzwerk *SN* hinsichtlich der Beobachtungsfolge O. Hierfür wird eine geeignete Kostenfunktion K herangezogen, die es im Laufe des Suchvorgangs zu minimieren gilt (vgl. Gl. 4.3):

$$\boxed{\begin{aligned} S_E &= \underset{S}{\operatorname{argmin}} \min_{\Xi} K(O,\Xi,S) \\ K(O,\Xi,S) &= Sc(O|\Xi) + Sc(\Xi|S) + Sc(S) \end{aligned}} \tag{4.36}$$

Das hierfür eingesetzte Suchverfahren beinhaltet im Wesentlichen drei unterschiedliche Mechanismen zur Begrenzung des benötigten Speicher- und Rechenaufwands, der anderenfalls weit oberhalb der durch derzeit verfügbare Ressourcen gesetzten Beschränkungen ausfallen würde:

- Beschränkung des Suchraums auf syntaktisch-semantisch konsistente Hypothesen (top-down)
- *Strahlsuche* (Pruning) als suboptimales Suchverfahren
- Wiederverwendung bereits aufgetretener Teilhypothesen durch *Rekombination* ([STA97B], S. 74 ff.)

Die Organisation des Suchvorgangs wurde soweit möglich in ihren Grundzügen von [STA97B], Kap. 4 und 5 übernommen. Dort findet sich auch eine ausführliche Darstellung der prinzipiellen Suchstrategie als aktiver EARLEY-*Chartparser* mit einer speichereffizienten dynamischen Suchraumorganisation mittels sog. *Itemlisten* für jedes Zeitfenster der Beobachtungsfolge. Der grundlegende Ablauf des hier vorgestellten erweiterten Suchverfahrens einschließlich der Maßnahmen zur Ausdehnung der syntaktisch-semantischen Modellierung auf die Handschriftverarbeitung ist in Abb. 4.7 als Struktogramm vereinfachend wiedergegeben.

Innerhalb des Hauptzyklus, der zur inkrementellen Abarbeitung der Beobachtungsfolge in eine Schleife über alle Merkmalvektoren bzw. Linienzüge ($i = 0 \ldots I$) gemäß ihrer zeitlichen Abfolge eingebettet ist, werden jeweils nacheinander vier Hauptphasen durchlaufen, in denen die verschiedenen probabilistischen Bewertungen zusammen mit den zugehörigen Maßnahmen zur Hypothesenbildung vorgenommen werden. Diese Maßnahmen umfassen in erster Linie die Vorhersage möglicher Nachfolgerscharen (Semantik), Übergänge sowie Element- und Offset-Emissionen (Syntax) und Musterverarbeitung. Das verfolgte *First-Last*-Abarbeitungsschema ist eine entscheidende Voraussetzung für eine echtzeitnahe Funktionsweise des Gesamtsystems: Der Suchvorgang startet unmittelbar nachdem der Benutzer mit der Eingabe beginnt, Schreib- und Sprechzeiten können damit systemseitig sinnvoll genutzt werden, und bei der Handschrifteingabe sind fortlaufende Korrekturen erlaubt, die in den meisten Fällen – wenn die betreffenden Segmente noch nicht in den Suchraum eingespeist wurden – ohne negative Auswirkung auf den weiteren Verlauf der Suche bleiben (siehe auch Kap. 7.1.1).

In jedem Hauptsuchschritt *i* werden dementsprechend ausschließlich Hypothesen weiterbearbeitet, die aufgrund der bisherigen Suche die Verarbeitungs-Endposition *i* aufweisen. Von diesen werden zuvor alle diejenigen Hypothesen verworfen, deren Gesamtscore oberhalb der aus bestem aktuellem Hypothesen-Gesamtscore zuzüglich Pruningdistanz ermittelten Pruningschwelle liegt. Diese Form der Strahlsuche bewirkt durch die ihr innewohnende Begrenzung der Suchraum-Bewertungsdynamik eine Selbstfokussierung des Suchvorgangs: Dieser wird deutlich effizienter, falls gerade nur eine oder wenige Hypothesen eine günstige Bewertung erzielen, hingegen gründlicher (aber auch rechenintensiver), wenn eine höhere Mehrdeutigkeit besteht, so dass dann entsprechend mehr Hypothesen im Suchstrahl enthalten bleiben, siehe [STA97B], S. 93 f.

Alle aufgrund einer der oben genannten Maßnahmen (Regelanwendungen im Sinne der Grammatik bzw. Musterbewertungen) neu erzeugten Hypothesen werden, gesondert nach Endposition *i*, in einer Warteschlange (*FIFO*-Speicher) verwaltet, so dass die globale Suchstrategie einer *Breitensuche* entspricht. Die oben erwähnte Rekombination (paarweise Vereinigung äquivalenter Teilhypothesen) erfordert auf diese Weise den geringsten Rechenaufwand, da jeder Rekombinationskandidat so lange wie möglich unbearbeitet bleibt und im Falle einer tatsächlichen Rekombination möglichst selten (aufgrund einer besseren Gesamtbewertung des Rekombinationspartners) erneut bearbeitet werden muss.

Ist nach der Rückkehr aus dem Endknoten eines gerade aktuellen Syntaktischen Moduls (Abb. 4.7, Phase 2a) mit anschließender Offset-Emission das Syntaktische Netzwerk (und damit das Wurzel-SM) der zugehörigen Hypothese vollständig abgearbeitet, so werden durch Vergleich mit der bisher besten vollständigen Hypothese der beste Gesamthypothesen-Endscore sowie der beste Wurzel-Semuntyp aktualisiert. Die Aufdeckung der Semantischen Gliederung mit dem insgesamt niedrigsten Gesamtscore (und damit des Decodierungsergebnisses) erfolgt nach Abarbeitung der gesamten Beobachtungsfolge durch *Trace-Back*, indem ausgehend vom Wurzelsemun (Typ und Wert) rekursiv alle zugehörigen Nachfolgersemuntypen und -werte aus dem abgearbeiteten Hypothesenbestand rekonstruiert werden.

In Phase 4 zeigt sich, wie in Kap. 4.6.1 (S. 40) angedeutet, einer der wesentlichen Vorzüge des einstufig-erwartungsgetriebenen Ansatzes: Insbesondere bei der Handschriftverarbeitung werden keinerlei isolierte bzw. externe Segmentierungsvorschriften mehr benötigt, wie sie etwa in [WIN97B], Kap. 3-5, sogar einen beträchtlichen Anteil des Gesamtverfahrens ausmachen. Die beste Gesamtsegmentierung resultiert als Nebenergebnis im Verlauf der Einzelschriftzeichenbewertung, da nach Vorgabe ausschließlich syntaktisch-semantisch konsistenter Schriftzeichenkombinationen die jeweils optimale Anzahl an Linienzügen konsumiert wird, bis die Gesamthypothese mit der besten Endbewertung vollständig vorliegt. Die Ausnutzung von (formelbezogenem) Kontextwissen aus den höheren Abstraktionsebenen Syntax und Semantik ersetzt also auf elegante Weise die sonst allein aufgrund regelbasierter, geometrischer Entscheidungskriterien zu treffende Vorhersage möglicher Segmentierungshypothesen (etwa mittels eines umfangreichen Symbolhypothesennetzes, siehe [WIN97B], S. 27).

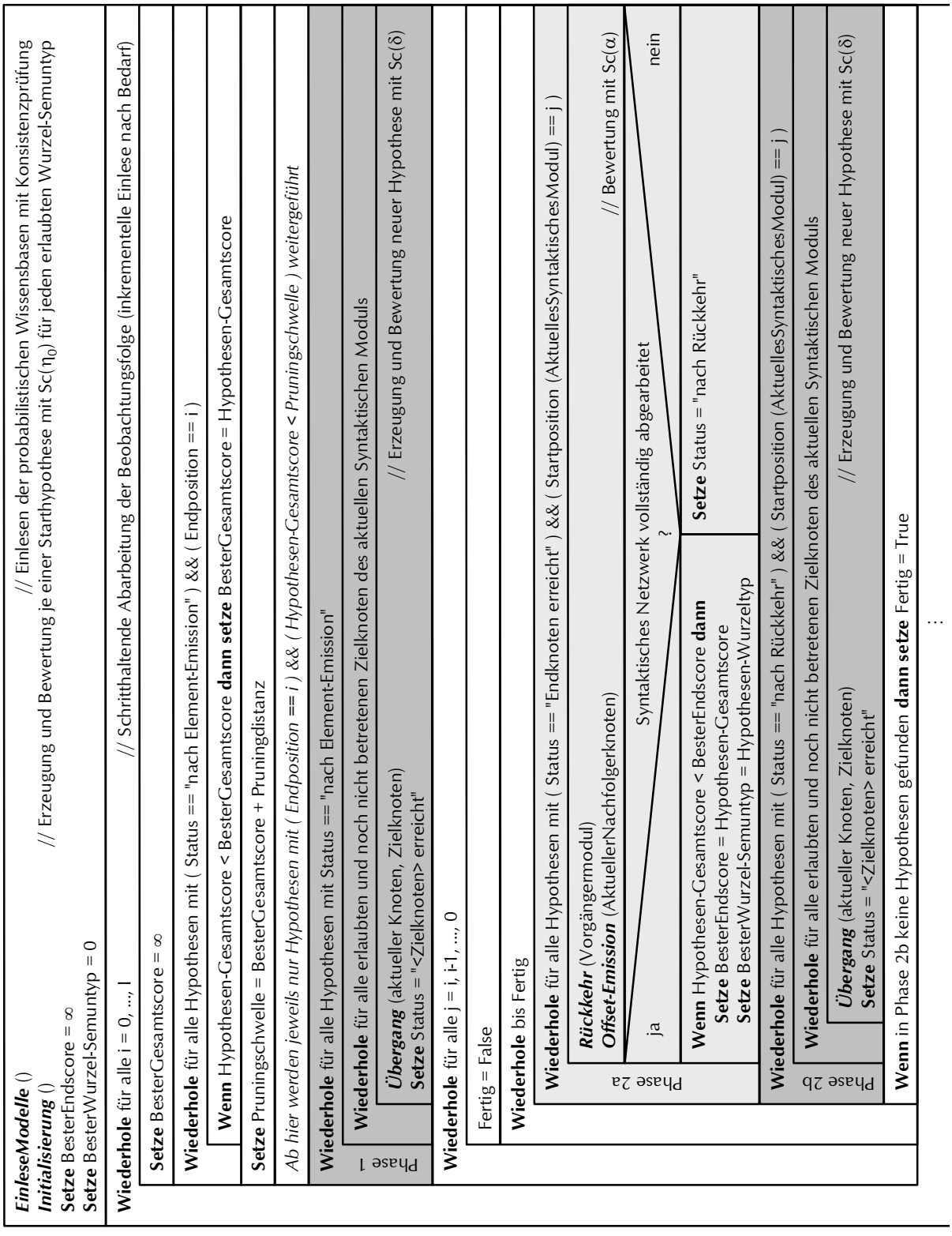


Abb. 4.7: Struktogramm zum Suchverfahren für die einstufig-probabilistische semantische Decodierung bei schritthaltender Verarbeitung, vereinfachte Darstellung.

...
Fertig = False
Wiederhole bis Fertig
Wiederhole für alle Hypothesen mit (Status == "Nachfolgerknoten erreicht")
Phase 3a
Nachfolgerschar definiert ?
ja
nein
BetreteNachfolgermodul (AktuellerNachfolgerknoten)
Setze Status = "Startknoten erreicht"
Wiederhole für alle erlaubten Nachfolgerscharen r
// Erzeugung und Bewertung neuer Hypothese mit Sc(η)
SetzeNachfolgerschar (r)
BetreteNachfolgermodul (AktuellerNachfolgerknoten)
Setze Status = "Startknoten erreicht"
Wiederhole für alle Hypothesen mit Status == "Startknoten erreicht"
Phase 3b
Wiederhole für alle erlaubten und noch nicht betretenen Zielknoten des aktuellen Syntaktischen Moduls
Übergang (aktueller Knoten, Zielknoten)
// Erzeugung und Bewertung neuer Hypothese mit Sc(δ)
Setze Status = "<Zielknoten> erreicht"
Wenn in Phase 3b keine Hypothesen gefunden dann setze Fertig = True
Wenn i == l dann Ende
Wiederhole für alle Hypothesen mit (Status == "Elementknoten erreicht")
Phase 4a
Ist aktueller Knoten ein B-Knoten ?
ja
nein
Wiederhole für alle erlaubten Semunwerte v und Elemente e+
// Erzeugung und Bewertung neuer Hyp. mit Sc(ε) u. Sc(β)
Element-Emission (v, e+, i)
// Startposition i
Wiederhole für alle erlaubten Elemente e-
// Erzeugung und Bewertung neuer Hypothese mit Sc(γ)
Element-Emission (e-, i)
// Startposition i
Wiederhole für alle laufenden Element-Emissionen e
Ph. 4b
Element-Schritt (e, i)
// Bearbeite nächsten Merkmalvektor/Linienzug (O/L)i in der Musterbewertungsschicht
Wiederhole für alle Semuntypen t und -werte v
Phase 4c
Wiederhole für alle Startpositionen j = 0, 1, ..., i-1
Elementscore = Element-Abruf (t, v, j)
// Abfrage der Musterbewertung
Wenn Elementscore != 0 dann
Element-Ende (t, v, j, Elementscore)
// Erzeugung und Bewertung neuer Hypothese mit Elementscore
Setze Status = "nach Emission"
Setze Endposition = i+1
// Einspeisung in den nächsten Suchzyklus
BesteSemantischeGliederung = Trace-Back (BesterWurzel-Semuntyp)
// Aufdecken der besten Semantischen Gliederung durch Rückverfolgung

5

Training

5.1 Semantisches Inventar

Bevor ein datenbasiertes Training der beteiligten Wissensbasen erfolgen kann, müssen die vorgesehenen mathematischen Formelkomponenten durch Angabe entsprechender Semuntypen, zugehöriger Semunwerte sowie deren jeweiliger Wertigkeiten (Nachfolgeranzahl) manuell festgelegt werden. Das resultierende *Referenzmodell* dient dann als Ausgangsbasis für die spätere Annotation des Trainingsmaterials, wird aber in der Regel zur Beseitigung von Inkonsistenzen (mehrfache bzw. fehlende Berücksichtigung möglicher Formelbestandteile) einige Male überarbeitet, bis eine endgültige Fassung vorliegt. Dies ist normalerweise dann der Fall, wenn der gesamte zu annotierende Datenbestand widerspruchsfrei bearbeitet werden konnte.

Das für die vorliegende Domäne erstellte Referenzmodell enthält in seiner Endfassung etwa 30 Typen und 180 Werte und ist in Anh. A.1 vollständig abgedruckt. Die darin integrierten, zur Einhaltung bestehender Operator-Rangordnungen und zur Einschränkung der zu modellierenden Nachfolgerscharen eingeführten Meta-Semuntypen werden in Kap. 5.4 erläutert.

5.2 Datenerhebung

Um eine möglichst große Vielfalt mathematischer Teilausdrücke in unterschiedlichem Kontext und Umfang, auch in geschachtelter Form, zur Verfügung zu haben, wurden ausschließlich vollständige, realistische Formeln zur Datenerhebung herangezogen. Die Datenerhebung fand in einer gewöhnlichen Büroumgebung statt und umfasste je zwei mit einem Standardmikrofon[13] aufgezeichnete Datensätze der 17 Referenzformeln laut Anh. A.4 von insgesamt 10 Sprechern sowie je drei am LCD-Tablett[14] eingegebene Fassungen der 16 Handschrift-Referenzformeln laut Anh. A.2 zuzüglich je

[13] SHURE SM10A Kopfbügelmikrophon

[14] WACOM Digitalisiertablett PL-400, siehe Abb. 5.1.

drei Niederschriften des gesamten Schriftzeicheninventars von insgesamt 18 Schreibern. Da die Spracherfassung in einer früheren Projektphase bei einer etwas eingeschränkten Formeldomäne durchgeführt wurde, erfolgte hierfür noch eine nachträgliche Erweiterung zur Abdeckung des gesamten gewünschten Funktionsumfangs. Das resultierende Inventar gesprochener Worte (incl. phonetischer Darstellung nach [WEL02]) bzw. handgeschriebener Symbole findet sich in Anh. A.5 bzw. A.3. Aus dem erhobenen Trainingsmaterial erhält man für die ca. 300 Worteinträge ca. 15000 Beispiele bei sprecherunabhängiger Betrachtung sowie zu den ca. 130 Handschriftsymbolen ca. 5000 Referenzen (Schriftzeichen und Offsets) pro Schreiber – zur Frage der Schreiberabhängigkeit siehe Kap. 5.5.3 und 5.7.2.

Abb. 5.1: Interface zur Handschrifterfassung (WACOM Digitalisiertablett PL-400).

5.3 Annotation

Um die verschiedenen Wissensbasen mit geeigneten Daten anhand des erhobenen Trainingsmaterials versorgen zu können, muss dieses von Expertenhand einer umfassenden Aufbereitung, bestehend aus Transkription, Segmentierung und Annotation, unterzogen werden. Diese nicht nur aufwendigen, sondern leider auch fehleranfälligen Maßnahmen sind für die spätere Zuverlässigkeit des Decodierungsverfahrens von entscheidender Bedeutung, so dass im Rahmen dieser Arbeit durch den Einsatz spezieller Werkzeuge versucht wurde, die notwendigen Manipulationen möglichst komfortabel, transparent und verifizierbar zu gestalten.

Abb. 5.2 skizziert den grundsätzlichen Ablauf als iterativen Prozess, in dessen Zentrum die Datenaufbereitung mithilfe der graphikorientierten Bedienumgebungen *SPCHTOOL* und *STKTOOL* steht. Einzelheiten zu deren Aufbau und Funktionsweise werden in den beiden nachfolgenden Absätzen 5.3.1 und 5.3.2 geschildert.

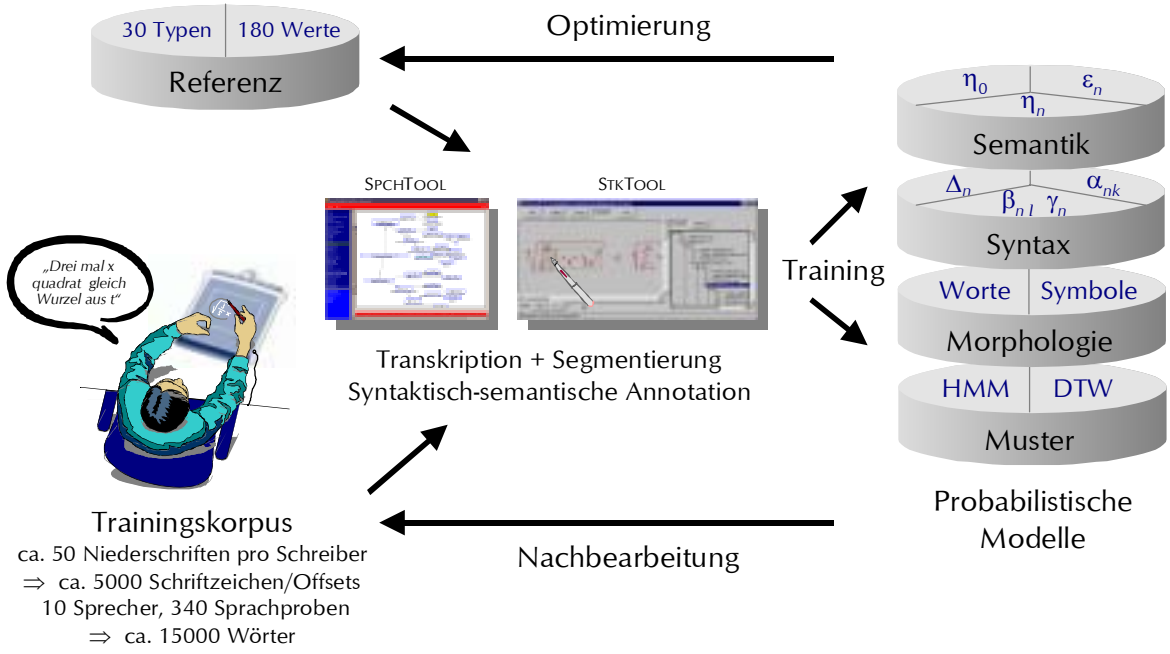


Abb. 5.2: Datenerhebung, Annotation und Training im Überblick.

5.3.1 Sprache

Das erhobene Sprachmaterial liegt zunächst ausschließlich als Sammlung abgespeicherter Audio-Dateien vor. Es erfolgt dann eine manuelle Verschriftung (Transkription) aller Spracheingaben, indem die zugehörigen Wortketten in einfachen Textdateien abgelegt werden. Dabei werden jeweils neu auftretende Worte automatisch in ein ebenfalls textbasiertes Wortlexikon aufgenommen, das später wiederum manuell durch passende Phonemfolgen (als Standard-Aussprachevarianten) nach [WEL02] ergänzt wird (siehe Kap. 5.6.1). Für häufig auftretende Teilphrasen besteht die Möglichkeit, diese zu Pseudoworten zusammenzufassen (z.B. 'ist_gleich'), wodurch in vielen Fällen auch die im nächsten Absatz beschriebene syntaktisch-semantische Zuordnung vereinfacht wird.

Zur syntaktisch-semantischen Annotation werden die so erzeugten Wortketten in ein entsprechendes Eingabefeld von SPCHTOOL übernommen (siehe Abb. 5.3 unten), woraufhin auf der Arbeitsfläche für jedes enthaltene Element (Wort) ein Kasten erscheint, der mit Elementbezeichner und -position (Ordnungszahl gemäß Sprechreihenfolge) versehen ist. Ausgehend von dieser wortnahen Darstellung entsteht durch Verschieben der Kästen (als Knoten) und deren paarweises Verknüpfen mit gerichteten Kanten eine hierarchische Struktur, die zur Semantischen Gliederung der zugrunde liegenden mathematischen Formel korrespondiert. Jeder (+)Elementkasten repräsentiert darin ein Semun, dem man aus einer zweifachen Liste (linker Fensterbereich) Typ und Wert zuweist. Für die Hinzufügung von Worten als (–)Elemente zu vorhandenen Kästen und für die Anfügung leerer Nachfolger ('/' in Abb. 5.3) zur Festlegung eines Teilastendes existieren weitere Mechanismen, siehe [SCH99].

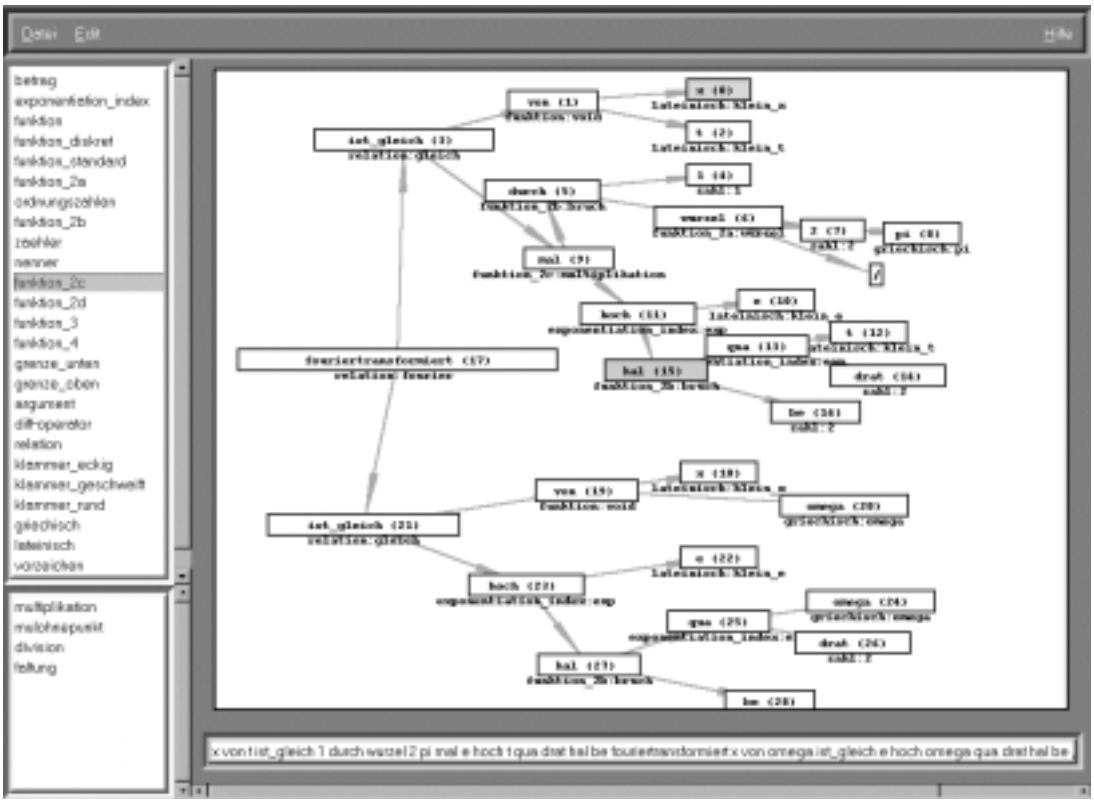


Abb. 5.3: Graphische Bedienoberfläche SPCHTOOL zur Annotation von Spracheingaben.

Als Resultat einer vollständigen Annotation wird die Semantische Gliederung incl. Elementpositionen in Textform abgelegt. Mittels weiterer Speicherungs- und Ladefunktionen ist ggf. auch eine spätere graphische Nachbearbeitung aller Trainingsdatensätze möglich.

5.3.2 Handschrift

Das Interface STKTOOL zur Annotation der Handschriftdaten wurde speziell für den Einsatz eines stiftbasierten Farb-LCD-Tabletts mit dementsprechendem visuellen Feedback entworfen. Das Bedienkonzept sieht neben verschiedenen Visualisierungsoptionen zur Bearbeitungskontrolle einen frei wählbaren Arbeitsablauf innerhalb der zu erledigenden Teilaufgaben vor, so dass jederzeit zwischen den einzelnen Arbeitsbereichen für Segmentierung, syntaktische und semantische Annotation gewechselt werden kann. Alle beteiligten Arbeitsschritte erfordern lediglich einfache stiftgestische und/oder Stiftauswahl-Operationen [HUN01B].

Erfassung. Die Testpersonen notieren jeden neuen Datensatz bei freier Positionierung auf einer in beiden Richtungen rollbaren Schreibfläche (elektronische Tinte). Die Speicherung erfolgt in einem internen binären Datenformat. Um die zeitliche Abfolge als syntaktisch relevante Information zu erhalten, sind nur Löschungen der zuletzt geschriebenen Linienzüge, im Sinne einer Mehrfach-Rückgängig-Funktion, erlaubt. Abb. 5.4 zeigt die Erfassung einer typischen Referenzformel in diesem Arbeitsbereich.

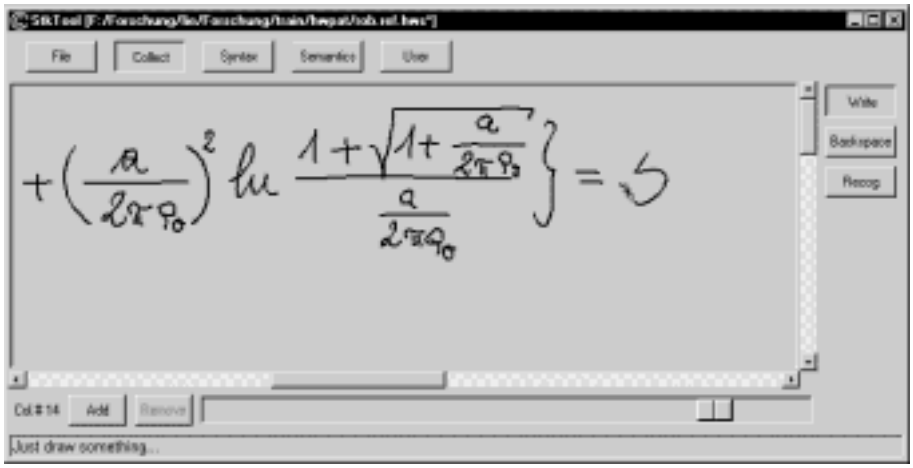


Abb. 5.4: STKTOOL, Arbeitsbereich Erfassung. Der untere Rollbalken dient zum Durchblättern der bereits aufgezeichneten Datensätze handgeschriebener Formeln. Mit der Schaltfläche 'Backspace' kann der Benutzer einen oder mehrere der zuletzt geschriebenen Linienzüge entfernen.

Segmentierung. Nachdem eine oder mehrere Handschriftproben zusammengetragen wurden, findet deren stückweise Segmentierung nach syntaktischen Elementen, also auf Schriftzeichenebene, im zugehörigen Arbeitsbereich statt. In dieser Phase werden zusammengehörige Linienzüge durch stiftgestisches Durchstreichen oder Umkreisen zu Segmenten gruppiert. Eine spezielle Farbcodierung sowie die (optionale) Einfassung der segmentierten Linienzuggruppen in umgebende Rechtecke dienen zur fortlaufenden Konsistenzprüfung. Während der Ausführung einer Stiftgeste werden alle betroffenen Linienzüge sukzessive farblich hervorgehoben, um die Gestenausführung zu erleichtern und den Segmentierungsfortgang inkrementell anzuzeigen.

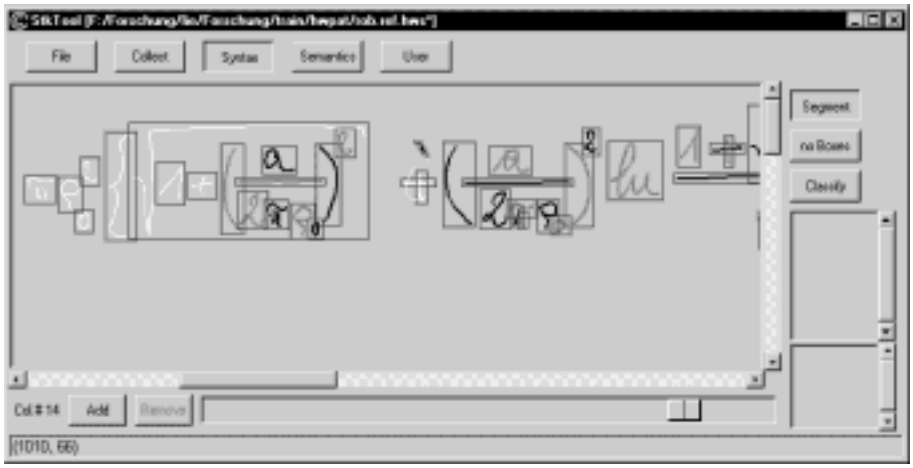


Abb. 5.5: STKTOOL, Arbeitsbereich Segmentierung. Im Ausgangszustand erscheinen alle erfassten handschriftlichen Linienzüge in alternierender Farbdarstellung gemäß Schreibreihenfolge (hier: abwechselnd schwarz und dunkelgrau). Während der Stiftgesten-Ausführung werden die angewählten Linienzüge farblich hervorgehoben (hier: weiß), danach weisen sie eine gemeinsame bounding box auf. Die Stiftgesten-Trajektorie wird während der Ausführung als dünne Spur angezeigt (hier: weiß). Die alternierende Farbgebung wird analog auf die entstandenen Segmentierungsgruppen übertragen. Linienzuggruppen, die bereits semantisch und/oder syntaktisch annotiert wurden (s.u.), werden in einer weiteren, einheitlichen Farbe dargestellt (hier: hellgrau).

Eine Besonderheit hinsichtlich des Eingabekomforts für den Endnutzer besteht in der Möglichkeit, bestimmte Schriftzeichen zu fragmentieren, wenn sie – in Abweichung von der Standardvorgabe – nicht ohne zeitliche Unterbrechung niedergeschrieben wurden: Durch die laut syntaktischem Modell (Gl. 4.13) erlaubte Zuweisung zweier oder mehrerer (+)Elemente zu einem Syntaktischen Modul können die jeweiligen Schriftzeichenfragmente mittels Definition zugehöriger Pseudo-Schriftzeichenklassen (siehe nächster Abschnitt) auf der semantischen Ebene wiederum als Einheit betrachtet werden. Der hier erhobene Datenbestand weist solche Fragmentierungen am häufigsten bei Wurzelausdrücken auf, wobei typischerweise zunächst der linke Teil des Wurzelzeichens ('√', Pseudoklasse 'rtprt') und daraufhin der Radikand geschrieben wird, bevor – in Orientierung an der horizontalen Ausdehnung des Radikanden – der abschließende Teil des Wurzelsymbols ('¬' oder '–') folgt. Ein ähnlicher Fall liegt bei der nachträglichen Verlängerung eines Bruchstriches vor, dieser wurde jedoch hier nicht untersucht. Prinzipiell bietet das vorgeschlagene Verfahren gleichwohl einen einfachen Weg, solche und vergleichbare Schreibgewohnheiten konsistent zu modellieren. Das Szenario der Schriftzeichen-Segmentierung ist in Abb. 5.5 illustriert.

Syntaktische Annotation. In diesem Arbeitsbereich wird jedem Segment (zusammengehörige Linienzuggruppe, s.o.) ein syntaktisches Element zugewiesen und damit ein Verweis auf eine im Schriftzeicheninventar (siehe Anh. A.3) enthaltene (Pseudo-)Symbolklasse gesetzt. Dies erreicht man durch a) Auswählen des gewünschten Eintrages aus einer zweifachen Liste (strukturiert gemäß Anh. A.3.2) und b) Selektion eines oder mehrerer zur gewählten Klasse passender Segmente mittels Stiftgestik. Auch dieser Arbeitsschritt wird durch geeignete Farbgebung unterstützt und kann durch Bewegen des abgehobenen Stiftes über die Arbeitsfläche jederzeit auf Konsistenz überprüft werden: Zum jeweils markierten Segment wird automatisch die momentan zugewiesene Symbolklasse angewählt und zusätzlich alle sonstigen in der aktuell bearbeiteten Niederschrift mit derselben Klasse annotierten Segmente hervorgehoben. Hat man etwa ein Exemplar einer mehrfach auftretenden Variablen (z.B. ein 'x') übersehen, so fällt dies sofort ins Auge und erfordert lediglich ein Stiftaufsetzen zur Behebung. Abb. 5.6 zeigt ein Beispiel zu dieser Funktionalität. Anhand der resultierenden syntaktischen Information findet als Nachverarbeitung die Merkmalextraktion gemäß Kap. 4.6.1 und die Eintragung der Referenzmuster für alle unterstützten Symbolklassen in eine DTW-Wissensbasis statt (siehe Kap. 5.7.2).

Semantische Annotation. Schließlich müssen die verschiedenen syntaktischen Konstituenten einer segmentierten und syntaktisch annotierten Schriftprobe mit semantischen Attributen versehen werden. Zu diesem Zweck wird die zur jeweiligen Referenzformel korrespondierende Semantische Gliederung als hierarchisches Listenobjekt in ein separates Teilfenster des Arbeitsbereiches geladen. (Die Erzeugung dieser Datenstruktur für sämtliche Referenzformeln geschieht zuvor mithilfe eines speziellen Parsers, der die konventionell mit einem LaTeX-kompatiblen Editor[15] gesetzten Formeln von LaTeX-Syntax in das Format Semantischer Gliederungen transformiert [Lie00].) Die Baumstruktur dieser Repräsentation wird sodann – abermals mittels Stiftgestik – auf die vorliegende Kollektion annotierter Schriftzeichen abgebildet, so dass schließlich jeder in der Formelniederschrift enthaltene semantisch geschlossene Teilausdruck (z.B. der Nenner eines Bruches) zum ent-

[15] Hier verwendet: TeXaide (Hersteller: DESIGN SCIENCE, Inc., Long Beach, CA, USA), Informationen unter http://www.mathtype.com/de/features/taform.stm .

sprechenden Semun korreliert ist. Analog zu den oben beschriebenen Arbeitsbereichen unterstützen eine geeignete Farbcodierung und die (hierarchische) Anzeige umschreibender Rechtecke die fehlerfreie Durchführung dieser Maßnahmen sowie eine fortlaufende Konsistenzkontrolle. Mit der vollständigen semantischen Zuordnung aller Formelkonstituenten liegt die zur rekursiven Offsetkalkulation über alle Strukturkomponenten nach Gl. 4.16 - 4.21 und zur Ermittlung der Element- und Übergangswahrscheinlichkeiten (Gl. 4.13 - 4.15) benötigte Zusatzinformation vor (siehe Kap. 5.5). Ein Beispiel zur semantischen Annotation ist in Abb. 5.7 dargestellt.

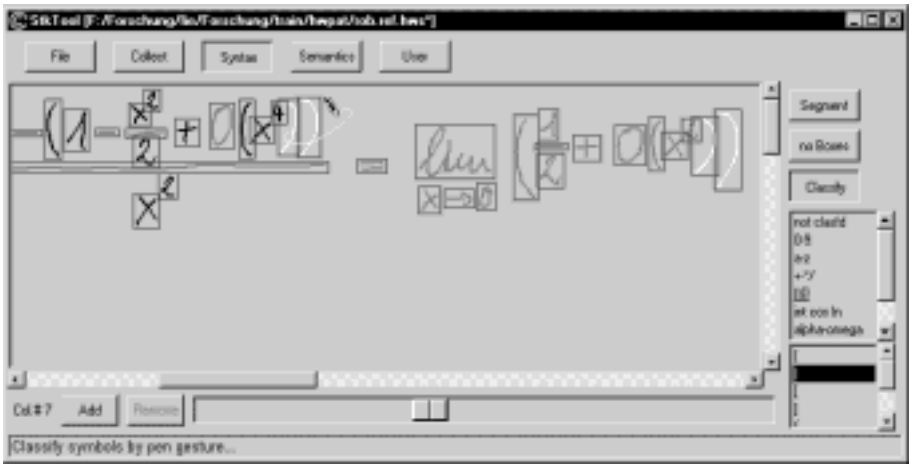


Abb. 5.6: STKTOOL, Arbeitsbereich Syntaktische Annotation. Zunächst wird eine verfügbare Symbolklasse ausgewählt – entweder durch Selektion aus einer Zweifachliste (rechts unten) oder durch Markierung eines Handschriftsegments, das bereits dieser Klasse zugewiesen wurde (in diesem Beispiel: eine der weiß dargestellten runden Klammern ganz rechts). Mittels Stiftgestik werden sodann weitere Segmente zur selben Klasse annotiert. In der Abbildung erfolgt dies gerade für weitere zwei Klammersymbole in einem Schritt. Zur Unterscheidung bereits annotierter bzw. noch nicht bearbeiteter Segmente dient wiederum eine entsprechende Farbgebung (hier: dunkelgrau bzw. schwarz).

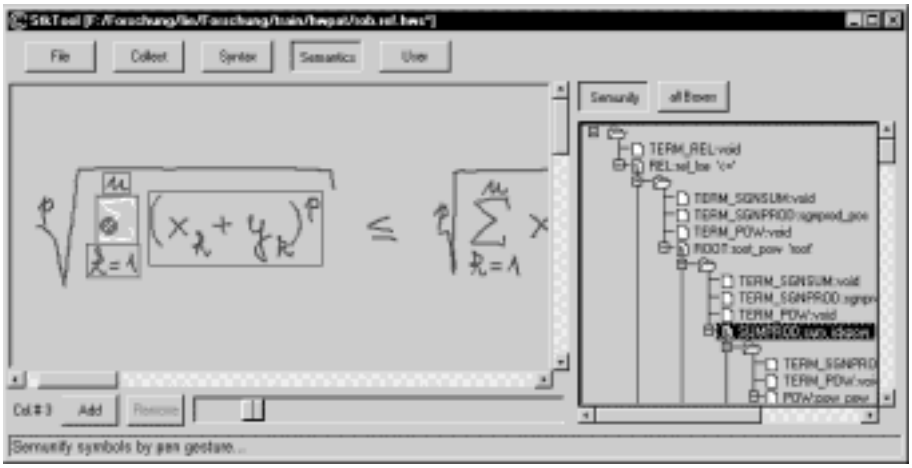


Abb. 5.7: STKTOOL, Arbeitsbereich Semantische Annotation. Das rechte Teilfenster beinhaltet die Semantische Gliederung zur eingegebenen Referenzformel in hierarchischer Listendarstellung. Durch Auswahl eines Semuns werden alle zugehörigen Schriftzeichen hervorgehoben (hier: weiß), außerdem werden alle Nachfolger-Teilausdrücke umrahmt. Am Beispiel der Summation erkennt man hieran, dass alle Nachfolger (untere/obere Grenze und Argument) vollständig annotiert wurden; bewegt man den abgehobenen Stift über die Schriftprobe, so lässt sich dies für jeden enthaltenen Teilausdruck veri-

fizieren, wobei in der Liste automatisch das jeweils zugewiesene Semun angesprungen wird. Alternativ ist auch umgekehrt eine Navigation über die Liste (mit umgekehrtem Effekt) möglich.

5.4 Semantisches Modell

Die probabilistische Modellierung auf der semantischen Ebene gemäß Gl. 4.5 - 4.8 erfordert die Abschätzung der dort definierten Parameter aus dem erhobenen Referenzmaterial, wofür das in [MÜL97], S. 59 ff. beschriebene iterative Trainingsverfahren (angepasst an die in Kap. 4.3 genannten Modifikationen) herangezogen werden kann. Es beruht weitgehend auf einer Häufigkeitsanalyse der insgesamt auftretenden Semuntypen, -werte und Nachfolgerscharen. Ein solches Training wurde zunächst anhand des annotierten Sprachkorpus (Kap. 5.3.1) durchgeführt, bezogen also auf die in Anh. A.4 aufgelisteten Referenzformeln. Es liegt auf der Hand, dass das aus dieser Vorgehensweise resultierende semantische Modell lediglich die mehr oder weniger zufälligen, in diesen Formeln auftretenden Kombinationen mathematischer Operatoren und Operanden widerspiegelt, so dass es für den Einsatz zur Decodierung beliebiger Testformeln aus der vorgegebenen Domäne bei deutlichen Strukturvariationen nicht geeignet sein kann. Anschaulich gesagt, wäre selbst der einfache Ausdruck $x = y$ nicht zulässig, da im Referenzkorpus kein Gleichheitsoperator auftritt, dem auf der rechten Seite ein Ausdruck vom Typ 'Variable' nachfolgt.

Da es andererseits außer Frage steht, dass gewisse statistische Bindungen in mathematischen Formeln vorhanden sind[16], ergeben sich zwei mögliche Schlussfolgerungen, um diese für die Formelerkennung nutzbar zu machen: 1) die Auswertung eines weitgehend erschöpfenden Formelkorpus anhand einschlägiger mathematischer Formelsammlungen, oder 2) eine Bündelung nach Anwendungsbereichen, z.B. „Physikalische Formeln“ oder „Kaufmännische Formeln“, für die dann mit entsprechend geringerem Aufwand getrennte Modellpakete erstellt werden können. Dabei ist zu beachten, dass ein solches semantisches Modell in der Regel nur einmal erstellt werden muss, da es weder vom Eingabemedium noch von den Schreib- oder Sprechgewohnheiten des Nutzers abhängig sein sollte. Weitergehende Zusammenhänge, beispielsweise das meist mehrfache Auftreten derselben Variablen innerhalb einer Formel, sind hier nicht berücksichtigt, da sie Abhängigkeiten höherer Ordnung darstellen, während die hier verwendete Modellierung nur direkte Nachfolger (erster Ordnung) in Betracht zieht.

Im Rahmen dieser Arbeit wurde keine dieser Möglichkeiten weiter verfolgt, da lediglich die prinzipielle Funktionsweise einer auf allen Ebenen probabilistischen Modellierung demonstriert werden sollte. Stattdessen wurde ausgehend von einem vorab erstellten Ausgangsmodell, das alle herangezogenen Referenzformeln umfasst, eine manuelle Generalisierung vorgenommen. Die aufgrund des „Baukastenprinzips“ entstehende hohe Vielfalt an Nachfolgerscharen konnte deutlich verringert werden, indem geeignete *Meta-Semuntypen* zur Repräsentation wiederkehrender Strukturen eingeführt wurden. Diese sind durchgängig mit dem Präfix 'TERM_' versehen, da sie spezielle Kategorien mathematischer Terme kennzeichnen, deren Auftreten vom jeweiligen Kontext abhängig ist. Zusätzlich sorgen diese Metatypen für eine konsistente Berücksichtigung der vorliegenden Opera-

[16] Um nur ein Beispiel zu nennen: Als untere Integrationsgrenze eines bestimmten Integrals tritt weitaus häufiger eine Gleichung der Form $x = 0$ auf als ein Wurzelausdruck oder ein Binomialkoeffizient.

tor-Präzedenzen (z.B. „Punkt vor Strich“), so dass ein Operator umso näher an die Wurzel einer Semantischen Gliederung gelangt, je niedriger sein Rang in Bezug auf die übrigen vorhandenen Operatoren ist.

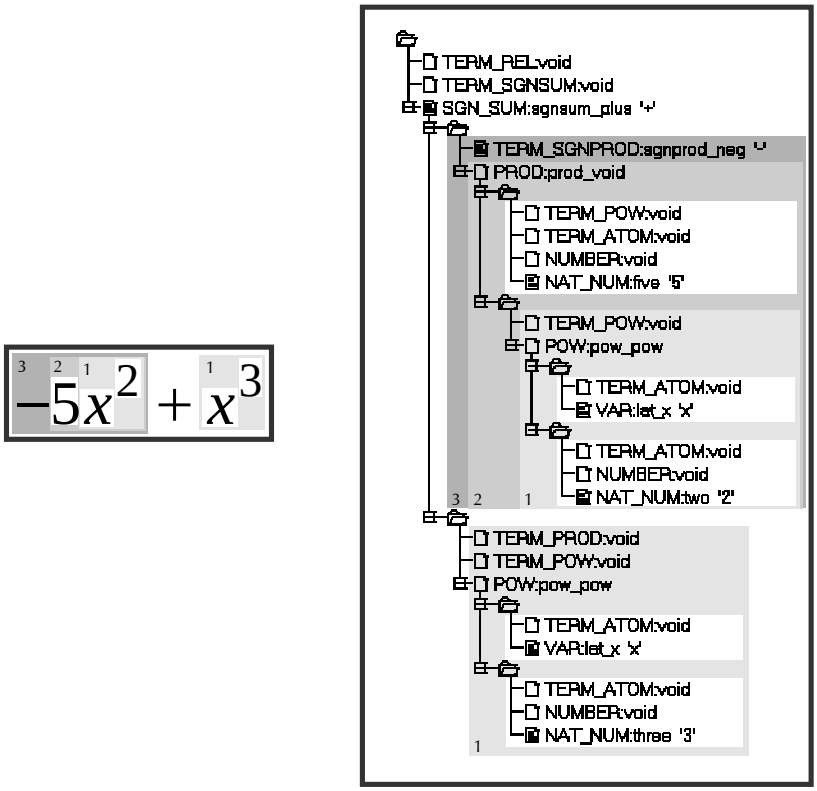


Abb. 5.8: Beispiel zur Modellierung mit Meta-Semuntypen.

Abb. 5.8 verdeutlicht die Modellierung mit Meta-Semuntypen anhand der einfachen Beispielformel $-5x^2 + x^3$, deren Semantische Gliederung als hierarchische Liste dargestellt ist. TERM_REL (Gleichungsterm) ist übergreifend als Wurzel-Semuntyp für alle zu erwartenden Formeln festgelegt, d.h. die Wurzelwahrscheinlichkeit η_0 laut Gl. 4.6 beträgt eins für $s_1.t$ = TERM_REL und null für alle anderen Semuntypen. Damit sind genau zwei Kategorien mathematischer Formeln möglich, nämlich einerseits (*Un-*)*Gleichungen* (TERM_REL mit Nachfolger REL) und andererseits, wie im Beispiel, allgemeine *Ausdrücke* (TERM_REL mit Nachfolger TERM_SGNSUM). Die Bezeichnung TERM_SGNSUM (Vorzeichensummenterm) ist folgendermaßen begründet[17]: Da Addition und Subtraktion die tiefstrangigen Grundoperatoren sind, ist jeder allgemeine Ausdruck als Summenterm aufzufassen, dessen innere Struktur wiederum aus Kombinationen höherrangiger Operatoren bestehen kann. Im Beispiel handelt es sich um die Summe zweier Terme, deren erster ($-5x^2$) vom Typ TERM_SGNPROD (Vorzeichenproduktterm) und deren zweiter (x^3) vom Typ TERM_PROD ist. Das Infix 'SGN' sorgt hier speziell dafür, dass der jeweils erste Summand eines Summenausdrucks ein negatives Vorzeichen tragen kann, im Gegensatz zu den übrigen Summanden (Ausdrücke der Form '$a \pm -b$' sind nicht erlaubt). Der erste Summand weist als innere Struktur das

[17] Der Einfachheit halber ist diese Betrachtung auf die Grundoperatoren Addition, Subtraktion, Multiplikation und Potenzierung beschränkt. Das semantische Modell berücksichtigt indessen das Wechselspiel aller unterstützten Operatoren durch entsprechende Meta-Semuntypen und deren Nachfolgerscharen.

Produkt (PROD) zweier Potenzterme (TERM_POW) auf, deren erster als Elementarterm (TERM_ATOM) die Zahl 5 enthält, während der zweite (x^2) als Potenz (POW) aus den beiden Elementartermen *x* und 2 ausgewiesen ist. Obwohl der zweite Summand (x^3) der übergeordneten Summe hier auch ein einfacher Potenzausdruck (POW) ist, ist er formal als Produktterm (TERM_PROD mit Nachfolgertyp TERM_POW, siehe Graustufe 1 in der Abbildung) repräsentiert, da er sich auf derselben semantischen Stufe befindet wie der „echte“ Produktterm $-5x^2$.

Als Konsequenz aus der oben genannten (semantischen) Unvollständigkeit der Referenzformelsammlung sind in dem so verallgemeinerten semantischen Modell alle vorhandenen Typ-, Wert- und Folgewahrscheinlichkeiten auf eins gesetzt, so dass de facto der Score-Anteil $Sc(S)$ (Gl. 4.10) für die A-priori-Auftrittswahrscheinlichkeit aller Semantischen Gliederungen den Wert null erhält und somit in Gl. 4.36 wegfällt. In Anh. A.6 ist das verwendete semantische Modell aufgelistet.

5.5 Syntaktische Modelle

Das syntaktische Modell ist entsprechend den drei darin beschriebenen Grundprozessen Übergänge, Element- und Offsetemissionen aus drei Teilmodellen – für gesprochene Sprache nur aus den ersten beiden – aufgebaut. Für das datenbasierte Training der Übergangs- sowie der Elementmodelle kann das in [STA97B], S. 47 ff. vorgestellte iterative Trainingsverfahren nach einer Verallgemeinerung hinsichtlich der hier vorgenommenen Modellerweiterungen eingesetzt werden. Das Offsetmodell (für geschriebene Formeln) ergibt sich durch Anwendung häufigkeitsbasierter Auswertungsmethoden in analoger Weise.

5.5.1 Übergangsmodelle

Sprache. Betrachtet man den aus der Datenerhebung (Kap. 5.2) hervorgegangenen Korpus gesprochener Formeln, so sind hinsichtlich der gewählten Sprechreihenfolge für die meisten der enthaltenen Formelkomponenten weitreichende, auch sprecherübergreifende Einschränkungen festzustellen. Dies hat zunächst seinen Grund in den bestehenden Konventionen bei der natürlichsprachlichen Kommunikation, wobei hier nicht unbedingt von einer grammatikalischen Korrektheit der auftretenden Formulierungen ausgegangen werden muss. Zudem trägt die Wortreihenfolge, wie in Kap. 4.4.4 kurz erwähnt, naturgemäß syntaktisch relevante Information, wie bereits der einfache Ausdruck 'x hoch y' belegt. In Anh. A.7.1 ist ein Auszug aus dem trainierten Übergangsmodell angegeben. Für den Semuntyp DIGIT (Ziffer) ergibt sich beispielsweise eine Übergangsmatrix, die nur einen definierten Pfad durch das Syntaktische Modul erlaubt. Beim Sprechen von Ziffernfolgen innerhalb zusammengesetzter Zahlen trat demnach keine Abweichung von der Standardvariante auf, so dass etwa die gebrochene Zahl 0,123 (repräsentiert als ZERO:void ← FRAC_NUM:void → DIGIT:one → DIGIT:two → DIGIT:three) grundsätzlich von links nach rechts „diktiert“ wird ('null komma eins zwei drei'). Für Brüche (Semuntyp FRAC) existieren hingegen im Trainingskorpus recht unterschiedliche sprachliche Umschreibungen (siehe Kap. 5.5.2) mit variierender Sprechreihenfolge, woraus eine dementsprechend stärker besetzte Übergangsmatrix resultiert.

Handschrift. Auch bei den handschriftlich eingegebenen Formeln nach Kap. 5.2 sind mitunter deutliche Präferenzen zu beobachten, was die Schreibreihenfolge innerhalb bestimmter Teilausdrü-

cke betrifft. Eine Abweichung von der gewohnten Links-Rechts-Schreibweise ist z.B. bei „einzeiligen“ Passagen der Form '$a + b - c = 0$' sehr selten. Allerdings wird im Ganzen gesehen eine wesentlich größere Freiheit beim Schreiben im Vergleich zum Sprechen beansprucht, wofür es zwei offensichtliche Ursachen gibt: Erstens ist, wie in Kap. 4.6.1 schon in Bezug auf die Einzelschriftzeichenbewertung erwähnt, das fertige Schriftbild für die Lesbarkeit einer Formel entscheidend und nicht der genaue Hergang seiner Entstehung. Zweitens wird ein Großteil der syntaktischen Information durch die zweidimensionale Schriftzeichenanordnung getragen, so dass ein Ausdruck wie z.B. $\sqrt[3]{x}$ unabhängig von der zeitlichen Abfolge seiner Bestandteile korrekt identifiziert werden kann. Hier sind von den 3! = 6 möglichen Folgen die vier Varianten 3$\rightarrow\sqrt{\ }\rightarrow$ x, $\sqrt{\ }\rightarrow$ 3$\rightarrow$ x, $\sqrt{\ }\rightarrow$ x $\rightarrow$ 3 sowie x $\rightarrow\sqrt{\ }\rightarrow$ 3 gebräuchlich. (Dass das Wurzelzeichen nicht zuletzt geschrieben wird, liegt auf der Hand.)

Aus diesen Gründen wurden in das Übergangsmodell für die Handschriftverarbeitung anstelle von datenbasierten Werten für jeden Semuntyp voll (bis auf die Hauptdiagonalen) besetzte Übergangsmatrizen aufgenommen. Dies hat zur Folge, dass eine beliebige Formel beispielsweise komplett „rückwärts“ geschrieben werden kann, was in der Praxis natürlich nicht zu erwarten ist. Mit anderen Worten bietet ein solches Modell gewisse Varianten bei der Schreibreihenfolge an, die von keinem Schreiber genutzt werden und damit zu eigentlich überflüssigen Decodierungs-Hypothesen führen. Auf der anderen Seite ist dadurch sichergestellt, dass keinerlei unnötige Beschränkungen der Schreibreihenfolge auferlegt werden, die den einen oder anderen Schreiber in seiner gewohnten Arbeitsweise beeinträchtigen könnten. Die einzige verbleibende Randbedingung ist die durch den Aufbau des verwendeten Syntaktischen Netzwerks verursachte *Schachtelungsvorschrift*, die sich wie folgt formulieren lässt:

> Jeder semantisch geschlossene Teilausdruck innerhalb
> einer Formel ist ununterbrochen niederzuschreiben.

Unter einem semantisch geschlossenen Teilausdruck ist dabei jeder denkbare Formelausschnitt zu verstehen, der durch einen vollständigen Teilast der Semantischen Gliederung repräsentiert wird (siehe Graustufen in Abb. 5.8). Hierzu ein Beispiel: Beim Schreiben eines Bruches ist die Reihenfolge von Zähler, Nenner und Bruchstrich jeweils wahlfrei – jede dieser Komponenten muss jedoch vollständig sein, bevor die nächste begonnen wird, und es dürfen auch keine (hier: dem Bruch) übergeordneten Formelkomponenten eingestreut werden. Interessanterweise konnte im Rahmen der vorliegenden Untersuchungen nachgewiesen werden, dass diese auf den ersten Blick nicht unerhebliche Beschränkung offenbar mit dem durchschnittlichen Schreibverhalten innerhalb der Domäne mathematischer Formeln absolut verträglich ist: Eine im Vorfeld durchgeführte Kontrolle auf der Basis einer Stichprobe[18] aus einer unabhängigen Handschrift-Datensammlung ergab eine ausnahmslose Übereinstimmung mit o.g. Vorschrift. Abgesehen von einigen (nachträglich korrigierten) Fehleingaben trifft dies ebenso auf das Referenzmaterial der vorliegenden Arbeit zu. In beiden Fällen wurden die Schreiber lediglich angewiesen, jedes *Schriftzeichen* ohne Unterbrechung niederzu-

[18] Die Probe bestand aus insgesamt 99 Online-Datensätzen handgeschriebener Formeln (je 11 Niederschriften von 9 Schreibern) des Original-Referenzmaterials von [WIN97B]. Sie wurden vom Autor freundlicherweise zu Vergleichszwecken zur Verfügung gestellt.

schreiben. Eine Beschränkung der Schreibreihenfolge war ihnen nicht bewusst, zumal die zeitliche Information in [WIN97B] – abgesehen wiederum von der Schriftzeichensegmentierung – überhaupt nicht ausgewertet wird.

5.5.2 Elementmodelle

Sprache. Bezüglich der Wortwahl erhält man aus dem datenbasierten Training recht unterschiedliche Ergebnisse, was die Vielfalt der je nach Semuntyp auftretenden Formulierungen betrifft (siehe Anh. A.7.2). Während z.B. Zahlenwerte (Semuntyp NUMBER und entsprechende Nachfolgertypen) und Variablenbezeichnungen (Semuntyp VAR) durch Konvention weitgehend festgelegt sind, liefert die Analyse gesprochener Brüche (FRAC) wechselnde Umschreibungen, zusammen mit den in Kap. 5.5.1 erwähnten Änderungen der Sprechreihenfolge. Für den Ausdruck '$^{x}\!/_{5}$' zum Beispiel stellt das Modell unter anderem folgende drei Sprechvarianten bereit:

- 'x geteilt durch fünf' — Knotenfolge: $A_1 \rightarrow B \rightarrow A_2$
- 'Bruchstrich, oben x, unten fünf' — Knotenfolge: $B \rightarrow A_1 \rightarrow A_2$
- 'x fünftel' — Knotenfolge: $A_1 \rightarrow A_2 \rightarrow B$

} siehe Übergangsmodell

Der zweitgenannte Fall tritt, manchmal auch in der Form 'großer Bruchstrich …', häufiger bei umfangreicheren Brüchen auf, wo offenbar ein gedanklicher Transfer von den Schreib- zu den Sprechgewohnheiten stattfindet. Für den dritten Fall wurde das Suffix '-tel' (Herkunft: 'Teil', gleichbedeutend mit 'geteilt durch') aus Konsistenzgründen als Pseudowort in das Elementmodell aufgenommen. Eine solche Sonderbehandlung ist allerdings nur bei derartigen Morphemketten ohne „Stammveränderung“[19] sinnvoll und daher eher exemplarischer Natur.

Handschrift. Grundsätzlich ist auch die Schriftzeichenwahl in Abhängigkeit vom Semuntyp unterschiedlich variabel. Jedoch ist hier von vornherein eine wesentlich stärkere Reglementierung gegeben, so dass im Normalfall für jede Kombination aus Semuntyp und -wert (z.B. DIGIT:one) pro Elementknoten des zugehörigen Syntaktischen Moduls nur ein festgelegtes Element (Schriftzeichen) existiert (z.B. '1'). Unterschiedliche Schreibweisen eines Schriftzeichens werden hingegen auf der Musterebene (DTW-Referenzmuster, siehe Kap. 5.7.2) verwaltet, es sei denn, es handelt sich um die in Kap. 5.3.2 angesprochene Fragmentierung eines Schriftzeichens (Beispiel Wurzelsymbol). In diesem Fall entstehen im Elementmodell Einträge für die entsprechenden (Pseudo-) Elemente (1. Elementknoten: 'rtprt' für '√', 2. Elementknoten: '¬' oder '–') als Alternativen zur zusammenhängenden „Normal“-Schreibweise ('root' für '$\sqrt{\ }$').

[19] Aus linguistischer Sicht besteht das Wort 'fünftel' aus dem Basismorphem 'fünf' und dem gebundenen (nicht eigenständigen) Endungsmorphem 'tel' [KUN00]. Da hier die Basis mit dem Wort 'fünf' (Element des Semuntyps DIGIT) übereinstimmt (keine Stammveränderung), ist eine getrennte Modellierung als Pseudowort 'tel' möglich. Das Gegenbeispiel 'drei' + 'tel' → 'drittel' zeigt die begrenzte Einsetzbarkeit dieser Methode.

5.5.3 Offsetmodell

Zur Modellierung der Offset-Scores nach Gl. 4.22 und 4.23 werden aus dem annotierten Trainingskorpus zunächst rekursiv (Inside-Out-Analyse, siehe S. 32) für alle auftretenden Strukturkomponentenpaare die zugehörigen Offset-Vektoren (Vorverarbeitung gemäß Kap. 4.4.4) berechnet. Diese werden sodann, in Abhängigkeit von Semuntyp und -wert, nach den zugrunde liegenden Strukturkomponentenpaarungen sortiert abgelegt. Für jede Offsetkomponente (jeweils $\Delta_{ij}\,\tilde{x}$, $\Delta_{ij}\,\tilde{y}$ und $\Delta_{ij}\,\tilde{g}$) schätzt man schließlich Mittelwert und Standardabweichung der zugehörigen Normalverteilung in der üblichen Weise ab ([RÅD97], S. 463 f.). Die resultierenden Offset-Matrizen für alle Semuntypen sind in Anh. A.7.3 für einen Schreiber vollständig angegeben. Eine schreiberunabhängige Modellierung ist zwar nach diesem Verfahren ebenso möglich, stellt sich aber in Verbindung mit der Modellierung auf Musterebene (Kap. 5.7.2) als nicht zufriedenstellend heraus (siehe Ergebnisse in Kap. 9.1.2).

5.6 Morphologische Modelle (Lexika)

5.6.1 Phonetisches Lexikon

Das im Laufe der Transkription aller Sprachdatensätze automatisch erstellte Wortverzeichnis (Kap. 5.3.1) bildet zusammen mit den Standard-Aussprachevarianten (in phonetischer Form) ein phonetisches Lexikon, aus dem die HMM-Musterbewertungsschicht die für ein angefragtes Element (Wort) zu verkettenden Phonemmodelle bestimmen kann. Das in Anh. A.5 aufgelistete alphabetische Lexikon wird hierfür noch in ein Baumlexikon überführt, so dass gemeinsame Wortbestandteile (z.B. Wortanfänge in 'Bruch' und 'Bruchstrich') nicht mehrfach verwaltet werden müssen.

5.6.2 Graphemisches Lexikon

Aus dem Handschriftkorpus ergibt sich analog ein Schriftzeichenlexikon, das die Klassenbezeichnungen für alle auftretenden Elemente (Handschriftsymbole) gemäß Anh. A.3.2 verwaltet. Die Zuordnung der DTW-Referenzmuster (Kap. 5.7.2) zu den Einträgen dieses Lexikons erfolgt bei mehreren Schreibweisen pro Schriftzeichen getrennt nach deren jeweiliger Linienzuganzahl (für die Musterbewertung nach Gl. 4.30). Bei jeder Anfrage aus der syntaktischen Ebene des semantischen Decoders hinsichtlich eines gegebenen Schriftzeichens kann dann selektiv auf den gewünschten Satz von Referenzmustern zugegriffen werden.

5.7 Musterbewertungs-Modelle

5.7.1 Hidden-Markov-Modelle

Die in Kap. 4.6.2 genannten Hidden-Markov-Phonemmodelle wurden unter Verwendung des hier erhobenen Sprachmaterials nach einem gängigen Standardverfahren neu trainiert. Hierfür fand zunächst eine halbautomatische Segmentierung (Phonemgrenzen-Detektion) der vorliegenden Audio-Dateien unter Zuhilfenahme der zugehörigen Transkriptionen sowie vortrainierter Phonemmodelle

statt. Nach der Vorverarbeitung des Audiomaterials zur Gewinnung von Trainings-Merkmalvektorsequenzen wurden die Parameter[20] der einzelnen HMMs nach dem *Maximum-Likelihood*-Verfahren mittels *VITERBI-Training* geschätzt. Eine detaillierte Darstellung des eingesetzten Verfahrens findet sich in [PFA00], S. 88 ff.

5.7.2 DTW-Wissensbasis

Referenzmuster. Auf der Grundlage der nach Kap. 5.3.2, S. 52 syntaktisch annotierten Handschriftdaten werden die zu jedem Element *e* (laut den Einträgen des Schriftzeichenlexikons) verfügbaren, vorverarbeiteten Linienzuggruppen $L_i^{(\delta_i)}$ des Referenzmaterials – getrennt nach ihrer Linienzuganzahl δ_i – als Referenzmuster $R_e^{(\delta_i)}$ für die DTW-Musterbewertung abgelegt (siehe Anh. A.8). Bei der Auswahl der Referenzformeln, ergänzt durch die Datensätze des Schriftzeicheninventars (Kap. 5.2), wurde dabei auf eine ausgewogene Zusammensetzung über alle Lexikoneinträge hinweg geachtet. Im Gegensatz zur HMM-Modellierung ist die DTW-Musterbewertung kein parametrisches Verfahren, bei dem jede zu modellierende Musterklasse (hier jede Schreibweise eines Schriftzeichens) anhand der Referenzdaten durch einen definierten Parametersatz beschrieben wird. Stattdessen erfolgt, wie in Kap. 4.6.1 beschrieben, für jede zu bewertende Linienzugsequenz ein Mustervergleich mit allen gleichartigen Referenzmustern der Wissensbasis zum angefragten Element *e*, um die bestmögliche Übereinstimmung zu ermitteln.

Datenanalyse und -reduktion. Um die Zuverlässigkeit dieses Ansatzes in der Praxis sicherzustellen und ggf. noch zu steigern, kommt ein spezielles Verfahren zur Nachbearbeitung der DTW-Wissensbasis zum Einsatz, das einerseits auf eine Erhöhung der *Diskriminanz* (Klassentrennbarkeit) und andererseits auf eine Verringerung der *Redundanz* durch Datenreduktion abzielt. Der vorhandene Bestand an Referenzmustern wird dabei iterativ nach folgendem Grundprinzip modifiziert:

Man führt nacheinander für jedes einzelne Referenzmuster (im Folgenden *R*) eine DTW-Musterbewertung im Vergleich zu allen anderen Referenzmustern der bestehenden Wissensbasis durch, ermittelt also dasjenige Referenzmuster, das zum betrachteten den geringsten Abstand aufweist („nächster Nachbar“, im Folgenden *N*). Für das weitere Vorgehen sind zwei Fälle zu unterscheiden:

1. Beide Muster *R* und *N* gehören zur selben Klasse. Es wird dann für beide Muster die Summe Σ der Abstände zu allen übrigen Referenzmustern dieser Klasse (genauer gesagt, der Unterklasse mit gleicher Linienzuganzahl, siehe oben) berechnet. Daraufhin wird das Muster (*R* oder *N*) mit kleinerem Σ aus der Wissensbasis entfernt, da das jeweils verbleibende Muster sich insgesamt deutlicher von den übrigen Vertretern dieser Klasse unterscheidet. Diese Maßnahme dient in erster Linie zur Verringerung der Redundanz, da zwar mehrere Referenzmuster pro Schreibweise eines Schriftzeichens erwünscht sind, diese sich aber voneinander – im Rahmen der in den Referenzdaten vorgefundenen Variationen – möglichst gut abgrenzen sollten.

[20] Die Modellierung der Wahrscheinlichkeitsdichtefunktionen erfolgte kontinuierlich, d.h. pro HMM-Zustand mittels einer gewichteten Überlagerung mehrerer Normalverteilungen.

2. *R* und *N* fallen in unterschiedliche Klassen. Diese „Fehlklassifikation“ ist zunächst als positiv zu bewerten, da sie belegt, dass kein weiteres Referenzmuster in der *R* zugeordneten Klasse existiert, das zu *R* redundant sein könnte. Somit wäre es naheliegend, das Referenzmuster *R* in der Wissensbasis zu belassen. Allerdings ist – als Grundbedingung für die Anwendbarkeit dieses Verfahrens – grundsätzlich jede beobachtete Schreibweise eines Schriftzeichens häufig genug im Trainingskorpus vertreten, um diese Schlussfolgerung zu hinterfragen: Die Zuordnung von *R* zu einer fremden Klasse deutet in der Praxis vielmehr darauf hin, dass eine für die Musterbewertung nachteilige Überschneidung dieser beiden Klassen im Bereich der beiden Muster *R* und *N* besteht. Bei ohnehin morphologisch verwandten Schriftzeichen wie etwa 'C' und '(' ist dies naturgemäß möglich, es kann jedoch auch ein Fehler bei der Annotation des Trainingsmaterials die Ursache sein. Setzt man abermals einen ausreichend umfangreichen und ausgewogenen Ausgangs-Datenbestand voraus, so ist es demnach von Vorteil, im vorliegenden Fall sowohl *R* als auch *N* aus der Wissensbasis zu eliminieren.

Bei der iterativen Anwendung dieser Vorgehensweise legt man eine untere Grenze für die Anzahl der pro Schreibweise jedes Schriftzeichens mindestens verbleibenden Referenzmuster fest, so dass die oben genannten Entscheidungskriterien ihre Gültigkeit behalten. Sinnvollerweise erfolgt die Datenreduktion getrennt nach Schreiber, woraufhin eine Zusammenführung der Wissensbasen mehrerer Schreiber und eine weitergehende Datenreduktion vorgesehen ist. Im Rahmen dieser Arbeit wurde das System ausschließlich schreiberabhängig eingesetzt, siehe Kap. 9.1.2.

6

Benutzertest

6.1 Vorbemerkung

Die geplante Integration mehrerer unterschiedlicher Modalitäten zur effizienten interaktiven Erfassung mathematischer Formeln erfordert zunächst ein Bedienkonzept, das einen sinnvollen Einsatz der zur Verfügung stehenden Eingabemöglichkeiten sowohl in unabhängiger als auch in kombinierter Form einschließt. Bereits in einer frühen Phase des Systementwurfs ist es zudem wünschenswert, einige Aussagen über die zu erwartende benutzerseitige Akzeptanz sowie die damit verbundene Gebrauchstauglichkeit des vorgesehenen Bedienschemas treffen zu können.

In den Bereich des *Usability Engineering* fällt in diesem Zusammenhang die Durchführung geeigneter Benutzertests an einer Auswahl freiwilliger Testpersonen, die der Zielgruppe potentieller Endanwender der zu erstellenden Applikation angehören sollten. In Abhängigkeit vom Entwicklungsstadium der Software und vom Funktionsumfang, den eine solche Untersuchung abdecken will, ist gegebenenfalls eine Simulation der Systemreaktion erforderlich, die sich nach dem jeweiligen Benutzerverhalten unter Einhaltung realistischer Leistungsmerkmale richtet. Dies bedeutet, dass ein authentischer Versuchsablauf gegenüber den Probanden unerlässlich ist, da sie nach der allgemein bevorzugten *WIZARD-OF-OZ*-Methode ([NIE94], S. 96) in dem Glauben belassen werden, mit einem voll funktionsfähigen Anwendungsprogramm konfrontiert zu sein. Der als WIZARD im Verborgenen den Ablauf steuernde Versuchsleiter hat dabei die anspruchsvolle Aufgabe, dies möglichst reibungslos und unauffällig zu tun. GEIGER et al. [GEI01] schlagen daher zur Vereinfachung dieser Vorgänge einen halbautomatisierten Versuchsablauf vor, um den WIZARD in kognitiver und aktorischer Hinsicht zu entlasten.

Die aus dem Benutzertest gewonnene Bewertung der vorgeschlagenen Bedienmechanismen liefert entweder brauchbare Anregungen zur Verbesserung des Interaktionskonzepts, die daraufhin in den bestehenden Prototypen eingearbeitet werden, oder sie erfordert – im ungünstigsten Fall – ein vollkommenes Redesign der Bedienumgebung. Diese Vorgehensweise lässt sich in zyklischer Form mehrmals wiederholen (*Usability Engineering Lifecycle*, siehe [NIE94], S. 71 ff.), bis ein bestimm-

tes Ausmaß an Übereinstimmung zwischen Systemmerkmalen und Benutzerwünschen erreicht worden ist.

6.2 Untersuchungsrahmen und Zielsetzung

Zur qualitativen und quantitativen Bewertung eines multimodalen Szenarios zur Formelerfassung wurde unter den folgenden Rahmenbedingungen ein zweigeteilter Benutzertest entworfen:

- Auf der Basis eines vorläufigen Bedienkonzepts soll der Umgang der Versuchspersonen (VP) mit einer interaktiven Arbeitsumgebung beurteilt werden, in der die handschriftliche Formeleingabe mit Mehrfach-Rückgängig-Funktion im Vordergrund steht.

- Zur Korrektur auftretender Erkennungsfehler wird in der ersten Testreihe natürliche Sprache in Verbindung mit Stiftgestik, in der zweiten Reihe zusätzlich konventionelle Interaktion angeboten. Die Teilnehmer haben außerdem die Freiheit, ihre Handschrifteingabe durch gleichzeitiges Mitsprechen beliebiger Formelbestandteile zu unterstützen. Im Vorgriff auf die in Kap. 9.1 angeführten Ergebnisse ist die rein sprachliche Eingabe kompletter Formeln von der Untersuchung ausgeschlossen.

- Die eingesetzte Bedienoberfläche stimmt mit einem bereits funktionsfähigen Prototypen des Gesamtsystems überein. Sie weist neben dem Handschrift-Eingabefeld nur einfache Schaltflächen auf, und zwar für Moduswechsel zwischen Normaleingabe und Korrektur (wahlweise auch mittels Stifttaste), Rückgängig-Funktion, Auswertung und Neueingabe. Anstelle einer echten Formelerkennung werden alle Systemreaktionen – mit Ausnahme des visuellen Feedbacks bei Schreibvorgang und Stiftgestik – im Rahmen eines WIZARD-OF-OZ-Aufbaus vom Versuchsleiter (VL) simuliert.

- Alle Testpersonen bearbeiten die gleiche Auswahl von Formeln und werden mit identischen, unterschiedlich fehlerhaften Erkennungsergebnissen konfrontiert, die sich an der Leistungsfähigkeit des Prototypen orientieren. Die Verwendung vorgefertigter Formeln und Resultate dient einerseits der Einhaltung einer realistischen Systemantwortzeit seitens des Versuchsleiters. Andererseits wird dadurch eine personenübergreifende Betrachtung ermöglicht, die einen Direktvergleich wiederkehrender Effekte umfasst (siehe Kap. 6.5.2).

Abgesehen vom frühen Teststadium liegt der Hauptgrund für die Simulation in der Unzumutbarkeit eines personenspezifischen Trainings des semantischen Decoders vor Versuchsbeginn. Der relativ hohe Aufwand für Datenerhebung und Annotation ist für einen potentiellen Langzeitnutzer vertretbar, nicht aber für einen freiwilligen Teilnehmer an einer ca. einstündigen Usability-Untersuchung. Ein weiteres Argument besteht in der Absicht, die VP zu einem möglichst freien Umgang mit den Systemkomponenten anzuregen, der durchaus über die Grenzen der existierenden Anwendung hinausgehen kann. Aus dem tatsächlichen Interaktionsverhalten lassen sich dann im Idealfall Benutzerwünsche ableiten, die bei Verwendung eines realen Systems durch entsprechend häufige Zurückweisung bzw. ein Fehlschlagen des Erkennungsvorgangs unterdrückt würden.

Die qualitative Auswertung erstreckt sich auf die Akzeptanz der zur Verfügung stehenden Interaktionswege und deren Einsatzweise (z.B. Sprechverhalten[21], Treffsicherheit und Wiederholungseffekte). Hierzu werden die objektiven Beobachtungen zusammen mit einer subjektiven Einschätzung durch Befragung herangezogen. Von quantitativem Interesse sind vorrangig Aussagen zum Zeitbedarf der Formelerfassung unter Einbeziehung zu erwartender Erkennerlaufzeiten. Hierzu wird insbesondere ein Vergleich zur rein konventionellen Interaktion gezogen, der über die eingangs genannte Voruntersuchung hinaus (siehe Kap. 1.1) den zu erwartenden Korrekturaufwand unter verschiedenen, realistischen Randbedingungen mitberücksichtigt.

6.3 Aufbau

Die Untersuchung wurde im Usability-Labor des Lehrstuhls für Mensch-Maschine-Kommunikation der Technischen Universität München durchgeführt. Abb. 6.1 zeigt den räumlichen Versuchsaufbau, Abb. 6.2 den Informationsfluss im schematischen Überblick. Die Darstellung bezieht sich auf die erste Testreihe; für die zweite Teiluntersuchung wurden Kontrollmonitor und Funktastatur durch einen weiteren PC ersetzt, der zur Fernsteuerung[22] des Versuchsrechners diente (die Probanden durften in dieser Variante die Tastatur selbst benutzen, siehe Kap. 6.4.2).

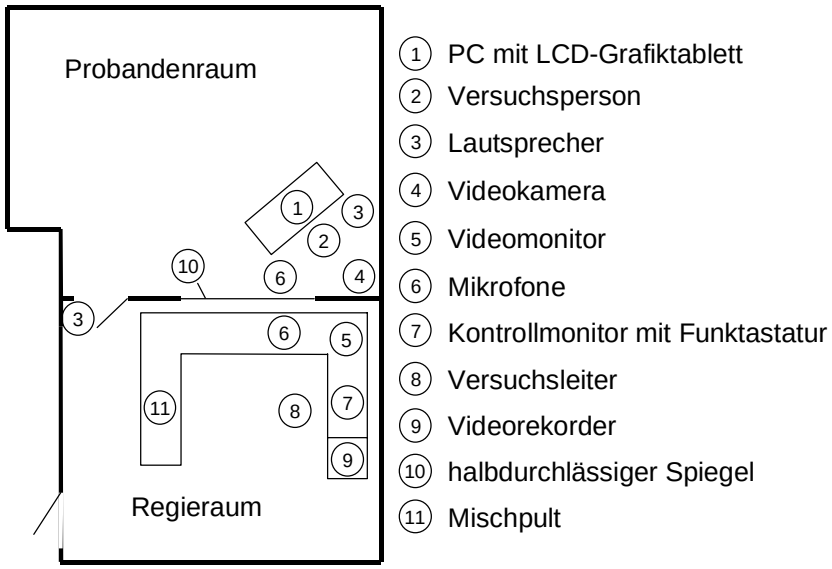


Abb. 6.1: Räumliche Anordnung für den Benutzertest im Usability-Labor.

Ein halbdurchlässiger Spiegel ⑩ schirmt die VP ② optisch von den Vorgängen im Regieraum ab. Von dort aus hat der VL ⑧ direkten Blickkontakt zum Versuchsgeschehen, das zudem über Videokamera ④ und -rekorder ⑨ zur späteren Auswertung aufgezeichnet wird. Ein zusätzlicher Monitor

[21] Ein weiterer Aspekt, der bereits in [SCH99] in Bezug auf gesprochene Formeln behandelt wird, ist die Mehrdeutigkeit von gesprochener Sprache. Sie ist in vielen Fällen nicht geeignet, um mathematische Formeln eindeutig ausdrücken zu können. Daher stellt sich die Frage, wie die Benutzer diesem Umstand Rechnung tragen und inwiefern eine Art Meta-Sprache Verwendung findet.

[22] Hierfür wurde die Client-Server-Software AT&T VNC (Virtual Network Computing) verwendet.

⑤ gibt das Videosignal wieder, um den VL über Ablaufdetails aus der Kameraperspektive zu informieren. Zur genaueren Rekonstruktion des Schreibverhaltens werden alle Handschriftdaten automatisch im internen Format des Prototypen auf Festplatte gespeichert. Kontrollmonitor und Funktastatur ⑦ sind zur Ablaufsteuerung direkt am Versuchsrechner ① angeschlossen, in der zweiten Testreihe erfolgt die Fernbedienung über eine Netzwerkverbindung zwischen Kontroll- und Versuchs-PC. Für die akustische Kommunikation (maßgeblich Rückfragen der VP und Anweisungen des VL) stehen in beiden Bereichen Raummikrofone ⑥ und Lautsprecher ③ zur Verfügung.

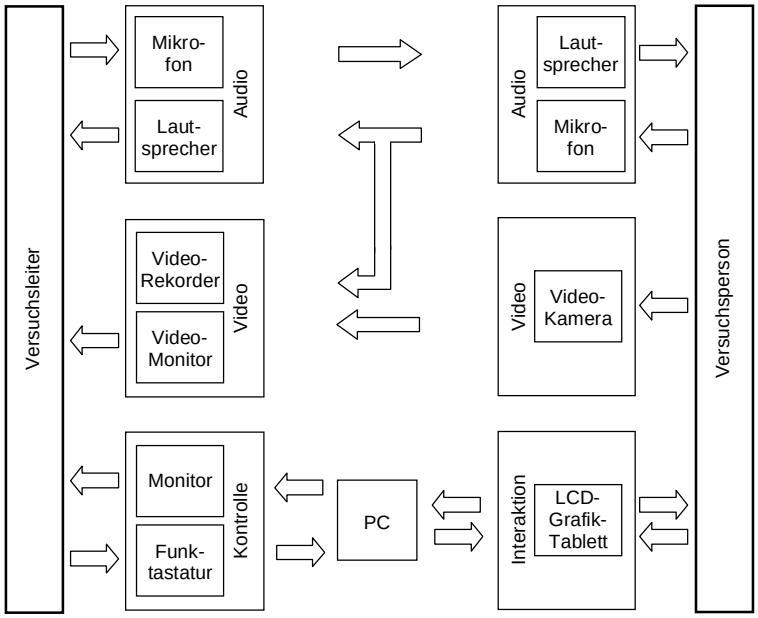


Abb. 6.2: Versuchsaufbau und Informationsfluss (Schema).

6.4 Durchführung

Das grundsätzliche Ablaufschema besteht in beiden Testreihen aus einem Wechsel zwischen der Präsentation einer einzugebenden Formel, der handschriftlichen Eingabe, der Visualisierung des (ggf. fehlerhaften) Erkennungsergebnisses, eines Korrekturversuches sowie der Rückgabe des Korrekturergebnisses. Die zweite Testreihe baut dabei inhaltlich auf Erkenntnissen aus der ersten Testreihe auf, wurde aber an einer disjunkten Personengruppe durchgeführt, um Lerneffekte zu vermeiden. In Abb. 6.3 sind verschiedene Perspektiven aus Regie- und Probandenraum vor und während der Versuchsdurchführung zu sehen.

Nach einer Kurzeinführung in die Funktionalität der Bedienelemente wird jede VP über den gewählten dreigeteilten Bildschirmaufbau aufgeklärt (siehe Abb. 6.4), bestehend aus Anweisungsfenster (Mitte), Eingabefenster (unten) und Ergebnisfenster (oben). Jegliche Zusatzinformation beschränkt sich auf das unbedingt notwendige Maß, um das Benutzerverhalten möglichst wenig zu beeinflussen.

Der VL verfügt über einen vorgefertigten Satz von Erkennungsergebnissen zu den verschiedenen Testformeln, die er nach einer angemessenen Antwortzeit in FrameMaker-Format (als Zielanwen-

dung des realen Systems) auf dem LCD-Display des Digitalisiertabletts erscheinen lässt. Dies geschieht mittels vorbelegter Tastenkombinationen in Abhängigkeit vom Versuchsgeschehen: Verschreibt sich die VP versehentlich oder misslingt ihr ein Korrekturversuch, so wird je nach Schweregrad eine Rückweisung der Eingabe simuliert oder aber die Bearbeitung der aktuellen Formel per Anweisung abgebrochen und zum nächsten Datensatz übergegangen.

Abb. 6.3: Versuchsdurchführung im Usability-Labor. Oben: Versuchsperson (links) und Versuchsleiter (rechts) im WIZARD-OF-OZ-Szenario. Unten: Blick aus dem abgedunkelten Probandenraum in den Regieraum (halbdurchlässiger Spiegel, links), Ansicht während des Versuchsablaufs (rechts).

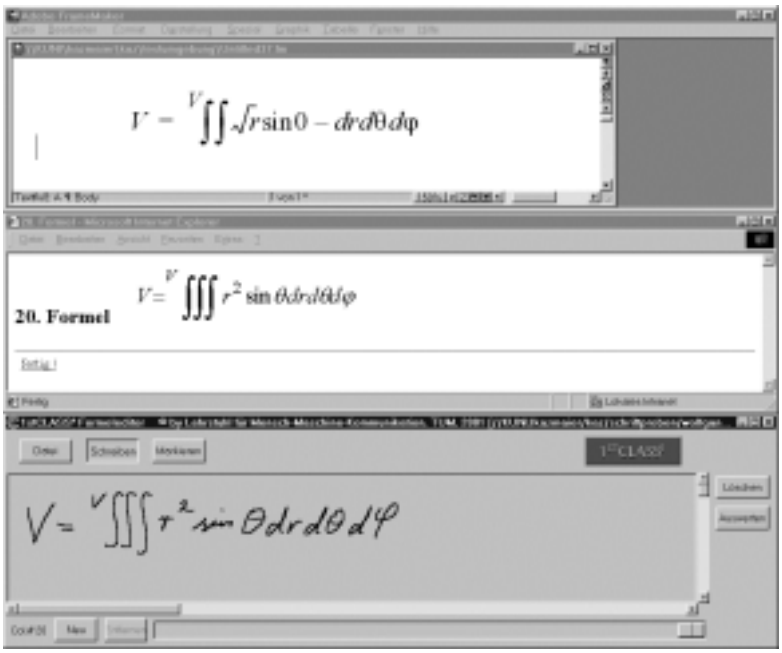


Abb. 6.4: Bildschirmaufbau für den Benutzertest, beide Versuchsreihen.

Im Anschluss an den Test werden anhand eines Fragebogens relevante persönliche Daten (z.B. Alter, Beruf und spezielle Vorkenntnisse) sowie die subjektiven Eindrücke der VP festgehalten. Neben einer skalenbezogenen Bewertung der einzelnen Interaktionswege werden dabei im Interview-Stil Aussagen zum Pro und Contra im Vergleich mit der „klassischen“ Formeleingabe (soweit den Teilnehmern bekannt), Anmerkungen und Verbesserungswünsche protokolliert.

6.4.1 Testreihe 1

An der ersten Teiluntersuchung nahmen 19 Personen teil. Das Durchschnittsalter betrug 26 Jahre, die Vorkenntnisse im Umgang mit dem Computer und in der mathematischen Formelnotation waren im Mittel leicht überdurchschnittlich. Jede vierte VP hatte bereits Erfahrung mit elektronischer Handschrifteingabe, während etwa die Hälfte der Teilnehmer mit Spracherkennungssystemen vertraut war. Die Zielgruppe der zu untersuchenden Anwendung war damit zufriedenstellend abgedeckt.

Der Versuchsumfang erstreckte sich auf 20 Formeln, von denen drei auf Anhieb korrekt zurückgegeben wurden. Bei allen übrigen Formeln wurden Erkennungsfehler eingestreut, die aus Schriftzeichen- und Strukturfehlern geringen bis mittleren Ausmaßes zusammengesetzt sind. Eine vollständige Auflistung findet sich in [KAZ02], S. 15 ff. Die Aufgabe der Fehlerkorrektur bestand jeweils in der Auffindung, Analyse und der eigentlichen Behebung der Fehler durch stiftgestische und sprachliche Interaktion. Die nicht unerhebliche „Denkzeit“ zwischen Präsentation und tatsächlichem Korrekturbeginn wurde insbesondere bei der quantitativen Auswertung zum Zeitbedarf berücksichtigt (siehe Kap. 6.5.3).

Im Testverlauf wurde die VP – abweichend von der sonst geübten Zurückhaltung des VL – zum parallelen Spracheinsatz animiert, indem auf eine Fehlerkennung einiger griechischer Symbole (ω und φ) ein entsprechender Hinweis folgte. Im Vorfeld war im Rahmen eines Probelaufs bereits festgestellt geworden, dass ein spontanes Mitsprechen von Formelteilen (ohne besonderen Anlass) sehr selten auftritt.

6.4.2 Testreihe 2

Den Teilnehmern der ersten Testreihe standen keine konventionellen haptischen Eingabegeräte zur Verfügung. Damit wurde die Zielsetzung unterstrichen, den Einsatz einer vorwiegend natürlichen Interaktion als alternatives Bedienkonzept zu untersuchen. Erwartungsgemäß kam als Zwischenergebnis der Wunsch nach einer zusätzlichen Verwendung von Maus und Tastatur auf, dem in der zweiten Testreihe auch entsprochen wurde.

Vorkenntnisse und Alter der neun Teilnehmer wichen nur geringfügig von der für die erste Testreihe genannten Zusammensetzung ab. Es wurde der gleiche Formelbestand wie oben verwendet, dieses Mal mit der freien Wahl, eine verbale oder aber konventionelle Korrektur (in FrameMaker) vorzunehmen. Hierfür fand zu Beginn der Untersuchung eine Kurzeinweisung in die Bedienung des FrameMath-Formeleditors statt.

6.5 Auswertung

Die nachfolgenden Ausführungen beschränken sich auf eine Zusammenfassung der wichtigsten Untersuchungsergebnisse. Für weitere Einzelheiten sei auf die umfassende Darstellung in [KAZ02] verwiesen.

6.5.1 Qualitative Aussagen

Objektiver Befund. Aus der Beobachtung des Benutzerverhaltens, ergänzt durch eine nachträgliche Sichtung der Versuchsaufzeichnungen, sind unter anderem folgende Tendenzen abzuleiten:

- Die Sprachkorrektur wird möglichst kurz und ohne Redundanzen vorgenommen. Meta-Sprache[23] bleibt dabei gänzlich ohne Bedeutung.
- Hauptsächlich wird zur Korrektur der fehlerhafte Term komplett (nach-)gesprochen („Ersetzung“).
- Bei bestimmten Fehlertypen wird die Korrektur so vorgenommen, dass lediglich ein Teilausdruck oder Schriftzeichen identifiziert und korrigiert wird, der bzw. das in den Augen der Versuchsperson für die fehlerhafte Struktur verantwortlich ist („Teilkorrektur“).
- Nur in Ausnahmefällen wird auf Sprache verzichtet und stattdessen die Handschrifteingabe nachgebessert („Fall-Back“).
- Paralleles Sprechen und Schreiben ist für die Benutzer wenig attraktiv. Wenn überhaupt, wird dies nur für kleinere Teilausdrücke oder einzelne Symbole praktiziert („unterstützende Eingabe“). Dieser Befund bestätigt die in [OVI94] aufgeführten Beobachtungen zur kontrastiven Verwendung von Handschrift und Sprache (siehe Kap. 2.2).
- Beim wahlweisen Einsatz von konventioneller oder verbaler Korrektur herrscht weitgehende Ausgewogenheit, wobei strukturell komplexere Fehler vorzugsweise konventionell behoben werden. Im direkten Vergleich gestaltet sich die Sprachkorrektur jedoch – bei jeweils identischem Fehler – erheblich einfacher, was den Testpersonen oft nicht bewusst zu sein scheint (siehe nächster Abschnitt und Kap. 6.5.3).

Subjektiver Befund. Die durchschnittliche Akzeptanz der verschiedenen Interaktionswege liegt vergleichsweise hoch, wie der untenstehende Auszug aus der Fragebogenauswertung erkennen lässt (Abb. 6.5). Ebenfalls aufgeführt sind die häufigsten Angaben zu Vor- und Nachteilen sowie zu Verbesserungsmöglichkeiten. Die Einschätzungen zur rein sprachlichen Formeleingabe sowie zur handschriftlichen Formelkorrektur sind in Anbetracht der Testbedingungen mehr hypothetischer Natur, bestätigen aber zumindest die Unterlegenheit der Sprache als Haupteingabemedium gegenüber Handschrift. Auffällig ist die relativ positive Bewertung der konventionellen Korrektur, die laut quantitativer Auswertung (Kap. 6.5.3) – im Hinblick auf den Zeitbedarf – der sprachlichen klar unterlegen ist. Die unterstützende, also parallele Spracheingabe wird zwar etwas schlechter bewer-

[23] Der Begriff Meta-Sprache bezieht sich hier auf die Verwendung von Füllwörtern, Umschreibungen und Aktionsanweisungen, z.B. 'bitte diesen Term hochstellen' oder 'das ist eine Potenz'.

tet als die sequentielle Sprachkorrektur, schneidet aber doch überraschend gut ab, wenn man den sehr seltenen Einsatz dieses Modus betrachtet. Als Hauptmodalität wird mit großer Übereinstimmung die Handschrifterfassung bevorzugt, und 27 von 28 Teilnehmern (96 %) finden insgesamt einen Formeleditor der vorgestellten Art attraktiver als eine konventionelle Lösung. Dieses Votum ist sicherlich von den genannten Vorteilen eines hohen Bedienkomforts ohne komplizierte Menüstrukturen beeinflusst, denen natürlich dennoch die vorgebrachten Nachteile und Verbesserungswünsche entgegenstehen, in erster Linie die bereits erwähnten Mehrdeutigkeiten innerhalb gesprochener Formeln, die zur Korrektur erforderliche Transferleistung (siehe hierzu Kap. 6.5.2) sowie die Problematik der Fehlerkennung schlechthin.

Wie gut hat Ihnen insgesamt die Möglichkeit der handschriftlichen und sprachlichen Eingabe gefallen ?	Mittelwert: Streuung:	4,5 0,7
Wie beurteilen Sie den Nutzen des Modus Sprache zur Eingabe einer kompletten Formel?	Mittelwert: Streuung:	3,0 1,6
Wie beurteilen Sie den Nutzen des Modus Sprache zur Unterstützung der handschriftlichen Formeleingabe?	Mittelwert: Streuung:	3,9 1,2
Wie beurteilen Sie den Nutzen des Modus Sprache zur Korrektur einer Formel ?	Mittelwert: Streuung:	4,3 0,9
Wie beurteilen Sie den Nutzen des Modus Handschrift zur Eingabe einer kompletten Formel ?	Mittelwert: Streuung:	4,5 0,8
Wie beurteilen Sie den Nutzen des Modus Handschrift zur Korrektur einer Formel ?	Mittelwert: Streuung:	3,7 1,3
Wie beurteilen Sie den Nutzen von Tastatur und Maus zur Korrektur einer Formel ?	Mittelwert: Streuung:	3,9 1,4
Würden Sie einen Formeleditor mit Handschrift- und Spracheingabe einem konventionellen Formeleditor vorziehen ?	Ja: Nein:	27 1

Vorteile:

- Intuitive und leichte Bedienung
- Komfort und Spaßfaktor
- Schnelligkeit und Effektivität, keine Menüs
- Mehr Modalitäten → bessere Erkennung; konventionelle Korrektur als Fall-Back

Nachteile:

- Kenntnis der Formelnotation notwendig
- Hemmung vor Kommunikation mit einer Maschine
- Mehrdeutigkeit von Sprache, komplexe Formeln schwer zu sprechen
- Transferleistung zwischen Eingabe und Fehler notwendig

Verbesserungswünsche:

- Alternativenauswahl bei Korrektur
- Freiere Handschriftkorrektur, Stift-Cursor ausblenden (Störfaktor)
- Automatischer Wechsel in den Modus Markieren nach Auswerten
- Tastaturverwendung (erste Testreihe), Kombination Haptik und Sprache
- Höhere Erkennungsleistung und –geschwindigkeit, schritthaltende Erkennung

Abb. 6.5: Auswertung der Fragebögen (beide Testreihen, Auszug). Die Mittelwerte und Streuungen (Standardabweichungen) beziehen sich auf eine Bewertungsskala von null (negativ) bis fünf (positiv).

6.5.2 Dekorrelationseffekt

Im Laufe der Untersuchungen trat ein interessanter Effekt auf, der hier am Beispiel einer Testformel separat vorgestellt wird. Es handelt sich um das Phänomen, dass die VP einen zu korrigierenden Teilausdruck in der fehlerhaften statt in der korrekten (handgeschriebenen) Form sprachlich wiedergibt. Die angeführte Formel liefert gleich ein zweifaches Beispiel für diesen Effekt und war auch am deutlichsten davon betroffen:

Vorgabe: $e = \lim_{x \to \infty} \left(1 + \frac{1}{x}\right)^{x}$ Erkennung: $e = \lim_{x \to 0} \left(1 + \frac{1}{x}\right)^{y}$

Die „falsche“ Korrektur sah in diesem Fall so aus:

Markieren: '∞' , *Sprechen:* 'null'; *Markieren:* 'x', *Sprechen:* 'ypsilon'.

Noch überraschender war die Kombination einer richtigen und einer falschen Einzelzeichenkorrektur innerhalb dieser Formel. In der ersten Testreihe korrigierten nur sieben VP beide Fehler wirksam, die anderen zwölf wiederholten entweder einen oder beide Erkennungsfehler verbal. Vereinzelt wurde einer VP auch während des Sprechens die Situation bewusst, was sich etwa so äußerte: 'Null, ach so, null, ach Schmarrn, eben gerade nicht...'. Die Mehrzahl der betreffenden Teilnehmer erfasste den Sachverhalt offenbar nachträglich (aufgrund des unveränderten Erkennungsergebnisses), so dass sie im weiteren Verlauf ein nochmaliges Auftreten dieses Effektes vermeiden konnten.

Dass es sich hier um einen systematischen Effekt handelt, der eng mit dem verwendeten Bedienkonzept zusammenhängt, liegt auf der Hand. Es konnten jedoch keine experimental-psychologischen Untersuchungen in der Literatur gefunden werden, die ein vergleichbares Szenario betreffen. Aus kognitiver Sicht lässt sich dieses Phänomen möglicherweise folgendermaßen erklären: Durch die Notwendigkeit, die Sprachkorrektur mittels Stiftgestik an der Handschrifteingabe vorzunehmen, ist der Versuchsperson die Korrelation zwischen dem (subjektiv korrekten) Handschrifteintrag und dem (objektiv falschen) Erkennungsergebnis mitunter nicht gegenwärtig. Sie sieht sich dann veranlasst – ganz im Sinne eines Korrekturvorgangs – den markierten Teilausdruck per Sprache durch etwas anderes zu ersetzen. Als Folge ihrer Missinterpretation der Niederschrift als „Fehler“ verwendet die Versuchsperson instinktiv den eigentlichen Fehler aus dem Erkennungsergebnis als vermeintlich richtigen Korrekturausdruck. In Anlehnung an diese Deutung wird dieses Verhalten daher als *Dekorrelationseffekt* bezeichnet. Es ist zu vermuten, dass in aktuellen und künftigen Untersuchungen zur multimodalen Interaktion ähnliche Beobachtungen gemacht werden können.

6.5.3 Quantitative Aussagen

Die quantitative Auswertung der Benutzerinteraktion konzentriert sich vor folgendem Hintergrund auf den Zeitaufwand, der auf die einzelnen Bearbeitungsabschnitte entfällt:

Nachdem über die qualitativen Vorzüge einer natürlichen Formelerfassung hinsichtlich ihrer Ergonomie und Intuitivität weitgehende Einigkeit herrscht, verbleibt als schwerwiegendste Einschränkung einer solchen Bedienumgebung zunächst die *Fehleranfälligkeit*, die allen erkennungsgestütz-

ten Eingabeverfahren grundsätzlich innewohnt. Im Gegensatz zu einer Vielzahl gängiger Anwendungsszenarien, in denen eine Eingabesequenz jeweils eine einzelne Systemfunktion, ggf. ergänzt durch entsprechende Parameter, ansprechen soll, kommt eine fehlerhaft erkannte mathematische Formel in der Regel keinem *Fehlschlagen* der erfolgten Eingabe gleich. Stattdessen ist es innerhalb eines interaktiven Formeleditors möglich und auch ausgesprochen sinnvoll, fehlerhafte Formelteile mit vertretbarem Zusatzaufwand nachzubessern, so dass letztendlich das gewünschte Ergebnis vorliegt[24]. Sieht man demnach von der gegebenen Lästigkeit jeder fehlerhaften Erkennung ab, so sind die Gesamterkennungsleistung auf der einen und der Gesamtzeitbedarf auf der anderen Seite gegeneinander abzuwägen. Anders gesagt, Bedienkonzept A mit einer gewissen Restfehlerrate (incl. einfacher Korrekturmaßnahmen) ist gegenüber einem konventionellen und dementsprechend robusten Bedienkonzept B zu bevorzugen, wenn es zugleich einen signifikanten Zeitvorteil für den Benutzer bietet.

Ausgehend von dieser Motivation wurde die Gesamtbearbeitungszeit für jede Formel zunächst in die drei Anteile Schreib-, Denk- und Korrekturzeit unterteilt. Geht man vorerst von einer Echtzeiterkennung bei Handschrift- und Spracheingabe aus (in Bezug auf die simulierte Systemantwort[25]), so endet die Schreibzeit mit der Fertigstellung der Handschrifteingabe bzw. der Präsentation des Erkennungsergebnisses, woran sich unmittelbar die Denkzeit (zur Fehlerauffindung, -analyse und Korrekturplanung) anschließt. Mit der Durchführung einer Stiftgeste bzw. dem Einsatz von Maus und Tastatur zur Fehlerbehebung beginnt sodann die Korrekturzeit, die mit dem Abschluss des Korrekturvorgangs und damit der Rückgabe des Endresultates endet. In Abb. 6.6 ist die durchschnittliche Zusammensetzung der Formelbearbeitungszeit aus diesen drei Beiträgen für die erste Testreihe (Stift-/Sprachkorrektur) auf der linken Seite dargestellt. Um eine realistische Abschätzung für die Gesamtbearbeitungszeit unter Berücksichtigung zu erwartender Erkennerlaufzeiten zu erhalten, sind auf der rechten Seite die Schreibzeit mit einem Faktor 1,5 und die Korrekturzeit mit einem Faktor 3 (in Anlehnung an den Leistungsstand des Prototypen) beaufschlagt. Als beachtenswertes Zwischenergebnis nimmt die aus Denk- und Korrekturzeit zusammengesetzte Fehlerbehebung mittels Stiftgestik und Sprache im Durchschnitt mehr als die Hälfte der Gesamtbearbeitungszeit in Anspruch.

Diese auf den ersten Blick ernüchternde Feststellung wird wiederum relativiert, wenn man sie in Bezug zur konventionellen Korrektur oder gar zur rein konventionellen Formelerfassung setzt. Abb. 6.7 stellt die Gesamtbearbeitungszeiten, wie sie sich aus den beiden Testreihen abhängig vom Korrekturmodus ergeben, dem zu erwartenden Gesamtzeitbedarf innerhalb eines konventionellen Formeleditors gegenüber. Die Abschätzung beruht auf dem in der Voruntersuchung (Tab. 1.1) ermittelten Zeitverhältnis von 1:7 (Anfänger) bzw. 1:5 (Experte) zwischen handschriftlicher und konventioneller Formelerfassung (reine Eingabezeiten). Legt man als Mittelmaß für durchschnitt-

[24] Diese Strategie setzt natürlich eine Begrenzung der Fehlerhaftigkeit in Schweregrad und Ausdehnung voraus. Das in dieser Arbeit vorgestellte Verfahren bewirkt in den meisten Fällen eine lokale Begrenzung auftretender Erkennungsfehler (siehe auch Kap. 9).

[25] Tatsächlich wurde während der Untersuchung durch den VL eine deutliche Antwortverzögerung simuliert. Diese diente aber in erster Linie der Glaubwürdigkeit des Versuchsablaufs und wurde für die quantitative Auswertung unberücksichtigt gelassen. Stattdessen erfolgte eine nachträgliche Skalierung der entsprechenden Messzeiten auf realistische Werte (siehe unten). „Schreib"- und „Korrektur"zeit enthalten also letztendlich einen Systemantwortbeitrag, der über das eigentliche Schreiben und Korrigieren hinausgeht.

lich geübte Benutzer den Wert 1:6 zugrunde und normiert die Darstellung auf den rein konventionellen Fall (Gesamtzeitanteil 100 % in Abb. 6.7 ganz rechts), dann sind die aus der multimodalen Interaktion ermittelten Messzeiten *nach* Berücksichtigung der Erkennerlaufzeiten (s.o., Faktor 1,5 für Handschrift) entsprechend einem zu erwartenden Zeitverhältnis von 1:4 auf einen Schreibzeitanteil von 25 % zu skalieren.

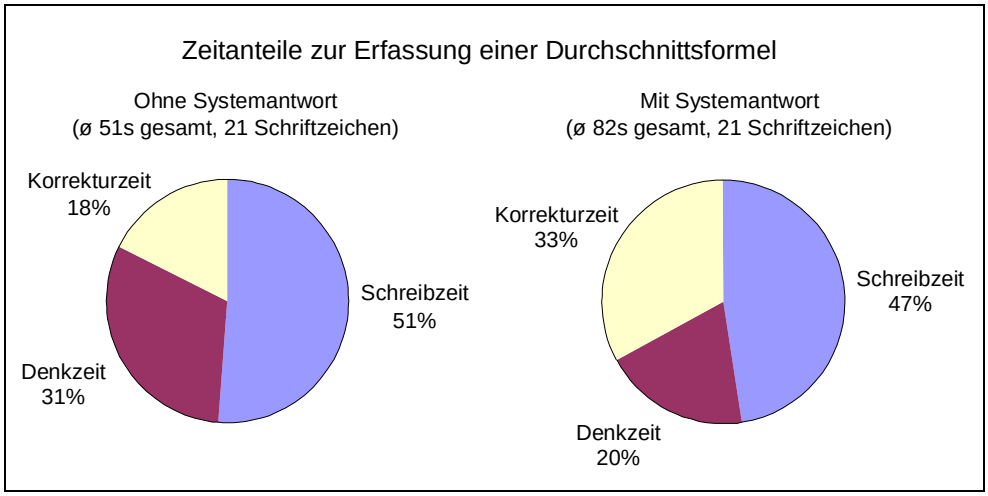


Abb. 6.6: Zusammensetzung der Gesamtbearbeitungszeit pro Formel (Durchschnittswerte bei handschriftlicher Formelerfassung mit Stift-/Sprachkorrektur). Nach Berücksichtigung zu erwartender Systemantwortzeiten (rechts) entfällt etwa die Hälfte des Zeitaufwands auf die Ersterfassung sowie ein Fünftel bzw. ein Drittel auf Korrekturplanung bzw. -durchführung.

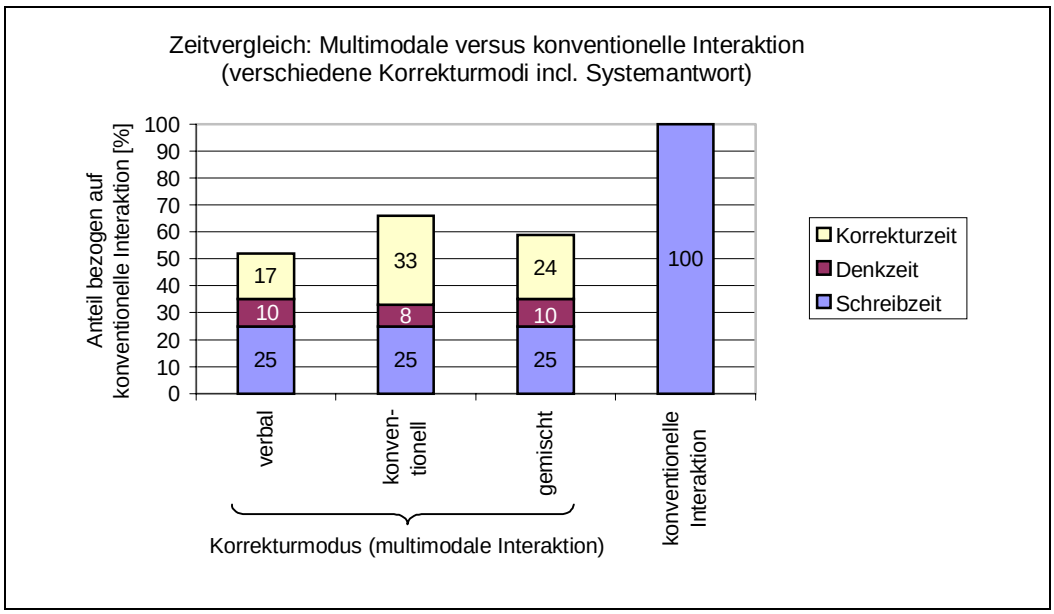


Abb. 6.7: Vergleich der Gesamtbearbeitungszeiten bei multimodaler und konventioneller Interaktion. Die multimodale Interaktion liefert einschließlich der verbalen Korrektur eines durchschnittlichen Erkennungsfehlers einen Zeitvorteil von etwa einem Faktor 2 gegenüber einer konventionellen Formelerfassung. Die konventionelle Korrektur erfordert – entgegen dem subjektiven Empfinden – ca. 50 % mehr Zeitaufwand als die sprachliche.

Der in Abb. 6.6 rechts angegebene Fall (erste Testreihe, verbaler Korrekturmodus) liefert damit einen Zeitvorteil von annähernd einem Faktor 2 (52 % des konventionellen Zeitaufwands), während die Ergebnisse der zweiten Testreihe (gemischter Korrekturmodus) mit 59 % etwas ungünstiger ausfallen. Dies hat seine Hauptursache in der durchschnittlich zeitintensiveren konventionellen gegenüber der verbalen Korrektur, wie die gesonderte Auswertung der konventionell durchgeführten Korrekturvorgänge in Abb. 6.7 erkennen lässt. Der dort ausgewiesene Zeitunterschied bei der Durchführung einer Korrektur (33:17 ≈ Faktor 2, wohlgemerkt bei dreifacher Echtzeit-Spracherkennung) reduziert sich aufgrund des additiven, im konventionellen Modus etwas geringer ausfallenden Denkzeitanteils auf insgesamt immerhin noch 1,5 (50 % zeitlicher Mehraufwand im konventionellen Korrekturmodus). Wie bereits in Kap. 6.5.1 angemerkt, steht dieser quantitative Befund im Widerspruch zur allgemein hohen Akzeptanz der konventionellen Korrektur. Dabei ist nicht zu vergessen, dass das subjektive Zeitempfinden nachweislich vom Beschäftigungsgrad der VP beeinflusst ist, so dass die Durchführung einer Korrektur mit Maus und Tastatur durchaus kürzer wirken kann als eine knappe Sprachanweisung mit anschließender Wartezeit. Hinzu kommt noch die nicht unbedingt intuitive Bedienweise des FrameMath-Editors (wenngleich ein allgemeiner Kritikpunkt an konventionellen Systemen): Die subjektive Bewertung der konventionellen Korrektur mag sich mehr auf den Umgang mit einem der VP vertrauten Formeleditor beziehen als auf den tatsächlichen Versuchsablauf.

Zusammenfassend ist festzuhalten, dass der Einsatz eines multimodalen Formeleditors der vorgeschlagenen Art einer konventionellen Lösung vorzuziehen ist, wenn die beschriebenen Korrekturstrategien im Durchschnitt zur Behebung auftretender Erkennungsfehler greifen und damit nach obiger Abschätzung eine Zeitersparnis bis zur Hälfte der Gesamtbearbeitungszeit resultiert.

7

Multimodale Interaktion

Unter Berücksichtigung der aus dem Benutzertest gewonnenen Erkenntnisse wurde eine Bedienumgebung für die multimodale Formelerfassung implementiert, deren Einsatzweise in diesem Kapitel näher vorgestellt wird. Um das Zusammenspiel der einzelnen Bearbeitungsstufen zu verdeutlichen, orientiert sich die Beschreibung an einem durchgängigen Beispiel zur Eingabe folgender Formel:

$$G_k^m \left(\sqrt[n]{\prod_n (k + a_n)} \right)^m$$

Zusätzlich zur semantischen Decodierung als Hauptgegenstand dieser Arbeit kommen dabei spezielle Methoden für die inkrementelle Selektion durch Stiftgestik, die Datenfusion aus Handschrift- und Spracheingabe sowie die Datentransformation in das MIF-Format (und zurück) zum Einsatz.

7.1 Handschrifterfassung

7.1.1 Schritthaltende Verarbeitung

Bei der Verarbeitung der handschriftlichen (analog auch der sprachlichen) Eingabe wird die durch den Schreibvorgang beanspruchte Zeitdauer systemseitig genutzt, um den Erkennungsvorgang voranzutreiben. Voraussetzung hierfür ist die auf S. 42 angesprochene First-Last-Abarbeitung, bei der die innerhalb des Suchverfahrens anfallenden Element-Emissionen (Phase 4a und b in Abb. 4.7) nur bis zur aktuell letzten Linienzugposition durchführbar sind. In der Praxis hält die Einspeisung der Handschriftdaten in den Suchraum für die ersten zehn bis zwanzig Linienzüge (je nach Schreibgeschwindigkeit und Rechnerleistung) mit der Eingabe Schritt, woraufhin sich ein zunehmender Laufzeitunterschied bemerkbar macht. Dabei ist zu bedenken, dass die Größe des Suchraums in der Grammatikschicht aufgrund der entstehenden Vielfalt möglicher Operatorkombinationen polynomial mit dem Umfang der Schreibfolge anwächst.

Wie in Abb. 7.1 oben zu sehen, wird der Bearbeitungsstand dem Benutzer durch fortschreitende Einfärbung (hier schwarz) der bereits untersuchten Linienzüge signalisiert[26]. Eine Handschriftkorrektur mittels Mehrfach-Rückgängig-Funktion bleibt damit ohne Auswirkung auf das laufende Suchverfahren, solange die Entfernung von Linienzügen nicht über die aktuelle Abarbeitungsposition hinausgeht. Im Beispiel löscht der Benutzer infolge eines Schreibfehlers die letzten drei Linienzüge, also 'm', ')' und '(', und setzt die Eingabe fort, bevor der Decoder bis zur betreffenden Position vorgedrungen ist ('a' wird weiterhin weiß angezeigt). Geht die Korrektur hingegen über den Abarbeitungsstand hinaus, so wird der Suchvorgang entsprechend wiederholt.

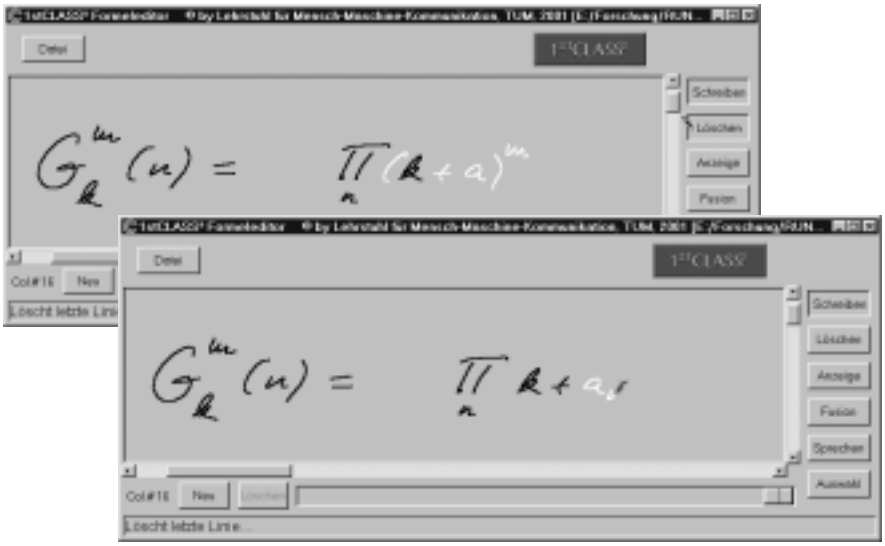


Abb. 7.1: Schritthaltende Verarbeitung und Handschriftkorrektur.

7.1.2 Zwischenauswertung

Zur Vermeidung aufwendiger Nachkorrekturen dient ein weiterer Mechanismus, der eine Zwischenkontrolle der Handschrifteingabe erlaubt (siehe Abb. 7.2): Zum gewünschten Zeitpunkt wird dem Benutzer das anhand der bisherigen Teileingabe ermittelte Zwischenergebnis der Formelerkennung im Zielformat des FrameMaker-Formeleditors angezeigt. Die dafür erforderliche Datentransformation ist im Zusammenhang mit der konventionellen Interaktion in Kap. 7.3.2 erläutert. Ein Erkennungsfehler der dargestellten Art (geklammerter Ausdruck) bleibt im Normalfall[27] auch nach weiterer Eingabe bestehen, so dass oft eine frühzeitige Behebung sinnvoll ist. Andererseits kann auch, wie im Beispiel, zunächst die Handschrifteingabe vervollständigt werden, wenn die Teilformel in ihrer Gesamtstruktur korrekt erkannt wurde. Der hier lokal begrenzte Fehler wird danach entweder per Stiftgestik und Sprache (Kap. 7.2) oder konventionell in FrameMaker korrigiert.

---

[26] Die weiß dargestellte öffnende Klammer '(' ist hier noch nicht verarbeitet, da der Teilausdruck 'k+a' nachträglich eingeklammert wurde. Der zuletzt verarbeitete Linienzug ist 'k'.

[27] Es gibt durchaus Gegenbeispiele, da die verbleibenden Formelteile ja u.U. Einfluss auf die Strukturbewertung des fehlerhaft erkannten Teilausdrucks haben. So kann etwa '$K-J$' isoliert als '$I<-J$' erkannt werden, nicht aber '$\sqrt{K-J}$' als '$\sqrt{I<-J}$'.

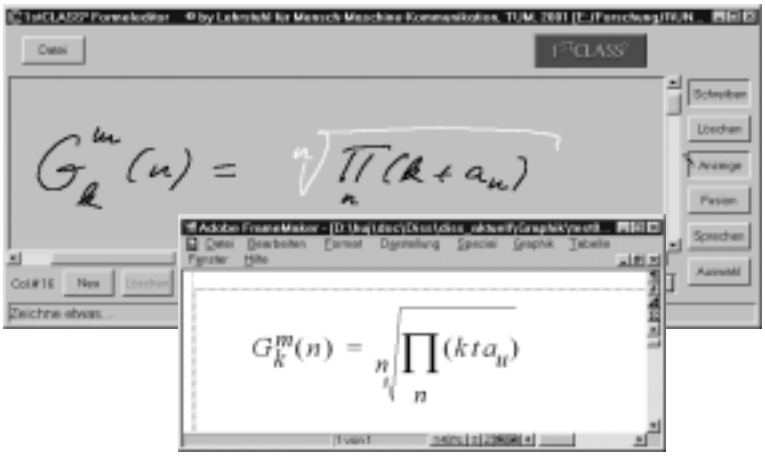


Abb. 7.2: Zwischenauswertung der Handschrifteingabe.

7.2 Stiftgestik und Sprachinteraktion

7.2.1 Intelligente Stiftgestik

Die Selektion eines zu korrigierenden Teilausdrucks erfolgt nach dem gleichen Prinzip wie die in Kap. 5.3.2 beschriebene stiftgestische Annotation. Während der Gestenausführung erfolgt ein visuelles Feedback, indem die ausgewählten Handschriftsegmente inkrementell hervorgehoben werden. Abb. 7.3 verdeutlicht die verwendete Vorgehensweise zur Interpretation der Stiftkurve, die zugleich durchstreichende und einkreisende Abschnitte enthalten kann.

Nach Methode 1 wird zunächst jedes Segment selektiert, sobald es von der Trajektorie der Stiftbewegung berührt oder geschnitten wird (Typ Durchstreichung, Schriftzeichen 'k' an der durch ° gekennzeichneten Stelle). Weiterhin wird ein Segment hinzugenommen, sobald es vollständig vom umgebenden Rechteck der bisherigen Stiftkurve umfasst wird (Typ Einkreisung, bei Stiftposition ° oberhalb von Schriftzeichen '+'). Bei der alternativen Methode 2 wird bezüglich der Durchstreichung ebenso verfahren, wobei nach jeder Selektion eines weiteren Segments nur noch die Teilkurve ab der betreffenden Position (°) betrachtet wird. Eine Selektion vom Typ Einkreisung findet dann statt, wenn das umschreibende Rechteck dieser Teilkurve ein Segment berührt (Position ° links oberhalb von '+'). Obwohl Methode 2 ähnlich gute Ergebnisse liefert, wird Methode 1 der Vorzug gegeben, da sie schreiberübergreifend etwas robuster funktioniert.

Die „Intelligenz“ der Stiftgestik besteht nun darin, dass mit jedem zusätzlichen Segment automatisch der kleinstmögliche semantisch geschlossene Teilausdruck ausgewählt wird. Am Beispiel in Abb. 7.3 bedeutet dies, dass bereits die Selektion von '+' die Hinzunahme der verbleibenden Segmente 'a' und 'n' bewirkt. Dabei wird der kleinste vollständige Teilast der Semantischen Gliederung ermittelt (Graustufe 3 in Abb. 7.3), der sowohl den vorher selektierten (Graustufe 1) als auch den hinzugekommenen Teilausdruck (Graustufe 2) umfasst. Mit dieser Maßnahme ist sichergestellt, dass nur solche Teilbereiche der Handschrifteingabe per Stiftgestik markiert werden können, deren Ersetzung (mittels Sprachkorrektur, siehe unten) unter Beibehaltung der restlichen Formelstruktur zulässig ist. Einerseits sind also damit die Möglichkeiten der stiftgestischen Referenzierung einge-

schränkt, aber andererseits wird so ohne Zutun des Benutzers dafür gesorgt, dass die Zusammenführung von Handschrift- und Spracheingabe auf einfache Weise erfolgen kann und nicht zu einem inkonsistenten Gesamtresultat führt.

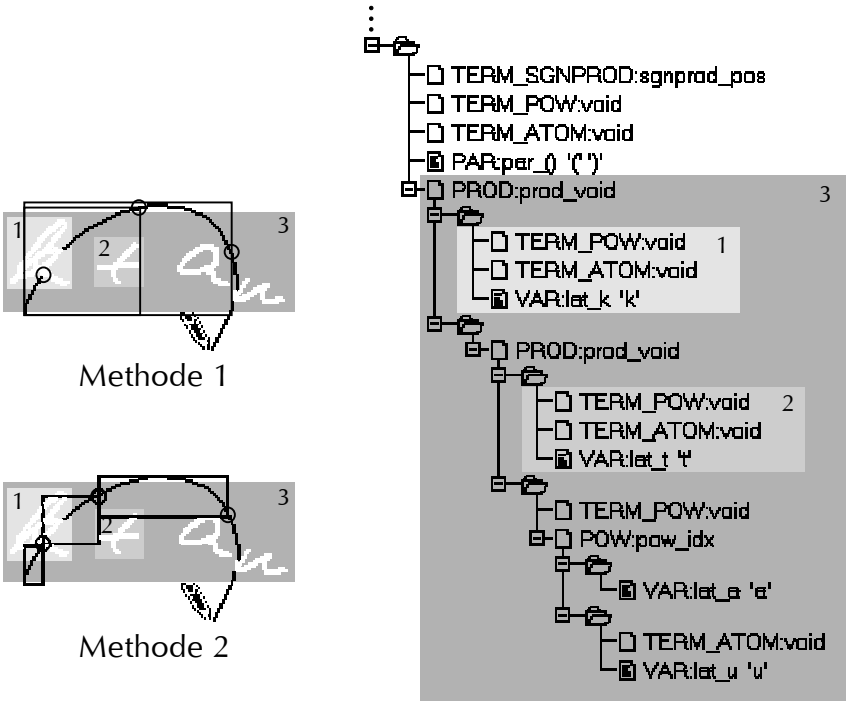


Abb. 7.3: Intelligente Stiftgestik zur Selektion semantisch geschlossener Teilausdrücke.

Am genannten Beispiel fällt auf, dass die Zuordnung der Handschriftsegmente zu entsprechenden Teilbereichen der Semantischen Gliederung eigentlich unsauber ist, da sie ja den fehlerhaften Abschnitt des Erkennungsergebnisses mit einschließt[28]. Der Sinn dieser Aktion besteht jedoch nicht in einer korrekten semantischen Zuordnung entsprechend der erfolgten Eingabe, sondern umgekehrt in der Benachrichtigung des Benutzers über die tatsächlich (aufgrund eines Erkennungsfehlers) entstandene Segmentierung des betreffenden Teilausdrucks. Der Benutzer wird dadurch (nach vorheriger Betrachtung des Erkennungsergebnisses) in die Lage versetzt, durch Stiftgestik zu ermitteln, in welchem Umfang eine Selektion und Ersetzung notwendig ist, um ein korrektes Gesamtergebnis zu erhalten. In der Praxis treten allerdings meist überschaubare Fälle (vergleichbar mit obigem Beispiel) auf, in denen die Korrelation zwischen Fehler und Niederschrift offensichtlich ist.

7.2.2 Sprachliche Modifikation

Mit dem Beginn einer stiftgestischen Interaktion wird automatisch der semantische Decoder zur Sprachverarbeitung gestartet, um den gesprochenen Korrekturausdruck zu identifizieren. Abb. 7.4

[28] Anmerkung: Dies betrifft nur den hier betrachteten Fall der Behebung eines Erkennungsfehlers. Eine stiftgestische Selektion ist aber in zwei weiteren (unkritischen) Fällen sinnvoll, nämlich erstens zum Korrigieren eines Schreibfehlers (richtige Erkennung einer falschen Teileingabe) und zweitens zur nachträglichen Modifikation (z.B. Erweiterung) eines Teilausdrucks bei ebenfalls korrekter Erkennung.

zeigt die Beispielgeste des vorigen Abschnitts im Kontext der Gesamtformel nach Vervollständigung der Handschrifteingabe an. Die gleichzeitige Spracheingabe 'k plus a Index n' soll den selektierten Teilausdruck (korrespondierend zum Teilergebnis $k\,t\,a_u$ aus Abb. 7.2) ersetzen, wofür eine Fusion der aus Handschrift- und Spracheingabe erhaltenen Semantischen Gliederungen erfolgt.

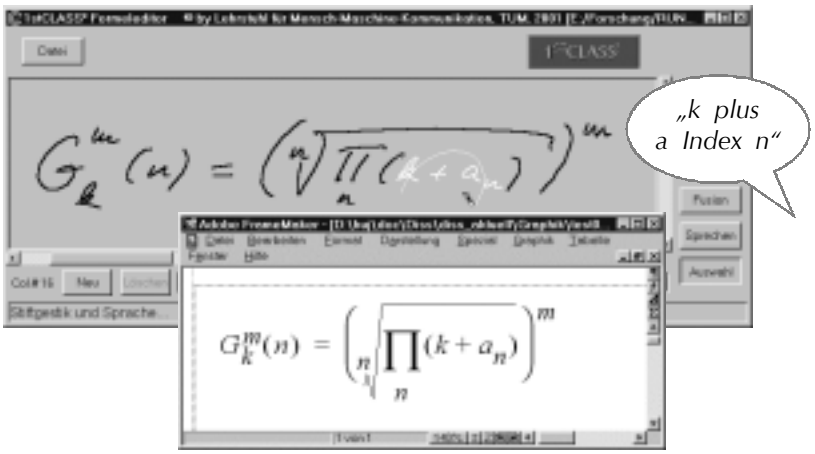


Abb. 7.4: Sprachkorrektur eines per Stiftgestik ausgewählten Teilausdrucks.

7.2.3 Datenfusion

Das Ziel der Datenfusion auf semantischer Ebene ist die konsistente Ersetzung eines mittels Stiftselektion identifizierten Teilastes der Semantischen Gliederung zur Handschrifteingabe durch den aus der Sprachkorrektur gewonnenen Bedeutungsinhalt. Als Konsistenzbedingung ist zu gewährleisten, dass der gesprochene Korrekturausdruck anstelle des zu ersetzenden Formelteils unter Beachtung des mathematischen Kontextes eingefügt werden kann, ohne dass dadurch die semantische Integrität der Formel verletzt wird.

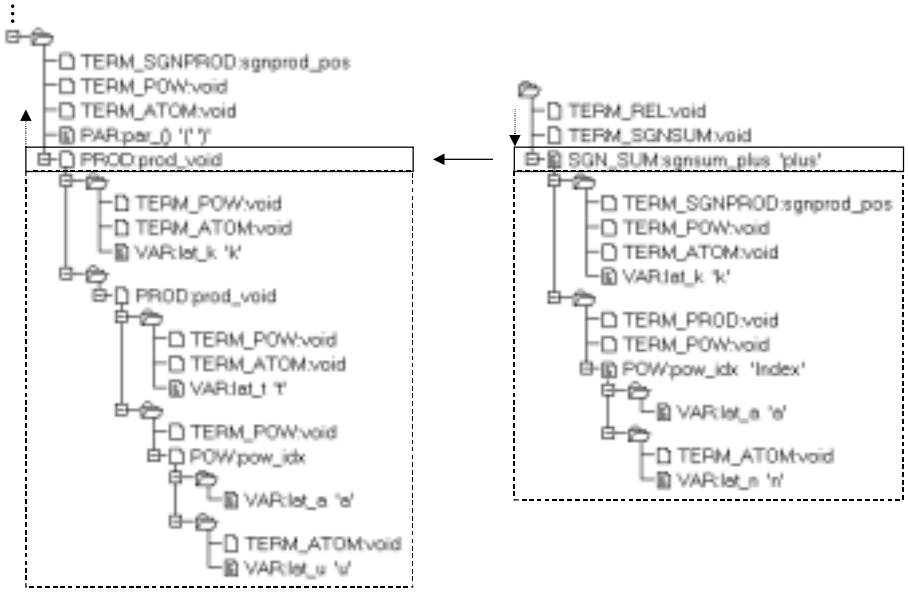


Abb. 7.5: Datenfusion zur Integration von Handschrift- und Spracheingabe.

Das semantische Modell dient vorrangig dazu, diese Konsistenz durch Festlegung aller erlaubten Nachfolgerscharen pro Semuntyp und -wert innerhalb der unterstützten mathematischen Domäne sicherzustellen. Daher besteht die erforderliche Konsistenzprüfung lediglich in der Auffindung eines geeigneten Paares von Semunen aus je einer der beiden betrachteten Semantischen Gliederungen, deren Austauschung (unter Mitnahme aller daran angebundenen Nachfolgersemune) mit den Vorgaben des semantischen Modells verträglich ist. Die Vorgehensweise ist in Abb. 7.5 am genannten Beispiel skizziert.

Nach Ausführung der Beispielgeste (Abb. 7.3) wird das in Abb. 7.5 links eingerahmte Semun PROD:prod_void als Wurzel des kleinstmöglichen semantisch geschlossenen Teilausdrucks (gestrichelter Rahmen) identifiziert. Die Semantische Gliederung zur Spracheingabe wird daraufhin absteigend sequentiell, bei ihrer Wurzel beginnend (vertikaler Pfeil in Abb. 7.5 rechts) nach einem Semuntyp durchsucht, der den betreffenden Semuntyp (hier: PROD) ersetzen kann. Hierzu wird auf die laut semantischem Modell möglichen Nachfolgerscharen des Vorgängersemuns (hier: PAR:par_(), siehe Anh. A.6, S. 111) zurückgegriffen. Die Suche endet spätestens beim ersten „syntaxtragenden“ Semun (-Symbol) bzw. vor der ersten Verzweigung in einen Teilast (-Symbol), damit tatsächlich der gesamte sprachliche Korrekturausdruck nach der Ersetzung in das Endergebnis eingeht.

Im Beispiel ist die Suche an der letztmöglichen Stelle (Semun SGN_SUM:sgnsum_plus) erfolgreich, da der Semuntyp SGN_SUM ebenso wie der zu ersetzende Typ PROD als Nachfolger von PAR:par_() vorgesehen ist. Im erfolglosen Fall würde die beschriebene Prozedur wiederholt, indem nunmehr aufsteigend sequentiell (vertikaler Pfeil links) die Vorgänger des selektierten Semuns als Ersetzungskandidaten überprüft werden. Dieser Vorgang wiederum endet spätestens *vor* dem ersten anzutreffenden syntaxtragenden Semun ( ) bzw. Verzweigungspunkt ( ), da sonst automatisch ein über den selektierten Teilausdruck hinausgehender Formelterm ersetzt werden würde. Dies wäre hier (nach Abb. 7.5) bereits beim direkten Vorgänger PAR:par_() der Fall, mit der Folge einer fälschlichen Substitution einschließlich der runden Klammern.

Der Benutzer ist somit selbst dafür verantwortlich, nur mathematisch sinnvolle Korrekturmaßnahmen zu treffen. Anderenfalls, etwa durch die Ansage 'k ist gleich a' im Beispiel, schlägt die gewünschte Datenfusion und damit der Korrekturversuch fehl.

7.3 Konventionelle Interaktion

Eine zusätzliche konventionelle Formelbearbeitung setzt die Übertragung des Erkennungsresultats in einen klassischen Formeleditor voraus. Die hier angebundene kommerzielle Textverarbeitungs-Software FrameMaker (mit FrameMath-Formeleditor) ist zugleich die Zielanwendung für die weitere Verwendung der erfassten Formeln.

7.3.1 Maker Interchange Format

Wie die Bezeichnung andeutet, steht mit MIF eine vom Hersteller ADOBE Systems Inc. definierte Schnittstelle für den beiderseitigen Datenaustausch zwischen FrameMaker und anderen Applikationen zur Verfügung. Es handelt sich dabei um eine standardisierte, hierarchische Markensprache

(prinzipiell vergleichbar mit gängigen Standards zur Dokumentenbeschreibung wie SGML oder XML), mit der sowohl die Struktur als auch der Inhalt von FrameMaker-Dokumenten beschrieben werden können. Ein in diesem Format erstelltes Dokument ist eine herkömmliche ASCII-Textdatei. Die genaue Spezifikation des Maker Interchange Formats ist aus [ADO97] zu entnehmen, insbesondere die Syntax zur Codierung mathematischer Formeln (S. 183 ff., „MIF Equation Statements“).

Während die Interpretation und Darstellung einer MIF-Datei als FrameMaker-Dokument sowie die MIF-Erzeugung in umgekehrter Richtung innerhalb von FrameMaker stattfindet, muss der Austausch zur anzubindenden Applikation in beide Richtungen unter Beachtung der MIF-Spezifikation extern realisiert werden. Die beiden folgenden Abschnitte betreffen die Transformation Semantischer Gliederungen in MIF-Dateien und zurück.

7.3.2 MIF-Transformation

Im Gegensatz zu manchen meist textsatzorientierten Markensprachen (z.B. LaTeX) wird eine mathematische Formel *F* in MIF grundsätzlich streng hierarchisch repräsentiert. Dies geschieht allgemein in der Form

$$F := `Op\,[\,*Form\,*\,Arglst\,]' \qquad Arglst := Arg\,\{,\,Arglst\} \qquad Arg := Val \mid Op\,[\,*Form\,*\,Arglst\,],$$

wobei die eckigen Klammern '[]' als wichtigste Syntax-Marken die Argumentliste *Arglst* jedes auftretenden Operators *Op* umrahmen, während die geschweiften Klammern '{}' hier nur die optionale Aneinanderreihung mehrerer Argumente *Arg* andeuten ('{}' sind kein Bestandteil der Syntax). Je nach Schachtelungstiefe treten als Argumente neben Werten *Val* (in der Regel alphanumerische Größen) weitere Operatoren mit zugehörigen Argumentlisten auf. Diese beiden Fälle sind oben durch '|' getrennt ('|' ist ebenfalls kein Syntaxbestandteil). Spezielle Formatierungs-Anweisungen *Form* werden, eingerahmt in '(* *)', der Argumentliste eines Operators vorangestellt.

Die hierarchische Form ist einerseits Voraussetzung für die FrameMaker-interne Formelinterpretation zum Zwecke der Weiterverarbeitung (Modifikation, Umformung oder Kalkulation), andererseits lässt sie sich dementsprechend konsequent in das ebenfalls hierarchische Format der Semantischen Gliederung (und umgekehrt) transformieren. Für die MIF-Transformation wurde eine modulare Programmstruktur gewählt, bei der die vorliegende Semantische Gliederung sequentiell (absteigend im Sinne der Darstellung in Abb. 7.6) verarbeitet wird [MAY00][TON02]. Durch rekursiven Aufruf der Hauptprozedur *master* des Transformationsmoduls S2MIF (vgl. Kap. 8.2) zur Erzeugung der zu jedem Semun korrespondierenden MIF-Syntax ergibt sich schließlich eine Zeichenkette, die die gesamte Formel repräsentiert. Zu jedem syntaxtragenden Semuntyp existiert hierfür eine Prozedur gleichen Namens (z.B. Prozedur REL), die den entsprechenden MIF-Ausdruck (z.B. 'equal [?,?]') in Abhängigkeit vom Semunwert (z.B. rel_equ) produziert. Eine Anpassung des bestehenden Funktionsumfangs, sei es zur Erweiterung vorhandener oder zur Hinzufügung neuer Formelbestandteile (Semuntypen), ist aufgrund des modularen Aufbaus in systematischer Weise möglich. In Anh. A.9 sind alle derzeit unterstützten Formelkomponenten in beiden Formaten gegenübergestellt. Die MIF-Notation 'newlimit[(*T"min"T*)?,?]' bzw. 'newlimit[(*T"max"T*)?,?]' zur Beschreibung von Minimum- und Maximum-Funktion (Semuntyp FNIDX) verweist auf zwei

Funktions-Templates, die nach dem Vorbild der Limes-Funktion eigens definiert wurden, da min und max nicht zum Standard-Funktionsumfang von FrameMath gehören.

Abb. 7.6 zeigt das Ergebnis der Datentransformation für das durch Datenfusion entstandene Endresultat der Beispielformel, wie es sodann gemäß Abb. 7.4 in FrameMaker visualisiert wird. Das MIF-Dokument beginnt mit einem Vorspann, der die nötigen Versionsangaben sowie allgemeine Dokumenteigenschaften und Formatierungsanweisungen enthält. Hier wird auf Standardeinstellungen zurückgegriffen, ebenso wie bei den durch den Marker '<Math' eingeleiteten Positionierungsangaben für die darzustellende Formel. Die eigentliche Formelspezifikation schließt an den Marker '<MathFullForm' an, hier beginnend mit dem übergeordneten Operator 'equal[' (Gleichung), gefolgt von den Formelstrukturen für die linke und rechte Gleichungsseite. Zur Unterdrückung der automatischen Einklammerung von Teilausdrücken (FrameMath-Standardeinstellung) ist bei einigen Operatoren die Formatanweisung '(*n*)' angegeben.


```
<MIFFile 5.50>
# Generated by FrameMaker 5.5.2L11a
# Options:
                    .
                    .
                    .
  <Math
   <Unique 998363>
   <RunaroundGap  0.0 pt>
   <BRect  7.64729 cm 0.89958 cm
    0.28209 cm 0.52917 cm>
   <MathFullForm
`
equal[
#         LINKE SEITE DER GLEICHUNG
#         ========================
 times[(*n*)
  power[
   indexes[
    0,1,char[G],char[k]],char[m]
   ],
   id[comma[char[n]]]
  ],
#        RECHTE SEITE DER GLEICHUNG
#        ========================
 power[
  id[sqrt[
   prod[(*n*)
    id[plus[(*n*)
     char[k],indexes[
      0,1,char[a],char[n]
     ]
    ]],
    char[n]
   ],
   char[n]]],
  char[m]
 ]
]
'


   > # end of MathFullForm
   <MathLineBreak  352.77781 cm>
   <MathOrigin  7.78833 cm 1.27 cm>
   <MathAlignment Center>
   <MathSize MathLarge>
  > # end of Math
                    .
                    .
                    .
# End of MIFFile
```

Abb. 7.6: Semantische Gliederung (links) und Maker Interchange Format (rechts) am Beispiel.

Wird diese MIF-Datei in FrameMaker geöffnet, so erscheint die Ergebnisformel gemäß Abb. 7.4 und kann nun unter Einsatz der vollen FrameMath-Funktionalität dort weiterverwendet werden. Es besteht also insbesondere nach wie vor Zugriff auf die Formelstruktur, ganz so, als wenn die Formel konventionell in FrameMaker erstellt worden wäre. Für die Auswahl von Teilausdrücken mit Maus und Tastatur gelten dabei ganz ähnliche Prinzipien wie bei der in Kap. 7.2.1 erläuterten Stiftgestik (semantisch geschlossene Selektion), auch hier mit der Motivation, nur strukturell konsistente Modifikationen zuzulassen. Aus Benutzersicht ist dies im Sinne eines durchgängigen Bedienschemas als Vorteil zu werten.

7.3.3 MIF-Rücktransformation

In Abb. 7.7 ist dargestellt, wie eine Weiterbearbeitung der Beispielformel aussehen könnte. Nach einer mathematischen Umformung der rechten Gleichungsseite, die in FrameMaker durch Sonderfunktionen unterstützt wird, wurde der Ausdruck G_k^m auf der linken Seite markiert und entfernt, so dass ein '?' als Platzhalter für eine fehlende Strukturkomponente angezeigt wird. Möchte der Anwender diesen Status in die Handschriftumgebung zurückleiten, um dort beispielsweise einen Ersatzausdruck für die aktuelle Markierung niederzuschreiben, besteht die Möglichkeit, eine Rücktransformation der (geänderten) MIF-Repräsentation in das Format der Semantischen Gliederung zu starten. Analog zur Hintransformation erfolgt dies wiederum durch sequentielle Abarbeitung und eine rekursive Ablaufstruktur, wobei in diesem Fall für jeden FrameMath-Operator eine gleichnamige Prozedur zur Produktion des äquivalenten Syntaxabschnittes dient [TON02].

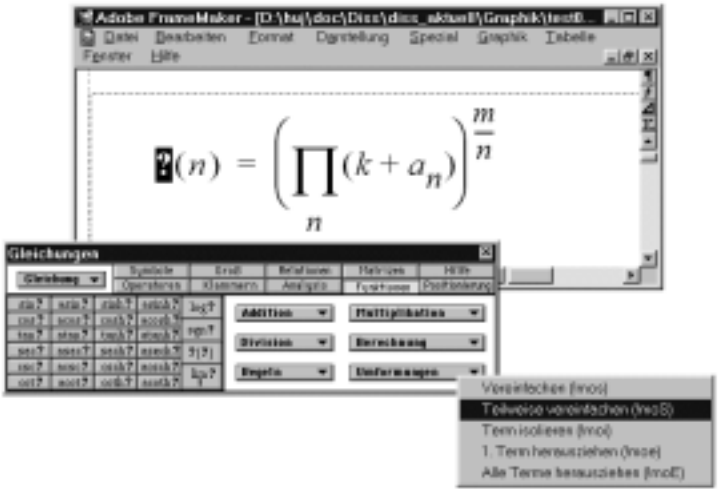


Abb. 7.7: Konventionelle Interaktion im FrameMath-Formeleditor.

Bei unvollständigen Formeln (siehe Beispiel) weist die MIF-Notation einen '?'-Platzhalter durch den Pseudo-Operator 'prompt[]' aus, der bei der Rücktransformation in den speziellen Semuntyp 'LEER' überführt wird. Auf diese Weise kann ein Querbezug zwischen einer erneuten Handschrifteingabe und einem fehlenden bzw. zu ergänzenden Formelteil hergestellt werden.

Die Rücktransformation kommt außerdem bei der akustischen Ausgabe von FrameMaker-Formeln zum Einsatz (siehe Kap. 7.5).

7.4 Parallele Handschrift- und Spracheingabe

Die oben behandelte *alternierende* Interaktion (vgl. Abb. 2.2) basiert auf einer vorwiegend handschriftlichen Formelerfassung, die durch serielle Korrektur- bzw. Modifikationsmöglichkeiten mittels Stiftgestik und Sprache oder auch Maus und Tastatur ergänzt wird. Beide Varianten werden nach den Ergebnissen der Benutzerstudie (Kap. 6.5) vom Anwenderkreis als komfortabel und aufgabenangemessen bewertet.

Unter Beibehaltung natürlicher Handschrift als allgemein akzeptiertem Haupteingabeverfahren liegt es nahe, das parallele Schreiben und Sprechen von Einzelsymbolen, Teilausdrücken oder gar einer gesamten Formel in das Bedienkonzept zu integrieren, was nach Abb. 2.2 als *synergistischer* Modus einzustufen ist. Für die Auswertung einer solchen bimodalen Interaktion ist eine Datenfusion sowohl auf signalnaher als auch auf semantischer Ebene denkbar. In beiden Fällen ist es von Vorteil, dass das Auftreten von Mehrdeutigkeiten in Handschrift oder Sprache oft in ganz unterschiedlichen Situationen zu erwarten ist. Allein bei den verschiedenen alphanumerischen Bezeichnern existiert eine Vielzahl von Paaren, die entweder in der einen oder der anderen Modalität (selten in beiden) leicht verwechselt werden können. Um nur einige Beispiele zu nennen (von benutzerspezifischen Fällen abgesehen):

Handschrift: *w* | ω ↔ 've:' | 'Om@ga' φ | 9 ↔ 'fi:' | 'nOYn' 0 | *O* ↔ 'nUl' | 'o:'

Sprache: 'be:' | 'de:' ↔ *b* | *d* 'EpsIlOn' | 'YpsIlOn' ↔ ε | *y* 'my:' | 'ny:' ↔ μ | ν

Über den Einzelzeichenaspekt hinaus kann es zu strukturellen Fehlentscheidungen kommen, wenn solche Mehrdeutigkeiten im Zusammenhang mit Operatorbezeichnern bestehen. Auch diese Fälle sind in Handschrift und Sprache recht verschieden gelagert, ebenso wie Mehrdeutigkeiten aufgrund der Schriftzeichenpositionierung, für die es in der Sprache oftmals signifikante Schlüsselworte gibt. Beispiele:

Handschrift: $\int$ | *S* ↔ 'Int@gra:l' | 'gro:sEs' $?_?$ | $\overset{?}{?}$ ↔ 'IndEks' | 'y:b6'

Sprache: 'gama:' | 'klam6' ↔ γ | ('mi:nUs' | 'si:nUs' ↔ - | *sin*

Natürlich ist die hier angesprochene Verschiedenheit von Handschrift- und Sprachmerkmalen in gleichem Maße bei der alternierenden Korrektur von Nutzen, wie die Evaluierung in Kap. 9.2 zeigt. Bei einer Simultaneingabe könnte sie sich hingegen zusätzlich präventiv, also zur frühzeitigen Vermeidung bestimmter Erkennungsfehler, auswirken. Die äußerst geringe Akzeptanz dieser Interaktionsform im Benutzertest (Kap. 6.5) lässt darauf schließen, dass zumindest Neulinge im Umgang mit einem handschrift- und sprachbasierten Formeleditor sich lieber auf das Schreiben alleine konzentrieren, um dann – nur bei tatsächlich auftretenden Fehlern – auf Sprache zu Nachbesserungszwecken zurückzugreifen. Die Unterstützung der parallelen Interaktion beschränkt sich daher im Rahmen dieser Arbeit auf eine exemplarische Umsetzung zur regelbasierten Disambiguierung von Einzelzeichenfehlern und wurde nicht in die Evaluierung des Gesamtsystems (Kap. 9) einbezogen.

7.5 Sprachproduktion

Die Semantische Gliederung liefert eine zeitfreie, medienneutrale Repräsentation des mathematischen Inhalts einer gegebenen Formel. Sie kann damit als *Intermedia*-Ebene dazu genutzt werden, eine automatische Transformation von einer Modalität zur anderen vorzunehmen. Fasst man innerhalb eines sprachverstehenden Systems verschiedene natürliche Muttersprachen als getrennte Modalitäten auf, so ist die maschinelle Sprachübersetzung [MÜL99] ein Spezialfall einer solchen Transformation, für den man neben einem gemeinsamen semantischen Modell je einen Satz syntaktischer Modelle (Übergangs- und Elementmodelle) für Quell- und Zielsprache benötigt.

Handgeschriebene Formeln. Um eine handgeschriebene Formel in gesprochener Form wiedergeben zu können, benötigt man zunächst die zugehörige Semantische Gliederung, die bei korrekter Erkennung als Decodierungsergebnis vorliegt. Zusätzlich zur MIF-Transformation (visuelle Ausgabe) ist nun eine *Sprachproduktion* möglich, indem man den Vorgang der semantischen Decodierung unter Verwendung des syntaktischen Modells für gesprochene Sprache in den inversen Vorgang der syntaktischen Codierung (ohne Einbeziehung der signalnahen Ebenen) umkehrt. Beschränkt man diese Aufgabe darauf, den laut syntaktischem Modell wahrscheinlichsten Gesamtausdruck Ξ_w (hier: Wortfolge) zur gegebenen Semantischen Gliederung S zu produzieren, dann ist der syntaktische Anteil $Sc(\Xi|S)$ der Kostenfunktion $K(O,\Xi,S)$ nach Gl. 4.36 hinsichtlich Ξ zu minimieren. Nach Gl. 4.24 gilt also[29]:

$$\Xi_w = \underset{\Xi}{\operatorname{argmin}}\, Sc(\Xi|S) = \underset{\Xi}{\operatorname{argmin}} \sum_{n=1}^{N} \left\{ \sum_{l=1}^{Y} Sc(\beta_{nl}) \right\} + Sc(\gamma_n) + \sum_{az \in Pfad(SM_n)} Sc(\delta_n^{az}) \Bigg\} \qquad (7.1)$$

Der zur Projektion der semantischen auf die syntaktische Repräsentationsebene dienende Mechanismus, nämlich das vollständige Durchlaufen des Syntaktischen Netzwerks *SN* gemäß Übergangs- und Elementmodell, wird nun unter dieser Randbedingung zur Sprachproduktion eingesetzt, so dass die gewünschte Formel unter Verwendung der häufigsten Formulierungen in Kombination mit der bevorzugten Sprechreihenfolge in textueller Form resultiert.

Typographische Formeln. Neben handschriftlichen Eingaben kommen auch konventionell erstellte, also in typographischer Form vorliegende Formeln als Quelle für die Sprachproduktion in Betracht. Mithilfe der MIF-Rücktransformation (Kap. 7.3.3) erhält man die zu einer in FrameMaker gesetzten Formel korrespondierende Semantische Gliederung, woraufhin man ebenso verfährt wie im vorigen Absatz beschrieben.

Prosodische Nachbearbeitung. Um den bei gesprochenen Formeln vielfach auftretenden Mehrdeutigkeiten bezüglich der Formelstruktur (siehe Kap. 9.1.1) entgegenzuwirken, werden die nach Gl. 7.1 produzierten Ausdrücke durch zwei Arten von Prosodie-Elementen ergänzt:

- Stimmanhebung mit kurzer Sprechpause (Komma-Effekt),

[29] Da es sich um sprachliche Ausdrücke handelt, entfallen hier die Offsetanteile am Gesamtscore.

↘ Stimmabsenkung mit Sprechpause (Punkt-Effekt).

Eine deutliche Verbesserung der Verständlichkeit wird – nach Überprüfung an den Handschrift-Referenzformeln nach Anh. A.2 – erreicht, indem man diese Elemente an folgenden Pfadstellen des Syntaktischen Netzwerks einfügt [MÜL02]:

↗ nach dem Endknoten jedes Syntaktischen Moduls, das zu einem Semun an der jeweils letzten Verzweigung der Semantischen Gliederung vor einem Teilastende korrespondiert,

↘ nach einem (+)Elementknoten jedes SMs, das zu einem Semun mit einem Abstand $d \leq 6$ vom Wurzelsemun korrespondiert.

Diese Maßnahme entspricht einer prosodischen Abgrenzung von Grobstrukturkomponenten (↘) sowie der innersten noch strukturtragenden Formelbestandteile (↗). Für die Beispielformel von S. 75 (Semantische Gliederung gemäß Abb. 7.6) erhält man damit als sprachliche Realisierung:

„ groß g index k hoch m ↗ von ↘ runde klammer auf n runde klammer zu ↗ ↘ ist gleich ↘ klammer auf n-te wurzel aus produkt über n aus klammer auf k plus a n ↗ klammer zu ↗ klammer zu hoch ↘ m ↘ “

An dem Teilausdruck 'a n' (für a_n) erkennt man, dass die laut Elementmodell häufigste Formulierung nicht immer auch die verständlichste Variante ist: Die Bezeichnung 'Index' anstelle der Auslassung (Element '-', siehe Anh. A.7.2) wäre hier vorteilhafter, tritt aber im Trainingskorpus etwas seltener auf. Eine Einbeziehung weniger häufiger Sprachmuster aus Übergangs- und Elementmodell wäre auch sinnvoll, um die Sprachausgabe abwechslungsreicher zu gestalten.

Im Gegensatz zu anderen Anwendungsdomänen ist eine linguistische Nachbearbeitung zur Berücksichtigung grammatikalischer Flexionsformen (ob deklinativ oder konjugativ) für gesprochene Formeln nicht erforderlich, da der hier übliche Sprachgebrauch flexionsfrei ist.

Sprachausgabe. Nach der Produktion eines sprachlichen Formelausdrucks in der oben angegebenen Textform wird dieser in eine phonetische Darstellung überführt und mittels einer korpusbasierten, konkatenativen Sprachsynthese (siehe Kap. 8.2) über die PC-Soundkarte akustisch wiedergegeben. Der Benutzer erhält so zusätzlich zur Visualisierung ein auditives Feedback über das Resultat der erfolgten Eingabe.

Mit der Übersetzung handgeschriebener in gesprochene Formeln wird die Fähigkeit zur Informations-Transformation zwischen diesen beiden Kanälen als Charakteristikum eines multimodalen Systems (siehe S. 12) demonstriert. Bezogen auf die konventionelle Eingabe als Quelle der Sprachproduktion besteht damit die Möglichkeit, ein inhaltlich strukturiertes Dokument aus einer Bild-/Textverarbeitungsumgebung (hier FrameMaker) im Sinne einer „Vorlesefunktion“ sprachlich auszugeben, wie sie im Bereich des *document voice read-back* zur Wiedergabe von nicht-textuellen Dokumentabschnitten Gegenstand der aktuellen Forschung ist.

8

Implementierung

8.1 Plattform

Die prototypische Umsetzung des Gesamtsystems erfolgte für den Einzelplatzbetrieb unter dem Betriebssystem MICROSOFT Windows NT 4.0. Als Entwicklungsumgebungen kamen in erster Linie MICROSOFT Visual C++ 6.0 für überwiegend algorithmische und rechenintensivere Komponenten sowie die Skriptsprache Tcl/Tk[30] 8.3 für oberflächennahe Module zum Einsatz.

8.2 Software

Abb. 8.1 zeigt den Informationsfluss zwischen den verschiedenen Systemkomponenten im Überblick. Die zentrale Ablaufsteuerung ist innerhalb der Hauptapplikation *1ST CLASS² FORMULA²* in Verbindung mit dem Datenfusionsmodul angesiedelt[31]. Hier werden Benutzerinteraktionen registriert und nach Bedarf Anfragen an den Semantischen Decoder zur Handschrift- bzw. Sprachanalyse gestartet. Intern werden ggf. Stiftgestik-Ereignisse ausgewertet, um die Decodierungsergebnisse für Handschrift und Sprache zueinander in Bezug zu setzen und per Datenfusion zu kombinieren. Die Datenkommunikation erfolgt beidseitig über INET-Socketverbindungen. Das zur Decodierung benötigte Modellwissen steht durch Zugriff auf die extern verfügbaren statischen Wissensbasen zur Verfügung.

Nach jeder abgeschlossenen natürlichen Interaktion (auf Wunsch auch zur Zwischenkontrolle) wird das Gesamtergebnis vom Transformationsmodul *S2MIF* an ADOBE FrameMaker 5.5 weitergeleitet, dort visualisiert und ggf. konventionell weiterbearbeitet. Nach Rücktransformation in *MIF2S* (zwecks weiterer sprach-/handschriftlicher Modifikation) ist eine akustische Formelausgabe (TTS, *Text to Speech*) über das Transformationsmodul *S2SPCH* mit angebundener Sprachsynthese mög-

---

[30] Siehe http://www.scriptics.com/ oder http://tcl.activestate.com/ .

[31] 1ST CLASS² (siehe S. 21) FOR MULtimodal Acquisition of mathematical FORMULAs

lich. Dabei werden die frei verfügbaren Module *TEXT2PHO* [POR00] und *MBROLI* [DUT96][MBR02] eingesetzt. Die Sprachwiedergabe kann auch direkt von der Ausgabe des Datenfusionsmoduls angesteuert werden.

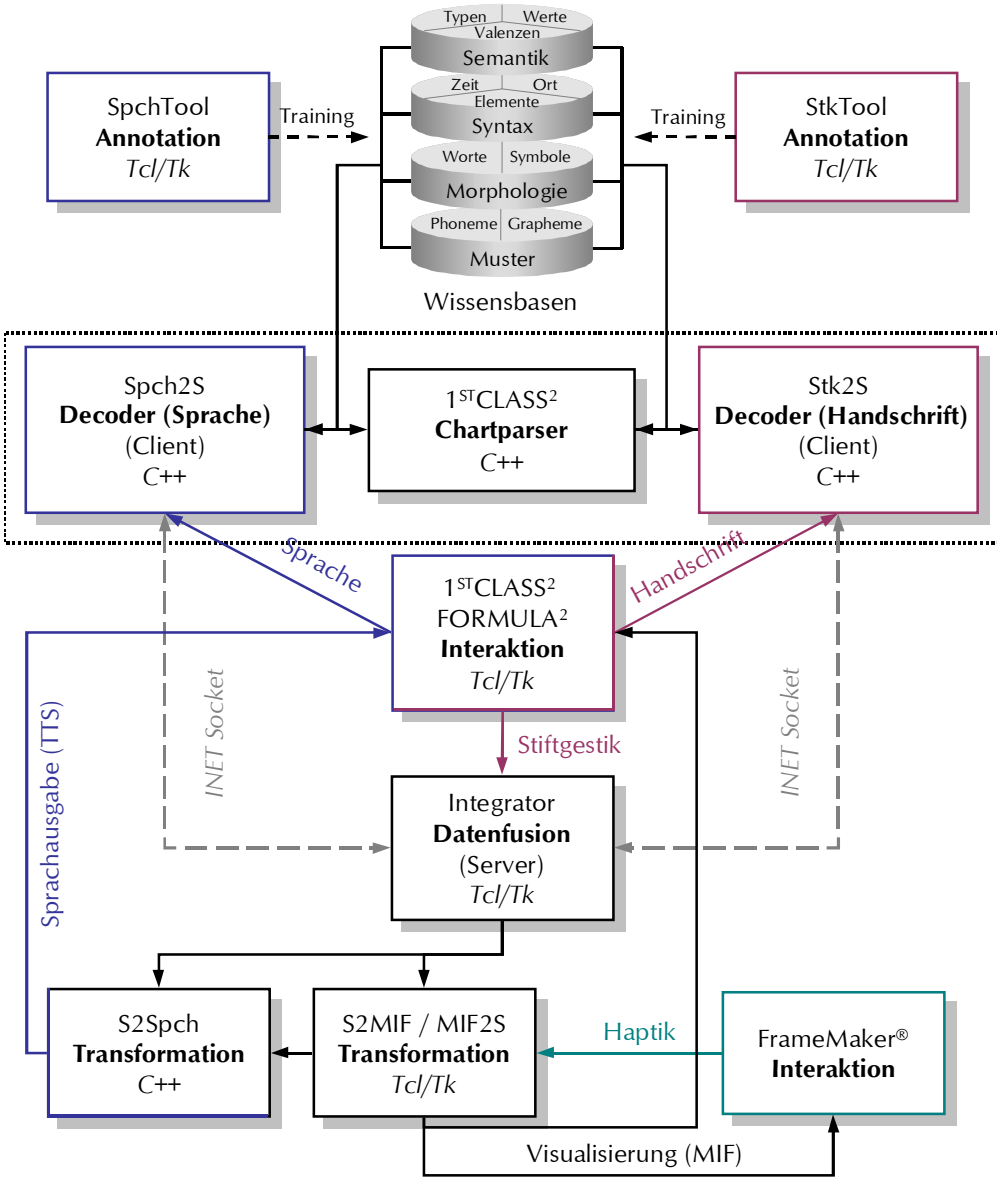


Abb. 8.1: Logisches Schema der Systemimplementierung.

9

Ergebnisse

9.1 Exklusiver Modus

9.1.1 Sprache

Die Extraktion des mathematischen Inhalts aus vollständig gesprochenen Formeln ist mit einigen Hindernissen verbunden, die sich wie folgt zusammenfassen lassen:

- **Struktureller Bereich.** Insbesondere auf der Skala von Teilausdrücken führen die Beschränkungen der natürlichen Sprache zu Mehrdeutigkeiten hinsichtlich der Rangordnung enthaltener Operatoren. Beispielsweise kann die Phrase 'a plus b geteilt durch c' die Terme $\frac{a+b}{c}$ bzw. $a+\frac{b}{c}$ repräsentieren. Diesem praktisch in jeder Formel auftretenden Sachverhalt steht allerdings die Beobachtung entgegen, dass die Erkennung der grundsätzlichen Formelstruktur im Ganzen betrachtet aufgrund des hier verfolgten Ansatzes gegenüber solchen „inneren“ Strukturfehlern weitgehend robust ist.

- **Symbolischer Bereich.** Mathematische Formeln bestehen typischerweise zu einem erheblichen Anteil aus alphabetischen Symbolen (Bezeichner für Variable oder Konstante). Da deren gesprochene Form oft nur aus einem oder zwei Phonemen besteht, kommt es dementsprechend häufig zur Verwechslung oder gar Auslassung und irrtümlichen Einfügung solcher Symbole im Erkennungsergebnis. Während die letzteren beiden Fälle durch die erwartungsgetriebene, konsistenzerhaltende Decodierung nur begrenzt (je nach Kontext) auftreten, verbleibt die Verwechslung alphabetischer Symbole als vorherrschende Fehlerquelle bei der Klassifikation umfangreicher gesprochener Formeln.

Evaluierung. Nachdem die Modelle für den Sprachbereich auf komplett gesprochene Formeln zurückgehen (Kap. 5.2), wurde auch eine Auswertung dieses Eingabemodus auf der Basis vollständiger Formeln vorgenommen. Die quantitative Bewertung der sprecherunabhängigen Erkennungsleistung nach Tab. 9.1 bezieht sich auf die Reklassifikation aller Trainingsäußerungen sowie auf

einen Fremdtest an insgesamt 72 Datensätzen (unabhängige Testformeln nach [WIN97B], S. 197, zwei Fremdsprecher).

	Erkennungsrate (gesprochene Formeln)		
	Semune	Formel	Struktur
Reklassifikation	95,2 %	42,3 %	76,2 %
Fremdtest	93,6 %	22,2 %	61,1 %

Tab. 9.1: Erkennungsrate bei vollständig gesprochenen Formeln (sprecherunabhängig). Die erste Spalte bezieht sich auf die Semunebene (Anteil der insgesamt richtig klassifizierten Semune), die zweite Spalte gibt den Anteil der fehlerfrei erkannten Formeln an, und die dritte Spalte schließt die Tolerierung falsch erkannter Variablenbezeichner bei korrekter Formelstruktur ein.

Aus dem relativ hohen Anteil richtig erkannter Semune folgt, dass der Fehlerumfang, also das Verhältnis von fehlerhaft zu korrekt klassifizierten Formelbestandteilen, im Mittel unter 7 % liegt. Darin sind allerdings diejenigen Strukturfehler unberücksichtigt, die durch eine fehlerhafte Verknüpfung korrekter Semune, also durch eine verfälschte Nachfolgerstruktur entstehen. Der Ausdruck $\sqrt{2\pi}$ (…ROOT→…(NAT_NUM…←PROD→…VAR)) würde z.B. bei der abweichenden Zuordnung NAT_NUM…←PROD→…ROOT…→…VAR als $2\sqrt{\pi}$ dargestellt, obwohl alle enthaltenen Semune richtig klassifiziert wurden. Die erste Spalte in Tab. 9.1 liefert daher nur einen Anhaltspunkt für die durchschnittliche Fehlerhaftigkeit über alle Datensätze, für die in Kap. 9.1.2 zur Evaluierung des Handschriftmodus der sog. Erkennungsgrad als aussagekräftigeres Maß herangezogen wird.

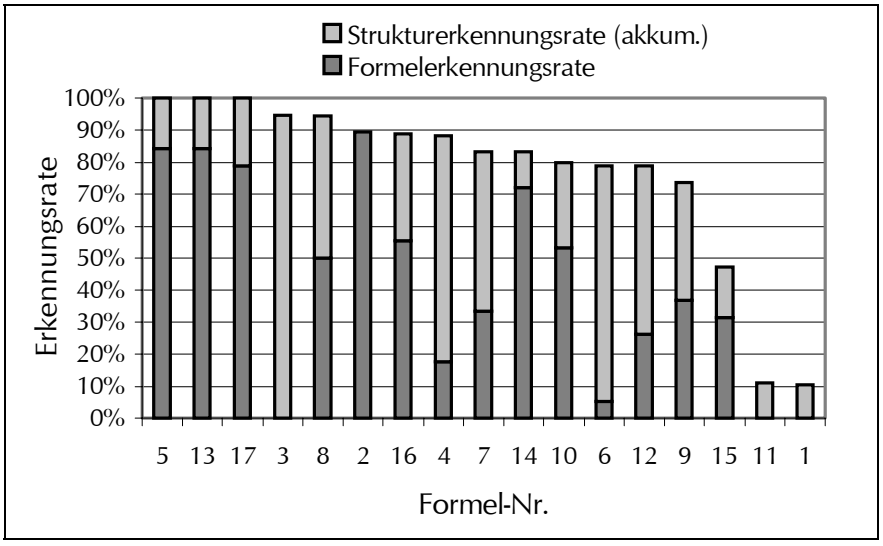


Abb. 9.1: Formelspezifische Erkennungsleistung (Reklassifikation). Die Formel-Nummern beziehen sich auf Anh. A.4 und sind nach Gesamterkennungsrate geordnet (Tolerierung von Variablenfehlern). Insgesamt nimmt der Anteil von alphanumerischen Bezeichnern und die Schachtelungstiefe nach rechts hin zu, während der Anteil von Operatorenbezeichnern abnimmt.

Als Folge der erwähnten Mehrdeutigkeiten in Kombination mit alphabetischen Fehlern wurde im Fremdtest nur etwa jede fünfte Formel fehlerfrei klassifiziert (zweite Spalte in Tab. 9.1). Mehr als

die Hälfte aller Formeln wurde hingegen strukturrichtig erkannt, wie die Auswertung unter Vernachlässigung einfacher Buchstabenfehler (dritte Spalte) zeigt[32]. Abb. 9.1 schlüsselt die formelbezogenen Erkennungsraten nach den einzelnen Referenzformeln gemäß Anh. A.4 (Reklassifikation) auf. Daraus geht hervor, dass insgesamt ein geringerer Anteil von alphanumerischen Bezeichnern und Operatorschachtelungen gegenüber sprachlich gut identifizierbaren Operatorbezeichnungen (z.B. 'Integral', 'Maximum') zu deutlich besseren Ergebnissen führt als der umgekehrte Fall. Dieser Umstand bestätigt die eingangs genannte Dominanz von alphabetischen und Rangordnungs-Fehlern.

Die Evaluierung nach vollständig gesprochenen Formeln ist lediglich als Zwischenergebnis anzusehen, da das umgesetzte Bedienkonzept Sprache als Korrekturmedium zur Modifikation lokal begrenzter Teilausdrücke vorsieht. Die dementsprechend kompakteren Äußerungen führen – im Vergleich mit den hier bewerteten Spracheingaben von durchschnittlich 40 Worten – zu wesentlich zuverlässigeren Ergebnissen (siehe Kap. 9.2).

9.1.2 Handschrift

Zur Auswertung der Erkennungsleistung bei komplett handgeschriebenen Formeln wurde das System nach der in Kap. 5 beschriebenen Vorgehensweise auf vier Benutzer (B1-B4) mit möglichst unterschiedlichem Schreibverhalten trainiert. Die Benutzerabhängigkeit beschränkt sich dabei auf Offset- und Schriftzeichenmodellierung und ist folgendermaßen begründet:

- **Offsets.** Die verwendete kontextfreie Modellierung durch einfache Normalverteilungen erweist sich bei allen beobachteten Schreibstilen als sehr zuverlässig und ausreichend diskriminativ. Beide enthaltenen Stilmittel, Positionierung und Skalierung, sind jedoch einem deutlichen Einfluss des Handschrifttypus (z.B. Schriftneigung und Schreibpräzision) unterworfen, so dass es bei Zusammenführung der Modelle für mehrere Schreiber schnell zu Klassenüberschneidungen kommt.
- **Schriftzeichen.** Die zu klassifizierenden handgeschriebenen Symbole sind größtenteils sehr einfach strukturiert und daher im Prinzip auch leicht zu modellieren. Ausschlaggebend für eine funktionierende Musterbewertung ist jedoch die Unterscheidbarkeit solcher einfachen Muster untereinander, so dass gerade die Ausnutzung der Schreibrichtungsinformation zusammen mit der Linienzugabfolge (Online-Merkmale) wertvolle Information hierfür liefert. Vergleicht man nun die auftretenden Symbolschreibweisen mehrerer Benutzer, dann ist zunächst eine beträchtliche Vielfalt bereits hinsichtlich des Schriftbildes zu beobachten, die unmittelbar zur Fehlzuordnung bestimmter Schriftzeichen von einem Benutzer zum anderen (und umgekehrt) führen kann. So verwenden einige Schreiber einen „Schreibschrift“-(Kursiv-)Stil für manche oder auch alle alphabetischen Zeichen (etwa 'x'), der eine Verwechslung mit druckschriftartigen Symbolen anderer Schreiber (hier z.B. 'κ') auslöst. Noch weitreichender sind die Abweichungen hinsichtlich der zeitlichen Abfolge, selbst bei äußerlich gleichartigem Schreibstil: Für die typischen

[32] Anmerkung: Die Sprachqualität variiert von Sprecher zu Sprecher, so dass die sprecherspezifische Strukturerkennungsrate bis zu 90 % erreicht.

Druckschreibweisen der Kleinbuchstaben 'x' und 't' sowie des '+'-Symbols existieren je acht mögliche Kombinationen aus Linienzugabfolge und Schreibrichtung, die zwar pro Schreiber in der Praxis auf je ein bis zwei Varianten reduziert (und damit in der Regel gut trennbar) sind, bei schreiberunabhängiger Betrachtung aber bewirken, dass diese Symbolklassen untereinander in Konflikt geraten[33].

Evaluierung. In Abb. 9.2 sind pro Benutzer zwei Beispieldatensätze aus dem Testkorpus dargestellt, der insgesamt 96 Niederschriften umfasst (Quelle: [RÅD97]). Die rechts angegebenen Erkennungsergebnisse weisen einige der systemtypischen Symbol- und Strukturfehler auf, die meist innerhalb lokal begrenzter Teilausdrücke auftreten. Der Strukturfehler bei Benutzer B1 ist darauf zurückzuführen, dass im Referenzformelbestand (siehe Anh. A.2) Binomialkoeffizienten unterrepräsentiert sind; ein entsprechendes Nachtraining des Offsetmodells wäre hier sinnvoll. Dagegen sind die bei den Benutzern B2 und B3 (jeweils zweite Formel) vorliegenden Strukturfehler durch je einen korrelierten Symbolfehler bedingt, der in beiden Fällen auf nicht im Training gesehenen Linienzugabfolgen beruht.

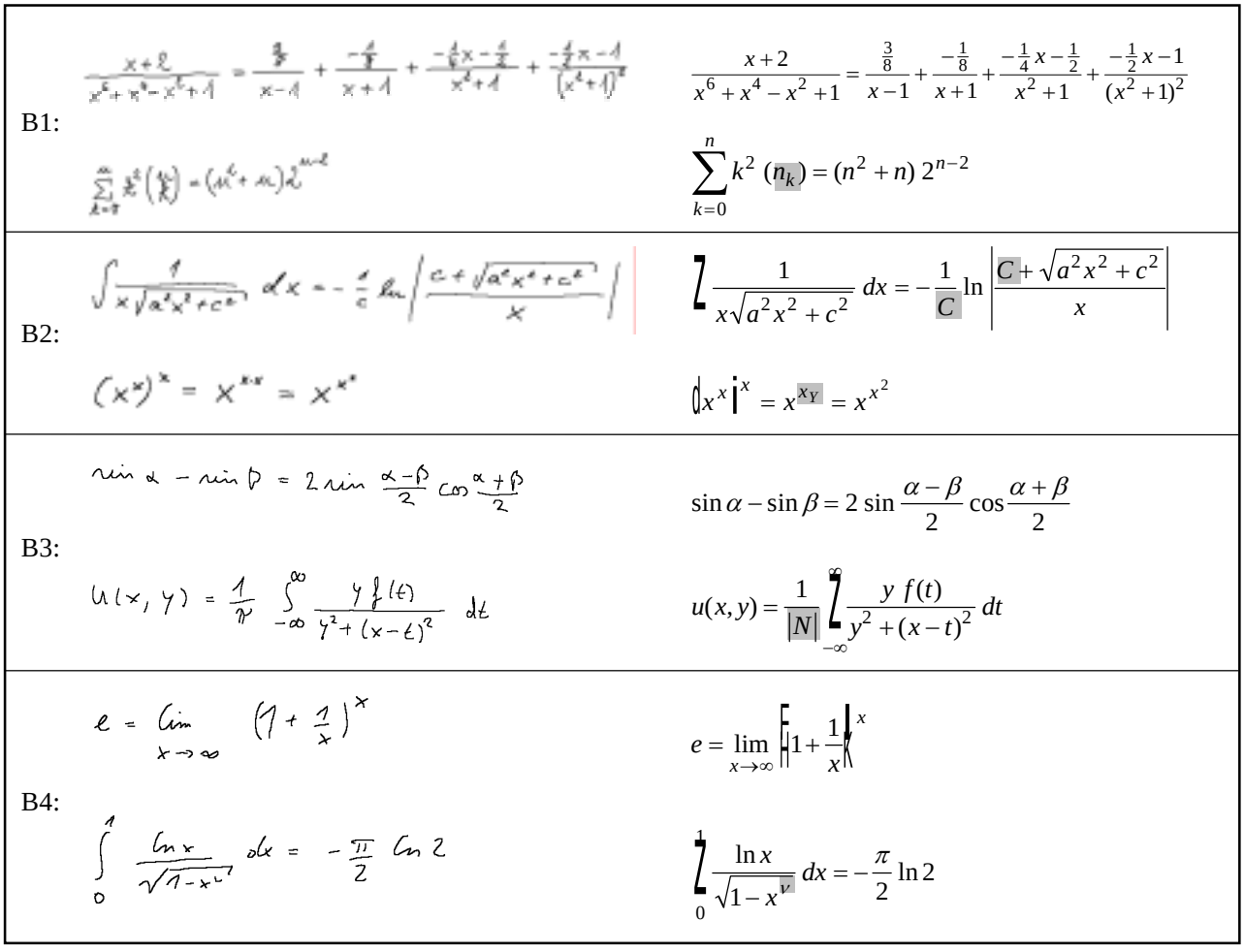


Abb. 9.2: Testformeln und Erkennungsergebnisse (vier verschiedene Benutzer, B1-B4, Auszug).

[33] Die Sprachverarbeitung ist dagegen trotz komplexerer Signale vergleichsweise einfach benutzerunabhängig zu realisieren. Das allgemeine Sprechverhalten ist durch eine Hauptaussprachevariante mit festgelegter Phonemabfolge pro Wort ausreichend gut modelliert.

Wie bereits in Kap. 6.5.3 angesprochen, ist es unangemessen, den Erfolg einer Formelerkennung an der Quote der fehlerfrei klassifizierten Datensätze zu messen. Die durch eine vollständige Formeleingabe übermittelte Informationsmenge liegt in einem Bereich, der eine auf Anhieb korrekte Erkennung auf dem derzeitigen Stand der Technik nicht erwarten lässt. Entscheidend aber für die Verwendung eines differenzierteren Evaluierungsmaßes ist die Frage der Verwertbarkeit eines fehlerbehafteten Resultats, die im vorliegenden Fall – außer bei Zurückweisung der Eingabe – positiv zu beantworten ist. Die Auswertung des Haupteingabemodus Handschrift nach Tab. 9.2 erfolgte daher nach dem lokalen bzw. globalen *Erkennungsgrad*, der folgendermaßen ermittelt wird:

Benutzer	Struktur-fehleranteil	Fehleranzahl pro Formel	Erkennungsgrad (handgeschriebene Formeln)	
			lokal	global
B1	0,55	2,3	92,9 %	94,7 %
B2	0,35	5,1 [34]	92,8 %	87,9 % [34]
B3	0,42	2,0	93,2 %	95,3 %
B4	0,28	2,8	91,2 %	93,4 %
Ø	**0,40**	**3,1**	**92,5 %**	**92,8 %**

Tab. 9.2: Evaluierung der rein handschriftlichen Formeleingabe (benutzerabhängig). Der Erkennungsgrad gibt die durchschnittliche prozentuale Korrektheit pro Formel (lokal) bzw. die durchschnittliche Quote der korrekt erkannten Konstituenten (Symbol- und Strukturanteile) über alle Formeln (global) an. Insgesamt treten bei durchschnittlich drei Einzelfehlern pro Formel weniger strukturelle (40 %) als symbolische Einzelfehler auf.

Für jede Testformel bestimmt man die Anzahl N_{sym} der enthaltenen Einzelzeichen (Operatoren- und Operandensymbole zusammengenommen) sowie die Anzahl N_{str} der enthaltenen Strukturbestandteile (mathematische Beziehung zwischen je zwei oder mehr Teilausdrücken). Für die zweite Formel (Benutzer B1) in Abb. 9.2 ergibt sich beispielsweise N_{sym} = 22 und N_{str} = 12. Dabei sind die durch das semantische Modell festgelegten Vorgaben (im Sinne der automatischen Beschränkung auf konsistente Formelhypothesen) berücksichtigt, so dass hier die Klammerung des Terms (n^2+n) als (unabhängiger) Strukturbestandteil zu werten ist, nicht jedoch die Klammerung des Binomialausdrucks, für den laut Modell eine Klammerung implizit vorgeschrieben ist.

Zudem berechnet man die Anzahl F_{sym} bzw. F_{str} aufgetretener Schriftzeichen- bzw. Strukturfehler[35], hier also F_{sym} = 0 (kein Schriftzeichenfehler) und F_{str} = 1. Für jede betrachtete Formel erhält man dann als Erkennungsgrad den Gesamtanteil richtig erkannter Konstituenten:

$$\text{Erkennungsgrad} = 1 - \frac{F_{sym} + F_{str}}{N_{sym} + N_{str}} \tag{9.1}$$

Die Angabe *lokal* in Tab. 9.2 bedeutet, dass die Auswertung formelbezogen erfolgt, also der durchschnittliche Erkennungsgrad pro Formel durch Mittelung nach Gl. 9.1 berechnet wird. Den *globalen*

[34] Ein Formeldatensatz wurde komplett zurückgewiesen (Erkennung fehlgeschlagen). Da für B2 jedoch ca. 70 % aller Formeln fehlerfrei erkannt wurden, ergibt sich dennoch ein relativ hoher lokaler Erkennungsgrad.

[35] In kombinierten Fällen (Beispiel zu B3 in Abb. 9.2) fallen bei beiden Werten entsprechende Beiträge an.

Erkennungsgrad erhält man hingegen, indem man in Gl. 9.1 die über alle Testdatensätze eines Benutzers summierten Werte einsetzt. Der Unterschied zwischen diesen beiden Größen besteht also darin, dass bei lokaler Betrachtung alle Fehler entsprechend der Komplexität der betroffenen Formeln gewichtet werden. Ein einzelner Schriftzeichenfehler in einer kompakten Formel erhält dann eine entsprechend stärkere Wertung als in einer sehr umfangreichen Formel, wie es auch der subjektiven Einschätzung des Benutzers entspricht.

Nach Tab. 9.2 variiert der Anteil insgesamt korrekt klassifizierter Formelbestandteile (globaler Erkennungsgrad) etwas stärker von Schreiber zu Schreiber, liegt aber im Mittel (∅) in der Nähe des durchschnittlichen lokalen Erkennungsgrades von annähernd 93 %. Diese Ergebnisse bilden zusammen mit der begrenzten strukturellen Ausdehnung der meisten Fehler eine brauchbare Ausgangsbasis für die alternierende Fehlerkorrektur, deren Wirksamkeit wie nachfolgend beschrieben untersucht wurde.

9.2 Alternierender Modus

Ausgehend von den Erkennungsergebnissen nach handschriftlicher Eingabe (Kap. 9.1.2) stellt sich die Frage, in welchem Maße die Formelkorrektur mittels Stiftgestik und Sprache zu einem verbesserten Gesamtergebnis führt. Daher wurde ein Anschlusstest durchgeführt, bei dem die Auswirkungen je eines Korrekturversuches pro fehlerhafter Formel bewertet wurden. Der bei Benutzer B1 in Abb. 9.2 angegebene Strukturfehler wurde also beispielsweise durch Selektion der betroffenen Schriftzeichen *n* und *k* sowie die Spracheingabe '*n über k*' nachbearbeitet.

Die aufgetretenen Erkennungsfehler lassen sich unter dem Aspekt des Korrekturaufwands grob in zwei Kategorien aufteilen:

1. Lokal begrenzte Fehler innerhalb eines geschlossenen Teilausdrucks bei einer mathematischen Schachtelungstiefe von höchstens zwei (Beispiel: $3\sqrt{Z} \rightarrow 3-\sqrt{2}$). Dabei kann eine Kombination aus Schriftzeichen- und Strukturfehlern vorliegen. Fehler dieser Kategorie sind in den meisten Fällen für eine Korrektur mit Stiftgestik und Sprache gut geeignet.
2. Ausgedehnte Fehler (Schachtelungstiefe drei oder mehr), zu denen auch entsprechend weit auseinander liegende Einzelfehler gerechnet werden, sowie Grobstrukturfehler, die den grundsätzlichen Formelaufbau betreffen (Extremfall: strukturfalsche Erkennung der übergeordneten mathematischen Beziehung, z.B. '=' bei einer Gleichung). Bei dieser Kategorie ist eine Sprachkorrektur entweder mit erheblichem Aufwand verbunden oder nicht intuitiv möglich. Bei Grobstrukturfehlern müsste unter Umständen die gesamte Formel nachgesprochen werden, was natürlich nicht in Betracht zu ziehen ist.

In Tab. 9.3 sind die Ergebnisse nach der alternierenden Stift-/Sprachkorrektur zusammengefasst. Der sog. kritische Formelanteil gibt den Prozentsatz der Formeln an, in denen Fehler der zweiten Kategorie auftraten und die daher ohne Korrektur belassen wurden (durchschnittlich etwa eine von 16 Formeln). Eine alternativ konventionelle Korrektur, die hier meist sinnvoller erscheint, wurde nicht evaluiert, da diese nicht zum Umfang des Erkennungsverfahrens der vorliegenden Arbeit gehört.

Benutzer	Kritischer Formelanteil	Evaluierung nach alternierender Korrektur	
		Erkennungsgrad (lokal)	Formelerkennungsrate
B1	4,2 %	97,9 %	87,5 %
B2	12,5 %	96,6 %	79,2 %
B3	0 %	98,3 %	91,7 %
B4	8,3 %	95,0 %	83,3 %
Ø	**6,3 %**	**96,9 %**	**85,4 %**

Tab. 9.3: Evaluierung nach einmaliger Fehlerkorrektur mittels Stiftgestik und Sprache.

Der Korrekturerfolg bei den verbleibenden Fehlern der ersten Kategorie ist nach dem oben definierten lokalen Erkennungsgrad und zusätzlich – da es sich potentiell um das Endresultat der Formeleingabe handelt – nach dem Anteil fehlerfrei zurückgegebener Formeln aufgeschlüsselt (Formelerkennungsrate). Es ergibt sich eine durchschnittliche Verbesserung gegenüber der ursprünglichen Handschriftanalyse um über vier Prozentpunkte, so dass eine vollständige Formel im Mittel zu 96,9 % korrekt erkannt wird. Darunter sind im Durchschnitt ca. sechs von sieben Formeln (85,4 %) nach der alternierenden Korrektur fehlerfrei. Die Persistenz der nicht erfolgreich bearbeiteten Fehler hat ihre Ursache einerseits in Abweichungen zwischen Stiftgestik- und Sprachmanipulation und andererseits in vereinzelt zu beobachtenden Fehlerkennungen hinsichtlich der nun kompakteren Spracheingaben. Die Spracherkennungsrate ist im Vergleich zu vollständig gesprochenen Formeln (Kap. 9.1.1) unter den vorliegenden Bedingungen ähnlich hoch (> 90 %) wie in typischen kommandoorientierten Anwendungsdomänen. Hierzu sei auf die parallel untersuchte Portierung des einstufigen Ansatzes zum Sprachverstehen in den Bereich von Fahrzeuganwendungen verwiesen [NEU02][STE99].

Insgesamt betrachtet liefert das vorgestellte Verfahren zur natürlichen Formelerfassung unter Einbeziehung der alternierenden Korrektur eine zeitsparende Alternative zur herkömmlichen Eingabe, wenn man einen Restfehleranteil von etwa drei Prozent der Formelbestandteile in Kauf nimmt. Im Hauptanwendungsbereich des Formelsatzes innerhalb einer Textverarbeitungsumgebung (z.B. FrameMaker) ist in der Praxis meist eine konventionelle Nachbearbeitung zur Feinabstimmung des Formel-Layouts notwendig, so dass die verbleibenden Erkennungsfehler im Zuge dieser Maßnahmen nachkorrigiert werden können. Nur vereinzelt – bei Zurückweisung der Eingabe oder weitreichenden Grobstrukturfehlern – ist eine Neueingabe, ggf. auf konventionelle Weise, notwendig.

Die in [STA97B], S. 87 f. und S. 128 getroffenen Aussagen zu Rechenaufwand und Speicherbedarf behalten ihre Gültigkeit auch für die Verarbeitung handgeschriebener Formeln. Auf dem zur Evaluierung verwendeten Standard-PC mit INTEL Pentium-III-Prozessor (450 MHz) steigt die Systemantwortzeit (incl. Eingabezeit) von ca. 60 Sekunden (bei einem Formelumfang von 30 Konstituenten) auf ca. 150 Sekunden (60 Formelbestandteile), entsprechend etwa 1,5- bis 2,5-facher Echtzeitverarbeitung (vgl. Kap. 6.5.3).

10

Diskussion und Ausblick

Für das Anwendungsszenario der mathematischen Formelerfassung wurde ein möglicher Weg aufgezeigt, um von der etablierten konventionellen Datenverarbeitung, wie sie in weiten Bereichen der heutigen Informationstechnologie vorherrscht, zu einer intuitiveren Mensch-Maschine-Interaktion zu gelangen. In Orientierung an den Bedürfnissen des Anwenders (*user-centered design*) bedeutet dies eine möglichst weitgehende Unterstützung der benutzergerechten Kommunikationsformen eines natürlichen Handschrift- und Sprachgebrauchs, wie er auch außerhalb elektronischer Arbeitsumgebungen anzutreffen ist.

Vom Standpunkt der Software-Ergonomie tritt dabei ein grundsätzlicher Konflikt auf, der in einem Zuwachs des systemseitigen Interpretationsspielraums besteht, wenn von der klassischen Auswahlmenü-Struktur eines systemgesteuerten Bedienablaufs abgerückt wird: Für den Anwender ist das Fehlen einer fortlaufenden Regulierung der Eingabe durch Systemvorgaben zwar eine Entlastung, die aber beim Auftreten von Auswertungsfehlern im beschriebenen Umfang wieder eingeschränkt wird. Auch die Einführung der vorgestellten Korrekturmöglichkeiten innerhalb eines nicht-konventionellen Bedienschemas bietet keine volle Garantie für eine fehlerfreie Formelerfassung. Es stellt sich daher unmittelbar die Frage nach sinnvollen Verbesserungs- und Erweiterungsmöglichkeiten des erarbeiteten Gesamtverfahrens.

Hier ist zunächst festzustellen, dass die erwartungsgetriebene Decodierung im zu erwartenden Maße sicherstellt, dass die Konsistenz des Erkennungsergebnisses gemäß semantischem Modell gewahrt bleibt. Der einstufige Top-Down-Ansatz liefert damit einen gewissen Ersatz für den genannten Wegfall einer permanenten Systemkontrolle während der Eingabe. Darüber hinaus liegt es wörtlich in der Hand des Benutzers, durch Zwischenauswertung die Interpretation seiner Eingabe stufenweise zu überprüfen. Um nun aber eine echte mathematische Konsistenzerhaltung bis hin zur Vorhersage mehrfach auftretender Variablenbezeichner (bzw. Abweisung unpassender Kombinationen) zu erreichen, wäre die Implementierung einer *kontextsensitiven* Grammatik als Basis für semantische und syntaktische Modellierung erforderlich und vielversprechend. Dadurch könnten (ggf. statistische) Abhängigkeiten höherer Ordnung erfasst werden, die nicht zuletzt – auf syntaktischer Ebene – zu einem detaillierteren Offsetmodell und damit unter Umständen auch zur schreiberunabhängigen Formelstrukturanalyse beitragen könnten. Die Vorteile der hier untersuchten kontextfreien Model-

lierung, ein ressourcenschonendes und entsprechend gut handhabbares Suchverfahren und die kompakte Repräsentation des syntaktisch-semantischen Wissens, gingen dann natürlich weitgehend verloren.

Eine solche Modellverfeinerung auf den höheren Abstraktionsebenen würde sich wiederum positiv bis hin zur Musterbewertung auswirken: Je plausibler die Symbolanfragen einer Teilhypothese im Durchschnitt hinsichtlich des Eingangssignals sind, desto zuverlässiger wird auch die Zuordnung zwischen Formelbausteinen und Eingabesegmenten (implizite Segmentierung) gelingen. Die Rechtfertigung für den Einsatz eines einfachen Handschrift-Mustervergleiches mittels DTW kann aufrechterhalten werden, solange ein beträchtlicher Anteil des Domänenwissens bereits in den höheren Verarbeitungsebenen ausgewertet wird. Ein Vergleich mit diskreten (in Anbetracht der einfachen Signale) HMMs unter Beibehaltung des Hauptsuchverfahrens wäre aber in jedem Fall interessant. Von größerer Bedeutung sind an dieser Stelle weitere Untersuchungen zur Vorverarbeitung und Merkmalbildung wegen des hohen Anteils von Einzelzeichen, die aus primitiven Graphemen in unterschiedlicher Kombination zusammengesetzt und damit oft schwer trennbar sind. Die zusätzliche Verwendung von schriftbildbasierten Offline-Merkmalen, wie sie in verschiedenen aktuellen Forschungsarbeiten auf diesem Gebiet genannt wird, bietet auch hier weiteres Verbesserungspotential. Ob damit jedoch auf absehbare Zeit eine schreiberunabhängige Schriftzeichenmodellierung (im Zusammenspiel mit den anderen Systemkomponenten) erreicht werden kann, ist schwer zu beurteilen.

Da auch die vorgeschlagenen Erweiterungen voraussichtlich kein fehlerfreies Gesamtsystem hervorbringen werden, ist die Schaffung weiterer Mechanismen zur flexibleren Korrektur und Nachbearbeitung von Formelteilen ein wichtiger Aspekt, um ein solches System für den Anwender insgesamt attraktiv erscheinen zu lassen. Nach den Ergebnissen des Benutzertests sollte dabei vorrangig eine Handschriftkorrektur beliebiger Teilausdrücke, etwa analog zur Stiftgestik-/Sprachmodifikation, in Betracht gezogen werden. Diese könnte insbesondere dann nutzbringend sein, wenn tatsächliche Schreibfehler oder die vereinzelt beobachteten trainingsfremden Schriftzeichenvarianten enthalten sind. Der kontrastive Einsatz von Handschrift und Sprache verspricht anderenfalls eher eine Erkennungsverbesserung aufgrund der erwähnten Unterschiede in Bezug auf die symbolspezifische Klassentrennbarkeit.

Entgegen dem allgemein bevorzugten sequentiellen Einsatz von Handschrift und Sprache ist eine Erhöhung der Fehlerrobustheit durch parallele Handschrift-/Spracheingabe und -auswertung nicht von der Hand zu weisen. Der in dieser Arbeit verfolgte gleichförmige Grundansatz zur Verarbeitung beider Interaktionsformen stellt eine gute Ausgangsbasis für die statistische Verrechnung bimodaler Eingangsdaten dar, sei es auf semantischer oder bereits auf signalnaher Ebene.

Nicht zuletzt wäre eine Adaption von Auswahl-, Einfüge- und Ersetzungsoptionen aus herkömmlichen graphikorientierten Editoren sinnvoll, um anhand von Erkennungsalternativen des Suchverfahrens (*N-best*-Hypothesen) beispielsweise durch Aufklappmenüs einzelne oder Gruppen von Formelsymbolen durch die gewünschten Einträge zu ersetzen. Dieser Gesichtspunkt unterstreicht nochmals das ursprüngliche Ziel der vorliegenden Arbeit, einer anwenderfreundlichen Symbiose zwischen konventioneller und natürlicher Mensch-Maschine-Kommunikation einen Schritt näher zu kommen.

A

Anhang

A.1 Referenzmodell

```
#****************************************
# REFERENCE MODEL
# for mathematical expressions
# Last Update: 02/07/18
# Format: type X value value ... value;
#         X value value ... value; ;
#****************************************

#0
DIGIT
        1
                zero
                one
                two
                three
                four
                five
                six
                seven
                eight
                nine
        ;
;

#1
NAT_NUM
        1
                one
                two
                three
                four
                five
                six
                seven
                eight
                nine
        ;
;

#2
ZERO
        1
                void
        ;
;

#3
FRAC_NUM
        2
                void
        ;
;

#4
NUMBER
        1
                void
        ;
;

#5
VAR
        1
                lat_a
                lat_b
                lat_c
                lat_d
                lat_e
                lat_f
                lat_g
                lat_h
                lat_i
                lat_j
                lat_k
                lat_l
                lat_m
                lat_n
                lat_o
                lat_p
                lat_q
                lat_r
                lat_s
                lat_t
                lat_u
                lat_v
                lat_w
                lat_x
                lat_y
                lat_z
                lat_A
                lat_B
                lat_C
                lat_D
                lat_E
                lat_F
                lat_G
                lat_H
                lat_I
```

```
                lat_J
                lat_K
                lat_L
                lat_M
                lat_N
                lat_O
                lat_P
                lat_Q
                lat_R
                lat_S
                lat_T
                lat_U
                lat_V
                lat_W
                lat_X
                lat_Y
                lat_Z
                gr_alpha
                gr_beta
                gr_chi
                gr_epsilon
                gr_gamma
                gr_delta
                gr_sm_phi
                gr_phi
                gr_eta
                gr_kappa
                gr_lamda
                gr_mu
                gr_nu
                gr_pi
                gr_sm_theta
                gr_theta
                gr_rho
                gr_sigma
                gr_tau
                gr_omega
                gr_xi
                gr_psi
                gr_zeta
                gr_Delta
                gr_Phi
                gr_Gamma
                gr_Omega
                gr_Psi
                partial
                infinity
        ;
;

#6
FUNCSPEC
        1
                func_cos
                func_sin
                func_tan
                func_cot
                func_cosh
                func_sinh
                func_tanh
                func_coth
                func_arccos
                func_arcsin
                func_arctan
                func_exp
                func_ln
                func_log
                func_arg
        ;
;

#7
PARAMLIST
        2
                void
        ;
;

#8
TERM_PARAMLIST
        1
                prmlst_()
                prmlst_[]
                prmlst_{}
        ;
;

#9
FUNCVAR_POW
        2
                fnvar_pow
                fnvar_idx
        ;
        3
                fnvar_idxpow
        ;
;

#10
FUNCSPEC_POW
        2
                fnspec_pow
        ;
;

#11
FUNC
        2
                func_bare
                func_par
        ;
;

#12
FNIDX
        2
                fnidx_lim
                fnidx_min
                fnidx_max
        ;
;


#13
PAR
        1
                par_()
                par_[]
                par_{}
                par_||
        ;
;

#14
FRAC
        2
                void
        ;
;

#15
ROOT
        1
                root_bare
        ;
        2
                root_pow
        ;
;

#16
BINOM
        2
                void
        ;
;
```

```
#17
FAC
        1
                void
        ;
;

#18
INT
        2
                int_bare
        ;
        3
                int_idx
        ;
        4
                int_idxpow
        ;
;

#19
SUMPROD
        1
                sum_bare
                prod_bare
        ;
        2
                sum_idx
                prod_idx
        ;
        3
                sum_idxpow
                prod_idxpow
        ;
;

#20
TERM_ATOM
        1
                void
        ;
;

#21
POW
        2
                pow_pow
                pow_idx
        ;
        3
                pow_idxpow
        ;
;

#22
TERM_POW
        1
                void
        ;
;

#23
PROD
        2
                prod_void
                prod_pnt
                prod_div
                prod_conv
        ;
;

#24
TERM_PROD
        1
                void
        ;
;

#25
TERM_SGNPROD
        1
                sgnprod_pos
                sgnprod_neg
        ;
;

#26
SUM
        2
                sum_plus
                sum_minus
        ;
;

#27
SGN_SUM
        2
                sgnsum_plus
                sgnsum_minus
        ;
;

#28
TERM_SGNSUM
        1
                void
        ;
;

#29
REL
        2
                rel_equ
                rel_gr
                rel_gre
                rel_ls
                rel_lse
                rel_lftarw
        ;
;

#30
TERM_REL
        1
                void
        ;
;
```

A.2 Referenzformeln[36] (Handschrift)

(H.1)
$$\frac{3x^4-9x^3+4x^2-34x+1}{x^5-15x^3+10x^2+60x-72}=\frac{1}{x-2}+\frac{-3}{(x-2)^3}+\frac{2}{x+3}+\frac{-5}{(x+3)^2}$$

(H.2)
$$-0.267\cdot 34.25a\left(-b+c(cx-y)+y\right)-5.3\cdot 248.98\,(x+y)(z-xy)$$

(H.3)
$$\sqrt[p]{\sum_{k=1}^{n}(x_k+y_k)^p}\le\sqrt[p]{\sum_{k=1}^{n}x_k^p}+\sqrt[p]{\sum_{k=1}^{n}y_k^p}$$

(H.4)
$$\pi(k,n,\varphi)=\int_0^{\varphi}\frac{1}{(1+n\sin^2\theta)\sqrt{1-k^2\sin^2\theta}}\,d\theta$$
$$=\int_0^{x}\frac{1}{(1+nt^2)\sqrt{(1-t^2)(1-k^2t^2)}}\,dt$$

(H.5)
$$\sum_{i=0}^{n}f(x_i)=\frac{1}{h}\int_{x_0}^{x_n}f(x)dx+\frac{1}{2}\left[f(x_0)+f(x_n)\right]$$
$$+\sum_{k=1}^{m}\frac{h^{2k-1}B_{2k}}{(2k)!}\left[f^{(2k-1)}(x_n)+f^{(2k-1)}(x_0)\right]+\frac{nB_{2m+2}h^{2m+2}}{(2m+2)!}f^{(2m+2)}(\xi)$$

(H.6)
$$N(x,y)=\frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}\exp\left(-\frac{1}{2(1-\rho^2)}\right.$$
$$\left.\left(\left(\frac{x-\mu_1}{\sigma_1}\right)^2-2\rho\left(\frac{x-\mu_1}{\sigma_1}\right)\left(\frac{y-\mu_2}{\sigma_2}\right)+\left(\frac{y-\mu_2}{\sigma_2}\right)^2\right)\right)$$

(H.7)
$$\lim_{x\to 0}\frac{1-\cos x}{x^2}=\lim_{x\to 0}\frac{1-\left(1-\frac{x^2}{2}+O\left(x^4\right)\right)}{x^2}=\lim_{x\to 0}\left(\frac{1}{2}+O\left(x^2\right)\right)=\frac{1}{2}$$

---

[36] Quelle: [RÅD97]

(H.8) $$\tan\left(\frac{\alpha}{2}-\frac{\varepsilon}{2}\right)=\sqrt{\frac{\tan\frac{s-b}{2}\tan\frac{s-c}{2}}{\tan\frac{s}{2}\tan\frac{s-a}{2}}}$$

(H.9) $$\int_V f(r,\varphi,\vartheta)dV=\int_{\varphi_1}^{\varphi_2}\int_{g_1(\varphi)}^{g_2(\varphi)}\int_{h_1(\vartheta,\varphi)}^{h_2(\vartheta,\varphi)} f(r,\varphi,\vartheta)r^2\sin\vartheta dr\,d\vartheta\,d\varphi$$

(H.10) $$Y_n(x)=\lim_{p\to n}Y_p(x)=\frac{2}{\pi}\left(\gamma+\ln\frac{x}{2}\right)J_n(x)-\frac{1}{\pi}\sum_{k=0}^{n-1}\frac{(n-k-1)!}{k!}\left(\frac{x}{2}\right)^{2k-n}$$
$$-\sum_{k=0}^{\infty}\left(H_k+H_{k+n}\right)\frac{(-1)^k}{k!(n+k)!}\left(\frac{x}{2}\right)^{2k-n}$$

(H.11) $$Y_n(x)=\frac{1}{\pi}\int_0^{\pi}\sin\left(x\sin t-nt\right)dt+\frac{1}{\pi}\int_0^{\infty}\left[e^{nt}+(-1)^n e^{-nt}\right]e^{-x\sinh t}dt$$

(H.12) $$\left|F(x+\Delta x)-F(x)\right|=\int_x^{x+\Delta x} f(t)dt\le\left(\max_t\left|f(t)\right|\right)\Delta x$$

(H.13) $$\cos^{2n}\alpha=\binom{2n}{n}\frac{1}{2^{2n}}+\frac{1}{2^{2n-1}}\sum_{k=1}^{n}\binom{2n}{n-k}\cos 2k\alpha$$

(H.14) $$F=\int_0^{2\pi}\int_0^{\rho_0}\sqrt{\rho^2+\left(\frac{a}{2\pi}\right)^2}\,d\rho\,d\varphi=\pi\rho_0^2\left\{\sqrt{1+\left(\frac{a}{2\pi\rho_0}\right)^2}+\left(\frac{a}{2\pi\rho_0}\right)^2\ln\frac{1+\sqrt{1+\left(\frac{a}{2\pi\rho_0}\right)^2}}{\frac{a}{2\pi\rho_0}}\right\}$$

(H.15) $$\int\frac{\sin^m x}{\cos^n x}dx=\frac{\sin^{m+1}x}{(n-1)\cos^{n-1}x}-\frac{m-n+2}{n-1}\int\frac{\sin^m x}{\cos^{n-2}x}dx$$
$$=-\frac{\sin^{m-1}x}{(m-n)\cos^{n-1}x}+\frac{m-1}{m-n}\int\frac{\sin^{m-2}x}{\cos^n x}dx$$

(H.16) $$G_k^m(n)=\left(\sqrt[n]{\prod_n(k+a_n)}\right)^m$$

A.3 Schriftzeichenbestand

A.3.1 Inventar

0123456789

a b c d e f g h i j k l m n o p q r s t u v w x y z

A B C D E F G H I J K L M N O P R S T U V W X Y Z

$\alpha\ \beta\ \chi\ \delta\ \varepsilon\ \phi\ \varphi\ \gamma\ \eta\ \kappa\ \lambda\ \mu\ \nu\ \pi\ \theta\ \vartheta\ \rho\ \sigma\ \tau\ \omega\ \xi\ \psi\ \zeta$

$\Delta\ \Phi\ \Gamma\ \Pi\ \Sigma\ \Omega\ \Psi$

cos sin tan cot cosh sinh tanh coth

arccos arcsin arctan exp ln log arg

lim min max

$, . + - * / ! \sqrt{\ } = < \leq \geq > \partial \rightarrow \infty$

$\{ [(|)] \}$

A.3.2 Lexikon (Elementverzeichnis für DTW-Wissensbasis)

0-9	0 1 2 3 4 5 6 7 8 9
a-z	a b c … z
A-Z	A B C … Z
alpha-omega	alpha beta chi … zeta
Delta-Psi	Delta Phi Gamma Pi Sigma Omega Psi
+-/*	, . + - * / ! root rtprt ¬ = < <= >= > -> partial infty
(){}	( ) [ ] { } |
int cos ln	int sin cos tan cot sinh … arcsin … exp ln log arg lim min max

A.4 Referenzformeln[37] (Sprache)

(S.1) $$x(t) = \frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{t^2}{2}} \circ\!\!-\!\!\bullet\, X(\omega) = e^{-\frac{\omega^2}{2}}$$

(S.2) $$\sin(2k) = 2 \cdot \sin(k) \cdot \cos(k) = \frac{2 \cdot \tan(k)}{1 + \tan(k)}$$

(S.3) $$F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega t} dt$$

(S.4) $$\sum_{m=0}^{M} b^m = \frac{1 - b^{M+1}}{1 - b}$$

(S.5) $$y(t) = h(t) * x(t) \circ\!\!-\!\!\bullet\, Y(\omega) = H(\omega) \cdot X(\omega)$$

(S.6) $$\frac{A \cdot (B + C)}{D \cdot (E + G)} = \frac{A \cdot B + A \cdot C}{D \cdot E + D \cdot G}$$

(S.7) $$P_g(t) = \left| g(t) \right|^2$$

(S.8) $$\frac{1}{j \cdot \omega} + \pi \cdot \delta(\omega) \bullet\!\!-\!\!\circ\, u(t)$$

(S.9) $$\frac{(T + I) \cdot (J + K)}{(L - N) \cdot (L + N)} = \frac{T \cdot J + T \cdot K + I \cdot J + I \cdot K}{L^2 - N^2}$$

(S.10) $$\left\{ \left[(2 + 3) \cdot \frac{(4 + 5)}{3} - 8 \right] \cdot 3 + 9 \right\} \cdot \frac{2}{5} = 12$$

(S.11) $$\sqrt[2]{\frac{\frac{3}{4} + \sqrt{\frac{9}{4}}}{9 \cdot \frac{5}{7}} + \frac{13}{20}} = 1$$

(S.12) $$\frac{w}{p} \cdot \sum_{q=0}^{\infty} \left((-1)^q \cdot \frac{1}{2^q} \right) = \frac{3w}{2p}$$

(S.13) $$H(z) = \sum_{r=-\infty}^{\infty} h(r) z^{-r}$$

(S.14) $$\sum_{\nu=1}^{\infty} \frac{(-1)^{\nu-1}}{\nu} = \ln(2)$$

(S.15) $$\frac{(2Q - 2R) \cdot (2O + 2R)}{(S + U) \cdot (S - U) + (V - W) \cdot (V + W) + 6Z^2}$$

(S.16) $$x[n] * y_1[n] = \sum_{m=-\infty}^{+\infty} x[m] \cdot y_1[n - m]$$

(S.17) $$y_1(t) = x(t) * h_1(t) = \int_{-\infty}^{\infty} x(s) \cdot h_1(t - s) ds$$

---

[37] Quelle: [WIN97B], S. 195 f.

A.5 Sprachvokabular und phonetisches Lexikon

-	<p:>
0	nUl
1	aIns
10	tse:n
11	Elf
12	tsv9lf
16	zECtse:n
17	zi:btse:n
2	tsvaI
20	tsvantsIC
3	draI
4	fi:6
5	fYnf
6	zEks
7	zi:b@n
8	axt
9	nOYn
a	a:
achte	axt@
aeh	E
aehm	Em
alpha	alfa:
arcuscosinus	a:kUsko:zInUs
arcuscosinus_von	a:kUsko:zInUsfOn
arcussinus	a:kUsi:nUs
arcussinus_von	a:kUsi:nUsfOn
arcustangens	a:kUstaNgEns
arcustangens_von	a:kUstaNgEnsfOn
argument_von	a:gu:mEntfOn
aus	aUs
aus_allem_die	aUsal@mdi:
aus_dem	aUsde:m
b	be:
be	b@
beta	bEta:
betrag	b@tra:k
betrag_von	b@tra:kfOn
bis	bIs
bruch	brUx
bruchstrich	brUxStrIC
c	tse:
chi	Ci:
cosinus	ko:zInUs
cosinus_von	ko:zInUsfOn
cosinushyperbolicus	ko:zInUshy:p6bo:lIkUs
cosinushyperbolicus_von	ko:zInUshy:p6bo:lIkUsfOn
cotangens	ko:taNgEns
cotangens_von	ko:taNgEnsfOn
cotangenshyperbolicus	ko:taNgEnshy:p6bo:lIkUs
cotangenshyperbolicus_von	ko:taNgEnshy:p6bo:lIkUsfOn
d	de:
daraus	daraUs
das	das
das_ganze	dasgants@
das_ganze_dividiert_durch	dasgants@dIvIdi:6tdU6C
das_ganze_geteilt_durch	dasgants@g@taIltdU6C
das_produkt	daspro:dUkt
das_produkt_aus	daspro:dUktaUs
davon_ist	dafOnIst
dazu_ist	datsu:Ist
delta	dElta
der	de:6
der_natuerliche_logarithmus_von	de:6naty:6lIC@lOgarItmUsfOn
die_fouriertransformierte	di:fU6je:transfO6mi:6t@
die_quadrat	di:kvadra:t
die_summe	di:zUm@
die_summe_aus	di:zUm@aUs
die_wurzel	di:vU6ts@l
die_zweite	di:tsvaIt@

die_zweite_quadrat	di:tsvaIt@kvadra:t
dirac	dIrak
dividiert_durch	dIvIdi:6tdU6C
drat	dra:t
drit	drIt
dritte	drIt@
durch	dU6C
e	e:
eckige_klammer_auf	EkIg@klam6aUf
eckige_klammer_zu	EkIg@klam6tsu:
entspricht	EntSprICt
epsilon	EpsIlOn
ergibt	E6gi:pt
ergibt_das	E6gi:ptdas
ergibt_fouriertransformiert	E6gi:ptfU6je:transfO6mi:6t
eta	Eta:
exponentialfunktion	EkspOnEntsja:lfUNktsjo:n
exponentialfunktion_von	EkspOnEntsja:lfUNktsjo:nfOn
f	Ef
fakultaet	fakUltEt
fourierruecktransformiert_in	fU6je:rYktransfO6mi:6tIn
fouriertransformiert	fU6je:transfO6mi:6t
fouriertransformiert_ist_das	fU6je:transfO6mi:6tIstdas
fouriertransformierte	fU6je:transfO6mi:6t@
fuenfte	fYnft@
g	ge:
gamma	gama:
gebrochen_durch	g@brOx@ndU6C
gebrochen_mit	g@brOx@nmIt
gefaltet	g@falt@t
gefaltet_mit	g@falt@tmIt
gegen	ge:gN
geht_gegen	ge:tge:gN
geschweifte_klammer_auf	g@SvaIft@klam6aUf
geschweifte_klammer_zu	g@SvaIft@klam6tsu:
geteilt_durch	g@taIltdU6C
gleich	glaIC
groesser	gr2:s6
groesser_als	gr2:s6als
groesser_gleich	gr2:s6glaIC
gross_a	gro:sa:
gross_b	gro:sbe:
gross_c	gro:stse:
gross_d	gro:sde:
gross_e	gro:se:
gross_f	gro:sEf
gross_g	gro:sge:
gross_h	gro:sha:
gross_i	gro:si:
gross_j	gro:sjOt
gross_k	gro:ska:
gross_l	gro:sEl
gross_m	gro:sEm
gross_n	gro:sEn
gross_o	gro:so:
gross_p	gro:spe:
gross_q	gro:sku:
gross_r	gro:sE6
gross_s	gro:sEs
gross_t	gro:ste:
gross_u	gro:su:
gross_v	gro:sfaU
gross_w	gro:sve:
gross_x	gro:sIks
gross_y	gro:sYpsIlOn
gross_z	gro:stsEt
grosser_bruchstrich	gro:s6brUxStrIC
h	ha:
hal	hal
hoch	ho:x
i	i:
im	Im
im_zaehler_eines_grossen_bruchs_im_nenner	ImtsEl6aIn@sgro:s@nbrUxsImnEn6

in	In
in_klammern	Inklam6n
index	IndEks
integral	Int@gra:l
integral_von	Int@gra:lfOn
invers_fouriertransformiert	InvE6sfU6je:transfO6mi:6t
inverse_fouriertransformation_in	InvE6s@fU6je:transfO6matsjo:nIn
inverse_fouriertransformation_ist_gleich	InvE6s@fU6je:transfO6matsjo:nIstglaIC
inverse_fouriertransformierte_ist	InvE6s@fU6je:transfO6mi:6t@Ist
iota	jOta:
ist	Ist
ist_die_inverse_fouriertransformation_von	Istdi:InvE6s@fU6je:transfO6matsjo:nfOn
ist_fouriertransformiert	IstfU6je:transfO6mi:6t
ist_gleich	IstglaIC
ist_invers_fouriertransformiert	IstInvE6sfU6je:transfO6mi:6t
j	jOt
k	ka:
kappa	kapa:
klammer_auf	klam6aUf
klammer_zu	klam6tsu:
klein_a	klaIna:
klein_b	klaInbe:
klein_c	klaIntse:
klein_d	klaInde:
klein_e	klaIne:
klein_f	klaInEf
klein_g	klaInge:
klein_h	klaInha:
klein_i	klaIni:
klein_j	klaInjOt
klein_k	klaInka:
klein_l	klaInEl
klein_m	klaInEm
klein_n	klaInEn
klein_o	klaIno:
klein_omega	klaInOm@ga
klein_p	klaInpe:
klein_q	klaInku:
klein_r	klaInE6
klein_s	klaInEs
klein_t	klaInte:
klein_u	klaInu:
klein_v	klaInfaU
klein_w	klaInve:
klein_x	klaInIks
klein_y	klaInYpsIlOn
klein_z	klaIntsEt
kleiner	klaIn6
kleiner_als	klaIn6als
kleiner_gleich	klaIn6glaIC
komma	kOma:
l	El
lambda	lamda:
limes	li:mEs
ln	ElEn
ln_von	ElEnfOn
logarithmus_naturalis_von	lOgarItmUsnatUra:lIsfOn
logarithmus_von	lOgarItmUsfOn
m	Em
mal	ma:l
maximum	maksi:mUm
maximum_ueber	maksi:mUmy:b6
minimum	mIni:mUm
minimum_ueber	mIni:mUmy:b6
minus	mi:nUs
mit	mIt
mue	my:
n	En
nach	na:x
nach_d	na:xde:
nenner	nEn6
neunte	nOYnt@
nue	ny:

o	o:
oben	o:b@n
omega	Om@ga
p	pe:
partial	patsja:l
phi	fi:
pi	pi:
plus	plUs
produkt	pro:dUkt
produkt_aus	pro:dUkt#aUs
psi	psi:
punkt	pUNkt
q	ku:
qua	kva
quadrat	kvadra:t
r	E6
rho	ro:
runde_klammer_auf	rUnd@klam6aUf
runde_klammer_zu	rUnd@klam6tsu:
s	Es
sechste	zEkst@
sieb	si:p
siebte	zi:pt@
sigma	zIgma:
sinus	si:nUs
sinus_von	si:nUsfOn
sinushyperbolicus	si:nUshy:p6bo:lIkUs
sinushyperbolicus_von	si:nUshy:p6bo:lIkUsfOn
stel	StEl
summe	zUm@
summe_aus	zUm@aUs
t	te:
tangens	taNgEns
tangens_von	taNgEnsfOn
tangenshyperbolicus	taNgEnshy:p6bo:lIkUs
tangenshyperbolicus_von	taNgEnshy:p6bo:lIkUsfOn
tau	taU
tel	t@l
teta	tEta
u	u:
ueber	y:b6
ueber_d	y:b6de:
ueber_das_ganze	y:b6dasgants@
ueber_den_ganzen_ausdruck	y:b6de:ngants@naUsdrUk
und	Unt
und_aus_all_diesem	UntaUsaldi:z@m
und_daraus	UntdaraUs
und_das_ganze	Untdasgants@
unendlich	UnEntlIC
unten	Unt@n
upsilon	UpsIlOn
v	faU
vierte	fi:6t@
von	fOn
von_in_klammern	fOnInklam6n
w	ve:
wurzel	vU6ts@l
wurzel_von	vU6ts@lfOn
x	Iks
xi	ksi:
y	YpsIlOn
z	tsEt
zeta	tsEta:
zu	tsu:
zum	tsUm
zum_qua	tsUmkva
zweite	tsvaIt@

A.6 Semantisches Modell

```
#*******************************************************************************
# SEMANTIC MODEL for mathematical expressions
#*******************************************************************************
DIGIT                   0.0    # semun type, root prob.
    1
    >DIGIT                                      1.0    # successor set, sequence prob.
    >LEER                                       1.0    #     "              "
        zero                                    1.0        # semun value, value prob.
        one                                     1.0        #   "            "
        two                                     1.0
        ...
    ;
;
NAT_NUM                 0.0
    1
    >DIGIT                                      1.0
    >LEER                                       1.0
        one                                     1.0
        two                                     1.0
        ...
    ;
;
ZERO                    0.0
    1
    >LEER                                       1.0
        void                                    1.0
    ;
;
FRAC_NUM                0.0
    2
    >NAT_NUM>DIGIT                              1.0
    >ZERO>DIGIT                                 1.0
        void                                    1.0
    ;
;
NUMBER                  0.0
    1
    >ZERO                                       1.0
    >NAT_NUM                                    1.0
    >FRAC_NUM                                   1.0
        void                                    1.0
    ;
;
VAR                     0.0
    1
    >LEER                                       1.0
        lat_a                                   1.0
        lat_b                                   1.0
        lat_c                                   1.0
        ...
    ;
;
FUNCSPEC                0.0
    1
    >LEER                                       1.0
        func_cos                                1.0
        func_sin                                1.0
        func_tan                                1.0
        ...
    ;
;
PARAMLIST               0.0
    2
    >TERM_SGNSUM>TERM_SGNSUM                    1.0
    >TERM_SGNSUM>PARAMLIST                      1.0
        void                                    1.0
    ;
;
TERM_PARAMLIST          0.0
    1
```

```
    >TERM_SGNPROD                                1.0
    >TERM_SGNSUM                                 1.0
    >PARAMLIST                                   1.0
        prmlst_()                                1.0
        prmlst_[]                                1.0
        prmlst_{}                                1.0
    ;
;
FUNCVAR_POW             0.0
    2
    >VAR>TERM_ATOM                               1.0
    >VAR>TERM_SGNSUM                             1.0
        fnvar_pow                                1.0
        fnvar_idx                                1.0
    ;
    3
    >VAR>TERM_ATOM>TERM_ATOM                     1.0
    >VAR>TERM_ATOM>TERM_SGNSUM                   1.0
    >VAR>TERM_SGNSUM>TERM_ATOM                   1.0
    >VAR>TERM_SGNSUM>TERM_SGNSUM                 1.0
        fnvar_idxpow                             1.0
    ;
;
FUNCSPEC_POW            0.0
    2
    >FUNCSPEC>TERM_ATOM                          1.0
    >FUNCSPEC>TERM_SGNSUM                        1.0
        fnspec_pow                               1.0
    ;
;
FUNC                    0.0
    2
    >FUNCSPEC>TERM_POW                           1.0
    >FUNCSPEC>PROD                               1.0
    >FUNCSPEC>TERM_SGNPROD                       1.0
    >FUNCSPEC_POW>TERM_POW                       1.0
    >FUNCSPEC_POW>PROD                           1.0
    >FUNCSPEC_POW>TERM_SGNPROD                   1.0
        func_bare                                1.0
    ;
    2
    >VAR>TERM_PARAMLIST                          1.0
    >FUNCSPEC>TERM_PARAMLIST                     1.0
    >FUNCVAR_POW>TERM_PARAMLIST                  1.0
    >FUNCSPEC_POW>TERM_PARAMLIST                 1.0
        func_par                                 1.0
    ;
;
FNIDX                    0.0
    2
    >PROD>REL                                    1.0
    >TERM_POW>REL                                1.0
        fnidx_lim                                1.0
    ;
    2
    >PROD>TERM_POW                               1.0
    >TERM_POW>TERM_POW                           1.0
        fnidx_min                                1.0
        fnidx_max                                1.0
    ;
;
PAR                      0.0
    1
    >SGN_SUM                                     1.0
    >PROD                                        1.0
    >TERM_POW                                    1.0
    >TERM_SGNPROD                                1.0
        par_()                                   1.0
        par_[]                                   1.0
        par_{}                                   1.0
    ;
    1
    >TERM_SGNSUM                                 1.0
```

```
        par_||                                      1.0
    ;
;
FRAC                    0.0
    2
    >TERM_SGNSUM>TERM_SGNSUM                        1.0
        void                                        1.0
    ;
;
ROOT                    0.0
    1
    >TERM_ATOM                                      1.0
    >TERM_SGNSUM                                    1.0
        root_bare                                   1.0
    ;
    2
    >TERM_ATOM>TERM_SGNSUM                          1.0
    >TERM_SGNSUM>TERM_SGNSUM                        1.0
        root_pow                                    1.0
    ;
;
BINOM                   0.0
    2
    >TERM_ATOM>TERM_ATOM                            1.0
    >TERM_ATOM>TERM_SGNSUM                          1.0
    >TERM_SGNSUM>TERM_ATOM                          1.0
    >TERM_SGNSUM>TERM_SGNSUM                        1.0
        void                                        1.0
    ;
;
FAC                     0.0
    1
    >TERM_ATOM                                      1.0
    >FUNCVAR_POW                                    1.0
        void                                        1.0
    ;
;
INT                     0.0
    2
    >TERM_SGNPROD>TERM_POW                          1.0
        int_bare                                    1.0
    ;
    3
    >TERM_SGNPROD>TERM_POW>TERM_POW                 1.0
        int_idx                                     1.0
    ;
    4
    >TERM_SGNPROD>TERM_POW>TERM_ATOM>TERM_ATOM      1.0
    >TERM_SGNPROD>TERM_POW>TERM_REL>TERM_ATOM       1.0
    >TERM_SGNPROD>TERM_POW>TERM_ATOM>TERM_SGNSUM    1.0
    >TERM_SGNPROD>TERM_POW>TERM_REL>TERM_SGNSUM     1.0
        int_idxpow                                  1.0
    ;
;
SUMPROD                 0.0
    1
    >TERM_ATOM                                      1.0
    >TERM_SGNPROD                                   1.0
        sum_bare                                    1.0
        prod_bare                                   1.0
    ;
    2
    >TERM_ATOM>TERM_POW                             1.0
    >TERM_SGNPROD>TERM_POW                          1.0
        sum_idx                                     1.0
        prod_idx                                    1.0
    ;
    3
    >TERM_ATOM>REL>TERM_ATOM                        1.0
    >TERM_SGNPROD>REL>TERM_SGNSUM                   1.0
        sum_idxpow                                  1.0
        prod_idxpow                                 1.0
    ;
```

```
;
TERM_ATOM                0.0
    1
    >NUMBER                                  1.0
    >VAR                                     1.0
    >BINOM                                   1.0
    >PAR                                     1.0
        void                                 1.0
    ;
;
POW                      0.0
    2
    >TERM_ATOM>TERM_ATOM                     1.0
    >TERM_ATOM>TERM_SGNSUM                   1.0
        pow_pow                              1.0
    ;
    2
    >VAR>TERM_ATOM                           1.0
    >VAR>TERM_SGNSUM                         1.0
        pow_idx                              1.0
    ;
    3
    >VAR>TERM_ATOM>TERM_ATOM                 1.0
    >VAR>TERM_ATOM>TERM_SGNSUM               1.0
    >VAR>TERM_SGNSUM>TERM_ATOM               1.0
    >VAR>TERM_SGNSUM>TERM_SGNSUM             1.0
        pow_idxpow                           1.0
    ;
;
TERM_POW                 0.0
    1
    >FUNC                                    1.0
    >FRAC                                    1.0
    >ROOT                                    1.0
    >FNIDX                                   1.0
    >FAC                                     1.0
    >INT                                     1.0
    >SUMPROD                                 1.0
    >TERM_ATOM                               1.0
    >POW                                     1.0
        void                                 1.0
    ;
;
PROD                     0.0
    2
    >TERM_POW>TERM_POW                       1.0
    >TERM_POW>PROD                           1.0
        prod_void                            1.0
        prod_pnt                             1.0
        prod_div                             1.0
        prod_conv                            1.0
    ;
;
TERM_PROD                0.0
    1
    >TERM_POW                                1.0
    >PROD                                    1.0
        void                                 1.0
    ;
;
TERM_SGNPROD             0.0
    1
    >TERM_POW                                1.0
    >PROD                                    1.0
        sgnprod_pos                          1.0
        sgnprod_neg                          1.0
    ;
;
SUM                      0.0
    2
    >TERM_PROD>TERM_PROD                     1.0
    >TERM_PROD>SUM                           1.0
        sum_plus                             1.0
```

```
        sum_minus                               1.0
    ;
;
SGN_SUM                 0.0
    2
    >TERM_SGNPROD>TERM_PROD                     1.0
    >TERM_SGNPROD>SUM                           1.0
        sgnsum_plus                             1.0
        sgnsum_minus                            1.0
    ;
;
TERM_SGNSUM             0.0
    1
    >TERM_SGNPROD                               1.0
    >SGN_SUM                                    1.0
        void                                    1.0
    ;
;
REL                     0.0
    2
    >TERM_SGNSUM>TERM_SGNSUM                    1.0
    >TERM_SGNSUM>REL                            1.0
        rel_equ                                 1.0
        rel_gr                                  1.0
        rel_gre                                 1.0
        rel_ls                                  1.0
        rel_lse                                 1.0
        rel_lftarw                              1.0
    ;
;
TERM_REL                1.0
    1
    >TERM_SGNSUM                                1.0
    >REL                                        1.0
        void                                    1.0
    ;
;
```

A.7 Syntaktische Modelle

A.7.1 Übergangsmodell (Sprache) – Auszug

```
#*****************************************************************************
# TRANSITIONS (SYNTACTIC MODEL)      - SPEECH -
#
# Filename:           \Forschung\Interface\TRAIN\HWPAT\Model\mathe_synUeber
# Last Modified:      02/02/27
# Number of Types:    51
#
#***Format:*********************************************************************
# ___________________________________________________________________________
# Type
#  P(sta->end)    P(sta-> C )    P(sta-> B )    P(sta->Aq1)    P(sta->Aq2)    ...
#  P( C ->end)    P( C -> C )    P( C -> B )    P( C ->Aq1)    P( C ->Aq2)    ...
#  P( B ->end)    P( B -> C )    P( B -> B )    P( B ->Aq1)    P( B ->Aq2)    ...
#  P(Aq1->end)    P(Aq1-> C )    P(Aq1-> B )    P(Aq1->Aq1)    P(Aq1->Aq2)    ...
#  P(Aq2->end)    P(Aq2-> C )    P(Aq2-> B )    P(Aq2->Aq1)    P(Aq2->Aq2)    ...
#          ...                                          ...            ...
# ___________________________________________________________________________
#
#***Abbreviations:****************
#
# sta:  Start Node
# end:  End Node
# B:    (+)Element Node
# C:    (-)Element Node
# Aq1:  1st Successor Node
# Aq2:  2nd Successor Node
# ...
#*********************************

DIGIT
       0             0             1             0
       0             0             0             0
       0             0             0             1
       1             0             0             0
...

FRAC
       0             0             0.0502392     0.949761      0
       0             0             1             0             0
       0.217703      0             0             0.0502392     0.732057
       0             0.00956938    0.722488      0             0.267943
       0.782297      0             0.217703      0             0
...

INT
       0             0.105263      0.894737      0             0             0      0
       0             0             1             0             0             0      0
       0             0             0             0             0             1      0
       1             0             0             0             0             0      0
       0             0             0             1             0             0      0
       0             0             0             0             0             0      1
       0             0             0             0             1             0      0
...

REL
       0             0             0             1             0
       0             0             0             0             1
       0             0.0292398     0             0             0.97076
       0             0             1             0             0
       1             0             0             0             0
...

ORDINAL
       0             0             0             1
       0             0             1             0
       1             0             0             0
       0             0.642857      0.357143      0
...
```

A.7.2 Elementmodell (Sprache) – Auszug

```
#*****************************************************************************
# ELEMENT EMISSIONS (SYNTACTIC MODEL)       - SPEECH -
#
# Filename:           \Forschung\Interface\TRAIN\HWPAT\Model\mathe_synEmiss
# Last Modified:      02/02/27
# Number of Types:    51
# Dictionary Size:    281
#
#***Format:*********************************************************************
#
#     SemunType
#       (-)Element                                          Element Probability
#       ...
#       ;
#       SemunValue
#              (+)Element                                   "
#              ...
#              ;
#       SemunValue
#              (+)Element                                   "
#              ...
#              ;
#       ...
#       ;
#
#*****************************************************************************


VAR
      aeh                                                  0.666667
      aehm                                                 0.333333
      ;
      lat_a
       a                                                   1.0
       ;
      ...
      lat_A
       gross_a                                             1.0
       ;
      ...
      gr_delta
       delta                                               0.842105
       dirac                                               0.157895
       ;
      ...
      ;
...

FUNCSPEC
      ;        # No (-)Elements
      func_sin
       sinus_von                                          0.631579
       sinus                                              0.368421
       ;
      func_ln
       ln_von                                             0.736842
       der_natuerliche_logarithmus_von                    0.0526316
       logarithmus_von                                    0.105263
       ln                                                 0.0526316
       logarithmus_naturalis_von                          0.0526316
       ;
...

FUNC
      ;
      func_bare
       -                                                  1.0
       ;
      func_par
       von                                                0.995181
       -                                                  0.00240964
       von_in_klammern                                    0.00240964
       ;
      ;
```

```
...

FRAC
      das_ganze                                        1.0
      ;
      void
       durch                                           0.205742
       geteilt_durch                                   0.30622
       gebrochen_mit                                   0.0239234
       dividiert_durch                                 0.0454545
       gebrochen_durch                                 0.0167464
       bruchstrich                                     0.0430622
       grosser_bruchstrich                             0.0215311
       tel                                             0.174641
       bruch                                           0.00239234
       ;
      ;

ROOT
      aus                                              0.931034
      daraus                                           0.0344828
      aus_dem                                          0.0344828
      ;
      root_bare
       wurzel                                          0.754386
       die_wurzel                                      0.210526
       wurzel_von                                      0.0350877
       ;
      root_pow
       wurzel                                          1.0
       ;
      ;


POW
      ;
      ...
      pow_idx
       -                                               0.561404
       index                                           0.421053
       unten                                           0.0175439
       ;
      ...
      ;
...

DIFF_OPERATOR
      ;
      void
       d                                               0.868421
       nach_d                                          0.0789474
       nach                                            0.0263158
       ueber_d                                         0.0263158
       ;
      ;
...

ORDINAL
      ...
      ;
      second
       die_zweite_quadrat                              0.0714286
       zweite                                          0.428571
       die_quadrat                                     0.214286
       quadrat                                         0.285714
       ;
      third
       dritte                                          1.0
       ;
      fourth
       vierte                                          1.0
       ;
      fifth
       fuenfte                                         1.0
       ;
      ...
      ;
...
```

A.7.3 Offsetmodell (Handschrift)

```
#****************************************************************
# SYNTACTIC OFFSET MODEL (derived from labeled reference formulas)
# test person: rob
#****************************************************************
DIGIT # semun type
        1 # number of pairs of structural components
              # offsetx                    offsety                    offsetg
              # mean        stdev          mean         stdev         mean         stdev
                0.545115    0.032808       -0.033433    0.044067      -0.059230    0.063833
    ;
NAT_NUM
        1
                0.526916    0.030667        0.025838    0.055806      -0.032228    0.071605
    ;
FRAC_NUM
        3                                                          # sym. size feat. omitted
               -0.197097    0.075287       -0.414738    0.041417       0.000000    0.000000
                0.361226    0.066491       -0.396209    0.032868       0.000000    0.000000
                0.558323    0.017296        0.018529    0.040526       0.017951    0.076871
    ;
PARAMLIST
        3
               -0.160124    0.087171       -0.345538    0.041263       0.000000    0.000000
                0.456985    0.044597       -0.318808    0.068242       0.000000    0.000000
                0.617109    0.053806        0.026730    0.046403       0.137083    0.113110
    ;
TERM_PARAMLIST
        3
                0.905175    0.037009       -0.006371    0.050423       0.007766    0.056089
                0.437740    0.066361        0.086478    0.058261      -0.084931    0.165347
               -0.467435    0.065017        0.092849    0.062643      -0.092697    0.153379
    ;
FUNCVAR_POW
        1
                0.539129    0.018117       -0.495143    0.038156      -0.501611    0.043055
        1
                0.418599    0.065620        0.342082    0.049640      -0.575702    0.198670
        3                                # default stdev (only one sample)
                0.347368    0.010000        0.341463    0.010000      -0.365212    0.010000
                0.568421    0.010000       -0.439024    0.010000      -0.503859    0.010000
                0.221053    0.010000       -0.780488    0.010000      -0.138647    0.010000
    ;
FUNCSPEC_POW
        1
                0.495293    0.024278       -0.460972    0.063256      -0.507186    0.085945
    ;
FUNC
        1
                0.584636    0.060678       -0.057980    0.098778      -0.155565    0.124615
    ;
FNIDX
        3
                0.540686    0.008013       -0.048698    0.034568      -0.089408    0.322872
                0.012699    0.009120        0.401270    0.047960      -0.324416    0.334926
               -0.527986    0.006288        0.449968    0.061065      -0.235008    0.140491
    ;
PAR
        3
                0.950851    0.024353       -0.040325    0.061192       0.002948    0.064138
                0.474083    0.021089        0.012291    0.066721      -0.160229    0.173556
               -0.476768    0.018304        0.052616    0.067732      -0.163176    0.168162
    ;
FRAC
        3
               -0.046003    0.059323       -0.249764    0.041300       0.000000    0.000000
               -0.013033    0.065278        0.352834    0.047134       0.000000    0.000000
                0.032970    0.084025        0.602599    0.065712      -0.023281    0.121340
```

```
    ;
ROOT
        1
                0.029745    0.013633        0.030826    0.031136        0.000000    0.000000
        3
                0.498778    0.010000       -0.431655    0.010000        0.000000    0.000000
                0.501222    0.010000        0.043165    0.010000        0.000000    0.000000
                0.002445    0.010000        0.474820    0.010000        0.000000    0.000000
        3
               -0.049057    0.010000       -0.038095    0.010000        0.000000    0.000000
               -0.471698    0.010000       -0.180952    0.010000        0.000000    0.000000
               -0.422642    0.010000       -0.142857    0.010000       -0.197866    0.010000
        6
                0.485522    0.006878       -0.449454    0.042497        0.000000    0.000000
                0.434492    0.029577        0.032086    0.019415        0.000000    0.000000
               -0.051030    0.035170        0.481540    0.024758        0.000000    0.000000
               -0.041489    0.007107       -0.124840    0.017003        0.000000    0.000000
               -0.527011    0.008149        0.324614    0.042077        0.000000    0.000000
               -0.475981    0.035801       -0.156926    0.018079       -0.271300    0.147762
    ;
BINOM
        6
                0.889650    0.001108       -0.064877    0.031994        0.009768    0.062643
                0.419178    0.027122       -0.264261    0.015187       -0.466506    0.023631
               -0.470472    0.026046       -0.199384    0.016806       -0.476273    0.039012
                0.450837    0.040468        0.243222    0.114158       -0.525440    0.050266
               -0.438813    0.039391        0.308099    0.082164       -0.535208    0.012379
                0.031659    0.013346        0.507482    0.098970       -0.058934    0.026635
    ;
FAC
        1
               -0.569257    0.043776       -0.071005    0.062265       -0.111383    0.064035
    ;
INT
        6
                0.809214    0.018770       -0.004219    0.007308        0.000000    0.000000
                0.382226    0.006262        0.011628    0.020140        0.000000    0.000000
               -0.426988    0.012766        0.015847    0.017659       -0.145777    0.036549
                0.928799    0.009853       -0.029117    0.012759        0.000000    0.000000
                0.119585    0.009397       -0.024898    0.010202       -0.176714    0.023726
                0.546573    0.005038       -0.040745    0.027741       -0.030938    0.029699
        10
                0.811189    0.010000        0.039216    0.010000        0.000000    0.010000
                0.398601    0.010000        0.039216    0.010000        0.000000    0.010000
               -0.412587    0.010000        0.000000    0.010000       -0.067849    0.010000
                0.919580    0.010000        0.029412    0.010000        0.000000    0.010000
                0.108392    0.010000       -0.009804    0.010000       -0.044400    0.010000
                0.520979    0.010000       -0.009804    0.010000        0.023449    0.010000
               -0.010490    0.010000        0.617647    0.010000        0.000000    0.010000
               -0.821678    0.010000        0.578431    0.010000       -0.262736    0.010000
               -0.409091    0.010000        0.578431    0.010000       -0.194886    0.010000
               -0.930070    0.010000        0.588235    0.010000       -0.218336    0.010000
        15
                0.862857    0.071683       -0.027474    0.045799        0.000000    0.000000
                0.461434    0.059783       -0.016569    0.028384        0.000000    0.000000
               -0.401423    0.063786        0.010905    0.029037       -0.122051    0.134833
                0.942710    0.037121       -0.021606    0.035727        0.000000    0.000000
                0.079852    0.040063        0.005868    0.019658       -0.031334    0.102176
                0.481276    0.059544       -0.005037    0.022402        0.090717    0.158254
                0.014094    0.017945        0.426568    0.065326        0.000000    0.000000
               -0.848764    0.078637        0.454041    0.086149       -0.370455    0.203240
               -0.447340    0.068861        0.443137    0.073887       -0.248404    0.131047
               -0.928616    0.045559        0.448174    0.085459       -0.339121    0.212729
                0.028834    0.028890       -0.368711    0.036337        0.000000    0.000000
               -0.834023    0.092260       -0.341237    0.047796       -0.331004    0.155694
               -0.432600    0.073766       -0.352142    0.043098       -0.208953    0.117827
               -0.913875    0.061261       -0.347104    0.038701       -0.299670    0.195257
                0.014741    0.020322       -0.795278    0.092960        0.039451    0.105192
    ;
```

```
SUMPROD
        1
                0.000000    0.000000         0.000000    0.000000         0.000000     0.000000
        3
                0.544118    0.010000        -0.057143    0.010000         0.000000     0.010000
                0.004902    0.010000         0.485714    0.010000         0.000000     0.010000
               -0.539216    0.010000         0.542857    0.010000        -0.347709     0.010000
        6
                0.511602    0.021863        -0.057978    0.045744         0.000000     0.000000
                0.007064    0.020333         0.370528    0.038428         0.000000     0.000000
               -0.504537    0.020510         0.428506    0.048293        -0.349981     0.112936
                0.010395    0.015821        -0.420003    0.048362         0.000000     0.000000
               -0.501206    0.028205        -0.362025    0.081381        -0.467230     0.128247
                0.003331    0.025299        -0.790531    0.085293        -0.117249     0.070269
    ;
POW
        1
                0.505163    0.037760       -0.425348     0.070740       -0.317880     0.117236
        1
                0.502271    0.072554        0.437355     0.124309       -0.445893     0.159644
        3
                0.459588    0.101038        0.374115     0.108288       -0.335146     0.070368
                0.544833    0.068125       -0.348209     0.120040       -0.236880     0.111302
                0.085245    0.161281       -0.722323     0.017933        0.098266     0.060618
    ;
PROD
        1
                0.527984    0.055469       -0.008037     0.086636        0.020773     0.135101
        3
               -0.101318    0.004541        0.017257     0.029243        0.000000     0.000000
                0.419683    0.011896       -0.012919     0.026884        0.000000     0.000000
                0.521001    0.007358       -0.030175     0.020046        0.077097     0.101145
    ;
TERM_SGNPROD
        1
                0.557952    0.049787        0.012947     0.079796        0.000000     0.000000
    ;
SUM
        3
               -0.296906    0.113336        0.032252     0.056082        0.000000     0.000000
                0.294872    0.085806       -0.010714     0.044855        0.000000     0.000000
                0.591778    0.073368       -0.042966     0.054298        0.039964     0.087417
    ;
SGN_SUM
        3
               -0.288363    0.092180        0.039353     0.071089        0.000000     0.000000
                0.339012    0.075709        0.011761     0.086503        0.000000     0.000000
                0.627375    0.059640       -0.027593     0.092289       -0.001879     0.114094
    ;
REL
        3
               -0.239960    0.129199        0.027192     0.053438        0.000000     0.000000
                0.371593    0.075442       -0.017385     0.081481        0.000000     0.000000
                0.611553    0.101892       -0.044576     0.089953       -0.124623     0.096135
    ;
```

A.8 DTW-Referenzmuster (Handschrift) – Protokollauszug

```
#****************************************************************
# CONTENTS OF SYMBOL REFERENCE DATA BASE (CREATION LOG FILE)
# test person: joschi
#****************************************************************
Saving hpt file joschi.sym.hpt with 136 pattern classes...
   Saving pattern class "0" with 3 patterns...
   Pattern Nr.: 0 (of class "0")
      Number of Strokes: 1
      Bounding Box: (67, 50), (98, 84)
      Pattern Vector Dimension: 7
      Number of Pattern Vectors: 74
      Number of interpolated Pattern Vectors: 0
      Pattern Vectors:
      # abs. coordinates    local direction     local curvature     pen up/down
     # x         y         cos       sin       cos       sin       ± 0.5
        0.000000  0.529412 -0.980581  0.196117  1.000000  0.000000  0.500000
        0.009804  0.480392 -0.980581  0.196117  0.992278 -0.124034  0.500000
        0.019608  0.431373 -0.948683  0.316228  0.964764 -0.263116  0.500000
        0.039216  0.392157 -0.894428  0.447213  0.926092 -0.377296  0.500000
        0.058824  0.352941 -0.759257  0.650791  0.894427 -0.447215  0.500000
        0.098039  0.323529 -0.600000  0.800000  0.976187 -0.216930  0.500000
        0.137255  0.294118 -0.600000  0.800000  0.997880  0.065080  0.500000
        0.176471  0.264706 -0.650792  0.759256  0.997880  0.065078  0.500000
        0.205882  0.235294 -0.650791  0.759257  0.997880 -0.065080  0.500000
        0.245098  0.205882 -0.600000  0.800000  0.997880 -0.065078  0.500000
        0.284314  0.176471 -0.600000  0.800000  0.997880  0.065080  0.500000
        0.323529  0.147059 -0.650792  0.759256  0.997880  0.065078  0.500000
        0.352941  0.117647 -0.650791  0.759257  0.997880 -0.065080  0.500000
        0.392157  0.088235 -0.599999  0.800001  0.979762 -0.200167  0.500000
        0.431373  0.058824 -0.485642  0.874158  0.948684 -0.316227  0.500000
        0.480392  0.039216 -0.316228  0.948683  0.922442 -0.386137  0.500000
        0.519608  0.029412 -0.110432  0.993884  0.975441 -0.220262  0.500000
        0.568628  0.029412 -0.099502  0.995037  0.996241  0.086628  0.500000
        0.617647  0.019608 -0.196115  0.980581  0.995228  0.097574  0.500000
        0.666667  0.009804 -0.196117  0.980580  0.996241 -0.086628  0.500000
        0.715686  0.000000 -0.110432  0.993884  0.952925 -0.303206  0.500000
        0.754902  0.000000  0.110432  0.993884  0.946260 -0.323408  0.500000
        0.803922  0.009804  0.216932  0.976187  0.977803 -0.209528  0.500000
        0.843137  0.019608  0.316228  0.948683  0.980155 -0.198230  0.500000
        0.892157  0.039216  0.406137  0.913812  0.972082 -0.234641  0.500000
        0.931373  0.058824  0.529999  0.847998  0.903063 -0.429508  0.500000
        0.970588  0.088235  0.759257  0.650791  0.794005 -0.607911  0.500000
        0.990196  0.127451  0.936329  0.351123  0.759257 -0.650791  0.500000
        1.000000  0.166667  1.000000  0.000000  0.849286 -0.527934  0.500000
        0.990196  0.215686  0.980581 -0.196116  0.980580 -0.196118  0.500000
        0.980392  0.264706  0.980580 -0.196118  0.995229  0.097570  0.500000
        0.970588  0.313726  0.995037 -0.099505  0.996240  0.086633  0.500000
        0.970588  0.362745  0.993884 -0.110430  0.975441 -0.220260  0.500000
        0.960784  0.401961  0.948683 -0.316227  0.922440 -0.386141  0.500000
        0.941176  0.450980  0.874157 -0.485644  0.948683 -0.316228  0.500000
        0.911765  0.490196  0.800000 -0.600000  0.990712 -0.135979  0.500000
        0.882353  0.529412  0.800000 -0.600000  1.000000  0.000001  0.500000
        0.852941  0.568628  0.800000 -0.599999  1.000000  0.000000  0.500000
        0.823529  0.607843  0.800000 -0.600000  0.997880 -0.065081  0.500000
        0.794118  0.647059  0.759256 -0.650792  0.989950 -0.141421  0.500000
        0.764706  0.676471  0.707107 -0.707107  0.997055 -0.076695  0.500000
        0.735294  0.705882  0.707107 -0.707107  0.997055 -0.076695  0.500000
        0.705882  0.735294  0.650792 -0.759256  0.989950 -0.141421  0.500000
        0.666667  0.764706  0.600000 -0.800000  0.979762 -0.200166  0.500000
        0.627451  0.794118  0.485644 -0.874157  0.974732 -0.223376  0.500000
        0.578432  0.813726  0.406139 -0.913811  0.996053 -0.088760  0.500000
        0.539216  0.833333  0.406137 -0.913812  0.996053  0.088755  0.500000
        0.490196  0.852941  0.485642 -0.874158  0.996053  0.088760  0.500000
        0.450980  0.882353  0.485644 -0.874157  0.982872 -0.184288  0.500000
        0.401961  0.901961  0.316227 -0.948683  0.922440 -0.386140  0.500000
        0.362745  0.911765  0.110430 -0.993884  0.907960 -0.419056  0.500000
        0.313726  0.911765 -0.110430 -0.993884  0.946261 -0.323403  0.500000
```

```
      0.274510  0.901961 -0.216929 -0.976187  0.977802 -0.209530  0.500000
      0.225490  0.892157 -0.316227 -0.948683  0.970142 -0.242538  0.500000
      0.186275  0.872549 -0.447215 -0.894427  0.955779 -0.294087  0.500000
      0.147059  0.852941 -0.581239 -0.813733  0.883789 -0.467886  0.500000
      0.117647  0.823529 -0.813733 -0.581239  0.883788 -0.467887  0.500000
      0.098039  0.784314 -0.894427 -0.447214  0.979662 -0.200653  0.500000
      0.078431  0.745098 -0.913811 -0.406139  0.989950 -0.141421  0.500000
      0.058824  0.696079 -0.948683 -0.316228  0.953073 -0.302742  0.500000
      0.049020  0.656863 -0.993884 -0.110431  0.857493 -0.514496  0.500000
      0.049020  0.607843 -0.976187  0.216931  0.907960 -0.419058  0.500000
      0.068628  0.568628 -0.948683  0.316228  0.994692 -0.102899  0.500000
      0.078431  0.519608 -0.948683  0.316228  0.995350 -0.096324  0.500000
      0.098039  0.480392 -0.913812  0.406138  0.995350 -0.096324  0.500000
      0.117647  0.431373 -0.913812  0.406138  0.990164 -0.139914  0.500000
      0.137255  0.392157 -0.847999  0.529998  0.958129 -0.286337  0.500000
      0.166667  0.352941 -0.759257  0.650791  0.954275 -0.298930  0.500000
      0.196078  0.323529 -0.650792  0.759256  0.976187 -0.216931  0.500000
      0.235294  0.294118 -0.600000  0.800000  0.988767 -0.149466  0.500000
      0.274510  0.264706 -0.529998  0.847999  0.983870 -0.178886  0.500000
      0.313726  0.245098 -0.447213  0.894428  0.999222 -0.039441  0.500000
      0.352941  0.225490 -0.496140  0.868243  0.998460  0.055472  0.500000
      0.382353  0.205882 -0.496140  0.868243  1.000000  0.000000  0.500000

Pattern Nr.: 1 (of class "0")
   Number of Strokes: 1
   Bounding Box: (44, 43), (64, 73)
   Pattern Vector Dimension: 7
   Number of Pattern Vectors: 80
   Number of interpolated Pattern Vectors: 0
   Pattern Vectors:
      0.066667  0.433333 -1.000000  0.000000  1.000000  0.000000  0.500000
      0.066667  0.400000 -1.000000  0.000000  1.000000  0.000000  0.500000
      0.066667  0.355556 -1.000000  0.000000  1.000000  0.000000  0.500000
      0.066667  0.311111 -1.000000  0.000000  0.989949 -0.141422  0.500000
      0.066667  0.266667 -0.989949  0.141422  0.894427 -0.447214  0.500000
      0.077778  0.233333 -0.894427  0.447214  0.800000 -0.599999  0.500000
      0.100000  0.200000 -0.707107  0.707107  0.868243 -0.496139  0.500000
      0.133333  0.177778 -0.554700  0.832051  0.980581 -0.196115  0.500000
      0.166667  0.155556 -0.554701  0.832050  1.000000 -0.000000  0.500000
      0.200000  0.133333 -0.554700  0.832051  1.000000 -0.000001  0.500000
      0.233333  0.111111 -0.554700  0.832051  0.992278 -0.124034  0.500000
      0.266667  0.088889 -0.447214  0.894427  0.952424 -0.304775  0.500000
      0.300000  0.077778 -0.274721  0.961524  0.976187 -0.216930  0.500000
      0.344445  0.066667 -0.242536  0.970142  0.999445 -0.033316  0.500000
      0.388889  0.055556 -0.242535  0.970143  0.999445  0.033315  0.500000
      0.433333  0.044445 -0.274722  0.961524  0.996815 -0.079743  0.500000
      0.466667  0.033333 -0.164400  0.986394  0.961524 -0.274722  0.500000
      0.500000  0.033333  0.000000  1.000000  0.986394 -0.164400  0.500000
      0.544445  0.033333  0.000000  1.000000  1.000000  0.000000  0.500000
      0.588889  0.033333  0.000000  1.000000  1.000000  0.000000  0.500000
      0.633333  0.033333  0.000000  1.000000  1.000000  0.000000  0.500000
      0.666667  0.033333  0.000000  1.000000  0.989949 -0.141422  0.500000
      0.700000  0.033333  0.141422  0.989949  0.970142 -0.242536  0.500000
      0.744445  0.044445  0.242536  0.970142  0.994692 -0.102898  0.500000
      0.788889  0.055556  0.242535  0.970143  0.999445 -0.033316  0.500000
      0.833333  0.066667  0.274722  0.961524  0.997054 -0.076697  0.500000
      0.866667  0.077778  0.316228  0.948683  0.982872 -0.184288  0.500000
      0.900000  0.088889  0.447214  0.894427  0.894428 -0.447212  0.500000
      0.933333  0.111111  0.707106  0.707108  0.763386 -0.645942  0.500000
      0.955556  0.144445  0.919145  0.393919  0.857492 -0.514498  0.500000
      0.966667  0.188889  0.970143  0.242535  0.991998 -0.126252  0.500000
      0.977778  0.233333  0.961523  0.274723  0.997054  0.076697  0.500000
      0.988889  0.266667  0.948684  0.316227  0.961523 -0.274723  0.500000
      1.000000  0.300000  1.000000  0.000000  0.825308 -0.564682  0.500000
      0.988889  0.333333  0.961524 -0.274720  0.919145 -0.393919  0.500000
      0.977778  0.377778  0.919145 -0.393919  0.952424 -0.304777  0.500000
      0.955556  0.411111  0.832050 -0.554701  0.928475 -0.371394  0.500000
      0.933333  0.444445  0.707105 -0.707109  0.958798 -0.284088  0.500000
      0.900000  0.466667  0.640184 -0.768222  0.995893 -0.090533  0.500000
      0.866667  0.500000  0.640185 -0.768221  0.994309 -0.106532  0.500000
```

```
      0.833333  0.522222  0.554701 -0.832050  0.994309 -0.106534  0.500000
      0.800000  0.544445  0.554700 -0.832051  1.000000 -0.000001  0.500000
      0.766667  0.566667  0.554700 -0.832050  0.992278 -0.124034  0.500000
      0.733333  0.588889  0.447214 -0.894427  0.964764 -0.263117  0.500000
      0.700000  0.600000  0.316229 -0.948683  0.989949 -0.141422  0.500000
      0.666667  0.611111  0.316228 -0.948683  0.999056 -0.043439  0.500000
      0.633333  0.622222  0.274720 -0.961524  0.997055 -0.076696  0.500000
      0.588889  0.633333  0.242536 -0.970142  0.999445 -0.033313  0.500000
      0.544445  0.644445  0.242536 -0.970142  0.999445  0.033313  0.500000
      0.500000  0.655556  0.274720 -0.961524  0.996815 -0.079746  0.500000
      0.466667  0.666667  0.164399 -0.986394  0.961524 -0.274720  0.500000
      0.433333  0.666667  0.000000 -1.000000  0.986394 -0.164399  0.500000
      0.388889  0.666667  0.000000 -1.000000  0.992278 -0.124034  0.500000
      0.344445  0.666667 -0.124034 -0.992278  0.961524 -0.274720  0.500000
      0.300000  0.655556 -0.274720 -0.961524  0.980580 -0.196117  0.500000
      0.266667  0.644445 -0.316228 -0.948683  0.990712  0.135978  0.500000
      0.233333  0.633333 -0.141423 -0.989949  0.980581  0.196117  0.500000
      0.188889  0.633333 -0.124035 -0.992278  0.976575 -0.215176  0.500000
      0.144445  0.622222 -0.351123 -0.936329  0.879543 -0.475819  0.500000
      0.100000  0.600000 -0.581238 -0.813734  0.910366 -0.413803  0.500000
      0.066667  0.566667 -0.707107 -0.707107  0.967459 -0.253029  0.500000
      0.033333  0.533333 -0.768222 -0.640184  0.928477 -0.371391  0.500000
      0.011111  0.500000 -0.919145 -0.393919  0.841695 -0.539953  0.500000
      0.000000  0.455556 -0.992278 -0.124036  0.919145 -0.393919  0.500000
      0.000000  0.411111 -1.000000  0.000000  0.992278 -0.124036  0.500000
      0.000000  0.366667 -1.000000  0.000000  0.986394 -0.164400  0.500000
      0.000000  0.333333 -0.986394  0.164400  0.894427 -0.447214  0.500000
      0.011111  0.300000 -0.894427  0.447214  0.863015 -0.505177  0.500000
      0.033333  0.266667 -0.768222  0.640184  0.973417 -0.229040  0.500000
      0.066667  0.233333 -0.768221  0.640185  0.994309  0.106532  0.500000
      0.088889  0.200000 -0.832050  0.554701  0.994309  0.106534  0.500000
      0.111111  0.166667 -0.832050  0.554700  0.980581 -0.196115  0.500000
      0.133333  0.133333 -0.707107  0.707107  0.923077 -0.384616  0.500000
      0.166667  0.111111 -0.554700  0.832050  0.948683 -0.316228  0.500000
      0.200000  0.088889 -0.447214  0.894427  0.992278 -0.124035  0.500000
      0.233333  0.077778 -0.447214  0.894427  1.000000 -0.000000  0.500000
      0.266667  0.055556 -0.447214  0.894427  0.998274 -0.058722  0.500000
      0.300000  0.044445 -0.393919  0.919145  0.998969 -0.045407  0.500000
      0.344445  0.022222 -0.406139  0.913811  0.999911  0.013333  0.500000
      0.400000  0.000000 -0.406139  0.913811  1.000000  0.000000  0.500000

Pattern Nr.: 2 (of class "0")
   Number of Strokes: 1
   Bounding Box: (62, 82), (70, 93)
   Pattern Vector Dimension: 7
   Number of Pattern Vectors: 39
   Number of interpolated Pattern Vectors: 0
   Pattern Vectors:
      0.181818  0.363636 -0.624692  0.780871  1.000000  0.000000  0.500000
      ...
      0.363636  0.181818 -0.196118  0.980580  1.000000  0.000000  0.500000

Saving pattern class "1" with 3 patterns...
Pattern Nr.: 0 (of class "1")
   Number of Strokes: 1
   Bounding Box: (124, 50), (146, 80)
   Pattern Vector Dimension: 7
   Number of Pattern Vectors: 49
   Number of interpolated Pattern Vectors: 0
   Pattern Vectors:
      0.611765  0.000000  0.554701 -0.832050  1.000000  0.000000  0.500000
      ...
...
```

A.9 Zuordnungstabelle für die MIF-Transformation

Math. Ausdruck	Beispiele	Semuntyp	Semunwerte	MIF-Syntax (Beispiel)
Ziffer	0 … 9	DIGIT	zero, …, nine	num[5,"5"]
Natürliche Zahl	1 … 9	NAT_NUM	one, …, nine	num[1,"1"]
Null	0	ZERO	zero	num[0,"0"]
Bruchzahl	0.12	FRAC_NUM	void	num[0.12,"0.12"]
Dezimalzahl	12.34	NUMBER	void	num[12.34,"12.34"]
Variable (lat.)	a, …, z	VAR	lat_a, …, lat_z	char[a], …, char[z]
	A, …, Z		lat_A, …, lat_Z	char[A], …, char[Z]
Variable (griech.)	α … ϕ φ … θ ϑ … ω	VAR	gr_alpha … gr_phi gr_sm_phi … gr_theta gr_sm_theta … gr_omega	char[alpha] … char[phi] char[varphi] … char[theta] char[vartheta] … char[omega]
	Δ Φ Γ Π Ω Ψ		gr_Delta gr_Phi gr_Gamma gr_Pi gr_Omega gr_Psi	char[Delta] char[Phi] char[Gamma] char[Pi] char[Omega] char[Psi]
Sonderzeichen	∂ ∞	VAR	partial infinity	char[cpartial] char[infty]
Sonderfunktion	sin cos tan cot sinh cosh tanh coth arcsin arccos arctan exp ln log arg	FUNCSPEC	func_sin func_cos func_tan func_cot func_sinh func_cosh func_tanh func_arccos func_arcsin func_arcsin func_arctan func_exp func_ln func_log func_arg	sin cos tan cot sinh cosh tanh coth asin acos atan exp ln log arg
Parameterliste	a,b[38]	PARAMLIST	void	a,b
Parameterlisten-term	(a,..,..) [a,..,..] {a,..,..}	TERM_PARAMLIST	prmlst_() prmlst_[] prmlst_{}	id[comma[a,..,..]] id[(*1*)comma[a,..,..]] id[(*2*)comma[a,..,..]]
Allg. Funktion mit Index und/oder Exponent	$a^b(\ldots)$ $a_c(\ldots)$ $a_c^b(\ldots)$	FUNCVAR_POW	fnvar_pow fnvar_idx fnvar_idxpow	power[a,b] indexes[0,1,a,c] power[indexes[0,1,a,c],b]
Sonderfunktion mit Exponent	$\sin^b a$	FUNCSPEC_POW	fnspec_pow	power[sin[a],b]

[38] a, b, c und x stehen jeweils stellvertretend für laut semantischem Modell erlaubte, im Prinzip beliebig komplexe mathematische Ausdrücke.

Math. Ausdruck	Beispiele	Semuntyp	Semunwerte	MIF-Syntax (Beispiel)
Allg. Funktion	cos a a(b)	FUNC[39]	func_bare func_par	cos[a] times[(*n*)a,id[b]]
Sonderfunktion mit Bereichsangabe	lim a b min a b max a b	FNIDX	fnidx_lim fnidx_min fnidx_max	lim[a,b] newlimit[(*T"min"T*)a,b] newlimit[(*T"max"T*)a,b]
Klammerung	(a) [a] {a} |a|	PAR	par_() par_[] par_{} par_||	id[a] id[(*i1i*)a] id[(*i2i*)a] abs[a]
Bruch	$\frac{a}{b}$	FRAC	void	over[a,b]
Wurzel	$\sqrt{a}$ $\sqrt[b]{a}$	ROOT	root_bare root_pow	sqrt[a] sqrt[a,b]
Binomial-koeffizient	$\binom{a}{b}$	BINOM	void	choice[a,b]
Fakultät	a!	FAC	void	fact[a]
Integral	$\int a\,dx$ $\int_b a\,dx$ $\int_b^c a\,dx$	INT	int_bare int_idx int_idxpow	int[times[(*n*)a,diff[x]]] int[times[(*n*)a,diff[x]],b] int[times[(*n*)a,diff[x]],b,c]
Summation und Produkt	$\sum a \quad \prod a$	SUMPROD	sum_bare prod_bare	sum[a] prod[a]
	$\sum_b a \quad \prod_b a$		sum_idx prod_idx	sum[a,b] prod[a,b]
	$\sum_b^c a \quad \prod_b^c a$		sum_idxpow prod_idxpow	sum[a,b,c] prod[a,b,c]
Exponentiation und Indizierung	a^b a_c a_c^b	POW	pow_pow pow_idx pow_idxpow	power[a,b] indexes[0,1,a,c] power[indexes[0,1,a,c],b]
Multiplikation, Division und Faltung	ab a · b a ÷ b a ⊗ b	PROD	prod_void prod_pnt prod_div prod_conv	times[(*n*)a,b] cdot[(*n*)a,b] div[a,b] otimes[a,b]
Produktterm, ggf. mit Vorzeichen	a -a	TERM_SGNPROD	sgnprod_pos sgnprod_neg	a minus[a]
Addition und Subtraktion	… a + b … a – b a + b ± … a – b ± …	SUM SGN_SUM	sum_plus sum_minus sgnsum_plus sgnsum_minus	… plus[(*n*)a,b] … plus[(*n*)a,minus[b]] plus[(*n*)a,b,…] plus[(*n*)a,minus[b],…]
Relation	a = b a > b a ≥ b a < b a ≤ b a → b	REL	rel_equ rel_gr rel_gre rel_ls rel_lse rel_lftarw	equal[a,b] greaterthan[a,b] geq[a,b] lessthan[a,b] leq[a,b] rightarrow[a,b]

[39] Der Semuntyp FUNC dient zur übergreifenden Festlegung, ob das Argument einer Funktion mit oder ohne Klammerung angegeben wird. Der zugehörige Semunwert func_bare (keine Klammern) tritt demgemäß nur in Verbindung mit Sonderfunktionen (Semuntyp FUNCSPEC oder FUNCSPEC_POW) auf.

Stichwortverzeichnis

Symbolverzeichnis

α_{nk}	Offsetwahrscheinlichkeiten im Nachfolgerknoten A_{nk} (zum Syntaktischen Modul SM_n)
β_{nl}	Elementwahrscheinlichkeiten im (+)Elementknoten B_{nl} (zu SM_n)
γ_n	Elementwahrscheinlichkeiten im (-)Elementknoten C_n (zu SM_n)
$\delta_n^{(1)}, \delta_n^{(2)}$	Schreibrichtungs- und Krümmungswinkel im Abtastpunkt n
$\Delta_n = \left[\delta_n^{az}\right]$	Matrix der Übergangswahrscheinlichkeiten von Ausgangs- zu Zielknoten $a \rightarrow z$ (zu SM_n)
$\Delta_i(e_h)$	Länge (in Linienzügen) der Linienzuggruppe ab Linienzugposition i, die dem Element e_h eines syntaktischen Gesamtausdrucks Ξ zugeordnet wird
η_0	Wurzelwahrscheinlichkeit für den Wurzelsemuntyp $s_1.t$
η_n	Folgewahrscheinlichkeit für die Nachfolgerschar r des Semuns s_n
ε_n	Wertwahrscheinlichkeit für den Semunwert $s_n.v$
$\varepsilon_x, \varepsilon_y, \varepsilon_g$	Binäre Gewichtungsfaktoren für die Komponentenfusion
g_S	Komponentenspezifische mittl. Schriftgröße (rekursive Offsetberechnung)
$\lambda_x, \lambda_y, \lambda_g$	Symbolspezifische Gewichtungsfaktoren für Mittelpunkt und Schriftgröße
λ_f	Formelstrukturgewicht
L_i	Linienzug, bestehend aus der i-ten zusammenhängenden Merkmalvektorfolge bei aufgesetztem Stift (Stiftstatus p = +½) der Beobachtungsfolge O
$L_i^{(\delta_i)}$	Linienzuggruppe, bestehend aus den δ_i nächsten Linienzügen ab Position i
$\vec{o}_{ij} = \left[\Delta_{ij}\tilde{x}, \Delta_{ij}\tilde{y}, \Delta_{ij}\tilde{g}\right]^T$	Offset-Vektor aus Positionierungs- und Skalierungsmerkmalen zum Strukturkomponentenpaar ij
$O = \left(\vec{O}_1, \cdots, \vec{O}_i, \cdots, \vec{O}_I\right)$	Beobachtungsfolge der Gesamtlänge I, bestehend aus Merkmalvektoren $\vec{O}_i$
$p_n = \pm½$	Stiftstatus (aufgesetzt oder angehoben) im Abtastpunkt n
$P(O\|\Phi)$	Bedingte Wahrscheinlichkeit für das Auftreten der Beobachtungsfolge O zur Phonem-/Visemfolge Φ
$P(\Phi\|\Xi)$	Bedingte Wahrscheinlichkeit für das Auftreten der Phonem-/Visemfolge Φ zum syntaktischen Ausdruck Ξ
$P(S)$	A-priori-Wahrscheinlichkeit für das Auftreten des Bedeutungsinhalts S

$P(\Xi \mid S)$	Bedingte Wahrscheinlichkeit für das Auftreten des syntaktischen Ausdrucks Ξ zum Bedeutungsinhalt S
$r = \{r_{n1}, \cdots, r_{nX}\}$	Nachfolgerschar von Semuntypen zum Semun s_n
$R = (\vec{R}_1, \cdots, \vec{R}_i, \cdots, \vec{R}_I)$	Referenzmuster der Gesamtlänge I, bestehend aus Merkmalvektoren $\vec{R}_i$
$S = \{s_n\},\ 1 \le n \le N$	Semantische Gliederung zur Repräsentation des Bedeutungsinhalts S, bestehend aus N Semunen s_n
S_E	Ergebnis der semantischen Decodierung (beste Semantische Gliederung)
$SN = \{SM_n\}, 1 \le n \le N$	Syntaktisches Netzwerk zur Semantischen Gliederung S, bestehend aus N Syntaktischen Modulen SM_n
t	Semuntyp
v	Semunwert
$\bar{x}_S, \bar{y}_S$	Komponentenspezifischer Mittelpunkt für die rekursive Offsetberechnung
$X(s_n.t,\ s_n.v) \ge 1$	Semantische Valenz (Nachfolgeranzahl) zum Semun s_n
$Y(s_n.t,\ s_n.v) \ge 0$	Anzahl der (+)Elementknoten B_{nl} im Syntaktischen Modul SM_n

Literatur

[ADO97] ADOBE Systems Inc.: FrameMaker 5.5 MIF Reference. Online Manual, San Jose, CA, USA, 1997.

[ALT02] ALTHOFF, F.; AL-HAMES, M.; MCGLAUN, G.; LANG, M.: *Towards a New Approach for Integrating Multimodal User Input Based on Evolutionary Computation*. Tagungsband „IEEE International Conference on Acoustics, Speech, and Signal Processing“ (ICASSP) 2002, Orlando, FL, USA, 13.-17.05.2002, IEEE Signal Processing Society, CD-ROM.

[AND68] ANDERSON, R. H.: *Syntax-directed Recognition of Handprinted Two-dimensional Mathematics*. In M. KLERER UND J. REINFELDS (Hrsg.), Interactive Systems for Experimental Applied Mathematics, S. 436-459, Academic Press, New York, USA, 1968.

[BEN00] BENOIT, B.; MARTIN, J.-C.; PELACHAUD, C.; SCHOMAKER, L.; SUHM, B.: *Audio-visual and Multimodal Speech-based Systems*. In D. GIBBON, I. MERTINS, R. K. MOORE (Hrsg.), Handbook of Standards and Resources for Spoken Language Systems, Kluwer Academic Publishers, 2000.

[BLO95] BLOSTEIN, D.: *General Diagram-Recognition Methodologies*. Tagungsband „International Workshop on Graphics Recognition“ (IWGR) 1995 University Park, PA, USA, 10./11.8.1995, S. 200-212.

[BLO97] BLOSTEIN, D.; GRBAVEC, A.: *Recognition of Mathematical Notation*. In P.S.P. WANG und H. BUNKE (Hrsg.), Handbook on Optical Character Recognition and Document Image Analysis, Kap. 21, S. 557-582, World Scientific Publishing, 1997.

[BOL80] BOLT, R. A.: *Put-that-there: Voice and Gesture at the Graphics Interface*. Computer Graphics Journal of the Association of Computing and Machinery, Bd. 14, No. 3, S. 262-270, 1980.

[CHA00] CHAN, K. F.; YEUNG, D. Y.: *Mathematical Expression Recognition: A Survey*. International Journal on Document Analysis and Recognition (IJDAR), Bd. 3, No. 1, 2000, S. 3-15.

[CHO59] CHOMSKY, N.: *On Certain Formal Properties of Grammars*. Information and Control, Bd. 2, No. 2, S. 137-167, Academic Press Inc., Elsevier Science, Juni 1959.

[CHO89] CHOU, P. A.: *Recognition of Equations Using a Two-dimensional Stochastic Context-free Grammar*. Tagungsband SPIE Conference on Visual Communications and Image Processing IV, Philadelphia, PA, USA, November 1989, Bd. 1199, S. 852-863.

[DIM95] DIMITRIADIS, Y. A.; CORONADO, J. L.: *Towards an ART Based Mathematical Editor that uses on-line handwritten symbol recognition*. Pattern Recognition, Bd. 28, No. 6, 1995, S. 807-822.

[DUT96] DUTOIT, T.; PAGEL, V.; PIERRET, N.; VAN DER VREKEN, O.; BATAILLE, F.: *The MBROLA Project: Towards a Set of High-Quality Speech Synthesizers Free of Use for Non-Commercial Purposes*. Tagungsband „4th International Conference on Spoken Language Processing“ (ICSLP 1996), Philadelphia, PA, USA, 3.-6.10.1996, Bd. 3, S. 1393-1396.

[GAZ85] GAZDAR, G.; KLEIN, E.; PULLUM, G. K.; SAG, I. A.: *Generalized Phrase Structure Grammar*. Basil Blackwell, Oxford, England, und Harvard University Press, Cambridge, MA, USA, 1985.

[GEI01] GEIGER, M.; NIESCHULZ, R.; ZOBL, M.; NEUSS, R.; LANG, M.: *Methods for Facilitation of Wizard-of-Oz Studies and Data Acquisition*. Tagungsband „9th International Conference on Human-Computer Interaction“ (HCI International 2001), New Orleans, LA, USA, 5.-10.8.2001, „Poster Sessions: Abridged Proceedings“, S. 191-193.

[GRB95] GRBAVEC, A.; BLOSTEIN, D.: *Mathematics Recognition Using Graph Rewriting*. Tagungsband „3rd International Conference on Document Analysis and Recognition“ (ICDAR 1995), Montreal, Kanada, 14.-15.8.1995, S. 417-421.

[GRO97] GROß, R.: *Run-On Recognition in an On-line Handwriting Recognition System*. Studienarbeit, Interactive Systems Lab, Carnegie Mellon University, Pittsburgh, PA, USA / Universität Karlsruhe, Juni 1997.

[HUL96] HULL, J. F.: *Recognition of Mathematics Using a Two-dimensional Trainable Context-free Grammar*. Master’s Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Juni 1996.

[HUN00A] HUNSINGER, J.; LANG, M.: *A Speech Understanding Module for a Multimodal Mathematical Formula Editor*. Tagungsband „IEEE International Conference on Acoustics, Speech, and Signal Processing“ (ICASSP) 2000, Istanbul, Türkei, 5.-9.6.2000, Bd. 4, S. 2413-2416.

[HUN00B] HUNSINGER, J.; LANG, M.: *A Single-Stage Top-Down Probabilistic Approach towards Understanding Spoken and Handwritten Mathematical Formulas*. Tagungsband „6th International Conference on Spoken Language Processing“ (ICSLP 2000), Peking, China, 16.-20.10.2000, Bd. 4, S. 386-389.

[HUN01A] HUNSINGER, J.; LIEB, R.; LANG, M.: *Real-Time Structural Analysis of Handwritten Mathematical Formulas by Probabilistic Context-Free Geometry Modelling*. Tagungsband „2001 International Symposium on Intelligent Multimedia, Video & Speech Processing“ (ISIMP 2001), Hong Kong, China, 2.-4.5.2001, S. 52-55.

[HUN01B] HUNSINGER, J.: *An Intuitive Pen-Gestural Interface for Syntactic-Semantic Annotation of Non-Cursive Handwritten Input*. Tagungsband „9th International Conference on Human-Computer Interaction“ (HCI International 2001), New Orleans, LA, USA, 5.-10.8.2001, Bd. 1, „Usability Evaluation and Interface Design“, S. 268-271.

[INT98] INTEL Corporation: INTEL Recognition Primitives Library Reference Manual. Best.-Nr. 637785-007, 1998.

[JÄG98] JÄGER, S.: *Recovering Dynamic Information from Static, Handwritten Word Images*. Dissertation, Fakultät für Angewandte Wissenschaften, Albert-Ludwigs-Universität Freiburg, 1998.

[KAZ02] KAZMAIER, H.: *Usability-Untersuchungen zur multimodalen Erfassung mathematischer Formeln*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 2002.

[KOS98] KOSMALA, A.; RIGOLL, G.: *On-Line Handwritten Formula Recognition Using Statistical Methods*. Tagungsband „14th Intern. Conference on Pattern Recognition“ (ICPR 1998) 1998, Brisbane, Australien, 16.-20.8.1998, S. 1306-1308.

[KOS99] KOSMALA, A.; RIGOLL, G.; LAVIROTTE, S.; POTTIER; L.: *On-Line Handwritten Formula Recognition using Hidden Markov Models and Context Dependent Graph Grammars*. Tagungsband „5th International Conference on Document Analysis and Recognition“ (ICDAR 1999), Bangalore, Indien, 20.-22.9.1999, S. 107-110.

[KOS00] KOSMALA, A.: *HMM-basierte Online Handschrifterkennung: ein integrierter Ansatz zur Text- und Formelerkennung*. Dissertation, Fakultät für Ingenieurwissenschaften, Gerhard-Mercator-Universität Duisburg, 2000.

[KUN00] KUNZE, J.: Morphologie. Manuskript zur Vorlesung, Lehrstuhl für Computerlinguistik, Humboldt-Universität zu Berlin, Stand WS 2000/01.

[LAN02] LANG, M.: *1. Sinnesorgane und Sinnesmodalitäten. 2. Interaktionsmodelle und Dialogformen.* 28 S. In: Handbuch der Ergonomie, Bd. 4, C-8.1.: Interaktion zwischen Mensch und Computer. Hrsg.: Bundesamt für Wehrtechnik und Beschaffung, Koblenz. Carl Hanser Verlag, München, 2002, 6. Ergänzungslieferung.

[LIE98] LIEB, R.: *Integration neuer Hidden-Markov-Modelle in ein sprachverstehendes System.* Studienarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 1998.

[LIE00] LIEB, R.: *Einstufig-probabilistische semantische Decodierung handgeschriebener mathematischer Formeln.* Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 2000.

[MAN94] MANKE, S.; FINKE, M.; WAIBEL, A.: *Combining Bitmaps with Dynamic Writing Information for On-Line Handwriting Recognition*. Tagungsband „12th International Conference on Pattern Recognition“ (ICPR 1994), Jerusalem, Israel, 9.-13.10.1994, S. 596-598.

[MAN95] MANKE, S.; FINKE, M.; WAIBEL, A.: *NPen++: A Writer Independent, Large Vocabulary On-Line Cursive Handwriting Recognition System*. Tagungsband „3rd International Conference on Document Analysis and Recognition“ (ICDAR 1995), Montreal, Kanada, 14.-15.8.1995, S. 403-408.

[MAR97] MARIANI, J. J.: *Spoken Language Processing and Multimodal Communication: A View from Europe*. Plenarvortrag, NSF Workshop on Human-Centered Systems: Information, Interactivity, and Intelligence (HCS), Arlington, VA, USA, 17.-19.2.1997.

[MAT99] MATSAKIS, N. E.: *Recognition of Handwritten Mathematical Expressions.* Master's Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Mai 1999.

[MAY00] MAYER, K.: *Erstellung eines Transformationsmoduls zur Überführung semantischer Gliederungen in MIF-Dokumente*. Fachpraktikumsbericht, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 2000.

[MBR02] MBROLA Project. Development Team Faculte Polytechnique de Mons, MULTITEL-TCTS Lab, Initialis Scientific Park, 1, Copernic Ave, B-7000 Mons, Belgien, http://tcts.fpms.ac.be/synthesis/mbrola.html, letzte Revision 08.05.2002.

[MIL98] MILLER, E. G.; VIOLA, P. A.: *Ambiguity and Constraint in Mathematical Expression Recognition*. Tagungsband „15th National Conference on Artificial Intelligence“ (AAAI-98), Madison, WI, USA, S. 784-791, Juli 1998.

[MÜL97] MÜLLER, J.: *Die semantische Gliederung zur Repräsentation des Bedeutungsinhalts innerhalb sprachverstehender Systeme.* Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 1997.

[MÜL99] MÜLLER, J.; STAHL, H.: *Speech Understanding and Speech Translation by Maximum a-posteriori Semantic Decoding*. Artificial Intelligence in Engineering, Bd. 13, No. 4, S. 373-384, Elsevier Science, 1999.

[MÜL02] MÜLLER, K.: *Vereinheitlichung eines syntaktisch-semantischen Modells zur Transformation handgeschriebener in gesprochene mathematische Formeln.* Bachelorarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 2002.

[NEU02] NEUSS, R.; HUNSINGER, J.; STENZEL, R.; LANG, M.: *Sprachgesteuerte Fahrerassistenz durch einstufig-probabilistisches Verstehen natürlich gesprochener Sprache.* VDI-Berichte 1678, Useware 2002 (Mensch-Maschine-Kommunikation/Design), Darmstadt, 11./12.6.2002, VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, VDI Verlag Düsseldorf, S. 55-60.

[NIE94] NIELSEN, J.: Usability Engineering. Morgan Kaufmann Publishers, San Francisco, CA, USA, Oktober 1994.

[NIG93] NIGAY, L.; COUTAZ, J.: *A Design Space for Multimodal Interfaces: Concurrent Processing and Data Fusion.* Tagungsband INTERCHI 1993, Amsterdam, Niederlande, S. 172-178, ACM Press, 1993.

[OVI94] OVIATT, S. L.; OLSEN, E.: *Integration Themes in Multimodal Human-Computer Interaction*. Tagungsband „3rd International Conference on Spoken Language Processing“ (ICSLP 1994), Acoustical Society of Japan, Yokohama, Japan, September 1994, Bd. 2, S. 551-554.

[OVI00] OVIATT, S. L.; COHEN, P. R.; WU, L.; VERGO, J.; DUNCAN, L.; SUHM, B.; BERS, J.; HOLZMAN, T.; WINOGRAD, T.; LANDAY, J.; LARSON, J.; FERRO, D.: *Designing the User Interface for Multimodal Speech and Gesture Applications: State-of-the-art Systems and Research Directions.* Human-Computer Interaction (Journal), Bd. 15, No. 4, S. 263-322, Lawrence Erlbaum Associates Inc., Mahwah, NJ, USA, 2000.

[PFA00] PFAU, T.: *Methoden zur Erhöhung der Robustheit automatischer Spracherkennungssysteme gegenüber Variationen der Sprechgeschwindigkeit.* Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2000.

[POR00] PORTELE, T.: *Txt2pho - German TTS Front End for the MBROLA Synthesizer*. Institut für Kommunikationsforschung und Phonetik, Universität Bonn, Poppelsdorfer Allee 47, D-53115 Bonn, http://www.ikp.uni-bonn.de/dt/forsch/phonetik/hadifix/HADIFIXforMBROLA.html, letzte Revision 4.7.2000.

[RÅD97] RÅDE, L.; WESTERGREN, B.; VACHENAUER, P.: Springers mathematische Formeln. 2. Auflage, Springer-Verlag Berlin/Heidelberg, 1997.

[RUS94] RUSKE, G.: Automatische Spracherkennung – Methoden der Klassifikation und Merkmalsextraktion. Oldenbourg-Verlag, München, Wien, 1994.

[RUS97] RUSKE, G.: Automatische Mustererkennung in der Sprachverarbeitung. Kurzmanuskript zur Vorlesung, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, Stand SS 1997.

[RUS98] RUSKE, G.; FALTLHAUSER, R.; PFAU, T.: *Extended Linear Discriminant Analysis (ELDA) for Speech Recognition*. Tagungsband „5th International Conference on Spoken Language Processing“ (ICSLP 1998), Bd. 3, S. 1095-1098, Sydney, Australien, Dezember 1998.

[RUS95] RUSSEL, S.; NORVIG, P.: Artificial Intelligence - A Modern Approach. Prentice Hall, 1995.

[SCH99] SCHULLER, B. W.: *Automatisches Verstehen gesprochener mathematischer Formeln.* Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 1999.

[SCH00] SCHÜTZENDORF, M.: *Integration eines DTW-Musterbewertungsverfahrens in einen statistischen semantischen Decoder zur Erkennung handgeschriebener mathematischer Formeln.* Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 1999.

[SED92] SEDGEWICK, R.: Algorithmen in C++. Addison-Wesley, 1992.

[SMI99] SMITHIES, S.; NOVINS, K.; ARVO, J.: *A Handwriting-Based Equation Editor*. Tagungsband „Graphics Interface“ (GI) 1999, Kingston, Ontario, Kanada, Juni 1999, S. 84-91.

[STA97A] STAHL, H.; MÜLLER, J.; LANG, M.: *Controlling Limited-Domain Applications by Probabilistic Semantic Decoding of Natural Speech*. Tagungsband „IEEE International Conference on Acoustics, Speech, and Signal Processing“ (ICASSP) 1997, München, April 1997, S. 1163-1166.

[STA97B] STAHL, H.: *Konsistente Integration stochastischer Wissensquellen zur semantischen Decodierung gesprochener Äußerungen.* Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 1997.

[STE99] STENZEL, R.: *Natürlichsprachliche Eingabe für das Automobil.* Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 1999.

[TON02] TONG, T. Y.: *Forward and Backward Transformation of Semantically Structured Mathematical Formulae to Maker Interchange Format (MIF).* Studienarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 2002.

[VO97] VO, M. T.; WAIBEL, A.: *Modeling and Interpreting Multimodal Inputs: A Semantic Integration Approach*. Technical Report CMU-CS-97-192, Carnegie Mellon University, Pittsburgh, PA, USA, Dezember 1997.

[WEL99] WELCH, B. B.: Practical Programming in Tcl and Tk. ISBN 0-13-022028-0, Prentice-Hall, New Jersey, USA, 1999.

[WEL02] WELLS, J.: *SAMPA [„s{mpA:] Speech Assessment Methods Phonetic Alphabet*. Department of Phonetics and Linguistics, University College London, Gower Street, London WC1E 6BT, http://www.phon.ucl.ac.uk/home/sampa/home.htm, letzte Revision 02.07.2002.

[WIN96] WINKLER, H.-J.: *HMM-Based Handwritten Symbol Recognition Using on-line and off-line Features.* Tagungsband „IEEE International Conference on Acoustics, Speech, and Signal Processing“ (ICASSP) 1996, Atlanta, GA, USA, 7.-10.5.1996, S. 3438-3441.

[WIN97A] WINKLER, H.-J.; LANG, M.: *Symbol Segmentation and Recognition for Understanding Handwritten Mathematical Expressions*. in A.C. DOWNTON und S. IMPEDOVO (Hrsg.), Progress in Handwriting Recognition, Tagungsband „5th International Workshop on Frontiers in Handwriting Recognition" (IWFHR-5), Essex, England, September 1996, S. 407-412, World Scientific, 1997.

[WIN97B] WINKLER, H.-J.: *Entwurf und Realisierung eines auf statistischen Ansätzen basierenden Systems zur Erkennung handgeschriebener mathematischer Formeln.* Dissertation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 1997.

[WU99] WU, L.; OVIATT, S. L.; COHEN, P. R.: *Multimodal Integration - A Statistical View.* IEEE Transactions on Multimedia, Bd. 1, No. 4, Dezember 1999, S. 334-341.

[ZHA00] ZHANG, Y.; LEVINSON, S.; HUANG, T.: *Speaker Independent Audio-visual Speech Recognition*. Tagungsband „IEEE International Conference on Multimedia and Expo“ (ICME) 2000, New York, USA, 30.7.-2.8.2000, Bd. 2, S. 1073-1076.

[ZWI99] ZWICKER, E.; FASTL, H.: Psychoacoustics – Facts and Models. 2., überarbeitete Auflage, Springer-Verlag Berlin/Heidelberg, 1999.