Lehrstuhl für Mensch-Maschine-Kommunikation
der Technischen Universität München

Ein generischer Ansatz zur Integration multimodaler Benutzereingaben

Frank Althoff

Lehrstuhl für Mensch-Maschine-Kommunikation
der Technischen Universität München

Ein generischer Ansatz zur Integration multimodaler Benutzereingaben

Frank Althoff

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.-Ing. Klaus Diepold
Prüfer der Dissertation	1.	Univ.-Prof. Dr.rer.nat. Manfred K. Lang, i.R.
	2.	Univ.-Prof. Gudrun J. Klinker Ph.D.
	3.	Univ.-Prof. Dr.-Ing Eckehard Steinbach

Die Dissertation wurde am 18.3.2004 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 17.9.2004 angenommen.

Vorwort

Die vorliegende Arbeit entstand während meiner Zeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Mensch-Maschine Kommunikation in der Fakultät für Elektrotechnik und Informationstechnik an der Technischen Universität München. Der primäre Dank gilt meinem Doktorvater Prof. Manfred Lang für die Möglichkeit, in einem inspirativen Umfeld sowohl an theoretisch fundierten Grundlagenkonzepten als auch an praxisbezogenen Anwendungen im Rahmen industrieller Kooperationsprojekte arbeiten zu können. Darüber hinaus möchte ich mich bei Prof. Gerhard Rigoll für wertvolle Anregungen im letzten Jahr meiner Tätigkeit am Lehrstuhl bedanken, bei Prof. Gudrun Klinker und bei Prof. Eckehard Steinbach für die Erstellung der weiteren Gutachten und bei Prof. Klaus Diepold für die Leitung der Prüfungskommission.

Ein spezieller Dank gebührt meinem langjährigen Zimmerkollegen und Forschungspartner Gregor McGlaun für die hochgradig effektive Zusammenarbeit in verschiedenen Projekten, die Umsetzung einer gemeinsamen Systemplattform, die Vermittlung einer interessanten Mischung aus bayrischer und texanischer Lebenskultur und nicht zuletzt für geschmeidige, weiß-blaue Doppelpässe. In diesem Zusammenhang möchte ich mich auch bei allen wissenschaftlichen Kollegen für ein überaus angenehmes Arbeitsklima und die konstruktiven Diskussionen bedanken, vor allem bei Björn Schuller für die produktive Teamarbeit im FERMUS-Projekt.

Weiterhin bedanke ich mich bei den Festangestellten des Lehrstuhls für die perfekte Unterstützung. Explizit hervorheben möchte ich die Leistungen von Heiner Hundhammer, Ernst Ertl und Peter Brand im Hinblick auf die Hilfe bei technischen Problemen und bei der Reparatur von essentiellen Arbeitsutensilien. An dieser Stelle danke ich herzlich auch allen Studenten, die in Form von unterschiedlichen Arbeiten insgesamt einen wichtigen Teil zu dem Gelingen der einzelnen Forschungsvorhaben beigetragen haben. Ein weiterer Dank gebührt den zahlreichen Teilnehmern der Benutzerstudien und verschiedenen Personen aus meinem persönlichen Umfeld für eine konstruktive Durchsicht ausgewählter Teile der vorliegenden Arbeit.

Ganz besonders bedanke ich mich auf diesem Weg noch einmal bei meiner Freundin Martina für ihre positiv motivierende Art und ihre Geduld. Eine signifikante Unterstützung der kreativen Denkprozesse ist zudem ihrem traumhaften Schokokuchen und den Bananen-Muffins zuzuschreiben, die in gleichem Maße von alten und neuen Kollegen mit einem absoluten Premium-Prädikat versehen worden sind. Abschließend möchte ich mich noch bei D. Adams, H.F. Storm und T. Kaserer für diverse Inspirationen auch außerhalb des universitären Alltags bedanken.

München, im Februar 2004 Frank Althoff

Kurzzusammenfassung

Im Rahmen der Entwicklung interaktiver Mensch-Maschine-Systeme kommt der Konzeption von effektiv und intuitiv bedienbaren Benutzerschnittstellen eine besondere Bedeutung zu. Wachsende funktionale Komplexität und die Beschränkung auf taktile Interaktionsformen haben zu langen Einarbeitungsphasen und zu Frustrationen bei den Benutzern geführt. Durch die stärkere Annäherung an zwischenmenschliche Kommunikationsgewohnheiten bieten multimodale Ansätze einen potenziellen Ausweg aus diesem Dilemma. In der vorliegenden Arbeit wird ein generischer Ansatz für die Realisierung entsprechender Systeme vorgestellt.

Den Ausgangspunkt bildet eine detaillierte Usability-Studie zu multimodalen Interaktionsparadigmen in zwei verschiedenen Applikationsdomänen. Vor dem Hintergrund einer parallelen Verfügbarkeit der fünf Eingabemodalitäten Touchscreen, Tastatur, Handgestik, Kopfgestik und Sprache werden sowohl unimodale als auch multimodale Systeminteraktionen untersucht. Der Fokus liegt auf einer Betrachtung der modalitätenspezifischen Interaktionszeiten, der intermodalen Zeitrelationen, der funktionalen Informationszusammensetzungen und der auftretenden, temporalen Überlagerungsmuster. Aus den Ergebnissen werden fundamentale Anforderungen für das Design multimodaler Benutzerschnittstellen abgeleitet.

Die vorgestellte Systemarchitektur operiert auf einer abstrakten, symbol-orientierten Beschreibungsebene. Proprietäre Benutzereingaben und Kontextinformationen werden mittels gerätespezifischer Konvertermodule in eine einheitliche, semantische Darstellung transformiert. Durch die formale Modellierung ist der Kern der Architektur in Bezug auf die verwendeten Erkennermodule frei skalierbar und kann darüber hinaus auf unterschiedliche Applikationsszenarien übertragen werden. Ein zentrales Integratormodul bildet die einzelnen Informationsanteile auf eine spezifische Systemfunktionalität ab. Charakteristisch ist der modulare Aufbau des Integrators aus einer symbolverarbeitenden Schicht, einer nachfolgenden Kontext-Evaluation und einem abschließenden Fehlermanagement. Die einzelnen Komponenten können jeweils über eine domänenspezifische Wissensbasis extern konfiguriert werden.

Den algorithmischen Schwerpunkt der Arbeit bildet die Einführung eines innovativen, hybriden Integrationsverfahrens für die Interpretation multimodaler Eingabedaten. Angelehnt an die Prinzipien der natürlichen Evolution konkurrieren dabei jeweils mehrere Hypothesen miteinander. In einem iterativen Optimierungsprozess werden einzelne Lösungsalternativen selektiert, rekombiniert und durch problem-adäquate genetische Operatoren manipuliert. Der maßgebliche Vorteil besteht darin, dass die funktionale Zusammensetzung aus redundanten, komplementären und rivalisierenden Informationsanteilen bereits implizit durch die Struktur des Algorithmus unterstützt wird. In Kombination mit einer regelbasierten Vorverarbeitung für die zeitliche Segmentierung zusammengehöriger Ereignisse ermöglicht dieser genetische Fusionsansatz einen flexiblen, intuitiven und zugleich fehlerrobusten Mensch-Maschine-Dialog.

Eine detaillierte Evaluierung des entwickelten Gesamtsystems vor dem Hintergrund realer und fiktiver Bediensituationen aus unterschiedlichen Applikationsdomänen zeigt das enorme Leistungspotenzial auf. Nach einer Anpassung der strukturellen Parameter lassen sich auf Standard-PC-Hardware auch unter Realzeitbedingungen optimale Integrationsergebnisse erzielen. Die praktische Anwendung der Systemarchitektur und des hybriden Integrationsverfahrens werden abschließend anhand von mehreren Demonstratoren unter Beweis gestellt.

Inhaltsverzeichnis

KAPITEL 1

Einleitung

Zwei Menschen stehen sich an einem trüben und verregneten Novembernachmittag an einer dicht befahrenen Straße in München gegenüber. Die eine Person ist eine 75jährige, leicht schwerhörige Frau aus dem Allgäu und die andere Person ist ein 32jähriger Mann aus Nordfriesland, der unter extremer Kurzsichtigkeit leidet. Der Mann fragt die Frau nach dem Weg zum Liebfrauendom. Die Frau schaut irritiert und macht eine entschuldigende Handgeste in Richtung ihrer Ohren. Der Mann nickt, wartet geduldig, bis ein lautes Sport-Motorrad vorbeigefahren ist und wiederholt seine Anfrage, dieses Mal aber etwas lauter und langsamer. Daraufhin deutet die Frau in eine Richtung hinter den beiden und sagt, dass man dazu an dem kleinen Maroni-Stand mit dem roten Dach nach links abbiegen muss. Dieses Mal zuckt der Mann mit den Achseln und weist auf seine dicken Brillengläser hin. Die Frau lächelt verständnisvoll und erklärt etwas ausführlicher, dass sich die Abzweigung zu der Kirche nach ungefähr einem Kilometer direkt hinter dem großen Geschäft eines bekannten Herrenausstatters befindet. Der Mann lächelt zurück, bedankt sich für die Auskunft und geht los.

In Hamburg gastiert seit April 2000 die international renommierte Ausstellung mit dem Titel *Dialog im Dunkeln, Entdeckung des Unsichtbaren* [61]. Die Besucher werden jeweils in kleinen Gruppen von einem blinden bzw. stark sehbehinderten Mitarbeiter durch eine Reihe von absolut lichtlosen Räumen geführt und erleben alltägliche Situationen wie den Spaziergang in einem Wald, den Einkauf auf einem Gemüsemarkt oder den Besuch einer Gaststätte unter vollkommen veränderten Randbedingungen. Aufgrund der komplett fehlenden visuellen Informationen verlassen sich die Teilnehmer nach einer anfänglichen Phase der Unsicherheit im Laufe der Zeit immer stärker auf ihren Gehör-, Geruchs- und Tastsinn. Neben einem Blindenstock bietet vor allem die ständige Kommunikation mit den anderen Besuchern und dem Tour-Guide eine wichtige Orientierungshilfe und trägt massiv zu dem individuellen Wohlbefinden bei. Durch die ungewohnten Erfahrungen werden von den meisten Besuchern nach dem Rundgang viele Kleinigkeiten im normalen Tagesablauf deutlich intensiver wahrgenommen.

Diese beiden Beispiele verdeutlichen eindrucksvoll die Mächtigkeit und zugleich die Robustheit der zwischenmenschlichen Kommunikation. Durch die parallele Auswertung unterschiedlicher Kommunikationskanäle kann sich der Mensch optimal auf den jeweiligen Kommunikationspartner einstellen und flexibel auf aktuelle Umgebungsänderungen reagieren. Darüber hinaus verfügt der Mensch über einen umfangreichen Erfahrungsschatz, der durch ständige Lern- und Anpassungsprozesse kontinuierlich erweitert wird. Von dieser Art der Kommunikation ist der Umgang mit einem technischen System noch weit entfernt.

1.1 Problembeschreibung

Im Rahmen der Entwicklung interaktiver Mensch-Maschine-Systeme besteht eine wesentliche Problematik darin, die zur Verfügung stehenden System-Funktionalitäten dem Anwender auf einfache und intuitive Weise zugänglich zu machen. Durch den kontinuierlich steigenden Funktionsumfang werden die Benutzer jedoch mit immer länger werdenden Einarbeitungszeiten konfrontiert. Darüber hinaus sind zumindest bei normalen Bildschirmarbeitsplätzen die Interaktionsmöglichkeiten immer noch stark limitiert. Die Eingaben erfolgen zumeist haptisch mittels Maus und Tastatur und die Ausgaben werden nur in visueller Form präsentiert. Vor dem Hintergrund komplexer Bedienabläufe führen diese Einschränkungen oftmals zu Frustrationen und einer partiell ablehnenden Haltung gegenüber technischen Systemen.

Eine potenzielle Lösung bietet die Verwendung von *multimodalen* Interfaces, bei denen mehrere Möglichkeiten bestehen, Informationen bilateral zwischen dem Benutzer und dem System auszutauschen. Im Vergleich zu *unimodalen* Benutzerschnittstellen, bei denen lediglich ein spezifischer Kanal für die Kommunikation genutzt werden kann, verspricht man sich von multimodalen Systemansätzen deutliche Vorteile. Durch die stärkere Annäherung an zwischenmenschliche Kommunikationsgewohnheiten kann neben kürzeren Lernphasen und einem erhöhten Bedienkomfort vor allem eine signifikante Steigerung der Fehlerrobustheit erzielt werden. Multimodalität impliziert jedoch nicht zwingend den Zusammenschluss möglichst vieler Modalitäten. Vielmehr kommt es darauf an, die einzelnen Interaktionskanäle vor dem Hintergrund einer spezifischen Bedienaufgabe sinnvoll zu kombinieren.

Die Umsetzung eines multimodalen Interaktionsparadigmas stellt erhöhte Anforderungen an das Design von Mensch-Maschine-Systemen. Es müssen sowohl architekturale als auch algorithmische Probleme gelöst werden, die bei der Verwendung konventioneller Schnittstellentechnologien nicht oder nur in viel geringerem Maße auftreten. Um die einzelnen Äußerungen und Handlungen des Benutzers vor dem Hintergrund der möglichen Interaktionen mit dem System korrekt interpretieren zu können, ist es nötig, die Daten von sämtlichen, zur Verfügung stehenden Informationsquellen miteinander zu verbinden und eine entsprechende Systemantwort zu generieren. Dieser Prozess wird zusammenfassend als *multimodale Integration* bezeichnet.

Bereits in der Entwurfsphase eines multimodalen Interaktionssystems stellt sich die Frage, inwiefern der Informationsfluss zwischen den beteiligten Eingabe-, Verarbeitungs- und Ausgabekomponenten koordiniert werden kann. Grundsätzlich wird zwischen zwei fundamentalen Problemen unterschieden. Eine wesentliche Aufgabe besteht zunächst darin, in dem multimodalen Informationsspektrum zusammengehörige Ereignisse zu identifizieren und wichtige von weniger relevanten Anteilen zu trennen. Dieser initiale Prozess wird als *Spotting* bezeichnet. Bezogen auf rein temporale Aspekte treten die Daten oftmals mit gewissen Latenzzeiten auf, die sowohl durch den Benutzer selbst als auch durch evtl. unterschiedliche Systemlaufzeiten von vorverarbeitenden Modulen verursacht werden können.

In der nachfolgenden *Fusion* werden die Benutzereingaben auf entsprechende Systemaktionen abgebildet und dem Anwender über geeignete Ausgabegeräte multimedial präsentiert. Dazu muss die temporale und die funktionale Zusammensetzung der Eingabedaten analysiert und im Kontext der aktuellen Bediensituation und der Zielapplikation interpretiert werden. Die Fusion der einzelnen Informationsanteile ist prinzipiell auf unterschiedlichen Abstraktionsebenen möglich. Speziell bei der Auswertung erkennergestützter Eingabemodule können die Daten noch mit divergierenden Konfidenzwerten behaftet sein.

1.2 Lösungsansatz

Das primäre Ziel der vorliegenden Arbeit besteht darin, ein generisches Konzept für die Verarbeitung multimodaler Benutzereingaben zu entwickeln. Zur Lösung der im letzten Abschnitt beschriebenen Probleme wird ein mehrstufiges Vorgehen propagiert. Insgesamt werden dazu drei größere, strukturell aufeinander aufbauende Themenbereiche behandelt. Den Ausgangspunkt bildet eine umfangreiche Benutzerstudie zum multimodalen Eingabeverhalten in zwei verschiedenen Applikationsdomänen. Die Resultate dieser Untersuchung fließen in Form einer benutzerzentrierten Anforderungsdefinition unmittelbar in das Design und die Implementierung einer geeigneten Basisarchitektur ein. Das zentrale Element dieser Architektur ist ein anwendungsunabhängiges, hybrides Integrationsmodul für die kontextsensitive Interpretation semantisch-dekodierter, multimodaler Informationsanteile. Abschließend wird das entwickelte Gesamtsystem vor dem Hintergrund realer und fiktiver Bediensituationen evaluiert und die praktische Nutzbarkeit anhand von verschiedenen Demonstratoren unter Beweis gestellt.

Detaillierte Benutzerstudien

Benutzer moderner Informations- und Kommunikationssysteme unterscheiden häufig nicht zwischen der zugrundeliegenden Systemfunktionalität und der dazugehörigen Benutzungsoberfläche. Die Bedienbarkeit einer Anwendung hat einen wesentlichen Einfluss auf die Beurteilung und die Akzeptanz des Gesamtsystems. Aus diesem Grund werden Anwender als elementare Entscheidungsträger immer stärker in die einzelnen Phasen des Entwicklungsprozesses eingebunden. Speziell im Rahmen der Konzeption interaktiver Applikationen mit mehreren Ein- und Ausgabemöglichkeiten kommt einer Untersuchung des Benutzerverhaltens im Vorfeld der eigentlichen Systemumsetzung eine besondere Bedeutung zu.

Um domänenübergreifende Anforderungen für das Design einer generischen Basisarchitektur ableiten zu können, werden in den Benutzerstudien zwei verschiedene Szenarien betrachtet. Die erste Anwendung konzentriert sich auf die Navigation in dreidimensionalen, virtuellen Welten in einer desktop-orientierten Arbeitsumgebung. Bei der zweiten Anwendung handelt es sich um die Bedienung von verschiedenen Diensten in einer Automobilumgebung.

Vor dem Hintergrund einer parallelen Verfügbarkeit der fünf Eingabemodalitäten Touchscreen, Tastatur, Handgestik, Kopfgestik und Sprache werden in beiden Szenarien nach einem weitgehend identischen Versuchsplan sowohl unimodale als auch multimodale Systeminteraktionen untersucht. Der Fokus liegt auf einer Betrachtung der modalitätenspezifischen Interaktionszeiten, der intermodalen Zeitrelationen, der funktionalen Informationszusammensetzungen und der auftretenden, temporalen Überlagerungsmuster. Durch die Analyse der Modalitätenkombinationen und der individuellen Modalitätenübergänge können darüber hinaus prototypische Verhaltensweisen ausgewählter Benutzergruppen ermittelt werden.

Entwurf einer generischen Basisarchitektur

Als zentrales Motiv wird ein generischer Ansatz zur Behandlung von symbol-orientierten, multimodalen Informationsanteilen verfolgt, der in der Lage ist, Informationen aus beliebig vielen Quellen zu verarbeiten. Daher können auch jeder Zeit weitere Geräte in das System integriert werden. Die Steuerung der Applikation basiert auf der Verwendung eines abstrakten Beschreibungsformalismus in Form einer kontextfreien Grammatik.

Proprietäre Benutzereingaben und Kontextinformationen werden mittels modulspezifischer Konvertermodule in eine semantische Darstellung transformiert. Die zugrundeliegende formale Sprache ermöglicht eine domänen- und geräte-invariante Repräsentation multimodaler Informationsanteile. Auf diese Weise können sowohl taktile Systeminteraktionen mittels Maus, Tastatur und Touchscreen als auch natürliche Kommunikationsformen wie spontansprachliche Äußerungen und dynamische Hand- und Kopfgesten übergreifend durch einen einheitlichen Formalismus beschrieben werden. Ein weiterer Vorteil besteht in der freien Skalierbarkeit der Architektur im Hinblick auf die Anzahl und den Typ der angeschlossenen Ein- und Ausgabemodule.

Die Verwendung einer formalen Sprache als Schnittstelle zwischen den einzelnen Komponenten erlaubt eine Konfiguration des Systems auf abstrakter Ebene. Das Spektrum der modellierten Funktionen und die Verarbeitungslogik lassen sich ad hoc auch ohne detaillierte Kenntnisse der technischen Spezifika modifizieren. Die einzelnen Komponenten können zudem über eine domänenspezifische Wissensbasis vollständig extern konfiguriert werden.

Das wichtigste Element der Systemarchitektur ist das Integratormodul. Charakteristisch ist der modulare Aufbau dieser Einheit aus drei, in sich verschachtelten Verarbeitungsebenen: einer initialen *Symbolfusion*, einer nachfolgenden *Kontext-Evaluation* und einem abschließenden *Fehlermanagement*. Der Fokus der vorliegenden Arbeit liegt auf einer effektiven Umsetzung der Symbolfusion. Die Identifikation, Klassifikation und Behandlung von Fehlern im multimodalen Mensch-Maschine-Dialog erfolgt in der Arbeit von McGlaun [93]. In Bezug auf die Basisarchitektur ergänzen sich die beiden Arbeiten zu einem konsistenten Gesamtsystem. Eine Überschneidung stellt der Umgang mit verschiedenen Kontextinformationen dar.

Hybrider Integrationsansatz

Für die Interpretation der multimodalen Eingabedaten wird ein innovativer, hybrider Integrationsansatz vorgestellt. Angelehnt an die Prinzipien der natürlichen Evolution konkurrieren dabei jeweils mehrere Integrationshypothesen miteinander. In einem iterativen Optimierungsprozess werden die einzelnen Lösungsalternativen unter Einbezug von domänenspezifischem Wissen bewertet und durch problem-adäquate, genetische Operatoren manipuliert. Gegenüber etablierten Integrationsansätzen besteht der maßgebliche Vorteil dieses Verfahrens darin, dass die funktionale Zusammensetzung aus redundanten, komplementären und rivalisierenden Informationsanteilen implizit durch die Struktur des Algorithmus modelliert wird.

Das Spottingverfahren und der Fusionsprozess sind funktional entkoppelt und können analog zu den anderen Komponenten der Architektur einfach ausgetauscht und modifiziert werden. Für die Identifikation zusammengehöriger Benutzereingaben und Kontextinformationen wird auf der Basis der Erfahrungen aus den Benutzerstudien eine Menge von anwendungsspezifischen Regeln definiert, die sich extern parametrisieren und erweitern lassen. Als Ergebnis der Vorverarbeitung wird ein abgeschlossenes Integrationsintervall aus zeitlich und semantisch assoziierten multimodalen Informationsanteilen abgeleitet, auf dem der nachfolgende genetische Fusionsprozess aufsetzt. Insgesamt ermöglicht dieser Ansatz einen flexiblen, intuitiven und zugleich fehlerrobusten Mensch-Maschine-Dialog.

Um die Leistungsfähigkeit der beiden Integrationsphasen an die Bedürfnisse im praktischen Einsatz anzupassen, werden die entwickelten Module einer zweistufigen Evaluierung unterzogen. Das Ergebnis der ersten Phase dient lediglich dazu, den Wertebereich der verschiedenen strukturellen Parameter im Hinblick auf existierende zeitliche Randbedingungen einzuschränken. In der zweiten Phase werden die identifizierten Konfigurationen dann in Bezug auf die Laufzeit und das jeweilige Erkennungsergebnis optimiert.

1.3 Gliederung der Arbeit

Die vorliegende Arbeit besteht aus fünf größeren, thematisch zusammengehörigen Blöcken. In Ergänzung zu dem aktuellen Abschnitt enthält das folgende Kapitel eine Einführung zu grundlegenden Begriffsterminologien, dem Stand der Technik bei der Entwicklung von Benutzerschnittstellen und dem Prinzip der multimodalen Interaktion. In diesem Zusammenhang wird auch kurz auf relevante Vorarbeiten am Lehrstuhl für Mensch-Maschine-Kommunikation der Technischen Universität München eingegangen.

Kapitel 3 und Kapitel 4 behandeln den Aufbau und die Ergebnisse der Benutzerstudien zur Untersuchung multimodaler Interaktionsmuster. Es werden die beiden Applikationsdomänen *Interaktion in virtuellen Welten* und *Infotainmentsysteme im Automobil* betrachtet. Der Fokus liegt auf einer umfassenden Diskussion der Gemeinsamkeiten und der Unterschiede sowohl bei der Planung und Durchführung als auch bei der Auswertung der aufgezeichneten Daten. Spezifische Informationen zu den verwendeten Fragebögen, den untersuchten Ereignisklassen und weiteren Ergebnissen sind in den Anhang A ausgelagert.

In Kapitel 5 wird ein Konzept für eine generische multimodale Basisarchitektur vorgestellt. Den Ausgangspunkt bilden eine Analyse existierender Systeme und eine Identifikation zentraler Architektur-Anforderungen, die sich zum Teil aus den Benutzerstudien ableiten. Einen wesentlichen Teil des Kapitels nimmt eine Beschreibung der einzelnen Integrationskomponenten ein, wobei auch auf den zugrundeliegenden, abstrakten Kommunikationsformalismus eingegangen wird. Hintergrundinformationen zu formalen Grammatiken und den verwendeten Darstellungskonventionen finden sich in Anhang B.

Den algorithmischen Kern der Arbeit stellt das Kapitel 6 dar. Auf Basis einer kurzen Einführung in die Grundlagen evolutionärer Algorithmen wird das hybride Integrationskonzept für die Verarbeitung multimodaler Benutzereingaben ausführlich diskutiert. Dabei werden getrennt voneinander sowohl die einzelnen Phasen der regelbasierten Vorverarbeitung als auch die strukturellen Komponenten der genetischen Datenfusion betrachtet. Auf der Basis einer problemadäquaten natürlichen Kodierung erfolgt die Herleitung einer quantitativen Bewertung für die einzelnen Integrationshypothesen.

Kapitel 7 diskutiert eine Evaluierung der beiden Integrationsphasen vor dem Hintergrund fiktiver Bediensituationen und aufgezeichneter Eingabemuster aus den Benutzerstudien. In einer Simulationsumgebung werden dazu unterschiedliche Parameterkonfigurationen im Hinblick auf Laufzeit- und Konvergenzverhalten untersucht. Das nachfolgende Kapitel 8 stellt verschiedene Demonstratoren vor, die auf Basis der multimodalen System-Architektur entwickelt worden sind. Den Schwerpunkt bildet die Beschreibung eines multimodalen Virtual-Reality-Browsers. Eine kurze Einführung in die verwendete VRML Sprachstruktur wird im Anhang C erläutert. Abschließend enthält Kapitel 9 eine gesamthafte Diskussion der Arbeit.

KAPITEL 2

Thematische Einführung

Dieses Kapitel beinhaltet eine Einführung in die Thematik der multimodalen Mensch-Maschine-Interaktion. Es vermittelt das nötige Basiswissen für das Verständnis der Begriffe und Methoden, die in den nachfolgenden Kapiteln verwendet werden. Im Einzelnen werden die Bedeutung fundamentaler Bezeichnungen in einem einheitlichen Kontext erklärt, prinzipielle Ein- und Ausgabekanäle identifiziert, auf formale Modelle zur Beschreibung der Mensch-Maschine-Kommunikation eingegangen und die historische Entwicklung desktop-bezogener Benutzerschnittstellen angesprochen. Multimodale Interaktionssysteme bilden den Schwerpunkt für den Rest des Kapitels. Nach der Einführung grundlegender Taxonomien werden das Potenzial multimodaler Interaktionen motiviert und einzelne Ergebnisse aus Benutzerstudien präsentiert. Darauf aufbauend werden einzelne Probleme bei der Integration von unterschiedlichen Benutzereingaben thematisiert, mögliche Lösungsstrategien aufgezeigt und ausgewählte multimodale Referenzsysteme vorgestellt. Ein spezieller Fokus dieses Kapitels liegt abschließend auf der Abgrenzung der vorliegenden Arbeit von anderen Arbeiten aus dem Umfeld des Lehrstuhls.

Die Darstellung der Grundlagen ist bewusst sehr kompakt gehalten. Im Hinblick auf ausführlichere Informationen sei der interessierte Leser auf die jeweiligen Literaturangaben verwiesen. Als Alternative bieten sich für eine einführende Darstellung in zusammenhängender Form zudem verschiedene Lehrbücher an, beispielsweise [40, 75, 113].

2.1 Grundlegende Begriffsterminologie

Vor dem Hintergrund der kontinuierlichen Weiterentwicklung der Informationstechnologie und der gleichzeitig zunehmenden Etablierung sowohl im privaten als auch im beruflichen Alltag kommt der Kommunikation zwischen technischen Systemen und dem Menschen als Benutzer dieser Systeme eine besondere Bedeutung zu. Für die Analyse, Konzeption und Evaluierung der entsprechenden Systemschnittstellen hat sich im Laufe der Zeit ein eigenständiger, hochgradig interdisziplinärer Forschungsbereich etabliert, an dem neben den klassischen Ingenieurwissenschaften, den Arbeitswissenschaften, der Ergonomie und der Informatik auch die Medizin und die Psychologie beteiligt sind. Die Wichtigkeit der Thematik zeigt sich auch an der Vielzahl internationaler Konferenzen (z.B. [1, 2, 3]), die speziell Fragestellungen zum Umgang mit technischen Systemen diskutieren.

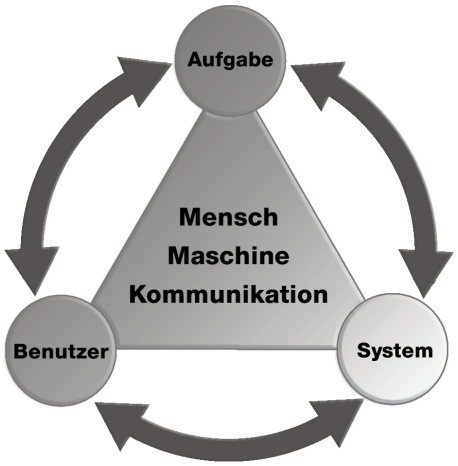


Abb. 2.1: Spektrum der Mensch-Maschine-Kommunikation

Im Folgenden wird zunächst auf verschiedene grundlegende Begriffe eingegangen. Wenn in irgendeiner, prinzipiell beliebigen Art und Weise menschliche Arbeitshandlungen mit der Benutzung einer Maschine verbunden sind, so spricht man allgemein von einem *Mensch-Maschine-System* (MMS)[75]. Unter dem Begriff der *Arbeit* wird in diesem Zusammenhang jede zielgerichtete Handlung des Menschen verstanden, um in einer existierenden Umgebung eine vorgegebene Aufgabe zu erfüllen. Das MMS stellt dabei die zentrale Verbindung zwischen Aufgabenstellung und Aufgabenerfüllung dar. In einem MMS verfolgt der Mensch immer bestimmte Ziele, wie beispielsweise die möglichst effiziente Durchführung einer spezifischen Aufgabe oder die Erzielung einer hohen Konformität in Bezug auf existierende Anforderungen.

In der Arbeitspsychologie wird der Begriff der *Maschine* abstrakt als eine technische Anlage definiert, die zur Realisierung eines stofflich-energetischen Prozesses dient [67]. Dabei kann es sich als Extrembeispiel auf der einen Seite um ein komplexes, in sich geschlossenes System wie etwa ein Raumschiff und auf der anderen Seite des Spektrums um ein einfaches Werkzeug wie beispielsweise einen Schraubendreher handeln. Im Kontext der vorliegenden Arbeit steht der Computer als modernes Arbeitsmittel eindeutig im Vordergrund der Betrachtungen. Daher werden aus Gründen einer vereinfachten Darstellung im Folgenden die Begriffe *Maschine*, *Computer* und *technisches System* synonym verwendet.

In der Literatur findet man häufig auch die Terminologie *Mensch-Maschine-Interaktion* (MMI) bzw. *Mensch-Maschine-Kommunikation* (MMK). Diese beiden Bezeichnungen fungieren im gleichen Maße als fachspezifischer Überbegriff für wissenschaftliche Arbeiten, die sich sowohl auf theoretischer als auch auf pragmatischer Ebene mit der Kommunikation zwischen Mensch und Maschine beschäftigen. Um die Bedeutung des Computers zu betonen wird oftmals auch der Begriff *Mensch-Computer-Interaktion* (MCI) verwendet. Wie Abbildung 2.1 konzeptuell skizziert, stellt das Dreieck aus den zentralen Instanzen Benutzer, System und Aufgabe und die wechselseitig zwischen den einzelnen Komponenten existierenden Relationen auf abstrakter Ebene das Spektrum der MMK dar.

Die deutschen Bezeichnungen sind an die international etablierten, angloamerikanischen Fachbegriffe *Human-Machine-Interaction* (HMI) und *Human-Computer-Interaction* (HCI) angepasst. Hinsichtlich der genauen Definition besteht jedoch keine absolute Einigkeit. Viel-

mehr fließen je nach Forschungsdisziplin unterschiedliche Aspekte ein. Eine oft zitierte Angabe stammt von der ACM Special Interest Group on Computer-Human Interaction unter der Leitung von Thomas Hewett[64]:

> Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.

Im Laufe der Zeit ist der Mensch als entscheidender Faktor für die Akzeptanz und den erfolgreichen Einsatz eines Systems immer stärker in den Entwicklungsprozess eingebunden worden [113]. Geprägt von den Erkenntnissen über menschliche Tätigkeiten und Leistungspotenziale aus verschiedenen Teilgebieten der Arbeitswissenschaften bezeichnet man die Disziplin zur Konzeption von computerbasierten Mensch-Maschine Systemen auch als *Software-Ergonomie*. Diese Forschungsrichtung beschäftigt sich mit der Analyse, Gestaltung und Evaluation von Informations- und Kommunikationssystemen, wobei der Mensch als Benutzer des Systems mit seinen individuellen Fähigkeiten und seinen sozialen Bedürfnissen im Mittelpunkt der Betrachtungen steht. Das Ziel besteht darin, die Arbeitsbedingungen bei der Mensch-Computer-Interaktion möglichst optimal an den Benutzer anzupassen.

Ein weiterer, essentieller Begriff im Kontext der MMK ist die Bezeichnung *Mensch-Maschine-Schnittstelle*. Direkt übertragen aus dem englischen Fachjargon wird dafür in rein deutschsprachigen Veröffentlichungen auch zunehmend der Begriff *Mensch-Maschine-Interface* benutzt. Als Abkürzungen finden sich die Buchstabenfolgen MMS bzw. MMI. Mit den beiden Begriffen werden Arbeiten assoziiert, die sich vor allem auf die Grenzfläche zwischen Mensch und Maschine beziehen. Im Gegensatz zu den Bezeichnungen Mensch-Maschine-System und Mensch-Maschine-Interaktion, die tendenziell eher auf abstrakter Ebene unterschiedliche Fragestellungen zur Konzeption und Struktur des Informationsaustausches behandeln, steht im Bereich der Mensch-Maschine-Schnittstelle speziell der technische Prozess der Kommunikation und die Gestaltung des Dialogs im Vordergrund.

Eine scharfe Trennung der einzelnen Fachbegriffe und den damit verbundenen Thematiken wird jedoch nicht konsistent durchgehalten. Vielmehr variiert die spezifische Bedeutung der Bezeichnungen in den unterschiedlichen Forschungsarbeiten. Die Vermischung der einzelnen Nomenklaturen geht zum Teil soweit, dass die Begriffe synonym gebraucht werden. Zusätzlich wird die Abgrenzung durch den Umstand erschwert, dass die Abkürzungen nicht eindeutig sind. Allein auf Basis der textuellen Form kann a priori nicht entschieden werden, was beispielsweise durch die Abkürzung MMI referenziert werden soll. Im Kontext der vorliegenden Arbeit bezieht sich die Abkürzung MMI daher immer speziell auf die Schnittstelle zwischen Mensch und Maschine und die Abkürzung MMK steht für die verschiedenen Aspekte der Interaktion im weiteren Sinne.

Informationsbegriff

Im Gegensatz zu der physischen Benutzung eines Werkzeugs oder dem Ablesen einer Anzeigetafel im Fußballstadion ist das Zusammenwirken von Mensch und Maschine im Rahmen der Bedienung rechnergestützter Systeme maßgeblich durch den bilateralen Austausch von *Informationen* geprägt. Nach Wiener [169] wird Information neben Materie und Energie häufig als dritter Grundbegriff der technischen Wissenschaften genannt. Für die Diskussion eines MMS

spielt die formale Beschreibung der kommunizierten Information daher eine zentrale Rolle. Wissenschaftlich ist der Informationsbegriff jedoch nicht einheitlich definiert. Je nach betrachteter Forschungsrichtung existieren verschiedene Darstellungsformen [137].

In der nachrichtentechnisch geprägten Definition von Shannon [148, 149] ist Information ein rein technisches Maß. Zur Untersuchung der Nachrichten auf einem evtl. gestörten Kanal zwischen einem Sender und einem Empfänger werden vorwiegend Methoden aus der mathematischen Statistik eingesetzt. Der Informationsgehalt eines übertragenen Zeichens hängt von der Wahrscheinlichkeit ab, mit der dieses Zeichen auftritt. Essentielle Voraussetzung für eine korrekte Übertragung ist die Übereinstimmung der Zeichenalphabete von Sender und Empfänger. Der Ansatz eignet sich zwar auf rein technischer Ebene zur vollständigen Beschreibung eines MMS, berücksichtigt aber weder Bedeutung, noch Wirkung der übertragenen Zeichen. Für eine umfassende Behandlung der verschiedenen Probleme im Kontext der Interaktion zwischen Mensch und Maschine ist der Informationsbegriff nach Shannon nicht ausreichend.

Aus diesem Grund wird in der vorliegenden Arbeit ein Informationsbegriff verwendet, der maßgeblich aus der Künstlichen Intelligenz (KI) stammt [33, 56]. In dem zugrundeliegenden Ansatz besteht eine inhärente Kopplung zwischen Informationen und deren Inhalt. Die dazugehörige Terminologie unterscheidet zwischen *Zeichen*, *Daten*, *Informationen* und *Wissen*. Zeichen sind die atomaren Bestandteile eines vorgegebenen Alphabets, beispielsweise die Elemente des ASCII-Codes. Daten entstehen durch die syntaktisch korrekte Kombination von einzelnen Zeichen in maschinell lesbarer Form. Informationen ergeben sich durch die Einbettung von Daten in einem konkreten Sinnzusammenhang. Auf diese Weise lassen sich Aussagen zu einem spezifischen Sachverhalt formal darstellen. Wissen resultiert aus der Interpretation und Kombination von vorhandenen Informationen. Um die Lesbarkeit der Arbeit zu erhöhen wird im Folgenden nicht näher zwischen der Daten- und der Informationsebene unterschieden. Die beiden Begriffe werden zu einer Kategorie zusammengefasst und synonym verwendet.

Durch den Fokus auf den Informationsaustausch unterscheidet sich die Kommunikation mit einem Computer deutlich von dem Umgang mit Maschinen im klassischen Sinne. Der Mensch kann den Informationsfluss aktiv steuern und erhält entsprechende Rückmeldungen von der Maschine. Darüber hinaus erfolgt der Informationsaustausch zumindest teilweise in vermittelter und interpretierter Form. Allgemein sind nicht alle Informationen über die Maschine, den Prozess und die Umgebung für den Menschen direkt wahrnehmbar. Durch den Informationsfluss koppelt sich der Benutzer an das System und es entsteht ein geschlossener Regelkreislauf. Auf die Schnittstelle zwischen den einzelnen Komponenten in diesem Kreislauf wird im Folgenden näher eingegangen.

2.2 Mensch-Maschine-Kommunikation

2.2.1 Ein- und Ausgabekanäle

Der Mensch ist in der Lage, über unterschiedliche Kanäle Informationen aufzunehmen, zu verarbeiten, zu bewerten und andere Informationen wieder abzugeben. Auf diese Weise ist es ihm möglich, mit anderen Menschen subjektiv einfach, effektiv und fehlerrobust zu kommunizieren. Durch die parallele Nutzung mehrerer Kanäle kann sich der Mensch an die jeweiligen Fähigkeiten des Gesprächpartners anpassen und flexibel auf die aktuelle Umgebung einstellen.

Zur Informationsaufnahme verfügt der menschliche Organismus über mehrere Möglichkeiten. Die Reize aus der Umwelt werden über *Sinne* bzw. entsprechende *Sinnesorgane* aufgenommen und interpretiert [137]. Grundsätzlich kann zwischen sechs verschiedenen Sinnen differenziert werden: Sehsinn, Hörsinn, Geruchssinn, Geschmackssinn, Gleichgewichtssinn und Haut- bzw. Tastsinn. Unter der letzten Kategorie werden mehrere Sinne wie beispielsweise Drucksinn, Kältesinn oder Schmerzsinn subsumiert. Mit diesen Sinnen können nach Geiser [51] jeweils einzelne *Sinnesmodalitäten* assoziiert werden, um die Art der aufgenommenen Information näher zu beschreiben. In der Reihenfolge der Sinne wird zwischen *visuellen*, *auditiven*, *olfaktorischen*, *gustatorischen*, *vestibulären* und schließlich *taktilen* Wahrnehmungskanälen unterschieden.

Bezogen auf die Abgabe von Informationen verfügt der Mensch ebenfalls über ausgeprägte Fähigkeiten. Das Spektrum der möglichen Kanäle für eine intentionale zwischenmenschliche Kommunikation ist im Vergleich zur Informationsaufnahme jedoch deutlich kleiner. Informationen werden entweder über den auditiven, den visuellen oder den taktilen Kanal übertragen. Dabei wird vor allem Sprache, oftmals gekoppelt mit entsprechender Gestik und Mimik verwendet. Zusätzlich können Informationen auch durch direkten Körperkontakt vermittelt werden. Generell spricht man in Bezug auf die Informationsaufnahme von *sensorischen* und in Bezug auf die Informationsabgabe von *motorischen* Fähigkeiten.

Ein technisches System ist sowohl in Bezug auf die Aufnahme als auch auf die Abgabe von Informationen wesentlich stärker eingeschränkt. Verglichen mit dem menschlichen Organismus stehen entweder deutlich weniger Kommunikationskanäle zur Verfügung oder die Kanäle sind vom Leistungspotenzial her massiv reduziert. Die Komplexität der vermittelten Informationen ist durch die sensorischen und motorischen Fähigkeiten des technischen Systems limitiert und im Allgemeinen immer noch signifikant kleiner als bei der zwischenmenschlichen Kommunikation. Der Austausch von Informationen zwischen Mensch und Maschine vor dem Hintergrund dieser deutlich divergierenden Randbedingungen stellt das Kernproblem der MMK dar.

Aus dem Spektrum der möglichen Sinnesmodalitäten sind für die technische Umsetzung vor allem der taktile, der visuelle und der auditive Kanal relevant. Die Nutzung der anderen Kanäle bezieht sich lediglich auf einzelne, ausgewählte Spezialanwendungen und kann für die allgemeine Diskussion vernachlässigt werden. Während bei den meisten aktuellen Computersystemen die Eingabe noch primär über den taktilen Kanal erfolgt, beispielsweise durch die Betätigung von Maus, Tastatur oder spezifischen Schalterflächen, ist die Ausgabe maßgeblich durch die Nutzung des visuellen Kanals in Form von Bildschirmanzeigen geprägt. Die Bedeutung des visuellen Kanals für die Eingabe und die des auditiven Kanals für Ein- und Ausgabe nimmt jedoch immer mehr zu. Bezogen auf die Informationsausgabe können nach Geiser [51] verschiedene Anzeigearten identifiziert werden. Für den visuellen Kanal spricht man von *optischer*, für den auditiven Kanal von *akustischer* und für den taktilen Kanal von einer *haptischen* Anzeige.

Interagiert ein Benutzer mit einem technischen System, so kann nach Schomaker [141] der entstehende Informationsfluss zwischen den beiden Kommunikationspartnern schematisch als Regelkreislauf dargestellt werden. Abbildung 2.2 veranschaulicht die einzelnen Informationskanäle, die über das Mensch-Maschine-Interface als zentrale Grenzfläche miteinander verbunden sind. Insgesamt wird zwischen vier primären Kanälen unterschieden. Der Benutzer gibt Informationen über seine Motorik weiter (HOC: *Human Output channels*), die von der Sensorik des Systems erfasst und verarbeitet werden (CIM: Computer Input Modalities). Auf der anderen Seite liefert das technische System über verschiedene Ausgabemedien Informationen zurück an den Benutzer (COM: Computer Output Media), die wiederum von dessen Sensorik

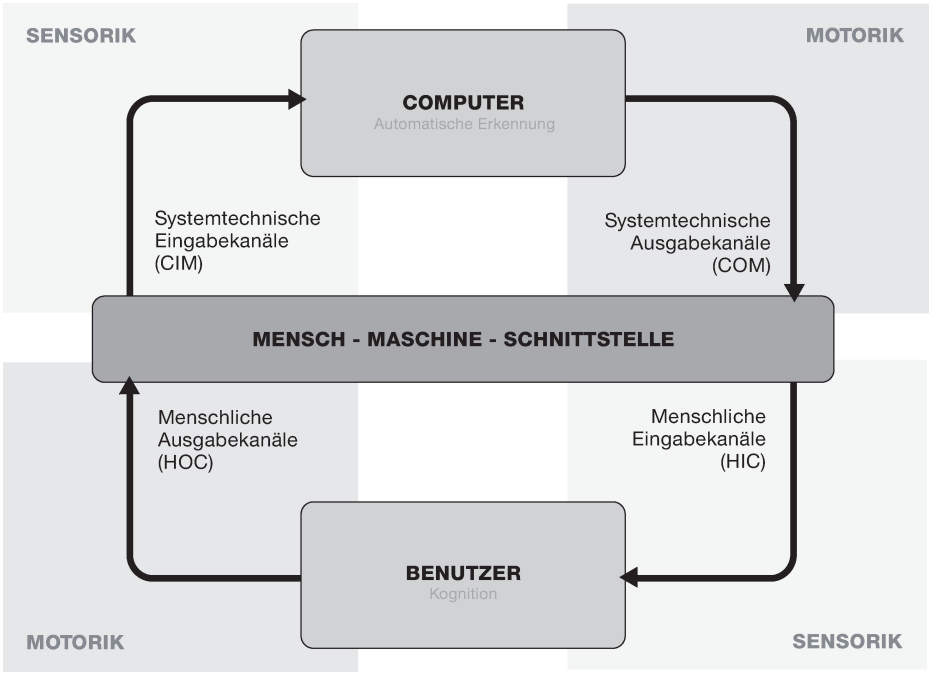


Abb. 2.2: Informationskanäle bei der Mensch-Maschine-Kommunikation (angelehnt an [141])

aufgenommen werden (HIC: Human input channels). Dieser Ansatz verfolgt eine systemorientierte Sichtweise, d.h. man spricht von Eingaben bezogen auf die linke Seite der Grafik mit dem Informationsfluss vom Benutzer zum technischen System und entsprechend von Ausgaben mit Bezug auf den Informationsfluss vom Computer zum Benutzer.

Die Begriffe *Modalität* und *Medium* spielen in diesem Zusammenhang eine wichtige Rolle. Daher soll kurz auf die spezifischen Unterschiede eingegangen werden. Die Nomenklatur orientiert sich an der von Maybury [92] propagierten Terminologie. Mit der Bezeichnung Modalität wird jeweils ein Informationskanal assoziiert, der sich direkt auf den entsprechenden menschlichen Sinn bezieht. Dabei sind in gleichem Maße Eingabe- und Ausgabekanäle des technischen Systems mit einbezogen. Als Medium wird demgegenüber der physikalische Informationsträger bezeichnet, der zur Übertragung, Speicherung oder Darstellung der Information genutzt wird.

Über die Nutzung der Modalitäten lässt sich die sensor-motorische Schnittstelle zwischen den kognitiven Verarbeitungskapazitäten des Menschen und den technischen Möglichkeiten moderner Informations- und Kommunikationssysteme formal spezifizieren. Je nach aktuellem Kontext kann sich eine Modalität M dabei als Oberbegriff sowohl auf eine Klasse zusammengehöriger Informationskanäle ($M \in \{M_{visuell}, M_{auditiv}, M_{taktil}\}$) als auch auf einen spezifischen Kanal innerhalb einer dieser Klassen ($M \in \{M_{taktil,1}, M_{taktil,2}, M_{taktil,3}, ...\}$) beziehen.

2.2.2 Formale Modellierung

Im Kontext der Mensch-Maschine-Kommunikation werden unterschiedliche Modelle zur formalen Spezifikation der Beziehungen zwischen den drei zentralen Entitäten Benutzer, Maschine und der zu erfüllenden Aufgabe verwendet. Auf einer abstrakten Ebene ermöglichen diese Modelle eine vollständige und präzise Beschreibung der Systeme und Systemschnittstellen. Sie stellen ein essentielles Hilfsmittel für eine wissenschaftlich fundierte Diskussion dar. Potenzi-

elle Problemstellungen im Design können auf diese Weise bereits zu einem frühen Zeitpunkt im Entwicklungsprozess adressiert werden. Speziell im Hinblick auf Automatisierung und Qualitätssicherung kommt der formalen Darstellung eine besondere Bedeutung zu.

Formale Modelle der Mensch-Computer-Interaktion sind die Basis für die Entwicklung von Werkzeugen und ermöglichen den strukturellen Vergleich von verschiedenen Ansätzen im Hinblick auf gemeinsame Problemstellungen. Insgesamt kann zwischen vier Modellkategorien unterschieden werden [137]: Aufgabenmodelle, Benutzermodelle, Dialogmodelle und Architekturmodelle. Mit Bezug auf Abbildung 2.1 setzen die einzelnen Kategorien jeweils andere Schwerpunkte im Spektrum der Mensch-Maschine-Kommunikation. Auf die spezifischen Ausprägungen wird im Folgenden detaillierter eingegangen. Zu jeder Modellkategorie werden ausgewählte Arbeiten kurz angesprochen. Für eine detaillierte Diskussion der Verfahren, sowie eine ausführlichere Darstellung weiterer Varianten sei auf die entsprechende Fachliteratur verwiesen.

Aufgabenmodelle

Im Fokus der Aufgabenmodelle steht die zu erfüllende Aufgabe. Die beiden anderen Komponenten aus Abbildung 2.1 Mensch und Maschine werden eher indirekt betrachtet. Aus der Beschreibung der Aufgabe werden sowohl funktionale als auch nicht-funktionale Anforderungen an das Design des Systems und speziell der Mensch-Maschine-Schnittstelle abgeleitet. Die Qualität der entstandenen Lösung wird nach der vollständigen Realisierung des Systems ebenfalls primär aus Sicht der Aufgabe bewertet.

Angelehnt an die Struktur des technisch motivierten OSI-Schichtenmodells hat Rasmussen [130] ein aufgabenbezogenes Handlungsmodell entwickelt, das zwischen Tätigkeiten auf drei verschiedenen Ebenen mit jeweils ansteigender kognitiver Komplexität unterscheidet. Die *fertigkeitsbasierte* Ebene ist durch Interaktionen geprägt, die aufgrund mehrfacher Anwendungen bereits zur Routine geworden sind. Auf der *regelbasierten* Ebene laufen Handlungen in einem deterministischen Prozess nach vorher erlernten Mustern ab. Demgegenüber sind Interaktionen auf der *wissensbasierten* Ebene dadurch charakterisiert, dass der Benutzer mit neuen Situationen und Problemen konfrontiert wird. Die Lösung der Aufgabe erfordert individuelle Entscheidungen und evtl. das Aufstellen eines Plans mit dedizierten Zwischenzielen.

Eine weitere Methode zur Modellierung der Interaktionen ist das von Card, Moran und Newell entwickelte GOMS-Modell [30]. Das Akronym GOMS steht für eine Menge von Zielen (goals), eine Menge von Operationen (operators), eine Menge von Methoden zum Erreichen der Ziele (methods) und eine Menge von Auswahlregeln (selection rules). Es wird davon ausgegangen, dass der Benutzer über eine hierarchische Struktur von Zielen und Teilzielen verfügt, die Aufgaben in atomare Teile zerlegt und mit den zur Verfügung stehenden Interaktionsmitteln selbstständig eine Strategie zur Lösung der Aufgabe erarbeitet. Auf diese Weise erhält man eine grobe Abschätzung des Lernaufwandes für den Erwerb von prozeduralem Benutzerwissen.

Während sich das GOMS-Modell vor allem für die Analyse der Mensch-Maschine Schnittstelle unter idealisierten Randbedingungen eignet, geht das ebenfalls von Card, Moran und Newell entwickelte Keystroke-Level Modell (KLM) einen Schritt weiter [31]. Unter Einbeziehung des aktuellen Bedienkontextes wird versucht, die Zeit zu schätzen, die ein Benutzer für die fehlerfreie Durchführung einer vorgegebenen Aufgabe benötigt. Auf diese Weise erlaubt das KLM Vorhersagen für die Ausführungsperformanz in realistischen Alltagssituationen und gestattet damit, auf Basis der ermittelten Zeiten unterschiedliche Systemvarianten quantitativ zu vergleichen. Als Einschränkung ist das KLM nur für geübte Nutzer optimal anwendbar.

Benutzermodelle

Den Ausgangspunkt für die Benutzermodelle bildet der Mensch mit seinen individuellen Fähigkeiten. Eine wichtige Rolle spielt das mentale Modell, das der Benutzer von der zu erfüllenden Aufgabe und von den Eigenschaften der Maschine hat. Im Hinblick auf detaillierte Kenntnisse über den Aufbau und die Funktionsweise der Maschine können Benutzer grob in zwei Klassen eingeteilt werden: Systementwickler und Endnutzer. Eine gute Mensch-Maschine-Schnittstelle zeichnet sich dadurch aus, dass die Differenz zwischen den mentalen Modellen dieser beiden Benutzerklassen möglichst gering ist [116]. Nur unter dieser Voraussetzung können systemtechnische Lösungen erarbeitet werden, die einfach zu erlernen sind.

Im Gegensatz zu den Aufgabenmodellen setzen die Benutzermodelle parallel an zwei Punkten aus dem Dreieck in Abbildung 2.1 an. Benutzer und Aufgabe werden in einem einheitlichen Kontext betrachtet. Eine Möglichkeit zur abstrakten Darstellung bietet die *Task Action Grammar* (TAG) [124]. Angelehnt an die Theorie der formalen Sprachen basiert die Beschreibung auf grammatikalischen Produktionsregeln, wobei atomare Handlungen, detaillierte Informationen über die Aufgabe und existierenden Regelschemata zueinander in Beziehung gesetzt werden. Die Intention des Benutzers wird vor dem Hintergrund einer spezifischen Aufgabe auf eine Menge von Aktionen abgebildet. Auf diese Weise erhält man eine formale Notation der Problemdomäne, die als systemtechnische Grundlage für die maschinelle Verarbeitung der Interaktionen zwischen Mensch und Maschine verwendet werden kann.

Eine über den TAG-Formalismus hinausgehende Beschreibungsmöglichkeit bietet die *User Action Notation* (UAN) [59]. In dieser Darstellung können die Fähigkeiten des Benutzers und die spezifischen Charakteristika der Aufgabe unabhängig voneinander modelliert werden. Die Notation eignet sich besonders für die Spezifikation von direktmanipulativen Benutzungsoberflächen. Durch die Verwendung von graphischen Symbolen für typische Benutzeraktionen, Feedbackverhalten des Systems und elementare Systemzustände ist die UAN auch für Nicht-Experten verständlich.

Architekturmodelle

Bei den Architekturmodellen steht die Realisierung eines funktionierenden Gesamtsystems im Mittelpunkt. Daher werden anders als bei den Aufgaben- und Benutzermodellen vor allem technische Aspekte im Rahmen der Interaktion zwischen Mensch und Maschine behandelt. Architekturmodelle bilden eine wichtige Basis für die einzelnen Phasen, die in einem Systementwicklungsprozess durchlaufen werden. Aus der Struktur des Systems können Auswirkungen auf die Aufgabenbewältigung und die Benutzerakzeptanz abgeleitet werden.

Im Kern gehen die meisten Modelle zur formalen Beschreibung interaktiver Systeme auf das *Seeheim Modell* [55] zurück. Grundlegendes Merkmal dieses Modells ist die strikte Trennung zwischen der Benutzerschnittstelle und der eigentlichen Applikation. Konzeptionell ist ein System dazu in drei logisch gekapselten Komponenten organisiert. Die *Präsentationseinheit* ist zuständig für systemnahe Ein- und Ausgabeoperationen, die *Dialogkontrolle* steuert den Austausch von Informationen mit unterschiedlichem Abstraktionsgrad und die *Anwendungsschicht* stellt eine dedizierte Schnittstelle zu den eigentlichen Funktionalitäten der Maschine zur Verfügung. Das Seeheim-Modell ermöglicht eine höhere Wiederverwendbarkeit einzelner Module, eine Aufgabenteilung zwischen Anwendungs- und Schnittstellen-Entwicklern und eine

einfachere Anpassung an existierende Standards. An mehreren Stellen ist das Modell relativ abstrakt, so wird beispielsweise weder die Art der Kommunikation noch die Struktur des Kontrollflusses spezifiziert.

Eine konsequente Weiterentwicklung des Seeheim-Modells ist das *User Interface Management System* (UIMS) [47, 65], bei dem speziell der Charakter als praktisches Werkzeug im Software-Engineering Prozess betont wird. Existierende graphische Oberflächenelemente und Interaktionstechniken werden in einer organisierten Entwicklungsumgebung zusammengestellt. Auf Basis einer formalen Spezifikation der Aufgabe und einer Definition der Oberfläche kann das Grundgerüst des zu realisierenden Systems aus den vorhandenen Bausteinen generiert werden.

Das IFIP-Modell [42] ist ungefähr zur gleichen Zeit entstanden. Die Abkürzung IFIP steht für International Federation for Information Processing. Im Unterschied zum Seeheim-Modell, das primär von Softwarespezialisten konzipiert wurde, ist das IFIP-Modell maßgeblich durch die Einflüsse von Psychologen und Ergonomen geprägt. In der Spezifikation der Interaktionen zwischen Mensch und Maschine werden nicht nur systeminterne Komponenten behandelt, sondern auch externe Faktoren aus der Arbeitsumgebung mit einbezogen. Das Modell verfügt dazu über eine zusätzliche Organisationsschnittstelle, die sowohl auf technische als auch auf soziale Randbedingungen eingeht.

Dialogmodelle

Im Gegensatz zu den Aufgaben-, Benutzer- und Architekturmodellen, die jeweils an einem oder maximal zwei Punkten aus dem MMK-Dreieck in Abbildung 2.1 ansetzen, stehen bei den Dialogmodellen vor allem die wechselseitigen Beziehungen zwischen den einzelnen Entitäten im Vordergrund. Auf diese Weise wird versucht, die anderen Modelle in einem größeren Kontext zu vereinen. Anforderungen an das Systemdesign werden aus der parallelen Betrachtung der Komponenten Mensch, Maschine und Aufgabe abgeleitet.

Die Basis für viele Dialogmodelle bilden Sprachmodelle, die ursprünglich aus der Computerlinguistik stammen. Moran hat eine *Command Language Grammar* (CLG) entwickelt [106], mit deren Hilfe eine Benutzungsschnittstelle durch vier Abstraktionsebenen hinreichend genau beschrieben werden kann. Auf der untersten, der *lexikalischen* Ebene erfolgt die Festlegung der elementaren Eingabeformen und der Menühierarchien. Die *syntaktische* Ebene beschreibt das Layout, die möglichen logischen Zustände und die zeitlichen Abläufe innerhalb der konkreten Schnittstelle. Auf der *semantischen* Ebene erfolgt eine Zuordnung von Eingaben und entsprechenden Systemreaktionen. Die *konzeptuelle* Ebene schließlich beinhaltet die wesentlichen Aspekte der Anwendung. Vergleicht man das Sprachmodell mit dem Seeheim-Modell, so entspricht die lexikalische Ebene der Präsentationseinheit, die syntaktische Ebene der Dialogsteuerung und die semantische Ebene der Anwendungsschnittstelle.

Ein Beispiel für eine dialog-orientierte Spezifikation von interaktiven Benutzungsschnittstellen ist die ISO Norm 9241 [71], in der allgemeine Anforderungen und Prinzipien zum Design von Mensch-Maschine-Systemen festgehalten sind. Zur formalen Beschreibung von Dialogmodellen sind verschiedene Notationen entwickelt worden [47]. Das Spektrum reicht von einfachen Formalismen wie Multi-Party-Grammatiken und nativen Zustandsübergangsdiagrammen (STD=State Transition Diagram) über baumartige Modellierungen von Menühierarchien und erweiterte, graphenbasierte Darstellungsformen wie Recursive Transition Networks (RTN) und Augmented Transition Networks (ATN) bis hin zu komplexen Modellierungen durch Petri-Netze und der Aufstellung umfangreicher Ereignismodelle.

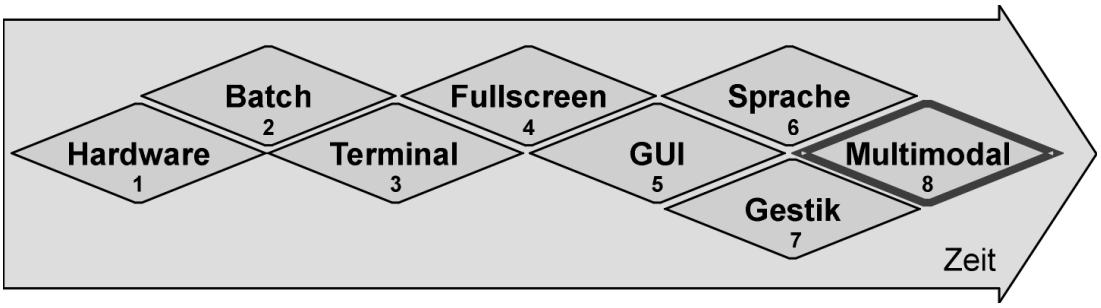


Abb. 2.3: Schematische Darstellung der Evolution von Mensch-Maschine-Schnittstellen

2.2.3 Historische Entwicklung

Um den wachsenden Anforderungen der verschiedenen Benutzergruppen hinsichtlich einer einfachen, effektiven und vor allem intuitiven Bedienbarkeit besser gerecht zu werden, haben sich im Laufe der Zeit verschiedene Arten von Mensch-Maschine-Schnittstellen herausgebildet. Abbildung 2.3 zeigt schematisch die groben Evolutionsschritte dieser Entwicklung im Hinblick auf desktop-orientierte Arbeitsumgebungen. Charakteristisch für die einzelnen Phasen ist, dass sowohl die Dimension der verwendeten Interaktionskanäle als auch der Abstraktionsgrad der zugrundeliegenden Bedienparadigmen deutlich gestiegen sind. Im Folgenden werden die einzelnen Entwicklungsstufen kurz angesprochen. Detaillierte Hintergrund-Informationen zu der historischen Entwicklung von Mensch-Maschine-Schnittstellen finden sich u.a. in [113].

Im Zeitalter der Röhrentechnologie bestanden die ersten Systeme vollständig aus mechanischen und elektro-mechanischen Bauteilen (*Hardware*). Eine Änderung im Ablauf war oftmals gleichbedeutend mit einer äußerst aufwendigen Neuverdrahtung der Maschine. Die Benutzer mussten daher über tiefgreifende Kenntnisse der zugrundeliegenden Systemarchitektur verfügen. Erste Frühformen einer softwarebasierten Interaktion zwischen dem Benutzer und dem System brachten sogenannte *Batch*-Systeme, bei denen einzelne Berechnungsschritte im Vorfeld auf Lochkarten festgelegt und anschließend dem Rechner zur Abarbeitung übergeben werden konnten. Zur Programmierung war immer noch viel Erfahrung im Umgang mit dem simplen, maschinennahen Beschreibungsformalismus nötig, in dem die Systembefehle kodiert wurden. Ein gravierendes Manko dieser Generation von Systemschnittstellen war die strikte zeitliche Trennung zwischen Eingabe, Verarbeitung und Ausgabe.

Speziell der Nachteil einer fehlenden direkten Rückkopplung wurde durch die Einführung von zeilenorientierten alphanumerischen *Terminals* egalisiert, bei denen der Benutzer über einen streng reglementierten Textdialog Informationen mit dem System austauschen konnte. Die Bedienung war schon weitestgehend von der eigentlichen Funktionsweise des Rechners abstrahiert. Der Benutzer des Systems musste lediglich die Kommandosprache beherrschen. Detailliertes Wissen über die Hardware war jedoch nicht mehr erforderlich. Eine deutliche Steigerung in der Interaktionsfreiheit brachten formularbasierte Dialogformen, bei denen der komplette Bildschirm zur Eingabe verwendet werden konnte (*Fullscreen*). Durch die Möglichkeit des freien Wechsels zwischen den einzelnen Eingabemasken konnte die Abarbeitung in gewissen Grenzen individuell geplant und an die speziellen Bedürfnisse und Vorlieben der Benutzer angepasst werden. Mit dieser Generation wurde die Kontrolle über den Ablauf des Dialogs zwischen Mensch und Maschine wesentlich stärker in Richtung des Benutzers verschoben.

Standard bei der Mehrzahl aktueller Systeme sind vollständig graphisch-basierte Benutzungsoberflächen (GUI=Graphical User Interface), bei denen frei skalierbare, zweidimensionale Fenster in mehreren Schichten hintereinander angeordnet sind. Im Zentrum steht das WIMP-Paradigma (WIMP=Window, Icon, Menu, Pointing) und das Konzept der direkten Manipulation [150]. Auf einem fiktiven Arbeitsbereich können verschiedene zweidimensionale Icons, die als Symbole für Anwendungen und Daten verwendet werden, mittels Maus und Tastatur frei positioniert und manipuliert werden. Ein dedizierter Eingabefokus legt dabei fest, an welches Fenster bzw. an welche Anwendung die aktuellen Benutzereingaben weitergeleitet werden. Die extensive Nutzung selbsterklärender Metaphern wie beispielsweise die Verwendung eines Papierkorbsymbols für das Löschen von Dateien ermöglicht einen flexiblen und intuitiven Zugang zu komplexen Systemfunktionalitäten. Gleichzeitig kann aber auch auf klassische, menü-basierte Interaktionsformen zurückgegriffen werden.

Der Hauptanteil der momentan verfügbaren Mensch-Maschine-Schnittstellen in desktoporientierten Arbeitsumgebungen basiert maßgeblich auf rein taktilen Interaktionsformen mittels Maus und Tastatur. Durch die Einführung weiterer Interaktionsparadigmen, die wesentlich stärker an die natürlichen Kommunikationsgewohnheiten des Menschen angepasst sind, kann sowohl die Flexibilität als auch die Robustheit des Dialogs mit dem technischen System erheblich verbessert werden [113, 116].

Die Verwendung von *Sprache* als Interaktionsmittel im Mensch-Maschine-Dialog ermöglicht eine einfache Beschreibung abstrakter Zusammenhänge. Sie stellt damit speziell für Benutzer, die im Umgang mit haptischen Bedienelementen relativ unerfahren sind oder aufgrund spezieller körperlicher Behinderungen die Geräte nicht oder nur eingeschränkt nutzen können, eine wertvolle Eingabealternative dar. In verschiedenen Domänen, speziell wenn es um die Beschreibung räumlicher Zusammenhänge geht, kann eine Eingabe oft einfacher und präziser mit verschiedenen Hand- oder Kopfgesten erfolgen. Darüber hinaus eignet sich eine gesten-basierte Eingabe vor allem auch in lauten Umgebungen, in denen die Spracheingabe mit verstärkten Erkennungsfehlern zu kämpfen hat [118].

Eine Kombination der verschiedenen Eingabe- und Ausgabemöglichkeiten vor dem Hintergrund komplexer multimedialer Anwendungen repräsentiert zur Zeit die höchste Form bei der Entwicklung der Bedienschnittstellen. Das zugrundeliegende Interaktionsparadigma bezeichnet man als *multimodale Interaktion* mit dem System. Die Konzeption, Entwicklung und Bewertung multimodaler Systeme und Systemschnittstellen ist Gegenstand zahlreicher Forschungsprojekte auf dem Gebiet der Mensch-Maschine-Kommunikation und auch zentrales Thema in der vorliegenden Arbeit.

2.3 Multimodale Interaktion

Der permanente Funktionszuwachs in computer-basierten Informationssystemen hat zur Folge, dass Benutzer mit stetig länger werdenden Einarbeitungsphasen konfrontiert werden. Zusätzlich müssen sie sich durch mangelnde Standardisierung immer wieder von neuem an anwendungsspezifische Interaktionsmetaphern anpassen. Die Bedienbarkeit ist, verglichen mit der Einfachheit und der Robustheit der zwischenmenschlichen Kommunikation, aus Benutzersicht bei vielen aktuellen Systemlösungen nur unzureichend gewährleistet. Häufig wird kritisiert, dass der Umgang mit der Maschine umständlich und kompliziert ist und dass die Reaktionen des Systems

auf verschiedene Eingaben des Benutzers oftmals nicht nachvollzogen werden können. Bei vielen Anwendern führt dies zu Frustrationen und langfristig zu einer ablehnenden Haltung gegenüber den technischen Systemen.

Im Vergleich zu den konventionellen, in Abschnitt 2.2.3 diskutierten unimodalen Systemschnittstellen bieten multimodale Ansätze eine Reihe von Vorteilen, mit denen sowohl die Performanz als auch die Akzeptanz moderner Informations- und Kommunikationssysteme deutlich gesteigert werden kann. Als ein wichtiger Punkt wird vielfach die Annäherung an die menschlichen Kommunikationsgewohnheiten hervorgehoben. Bedienkomfort und Bediensicherheit können durch die synergistische Nutzung von visuellen, akustischen und taktilen Kommunikationskanälen sowohl auf der Eingabe- als auch auf der Ausgabeseite erhöht werden. Ein weiterer Vorteil besteht darin, dass die einzelnen Modalitäten nach persönlichen Präferenzen individuell genutzt und zudem noch situationsspezifisch gewichtet werden können. Letztendlich führt dies zu einer geringeren Anzahl an Eingabefehlern und zu einer Verkürzung der Interaktionszeiten.

Multimodale Systeme ermöglichen somit für Experten und normale Benutzer in gleichem Maße einen einfachen, fehlerrobusten, effektiven und vor allem intuitiven Mensch-Maschine-Dialog [92, 118, 166]. Auf Basis einer grundlegenden Begriffstaxonomie wird im Folgenden näher auf ausgewählte empirische Ergebnisse im Rahmen der Bedienung multimodaler Systeme und auf strukturelle Lösungsansätze zur Verbindung unterschiedlicher Informationsquellen eingegangen.

2.3.1 Definition und Taxonomie

In direkter Analogie zu der Vielfalt an Begriffsdefinitionen in Abschnitt 2.1 ist auch die Bezeichnung *Multimodalität* nicht eindeutig definiert. Die unterschiedlichen wissenschaftlichen Publikationen zu dieser Thematik fokussieren jeweils andere Teilaspekte, stimmen jedoch im Hinblick auf zwei zentrale Charakteristika weitgehend überein. Ein *multimodales System* ist vor allem dadurch gekennzeichnet, dass der Benutzer zum einen über mehrere Eingabe- und Ausgabemöglichkeiten mit der Maschine kommunizieren kann und dass zum anderen die einzelnen Informationskanäle in sinnvoller Weise zusammenwirken. Unter dem Einfluss verschiedener Forschungsprojekte findet sich in [26] eine allgemeine Definition, die auch für die vorliegende Arbeit als zentrales Leitmotiv verwendet wird:

> Multimodal systems represent and manipulate information from different human communication channels at multiple levels of abstraction. Multimodal systems can automatically extract meaning from multimodal, raw input data, and, conversely they produce perceivable information from symbolic abstract representations.

Eine grundlegende Taxonomie zur Klassifikation multimodaler Systeme ist im Rahmen des MIAMI-Projektes [60] von Nigay und Coutaz erarbeitet worden. Auf Basis einer modalitätenorientierten Betrachtung wird ein multimodales System anhand von drei Dimensionen bewertet: *Abtraktionsgrad*, *Nutzung* und *Fusion*. Durch diese drei Merkmale wird ein Klassifikationsraum aufgespannt (engl. *design space*), der ein mögliches Schema für die Einordnung und den Vergleich unterschiedlicher Systeme bereitstellt. Abbildung 2.4 veranschaulicht die zentralen Zusammenhänge zwischen den einzelnen Achsen und führt die englischen Fachbezeichnungen für die einzelnen Kategorien ein.

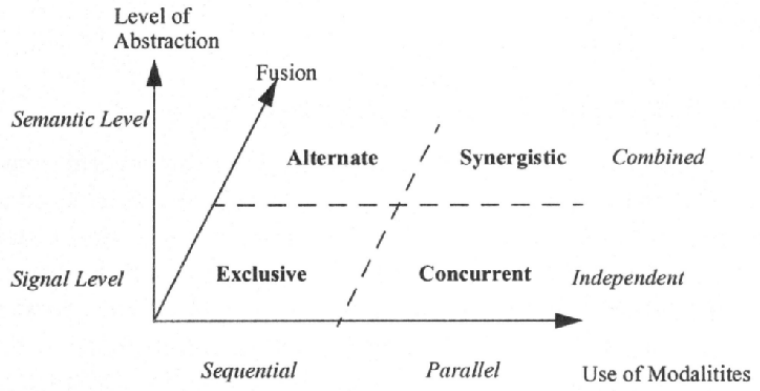


Abb. 2.4: Klassifikationsraum zur Einordnung multimodaler Systeme (nach [114])

Die Dimension Abstraktionsgrad (engl. *level of abstraction*) bezieht sich auf das technische Niveau, auf dem die übertragenen Informationen verarbeitet werden können. Das Spektrum reicht von binären Signalsequenzen bis hin zu komplexen semantischen Ausdrücken. Unter der Dimension Nutzung (engl. *use of modalities*) werden die zeitlichen Relationen im Gebrauch der Modalitäten subsumiert. Diese können entweder parallel oder sequentiell verwendet werden. Mit der Dimension Fusion wird schließlich beschrieben, auf welche Art und Weise die verschiedenen Informationen miteinander verbunden werden. Multimodale Ein- und Ausgaben können sowohl strikt getrennt voneinander als auch gemeinsam interpretiert werden.

Nach Nigay und Coutaz zeichnen sich multimodale Systeme dadurch aus, dass die einzelnen Informationen bereits mit einer konkreten Bedeutung behaftet sind, d.h. die Verarbeitung erfolgt auf einer semantischen Ebene. Aus diesem Grund wird die Dimension Abstraktionsgrad in vielen Diskussionen häufig auch weggelassen. Basierend auf der Klassifikation anhand der beiden anderen Dimensionen Fusion und Nutzung können, wie in Abbildung 2.4 gezeigt, vier prinzipielle Kategorien von multimodalen Interaktionen identifiziert werden: *exklusiv*, *simultan* (engl. concurrent), *alternativ* und *synergistisch*.

Im Rahmen exklusiver Interaktionen werden die Informationen strikt sequentiell verarbeitet, d.h. zu einem dedizierten Zeitpunkt kann der Benutzer immer nur eine Modalität nutzen. Zwischen aufeinanderfolgenden Eingaben bestehen keinerlei funktionale Relationen. Für die Klasse der alternativen Interaktionen erfolgt die Eingabe zwar auch sequentiell, die Eingaben können aber zueinander in Beziehung gesetzt werden. Bei simultanen Eingaben kann parallel mit mehreren Modalitäten interagiert werden, die Interpretation der einzelnen Informationen erfolgt jedoch vollständig getrennt. Synergistische Interaktionen schließen die anderen Kategorien mit ein. Sie stellen die höchste Form multimodaler Interaktionen dar. Die zeitlich parallel erfolgenden Eingaben durch die einzelnen Modalitäten werden in einem funktionalen Gesamtzusammenhang gemeinsam interpretiert.

Eine alternative, stärker funktionsorientierte Taxonomie ist von der Forschungsgruppe um Martin [90] vorgestellt worden. In dieser Darstellung wird grundsätzlich zwischen *Typen* und *Zielen* bei der Kombination von mehreren Modalitäten unterschieden. Die Ziele beschreiben allgemeine Anforderungen an die Mensch-Maschine-Kommunikation wie beispielsweise Intuitivität, Effektivität oder die Fähigkeit des Systems, sich sowohl an verschiedene Benutzer als auch an verschiedene Bediensituationen optimal anpassen zu können. Bei den Typen wird zwi-

schen sechs Kategorien unterschieden: *Redundanz*, *Komplementarität*, *Simultanität*, *Äquivalenz*, *Spezialisierung* und *Transfer*. Im Folgenden wird kurz auf die Bedeutung dieser Kategorien eingegangen, eine detaillierte formale Beschreibung findet sich in [91].

Die ersten drei Kategorien beziehen sich auf die funktionale Zusammensetzung einer isoliert betrachteten Eingabe. Redundanz bedeutet, dass über unterschiedliche Modalitäten dieselbe Information übertragen wird. Jede Eingabe ist für sich vollständig bedeutungstragend. Im Fall der Komplementarität werden Informationen zwar auch über mehrere Kanäle übertragen, eine sinnvolle Interpretation ergibt sich aber erst durch die Zusammensetzung der Teilinformationen aus den einzelnen Modalitäten. Mit Simultanität werden zeitlich parallele Eingaben bezeichnet, die unabhängig voneinander interpretiert werden können. Auf diese Weise kann der Benutzer mit verschiedenen Modalitäten gleichzeitig mehrere Systemfunktionen nutzen.

Die letzten drei Kategorien beziehen sich eher auf generelle Systemeigenschaften. Äquivalenz bedeutet, dass zum Auslösen einer spezifischen Funktionalität unterschiedliche Modalitäten benutzt werden können. Eingaben aus verschiedenen Modalitäten sind untereinander gleichwertig. Im Gegensatz dazu bedeutet Spezialisierung, dass für bestimmte Funktionalitäten exklusiv nur eine spezifische Modalität genutzt werden kann. Es existiert eine direkte Relation zwischen Systemfunktionalität und möglicher Eingabemodalität. Mit Transfer wird die Eigenschaft des Systems bezeichnet, dass sich die Eingabe mit einer Modalität M_1 auf die Eingabe einer anderen Modalität M_2 beziehen kann und erst durch die gemeinsame Interpretation eine gültige Funktionalität ausgelöst werden kann.

In Ergänzung zu diesen sechs unterschiedlichen Kombinationsarten wird in [11] mit *Konkurrenz* bzw. *Rivalität* noch eine weitere Kategorie eingeführt. Dieser Typ wird zur Beschreibung der funktionalen Zusammensetzung von Benutzereingaben verwendet, bei denen die Bedeutungsinhalte der einzelnen Informationen, die über unterschiedliche Modalitäten übertragen werden, in Bezug auf eine spezifische Systemfunktionalität nicht übereinstimmen. Je nach Grad der Übereinstimmung spricht man auf der einen Seite von *partieller* Konkurrenz, wenn die Daten zwar nicht identisch, aber zumindest in einer gewissen Relation zueinander stehen, und auf der anderen Seite von *vollständiger* Konkurrenz, wenn sich die Daten widersprechen.

In Bezug auf die beiden vorgestellten Taxonomien zielt die vorliegende Arbeit auf die Umsetzung eines Systems, mit dem symbol-orientierte Benutzereingaben, die von verschiedenen Erkennermodulen bereits vorverarbeitet worden sind, auf semantischer Ebene synergistisch kombiniert werden können. Der Bedienkomfort soll durch einen möglichst flexiblen Systemzugang verbessert werden. Die Eigenschaften Redundanz, Komplementarität und Äquivalenz spielen in diesem Zusammenhang eine besondere Rolle, wobei die Behandlung rivalisierender Benutzereingaben nicht vernachlässigt werden darf. Bezogen auf die Gesamtheit aller Systemfunktionalitäten wird generell ein möglichst geringer Anteil an Spezialisierung angestrebt.

2.3.2 Empirische Untersuchungen

Die prinzipielle Leistungsfähigkeit multimodaler Systemschnittstellen wird in vielen wissenschaftlichen Publikationen thematisiert. Im Gegensatz dazu existieren in der Literatur aber nur wenige Arbeiten, in denen auf konkrete Ergebnisse aus Benutzerstudien zum multimodalen Interaktionsverhalten eingegangen wird. Zumeist finden sich nur vereinzelte qualitative Bemerkungen im Rahmen der Diskussion ausgewählter Referenzsysteme, quantitative Ergebnisse fehlen oft gänzlich.

Eine detaillierte Untersuchung mit dem Fokus auf der Bestimmung von intermodalen Zeitrelationen ist von der Forschungsgruppe um Oviatt durchgeführt worden [119]. Untersuchungsgegenstand war ein bi-modales Interface zur Planung und Durchführung von verschiedenen Aufgaben im Bereich Immobilienmanagement. Als Eingabemodalitäten standen den Benutzern Sprache und Stiftgestik zur Verfügung. An dem Versuch nahmen 18 Personen teil und es wurden insgesamt 871 Interaktionen ausgewertet. Die Verteilung der Interaktionsmuster wurde klar durch unimodale Sprachäußerungen dominiert (63,5% aller Interaktionen), gefolgt von kombinierten Eingaben mit beiden Modalitäten (19%) und isoliert auftretenden Stiftgesten (17,5%).

Im Gegensatz zu den initialen Erwartungen sind in Bezug auf die multimodalen Interaktionen kombinierte Eingaben aus Sprachinteraktionen und der Anfertigung einer Skizze am häufigsten eingesetzt worden (86%). In 42% der Fälle erfolgten die einzelnen Eingabeanteile mit zeitlicher Überlappung, in 32% mit einer kleinen Pause sequentiell nacheinander und in weiteren 12% der Eingaben wurde zuerst eine Zeichnung erstellt, die dann mit Hilfe von Sprache und einzelnen Markierungen ergänzt wurde. Bei den restlichen 14% der multimodalen Interaktionen wurden sprachliche Äußerungen nur mit einer Zeigegeste kombiniert.

Als eines der wichtigsten Untersuchungsergebnisse konnte festgestellt werden, dass die schriftlichen Eingaben signifikant häufiger vor den sprachlichen Interaktionen auftraten (57% aller multimodalen Eingaben). Nur in 14% der Fälle wurde die Zeichnung nach der sprachlichen Eingabe erstellt. In den restlichen 29% erfolgten beide Eingaben quasi-gleichzeitig innerhalb eines Zeitfensters von 0,1 Sekunden. Diese Ergebnisse flossen als empirisches Wissen unmittelbar in die Realisierung des späteren multimodalen Prototyps ein. Eine detaillierte Analyse der Interaktionszeiten ergab weiterhin, dass alle multimodalen Interaktionen innerhalb einer Zeitspanne von maximal vier Sekunden vollständig abgeschlossen waren.

Unter Verwendung einer ähnlichen Systemarchitektur evaluierte Cohen [35] zeitbezogene multimodale Interaktionsmuster in einem computer-basierten Mensch-Mensch Kommunikationsszenario zur Planung von militärischen Sondereinsätzen. Insgesamt nahmen nur vier Testpersonen an der Evaluierung teil, von denen lediglich die Interaktionen von drei Beteiligten quantitativ ausgewertet werden konnten. Im Gegensatz zu der vorherigen Studie machten hier die multimodalen Interaktionen einen Anteil von mehr als zwei Drittel aller aufgezeichneten Benutzereingaben aus (69%). In dieser Hinsicht divergieren die Ergebnisse der beiden Studien deutlich. Eine Auswertung der zeitlichen Relationen ergab, dass 89% der Eingaben simultan erfolgten und nur 11% mit einer gewissen zeitlichen Verzögerung. In direkter Analogie zu den anderen Ergebnissen wurde bei kombinierten Interaktionen gestenbasierte Interaktionen wesentlich häufiger vor den sprachlichen Äußerungen beobachtet (82%).

Im Zusammenhang mit der Entwicklung von automatischen Spracherkennungssystemen hat Suhm [159, 160] vor dem Hintergrund einer Diktieraufgabe das Potenzial einer multimodalen Fehlerkorrektur mit konventionellen Korrekturmechanismen wie nochmaliges Sprechen des fehlerhaften Wortes, Auswahl aus einer Liste von vergleichbaren Wörtern und Eingabe über die Tastatur verglichen. Die Ergebnisse zeigen, dass für durchschnittliche Computerbenutzer die multimodale Fehlerkorrektur deutlich schneller ist. Lediglich Experten konnten mit der Eingabe über die Tastatur eine höhere Effizienz erzielen. In der multimodalen Umgebung konnte nach einer gewissen Eingewöhnungsphase für alle Benutzer zudem ein Großteil der fehlerhaften Eingaben bereits im Vorfeld vermieden werden. Andere Untersuchungen von Oviatt [122] bestätigen die Beobachtung, dass im Umgang mit multimodalen Mensch-Maschinen-Schnittstellen zum einen weniger Bedienfehler auftreten und zum anderen systemseitige Erkennerfehler einfacher und schneller aufgelöst werden können.

Die Diversität der dargestellten Ergebnisse verdeutlicht eindrucksvoll die Abhängigkeit von der jeweils betrachteten Applikationsdomäne und der Versuchsumgebung. Aufgrund der geringen Anzahl der verfügbaren Eingabemodalitäten müssen die Ergebnisse jedoch sorgfältig interpretiert werden. Für die Konzeption eines generischen Ansatzes zur Integration multimodaler Benutzereingaben, in dem prinzipiell beliebig viele Modalitäten verwendet werden können, sind die Ergebnisse nur begrenzt aussagekräftig. In Bezug auf die Auswertung der eigenen Benutzerstudien in der vorliegenden Arbeit bietet sich aber die Verwendung ähnlicher Klassifikationsschemata an.

2.3.3 Informationsfusion

Im Rahmen der Integration multimodaler Informationen müssen verschiedene architekturale und algorithmische Probleme gelöst werden, die bei der Verwendung konventioneller Schnittstellentechnologien nicht oder nur in geringerem Maße auftreten. Bezogen auf rein temporale Aspekte können die einzelnen Benutzereingaben sowohl parallel als auch mit gewissen Latenzzeiten auftreten. Diese Verzögerungen können durch den Benutzer, aber auch durch unterschiedliche Systemlaufzeiten von vorverarbeitenden Modulen verursacht werden. Im Hinblick auf die semantische Zusammensetzung der Benutzereingaben können die Informationen redundante, komplementäre oder auch rivalisierende Anteile besitzen. Die Ergebnisse der Erkennungsprozesse können zudem noch mit unterschiedlichen Konfidenzwerten behaftet sein.

Benutzereingaben aus unterschiedlichen Quellen können auf verschiedenen Ebenen miteinander verbunden werden. Gourdol [54] unterscheidet drei Stufen der Informationsfusion. Im Rahmen der *lexikalischen Fusion* werden die Daten bereits auf Sensorebene miteinander verbunden. Da die Sensorsignale der einzelnen Modalitäten oftmals stark divergierende Charakteristika aufweisen, kann eine lexikalische Fusion nur bei strukturell sehr ähnlichen Informationsquellen eingesetzt werden, beispielsweise wenn sich beide Quellen auf denselben Sinneskanal beziehen. Bei der *syntaktischen Fusion* werden einzelne Erkennermerkmale miteinander zu einem komplexen Merkmalsvektor zusammengesetzt und als kombinierte Daten klassifiziert. Diese Fusionsart kann besonders dann erfolgreich eingesetzt werden, wenn die einzelnen, modalitätenspezifischen Merkmale in hohem Maße miteinander korreliert sind. Die *semantische Fusion* bezieht sich analog zu dem in Abschnitt 2.3.1 vorgestellten Schema von Nigay und Coutaz auf die Verbindung von Bedeutungsinhalten. Als Teilinformationen werden dazu die Ergebnisse individueller Erkennungsprozesse verwendet.

Die Integrationstechnologie, bei der die Informationsfusion maßgeblich auf den beiden ersten Ebenen abläuft, wird allgemein als *frühe Fusion* (engl. early fusion) bezeichnet [118]. Der Name leitet sich daraus ab, dass die Daten von den verschiedenen Modalitäten bereits zu einem frühen Zeitpunkt gemeinsam in einen zentralen Erkennungsprozess einfließen. Die Integrationsmodule basieren zumeist auf etablierten Ansätzen aus der klassischen Mustererkennung wie Hidden Markov Modellen, künstlichen neuronalen Netzen oder ähnlichen Verfahren. Ein Problem bei dieser Art von Systemen besteht darin, ausreichend Trainingsmaterial zur Verfügung zu stellen. Aufgrund der hohen Dimension der Merkmalsvektoren gestaltet sich dies oftmals als extrem aufwendiger Prozess. Das Training der Klassifikationsmodule impliziert zudem eine Anpassung an die entsprechende Applikationsdomäne.

Im Gegensatz dazu bezeichnet man die Integrationstechnologie, bei der die Informationsfusion maßgeblich auf der semantischen Ebene stattfindet, als *späte Fusion* (engl. late fusion) [118]. Die dazugehörige Integrationskomponente baut auf einzelnen, in sich abgeschlossenen Er-

kennermodulen auf. Eine Kombination der Daten auf einer rein semantischen Ebene ermöglicht eine hohe Flexibilität und Wiederverwendbarkeit der entwickelten Systemlösungen, da die einzelnen Erkennermodule a priori nicht aufeinander abgestimmt werden müssen. Verglichen mit merkmalsbasierten Fusionsansätzen lassen sich semantische Fusionsansätze daher wesentlich einfacher in Bezug auf den Funktionsumfang erweitern und auf andere Applikationsdomänen portieren.

Strukturelle Integrationssansätze

Zur Integration multimodaler Benutzereingaben existieren unabhängig von der betrachteten Fusionsebene prinzipiell verschiedene Ansätze. Bei der *temporalen Integration* werden ausschließlich die zeitlichen Relationen der Eingabedaten betrachtet [115]. Einzelne Informationen werden genau dann als zusammengehörige Komponenten eingestuft, wenn sie sich entweder zeitlich direkt überlappen oder so nah beieinander liegen, dass sie in ein spezifisches Integrationsintervall fallen. Der maßgebliche Nutzen temporaler Integrationsansätze besteht darin, zu bestimmen, ob die Daten aus den einzelnen Modalitäten isoliert betrachtet werden können oder ob sie im Kontext von anderen Informationen zu interpretieren sind. Eine rein zeitbezogene Fusion der Eingabedaten liefert zwar gute Ergebnisse, reicht aber im Allgemeinen nicht aus, um Benutzerintentionen vollständig korrekt zu interpretieren.

Im Rahmen der *regelbasierten Integration* werden die Eingabedaten nach einem deterministischen Regelkodex miteinander verbunden [166, 120]. Die möglichen Kombinationen sind zumeist in einer externen Wissensbasis jeweils mit einer spezifischen Bewertung abgelegt. Alle eingehenden Daten werden mit den Informationen in der Wissensbasis verglichen. Auf diese Weise können zusammengehörige Eingaben identifiziert und gemeinsam interpretiert werden. Regelbasierte Verfahren werden in vielen Systemen sehr erfolgreich eingesetzt, da sie gute Integrationsergebnisse liefern. Eine Problematik regelbasierter Ansätze besteht darin, dass mit der Anzahl der gespeicherten Regeln im ungünstigsten Fall die Komplexität der Regelkorpora exponentiell zunimmt [154].

Ein weiterer Fusionstyp ist die *statistische Integration*. Auf Basis der einzelnen Benutzereingaben produziert jedes Erkennermodul eine Liste von Ergebnissen mit entsprechenden Wahrscheinlichkeiten. Eine quantitative Bewertung für zusammengehörige Eingaben lässt sich dann beispielsweise aus dem Kreuzprodukt der individuellen Erkennerergebnisse berechnen [172]. Eine andere Möglichkeit besteht darin, die Menge der einzelnen Ergebnisse als vereinigtes Muster zu betrachten und in einem nachgeschalteten Klassifikationsmodul zu untersuchen. Um die Leistung des Fusionsprozesses zu steigern, können zusätzlich noch empirische Daten wie generelle Auftrittswahrscheinlichkeiten der einzelnen Eingabekombinationen in den Bewertungsprozess mit einbezogen werden. Die Integration von Zusatzwissen führt aber auch zu einer erhöhten Abhängigkeit von der spezifischen Applikationsdomäne.

Um von den Vorteilen der verschiedenen Fusionstechnologien optimal zu profitieren, sind verschiedene hybride Verfahren entwickelt worden, mit denen die einzelnen Methoden vor dem Hintergrund einer spezifischen Applikation individuell kombiniert werden können. In [120] wird beispielsweise ein hierarchischer, team-orientierter Ansatz diskutiert. Über mehrere Stufen hinweg werden die Ergebnisse der einzelnen Erkennermodule mit Metadaten aus vorherigen Ebenen verbunden. Auf einer Ebene existieren dabei immer mehrere Integrationsalternativen, in denen die einzelnen Ergebnisse individuell spezifisch gewichtet werden. Letztendlich setzt sich die am besten bewertete Kombination der Eingabedaten durch.

2.4 Multimodale Demonstrationssysteme

Auf Basis der interdisziplinären Arbeiten zum Thema Mensch-Maschine-Kommunikation ist eine Vielzahl von Systemen entstanden, die das Leistungspotenzial multimodaler Interaktionsstrategien interaktiv erlebbar machen. Vor dem Hintergrund zum Teil stark divergierender Applikationsszenarien werden dabei unterschiedliche wissenschaftliche und praxisbezogene Fragestellungen adressiert und potenzielle Lösungsansätze vorgestellt. Im Folgenden wird kurz auf einige ausgewählte Systeme aus dem Bereich der semantischen Fusion eingegangen. Eine detaillierte Auflistung und Diskussion weiterer Arbeiten findet sich beispielsweise in [26, 76, 129].

Das erste wissenschaftliche dokumentierte, multimodale Interaktionssystem trägt den Namen *Put That There* und ist von einer Gruppe um Bolt [29] am MIT MediaLab entwickelt worden. Der Benutzer sitzt in einem Sessel vor einer großen Projektionsfläche, auf der unterschiedliche geometrische Objekte dargestellt werden. Die Größe, Position und Orientierung dieser Objekte kann durch sprachliche Äußerungen und optional daran gekoppelte, referenzierende Zeigegesten manipuliert werden. Mit dem System sollte untersucht werden, in welcher Weise sich die Eingaben über die Kommunikationskanäle Sprache und Gestik ergänzen.

Geplant als universal verwendbare Schnittstelle zwischen einem Benutzer und einem Expertensystem hat Wahlster [163] mit dem *XTRA*[1]-System ein multimodales Interface zur Unterstützung bei der Bearbeitung eines Steuererklärungsformulars entwickelt. Über eine Tastatur können natürlichsprachliche Anweisungen eingegeben werden. Der örtliche Bezug dieser Eingabe wird zeitlich sequentiell dazu durch eine einfache Zeigegestik selektiert. Die Fusion der Benutzereingaben wird auf abstrakter Ebene über einen Unifikationsmechanismus realisiert. In dem Schema von Nigay und Coutaz ist XTRA ein Beispiel für die Kategorie der alternativ operierenden Fusionssysteme.

In dem von Neal [110] entwickelten System *CUBRICON* kann der Benutzer vor dem Hintergrund eines militärischen Planungsszenarios von Luftwaffeneinsätzen Maus- und Spracheingaben in verschiedenen Ausprägungen kombinieren. Durch eine Auswertung der Benutzereingaben über eine vierstufige Hierarchie von Wissensbasen kann das System auch sinnvoll mit Ambiguitäten aus der Applikationsdomäne umgehen. Die Eingaben müssen ebenfalls noch in sequentieller Form erfolgen. Der Fusionsmechanismus basiert auf einem generalisierten ATN[2] Formalismus, der über einen speziellen Parser ausgewertet wird.

Das MATIS-System von Nigay und Coutaz [115] ist eine innovative Benutzerschnittstelle für einen Flugauskunftsdienst. Als Kommunikationsmedien stehen Tastatur, Maus und Sprache zur Verfügung. Die Eingaben können dabei sowohl in unimodaler als auch in multimodaler Form erfolgen. Alle Modalitäten bieten prinzipiell die gleiche Ausdruckskraft (Äquivalenz-Prinzip). Zusätzlich kann der Benutzer auch mehrere Anfragen parallel bearbeiten. Im Rahmen der Integration werden die strukturellen Informationsanteile in den einzelnen Modalitäten identifiziert und in einem regelbasierten Verfahren zueinander in Beziehung gesetzt.

Koons [78] beschreibt ein System zur Manipulation von dreidimensionalen Objekten in einer virtuellen Welt, das beispielsweise für Konstruktionsaufgaben eingesetzt werden kann. Im Gegensatz zu den anderen Systemen kommuniziert der Benutzer mit dem Computer ausschließlich über natürliche Interaktionsformen wie Sprache und Handgestik. Zusätzlich können auch die Augenbewegungen ausgewertet werden. Der taktile Kanal wird vollkommen vernachlässigt. Das

[1] XTRA ist eine Abkürzung für eXpert TRAnslator

[2] ATN steht für augmented transition network

Repertoire an Gesten ist extrem umfangreich und erlaubt neben der einfachen Referenzierung von Objekten auch die Verwendung selbstdefinierter Symbole und in Ansätzen sogar die interaktive Gestaltung der Objekte. Die Benutzerintentionen setzen sich meistens aus Eingaben über mehrere Modalitäten zusammen (Transfer-Prinzip). Auf Basis einer abstrakten Darstellung werden diese Einzelinformationen dann regelbasiert fusioniert.

Das mit Abstand meist diskutierte multimodale System ist unter der Bezeichnung *Quickset* von Oviatt und Cohen [36, 37, 120] am Oregon Graduate Institute for Science and Technology entwickelt worden. Die dazugehörige Systemplattform stellt eine Interaktionsumgebung für eine Vielzahl von verteilten Anwendungen zur Verfügung. Benutzerbefehle können zeitlich parallel über Sprache, Stift- und Handgestik eingegeben werden. Die zentrale Integrationskomponente basiert auf einem mehrdimensionalen Unifikationsalgorithmus, der sowohl im Hinblick auf neue Eingabegeräte als auch auf neue Applikationsdomänen erweitert werden kann.

Multimodale Mensch-Maschine-Interaktion ist das zentrale Thema groß angelegter, nationaler und internationaler Forschungsprojekte, wie beispielsweise *MIAMI* [60], *EMBASSI* [164] oder *SMARTKOM* [165]. Einen Schwerpunkt in vielen Projekten bilden virtuelle Arbeitsumgebungen. Engelmeier [43] hat ein medizinisches Diagnosesystem entwickelt, mit dem Ärzte durch grafisch aufbereitete dreidimensionale Patientendaten navigieren können, die aus der Computer Tomographie gewonnen werden. Latoschik [87] hat im SGIM-Projekt gesten- und sprachbasierte Interaktionsformen vor großen virtuellen Anzeigeflächen untersucht. Pavlovic [123] benutzt relativ einfache Virtual-Reality Hardware zur Planung von militärischen Sondereinsätzen. Charakteristisch für viele Systeme dieser Art verfolgt Waibel [166] einen klassischen, regelbasierten KI-Ansatz über die Auswertung von Frames auf Basis eines problemspezifischen Zustandsautomaten, um medizinisches Bildmaterial multimodal präsentieren und bearbeiten zu können.

2.5 Arbeiten aus dem Umfeld des Lehrstuhls

Die vorliegende Arbeit entstand im Rahmen von unterschiedlichen Forschungsprojekten am Lehrstuhl für Mensch-Maschine-Kommunikation der Fakultät für Elektrotechnik und Informationstechnik an der Technischen Universität München [85]. An diesem Lehrstuhl hat die Erforschung intuitiver, natürlicher Interaktionsparadigmen zur Erleichterung des Umgangs mit komplexen technischen Systemen eine lange Tradition. Auf einige zentrale Ergebnisse wird im Folgenden kurz eingegangen, da sie im Zusammenhang mit der Konzeption eines Systems zur Verarbeitung multimodaler Benutzereingaben von besonderer Bedeutung sind.

In einer grundlegenden Arbeit haben Müller und Stahl [104, 157] einen einstufigen, stochastischen Ansatz für ein sprecherunabhängiges, sprachverstehendes System entwickelt. Der Bedeutungsinhalt einer sprachlichen Äußerung wird durch eine semantisch-syntaktische Gliederung repräsentiert. Mit Ausnahme der Vorverarbeitung des sprachlichen Signals läuft der Suchprozess nach der semantischen Gliederung erwartungsgesteuert ab. Das System erzeugt mehrere Äußerungs-Hypothesen und vergleicht sie mit der aufgezeichneten sprachlichen Äußerung des Benutzers. Ein essentieller Vorteil des Ansatzes besteht in der Domänen-Unabhängigkeit der entwickelten Algorithmen. Lediglich die stochastischen Wissensbasen müssen für jede Anwendungsdomäne neu trainiert werden.

Winkler [170] beschreibt ein System zur Erkennung handgeschriebener mathematischer Formeln. Mit einer Kombination aus wissensbasierten und stochastischen Verfahren können zusammengehörige Symbolkonstrukte robust segmentiert, erkannt und analysiert werden. In den

einzelnen Verarbeitungsstufen besteht die Möglichkeit, auftretende Variationen in der Güte der Erkennungsergebnisse durch neu erworbenes Wissen aus der Problemdomäne automatisch aufzulösen. Auf Basis der Arbeiten von Müller, Stahl und Winkler hat Hunsiger [68, 69] ein multimodales System zur automatischen Formel-Erfassung und Verarbeitung entwickelt. Die Besonderheit seines Ansatzes liegt in der Integration aller notwendigen Systemkomponenten in einem erwartungsgetriebenen, einstufig-probabilistischen Dekodierungsverfahren, das die Benutzereingaben über Handschrift, Sprache und Stiftgestik in eine einheitliche Darstellung transformiert und als komplexes Muster analysiert. Das System ist ein prototypisches Beispiel für eine domänenspezifische frühe Fusion.

Im Hinblick auf eine gestenbasierte Mensch-Maschine-Interaktion hat Morguet [107] ein videobasiertes Verfahren für die kontinuierliche Segmentierung und Klassifikation dynamischer Handgesten in einer Desktop-Umgebung vorgestellt. Die Verarbeitung der räumlich-zeitlichen Bildsequenzen basiert auf einer stochastischen Modellierung mit angepassten Hidden-Markov-Modellen. Zobl [176] hat die Leistungsfähigkeit und die Praxistauglichkeit des Systems erheblich erweitert und u.a. auch auf die speziellen Randbedingungen im Rahmen einer automobilen Nutzung angepasst. Geiger [50] verfolgt einen alternativen Ansatz zur Interpretation von Hand- und Kopfgesten, der auf der räumlich-zeitlichen Auswertung von Abstandsinformationen durch ein Feld von Infrarot-Sensoren basiert. Mit seinem System lassen sich ebenfalls unter Realzeitbedingungen hervorragende Erkennungsraten erzielen.

Neuss [111] hat in seiner Arbeit die unterschiedlichen Eigenschaften verschiedener Modalitäten untersucht, fundamentale Anforderungen an ein multimodales Interaktionssystem abgeleitet und einen Prototypen zur Bedienung von Komfortfunktionen im Automobil entwickelt. Im Vordergrund steht dabei die konsequente Umsetzung eines benutzerzentrierten Entwicklungsprozesses. Niedermaier [112] hat die Konzeption multimodaler Mensch-Maschine-Schnittstellen im Automobil auf eine breitere, stärker theoretisch ausgerichtete Basis gestellt. Im Zentrum seiner Arbeit steht die formale Repräsentation des Bediendialogs. Auf der Grundlage dieser Modellierung werden verschiedene Strategien identifiziert, mit denen bereits im Spezifikationsprozess die spätere Benutzbarkeit des Systems gewährleistet werden kann. Beide Arbeiten haben effiziente Lösungsansätze für die Fusion seriell-redundanter Benutzereingaben entwickelt und an die speziellen Randbedingungen im Automobil angepasst.

Die vorliegende Arbeit geht einen deutlichen Schritt weiter. Von den anderen Arbeiten aus dem Umfeld des Lehrstuhls für Mensch-Maschine-Kommunikation grenzt sie sich vor allem dadurch ab, dass der Schwerpunkt auf der Entwicklung eines generischen Ansatzes zur Integration multimodaler Benutzereingaben liegt. Eine fundamentale Anforderung besteht darin, a priori weder die Applikationsdomäne noch die Art und die Anzahl der Ein- und Ausgabemodalitäten festzulegen. Darüber hinaus können neben redundanten und komplementären Informationsanteilen auch rivalisierende Benutzereingaben erkannt und verarbeitet werden. Die folgenden Kapitel konzentrieren sich auf die Beschreibung der einzelnen Phasen und strukturellen Elemente dieses Ansatzes.

KAPITEL 3

Konzeption einer Basisuntersuchung

In diesem Kapitel wird das Konzept einer Benutzerstudie zur Evaluierung multimodaler Interaktionsmuster beschrieben. Um Anforderungen für das Design einer generischen Basisarchitektur ableiten zu können, stehen zwei unterschiedliche Anwendungsszenarien im Zentrum der Untersuchung: eine desktop-orientierte Virtual-Reality Applikation (DVA) und automotive Infotainment Applikation (AIA). Nach einer knappen Motivation zum benutzerzentrierten Entwicklungsprozess werden unabhängig von der konkreten Zielapplikation die zentralen Ziele der Studie erläutert, die zur Verfügung stehenden Eingabemodalitäten vorgestellt und der prinzipielle Ablauf der einzelnen Versuchsphasen erklärt. Die beiden folgenden Abschnitte beinhalten eine detaillierte Beschreibung der speziellen Versuchsaufbauten, wobei auf die entwickelten Software-Prototypen, die jeweiligen Testumgebungen mit den lokalen Randbedingungen und die Aufgaben eingegangen wird, mit denen die Probanden im Test konfrontiert werden. Das Kapitel schließt mit der Darstellung einiger Zusatzprogramme, die sowohl in der konkreten Testphase als auch bei der Evaluierung der Ergebnisse von essentieller Bedeutung sind.

3.1 Benutzerzentrierter Entwicklungsprozess

Gerade bei technisch komplexen Anwendungen hat der allgemeine Bedienkomfort einen signifikanten Einfluss auf die Beurteilung des Gesamtsystems. Da die Benutzer häufig nicht zwischen der eigentlichen Systemfunktionalität und der dazugehörigen Benutzungsoberfläche unterscheiden, wird die Akzeptanz eines konkreten Produktes in zunehmenden Maße über die Qualität des Mensch-Maschine-Interfaces (MMIs) bestimmt [116]. Nach der für bildschirmorientierte Arbeitsplätze entwickelten ISO Norm 9241 Teil 11 [71] wird die *Gebrauchstauglichkeit* (englischer Fachbegriff: *Usability*) einer Benutzungsoberfläche definiert als „das Ausmaß, in dem ein Produkt von einem bestimmten Nutzer in einem bestimmten Nutzungskontext gebraucht werden kann, um bestimmte Ziele effektiv, effizient und mit Zufriedenheit zu erreichen“.

Den Prozess zur Bewertung und zur Verbesserung der Usability bezeichnet man als *Usability Engineering*. In die Beurteilung fließen neben quantitativen auch qualitative Größen. Während unter der *Effektivität der Bedienung* die Genauigkeit und Vollständigkeit subsumiert wird, mit der eine vorgegebene Aufgabe durchgeführt werden kann, gibt die *Effizienz der Bedienung* den

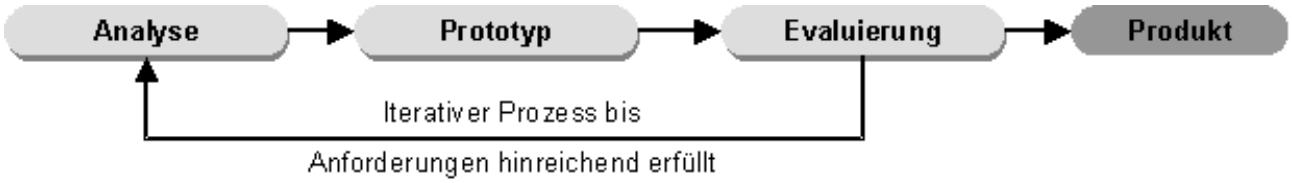


Abb. 3.1: Stufen des benutzerzentrierten Entwicklungsprozesses (nach ISO Norm 13407 [70])

im Verhältnis dazu benötigten zeitlichen, wirtschaftlichen und personenspezifischen Aufwand an [113]. Unter der *Zufriedenheit* wird nach der Definition die subjektive Einstellung gegenüber der Nutzung eines Systems und die Freiheit von etwaigen Beeinträchtigungen verstanden.

Die Methodik zur Evaluation interaktiver Benutzungsschnittstellen werden ebenfalls in der ISO Norm 9241 spezifiziert (Teil 10). Den Kern dieser Norm bildet die strikte Forderung nach allgemeinen Prinzipien zur Gestaltung des Dialogs zwischen Benutzer und System, die auf der menschlichen Wahrnehmung und Kognition beruhen und von Psychologen und Software-Ergonomen gemeinsam entwickelt worden sind.

Insgesamt kann ein hoher Bedienkomfort durch die konsequente Umsetzung eines benutzerorientierten Entwicklungsprozesses erreicht werden. Das Konzept der Benutzerzentrierung bedeutet in diesem Zusammenhang nicht nur, dass umfassende Hintergrundinformationen über die Gruppe der potenziellen Endanwender gesammelt werden, sondern dass die Benutzer als elementare Entscheidungsträger in den zentralen Phasen der Systemkonzeption unmittelbar integriert sind. Abbildung 3.1 gibt einen Überblick über die fundamentalen Schritte in einem benutzerorientierten Entwicklungszyklus. Im Unterschied zu einem klassischen, funktionsorientierten Entwicklungsprozess, beispielsweise nach dem Wasserfallmodell oder dem Spiralmodell [153], werden auf Basis einer detaillierten Anforderungsanalyse die entwickelten Konzepte in einem iterativen Verfahren verschiedenen Benutzertests unterzogen, bei denen die Gebrauchstauglichkeit von prototypisch umgesetzten Demonstratoren im Hinblick auf ausgewählte Funktionalitäten evaluiert wird. Nach der formalen Definition in der ISO Norm 13407 [70] wird der Zyklus aus Analyse, Prototyp-Design und Evaluierung solange durchlaufen, bis die initial formulierten Anforderungen hinreichend genau erfüllt sind. Der Bewertungsprozess mittels Benutzerstudien wird auch als *Usability Testing* bezeichnet.

Im Rahmen der Konzeptionsphase interaktiver Applikationen, bei denen prinzipiell mehrere Ein- und Ausgabemöglichkeiten zur Kommunikation mit dem technischen System angeboten werden sollen, ist die Beurteilung des Bedienkomforts ein essentieller Faktor zur Abschätzung der ökonomischen Randbedingungen, unter denen das spätere Produkt entwickelt werden muss. In Benutzerstudien können sowohl präferierte Interaktionsmöglichgkeiten für unterschiedliche Nutzergruppen und Kontexte ermittelt werden, als auch potenzielle Interferenzen parallel ausgeführter Bedienaufgaben untersucht werden. Ein Beispiel für ein Szenario mit sich überlagernden Bedienaufgaben ist die Steuerung von Infotainment-Applikationen während des Autofahrens, die auch in dieser Benutzerstudie von zentraler Bedeutung ist.

Darüber hinaus stellen Methoden des Usability Engineerings ein effektives Mittel zur Verfügung, um das generelle Potenzial multimodaler Bedienparadigmen evaluieren zu können. Im Vorfeld der Entwicklung können auf diese Weise richtungsweisende Aussagen getroffen werden, ob die Kontrolle einer spezifischen Systemfunktionalität über eine neue Modalität für den durchschnittlichen Benutzer überhaupt eine Erleichterung darstellt.

Die strategische Anwendung eines vollständig benutzerzentrierten Entwicklungsprozesses bei der Konzeption eines einfachen multimodalen Demonstrators in der Automobildomäne ist zentraler Gegenstand der Dissertationsarbeit von Robert Neuss [111]. Mit Bezug auf die dort entwickelten Methoden konzentriert sich in dem speziellen Kontext der vorliegenden Arbeit die Nutzung des benutzerzentrierten Prozesses vor allem darauf, in einer Voruntersuchung prototypisches multimodales Interaktionsverhalten in verschiedenen Applikationsdomänen zu explorieren. Die weiteren Schritte des iterativen Design-, Implementations- und Testdurchlaufs werden nur partiell behandelt. Auf Basis der empirischen Untersuchungen wird ein generisches Konzept entwickelt und in mehreren Demonstratoren prototypisch umgesetzt. Somit wird zumindest einmal der vollständige Zyklus eines benutzerzentrierten Designprozesses durchlaufen. Eine iterative Evaluierung erfolgt jedoch nur in ausgewählten Bereichen. In Abschnitt 8.3.2 werden beispielsweise verschiedene Ergebnisse bezogen auf die Akzeptanz der weiterentwickelten Prototypen vorgestellt.

3.2 Allgemeines Versuchsdesign

3.2.1 Zentrale Fragestellungen

Das primäre Ziel der Benutzerstudien ist die Evaluierung multimodaler Interaktionsmuster bei der Bedienung komplexer Informationssysteme. Vor dem Hintergrund der parallelen Verfügbarkeit mehrerer Eingabekanäle sind die Verteilungen sowohl der Einzelmodalitäten als auch der mögliche Modalitätenkombinationen von Interesse. Ein weiteres Ziel besteht darin, den Anteil der Benutzerintentionen an komplementären, redundanten und auch rivalisierenden Informationsentitäten zu bestimmen. Durch die Analyse der individuellen Modalitätenübergänge sollen Rückschlüsse über prototypische Verhaltenweisen spezifischer Benutzergruppen gewonnen werden. Von zusätzlichem Interesse ist eine detaillierte Untersuchung der einzelnen, modalitätenbezogenen Interaktionszeiten und den intermodalen Zeitrelationen, wobei zwischen der prinzipiellen Struktur und modalitätenspezifischen Überlagerungen unterschieden werden muss.

Aus den erzielten Ergebnissen sollen sowohl fundamentale funktionale als auch nicht-funktionale Randbedingungen für das Design einer generischen mutimodalen Basisarchitektur abgeleitet werden. Zusätzlich sollen die Ergebnisse in Form von empirischem Kontextwissen unmittelbar in den Integrationsprozess einfließen und diesen damit insgesamt robuster machen. Das im Verlauf der Benutzertests anfallende Datenmaterial wird zudem manuell annotiert (siehe Abschnitt 4.2.3) und als Basis für die Evaluierung verschiedener Integrationskonzepte verwendet.

3.2.2 Verfügbare Eingabemodalitäten

In der Basisunteruntersuchung werden a priori keinerlei spezifische Einschränkungen bezüglich der verwendbaren Eingabemöglichkeiten gemacht. Um diese Anforderung hinreichend genau zu erfüllen, muss die Systemschnittstelle ein breites Spektrum an verschiedenen Interaktionsmöglichkeiten zur Verfügung stellen. Neben den klassischen haptischen Bedienschnittstellen, wie beispielsweise Maus oder Tastatur, müssen auch semantisch höherwertige Eingabemöglichkeiten, wie beispielsweise natürliche Sprache und dynamische Hand- und Kopfgestik, unterstützt werden.

Die individuellen Eingabemodalitäten sind dabei in einer Weise konzipiert, dass prinzipiell der volle Funktionsumfang der Zielapplikation ansteuerbar ist, soweit dies technisch realisierbar ist. Im Kontext einer konkreten Applikation bedeutet die Erfüllung dieser Anforderung, dass eine spezifische Funktionalität des Systems sowohl über ein haptisches Bedienteil, als auch per Sprache und Handgestik initiiert werden kann. Die weitestgehende Erfüllung dieser Anforderung ist eine unverzichtbare Voraussetzung für eine effektive und fehlerrobuste synergistische Multimodalität nach dem in Abschnitt 2.3.1 beschriebenen Schema.

Bezogen auf die Klasse der taktilen Eingabegeräte unterstützen die beiden Prototypen im Test einen Touchscreen als universellen Mausersatz. Dieser Bildschirm ist gleichzeitig das primäre Anzeigeelement im multimodalen Systemsetup. Als weiterer Vertreter der taktilen Eingabegeräte stehen domänenspezifische Tastenkonsolen zur Verfügung. In dem DVA Szenario ist dies einfach eine Standard PC Tastatur, während in der Automobilumgebung eine in die Armauflage des Versuchsfahrzeugs integrierte Tastenkonsole Anwendung findet, die unverändert von verschiedenen Vorversuchen am Institut übernommen wurde [111].

Durch die Einführung weiterer Interaktionsparadigmen, vor allem den Gebrauch von Sprache und Gestik und Kombinationen zwischen den einzelnen Modalitätengruppen, stellen die Testinterfaces eine natürlichere und flexiblere Form der Mensch-Maschine-Kommunikation zur Verfügung, die deutlich stärker als die rein taktile Interaktion an die realen Kommunkationsgewohnheiten des Menschen angepasst ist.

Speziell die Sprache erlaubt einen einfachen und direkten Zugang auch zu komplexen Funktionalitäten ohne dabei den Augenkontakt von der eigentlichen Bedienaufgabe abwenden zu müssen. Dies ist beispielsweise bei der Bedienung von MMIs in der Automobilumgebung von großer Bedeutung, da der Blick ständig auf die Fahrsituation gerichtet werden sollte und nicht auf spezielle haptische Bedienkonsolen. Zudem können die Hände bei rein sprach-basierter Eingabe primär zum Steuer des Fahrzeugs eingesetzt werden, um damit auch in unvorhersehbaren Gefahrensituationen besser und schneller reagieren zu können. Auch in einem desktoporientierten Szenario bietet die Sprache als zusätzliche Interaktionsform erhebliche Vorteile. Beispielsweise kann der Benutzer per Maus und Tastatur eine Textverarbeitung bedienen während per Sprache parallel dazu diverse Zusatzfunktionen wie Terminplaner, Audio-Applikationen oder auch Email-Programme kontrollierbar sind.

In verschiedenen Domänen, speziell wenn es um die Beschreibung räumlicher Zusammenhänge geht, können einzelne Informationszusammenhänge oftmals wesentlich einfacher und zudem meist deutlich präziser mittels Hand- und Kopfgesten ausgedrückt werden als den kompletten Sachverhalt per Sprache zu umschreiben [155]. Darüber hinaus stellt eine gesten-basierte Eingabe eine interessante Alternative speziell in lauten Umgebungen dar, in denen aktuelle Spracherkennungssysteme mit verstärkten Fehlklassifikationen zu kämpfen haben [107]. Gestik ist zudem für verschiedene Personengruppen eine wertvolle Eingabealternative, die sich durch partielle Behinderungen nicht deutlich artikulieren oder durch mangelnde Feinmotorik mit taktilen Eingabegeräten nicht effektiv genug umgehen können. Obwohl mittels Kopfgesten im Allgemeinen nur ein stark eingeschränktes Vokabular modelliert werden kann, stellen sie in verschieden Bediensituationen, beispielsweise bei Entscheidungsfragen in einem Dialogablauf, eine wertvolle Ergänzung zu Handgestik und Sprache dar [50].

Um von den individuellen Vorteilen der strukturell unterschiedlichen Eingabekanäle optimal profitieren zu können, hat der Benutzer in dem hier vorgestellten experimentellen Aufbau die Möglichkeit, zwischen fünf verschiedenen Eingabemöglichkeiten zu wählen: konventionelle

Eingabe mittels **T**ouchscreen (T) oder T**a**stenkonsole (A), **s**prachliche Eingabe (S) und gestenbasierte Eingabe sowohl durch dynamische und statische **H**andgesten (H) als auch durch **K**opfgesten (K). Die Menge der verschiedenen Eingabemodalitäten wird im Folgenden mit $\mathcal{M}_{IN} = \{T, A, H, K, S\}$ bezeichnet und ein Element aus dieser Menge allgemein über den Buchstaben m referenziert.

Rein formal lässt sich mit den im Versuch zur Verfügung stehenden Eingabemodalitäten ein bimodaler Interaktionsraum $\mathcal{K}$ von zehn Modalitätenkombinationen aufstellen, wobei der Reihenfolge der beteiligten Einzelmodalitäten in dieser allgemeinen Darstellung keine Bedeutung zukommt, d.h. $K_{m_1 m_2} = K_{m_2 m_1}$ mit $m_1, m_2 \in \mathcal{M}_{IN}$:

$$\mathcal{K} = \{K_{TA}, K_{TH}, K_{TK}, K_{TS}, K_{AH}, K_{AK}, K_{AS}, K_{HK}, K_{HS}, K_{KS}\} \tag{3.1}$$

Im Kontext spezieller domänenbezogener Applikationen ist nicht immer jede Modalitätenkombination voll einsatzfähig. So ist es beispielsweise bereits aus reinen Sicherheitsaspekten wenig sinnvoll, in der Automobilumgebung die parallele Bedienung durch Handgestik und Touchscreen zuzulassen, da zumindest eine Hand immer am Lenkrad bleiben sollte. Zudem hängt die prinzipielle Kombinationsmöglichkeit zweier Eingabemodalitäten von der Mächtigkeit des Interaktionsvokabulars ab, das von den verwendeten Erkennermodulen unterstützt wird.

Die zulässigen Hand- und Kopfgesten orientieren sich im Wesentlichen an den Ergebnissen aus anderen Benutzerstudien und Voruntersuchungen, die am Lehrstuhl für Mensch-Maschine-Kommunikation durchgeführt wurden [27, 96, 132]. Bezogen auf die Sprache als Eingabemedium werden sowohl kommandohafte Äußerungen, als auch spontane, natürlich-sprachliche Äußerungen akzeptiert. Um ein Sprachkommando zu initiieren, wird das sog. *Open-Microphone Paradigma* verwendet, bei dem Benutzer analog zu der zwischenmenschlichen Kommunikation eine Anforderung an das System aus dem freien Verlauf formulieren können, ohne dabei vorher den Interaktionswunsch beispielsweise durch das Drücken einer speziellen Taste (*Push-To-Talk Paradigma*) oder die die Benutzung eines speziellen Schlüsselwortes (*Keyword-Initialising*) signalisieren zu müssen. Ein Vergleich der verschiedenen Initialisierungsstrategien zur Bedienung von Infotainment-Applikationen im Automobil wird in [111] diskutiert.

Versuchsmethodik

Die Funktionalität der beiden Software-Prototypen ist in Teilbereichen nach einem klassischen Wizard-of-Oz Schema [113] realisiert. Während die rein taktilen Benutzerinteraktionen mit dem Touchscreen und der Tastenkonsole unmittelbar in Systemreaktionen umgesetzt werden, wird die Erkennung der semantisch höherwertigen Modalitäten durch einen menschlichen Versuchsleiter simuliert. Dieser sog. *Wizard* beobachtet die Testpersonen von einem abgetrennten Raum aus, interpretiert die sprach- und gestenbasierten Interaktionen und löst die entsprechenden Systemreaktionen manuell aus. Die Handlungen des Versuchsleiters werden in Form von abstrakten Befehlen an das Testinterface zurückgeleitet und bewirken die ursprünglich vom Benutzer intendierte Funktionalität. Für den Benutzer hat es den Anschein, als ob das System real funktionieren würde.

Der Wizard ist in dieser Basisuntersuchung dazu angeleitet, im Falle von mehrdeutigen Eingabeszenarien, die Benutzerinteraktion bestmöglich im Kontext der jeweiligen Applikation und der aktuellen Bediensituation zu interpretieren. Durch diesen hoch-kooperativen Wizard wird garantiert, dass der multimodale Interaktionsfluss nicht durch individuelle Unterschiede in der

Artikulation und Variationen des Gestenvokabulars negativ beeinflusst wird. Die simulierte Erkennung wird einer automatischen Erkennung vorgezogen, um das Versuchergebnis nicht durch indeterministisch auftretende Fehlerkennungen der einzelnen Erkennermodule zu verfälschen. Um zusätzlich auch von personenbezogenen Leistungsschwankungen des Wizards abstrahieren zu können, besteht eine essentielle Voraussetzung für eine möglichst konforme Generierung der Versuchsdaten darin, in beiden Domänen für alle Tests den gleichen Versuchsleiter einzusetzen.

3.2.3 Versuchsablauf

Der Versuch zur Evaluierung multimodaler Interaktionsmuster ist von seinem prinzipiellen Ablauf nahezu identisch für beide Applikationsdomänen. Leichte Variationen gibt es lediglich in dem spezifischen Versuchsaufbau und den Aufgaben, die von den Testpersonen abgearbeitet werden müssen. Strukturell kann jeder Versuch in drei zentrale Phasen unterteilt werden: Einführung (Phase I), modalitätenspezifisches Training (Phase II) und Haupttest (Phase III). Auf die Konzeption der einzelnen Versuchsphasen wird im Folgenden näher eingegangen.

Phase I: Anfangsfragebogen und Einführung

Nach der Begrüßung wird die Testperson zunächst gebeten, einen einordnenden Fragebogen auszufüllen. Dabei werden sowohl demographische Daten (Alter, Geschlecht, etc.) als auch spezifische Kenntnisse in der jeweiligen Applikationsdomäne erfragt. Um den zeitlichen Aufwand für das Ausfüllen und auch für die spätere Auswertung möglichst gering zu halten, wird größtenteils auf die Methode des semantischen Differentials [113] zurückgegriffen. Die Versuchspersonen müssen bezogen auf verschiedene Technologien und Interaktionsmöglichkeiten ihre Erfahrungen ähnlich zu einem Notenschema in einer sechsstufigen Skala zwischen den Prädikaten *sehr viel Erfahrung* (1) und *keine Erfahrung* (6) quantisieren. Der komplette Anfangsfragebogen ist in Abbildung A.1 dargestellt. Zusätzlich müssen die Testpersonen noch ihr schriftliches Einverständnis zu der Aufnahme von Audio- und Videodaten zu geben, wobei ihnen zugesichert wird, dass das Material ausschließlich für Forschungszwecken genutzt wird.

Um die Testpersonen besser einordnen zu können und im weiteren Verlauf der Auswertungen eine gruppenspezifische Diskussion einzelner Interaktionsstile zu ermöglichen, werden die Teilnehmer gemäß ihrer Antworten im Anfangsfragebogen a priori in drei disjunkte Klassen eingeteilt: Anfänger (ANF), Normalnutzer (NOR) und Experten (EXP). In diese Bewertung gehen die individuell unterschiedlichen Erfahrungen beim generellen Umgang mit informationstechnischen Systemen nach einem festgelegten Schema mit verschiedenen Gewichten ein. Die Buchstabenkürzel werden bei der Diskussion der Versuchergebnisse (Kapitel 4) häufig verwendet, um die jeweilige Benutzerklasse zu referenzieren.

In der ersten Phase lernt der Benutzer in einer interaktiven Trainingssitzung, die zusammen mit dem Versuchsleiter direkt in der Testumgebung durchgeführt wird, den Software-Prototypen kennen. Dabei besteht ausreichend Zeit, sich intensiv mit den verschiedenen Funktionalitäten auseinanderzusetzen. Der Versuchsleiter achtet extrem darauf, dass keine Funktionsgruppen ausgelassen werden. Bezogen auf die Eingabemöglichkeiten werden in dieser Phase vor allem die taktilen Interaktionen trainiert. Erst wenn die Versuchsperson eine gewisse Sicherheit im Umgang mit dem Testsystem zeigt, werden auch die anderen Interaktionsformen motiviert. Dazu wird der prinzipielle Gebrauch von Sprache und Gestik ausführlich erklärt.

Abschließend weist der Versuchsleiter noch einmal explizit darauf hin, dass im Test die einzelnen Modalitäten auch kombiniert werden können. Im Gegensatz zu den taktilen Interaktionen können die semantisch höherwertigen Interaktionsformen in der ersten Phase jedoch noch nicht ausprobiert werden, da ein entsprechender Wizard im Kontrollraum fehlt.

Obwohl die Benutzer prinzipiell zur Kommunikation mit dem System ein eigenständiges Sprach- und Gestenvokabular verwenden dürfen, erklärt der Versuchsleiter in dieser Phase mögliche Eingabealternativen zu einer rein taktilen Interaktionsform. Dazu gehört beispielsweise eine dynamische Wischbewegung mit der flachen Hand nach links oder rechts, um eine entsprechende Richtungsinformation zu spezifizieren. Gleichzeitig wird ein entsprechender Sprachbefehl motiviert. Im Gegensatz zu einem klassischen Usability-Versuch wird der Testperson ebenfalls erklärt, dass die Erkennung der sprach- und gestenbasierten Eingaben durch einen menschlichen Wizard unterstützt wird. Die Versuchspersonen werden gebeten, evtl. auftretende Fehlerkennungen und kurzweiligen Verzögerungen im Versuchsablauf zu akzeptieren und nicht primär mit in die Beurteilung des Gesamtsystems einfließen zu lassen. An dieser Stelle weist der Versuchsleiter auch noch einmal deutlich darauf hin, dass das Hauptziel der Versuche nicht in der Perfektion einzelner Erkennermodule besteht sondern in einer detaillierten Untersuchung des multimodalen Interaktionsverhaltens.

Phase II: Modalitätenspezifisches Training

In der zweiten Phase des Versuchs befindet sich die Versuchsperson alleine im Versuchslabor. Der Versuchsleiter sitzt räumlich getrennt davon in einem anderen Raum und überwacht die Interaktionen der Versuchsperson über Audio- und Videokanäle. Der Hauptzweck dieser Phase ist ein modalitätenspezifisches Training der Versuchspersonen vor dem Hintergrund konkreter Bedienaufgaben und alternierender Eingabemöglichkeiten. Insgesamt besteht dieser Versuchsteil aus vier Blöcken, in denen die Testperson jeweils verschiedene Modalitätenkombinationen K_i mit $K_i \in \mathcal{K}$ zur Lösung identischer Bedienaufgaben einsetzen soll. Im ersten Block sind dazu die spezifische Kombinationen Touchscreen und Tastenkonsole erlaubt (K_{TA}), im zweiten Block Touchscreen und Sprache (K_{TS}), im dritten Block Sprache und Handgestik (K_{SH}) und im vierten Block schließlich Sprache und Kopfgestik (K_{SK}).

Nach jedem dieser Versuchsblöcke wird die Testperson gebeten, in einem kurzen Fragebogen die Bedienbarkeit des Testsystems mit dem jeweiligen Interaktionsparadigma zu bewerten. Im Stil der Anfangsbefragung wird auf die Methode des semantischen Differentials zurückgegriffen. Anhand der fünf Adjektivpaare *angenehm* vs. *unangenehm* (SD_1), *einfach* vs. *umständlich* (SD_2), *effektiv* vs. *hinderlich* (SD_3), *entlastemd* vs. *belastend* (SD_4) und *sicher* vs. *unsicher* (SD_5) muss jede Versuchsperson die Qualität des Interfaces auf einer sechsstufigen Skala beurteilen. Zusätzlich besteht die Möglichkeit, in einem separaten Feld allgemeine Anmerkungen zu machen und Probleme bei der Bedienung interaktiv mit dem Wizard zu diskutieren. Eine vollständige Angabe des Zwischenfragebogens findet sich in Abbildung A.2.

Phase III: Haupttest mit freier multimodaler Interaktion

Das eigentliche Testszenario wird in der dritten Phase des Versuchs bereitgestellt. Alle Testpersonen verfügen durch das vorangegangene Training über erste Erfahrungen im Umgang mit den einzelnen Modalitäten. Vor dem Hintergrund einer deutlich komplexeren Bedienaufgabe

können die Benutzer nun sämtliche zur Verfügung stehenden Interaktionsformen zur Bewältigung der Aufgabe einsetzen. Notationstechnisch wird diese Kombinationsmöglichkeit mit dem Symbol K_{ALL} beschrieben. Dabei besteht gegenüber der zweiten Phase keinerlei vorgegebene Bindungen mehr hinsichtlich der Art und Weise der Interaktion. Die Versuchspersonen werden zu Beginn der dritten Phase vom Wizard noch einmal ausdrücklich motiviert, Kombinationen der einzelnen Eingabemodalitäten einzusetzen.

Im Anschluß an die Bedienaufgaben müssen die Versuchspersonen noch einen letzten Fragebogen ausfüllen, um die Bedienbarkeit des Systems im multimodalen Umfeld mit allen fünf zur Verfügung stehenden Eingabemöglichkeiten zu bewerten. Im Gegensatz zur Zwischenbefragung wird zusätzlich noch ausführlich auf potenzielle Schwachstellen und problematische Situationen beim Umgang mit dem System eingegangen. Zur Auswertung werden neben verschiedenen semantischen Differentialen modalitätenspezifische Prioritäten bzgl. der Bedienung einzelner Systemfunktionalitäten quantitativ erfasst. Die beiden letzten Teile des Fragebogens sind in den Abbildungen A.3 und A.4 dargestellt. Als Dankeschön für die Teilnahme an der Benutzerstudie erhalten die Testpersonen abschließend noch eine kleine Gratifikation.

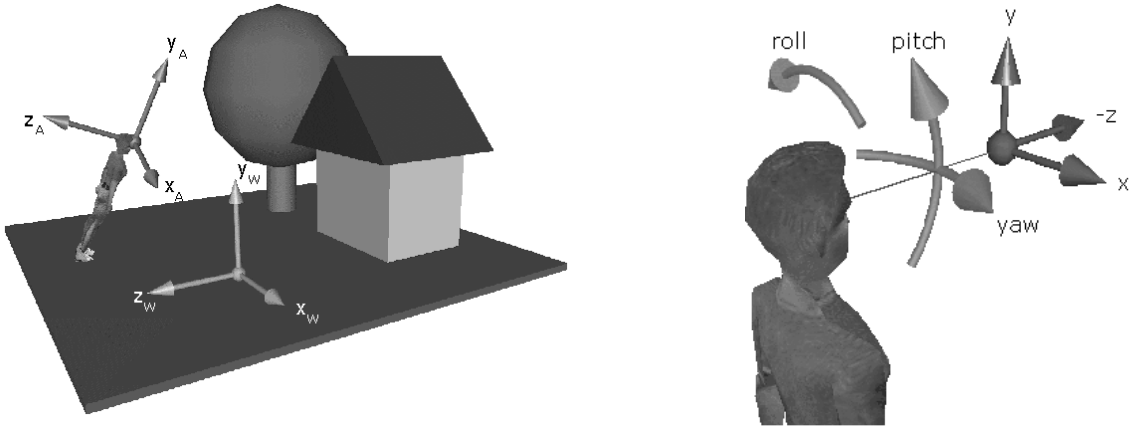
3.3 Szenario I: Interaktion in virtuellen Welten

Im Zentrum des ersten Testszenarios steht eine desktop-orientierte Virtual-Reality Applikation (DVA) zur Interaktion in beliebigen virtuellen Welten. Der Benutzer kann sich dabei über verschiedene Modalitäten innerhalb der virtuellen Welt bewegen und vorhandene Objekte manipulieren. Das Funktionsvokabular des Software-Prototypen orientiert sich massgeblich an den Grundfunktionalitäten, die in der abstrakten Sprache VRML[1] definiert sind [72]. Dieser plattform- und geräteunabhängige Sprachstandard emöglicht auf einfache Weise, dreidimensionale interaktive Szenarien zu beschreiben und gleichzeitig die möglichen Interaktionen mit Objekten dieser virtuellen Welt zu spezifizieren. Da die zugrundeliegende Technologie für diese Benutzerstudie und allgemein auch für andere Teile dieser Arbeit von besonderer Bedeutung ist, werden Hintergründe zu Virtual-Reality Systemen und nähere Details zu dem VRML Sprachstandard in Anhang C kurz erläutert. Dabei wird wesentlich detaillierter sowohl auf zentrale Fachbegriffe, als auch auf systemtechnische Umsetzungen eingegangen.

3.3.1 Testsystem

Eine fundamentale Aufgabe bei der Konzeption dreidimensionaler Bedienschnittstellen besteht in der intuitiven Umsetzung der zentralen Interaktionsparadigmen Navigation, Manipulation und Kommunikation. Grundlage für eine, im Sinne des Anwenders einfache und effektive Bedienbarkeit der Systemfunktionalitäten ist die Lösung des Orientierungsproblems in der virtuellen Welt, d.h. wie kann der Benutzer virtuelle Größenverhältnisse abschätzen und sowohl seine Blickrichtung als auch seine aktuelle Position ändern. Bezogen auf dieses Funktionsspektrum konzentriert sich das Leistungsangebot des Testsystems ausschließlich auf Aspekte der Navigation, d.h. die Versuchspersonen müssen sich durch eine virtuelle Welt bewegen. Als Darstellungsform wird eine First-Person-View gewählt, bei der der Benutzer direkt sieht, was eine

[1] **VRML** = Virtual-Reality Modelling Language



(a) (b)

Abb. 3.2: (a) Zusammenhang zwischen einem festen Weltkoordinatensystem WKS und einem dazu lokal veränderlichen Avatarkoordinatensystem AKS und (b) genauere Spezifikation des benutzerzentrierten AKS mit Achsenbeschriftungen (x,y,z) und charakteristischen Bezeichnern (yaw, pitch, roll) für die Drehungen um die einzelnen Achsen

virtuelle Person in der Szene sehen würde und sich damit unmittelbar in die Szene versetzt fühlt. Obwohl es auch mit dem verwendeten Software-Prototypen ohne weiteres möglich wäre, wird explizit kein Avatar in der Szene abgebildet. Unter Verwendung dieses Darstellungsparadigmas ist eine Änderung der aktuellen Position oder Orientierung gleichbedeutend mit der Bewegung einer virtuellen Kamera durch die dreidimensionale Szene. Die Testperson kann dabei das komplette Spektrum an Bewegungsmöglichkeiten ausschöpfen.

In VRML werden sowohl die einzelnen Objekte in der 3D Szene als auch etwaige Benutzerinteraktionen mit Bezug zu einem festen, rechtshändig-orthogonalen Weltkoordinatensystem (WKS) definiert, das durch die Achsen x_w, y_w und z_w aufgespannt wird. Die Navigation innerhalb dieser Welt kann durch die Transformation eines lokalen, avatarbezogenen Koordinatensystems (AKS) beschrieben werden, welches durch die, jeweils rechtwinklig zueinander stehenden Achsen x_a, y_a und z_a aufgespannt wird. Auf diese Weise ist beispielsweise ein Blick nach rechts auf eine Rotation der virtuellen Kamera um die vertikale Achse der Darstellungsebene (y_a) übertragbar. Abbildung 3.2(a) verdeutlicht den Zusammenhang zwischen dem Welt- und dem Avatarkoordinatensystem. Mathematisch lassen sich Navigationsoperationen durch die Transformationsbeziehung $p_{neu} = T_{4x4} \cdot p_{alt}$ ausdrücken, wobei ein Objekt p_{alt} in WKS linksseitig mit einer homogenen Transformationsmatrix T_{4x4}, die sowohl translatorische als auch rotatorische Veränderungen abdeckt, multipliziert wird und als Resultat eine modifizierte Position und Orientierung p_{neu} des Ausgangsobjektes liefert. Die Verwendung homogener Koordinaten ermöglicht in diesem Zusammenhang einen einheitlichen Beschreibungsformalismus aller geometrischen und perspektivischen Transformationen [48].

In Bezug auf die einzelnen AKS-Achsen können insgesamt sechs verschiedene Bewegungsfreiheitsgrade identifiziert werden. Diese atomaren Bewegungen entsprechen den möglichen Benutzerinteraktionen und werden im Folgenden jeweils mit einer charakteristischen Abkürzung erläutert. Betrachtet man rein translatorische Bewegungen, so kann sich die Testperson in der horizontalen Bezugsebene (y_a, z_a-Ebene) vorwärts und rückwärts bewegen (TVR), oder ihre

Cluster	Aktion	Bewegung	Richtung	AKS-Achsenbezug	T	A	H	K	S
TVR	A_1	Translation	vorwärts	entlang negativ z_a	x	x	x	-	x
	A_2	Translation	zurück	entlang positiv z_a	x	x	x	-	x
TLR	A_3	Translation	rechts	entlang positiv x_a	x	x	x	-	x
	A_4	Translation	links	entlang negativ x_a	x	x	x	-	x
TOU	A_5	Translation	noch oben	entlang positiv y_a	x	x	x	-	x
	A_6	Translation	noch unten	entlang negativ y_a	x	x	x	-	x
RLR	A_7	Rotation	rechts	negative Drehung um y_a	x	x	x	x	x
	A_8	Rotation	links	positive Drehung um y_a	x	x	x	x	x
ROU	A_9	Rotation	nach oben	positive Drehung um x_a	x	x	x	x	x
	A_{10}	Rotation	nach unten	negative Drehung um x_a	x	x	x	x	x
RRO	A_{11}	Rotation	rechts	negative Drehung um z_a	x	x	x	x	x
	A_{12}	Rotation	links	positive Drehung um z_a	x	x	x	x	x

Tab. 3.1: Kommando-Cluster und prinzipiell mögliche System-Aktionen in dem DVA Szenario zusammen mit einer mathematischen Beschreibung bezogen auf das AKS aus Abbildung 3.2(b); zusätzlich ist die Umsetzung der einzelnen Aktionen für die jeweiligen Modalitätengruppen angegeben, wobei ein '**x**' für eine vollständige und ein '**-**' für eine partiell eingeschränkte Nutzung steht

Position in der Darstellungsebene (x_a, y_a-Ebene) horizontal nach links und rechts (TLR) bzw. vertikal nach oben und unten (TOU) verändern. Bezogen auf die rein rotatorische Bewegungen sind in Abbildung 3.2(b) bereits Namen für die Drehungen um die einzelnen Koordinatenachsen enthalten. Eine Drehung des sichtbaren Bereiches nach links und rechts wird als *yaw* bezeichnet (RLR), eine Drehung nach oben oder unten nennt man *pitch* (ROU) und eine Kippen der dargestellten Szene um die z_a-Achse heißt *roll* (RRO). Letztere entspricht einer Drehung um die optische Achse der virtuellen Kamera.

In Tabelle 3.1 werden die sechs Kommando-Cluster mit den daraus resultierenden zwölf möglichen Bewegungsformen in Relation zu den zugrundeliegenden Transformationen im AKS gesetzt. Zusätzlich enthält die Tabelle Informationen über die prinzipielle modalitätenspezifische Umsetzung der jeweiligen Systemaktion. Dabei steht ein '**x**' für eine vollständige und ein '**-**' für eine partiell eingeschränkte Nutzung. Lediglich von der Eingabe per Kopfgesten wird nicht das volle Leistungsspektrum erwartet, da in unterschiedlichen Benutzerstudien [27, 96, 132] nachgewiesen wurde, dass sich Kopfgesten größtenteils aus rotatorischen Bewegungen zusammensetzen und rein translatorische Bewegungen von den Benutzern als hochgradig umständlich und nicht-intuitiv bewertet wurden. In dem DVA Szenario besteht dennoch die Möglichkeit, auch mit dem Kopf translatorische Bewegungen durchzuführen, es wird den Probanden in der initialen Trainingsphase aber nicht explizit gezeigt. Wenn eine Versuchsperson jedoch von sich aus translatorische Kopfgesten verwendet, so wird die Eingabe vom Versuchsleiter akzeptiert.

Abschließend sei noch kurz erwähnt, dass sich sämtliche Bewegungen auf eine Bewegung des virtuellen Avatars beziehen. Da dieser Avatar nicht direkt in der Szene abgebildet wird, hat das für den Benutzer den Effekt, dass sich der aktuelle Bezugspunkt in der Welt genau in inverser Weise bewegt. Das bedeutet, dass wenn der Benutzer eine Bewegung nach vorne auslöst, die Welt als Resultat dieser Transformation auf die Testperson zukommt oder wenn sich der Benutzer um die optische Achse der Kamera nach rechts kippt, es den Anschein hat, als würde die Welt nach links kippen.

Software-technische Umsetzung

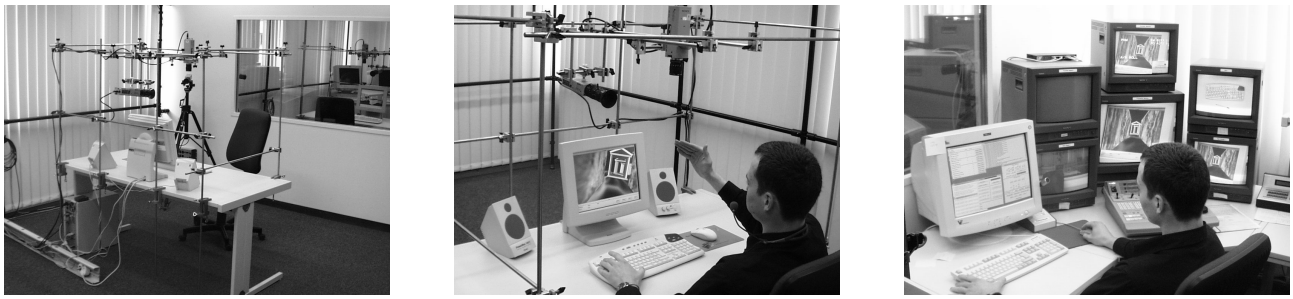
Das im Benutzertest verwendete visuelle Frontend basiert auf dem plattformunabhängigen, frei erhältlichen FreeWRL-Browser [173]. Verglichen mit anderen VRML-Browsern weist FreeWRL zwar den Nachteil auf, dass keine Gravitation simuliert werden kann und auch die Kollisionsdetektion nur prototypisch umgesetzt ist, dafür steht aber der Source-Code des weitgehend in PERL implementierten Browsers frei zur Verfügung und kann unter moderatem Aufwand an die speziellen Bedürfnisse eines multimodalen Testszenarios angepasst werden. Die ursprüngliche FreeWRL Funktionalität wurde für die Benutzerstudie an verschiedenen Stellen modifiziert und erweitert. Zu diesen Änderungen gehörte vorangig die Einführung diskreter, inkrementeller Bewegungen, da im Vergleich zu rein taktilen Interaktionen eine kontinuierliche Navigation mit semantisch höherwertigen Modalitäten wegen eines fehlenden direkten Feedbacks schwer zu realisieren ist. Details zu dem FreeWRL Browser werden in [10, 156] erläutert.

Im Benutzerinterface ist unterhalb des Browsers eine Feedback-Komponente integriert (siehe Abbildung 3.4a), die den Benutzer über die korrekte Ausführung der einzelnen Systemfunktionalitäten informiert und ihn mit verschiedenen Status-Informationen versorgt. Das Feedback-Fenster enthält neben einfachen Textfeldern und selbst-erklärenden graphischen Icons für die zentralen Systemfunktionalitäten auch eine Liste der letzten Interaktionen. Zusätzlich zum visuellen Feedback ist ebenfalls ein akustisches Feedback in Form von kurzen Signaltönen zur Bestätigung durchgeführter Systemaktionen realisiert. Die einzelnen Kontrollparameter, beispielsweise die Update-Frequenz und die Detail-Stufe, können direkt vom Benutzer nach individuellen Vorlieben und Bedürfnissen angepasst werden.

3.3.2 Testumgebung

Die Benutzerstudie wird im Usability-Labor des Lehrstuhls für Mensch-Maschine-Kommunikation der Technischen Universität München durchgeführt [80]. Dieses Labor besteht aus zwei getrennten Räumlichkeiten und bietet optimale Rahmenbedingungen für einen störungsfreien Versuchsablauf. Die Testperson befindet sich in einem *Testraum* und bedient an einem normalen Schreibtischarbeitsplatz das Testinterface. Zur Beobachtung der Versuchsperson ist dieser Raum mit mehreren Video-Kameras und Mikrophonen ausgestattet. Der Versuchsleiter befindet sich im *Kontrollraum*, in dem die Interaktionen der Testperson auf verschiedenen Video-Monitoren überwacht und für eine spätere Offline-Analyse aufgezeichnet werden. Von hier werden sowohl die Testabläufe koordiniert als auch die einzelnen Funktionalitäten des Software-Prototypen gesteuert. Die beiden Räume sind durch einen halbdurchlässigen Spiegel getrennt, so dass zwar der Versuchsleiter direkt sehen kann, was die Versuchsperson macht, nicht aber umgekehrt. Spezifische Details zu dem Aufbau des Usability-Labor und der technischen Ausstattung werden ausführlich in [111] diskutiert.

Die Abbildung 3.3 vermittelt einen Eindruck über den unmittelbaren Testaufbau für die Interaktion mit dem multimodalen VRML Browser in dem DVA-Szenario. Die Testperson befindet sich an einem herkömmlichen Computerarbeitsplatz, der mit einem 15" Touchscreen-Monitor, Maus, Tastatur und Multimediahardware ausgestattet ist. Programmtechnisch wird der Touchscreen wie ein herkömmliches direkt-manipulatives GUI behandelt. Interaktionen mit dem Touchscreen werden auf Aktionen der Standard-Mouse abgebildet, d.h. wenn beispielsweise der Benutzer mit dem Finger eine Stelle auf dem Touchscreen berührt entspricht dieses Ereig-



(a) (b) (c)

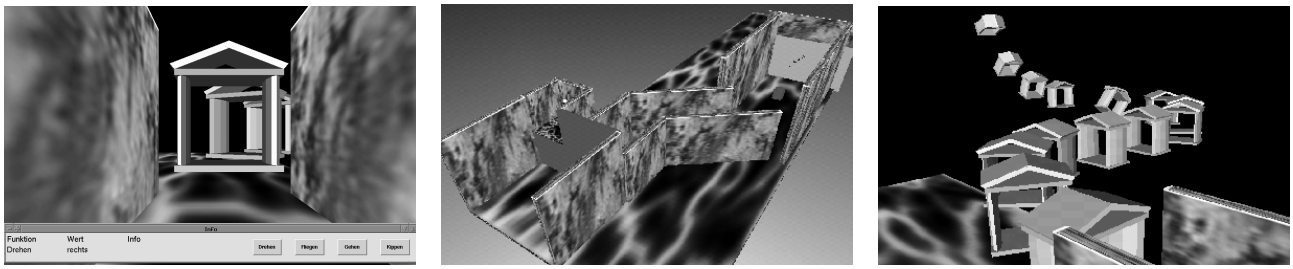
Abb. 3.3: Testumgebung zur Bedienung eines multimodalen VRML-Browsers im DVA-Szenario; (a und b) Spezifischer Aufbau im abgeschlossenen Testraum und (c) Arbeitsplatz des Versuchsleiters im Kontrollraum des Usability-Labors [111]

nis einem Einfach-Klick mit der linken Maustaste an der entsprechenden Position. Analog dazu lässt sich auch eine Drag-Funktion simulieren, indem ein Finger unter kontinuierlichem Druck auf die Bildschirmoberfläche über den anvisierten Bereich bewegt wird. Als Rückmeldung über die korrekte Durchführung einzelner Interaktionen ist ein akustisches Feedback in Form eines einfachen kurzen Signaltons realisiert, der jedes Mal ertönt, wenn eine Interaktion mit dem Touchscreen stattgefunden hat.

Um eine einfache Touchscreenbedienung zu ermöglichen, sitzt der Benutzer frontal relativ nah vor dem Monitor. An einem Trapez aus Stahlstangen, das um den Arbeitsplatz herum aufgebaut ist, können mehrere Video-Kameras variabel positioniert werden. In dem Testaufbau nimmt eine Kamera die Versuchsperson frontal von vorne auf, um eine Auswertung der Kopfgesten zu ermöglichen. Die zweite Kamera fokussiert den Tastaturbereich senkrecht von oben und gestattet auf diese Weise eine einfache Beobachtung potenzieller Handgesten. Eine dritte Kamera nimmt die gesamte Szene von schräg hinten auf, um die visuelle Darbietung der Aktionen im Testinface und die Benutzerinteraktionen im globalen Kontext zu erfassen. Ein Richtmikrophon nimmt die sprachlichen Äußerungen der Testperson auf. Über ein Video-Mischpult im Kontrollraum werden diese drei Kamerabilder und die Audioaufnahmen zeitsynchron auf ein S-VHS Videoband übertragen, das die Grundlage für die spätere Auswertung der Versuchsdaten bildet. Der Versuchsleiter kann zusätzlich über die Lautsprecher des Testrechners mit der Versuchsperson kommunizieren.

3.3.3 Aufgabenbeschreibung

In den vier Testblöcken der zweiten Phase müssen die Versuchspersonen jeweils unter Anwendung verschiedener Modalitäten durch einen Tunnel navigieren (siehe Abbildung 3.4(b)) und dabei pro Block drei Aufgabensequenzen lösen. Die erste Aufgabe besteht darin, sich durch die Rechts/Links-Biegungen zu bewegen. Dazu sind hauptsächlich translatorische und rotatorische Bewegungen in der horizontalen Bezugsebene des Benutzers anzuwenden (TFR, TLR und RLR). Nach erfolgreicher Passage der Biegungen befindet sich die Versuchsperson in einem längeren Gang. Am Ende dieses Ganges steht an der Wand eine Text-Nachricht, die um 45° relativ zum Boden des Tunnels gedreht ist. Die zweite Aufgabe besteht darin, das Sichtfeld der virtuellen Kamera in einer Weise zu verändern, dass der Schriftzug in einer rein horizontalen



(a) (b) (c)

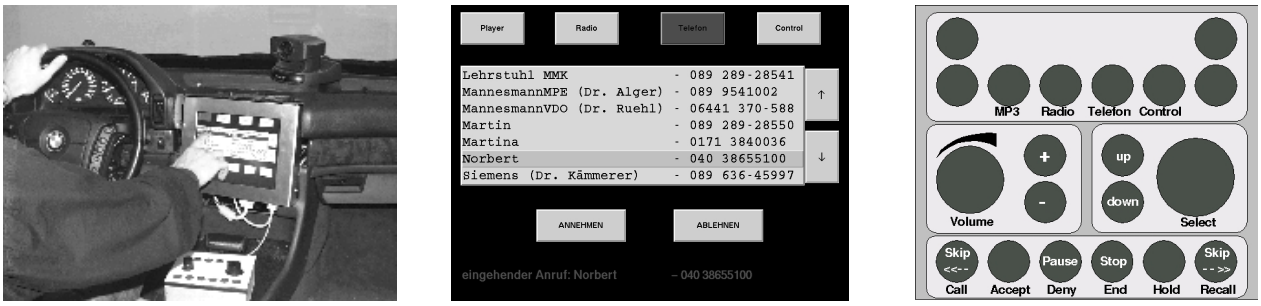
Abb. 3.4: (a) Darstellung des visuellen Benutzerinterfaces mit VRML Browser und Feedback-Komponente, sowie (b) eine globale Sicht auf die VRML Welt aus den vier Trainingsblöcken und (c) auf die Sequenz aus verschiedenen Torbögen der dritten Testphase

Lage klar abgelesen werden kann. Anschließend muss sich die Testperson wieder gerade zum Boden des Tunnels ausrichten. Die dazu benötigten Transformationen bestehen aus Drehungen um die optische Achse der Kamera (RRO). Auf dem Boden kurz vor dem Tunnelende befindet sich ein oben offener Kasten. Als dritte Aufgabe in dieser Phase muss die Versuchsperson von oben in die Kiste schauen und feststellen, was für ein Gegenstand darin versteckt ist um sich anschließend wiederum senkrecht auf den Boden zu stellen. Bei der Lösung dieser Aufgabe sind translatorische Bewegungen in der Bildebene (TOU) und Drehungen um die horizontale Darstellungsachse (ROU) involviert.

Die Navigationsaufgaben im vollständig multimodalen Testblock der dritten Versuchsphase sind weitaus komplexer. Dabei werden auch die einzelnen Bewegungsgruppen wesentlich stärker im Kontext der Aufgabe integriert. Der Ausgangspunkt ist analog zu dem Szenario aus der zweiten Versuchsphase der Tunneleingang. Die Versuchsperson muss zunächst in den kleinen Zwischengang navigieren, der ansatzweise links oben am Anfang des Tunnels in der Abbildung 3.4(b) zu erkennen ist. Dort befindet sich eine Textmeldung mit der nächsten Aufgabe. Der Benutzer muss zurück zum Ausgangspunkt navigieren und von dort aus dem Labyrinth bis zum Ende zu folgen. Dabei müssen die schon bekannten Rechts/Links-Biegungen passiert werden. Anders als in der zweiten Phase des Versuchs ist der Tunnel hinten offen und es schließt sich eine Sequenz von 21 Torbögen an (siehe Abbildung 3.4(c)). Die dritte Aufgabe im Hauptversuch besteht darin, sich möglichst schnell durch die einzelnen Bögen zu bewegen. Als Nebenbedingung wird verlangt, dass man aufrecht durch die Tore navigiert, d.h. die horizontale Darstellungsachse (x_a) sollte parallel zu dem Fundament der Bögen ausgerichtet sein. Speziell in dem letzten Teil müssen die erlernten Bewegungen aus der Trainingsphase wechselseitig kombiniert werden.

3.4 Szenario II: Infotainmentsysteme im Automobil

Im Zentrum des zweiten Testszenarios steht die multimodale Bedienung von automotiven Infotainment Applikationen (AIA) über verschiedene Eingabekanäle. Das zugrundeliegende Funktionsvokabular hat sich im Laufe des FERMUS-Projektes [82, 83, 84] für verschiedene Untersuchungen und als Demonstrator für die Evaluierung unterschiedlicher Fehlermanagementstrategien kontinuierlich weiterentwickelt. Im Rahmen der hier beschriebenen Benutzerstudie wird eine frühe Evolutionsstufe des Testsystems eingesetzt, bei der der Funktionsschatz auf die wichtigesten Grundfunktionen beschränkt ist.



(a) (b) (c)

Abb. 3.5: (a) Arbeitsumgebung für die Testpersonen im AIA-Szenario mit Touchscreen und Tastenkonsole eingebaut in das Laborfahrzeug, (b) struktureller Aufbau der visuellen Anzeigekomponente dargestellt im Telefon-Modus und (c) Belegungsplan für die Tastenkonsole

3.4.1 Testsystem

Losgelöst von dem visuellen Erscheinungsbild besteht die primäre Funktionalität des Interfaces in der integralen Steuerung von Audio- und Kommunikationsgeräten in einer Automobilumgebung. Abbildung 3.5(a) zeigt den strukturellen Aufbau des Testinterfaces. Das Design kann in vier von einander unabhängige horizontale Bereiche eingeteilt werden. Mittels der oberen Buttons hat der Benutzer die Möglichkeit, die vier grundsätzlichen Betriebsmodi des Testsystems anwählen. Dies sind ein MP3-Modul, ein Radiomodul, ein Telefonmodul und ein dediziertes Kontroll-Modul.

Als zentrales Designelement beinhaltet der mittlere Teil des Interface eine Listendarstellung, in der modusabhängig die aktuell verfügbaren Elemente dargestellt und ausgewählt werden können (Lieder, Radiosender und Telefonbucheinträge). In dieser Liste kann sowohl über direktes Anwählen einzelner Einträge, als auch indirekt über die beiden rechts angrenzenden Richtungs-Buttons navigiert werden. Am unteren Ende der Liste angekommen, springt der Fokus automatisch wieder auf den ersten Eintrag, d.h. bezogen auf die visuelle Repräsentation wird die zugrunde liegende Listenstruktur zyklisch angehängt. Die Interaktionsflächen im unteren Bereich des Interfaces wechseln in Abhängigkeit vom aktuellen Modus und dem jeweiligen Kontext. Als letzten horizontalen Bereich enthält das Interface am unteren Rand zusätzlich eine frei parametrisierbare Informationszeile, die den Benutzer über den momentanen Status des Systems informiert, beispielsweise um eingehende Anrufe neben dem akustischen Signal auch optisch zu präsentieren.

Im MP3-Modus existieren fünf kontextspezifische Buttons. Analog zu einem herkömmlichen CD-Player kann in der aktuellen Wiedergabeliste vor- und zurückgesprungen (*Skipping*), das Abspielen des ausgewählten Titels gestartet, die Wiedergabe des laufenden Titels gestoppt und die Pause-Funktion aktiviert werden. Der Radiomodus verfügt demgegenüber lediglich über drei moduseigene Buttons, mit denen ein Radiosender aus der Liste ausgewählt bzw. zum nächsten oder vorherigen Radiosender gewechselt werden kann.

Die Telefonfunktionalitäten des Testinterfaces beschränken sich auf die wichtigsten Grundfunktionen eines normalen Autotelefons. Der Telefonmodus ist von der logischen Struktur noch einmal in drei Submodi unterteilt, die jeweils über zwei Buttons verfügen. Standardmäßig befindet sich der Benutzer im ersten Submenü, in dem er einen Teilnehmer aus der Liste anwählen

Cluster	Aktion	Beschreibung	T	A	H	K	S
PLY	A_1	MP3-Player starten	x	x	-		x
	A_2	MP3-Player stoppen	x	x	-		x
	A_3	Player in Pause-Mode setzen	x	x			x
	A_4	Skippen zum nächsten Titel	x	x	x	x	x
	A_5	Skippen zum vorherigen Titel	x	x	x	x	x
	A_6	Aktivierung der Repeat-Funktion		x	-		x
	A_7	Aktivierung der Scan-Funktion		x			x
	A_8	Aktivierung der Zufallswiedergabe		x			x
RAD	A_9	Skippen zum nächsten Radiosender	x	x	x	x	x
	A_{10}	Skippen zum vorherigen Radiosender	x	x	x	x	x
TEL	A_{11}	Anruf initiieren	x	x	x		x
	A_{12}	Aktuelle Verbindung beenden	x	x	x		x
	A_{13}	Eingehenden Anruf annehmen	x	x	x	x	x
	A_{14}	Eingehenden Anruf ablehnen	x	x	x	x	x
	A_{15}	Wahlwiederholung	x	x			x
LIS	A_{16}	Einen Eintrag in der Liste nach oben gehen	x	x	x	x	x
	A_{17}	Einen Eintrag in der Liste nach unten gehen	x	x	x	x	x
	A_{18}	Aktuellen Eintrag der Liste auswählen	x	x	x	-	x
CTL	A_{19}	MP3-Modus aktivieren	x	x			x
	A_{20}	Radio-Modus aktivieren	x	x			x
	A_{21}	Telefon-Modus aktivieren	x	x			x
	A_{22}	Kontrollmodus aktivieren	x	x			x
	A_{23}	Lautstärke erhöhen	x	x	x	-	x
	A_{24}	Lautstärke erniedrigen	x	x	x	-	x
	A_{25}	Mute-Funktion aktivieren	x	x	-		x

Tab. 3.2: Kommando-Cluster und mögliche System-Aktionen in dem AIA Szenario; zusätzlich ist die Umsetzung der einzelnen Aktionen für die jeweiligen Modalitäten angegeben, wobei ein '**x**' für eine vollständige und ein '**-**' für eine partiell eingeschränkte Nutzung steht und ein leeres Feld anzeigt, das die jeweilige Aktion mit der spezifischen Modalität nicht ausgelöst werden kann.

oder die zuletzt gewählte Rufnummer erneut wählen kann. Empfängt das Testsystem einen eingehenden Anruf, wie in Abbildung 3.5(b) dargestellt, wird in das zweite Submenü gewechselt und der Benutzer kann das Gespräch entweder akzeptieren oder ablehnen. Kommt eine Verbindung zwischen zwei Kommunikationspartnern zustande, so aktiviert das Interface in das dritte Submenü, in dem das Gepräch beendet oder temporär gehalten werden kann.

Im Kontrollmodus besteht die Möglichkeit, die Lautstärke der Audiowiedergabe in diskreten Stufen anzuheben bzw. abzusenken. Weiterhin kann die Lautstärke auf zehn Prozent des aktuellen Wertes reduziert werden (*Mute-Funktion*). Eine erneute Aktivierung dieser Funktion bewirkt eine Wiederherstellung des ursprünglichen Wertes. Darüber hinaus können hier auch spezielle Voreinstellungen hinsichtlich verschiedener Systemparameter und Feedbackverhalten definiert und interaktiv manipuliert werden, um sie an die persönlichen Bedürfnisse der einzelnen Benutzer anzupassen.

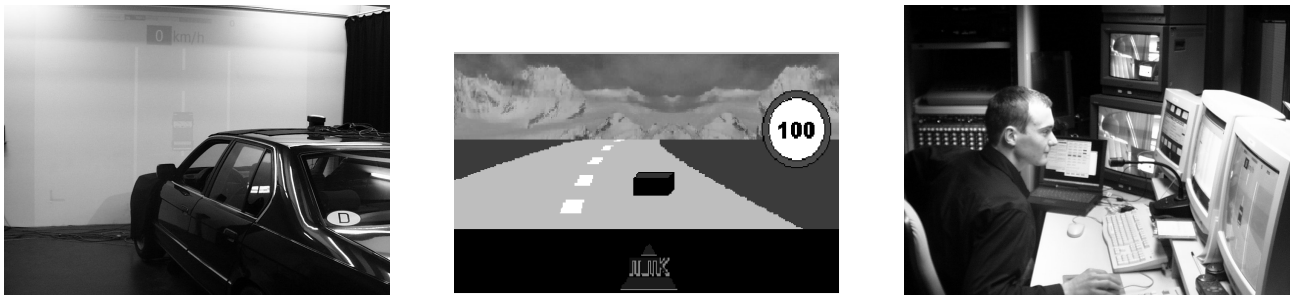
Tabelle 3.2 fasst die Funktionalitäten des Testsystems in fünf Kommando-Cluster zusammen: Wiedergabesteuerung des MP3-Players (PLY), Radiofunktionen (RAD), Telefonfunktionen (TEL), Steuerungsfunktionen zum Navigieren und Auswählen in der Liste (LIS) und schließlich Kontrollfunktionen (CTL) für Modusauswahl und Lautstärkeeinstellungen. Analog zu dem DVA Szenario (Tabelle 3.1) wird den einzelnen Systemfunktionalitäten eine Aussage über die modalitätenspezifischen Umsetzbarkeit zugeordnet. Dabei steht ein '**x**' wiederum für eine vollständige und ein '**-**' für eine partiell eingeschränkte Nutzung. Wenn a priori sichergestellt ist, dass eine Funktionalität mittels einer spezifischen Modalität nicht initiiert werden kann, so enthält die Tabelle an der entsprechenden Stelle keinen Eintrag. Beispielsweise lassen sich die einzelnen Betriebsmodi weder per Hand- noch per Kopfgestik ansteuern. Insgesamt bietet das Testinterface im AIA-Szenario 22 verschiedene Systemaktionen.

Kombinierte Tasten-/Reglerkonsole

Neben dem Touchscreen steht ein weiteres haptisches Bedienelement zur Verfügung. Mittels eines separaten Tastenblocks, der ebenfalls in der Mittelkonsole des Testfahrzeugs angebracht ist, hat der Benutzer die Möglichkeit, die einzelnen Funktionalitäten des Systems alternativ zu den direkt manipulativen Interaktionsflächen des Touchscreens auch über konventionelle Tasten zu steuern. Diese Tastenkonsole wurde bereits erfolgreich auch in anderen Benutzerstudien eingesetzt [111]. Die Belegung der einzelnen Tasten für das AIA-Szenario ist in Abbildung 3.5(c) dargestellt. Dabei wurde weitgehend versucht, eine geometrische Analogie zu den Interaktionsflächen des Touchscreens herzustellen.

Im oberen Bereich lassen sich über die vier mittleren Tasten die Modi des Testsystems auswählen. Der mittlere Bereich mit seinen zwei Dreh-Drückstellern ermöglicht eine kontinuierliche Einstellung der Lautstärke (linker Drehknopf) und die Auswahl eines Elementes aus der aktuellen Liste (rechter Drehknopf). Dabei bewirkt eine Drehung nach links eine Bewegung des Listenfokus nach oben und eine Drehung nach rechts entsprechend eine Bewegung nach unten. Der aktuelle Eintrag wird durch einen Druck auf den Steller ausgewählt. Kontextspezifisch wird dann entweder der aktuelle MP3-Titel bzw. der selektierte Radiosender gespielt oder der aktuelle Teilnehmer aus der Telefonliste angerufen. Als Alternative zu den Drehdrückstellern können auch die nebenstehenden Wipptasten verwendet werden. Der untere Bereich des Tastenblocks wird ebenfalls modusspezifisch genutzt. Das Skippen in der Wiedergabeliste mit den beiden äußeren Tasten ist sowohl im MP3 als auch im Radiomodus möglich. Die mittleren vier Tasten haben im Radiomodus keine Bedeutung. Im Gegensatz zum Touchscreen stehen im Telefonmodus alle Funktionalitäten parallel zur Verfügung.

Der Tastenblock übermittelt seine Befehle über eine definierte Schnittstelle an das Interface bzw. die anderen Applikationen. Dies geschieht in diesem Fall aber nicht direkt, sondern über eine eigens dafür entwickelte Anwendung (*KeyMapper*). Bei Betätigung der Tasten oder Drehsteller generiert der Tastenblock keine eigenen Befehle im Sinne der Applikation, sondern definierbare *Tastatur-Events*, d.h. für jede Aktion wird ein charakteristisches ASCII-Zeichen ausgegeben. Den Drehreglern wird pro Richtung eine dedizierte Buchstabenfolge zugeordnet, die bei Betätigung des Knopfes zyklisch ausgegeben werden. Systemseitig ist dabei nicht unterscheidbar, ob die Events von einer regulär an das System angeschlossenen Standard-Tastatur oder von dem speziell entwickelten Tastenblock erzeugt wurden. Der KeyMapper fragt die



(a) (b) (c)

Abb. 3.6: Testumgebung für die Benutzerstudien in der Automobilumgebung; (a) Testraum mit dem Versuchsfahrzeug vor der Projektionsfläche, (b) Ausschnitt aus der dreidimensionalen Fahraufgabe und (c) Arbeitsumgebung des Versuchsleiters im abgetrennten Regie-Raum

eingehenden Events ab und generiert daraus, je nach empfangenem Zeichen, den passenden Befehl der hinterlegten kontextfreien Grammatik. Die Zuordnung ist dabei in einer separat zu ladenden Datei festgelegt (*EventMap*), so dass zur Steuerung unterschiedlicher Applikationen und Interfaces lediglich eine neue EventMap geladen werden muss. Darüber hinaus ermöglicht dieser Formalismus auch eine dynamische Umkonfiguration des aktuellen Befehlswortschatzes zur Laufzeit. Damit ist es beispielsweise möglich, die Bedeutung der Tasten kontextspezifisch umzulegen oder in gewissen Situationen ausgewählte Teile der haptischen Bedienung komplett zu sperren.

3.4.2 Testumgebung

Um aussagekräftige Ergebnisse zu erhalten, wird die Benutzerstudie bezogen auf das AIA-Szenario in einer Umgebung durchgeführt, die eine gewisse Nähe zu den Rahmenbedingungen und Bediensituationen in einem realen Automobil suggeriert. Der Lehrstuhl für Mensch-Maschine-Kommunikation verfügt dazu über ein spezielles Usability-Labor, das sog. *Navigation Lab* [82], in dem der Umgang mit multimedialen Interfaces vor dem Hintergrund einer konkreten Fahraufgabe untersucht und bewertet werden kann. Auf diese Weise ist es möglich, auch unter Laborbedingungen eine angemessene kognitive Belastung der Versuchsperson generieren zu können. Es besteht wie in realen Feldtests die Notwendigkeit, explizit auf die Gewährleistung der Verkehrssicherheit achten zu müssen.

Das Navigation-Lab besteht ebenfalls aus zwei getrennten Bereichen. In dem größeren der beiden Räume, dem Testraum, befindet sich eine BMW-Limousine, die zentral vor einer großen Leinwand positioniert ist (siehe Abbildung 3.6(a)). Auf den ersten Blick wirkt dieses Auto bis auf den fehlenden Motor wie eine identische Kopie eines normalen Serienfahrzeugs. Der Innenraum ist jedoch im Rahmen von mehreren Benutzerstudien gegenüber dem Original an verschiedenen Stellen modifiziert und auf diese Weise an die speziellen Testbedürfnisse angepasst worden. Zu diesen Änderungen gehört beispielsweise der Einbau eines 10”-Touchscreens in der Mittelkonsole. Dieser Monitor ist mit einem einfachen, handelsüblichen PC unter der Motorhaube verbunden und dient als zentraler MMI-Rechner sowohl für die Darstellung der Bedienoberfläche als auch für die Steuerung der Audio-Applikationen. Zudem ist das Testfahrzeug mit mehreren Videokameras und Mikrophonen ausgestattet, mit denen die Interaktionen

der Versuchsperson aus verschiedenen Blickwinkeln optimal beobachtet werden können. Um die Ablenkung der Probanden durch externe Einflüsse so gering wie möglich zu halten, kann der Bereich links, rechts und hinter dem Wagen durch schwarze Tücher verhängt werden. Details zu den technischen Spezifika des Labors werden ausführlich in [50, 111] behandelt.

In direkter Analogie zu dem speziellen Aufbau im DVA-Szenario (siehe Abschnitt 3.3.2) fokussiert eine Kamera die Versuchsperson frontal von vorne, um auftretende Kopfgesten zu erfassen. Eine zweite Kamera ist an der Fahrzeugdecke installiert, um senkrecht von oben im Bereich um den Schalthebel mögliche Handgesten aufzunehmen. Eine weitere Kamera ist im Fond des Wagens positioniert und nimmt neben dem Testinterface die direkten taktilen Interaktionen mit dem Touchscreen und der speziellen Tastenkonsole auf. Die Audio- und Videosignale werden in den zweiten Raum, den sog. *Regieraum* übertragen, in dem sie zeitsynchron aufgezeichnet werden. Sie dienen als Grundlage für die Simulation einer sprach- und gestenbasierten Eingabe. Der Versuchsleiter interpretiert die semantisch höherwertigen Interaktionen der Versuchsperson und löst über das Rechnernetzwerk ferngesteuert die korrespondierende Funktionalität auf dem MMI-Rechner aus.

Fahraufgabe

In einem im normalen Straßenverkehr bewegten Auto stellt die Bedienung von Infotainment-Applikationen lediglich eine Sekundäraufgabe dar, welche dem sicheren Steuern des Fahrzeugs eindeutig untergeordnet ist. Der Fahrer muss den größten Teil seiner Aufmerksamkeit dieser Primäraufgabe zuwenden, jede Ablenkung kann unter Umständen eine signifikante Beeinträchtigung der Fahrsicherheit nach sich ziehen. Damit auch in einem Benutzertest zur Evaluierung automotiver MMIs diese real-existierenden Randbedingungen simuliert werden können ist es nötig, die Versuchsperson mit einer Aufgabe zu konfrontieren, die mit der kognitiven Belastung beim normalen Autofahren verglichen werden kann. Zu diesem Zweck ist am Lehrstuhl eine einfache Fahraufgabe entwickelt worden [82, 142].

Über einen Video-Beamer, der an der Decke des Labors befestigt ist, wird auf die weiße Wand vor dem Testfahrzeug eine virtuelle Straße in räumlicher Ausdehnung projiziert, die das Sichtfeld des Benutzers vollständig einnimmt. Die dreidimensionale Visualisierung dieser Straße ermöglicht eine bessere Antizipation der Fahrstrecke ähnlich einer normalen Autofahrt. Mittels der Pedalerie und dem Lenkrad im Testfahrzeug kann der Benutzer auf die Geschwindigkeit bzw. auf die Position des fiktiven Fahrzeugs Einfluss nehmen. Analog zu dem DVA-Szenario erfolgt die Darstellung im First-Person-Mode, was den natürlichen Eindruck einer normalen Autofahrt weiter steigert. Für den sicheren Umgang mit dem Simulator bedarf es einer gewissen Eingewöhnungsphase, da im Gegensatz zu weitaus aufwendigeren Fahrsimulatoren keinerlei Rückmeldungen über auftretende Trägheitskräfte umgesetzt sind, die beispielsweise beim Beschleunigen oder Bremsen in einem realen Fahrzeug aufteten würden.

Die Aufgabe der Versuchsperson besteht darin, dem virtuellen Straßenverlauf zu folgen und dabei eine vorgegebene Geschwindigkeit einzuhalten. Zur Ablenkung der Probanden können verschiedene audio-visuelle Effekte eingespielt werden, unter anderem Lichtblitze, wechselnde Geschwindigkeitsbeschränkungen, Lärmbelästigungen durch hupende Fahrzeuge oder plötzlich auf der Fahrbahn auftauchende Hindernisse. Darüber hinaus ist es möglich, die Darstellung der Szene in Bezug auf verschiedene Umgebungssituationen wie Beleuchtungscharakteristika (Tag- oder Nachtfahrt) oder verschiedene Jahreszeiten flexibel anzupassen. Verlässt die Versuchsper-

son die Fahrbahn, so wird zusätzlich ein akustisches Warnsignal eingespielt, um die Aufmerksamkeit wieder stärker auf die Primäraufgabe zu richten. Die Fahraufgabe kann komplett durch den Versuchsleiter gesteuert werden. In einem eigenen Protokoll werden sämtliche Fahrdaten wie Geschwindigkeit, Lenkwinkel, Zusammenstöße und Fahrbahnabweichungen festgehalten, um später die Fahrleistung der einzelnen Probanden im Kontext der jeweiligen Bediensituation interpretieren zu können.

3.4.3 Aufgabenbeschreibung

Nachdem die Testperson in einer gemeinsamen Trainingsphase mit dem Versuchsleiter sowohl die Funktionalitäten des Testinterfaces als auch den Umgang mit der Fahraufgabe erlernt hat, muss sie in der zweiten Versuchsphase jeweils mit unterschiedlichen Modalitäten eine Sequenz von 16 verschiedenen Bedienaufgaben erledigen. Diese Aufgaben sind bezogen auf die in Tabelle 3.2 identifizierten Kommando-Cluster annähernd gleichverteilt. Abschnitt A.1 enthält eine detaillierte Auflistung der einzelnen Aufgaben. Neben der Bedienung des Infotainment-Systems müssen die Probanden gleichzeitig eine relativ einfach gehaltene Fahraufgabe bewältigen.

Im eigentlichen Hauptversuch in der Phase 3 werden die Versuchsteilnehmer mit einer deutlich komplexeren Fahraufgabe konfrontiert. Um einem Trainingseffekt aus der zweiten Versuchsphase vorzubeugen ist die Strecke jetzt wesentlich kurviger, es werden häufiger unterschiedliche Geschwindigkeitsbeschränkungen eingeblendet und die Testpersonen werden sowohl durch entgegenkommende Fahrzeuge als auch durch Hindernisse auf der eigenen Fahrbahn stärker gefordert. Vor diesem Hintergrund müssen 23 verschiedene Bedienaufgaben abgearbeitet werden, bei dem das volle Spektrum an zur Verfügung stehenden Modalitäten ausgeschöpft werden kann. Die Beschreibung der einzelnen Aufgaben kann der zweiten Tabelle in Abschnitt A.1 entnommen werden.

3.5 Werkzeuge zur Versuchsdurchführung

Um aus einer Benutzerstudie aussagefähige und vergleichbare Daten zu gewinnen, müssen die Rahmenbedingungen und der Versuchsablauf für alle Versuchspersonen identisch sein. Für die Auswertung der audiovisuellen Versuchsdaten ist es zudem wichtig, sowohl sämtliche Interaktionen der Probanden mit dem Testsystem als auch kontrollierende Eingriffe von Seiten des Versuchsleiters zeitgenau zu erfassen und mit den aufgenommenen Videodaten zu synchronisieren. In einem multimodalen System-Setup ist der Versuchsleiter durch die parallele Überwachung mehrerer Eingabekanäle einer enormen kognitiven Belastung ausgesetzt. Um dennoch eine effektive Versuchsdurchführung garantieren zu können muss der Wizard von möglichst vielen anderen Kontrollaufgaben entbunden bzw. durch das System aktiv unterstützt werden.

Aus diesem Grund ist im Rahmen der verschiedenen Projektarbeiten am Lehrstuhl eine spezielle, domäneninvariante Software-Umgebung zur semi-automatischen Durchführung von multimodalen Usability-Versuchen nach dem Wizard-of-Oz Prinzip entwickelt worden, die im Folgenden abkürzend als *UsaWiz* (Uasbility Wizard) bezeichnet wird [142]. Abbildung 3.7 gibt einen strukturellen Überblick über die einzelnen Komponenten des Systems. Das zugrundeliegende Konzept ermöglicht eine Programmierung und damit eine weitgehende Automatisierung der Versuchsabläufe. Basierend auf einer Scriptdatei (*Runchart*), werden von einer zentralen

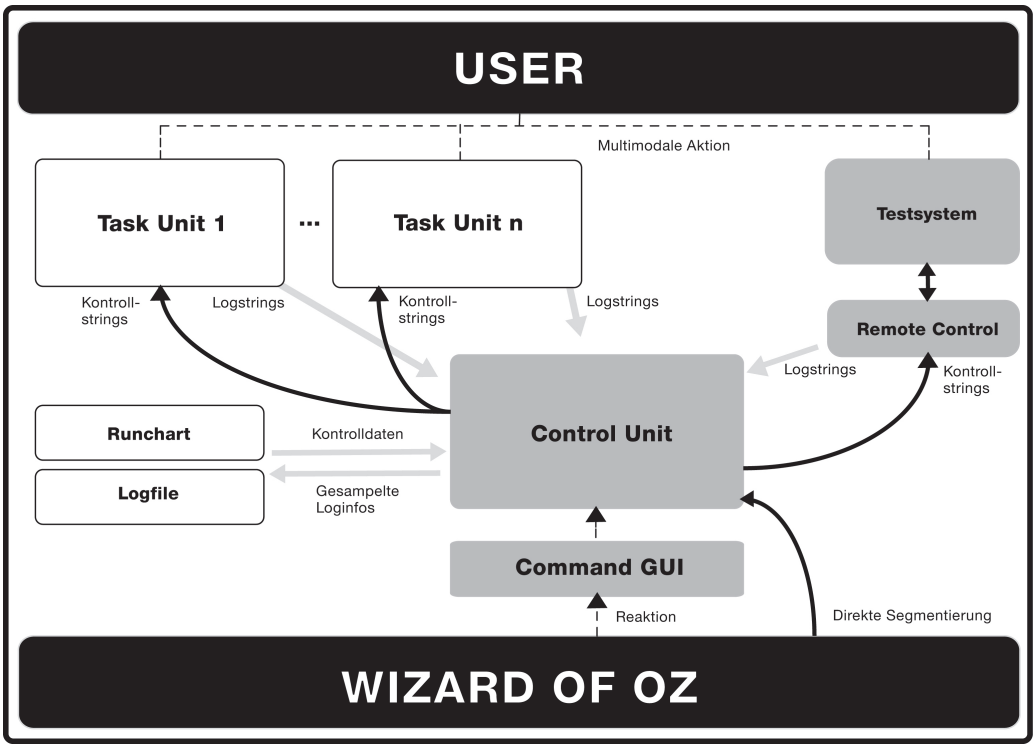


Abb. 3.7: Struktureller Systemüberblick für das *UsaWiz*-Konzept zur semi-automatischen Durchführung von multimodalen Benutzerstudien nach dem Wizard-of-Oz Prinzip

Steuereinheit (*CU, Control-Unit*) die angeschlossenen Komponenten (*Task Units*) kontrolliert. Dazu gehört beispielsweise ein Modul zur Audio-Ausgabe der im Vorfeld aufgezeichneten Anweisungen (*WizSpeech*) oder auch die Fahraufgabe in der automobilen Testumgebung. Gleichzeitig werden sämtliche anfallenden Daten mit einem millisekundengenauen Zeitstempel versehen und in ein zentrales *LogFile* geschrieben. Aus dieser Datei können die Versuchsdaten später mittels eines speziellen Tools ausgelesen und automatisch ausgewertet werden.

Die Basis für die semi-automatische Durchführung der Versuche ist die abstrakte Beschreibung des Versuchsablaufs in einem eigens dafür entwickelten Formalismus (*XCL = eXperiment Control Language*). Dieser ermöglicht eine blockorientierte Definition der Aufgaben im Test (*Tasks*), eine dedizierte Ansteuerung der einzelnen Systemkomponenten sowie die Synchronisation von verschiedenen Audio- und Videoquellen. Darüber hinaus werden zeitgesteuerte Versuchsunterbrechungen, fundametale Schleifen- und Abfragestrukturen und spezielle Warnhinweise für den Versuchsleiter unterstützt. Während des iterativen Redesigns eines Versuchs können durch einfache Editierung der Runchart etwaige Änderungen und Erweiterungen am generellen Ablauf ohne größeren Aufwand durchgeführt werden. Spezifische Details zur XCL-Notierung werden in [82] ausführlich erklärt.

Die Control-Unit interpretiert die Runchart-Datei und arbeitet sie taskweise ab, wobei an definierten Stellen immer wieder auf eine Bestätigung vom Versuchsleiter gewartet wird. Zusätzlich besteht für den Wizard die Möglichkeit, die aktuelle Ansagen mehrfach abspielen zu lassen, spezifische Benutzeraufgaben zu wiederholen oder zu überspringen und in der Liste der definierten Benutzeraufgaben an eine spezielle Position zu wechseln. An geeigneten Stellen können Ver-

zweigungen in die Runchart eingebaut werden, die vom Versuchsleiter durch einfaches Klicken eines Auswahlbuttons initiiert werden. Auf diese Weise ist es möglich, eine kontextsensitive, vorab definierte Reaktion auf verschiedene prototypische Benutzereingaben zu realisieren. Der Versuchsleiter muss die eigentliche Systemfuntionalität nicht mehr selbstständig auslösen, sondern kann bei intensiver Kenntnis der Ablauflogik ein passendes Systemfeedback erzeugen.

Damit der Software-Prototyp im Rahmen von simulierten Benutzereingaben extern angesteuert werden kann ist im UsaWiz-Konzept ein einfaches, buttonbasiertes Interface entworfen worden (*WizardGUI*), das auf Knopfdruck des Versuchsleiters das entsprechende, vom Benutzer intendierte Kommando über einen TCP/IP-Socket an das Testsystem sendet. Das UsaWiz-Konzept hat sich bereits bei der Durchführung von zahlreichen Benutzerstudien am Lehrstuhl in unterschiedlichen Domänen ausserordentlich bewährt (beispielsweise in [81, 15, 14, 11, 135]), da es nachweislich den Versuchsleiter bei der Steuerung der verschiedenen Teilfunktionalitäten im Testaufbau entlastet. Durch die daraus folgende Reduktion der kognitiven Belastung werden fehlerhafte Reaktionen des Wizards weitestgehend minimiert. Der Versuchslieter kann sich auch bereits während des Versuchsablauf intensiv mit der Beobachtung der Probanden beschäftigen und sich ggf. zusätzliche Notizen machen.

KAPITEL 4

Ergebnisse der Benutzerstudien

In diesem Kapitel werden die Ergebnisse der Benutzerstudien zur Evaluierung multimodaler Interaktionsmuster für die beiden ausgewählten Beispieldomänen Desktop Virtual-Reality Browser (DVA) und automotive Infotainmentsysteme (AIA) umfassend beschrieben. Die Basis für die späteren Auswertungen bilden einige allgemeine Anmerkungen über die spezielle Struktur des Datenmaterials und über verschiedene vorverarbeitende Verfahren, die zur Extraktion der qualitativen und quantitativen Ergebnisse angewendet werden. Der Hauptteil des Kapitels besteht aus einer domänenspezifischen Diskussion der Versuchsergebnisse, wobei sowohl auf die strukturellen zeitlich-semantischen Relationen als auch auf die subjektiven Erfahrungen der Benutzer im Test eingegangen wird. Basierend auf den gewonnenen Erkenntnissen werden die Test-Domänen anschließend im globalen Kontext verglichen.

4.1 Datenmaterial

Im Laufe der Benutzerstudien ist vielfältiges, unterschiedlich strukturiertes Datenmaterial entstanden. Die Hauptquellen für die quantitative Analyse der Versuchsdaten bilden die Videoaufnahmen und die Versuchsprotokolle, die im Rahmen der UsaWiz-Umgebung (siehe Abschnitt 3.5) automatisch für die einzelnen Versuchsteilnehmer erstellt worden sind. Ein wichtiger Schritt im Vorfeld der Datenauswertung besteht darin, in einer manuellen Nachbearbeitungsphase diese Protokolldaten in Relation zu den audiovisuellen Aufnahmen der Spracheingaben, Handgesten und Kopfgesten zu setzen.

Die aufbereiteten Daten werden dann automatisch block- und domänenspezifisch im Hinblick auf die verwendeten Kommandostrukturen, die auftretenden Modalitätenverteilungen und Kombinationen, die Übergänge zwischen den einzelnen Modalitäten und die beobachteten Zeitrelationen untersucht. Generell wird bei der Evaluierung zwischen unimodalen und multimodalen Benutzerinteraktionen unterschieden. Das Hauptinteresse der gesamten Auswertung gilt vor allem einer detaillierten Analyse der Benutzereingaben, die sich strukturell aus mehreren Informationsanteilen zusammensetzen. Im Fall einer multimodalen Systeminteraktion kommt der Bestimmung der einzelnen Eingabezeiten eine besondere Rolle zu. Durch den Vergleich der jeweiligen Start- und Endzeiten können die zeitliche Abfolge und die Überlagerungsart der beteiligten Modalitäten ermittelt werden.

Die Grundlage für die qualitative Evaluierung der Versuchsdaten bilden die verschiedenen Fragebögen, die während und nach den Versuchen von den Testpersonen ausfüllt wurden. Zusätzlich werden in die Auswertung die persönlichen Notizen des Versuchsleiters mit einbezogen, die sich dieser im Laufe der Versuche gemacht hat, um einerseits interessante Punkte für eine dedizierte Offline-Analyse zu markieren und andererseits direkt hervorstechende prototypische Verhaltensweisen schon zur Laufzeit klassifizieren zu können. Als dritte Datenquelle für die qualitative Analyse werden die Anmerkungen aufgenommen, die von den Probanden in der abschließenden Diskussion mit dem Versuchsleiter geäußert und auf Videomaterial festgehalten wurden. Speziell diese Daten geben wertvolle Hinweise zur Identifikation potenzieller Designschwächen des aktuellen Interfaces und zu entsprechenden Verbesserungen.

Durch das Ausfüllen des Anfangsfragebogens werden die einzelnen Benutzer bezüglich ihrer Erfahrung im Umgang mit technischen Systemen in drei disjunkte Klassen eingeteilt: Anfänger (**ANF**), Normalnutzer (**NOR**) und Experten (**EXP**). Der Großteil der Versuchsdaten wird im Laufe der Auswertung sowohl gruppenspezifisch, als auch allgemein für alle beteiligten Versuchspersonen diskutiert. Im Folgenden wird der Begriff Versuchsperson auch mit *VP*, die Mehrzahl mit *VPen* abgekürzt. Als übergreifende Bezeichnung für die Gruppe mit sämtlichen Versuchsteilnehmern wird zusätzlich die Abkürzung **AVP** eingeführt.

Formal lassen sich die einzelnen Benutzerklassen durch entsprechende Mengen beschreiben. Sei im Folgenden $\mathcal{U} = \{ANF, NOR, EXP, AVP\}$ die abstrakte Darstellung der möglichen Kategorien und U mit $U \in \mathcal{U}$ die Bezeichnung für eine spezifische Gruppe. Gibt man mit N_U^{VP} allgemein die Anzahl der VPen in dieser Gruppe an, dann enthält

$$\mathcal{B}_U = \{B_{U,1}, B_{U,2}, ..., B_{U,N_U^{VP}}\} \quad \text{mit} \quad N_U^{VP} \in \mathbb{N} \tag{4.1}$$

eine Auflistung der entsprechenden Benutzer. Über den Index x mit $x \in \{1..N_U^{VP}\}$ existiert eine einfache Möglichkeit, die einzelnen VPen $b \in \mathcal{B}_U$ aus der Klasse U eindeutig zu referenzieren. Daher wird in der folgenden Diskussion zur Angabe eines Benutzers ausschließlich der entsprechende Index verwendet. Für die Gesamtheit aller Benutzer im Test werden zudem mit $\mathcal{B} := \mathcal{B}_{AVP}$ und analog mit $N_{VP} := N_{AVP}^{VP}$ zentral abkürzende Schreibweisen definiert.

4.2 Auswertungsverfahren

4.2.1 Versuchsprotokolle

In den automatisch angelegten Protokolldateien sind sämtliche Interaktionen der Benutzer mit dem Testinterface, die Eingriffe des Versuchsleiters und die internen Nachrichten zwischen den anderen Systemkomponenten jeweils mit einem millisekundengenauen Zeitstempel festgehalten. Unabhängig von der spezifischen Funktion und Herkunft werden diese Ereignisse im Folgenden übergreifend als *Events* bezeichnet und zentral dem Symbol E zugeordnet. In dieser Arbeit werden allgemein die Begriffe *Ereignis* und *Event* synonym verwendet.

Durch das sequentielle Auslesen der Protokolldatei kann jedem Event ein eindeutiger *Event-Index* zugeordnet werden. Bezeichnet man mit $t(E)$ den Zeitstempel eines Events E, mit $x \in \{1..N_{VP}\}$ einen beliebigen Benutzer und mit N_x^{PE} die Anzahl sämtlicher, in der Log-Datei protokollierten Ereignisse einer spezifischen Sitzung mit diesem Benutzer, so lässt sich der

komplette personenspezifische Versuchsablauf ($\mathcal{E}_x$) formal als Menge von Ereignissen darstellen, in der die einzelnen Elemente einer zeitlichen Ordnungsrelation unterliegen:

$$\mathcal{E}_x = \{E_{x,1}, E_{x,2}, ..., E_{x,N_x^{PE}}\} \quad \text{mit} \;\; t(E_{x,i}) < t(E_{x,j}) \;\; \forall\, i < j \quad i,j \in \{1..N_x^{PE}\} \tag{4.2}$$

Die Menge aller Ereignisse von sämtlichen Benutzern ($\mathcal{E}$) ergibt sich aus der Vereinigung der einzelnen $\mathcal{E}_x$. In dieser Darstellung gilt die strikte Ordnungsrelation nicht, da die absoluten Zeitstempel von unterschiedlichen Benutzern aus verschiedenen Versuchen nicht direkt verglichen werden können. Die Gesamtanzahl aller protokollierten Ereignisse (N^{PE}) erhält man durch Summierung der einzelnen N_x^{PT}.

$$\mathcal{E} = \bigcup_{x=1}^{N_{VP}} \mathcal{E}_x \qquad \text{und} \qquad N^{PE} = \sum_{x=1}^{N_{VP}} N_x^{PE} \tag{4.3}$$

Eine wichtige Aufgabe bei der Analyse der einzelnen Protokolldateien besteht darin, die personenbezogenen Events $E_i \in \mathcal{E}_x$ mit $i \in \{1..N_x^{PT}\}$ verschiedenen Gruppen, sog. *Event-Klassen* zuzuordnen. Um eine spezifische Ereignisklasse notationstechnisch eindeutig beschreiben zu können, wird jeweils ein qualifizierender Bezeichner $´EE´$ eingeführt und als hochgestellter Index mit dem Eventsymbol verknüpft (E^{EE}). Zusätzliche Parameter, wie beispielsweise Laufindizes zur Identifikation einzelner Benutzer oder weitere, näher qualifizierende Bezeichner (NN) werden, durch Kommata getrennt, als tiefgestellter Index mit dem Eventsymbol verknüpft ($E_{x,NN}^{EE}$).

Im Folgenden werden die unterschiedlichen Ereignistypen im jeweiligen Kontext Schritt für Schritt erklärt und die dazugehörigen Bezeichner eingeführt. Als zentrale Referenz dienen die Tabellen A.1, A.2 und A.3 im Anhang. Die funktionale Zusammensetzung der einzelnen Events und die strukturellen Abhängigkeiten von den anderen Ereignissen können vollständig durch eine kontextfreie Grammatik beschrieben werden (CFG). Für die formale Notation der grammatikalischen Regeln wird die erweiterte Backus-Naur-Form (EBNF)[138] verwendet. Anhang B.1 enthält eine kurze Einführung in die entsprechende Begriffsterminologie.

Grundsätzlich existieren auf der höchsten Beschreibungsebene direkte Benutzerereignisse (E^{TA}), Wizard-Ereignisse (E^{WIZ}), Systemereignisse (E^{SYS}) und externe Ereignisse (E^{EXT}). In der formalen Notation lässt sich ein abstraktes Ereignis E wie folgt spezifizieren:

$$< E > \;::=\; < E^{TA} > \;|\; < E^{WIZ} > \;|\; < E^{SYS} > \;|\; < E^{EXT} > \tag{4.4}$$

Bezogen auf die möglichen Interaktionen mit dem Testinterface muss systemtechnisch zwischen E^{TA} und E^{WIZ} unterschieden werden. Ein direktes Benutzerereignis bezieht sich ausschließlich auf eine taktile Interaktion der VP mittels Touchscreen (E^T) oder Tastenkonsole (E^A). Diese Eingabe erfolgt unmittelbar am Versuchsrechner und löst direkt eine entsprechende Systemaktion aus. Die Abkürung E^{TA} ergibt sich aus den beteiligten Modalitäten:

$$< E^{TA} > \;::=\; E^T \mid E^A \tag{4.5}$$

Demgegenüber versteht man unter einem Wizard-Ereignis eine externe, vom Versuchsleiter initiierte Funktion des Testsystems. Bei einem solchen Ereignis kann es sich zum einen um eine indirekte Interaktion des Benutzers mittels Sprache (E^S), Handgestik (E^H) oder Kopfgestik

(E^K) handeln, die vom Wizard im Kontrollraum interpretiert und erst dann per Fernsteuerung in eine Systemaktion umgesetzt wird. Zum anderen kann ein Wizard-Ereignis auch einen zusätzlichen Kontrolleingriff (E^{WC}) repräsentieren, dem keine explizite Benutzereingabe zugrunde liegt. Neben allgemeinen Anweisungen zur Steuerung des Versuchsablaufs sind damit vor allem Hilfestellungen des Versuchsleiters gemeint, um beispielsweise im DVA-Szenario den virtuellen Avatar nach fehlerhaften Navigationsanweisungen wieder richtig auszurichten oder im AIA-Szenario eine vordefinierte Lautstärke einzustellen.

$$< E^{WIZ} > \ ::= \ E^H \mid E^K \mid E^S \mid < E^{WC} > \tag{4.6}$$

Ein Benutzerereignis (E^{BE}) setzt sich allgemein aus einer Verkettung von direkten, taktilen Eingaben mittels Touchscreen und Tastenblock und indirekten, semantisch höherwertigen Eingaben mittels Handgestik, Kopfgestik und Sprache zusammen, wobei die einzelnen Modalitäten von den Benutzern frei kombiniert werden können. Da in der abstrakten EBNF-Notation keine Informationen über die temporalen Kontexte der einzelnen Eingaben modelliert werden, ist die Reihenfolge der Symbole in dieser Darstellung nicht von Bedeutung.

$$\begin{aligned} < E^{BE} > \ &::= \ E^T \ \{E^A\} \ \{E^H\} \ \{E^K\} \ \{E^S\} \\ &::= \ \{E^T\} \ E^A \ \{E^H\} \ \{E^K\} \ \{E^S\} \\ &::= \ \{E^T\} \ \{E^A\} \ E^H \ \{E^K\} \ \{E^S\} \\ &::= \ \{E^T\} \ \{E^A\} \ \{E^H\} \ E^K \ \{E^S\} \\ &::= \ \{E^T\} \ \{E^A\} \ \{E^H\} \ \{E^K\} \ E^S \end{aligned} \tag{4.7}$$

In dieser Regel kommt zum ersten Mal die erweiterte Form der BNF zum Einsatz. Die geschweiften Klammern bedeuten, dass das eingeschlossene Symbol beliebig oft wiederholt oder auch komplett weggelassen werden kann. Zumindest über eine Modalität muss aber eine konkrete Interaktion erfolgen. Die genaue Struktur einer Eingabe E^{BE} ist zentraler Gegenstand der Untersuchungen und wird daher später im Text in weiteren Definitionen noch stärker präzisiert.

Neben den taktilen Interaktionen wird analog zu Gleichung 4.5 auch für die semantisch höherwertigen Eingaben ein eigenständiges Ereignis-Symbol E^{HKS} eingeführt, das sich aus den Abkürzungen der einzelnen Modalitäten zusammensetzt.

$$< E^{HKS} > \ ::= \ E^H \mid E^K \mid E^S \tag{4.8}$$

Im Rahmen dieser Studie beschränkt sich die Analyse der Benutzerereignisse ausschließlich auf konkrete, intentionale Eingaben. Zusätzliche Informationen, beispielsweise über den emotionalen Zustand des Benutzers oder den aktuellen Status der Applikation, aber auch von der Versuchsperson unbeabsichtigt geäußerte, nicht primär auf die Kommunikation mit dem Testsystem ausgerichtete Artikulationen werden nicht mit in die Auswertung einbezogen. Die Diskussion der Interaktionen im globalen Kontext wird umfassend in anderen, speziell dafür konzipierten Benutzerstudien beschrieben ([175]).

Als Systemereignisse (E^{SYS}) bezeichnet man Rückmeldungen des Testsystem über den aktuellen Zustand, laufende Prozesse oder durchgeführte Aktionen. Alle anderen Ereignisse werden externen Modulen zugeordnet (E^{EXT}). In diese letzte Klasse fallen beispielsweise sämtliche automatisch initiierten Steuerungsfunktionen für die einzelnen Task-Units der UsaWiz-Umgebung.

Da jede Systemkomponente über einen eindeutigen Bezeichner verfügt und dieser zusammen mit den einzelnen Ereignissen in der Protokolldatei abgelegt wird, kann die Zuordnung der Events über einen einfachen Parser-Mechanismus erfolgen. Die zugrundeliegende Kommunikation zwischen den einzelnen Systemkomponenten wird ausführlich im Rahmen der architekturalen Beschreibung des Gesamtsystems behandelt.

Im Vorfeld der weiteren Untersuchungen wird aus den Protokolldateien jeweils pro Versuchsperson x mit $x \in \{1..N_{VP}\}$ eine Reihe von charakteristischen quantitativen Daten extrahiert. Dabei unterscheidet man mit ansteigendem Detaillierungsgrad zwischen globalen, blockspezifischen, aufgabenspezifischen und schließlich interaktionsspezifischen Werten. Als wichtigster Wert wird in dieser Phase beispielsweise die Anzahl der Benutzerereignisse (N_x^{BE}) bestimmt, die als Grundlage zur Berechnung verschiedener Häufigkeitswerte dient. Darüber hinaus werden die Anzahl der kontrollierenden Wizardeingriffen, die Anzahl der Systemmeldungen und die Anzahl und spezifische Bedeutung der externen Events ermittelt. Zusätzlich wird festgestellt, wie oft der Versuchsablauf durch den Wizard unterbrochen wurde und wie oft einzelne Versuchsansagen oder komplette Aufgabenblöcke wiederholt wurden. Im Rahmen der Untersuchungen in der Automobilumgebung werden auch die Fahrleistungen der Testpersonen quantitativ erfasst und zu den anderen Ergebnissen in Beziehung gesetzt [99].

4.2.2 Häufigkeitsanalysen

Ein wichtiges Instrument bei der Auswertung sind Häufigkeitsanalysen der individuellen Events. Sei E^{EE} die Beschreibung für eine spezielles, in diesem Fall rein fiktives Ereignis ´EE´, dann gibt N_x^{EE} die absolute Häufigkeit von E^{EE} für eine Versuchsperson x mit $x \in \{1..N_{VP}\}$ an. Wesentlich aussagekräftiger als der Absolutwert N_x^{EE} ist oftmals der personenspezifische Mittelwert $\bar{N}_x^{EE}$ von N_x^{EE} im Verhältnis zu der Anzahl sämtlicher Benutzerinteraktionen (N_x^{BE}). Alternativ kann als tiefgestellter Index auch der Bezeichner für eine spezielle Benutzergruppe U mit $U \in \mathcal{U}$ verwendet werden, um mit N_U^{EE} die absolute Häufigkeit des Ereignisses E^{EE} und mit $\bar{N}_U^{EE}$ den gruppenspezifischen Durchschnittswert anzugeben:

$$\bar{N}_U^{EE} = \frac{1}{N_U^{VP}} \sum_{x=1}^{N_U^{VP}} N_x^{EE} \tag{4.9}$$

Für viele interessante Auswertungsdaten müssen relative Häufigkeiten berechnet werden, die sich aus dem Quotienten von zwei direkt gemessenen Häufigkeiten ergeben. Sei N^{REE} beispielsweise die relative Häufigkeit von Event E^{EE1} mit Bezug zu der Häufigkeit von Ereignis E^{EE2}. Um die Darstellung etwas zu vereinfachen, wird im Folgenden der gruppenspezische Index U nicht explizit mitgeführt. Zusätzlich wird allgemein davon ausgegangen , dass bereits durch die Struktur der Daten die Bildung einer relativen Häufigkeit zulässig ist, d.h. $N^{EE2} \geq N^{EE1}$. Zur Berechnung von N^{REE} existieren prinzipiell zwei unterschiedliche Möglichkeiten:

1. Die einzelnen Werte N^{EE1} und N^{EE2} werden getrennt für jede VP x mit $x \in \{1..N_{VP}\}$ aufsummiert und aus dem Quotienten dieser absoluten Häufigkeiten wird der relative Mittelwert $\bar{N}^{REE\_1}$ für die gesamte Gruppe gebildet:

$$\bar{N}^{REE\_1} = \frac{\sum_{x=1}^{N_{VP}} N_x^{EE1}}{\sum_{x=1}^{N_{VP}} N_x^{EE2}} \tag{4.10}$$

2. Zuerst werden die relativen Häufigkeiten $\bar{N}_x^{REE}$ für jede VP x mit $x \in \{1..N_{VP}\}$ berechnet und anschließend aus der Summe der gemittelten Häufigkeiten der relative Mittelwert N^{REE_2} für die gesamte Gruppe ermittelt:

$$\bar{N}^{REE\_2} = \frac{1}{N_{VP}} \sum_{x=1}^{N_{VP}} \frac{N_x^{EE1}}{N_x^{EE2}} \tag{4.11}$$

Im Rahmen der Auswertungen in dieser Arbeit wird für die Berechnungen der relativen Häufigkeitswerte ausschließlich der zweite Weg gewählt ($\bar{N}^{REE} := \bar{N}^{REE\_2}$). Auf diese Weise kann einer signifikanten Beeinflussung des Gesamtergebnisses durch vereinzelt auftretende Extremwerte, die bei der Berechnung über den zentralen Summenquotienten in der ersten Möglichkeit entstehen können, effektiv entgegengewirkt werden. Demgegenüber wird in der zweiten Möglichkeit ein zweistufiger Mittelwertbildungsprozess durchlaufen. Bei den Berechnungsprozessen muss beachtet werden, dass die beiden Ergebnisse nicht notwendigerweise identisch sind ($\bar{N}^{REE\_1} \neq \bar{N}^{REE\_2}$). Daher ist für eine sinnvolle Interpretation der ausgewerteten Daten die Kenntniss der gewählten Berechnungsart von essentieller Bedeutung.

4.2.3 Post-Klassifikation der Wizard-Events

Bei der automatischen Auswertung der Interaktionen mit dem Testsystem steht man vor einem zentralen Problem. Auf der Basis der lexikalischen Form der Informationen in der Protokolldatei kann nicht eindeutig entschieden werden, ob es sich bei einem spezifischen Wizard-Event E_i^{WIZ} um eine indirekte Benutzereingabe oder um einen zusätzlichen Steuerungs- und Kontrolleingriff handelt. Da die vom Versuchsleiter interpretierten Eingaben in einer abstrakten Syntax kodiert sind, ist darüber hinaus aus der Datei nicht ersichtlich, welche Einzelmodalitäten in welchem Umfang einer simulierten Aktion zugrunde liegen. Zur Lösung dieses Referenzproblems müssen die Daten der Protokolldatei und die Videoaufnahmen zuerst auf einer gemeinsamen Zeitbasis synchronisiert und dann manuell annotiert werden.

Synchronisation der Datenquellen

Die zeitliche Synchronisation der Protokolldaten und der Videoaufnahmen lässt sich relativ einfach realisieren. Jeweils zu Beginn eines Versuchsblocks wird von der RunChart ein optisches Synchronisationsereignis ausgelöst, bei dem die Kontrollbuttons in dem Test-Interface für drei Sekunden ausgeblendet werden. In der Protokolldatei ist dieses Event mit einem exakten Zeitstempel im Millisekundenbereich ablesbar. Auf dem Videoband kann das entprechende Einzelbild, bei dem dieses Ereignis eintritt, ebenfalls ohne großen Aufwand identifiziert werden. Die Übereinstimmung dieser beiden Zeiten bildet die Berechnungsbasis für die Auswertung von sämtlichen zeitbezogenen Ergebnissen in den Benutzerstudien dieser Arbeit. Um etwaige Ungenauigkeiten bei der Bestimmung der Synchronisationszeiten zu minimieren, wird aus den fünf möglichen Synchronisationspunkten jeweils zu Beginn der einzelnen Blöcke ein Mittelwert gebildet. Voraussetzung dafür ist, dass das Videoband den ganzen Versuch mitläuft und nicht zwischen den einzelnen Versuchsblöcken angehalten wird.

Der Video-Stream besteht aus einzelnen Frames, wobei die verwendeten Aufnahmegeräte über eine zeitliche Auflösung von 25 Einzelbildern pro Sekunde verfügen. Die aus dem Videomaterial extrahierten Zeitangaben haben demnach eine maximale Genauigkeit von 0,04 (1/25)

Sekunden. Bezeichnet man mit LT die Zeitangabe aus der Protokolldatei, angegeben in Millisekunden [ms] und mit VT die Zeit auf dem Videoband, aufgeschlüsselt in Stunden, Minuten, Sekunden und Frames [HH:MM:SS:FF], so lassen sich unter Ausnutzung der abgelesenen Synchronisationsbeziehung $LT_{sync} = VT_{snyc}$ die unterschiedlichen Zeitreferenzen zu jedem beliebigen Zeitpunkt t ineinander umrechnen. Beide Zeitangaben werden in absoluten Werten relativ zum Start des Protokollvorgangs bzw. der Videoaufnahme angegeben. Wegen der einfacheren Form wird für die quantitative Auswertung der temporalen Versuchsdaten im Folgenden die Zeitangabe aus der Protokolldatei als zentrale Referenzgröße verwendet.

Manuelle Nachbearbeitung

Unter Zuhilfenahme der Videodaten werden die einzelnen Wizard-Events E^{WIZ}, die einer konkreten Benutzereingabe entsprechen, nachträglich markiert und modalitätenspezifischen Benutzer-Events E^{BE} zugeordnet. Dieses Vorgehen wird auch als *SHM-Klassifikation* bezeichnet, da die semantisch höherwertigen Benutzereingaben mit Handgestik, Kopfgestik und Sprache entsprechend ihrem Bedeutungsinhalt aufgeschlüsselt werden. Da die taktilen Eingaben direkt vom System umgesetzt werden und mit einem eindeutigen Bezeichner in der Protokolldatei abgelegt sind, müssen sie nicht noch einmal gesondert klassifiziert werden.

Kann einem protokollierten Benutzerereignis E_i^{BE} eindeutig eine sowohl zeitlich als auch semantisch isolierte Interaktion mittels einer der zur Verfügung stehenden Eingabemodaliäten $m \in \mathcal{M}_{IN}$ zugeordnet werden, die im Hinblick auf das Funktionsvokabular des Testsystem syntaktisch vollständig und korrekt ist, so bezeichnet man E_i^{BE} als eine *unimodale Benutzerinteraktion*. Für dieses Ereignis wird das Symbol E^{UM} eingeführt. Die Interaktion resultiert direkt aus der Eingabe mit der entsprechenden Modalität:

$$< E^{UM} > \; ::= \; E^T \mid E^A \mid E^H \mid E^K \mid E^S \tag{4.12}$$

Falls umgekehrt mehrere Benutzereingaben mit dem protokollierten Ereignis E_i^{BE} assoziiert werden können, so handelt es sich um eine *multimodale Benutzerinteraktion* (E^{MM}). Eine multimodale Eingabe wird im Rahmen dieser Arbeit allgemein auch als *kombinierte* Eingabe bezeichnet. Im Sinne einer Benutzerinteraktion können die Begriffe *multimodal* und *kombiniert* daher vollständig synonym benutzt werden. Diese Terminologie ist im Forschungsumfeld nicht einheitlich festgelegt. Es existieren auch Arbeiten (z.B. [87, 34]), bei denen eine Kombination mehrerer Modalitäten automatisch eine ausschließlich synergistische Informationszusammensetzung impliziert.

Allgemein können Benutzereingaben nach der Anzahl der beteiligten Modalitäten in verschiedene Kategorien eingeteilt, wobei man grundsätzlich unimodale und multimodale Interaktionen unterscheidet. In Ergänzung zu Gleichung 4.7 lässt sich die folgende Regel aufstellen:

$$< E^{BE} > \; ::= \; < E^{UM} > \; \mid \; < E^{MM} > \tag{4.13}$$

Multimodale Interaktionen können im Hinblick auf die Anzahl in weitere Klassen unterteilt werden. Bei zwei Modalitäten spricht man von einer *bimodalen* Interaktion (E^{BMM}) und bei drei Modalitäten entsprechend von einer *trimodalen* Interaktion (E^{TMM}). Falls zusammenhängende Informationen über mehr als drei Modalitäten kommuniziert werden, so bezeichnet man

die Eingabe als *komplex multimodal* (E^{KMM}). Formal setzt sich beispielsweise E^{BMM} aus der Verkettung von zwei unimodalen Interaktionen zusammen.

$$< E^{MM} > \ ::= < E^{BMM} > \ | \ < E^{TMM} > \ | \ < E^{KMM} > \tag{4.14}$$

$$< E^{BMM} > \ ::= < E^{UM} > < E^{UM} > \tag{4.15}$$

$$< E^{TMM} > \ ::= < E^{UM} > < E^{UM} > < E^{UM} > \tag{4.16}$$

$$< E^{KMM} > \ ::= < E^{UM} > < E^{UM} > < E^{UM} > \ \{< E^{UM} >\} \tag{4.17}$$

Neben der Anzahl der beteiligten Modalitäten kann eine multimodale Benutzerinteraktion auch anhand der funktionalen Zusammenhänge der einzelnen Informationsanteile klassifiziert werden. Bezogen auf das bereits in Abschnitt 2.3.1 vorgestellte Schema bezeichnet man multimodale Benutzerinteraktion als *redundant* (E^{RED}), falls Elemente mit gleichem Bedeutungsinhalt in mehreren unterschiedlichen Kommunkationskanälen enthalten sind. Analog nennt man E^{MM} *rivalisierend* (E^{RIV}), falls einzelne Informationsanteile in Bezug auf die Semantik in direkter Konkurrenz zueinder stehen. Schließlich werden multimodale Benutzerinteraktionen, bei denen sich die Informationsanteile in den einzelnen Kanälen gegenseitig ergänzen als *komplementär* bezeichnet (E^{CMP}). Diese letzte Klasse bezieht sich auf eine synergistische Nutzung der Modalitäten. In Ergänzung zu Gleichung 4.14 ergibt sich damit folgende Regel:

$$< E^{MM} > ::= < E^{RED} > \ | \ < E^{RIV} > \ | \ < E^{CMP} > \tag{4.18}$$

Komplementäre Benutzerinteraktionen sind oft mit den anderen Ereignisklassen verknüpft. Um die speziellen Relationen auch formal beschreiben zu können, werden daher drei zusätzliche Ereignisklassen eingeführt. Man bezeichnet ein multimodales Ereignis E^{CMP} genauer als *redundant-komplementär* (E^{REC}) bzw. *rivalisierend-komplementär* (E^{RIC}), falls Teilinformationen in gleichen funktionellen Gruppen redundant bzw. rivalisierend in unterschiedlichen Modalitäten vorhanden sind. Darüber hinaus bedeutet *harte Komplementarität* (E^{HAC}), dass die einzelnen Informationen für unterschiedliche funktionelle Gruppen von verschiedenen Eingabemodalitäten stammen, d.h., in diesem Fall liegt explizit werder Redundanz noch Konkurrenz in den verschiedenen Teilen der kompletten Benutzerinteraktion vor.

$$< E^{CMP} > ::= < E^{REC} > \ | \ < E^{RIC} > \ | \ < E^{HAC} > \tag{4.19}$$

Der nachträgliche Versuchsdurchlauf zur Klassifikation der SHM Eingaben wird durch den Einsatz eines speziell entwickelten Software-Tools extrem vereinfacht. Bezieht man den Zeitaufwand zur Synchronisation der Datenquellen mit ein, so ist mit etwas Training eine qualitativ hochwertige Nachbearbeitung in zweifacher Echtzeit erreichbar, d.h. für eine Stunde Videomaterial müssen mindestens zwei Stunden manuelle Analyse aufgewendet werden, um die Wizard-Events den ursprünglichen sprach- und gestenbasierten Benutzereingaben zuzuordnen. Ohne die SHM-Klassifikation ist eine sinnvolle Interpretation der Versuchsergebnisse im multimodalen Kontext nicht möglich.

4.2.4 Temporale Annotationen

Eine weitere Aufgabe im Zusammenhang mit der Nachbearbeitung besteht darin, sowohl die Eingabezeiten mit den einzelnen Modalitäten, als auch die zeitlichen Überlagerungen der kombinierten Benutzerinteraktionen zu bestimmen. Die Auswertungen im Rahmen dieser Benutzerstudie konzentrieren sich dabei primär auf die semantisch höherwertigen Modalitäten Sprache,

Hand- und Kopfgestik, obwohl auch für die Interaktionen mit den taktilen Eingabegeräten signifikante Bediendauern anfallen. Eine detaillierte Analyse speziell von taktilen Interaktionen ist in anderen Arbeiten am Institut mit leicht veränderten Versuchsaufbauten durchgeführt worden ([50, 175]).

Das bereits mehrfach angesprochene Bearbeitungstool verfügt zusätzlich über die Möglichkeit, mit jeder indirekten Wizardeingabe E_i^{WIZ} eine modalitätenspezifische Start- und Endzeit zu assoziieren und zentral mit dem Datensatz abzulegen. Diesen Vorgang bezeichnet man als *temporales SHM-Labeling*. Für die Bestimmung der sprachbasierten Interaktionen wird sowohl das akustische als auch das visuelle Material ausgewertet. Bezogen auf gestenbasierte Eingabeformen werden die Änderungen von einer definierten Ausgangslage über die komplette Durchführung der Gesten bis zum erneuten Erreichen des Ausgangszustands unmittelbar aus dem Videomaterial bestimmt. Ein analoges Vorgehen wäre auch für die Auswertung der taktilen Bediendauern möglich gewesen. Da die taktilen Interaktionen jedoch nicht im Vordergrund der Untersuchung standen, wurde aus Zeitgründen darauf verzichtet. Stattdessen werden in der generellen Zeitenauswertung die Interaktionen mit dem Touchscreen und der Tastatur a priori mit infinitesimaler Bedienzeit angesetzt.

Im direkten Vergleich zu der SHM-Klassifikation ist das Labeling extrem zeitaufwenig. Beispielsweise kann bei bimodalen Interaktionen, bezogen auf die Länge der einzelnen Interaktionen, bestenfalls mit einem Faktor von 25-facher Echtzeit als Aufwand für die Bearbeitung gerechnet werden. Dies liegt unter anderem daran, dass sich das Auffinden der genauen Start- und Endzeiten mitunter sehr viel Sorgaflt erfordet, da das Videomaterial mehrfach Einzelbild für Einzelbild untertersucht werden muss. Dieser Analysevorgang ist zudem für jede der beteiligten Modalitäten nötig. Bei trimodalen Interaktionen erhöht sich der Aufwand entsprechend.

Im Rahmen dieser Benutzerstudie konnte daher nur ein repräsentativer Bruchteil aller SHM-Interaktionen zeitlich erfasst werden. Die Auswertung der Zeitrelationen konzentriert sich ausschließlich auf Block V in der dritten Phase der Versuche mit der freien multimodalen Interaktion. Eine weitere Reduktion des Aufwands besteht in der Selektion charakteristischer Versuchspersonen $\mathcal{B}_{ZVP} = \{B_1^Z, B_2^Z, ...B_{N_{ZVP}}^Z\}$, wobei die Anzahl der für die Zeitenanalyse ausgewählten VPen (N_{ZVP}) im Allgemeinen deutlich kleiner ist als die Gesamtanzahl aller Versuchsteilnehmer N_{VP}. In dieser Menge sollten die jeweiligen Benutzergruppen möglichst zu gleichen Teilen vertreten sein.

Einzelne Benutzer aus $\mathcal{B}_{ZVP}$ werden jeweils wieder über den Index referenziert. Pro Benutzer y mit $y \in \{1..N_{ZVP}\}$ wird dann in der nächsten Stufe eine limitierte Anzahl N_y^{ZBE} mit $N_y^{ZBE} \ll N_y^{BE}$ prototypischer Interaktionsmuster ausgewählt und manuell annotiert. Eine solche, zeitlich bearbeitete Benutzereingabe wird mit E^{ZBE} bezeichnet. Die Anzahl der personenspezifischen Ereignisse kann dabei variieren, d.h. im Allgemeinen gilt $N_{y1}^{ZBE} \neq N_{y2}^{ZBE}$ für alle Versuchspersonen $y_1 \neq y_2$ und $y_1, y_2 \in \{1..N_{ZVP}\}$. Als Resultat ergibt sich eine personenspezifische Auflistung verschiedener Eingabe-Events:

$$\mathcal{E}_y^{ZBE} = \{E_{1,y}^{ZBE}, E_{2,y}^{ZBE}, ..., E_{N_{ZBE,y}}^{ZBE}\} \quad \text{mit} \quad N_{ZBE,y} := N_y^{ZBE} \quad \text{und} \quad N_y^{ZBE} \ll N_y^{BE} \tag{4.20}$$

Die Menge aller zeitlich annotierten Benutzerereignisse ($\mathcal{E}^{ZBE}$) ergibt sich analog zu Gleichung 4.3 aus der Vereinigung der personenspezifischen Ergeignismengen $\mathcal{E}_y^{ZBE}$ und die Mächtigkeit dieser Menge (N^{ZBE}) entsprechend aus der Summe der einzelnen N_y^{ZBE}.

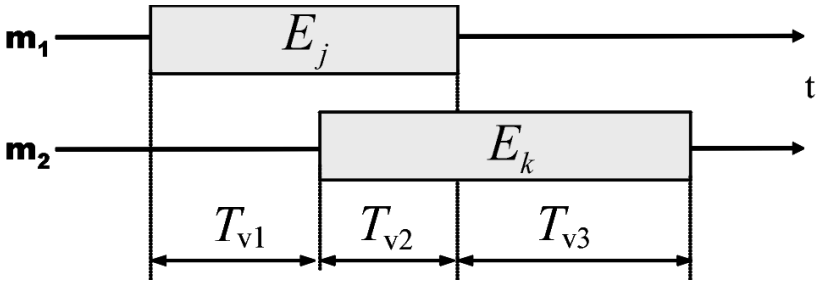


Abb. 4.1: Visualisierung einer zeitlichen Überlagerung zweier Benutzereingaben E_j und E_k mit den Modalitäten $m_1, m_2 \in \mathcal{M}_{IN}$ und daraus resultierenden Verzögerungszeiten T_{v1}, T_{v2}, T_{v3}

4.2.5 Multimodale Zeitschemata

Bei der Analyse der Zeitrelationen stehen zwei Charakteristika im Vordergrund: die zeitliche Reihenfolge und die Überlagerungsart der beteiligten Modalitäten. Im Rahmen der erklärenden Darstellung wird für jede Modalitäte m mit $m \in \mathcal{M}_{IN}$ jeweils eine kontinuierliche Zeitachse angenommen, auf der die einzelnen Benutzereingaben als Symbole aufgetragen werden. Die horizontale Ausdehnung eines Symbols auf diesen Achsen ist direkt proportional zu der realen Länge der Benutzerinteraktion. Im Sonderfall der taktilen Interaktionen fallen Anfangs- und Endzeitpunkt aufgrund der infinitesimal angenommenen Interaktionsdauer zusammen und werden daher nur als Punkte dargestellt. Die einzelnen Zeitachsen zur Repräsentation der modalitätenbezogenen Ereignisse sind auf einen globalen Zeitstempel synchronisiert. Die dazu benötigen Verfahren werden in der Architekturbeschreibung (Abschnitt 5.5.2) erklärt.

Sei $E_{x,i}^{BE}$ eine konkrete Eingabe des Benutzers $x \in \{1..N_{ZVP}\}$, der bezogen auf den Versuchsablauf ein eindeutiger Event-Index i mit i mit $i \in \{1..N_x^{PE}\}$ zugeordnet werden kann, dann gibt $t_s(E_{x,i}^{BE})$ die genaue Startzeit und $t_e(E_{x,i}^{BE})$ entsprechend die Endzeit der Interaktion $E_{x,i}^{BE}$ an. Der generelle Zeitstempel der Interaktion wird mit $t(E_{x,i}^{BE})$ bezeichnet und unter Vernachlässigung etwaiger Kanallaufzeiten in dieser Benutzerstudie direkt mit der Endzeit der Eingabe assoziiert ($t(E_{x,i}^{BE}) = t_e(E_{x,i}^{BE})$). Für die beiden Zeitpunkte gilt zudem mit $t_s(E_{x,i}^{BE}) < t_e(E_{x,i}^{BE})$ eine strenge Monotonieanforderung. Die Dauer der Interaktion ($T(E_{x,i}^{BE})$) ergibt sich unmittelbar aus der Differenz der beiden Zeitmarken t_e und t_s:

$$T(E_{x,i}^{BE}) = t_e(E_{x,i}^{BE}) - t_s(E_{x,i}^{BE}) \quad \forall\, i \in \{1..N_x^{ZBE}\} \tag{4.21}$$

Im Folgenden wird allgemein eine multimodale Benutzerinteraktion E_i^{MM} betrachtet, die sich strukturell aus zwei beliebigen Benutzereingaben zusammensetzt. Zur Vereinfachung der Darstellung wird der Index für die VP nicht mehr explizit mitgeführt. Seien $E_j^{m_1}$ und $E_k^{m_2}$ die Bezeichnungen für diese beiden Benutzerereignisse mit den Modalitäten $m_1 \neq m_2$ und $m_1, m_2 \in \mathcal{M}_{IN}$ und den Indizes $j, k \in \{1..N^{PE}\}$. Weiterhin kann o.B.d.A. angenommen werden, dass $E_j^{m_1}$ in der globalen Liste der Ereignisse vor $E_k^{m_2}$ gespeichert ist, d.h. mit $j < k$ verfügt die zweite Interaktion über einen höheren Laufindex. Die charakteristischen Unterschiede zur Beschreibung der zeitlichen Relationen von $E_j^{m_1}$ und $E_k^{m_2}$ ergeben sich aus den Differenzen der synchronisierten Start- und Endmarken und dienen als Grundlage zur automatischen Klassifikation der vorliegenden Überlagerungsart. Für die weitere Beschreibung der Zeitrelationen werden bei den einzelnen Symbolen die Modalitätenbezeichnungen weggelassen, da sich die Modalität m einer Eingabe E_i^m auch eindeutig über den Index i des Events rekonstruieren lässt.

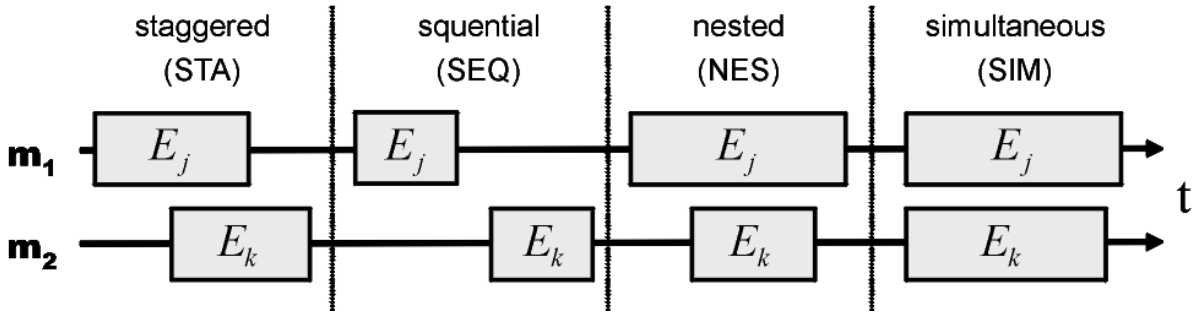


Abb. 4.2: Prinzipiell unterschiedliche Überlagerungsarten zweier Benutzereingaben E_j und E_k; angegeben sind die Fachbegriffe nach Oviatt [119] und die Abkürzungen für die entsprechenden Ereignisse

In Abbildung 4.1 sind für den Fall einer überlagerten bimodalen Interaktion von E_j und E_k drei Verzögerungszeiten T_{v1}, T_{v2} und T_{v3} ausgewiesen, die in der weiteren Diskussion eine besondere Rolle spielen. Während T_{v1} den Vorlauf der ersten und T_{v3} den Nachlauf der zweiten Interaktion spezifiziert, gibt T_{v2} den Zeitraum der direkten Überlagerung an. Bei einer entsprechend andersartigen Konstellation der beiden Interaktionen können T_{v2} und T_{v3} auch negative Werte annehmen. Die einzelnen Zeiten berechnen sich wie folgt:

$$T_{v1}(E_i^{MM}) = t_s(E_k) - t_s(E_j) \tag{4.22}$$

$$T_{v2}(E_i^{MM}) = t_e(E_j) - t_s(E_k) \tag{4.23}$$

$$T_{v3}(E_i^{MM}) = t_e(E_k) - t_e(E_j) \tag{4.24}$$

Auf Basis dieser Daten können prinzipiell vier unterschiedliche Überlagerungstypen identifiziert werden. In Anlehnung an die von Oviatt [119] geprägten Begiffe spricht man allgemein von gestaffelten (*staggered*), sequentiellen (*sequential*), eingebundenen (*nested*) und von simultanen (*simultaneous*) multimodalen Interaktionen. Die dazugehörigen Ereignisse werden bezogen auf die englischen Fachtermini jeweils mit den ersten drei Buchstaben abgekürzt: E^{STA}, E^{SEQ}, E^{NES} und E^{SIM}. Einen Überblick zu den möglichen Überlagerungstypen vermittelt auch die Darstellung in Abbildung 4.2. Da die einzelnen Zeitmarken nur framegenau (1/25 sec.) bestimmt werden können, muss eine Ungenauigkeit in diesem Bereich akzeptiert werden. Simultane Eingaben haben nach der Definition einen gemeinsamen Anfangspunkt und zusätzlich wird angenommen, dass E_j immer auch die zeitlich längere Interaktion repräsentiert. Mathematisch lassen sich die einzelnen Relationen wie folgt formulieren:

$$E_i^{MM} := \begin{cases} E_i^{STA} & \text{falls} \quad t_s(E_k) < t_e(E_j) \quad \wedge \quad t_e(E_k) > t_e(E_j) \\ E_i^{SEQ} & \text{falls} \quad t_s(E_k) > t_e(E_1) \\ E_i^{NES} & \text{falls} \quad t_s(E_k) < t_e(E_j) \quad \wedge \quad t_e(E_k) < t_e(E_j) \\ E_i^{SIM} & \text{falls} \quad t_s(E_j) = t_s(E_k) \quad \wedge \quad t_e(E_j) > t_e(E_k) \end{cases} \tag{4.25}$$

In Ergänzung zu Gleichung 4.23 wird speziell für sequentielle Interaktionen eine zusätzliche Verzögerungszeit T_{V4} eingeführt, die im Gegensatz zu T_{V2} den zeitlichen Abstand zweier Benutzereingaben, die separat nacheinander durchgeführt werden, als positiven Wert angibt:

$$T_{v4}(E_i^{MM}) \;=\; t_s(E_k) - t_e(E_j) \;=\; -T_{V2}(E_i^{MM}) \tag{4.26}$$

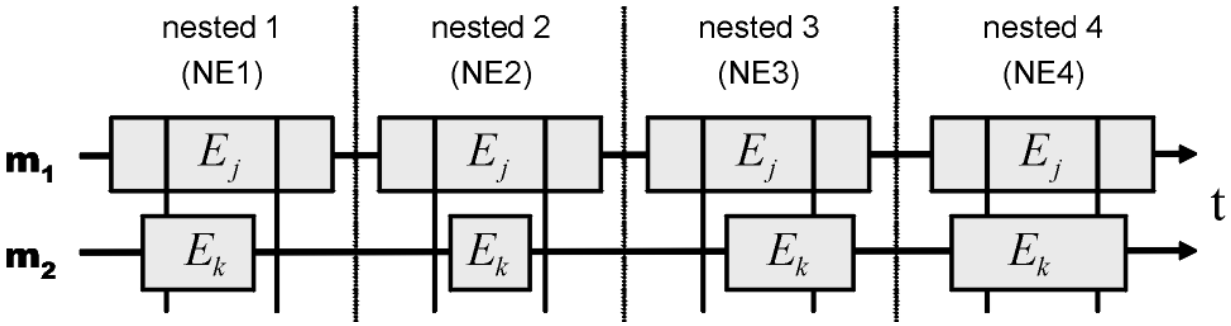


Abb. 4.3: Spezielle Variationen für eingebettete (*nested*) multimodale Benutzereingaben

Da speziell in der dritten Überlagerungsgruppe E^{NES} sowohl bei der Länge $T(E_k)$, als auch bei dem Anfangszeitpunkt $t_s(E_k)$ der zweiten Interaktion im Vergleich zu der ersten Interaktion E_j in den Versuchen sehr starke Schwankungen zu beobachten waren, werden für die Auswertung vier zusätzliche Untergruppen E^{NE1} bis E^{NE4} eingeführt. Dazu wird die Gesamtdauer der ersten Interaktion in drei Teilstücke zerlegt und die trennenden Zeitmarken $t_a = 0,25\ T(E_j)$ und $t_b = 0,75\ T(E_j)$ definiert. Abbildung 4.3 verdeutlicht die entsprechenden Zusammenhänge für eine bimodale Benutzereingabe. Beispielsweise bei dem Überlagerungstyp E^{NE1} beginnt die zweite Interaktion E_k kurz nach E_j, ist aber bezogen auf die Gesamtdauer wesentlich kürzer. Im Fall von E^{NE4} beginnen beide Interaktionen kurz nacheinander und sind auch annähernd gleichlang. Die anderen beiden Typen ergeben sich in analoger Weise. Für die Vor- und Nachlaufzeiten T_{v1} und T_{v3} aus den Gleichungen 4.22 und 4.24 ergeben sich folgende Beziehungen:

$$E^{NES} = \begin{cases} E^{NE1} & \text{falls} \quad T_{v1} < t_a \ \wedge \ t_a < T_{v3} < t_b \\ E^{NE2} & \text{falls} \quad t_a < T_{v1} < t_b \ \wedge \ t_a < T_{v3} < t_b \\ E^{NE3} & \text{falls} \quad t_a < T_{v1} < t_b \ \wedge \ T_{v3} < t_a \\ E^{NE4} & \text{falls} \quad T_{v1} < t_a \ \wedge \ T_{v3} < t_a \end{cases} \tag{4.27}$$

Die oben beschriebenen Relationen gelten in dieser ausführlichen Form nur für multimodalen Interaktionen mit den semantisch höherwertigen Modalitäten Handgestik, Kopfgestik und Sprache. Da taktile Interaktionen von der zeitlichen Ausdehnung her nicht vollständig korrekt erfasst werden, wird im Falle einer kombinierten Eingabe mit Touchscreen und Tastenkonsole lediglich geprüft, ob E^T und E^A durch E^H, E^K bzw. E^S zeitlich überlagert sind oder ob sie eine sequentielle Ordnung aufweisen.

Nach erfolgreicher Annotation wird in einem erneuten Durchlauf die aufbereitete Protokolldatei bezogen auf rein temporale Aspekte analysiert. Neben verschiedenen Bearbeitungszeiten werden an dieser Stelle auch die Frequenz der isolierten Benutzer- und Systemereignisse und die Zeitspannen zwischen den einzelnen Ereignisklassen ausgewertet. Sei b mit $b \in \{1..5\}$ ein Indikator für einen speziellen Versuchsblock und N_b^{AB} die Anzahl der Aufgaben in diesem Block, dann gibt T_b^G die benötigte Zeit für die Bearbeitung aller Aufgaben in Block b und $T_{i,b}^A$ mit $i \in \{1..N_b^{AB}\}$ die Zeit für eine einzelne Aufgabe an. Die einzelnen Zeiten werden speziell im Rahmen einer Effizienzanalyse benötigt.

4.3 Desktop Virtual-Reality (DVA)

An der Versuchsreihe in der Desktop Virtual-Reality Umgebung haben insgesamt 40 Personen teilgenommen. Mit 33 männlichen und nur sieben weiblichen Versuchspersonen wird das Verhältnis der Geschlechter deutlich von den männlichen VPen dominiert. Das Durschnittsalter liegt bei $\bar{J}_{VP} = 28,8$ Jahren mit einer Standardabweichung $\delta_J = 6,5$. Die älteste Versuchsperson war 49 und die jünge VP 21 Jahre alt. Die Versuchsgruppe besteht zu einem großen Teil aus Studenten der Fakultät für Elektro- und Informationstechnik und der Informatik (14 Personen), die zweitstärkste Fraktion mit zwölf Teilnehmern stellen wissenschaftliche Mitarbeiter verschiedener Lehrstühle der Technischen Universität München. Darüber hinaus verfügen die restlichen 14 Teilnehmer über eine breit gestreute schulische Ausbildung und sind in verschiedenen Berufen tätig.

Bei der Auswahl der einzelnen Versuchspersonen wurde extrem großer Wert darauf gelegt, dass keiner der Teilnehmer vorher über intensive Erfahrungen mit multimodalen Bedienschnittstellen aus dem Virtual-Reality Umfeld verfügte. Im Hinblick auf die konkrete Testdomäne haben die Testteilnehmer damit eine vergleichbare Ausgangssituation. Die Auswertung der Antworten in dem Anfangsfragebogen bzgl. der allgemeinen Kenntnisse und Erfahrungen mit verschiedenartigen Informationssystemen und Eingabemedien ergibt a-priori eine gruppenspezifische Einteilung der Teilnehmer in elf Anfänger, 20 Normalnutzer und neun Experten.

In den ersten beiden Versuchsphasen haben die Versuchsteilnehmer die Gelegenheit, ausreichend Erfahrung im Umgang sowohl mit dem Testsystem als auch mit den einzelnen Eingabemöglichkeiten zu sammeln. Die dritte Phase des Versuchs (Block V), bei dem ohne jegliche Einschränkungen sämtliche Eingabemodalitäten in gleichem Maße zur Verfügung stehen, stellt das zentrale Testszenario zur Evaluierung der multimodalen Interaktionsparadigmen dar. Daher konzentrieren sich die folgenden Ausführungen zu den Ergebnissen auch hauptsächlich auf diesen einen Versuchsteil. Daten zu den anderen Versuchsblöcken werden, sofern sie für das Gesamtverständnis von Bedeutung sind, entsprechend mitdiskutiert. Der interessierte Leser findet weiterführendes Datenmaterial zu allen Versuchsteilen in Anhang A.

Automatische Datenanalyse

Im Anschluss an die manuelle Klassifikation der semantisch höherwertigen Interaktionen läuft die Analyse der quantitativen Versuchsdaten weitgehend automatisch ab. Pro Versuchsperson $x \in \{1..N_{VP}\}$ wird jede protokollierte Benutzerinteraktion $E^{BE}_{x,i}$ mit $i \in \{1..N^{BE}_x\}$ zunächst in Hinblick auf die semantische Zusammensetzung der resultierenden Aktion untersucht. Danach wird bestimmt, ob es sich grundsätzlich um eine unimodale ($E^{UM}_{x,i}$) oder um eine multimodale ($E^{MM}_{x,i}$) Benutzereingabe handelt. Für den Fall einer multimdoalen Eingabe wird zusätzlich die Anzahl der beteiligten Modaliäten ermittelt und die funktionalen Relationen der einzelnen Interaktion untersucht (redundant, rivalisierend oder komplementär). Als Ergebnis erhält man die Zuordnung von $E^{BE}_{x,i}$ in mehrere Event-Klassen. Im lokalen Kontext von jeweils zwei direkt aufeinanderfolgenden Benutzereingaben E^{BE}_j und E^{BE}_k werden zusätzlich noch die Modalitätenübergänge analysiert.

Innerhalb einer Ereignis-Klasse wird nicht explizit jedes auftretende Ereignis festgehalten, sondern pro Versuchsperson x mit $x \in \{1..N_{VP}\}$ die Anzahl, die Mittelwerte, die Extremwerte und die Standardabweichungen der jeweiligen Daten gespeichert. Zusätzlich werden auch für die einzelnen Gruppen U mit $U \in \mathcal{U}$ die entsprechenden Werte ermittelt. Programmtechnisch

DVA	TVR	TLR	TOU	ROU	RLR	RRO
ANF	50,7	8,9	8,4	5,5	16,8	9,5
NOR	51,4	10,1	4,6	5,2	16,5	12,1
EXP	52,4	10,3	6,4	8,7	13,9	8,4
AVP	51,4	9,8	6,1	6,1	16,0	10,6

(a)

DVA	A^{VK}	A^{TK}
ANF	58,2	41,8
NOR	73,6	26,4
EXP	72,6	27,4
AVP	69,5	30,5

(b)

Tab. 4.1: (a) Prozentuale Verteilung der Kommando-Typen bezogen auf die Systemfunktionalitäten des DVA-Testsystems und (b) bezogen auf Vollkommandos (A^{VK}) und Teilkommandos (A^{TK}) für die einzelnen Benutzerklassen $U \in \mathcal{U}$

werden diese Daten in einer gruppenspezifischen Matrix organisiert, bei der in den Spalten die einzelnen Versuchspersonen und in den Zeilen die verschiedenen Event-Klassen mit den jeweiligen Häufigkeiten aufgetragen sind. Im nächsten Schritt findet eine Konsistenzprüfung statt und zentrale zentrale Aussagen werden herausgefiltert.

Danach werden für die Eingaben in dem reduzierten Datenmaterial E_j^{ZBE} mit $j \in \{1..N^{ZBE}\}$ die Dauern der jeweiligen Interaktionen bestimmt. Bei kombinierten Eingaben werden zusätzlich, wie in Gleichung 4.22 bis 4.24 für den bimodalen Fall beschrieben, einzelne charakteristische Zeitrelationen berechnet. Unter Betrachtung der spezifischen Modalitäten und der Reihenfolge ihrer Anfangszeitpunkte erfolgt dann eine Bestimmung des vorliegenden Überlagerungstyps. Auf diese Weise lassen sich Kategorien von gleichartigen Interaktionsmustern ableiten. So bezeichnet beispielsweise $\mathcal{E}_{HS}^{STA}$ die Menge aller bimodalen Interaktionen, an denen die Modalitäten Handgestik und Sprache beteiligt sind, wobei E^H zeitlich vor E^S kommt und die einzelnen Eingaben zeitlich gestaffelt überlagert sind. Für trimodale und komplex multimodale Interaktionen wird das Auswertungsverfahren jeweils paarweise mit den beteiligten Modalitäten iteriert. Die Aufschlüsselung der individuellen Ereignisklassen ist abschließend in den Tabellen A.1 bis A.2 noch einmal zusammenfassend dargestellt.

4.3.1 Kommandostruktur

Die Verteilung der Kommando-Typen für die freie Interaktion im fünften Versuchsblock wird in Tabelle 4.1(a) gezeigt. Dargestellt sind die durchschnittlichen Mittelwerte nach Gleichung 4.11 für die einzelnen Benutzerklassen U mit $U \in \mathcal{U}$. Das Ergebnis steht in direkter Beziehung zu den speziellen Aufgaben, die den Versuchspersonen fest vorgegeben waren. Es stellt daher kein allgemein gültiges Ergebnis für die Interaktionen mit dem entsprechenden Software-Prototypen dar, sondern dient im Kontext der anderen Ergebnisse vielmehr dem Vergleich zwischen den einzelnen Testpersonen und den verschiedenen Benutzerklassen.

Insgesamt sind im Rahmen der freien Interaktion von sämtlichen Versuchspersonen 9734 Systemkommandos in den Logfiles aufgezeichnet und analysiert worden. Die Anzahl der Benutzereingaben N_x^{BE} variiert für die einzelnen Versuchspersonen dabei von 85 zu 679 sehr stark. Als durchschnittlicher Wert ergibt sich pro VP eine Anzahl von $\bar{N}^{BE} = 237,4$ Interaktionen mit einer Standardabweichung von $\sigma_{BE} = 115,3$. Die genauen Werte für die jeweiligen Benutzerklassen und auch für die anderen Versuchsblöcke befinden sich im Anhang A.

Den Hauptanteil mit im Schnitt über 51% aller Interaktionen stellen translatorische Vorwärts- und Rückwärtsbewegungen (TFB) dar, wobei sich die einzelnen Benutzerklassen nicht signifikant unterscheiden. Die zweithäufigste Kommando-Klasse bilden Links/Rechts-Drehungen mit im Mittel 16% aller Interaktionen. Während die Klasse der Anfänger und Normalnutzer nahezu gleiche Werte aufweisen, liegt der Durchschnitt bei den Experten knapp 2% darunter. Vergleicht man die Anteile der Rotationen in der Bildebene (RRO), so läßt hier ein deutlicher Mehranteil für die Gruppe der Expertennutzer beobachten.

Mit Bezug auf den semantischen Gehalt setzt sich eine Systemaktion im DVA-Szenario (A^{DVA}) allgemein aus einem funktionellen Teil (A^F) und einer Parameterteil (A^P) zusammen (siehe auch Tabelle 3.1). Der funktionelle Teil spezifiziert Art der Bewegung und der Parameter gibt die Richtung an.

$$< A^{DVA} > ::= < A^F > < A^P > \tag{4.28}$$

In dem reduzierten Vokabular des Testsystems beschränken sich die Variationsmöglichkeiten für A^F auf translatorische (A^{TRA}) und rotatorische (A^{ROT}) Bewegungen. Demgegenüber stehen für den Parameter A^P mit Bezug auf die aktuelle Position des Benutzers in der virtuellen Welt sechs Richtungen zur Verfügung: vor (A^{VOR}), zurück (A^{ZUR}), links (A^{LI}), rechts (A^{RE}), nach oben (A^{NOB}) und nach unten (A^{NUN}). Die Abkürzungen für die entsprechenden Kommandos resultieren aus den Anfangsbuchstaben der richtungsanzeigenden Adjektive. In EBNF-Notation ergeben sich die folgenden Zusammenhänge:

$$< A^F > \ ::= \ A^{TRA} \mid A^{ROT} \tag{4.29}$$

$$< A^P > \ ::= \ A^{VOR} \mid A^{ZUR} \mid A^{LI} \mid A^{RE} \mid A^{NOB} \mid A^{NUN} \tag{4.30}$$

Eine gültige Benutzerinteraktion kann entweder beide Komponenten oder nur einen speziellen Teil enthalten. Im ersten Fall bezeichnet man die mit dem Eingabeereignis verknüpfte Aktion als *Voll-Kommando* A^{VK}, im zweiten Fall als *Teil-Kommando* A^{TK}. Bei dieser Klassifikation wird zunächst nicht unterschieden, ob ein Teil-Kommando aus einem funktionellen Anteil oder einem isolierten Parameterwert besteht.

$$< A^{VK} > \ ::= \ < A^F > < A^P > \tag{4.31}$$

$$< A^{TK} > \ ::= \ < A^F > \ \mid \ < A^P > \tag{4.32}$$

Die Verteilung der entsprechenden Interaktionsmuster jeweils für die einzelnen Benutzerklassen $U \in \mathcal{U}$ ist in Tabelle 4.1(b) dargestellt. Es fällt auf, dass die Gruppe der normalen Nutzer und der Experten mit 73,6% bzw. 72,6% Anteil an Voll-Kommanods sehr ähnliche Verteilungswerte aufweisen. Bei den Anfängern kann demgegenüber mit einem Anteil von fast 42% eine deutliche Tendenz zu einer wesentlich stärker ausgeprägten Nutzung von Teil-Kommandos beobachtet werden. Eine detaillierte Untersuchung des Datenmaterials liefert den Grund für die Diskrepanz in den gruppenspezifischen Verteilungswerten. Während bezogen auf sämtliche Eingaben sowohl Normalnutzer als auch Experten nur sehr vereinzelt isolierte Funktionsanweisungen geben, ist speziell dieser Anteil bei den Anfängern deutlich höher. In der Praxis bedeutet das, dass Anfänger die semantische Struktur einer Navigationsanweisung wesentlich häufiger aufspalten und die einzelnen Teilbefehle zeitlich getrennt formulieren. Die Komplexität der Systembefehle ist somit im Mittel bei den Anfängern insgesamt geringer als bei den beiden anderen Benutzergruppen.

DVA	ANF	NOR	EXP	AVP
E^{UM}	73,8	81,1	81,5	79,8
E^{MM}	26,2	17,9	18,5	20,2

(a)

DVA	ANF	NOR	EXP	AVP
E^{UM}_{TA}	56,5	74,0	67,0	68,0
E^{UM}_{HKS}	43,5	26,0	33,0	32,0

(b)

Tab. 4.2: (a) Prozentuale Verteilung der unimodalen (E^{UM}) und multimodalen (E^{MM}) Benutzereingaben im DVA-Szenario für den fünften Versuchsblock bei den unterschiedlichen Benutzergruppen $U \in \mathcal{U}$ und (b) zusammengefasste prozentuale Verteilung der taktilen (E^{UM}_{TA}) und der semantisch höherwertigen Eingaben (E^{UM}_{HKS}) bezogen ausschließlich auf die unimodalen Benutzerinteraktionen

4.3.2 Modalitätennutzung

Im Rahmen einer modalitätenspezifischen Nutzungsanalyse werden die einzelnen Benutzereingaben zunächst in unimodale (E^{UM}) und multimodale (E^{MM}) Interaktionen mit dem Testsystem unterteilt. Tabelle 4.2(a) zeigt die prozentualen Anteile dieser beiden Ereignisklassen für die einzelen Benutzergruppen U mit $U \in \mathcal{U}$ bezogen auf den fünften Versuchsblock. Die Ergebnisse stellen jeweils Mittelwerte der prozentualen Anteile innerhalb der einzelnen Benutzerklassen dar. Zum Vergleich sind im Anhang, Tabelle A.4 auch die gruppenspezifischen Werte für die komplette zweite Versuchsphase angegeben.

Der Anteil der unimodalen Benutzereingaben im DVA-Szenario überwiegt deutlich. Dieses Ergebnis entspricht vollständig den initialen Erwartungen. Die Aufgaben sind relativ einfach strukturiert und auch das zur Verfügung stehende Applikationsvokabular ist auf wenige essentielle Funktionen begrenzt. Darüber hinaus können die Funktionen des Testsystems vollständig durch einzelne Modalitäten angesteuert werden. Aus diesen Gründen besteht a priori kein Zwang zu einer multimodalen Kommunikationsweise. Eine detaillierte, aufgabenbezogene Analyse des Datenmaterials ergibt, dass der Anteil der multimodalen Benutzereingaben mit ansteigender Komplexität der Bedienaufgaben deutlich zunimmt. Obwohl im Mittel bei etwa vier von fünf Interaktionen ausschließlich unimodale Eingaben beobachtet werden, stellen kombinierte multimodale Eingaben einen unverzichtbaren Kommunikationsstil dar, da sie vor allem bei einem logischen Wechsel der anzusteuernden Systemfunktionalitäten eingesetzt werden.

Modalitätenverteilung

Die prozentuale Verteilung der unimodalen Systeminteraktionen E^{UM} auf die einzelnen Modalitäten $m \in \mathcal{M}_{IN}$ wird, gemittelt über sämtliche Benutzer der jeweiligen Klasse $U \in \mathcal{U}$, in Tabelle 4.3(a) dargestellt. Offensichtlich favorisieren im Falle einer unimodalen Eingabe alle Benutzergruppen die Tastatur als primäres Eingabegerät. Der Durchschnitt liegt mit 40,2% deutlich über den Anteilen der anderen Modalitäten. Als nächstbeste Wahl bevorzugen Normalnutzer mit 33,9% und Experten mit 29,9% den Touchscreen, während Anfänger mit 24,0% die Sprache als isolierte Eingabemodalität nutzen. Handgestik folgt zumindest bei den Anfängern mit 14,9% als dritte Wahl noch relativ dicht. Stellt man allgemein für die unimodalen Eingabe die prozentualen Anteile der semantisch höherwertigen (E^{UM}_{HKS}) und der taktilen Interaktionen (E^{UM}_{TA}) gegenüber, so lässt sich bei den Anfängern eine deutlich stärker vorhandene Tendenz zugunsten einer sprach- und gestenbasierten Eingabeform identifizieren. Tabelle 4.2(b) enthält das genaue Zahlenmaterial.

DVA	E_T^{UM}	E_A^{UM}	E_H^{UM}	E_K^{UM}	E_S^{UM}
ANF	13,1	43,4	14,9	4,6	24,0
NOR	33,9	40,1	10,2	4,3	11,5
EXP	29,9	37,1	16,0	7,4	9,6
AVP	27,8	40,2	12,8	5,1	14,1

(a)

DVA	E_T^{MM}	E_A^{MM}	E_H^{MM}	E_K^{MM}	E_S^{MM}
ANF	30,2	33,5	43,8	38,4	65,2
NOR	37,7	47,0	26,3	18,2	76,4
EXP	33,2	42,0	20,7	19,7	86,0
AVP	34,8	42,5	29,2	23,5	76,0

(b)

Tab. 4.3: (a) Prozentuale Verteilung der unimodalen (E^{UM}) und (b) der multimodalen (E^{MM}) Benutzereingaben im DVA-Szenario für die dritte Versuchsphase mit der freien Interaktion, aufgeteilt nach den unterschiedlichen Benutzergruppen $U \in \mathcal{U}$

In Bezug auf die multimodalen Systeminteraktionen E^{MM} dominiert bei allen Benutzergruppen die Sprache als primäre Eingabemodalität. Während sie bei den Anfängern an 65,2% aller kombinierten Eingaben beteiligt ist, liegt der Sprachanteil mit 76,4% bei den Normalnutzern und 86,0% bei den Experten sogar noch wesentlich höher. Als zweitwichtigste Modalität kristallisiert sich für die Guppen NOR und EXP mit 47% bzw. 42% die Tastatur, bei ANF dagegen mit 43,8% die Handgestik heraus. Insgesamt sind bei den Anfängern neben Sprache und Handgestik die Beteiligungen der anderen Modalitäten wesentlich gleichmäßiger verteilt als bei den beiden anderen Gruppen. Bei den Normalnutzern und den Experten ist eine klare Präferenz für die haptischen Modalitäten zu beobachten, Hand- und Kopfgestik folgen mit weitaus größerem Abstand als bei den Anfängern. Ein häufig beobachtetes Interaktionsmuster ist gruppenübergreifend der Einsatz von Sprache zur Auswahl einer Bewegungsart kombiniert mit Haptik zur Spezifikation der Richtung.

Fasst man die Anteile der taktilen (E_{TA}^{MM}) und der semantisch höherwertigen Interaktionen (E_{HKS}^{MM}) bezogen auf multimodale Benutzereingaben zusammen, so ergibt sich gegenüber den unimodalen Werten in Tabelle 4.2(b) ein massiv verändertes Verhältnis. Bei allen Benutzergruppen sind die sprach- und gestenbasierte Eingabeformen zusammengenommen deutlich in der Überzahl. Besonders eindrucksvoll lässt sich diese Tendenz bei den Anfängern nachweisen. Der zusammengenommene Anteil von Touchscreen und Tastatur beträgt im komibinierten Einsatz weniger als die Hälfte der Anteile der anderen Modalitäten.

Modalitätenkombinationen

Im Rahmen der Evaluierung prototypischer Interaktionsmuster kommt einer detaillierten Untersuchung der aufgetretenen Modalitätenkombinationen eine besondere Rolle zu. Für die weitergehende Analyse einer multimodalen Benutzereingabe E_i^{MM} wird zunächst die konkrete Anzahl N_i der beteiligten Modalitäten bestimmt. Tabelle 4.4(a) veranschaulicht die prozentuale Verteilung der bimodalen E^{BMM}, trimodalen E^{TMM} und komplex multimodalen E^{KMM} Interaktionen für die einzelnen Benutzergruppen U mit $U \in \mathcal{U}$. Mit im Mittel über 83% werden, bezogen auf alle Versuchspersonen, überwiegend immer genau zwei Modalitäten miteinander kombiniert. Dabei lässt sich von den Anfängern über die Normalnutzer bis hin zu den Experten ein kontinuierlicher Anstieg des Anteils an bimodalen Interaktionen beobachten. Trimodale Interaktionen machen zumindest bei den Anfängern noch fast ein Viertel aller kombinierten

DVA	ANF	NOR	EXP	AVP
E^{BMM}	69,7	86,0	92,7	83,6
E^{TMM}	24,5	9,3	7,3	12,6
E^{KMM}	5,8	4,7		3,8

(a)

DVA	ANF	NOR	EXP	AVP
E^{RED}	59,7	27,8	27,9	36,6
E^{RIV}	11,1	2,7	6,3	5,8
E^{CMP}	78,8	91,8	96,6	89,3

(b)

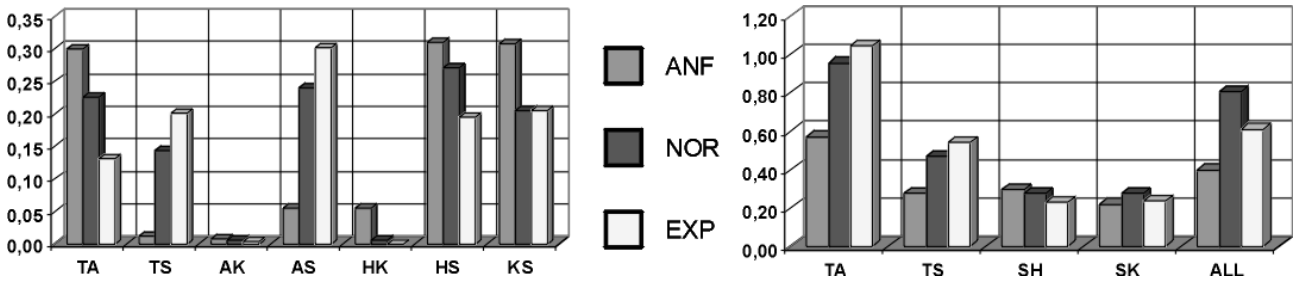
Tab. 4.4: (a) Prozentuale Verteilung der bimodalen (E^{BMM}), trimodalen (E^{TMM}) und komplex multimodalen (E^{KMM}) Interaktionen und (b) prozentuale Verteilung der strukturellen Zusammensetzung aus redundanten (E^{RED}), rivalisierenden (E^{RIV}) und komplementären (E^{CMP}) multimodalen Benutzereingaben; aufgeteilt nach den einzelnen Benutzerklassen $U \in \mathcal{U}$

Aktionen aus und stellen daher ebenfalls einen wichtigen Kommunikationsstil dar. Eingaben mit mehr als drei beteiligten Modalitäten treten bei Anfängern mit 5,8% und Normalnutzern mit 4,7% nur sehr vereinzelt, bei Experten überhaupt nicht auf.

Tabelle 4.4(b) veranschaulicht die Struktur im Hinblick auf die funktionale Zusammensetzung der kombinierten Eingaben. Dargestellt sind die Anteile der redundanten (E^{RED}), rivalisierenden (E^{RIV}) und komplementären (E^{CMP}) Benutzereingaben bezogen auf sämtliche multimdodalen Interaktionen E^{MM} im fünften Versuchsblock. Mit deutlich über 85% haben alle Benutzergruppen komplementäre Informationszusammensetzungen am häufigsten eingesetzt. Interessanterweise kann von den Anfängern über die Normalnutzer bis hin zu den Experten analog zu den Werten aus Tabelle 4.4(a) auch hier ein kontinuierlicher Anstieg in dem respektiven Kommunikationsstil beobachtet werden. Dieses Resultat kann damit begründet werden, dass speziell die Gruppe der Anfänger verstärkt auch redundante Eingaben verwenden. Mit 59,7% ist der Anteil der redundanten Interaktionen mehr als doppelt so groß im Vergleich zu den Normalnutzern mit nur 27,8% und den Experten mit 27,9%. Das Ergebnis unterstreicht eindrucksvoll die generelle Beobachtung, dass kombinierte Eingaben bei den Anfänger oftmals gleichzeitig sowohl komplementäre als auch redundante Informationsanteile enthalten.

Gemittelt über sämtliche Versuchsteilnehmer sind nur an 12,5% aller komplementären multimodalen Interaktionen mehr als zwei verschiedene Modalitäten beteiligt. Auch dieses Ergebnis bestätigt die initialen Erwartungen, dass die rein bimodalen Interaktionen den Großteil der kombinierten Eingaben ausmachen. Harte, im Sinne von Nigay und Coutaz synergistische multimodale Eingaben (E^{HAC}), bei denen bezogen auf die strukturellen Kommandoanteile keine intramodalen Abhängigkeiten vorhanden sind, können immerhin noch in 67,3% aller komplementären Interaktionen beobachtet werden.

Bezogen auf die Struktur der Systemkommandos bei kombinierten Benutzereingaben hat sich herausgestellt, dass sowohl redundante als auch rivalisierende Interaktionen ausschließlich aus Parameterangaben (A^P) bestehen. Über zwei Drittel aller redundanten Interaktionen setzen sich aus den Modalitäten Sprache und Handgestik zusammen. Bei den komplementären Eingaben werden die Funktion (A^F) und der Parameter (A^P) durch unterschiedliche Modalitäten spezifiziert. In 42% dieser Fälle besteht die Interaktion aus taktilen Eingaben, in weiteren 34% wird eine Spracheingabe durch eine taktile Interaktion ergänzt. Eingaben sowohl mit redundanten als auch mit komplementären Anteilen (E^{REC}) treten in verschiedenen Formen auf. Häufig



(a) (b)

Abb. 4.4: (a) Prozentuale Verteilung der bimodalen Kombinationen $K_{m_1 m_2}$ mit $m_1, m_2 \in \mathcal{M}_{IN}$ im DVA-Szenario und (b) Effizienz von kombinierten Eingaben, gemessen aus der Anzahl der Interaktionen pro Zeit[1/s]; pro Kombination repräsentiert in beiden Diagrammen jeweils die linke Säule die Anfänger (ANF), die mittlere die Normalnutzer (NOR) und die rechte Säule die Experten (EXP)

werden A^F und A^P als Vollkommando durch die Sprache und A^P zusätzlich noch durch Hand- oder Kopfgestik angegeben. Kombinationen von zwei Vollkommandos oder bei trimodalen Eingaben von drei Teilkommandos sind im Testmaterial nicht enthalten.

Die prozentuale Verteilung der bimodalen Modalitätenkombinationen $K_{m_1 m_2}$ mit $m_1, m_2 \in \mathcal{M}_{IN}$ ist gemittelt über sämtliche VPen in Abbildung 4.4(a) darsgestellt. Analog zu Gleichung 3.1 ist die Reihenfolge der einzelnen Modalitäten in der Darstellung willkürlich, d.h. $K_{m_1 m_2}$ und K_{m_2, m_1} werden zu einer Gruppe zusammengefasst. Von den zehn prinzipiell möglichen Kombinationen fehlen K_{TK}, K_{TH} und K_{AH}, da diese Konstellationen im Versuch nicht aufgetreten sind. Die Gruppe der Anfänger setzt zu etwa gleichen Anteilen die taktilen Kombinationen (K_{TA}) und die Kombination von Sprache mit Handgestik (K_{HS}) bzw. mit Kopfgestik (K_{KS}) ein. Während Anfänger die Verbindung von Touchscreen und Sprache (K_{TS}) kaum nutzen, nimmt diese Kombination bei den Normalnutzern mit 13,1% und bei den Experten mit 19,4% einen signifikanten Anteil ein.

Die Kombinationen K_{AK} und K_{HK} werden von allen Benutzergruppen kaum genutzt. Einen interessanten Fall stellt die Verbindung der Modalitäten Tastatur und Sprache (K_{AS}) dar. Von den Anfängern nur in 5% der Fälle angewendet, macht sie bei den Normalnutzern mehr als ein Fünftel und bei den Experten sogar fast ein Drittel aller Kombinationen aus. Bei keiner anderen Kombination sind die gruppenspezifischen Werte so unterschiedlich. Anfänger scheinen die Kombination von taktilen Modalitäten und Sprache nicht so sehr zu mögen. Bei den beiden anderen Gruppen werden diese Kombinationen häufig benutzt.

4.3.3 Effizienzuntersuchung

Die einzelnen Modalitätenkombinationen eignen sich unterschiedlich gut zur Erfüllung der Aufgaben. Abbildung 4.4(b) veranschaulicht die Effizienz von kombinierten Benutzereingaben zur Bedienung des Testsystems im DVA-Szenario. Im Kontext dieser Untersuchung wird die Effizienz definiert als die Anzahl der durchgeführten Aktionen pro Zeit und in der Einheit [1/s] angegeben. Es werden dazu die vier Testblöcke mit den fest vorgeschriebenen Modalitätenkombinationen der zweiten Phase ($K_{TA}, K_{TS}, K_{HS}, K_{KS}$) im Vergleich zu der freien Interaktion (K_{ALL}) in der dritten Phase evaluiert.

Die Gruppe der Normalnutzer und Experten arbeiten am effizientesten, wenn sie den Touchscreen als primäres Eingabegerät mit Tastatur oder Sprache kombinieren. Bei den Anfängern setzt sich hingegen der Trend zu den semantisch höherwertigen Modalitäten fort. Interessanterweise arbeiten sie mit Handgestik und Sprache effizienter als die beiden anderen Gruppen. Im Fall von K_{HS} und K_{KS} sind allgemein die gruppenspezifischen Unterschiede sehr gering. Für die anderen Kombinationen entspricht das Ergebnis fast vollständig den Erwartungen, dass von den Anfängern zu den Experten ein streng monotoner Anstieg in den Effizienzwerten zu beobachten ist.

Die rein taktilen Interaktionsformen mittels Touchscreen und Tastatur (K_{TA}) stellen im Testaufbau für alle Benutzerklassen eindeutig die effektivste Form der Modalitätenkombination dar. Vergleicht man die Performance der anderen Kombinationen K_{TS}, K_{SH} und K_{KS} mit der von K_{ALL} in Block V, so lässt sich generell ein signifikanter Vorteil für die freie Interaktionsform feststellen. Mit 0,8 [1/s] gegenüber 0,61 [1/s] agieren die Normalnutzer hier sogar knapp 32% effizienter als die Gruppe der Experten.

Modalitätenübergänge

Einen weiteren Schwerpunkt der Benutzerstudie bildet die Untersuchung der Übergänge zwischen den verwendeten Modalitäten bei aufeinanderfolgenden Benutzerinteraktionen. Dazu wird pro Ausgangsmodalität m_s mit $m_s \in \mathcal{M}_{IN}$ die Anzahl der relativen Übergänge auf die in der jeweils folgenden Interaktion benutzte Zielmodalität m_t mit $m_t \in \mathcal{M}_{IN}$ ermittelt. Sei $N^{MU}_{m_s,m_t}$ die absolute Anzahl der Modalitätenübergänge von Eingabemedium m_s auf m_t, dann ergibt sich die modalitätenspezifische Gesamtanzahl der Übergänge ($N^{GMU}_{m_s}$) durch einfache Summierung der individuellen Übergänge:

$$N^{GMU}_{m_s} = \sum_{m_t \in \mathcal{M}_{IN}} N^{MU}_{m_s,m_t} \qquad \forall\, m_s \in \mathcal{M}_{IN} \tag{4.33}$$

Die Anzahl der relativen Modaliätenübergänge ($N^{RMU}_{m_s,m_t}$) wird aus dem Quotienten der individuellen Übergänge ($N^{MU}_{m_s,m_t}$) und der entsprechenden Basis $N^{GMU}_{m_s}$ berechnet:

$$N^{RMU}_{m_s,m_t} = \frac{N^{MU}_{m_s,m_t}}{N^{GMU}_{m_s}} \qquad \forall\, m_s, m_t \in \mathcal{M}_{IN} \tag{4.34}$$

Tabelle 4.5(a) zeigt die relativen Modalitätenübergänge gemittelt über alle Versuchspersonen. Identische Übergänge auf die Ausgangsmodalität sind gesondert hervorgehoben. Sie stellen bis auf Kopfgesten die häufigste Übergangsform dar. Vergleicht man allgemein die Übergänge bei den taktilen und den semantisch höherwertigen Modalitäten, so lässt sich bei Sprache, Hand- und Kopfgestik eine wesentlich größere Streuung der Übergänge beobachten.

In Ergänzung zu den relativen Übergängen werden pro Versuchsperson $x \in \{1..N_{VP}\}$ die durchschnittlichen Verweildauern für die einzelnen Modalitäten ermittelt. Sei $N^{IMU}_{x,m}$ die Anzahl der identischen Übergänge bezogen auf die Modaliät $m \in \mathcal{M}_{IN}$ mit $N^{IMU}_{x,m} := N^{MU}_{x,m_s,m_t}$ mit $m{=}m_s{=}m_t$. Bezeichnet man im Folgenden mit $N^{SES}_{x,m}$ pro Versuchsperson die Anzahl der abgeschlossenen Interaktionssequenzen unter kontinuierlicher Beteilung der Modalität m, dann ergibt sich die $N^{VD}_{x,m}$ aus dem Quotienten von $N^{IMU}_{x,m}$ und $N^{SES}_{x,m}$:

$$N^{VD}_{x,m} = \frac{N^{IMU}_{x,m}}{N^{SES}_{x,m}} \qquad \forall\, m \in \mathcal{M}_{IN} \tag{4.35}$$

DVA	T_t	A_t	H_t	K_t	S_t
T_s	**72,86**	10,56	2,54	2,68	11,36
A_s	7,04	**79,43**	2,46	1,85	9,22
H_s	4,17	5,83	**54,40**	7,99	27,60
K_s	6,15	9,74	16,91	**35,61**	31,60
S_s	9,60	16,16	21,33	12,23	**40,67**

(a)

DVA	ANF	NOR	EXP	AVP
T	4,14	6,29	5,89	5,83
A	4,77	6,68	12,33	7,06
H	3,48	3,87	2,99	3,53
K	1,08	1,26	1,33	1,21
S	3,11	0,93	1,01	1,32

(b)

Tab. 4.5: (a) Prozentuale Verteilung der relativen Modalitätenübergänge $N^{RMU}_{m_s,m_t}$ im DVA-Szenario von der Ausgangsmodalität m_s auf die Zielmodalität m_t mit $m_s, m_t \in \mathcal{M}_{IN}$, gemittelt über alle Versuchsteilnehmer; identische Übergänge ($m_s \equiv m_t$) sind in Fettdruck gesondert hervorgehoben; (b) durchschnittliche modalitätenspezifische Verweildauer $N^{VD}_{U,m}$ für die einzelnen Eingabemodalitäten $m \in \mathcal{M}_{IN}$ und die verschiedenen Benutzerklassen $U \in \mathcal{U}$

Als Einheit wird für $N^{VD}_{x,m}$ nicht die Zeit, sondern die Anzahl der Interaktionen betrachtet. Ein Wert von $N^{VD}_{m} = 1$ bedeutet, dass die Modalität m im Mittel zweimal direkt hintereinander benutzt wird und der Benutzer dann zu einer anderen Eingabeform wechselt. Tabelle 4.5(b) zeigt die gemittelten Ergebnisse $N^{VD}_{U,m}$ für die jeweiligen Benutzerklassen $U \in \mathcal{U}$. Bei der Betrachtung dieser Werte fällt auf, dass alle Benutzergruppen eine hohe Affinität zur Tastatur aufweisen. Verglichen mit den anderen Modalitäten wird sie am seltesten gewechselt. Von den Anfängern über die Normalnutzer bis hin zu den Experten lässt sich ein signifikanter Anstieg in den gemittelten Verweildauern beobachten. In keiner anderen Eingabemodalität sind die gruppenspezifischen Unterschiede so deutlich. Auffällig ist auch, dass die Gruppe der Anfänger prozentual deutlich länger an der Sprache als Eingabemodalität festhält als die beiden anderen Gruppen.

Die Resultate zu den relativen Modalitätenübergänge und den durchschnittlichen Verweildauern bestätigen die initialen Erwartungen im Vorfeld des Versuchs, dass die Benutzer allgemein in einer rein taktilen Interaktionsart stärker verweilen als bei der Benutzung semantisch höherwertiger Modalitäten. Dieses Verhalten kann zudem für sämtliche Benutzergruppen eindeutig nachgewiesen werden. Speziell nach einer Interaktion mittels Kopfgestik ist in den meisten Fällen ein unmittelbarer Wechsel der Modalität zu beobachten. Etwas entgegen der Erwartungen ist das Ergebnis, dass im Mittel mit über 54% an der Handgestik als Eingabemodalität noch stärker festgehalten wird als mit knapp 41% an der Sprache. Noch deutlicher wird dieses Ergebnis bei der Betrachtung der gemittelten Verweildauern. Mit einem Wert von $N^{VD}_{AVG,H} = 3,5\%$ Interaktionen ist die Verweildauer bei der Handgestik fast dreimal größer als der entsprechende Wert für die Sprache mit $N^{VD}_{AVG,S} = 1,3\%$ Interaktionen.

4.3.4 Zeitrelationen

Neben der prinzipiellen Feststellung der Art und Anzahl der multimodalen Benutzerinteraktionen werden in dieser Benutzerstudie vor allem auch die zeitlichen Relationen zwischen den einzelnen Eingabekomponenten detailliert untersucht. Wie bereits in Abschnitt 4.2.4 beschrieben, wäre das temporale Labeling für alle aufgezeichneten semantisch höherwertigen Interak-

DVA	unimodal E^{UM}			multimodal E^{MM}			allgemein E^{BE}		
	E_H^{UM}	E_K^{UM}	E_S^{UM}	E_H^{MM}	E_K^{MM}	E_S^{MM}	E_H^{BE}	E_K^{BE}	E_S^{BE}
ANF	1,11	1,31	0,98	0,96	1,34	1,01	1,01	1,15	0,98
NOR	0,84	1,14	0,93	0,97	1,17	0,87	0,98	1,15	0,90
EXP	1,28	1,27	1,09	1,32	1,22	1,17	1,32	1,30	1,05
AVP	1,04	1,19	0,98	1,01	1,25	0,96	1,06	1,16	0,96

Tab. 4.6: Durchschnittliche Interaktionsdauern in [sec] der semantisch höherwertigen Modalitäten H, K, S im DVA-Szenario, aufgeteilt nach den einzelnen Benutzerklassen $U \in \mathcal{U}$ und nach unimodalen (E^{UM}), multimodalen (E^{MM}) und allgemeinen Benutzereingabe (E^{BE})

tionen sämtlicher Versuchspersonen deutlich zu zeitintensiv. Daher muss für die Auswertung der Zeitrelationen möglichst repräsentatives Datenmaterial ausgewählt werden. Die wichtigste Einschränkung besteht in einer strikten Konzentration auf den fünften Versuchsblock.

Von den insgesamt 9734 in dieser Phase protokollierten Interaktionen, die von allen 40 Personen aufgezeichnet wurden, sind 1530 semantisch höherwertige Eingaben von zusammen 19 Personen manuell selektiert und zeitlich erfasst worden. Unter den ausgewählten Personen waren sechs Anfänger, zehn Normalnutzer und drei Experten. Bezogen auf die einzelnen Modalitäten wurden 537 Handgesten, 224 Kopfgesten und 769 Spracheingaben analysiert. Diese Eingaben verteilen sich auf 764 unimodale und 355 multimodale Interaktionen. Bei der Auswahl der Interaktionsklassen und Benutzer für die zeitliche Analyse wurde mit extremer Sorgfalt auf die Einhaltung einer möglichst homogenen Struktur geachtet.

Die spezifischen Nutzungsmuster der Eingabemodalitäten variieren bei den Benutzern sehr stark. Für die ausgewählten 19 Versuchspersonen ist beispielhaft die allgemeine Verteilung der Modalitäten in Abbildung A.5 graphisch visualisiert. Während einige der Testpersonen zu fast gleichen Teilen taktile und sprachbasierte Eingabe verwenden, bevorzugen andere fast ausschließlich Touchscreen und Tastatur. Auffällig ist auch die Tatsache, dass einige der VPen eindeutig eine bestimmte Modalität deutlich zu favorisieren scheinen. Detaillierte nutzungsspezifische Untersuchungen haben ergeben, dass in einem solchen Fall die präferierte Modalität exklusiv unimodal verwendet wird.

Interaktionsdauern

Betrachtet man zunächst unabhängig vom konkreten Interaktionsstil die durchschnittliche Dauer der semantisch höherwertigen Interaktionen, so lässt sich fesstellen, dass die einzelnen Zeiten in der Gesamtbetrachtung Werte von etwa einer Sekunde annehmen. Die entsprechenden Werte sind, nach Modalitäten getrennt in der Tabelle 4.6 in den drei rechten Spalten dargestellt. Während die Gruppe der Normalnutzer und der Anfänger sehr nahe am Durchschnitt liegen, weisen die Experten durchgehend etwas höhere Werte auf. Dies liegt vor allem an der verstärkten Neigung der Experten, Kommandos genauer zu präzisieren. Interessanterweise zieht sich dieses Verhalten durch alle Modalitäten. So werden z.B. bei den Sprachkommandos oft ganze Sätze verwendet und auch die Gesten sind meist deutlicher ausgeprägter mit klarer definierten Anfangs- und Endzuständen als bei den beiden anderen Gruppen.

DVA	E^{STA}	E^{SEQ}	E^{SIM}	E^{NES}	E^{NE1}	E^{NE2}	E^{NE3}	E^{NE4}
ANF	37,1	7,0	5,6	50,3	21,1	3,0	10,0	16,2
NOR	69,2	8,4	3,4	18,9	1,8	2,7	8,2	6,2
EXP	46,4	7,1	3,6	42,8	7,1		9,8	25,9
AVP	55,2	7,8	4,2	32,9	9,3	2,5	9,0	12,1

Tab. 4.7: Prozentuale Verteilung der Interaktionsmuster für die reduzierte Benutzermenge $\mathcal{B}^{ZVP}$, aufgeteilt nach den möglichen Überlagerungstypen und nach den jeweiligen Benutzerklassen $U \in \mathcal{U}$

Konzentriert man sich auf die unimodalen Interaktionen (erster vertikaler Block in Tabelle 4.6), so lassen sich nur geringe Abweichungen vom allgemeinen Durchschnitt feststellen. Dies ist zum einen darauf zurückzuführen, dass der Anteil der unimodalen Benutzereingaben etwa 80% aller Systeminteraktionen ausmacht, zum anderen sind auch die Abweichung der multimodalen Interaktionszeiten (mittlerer vertikaler Block in Tabelle 4.6) von den Durchschnittswerten sehr klein. Eine genauere Analyse ergibt beim direkten Zeitenvergleich der unimodalen und der multimodalen Benutzereingaben keine statistisch signifikanten Unterschiede.

Trotz ihres relativ geringen Anteils von nur etwa 20% stehen die multimodalen Interaktionen im Mittelpunkt der Untersuchungen. Da sie sich aus mehreren, modalitätenübergreifenden Benutzereingaben zusammensetzen, ist ihr Anteil bezogen auf die jeweiligen Modalitäten wesentlich höher. Im reduzierten Datenmaterial beträgt der Anteil der multimodalen Benutzereingaben E^{MM}, gemittelt über die beteiligten Personen 56,3%. Mit einer Anzahl von 299 Interaktionen besteht der Hauptanteil aus bimodalen Eingaben E^{BMM}. Die restlichen 56 Eingaben können der Klasse der trimodalen Interaktionen E^{TMM} zugeordnet werden. Da aber nur 6 der 19 hier betrachteten Versuchspersonen überhaupt trimodal interagiert haben, besitzt das Ergebnis nur eingeschränkte Aussagekraft. Hinzu kommt noch, dass über 80% aller trimodalen Benutzerinteraktionen von einer einzigen Person stammen. Aus diesem Grund liegt der Fokus eindeutig auf einer Analyse der bimodalen Eingaben.

Überlagerung

Tabelle 4.7 präzisiert die prozentuale Verteilung der bimodalen Interaktionen für die reduzierte Benutzermenge $\mathcal{B}^{ZVP}$ im Hinblick auf die möglichen, in den Gleichungen 4.25 definierten Überlagerungstypen staggered (E^{STA}), sequential (E^{SEQ}), simultaneous (E^{SIM}) und nested (E^{NES}). Die letzte Klasse wird nach den in 4.27 aufgestellten Beziehungen noch einmal in vier Untergruppen unterteilt. Es ist deutlich zu erkennen, dass es sich bei den meisten Interaktionen um zeitlich überlagerte Benutzereingaben handelt. Bei den Normalnutzern dominiert dieser Typ mit fast 70% ganz klar.

Sequentielle und simultane Interaktionen kommen nur selten vor. Fasst man die vier nested-Kategorien zusammen (N^{NES}), so ist ihr Gesamtanteil vor allem bei der Gruppe der Anfänger und der Experten mit jeweils über 41% ebenfalls von großer Bedeutung. Experten neigen mit einem Anteile von fast 26% an allen bimodalen Benutzereingaben mehrheitlich zu dem Überlagerungstyp E^{NE4}. Dahingegen tendiert die Gruppe der Anfänger mit einem Anteil von über 21% eher zu E^{NE1}.

DVA	Handgestik E^H					Sprache E^S				
	E_H^{STA}	E_H^{SEQ}	E_H^{NEC}	E_H^{NEN}	E_H^{SIM}	E_S^{STA}	E_S^{SEQ}	E_S^{NEC}	E_S^{NEN}	E_S^{SIM}
ANF	1,02	0,85	1,26	0,67	0,88	1,08	0,84	1,21	0,51	0,80
NOR	0,91	0,58	1,43	0,60	0,83	1,11	0,84	1,13	0,67	1,03
EXP	1,37		1,40		1,22	1,10		1,36	0,85	0,96
AVP	1,02	0,75	1,40	0,71	1,00	1,06	0,88	1,29	0,63	1,03

Tab. 4.8: Durchschnittliche Interaktionsdauern in [sec] der verschiedenen Überlagerungstypen am Beispiel der Handgestik (E^H) und der Sprache (E^S), aufgeteilt nach den einzelnen Benutzerklassen

Berechnet man die durchschnittlichen Interaktionsdauern innerhalb dieser Überlagerungsklassen, so lassen sich deutliche Unterschiede feststellen. Die Ergebnisse für kombinierte Eingaben, an denen die Modalitäten Handgestik (E^H) oder Sprache (E^S) beteiligt sind, werden in Tabelle 4.8 dargestellt, die dazugehörigen Werte für Interaktionen mittels Kopfgestik (E^K) sind in den Anhang, Tabelle A.5 ausgelagert. Freigelassene Einträge in der Tabelle zeigen an, dass keine oder nicht ausreichend viele Daten für die jeweilige Kategorie vorhanden sind.

Bei eingebundenen Benutzerinteraktionen E^{NES} ist per Definition die zweite Eingabe $E_{m_2}^{BE}$ kürzer als die erste Interaktion $E_{m_1}^{BE}$. Um im Rahmen der Zeitendiskussionen allgemein zwischen den beiden Eingaben unterscheiden zu können, werden zwei weitere Kategorien eingeführt. Die längere der beiden Interaktionen wird ebenfalls nach Oviatt als *nested-contain* (E^{NEC}) bezeichnet, da sie die Eingabe $E_{m_2}^{BE}$ vollständig enthält. Analog nennt man die kürzere zweite Interaktion *nested-nested* (E^{NEN}) und unterstreicht damit explizit den Zustand der vollständigen Einbindung. Wie die unterschiedlichen Ergebnisse für E^{NEC} und E^{NEN} zeigen, lohnt sich der Aufwand einer getrennten Auswertung. Bei den anderen Interaktionsarten bestehen diese Differenzen hingegen nicht. Allgemein konnte nachgewiesen werden, dass die einzelnen Interaktionszeiten keine signifikante Abhängigkeit von der Reihenfolge der entsprechenden Anfangszeitpunkte aufweisen.

Analysiert man zuerst die Zeiten für alle beteiligten Benutzer AVP, so lassen sich bereits bestimmte Tendenzen identifizieren. Die gemittelten Zeiten der gestaffelten E^{STA} und der simultanen E^{SIM} Eingaben liegen durchgehend gleichauf im Bereich von etwa einer Sekunde. Im Gegensatz dazu weisen die sequentiellen Interaktionen E^{SEQ} im Mittel deutlich kürzere Längen auf. Vergleicht man die beiden Varianten bei eingebundenen Interaktionen, so stellt sich heraus, dass die Zeiten für E^{NEC} in den meisten Fällen ungefähr doppelt so lang sind wie die für E^{NEN}. Zudem liegen beide Zeiten deutlich über bzw. unter dem Durchschnitt. Die in der Gesamtbetrachtung gemachten Beobachtungen lassen sich auch innerhalb der Benutzergruppen bestätigen. Nur in Ausnahmefällen weisen die Zeiten größere Abweichungen auf. Allgemein kann daraus gefolgert werden, dass die Interaktionsdauer eine deutlich erkennbare Abhängigkeit von der verwendeten Überlagerungsart aufweist. Diese Erkenntnis kann vor allem bei Sequenzen von dicht aufeinanderfolgenden multimodalen Interaktionen entscheidend zur Identifikation der einzelnen Systemaktionen beitragen.

Eine weitere wichtige Größe ist die Verzögerungszeit T_{v2} zwischen zwei zusammengehörigen Benutzereingaben. Für E^{STA} liegt diese Zeit bei 14 der 19 Testpersonen im Mittel unterhalb von 0,5s. In diesem Zusammenhang hat sich herausgestellt, dass gerade bei Testpersonen, die

selten multimodal interagieren, klare Abweichungen von diesem Wert zu beobachten sind. In der Gruppe der eingebundenen Interaktionen E^{NES} sind die durchschnittlichen Verzögerungszeiten in den Fällen E^{NE2} und E^{NE3} bereits aufgrund der strukturellen Definition mit ca. 0,6s nachvollziehbar deutlich höher als bei E^{NE1} und E^{NE4} mit ca. 0,15s.

Bezogen auf die sequentiellen Interaktionen E^{SEQ} liegt die Verzögerungszeit T_{V4} zwischen den einzelnen Benutzereingaben im Durchschnitt bei 0,34s. Dabei konnte eine interessante Beobachtung gemacht werden. Kaum einer der Einzelwerte befindet sich in der unmittelbaren Nähe der gemittelten Zeit. Die meisten Werte liegen entweder in einem Bereich zwischen 0,04s und 0,12s oder aber zwischen 0,6s und 1,2s. Bei den zusätzlich erfassten sequentiellen Interaktionen mit Touchscreen oder Tastatur beträgt der zeitliche Abstand T_{V4} ca. 2s. Dieser Wert liegt signifikant über den gemittelten anderen Zeiten. In den hier am häufigsten auftretenden Szenarien ´Maus nach Sprache´ und ´Tastatur nach Sprache´ liegt der Durchschnittwert deutlich über zwei Sekunden, in Ausnahmefällen beträgt T_{V4} bei einzelnen Personen Werte von über sechs Sekunden. Genauere Betrachtungen dieser Fälle haben ergeben, dass die Nutzer bei den haptischen Eingabegeräten wesentlich zurückhaltender agieren als bei Sprache oder Gestik, d.h. sie überlegen beispielsweise deutlich länger, welche Taste im Kontext einer sprachlichen Äußerung zur Vollendung eines Befehls gedrückt werden muss.

Modalitätenreihenfolge

Ebenso wie die prinzipielle Verteilung der unterschiedlichen Überlagerungsarten sind im Rahmen multimodaler Benutzereingaben die konkreten Reihenfolgen der benutzten Modalitäten Gegenstand der Untersuchungen. Abbildung 4.5 veranschaulicht auf der Ordinate die absoluten Häufigkeiten aller zeitlich erfassten bimodalen Interaktionen mit dem Testsystem. Auf der Abszisse sind dazu die bekannten Abkürzungen für die beteiligten Modalitäten in der jeweiligen Reihenfolge aufgelistet. Beispielsweise steht das HS abkürzend für eine multimodale Eingabe, bei der Handgestik vor Sprache eingesetzt wird. Im Sonderfall der simultanen Eingaben werden die Modalitätenkürzel durch ein & getrennt. In dieser Darstellung sind zusätzlich auch die Kombinationen mit den haptischen Eingabegeräten aufgeführt.

Eine genauere Betrachtung der zugrunde liegenden Daten ergibt, dass die Anteile der bimodalen Interaktionen mit der Sprache zusammen ca 74% ausmachen. Besonders bei ausschließlich semantisch höherwertigen Kombinationen (E^{HKS}) ist Sprache bis auf ganz wenige Ausnahmen an allen multimodalen Eingaben beteiligt. Es fällt auf, dass die sprachlichen Anteile mehrheitlich erst an zweiter Stelle auftreten. Insgesamt wird mit einem Anteil von fast 30% an den evaluierten Daten die Kombination von Sprache und Handgestik (HS,SH,S&H) deutlich am häufigsten verwendet. In zwei Drittel dieser Fälle folgt die sprachliche Eingabe zeitlich auf die Handgestik (HS). Äußerst selten werden die beiden Modalitäten simultan verwendet. Der Anteil der rein haptisch kombinierten Interaktionen (TA,AT) macht zusammen über 22% aller multimodalen Benutzereingaben aus. Mit 18% Gesamtanteil wird die Reihenfolge Tastatur und dann Touchscreen (AT) besonders häufig verwendet. Eine typische Eingabe in dieser Katergorie besteht in der Auswahl der Navigationsart über die Tastatur und der Spezifikation der Richtung über den Touchscreen. Allgemein lässt sich bei der Betrachtung der Daten feststellen, dass die Benutzer im Test klare Präferenzen bzgl. der Modalitätenkombinationen haben. Mit einem Anteil von über 90% verteilen sich die bimodalen Benutzereingaben auf sechs der 23 prinzipiell möglichen Eingabemuster.

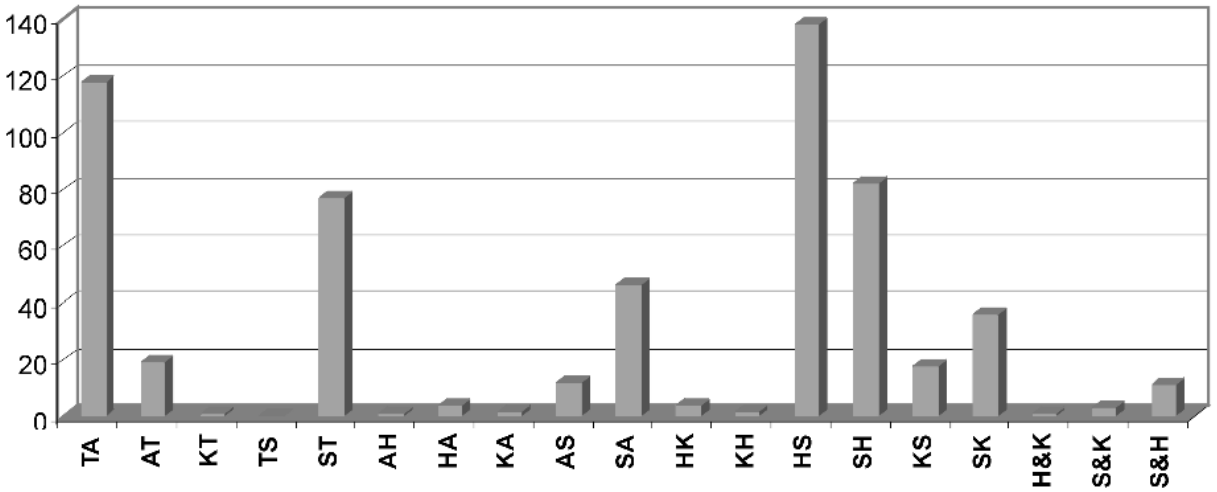


Abb. 4.5: Absolute Häufigkeiten aller zeitlich erfassten bimodalen Interaktionen in der auftretenden Reihenfolge $(m_1 m_2)$ mit $m_1, m_2 \in \mathcal{M}_{IN}$ im DVA-Szenario; die Buchstaben stehen wiederum abkürzend für T=Touchscreen, A=Tastatur, H=Handgestik, K=Kopfgestik und S=Sprache; bei simultanen Interaktionen sind die beiden Modalitätensymbole durch ein & getrennt;

Zu den trimodalen Interaktionen können aufgrund der ungleichmäßigen Datenverteilung keine statistisch relevanten Aussagen gemacht werden. Die Durchschnittszeiten der paarweise untersuchten Interaktionen weichen allerdings nicht signifikant von den allgemeinen Mittelwerten aus Tabelle 4.8 ab. Analysiert man die einzelnen Überlagerungsarten, so lassen sich mit einem Gesamtanteil von über 60% hauptsächlich zeitlich gestaffelte (E^{STA}) Benutzereingaben feststellen. Innerhalb der 56 ausgewerteten trimodalen Modalitätenkombinationen kommen sequentielle Eingaben kaum vor. Sie sind zudem immer gekoppelt mit anderen Überlagerungsarten. Zweifach sequentielle Eingaben mit dedizierten Pausen zwischen den drei beteiligten Modalitäten sind nicht im Datenmaterial zu diesem Benutzertest enthalten.

Zeitabstände zwischen den Ereignisklassen

Ein weiterer wichtiger Punkt im Rahmen der Zeitanalysen ist die Untersuchung der durchschnittlichen Zeitabstände zwischen den einzelnen Ereignisklassen. Aus diesen Daten lassen sich beispielsweise wichtige Informationen über die notwendige Länge möglicher Integrationszeitfenster schließen. Sei $T^{BE}_{x,i,j}$ die absolute Zeitdifferenz zwischen zwei Benutzerereignissen $E^{BE}_{x,i}$ und $E^{BE}_{x,j}$ mit $j > i$ und $i, j \in \{1..N^{BE}\}$ eines spezifischen Benutzers $x \in \{1..N_{VP}\}$. Weiterhin wird angenommen, dass die beiden Eingaben direkt aufeinander folgen. Über die semantische Relation wird nichts ausgesagt, d.h. es kann sich sowohl um getrennte unimodale Interaktionen als auch um eine zusammengehörige multimodale Eingabe handeln. Die durchschnittliche Zeit zwischen den Benutzerinteraktionen wird mit $\bar{T}^{BE}$ bezeichnet.

In direkter Analogie zu dieser Notation bezeichnet $T^{SYS}_{x,k,l}$ mit $k < l$ die Zeitdifferenz zwischen zwei aufeinander folgenden Systemereignissen $E^{SYS}_{x,k}$ und $E^{SYS}_{x,l}$ und $T^{SR}_{x,i,k}$ die Reaktionszeit des Systems, um eine Benutzereingabe $E^{BE}_{x,i}$ in eine entsprechende Systemaktion $E^{SYS}_{x,k}$ umzusetzen. Die Werte $\bar{T}^{SYS}$ und $\bar{T}^{SR}$ geben wiederum die gemittelten personenspezifischen Zeiten an. Als Darstellungseinheit werden Millisekunden verwendet.

DVA	Phase II				Phase III	
Bewertung (1)	TA	TS	SH	SK	ALL	Bewertung (6)
angenehm (SD_1)	2,3 (1,12)	2,5 (1,24)	2,7 (1,58)	2,8 (1,39)	2,1 (1,01)	unangenehm
einfach (SD_2)	2,6 (1,26)	2,7 (1,35)	2,7 (1,63)	2,8 (1,56)	2,4 (1,29)	umständlich
effektiv (SD_3)	2,6 (1,35)	2,6 (1,25)	2,9 (1,48)	2,9 (1,25)	2,3 (1,27)	hinderlich
entlastend (SD_4)	2,9 (1,37)	2,7 (1,17)	3,0 (1,46)	3,1 (1,36)	2,4 (1,03)	belastend
sicher (SD_5)	3,0 (1,01)	3,0 (1,26)	3,1 (1,36)	3,1 (1,39)	2,8 (1,24)	unsicher

Tab. 4.9: Durschnittliche Bewertungen und die dazugehörige Standardabweichungen für die Modalitätenkombinationen (TA, TS, SH, SK) in Phase II des DVA-Versuchs und für die freie Interaktion (ALL) in Phase III anhand von fünf semantischen Differentialen auf einer sechsstufigen Skala

Tabelle 4.14(a) enthält die entsprechenden Werte für die einzelnen Benutzergruppen. Erwartungsgemäß ist die Systemreaktionszeit mit im Mittel 45ms sehr kurz. Bei den Anfängern fällt auf, dass sowohl zwischen den einzelnen Benutzerinteraktionen als auch zwischen den Systemaktionen signifikant mehr Zeit vergehr als bei den beiden anderen Benutzergruppen. Weiterhin fällt auf, dass bei allen Benutzergruppen $\bar{T}^{SYS}$ ungefähr doppelt so groß ist wie $\bar{T}^{BE}$ und dass Normalnutzer die höchste Frequenz an Benutzerereignissen aufweisen.

4.3.5 Subjektive Benutzererfahrungen

Ein weiteres Instrument speziell zur Bewertung der Bedienbarkeit des Testsystems und zur Evaluierung der Benutzerzufriedenheit ist die Analyse der verschiedenen Fragebögen, die von den Versuchsteilnehmern im Vorfeld und nach den einzelnen Versuchsblöcken ausgefüllt werden mussten. Als allgemeines Ergebnis kann vorweggenommen werden, dass die Auswertungen mit den Resultaten der quantitativen Datenanalyse weitgehend übereinstimmen.

In Tabelle 4.9 sind beispielhaft für alle Benutzer die qualitativen Bewertungen für das Testsystem in den einzelnen Phasen des Versuchs anhand von fünf Adjektivpaaren dargestellt. Bezogen auf die vorgeschriebenen Kombinationen wird die taktile Kombination von Touchscreen und Tastatur bis auf SD_4 am besten bewertet. Die Kombination von Sprache und Kopfgestik erhält in allen Kriterien sehr schlechte Bewertungen. Als wichtigstes Ergebnis kann festgestellt werden, dass die freie Kombination der Modalitäten in der Phase III bezogen auf sämtliche Adjektivpaare die mit Abstand besten Bewertungen aufweist.

Eine gruppenspezifische Analyse der persönlichen Modalitätenpräferenzen ergibt trotz hoher individueller Schwankungen und spezifischer Vorlieben für einzelne Interaktionsstile verschiedene generelle Tendenzen. Sowohl Experten als auch Normalnutzer bewerten allgemein die Kombination von Sprache und Touchscreen am besten, wohingegen Anfänger die Kombination von Sprache und Handgestik bevorzugen. Alle Versuchspersonen haben zum Ausdruck gebracht, dass sie am besten mit dem Testsystem umgehen können, wenn sie die Möglichkeit haben, aus den zur Verfügung stehenden Modalitäten frei zu wählen.

Über 96% der Benutzer sind der Meinung, dass die Kombination von unterschiedlichen Modalitäten die generelle Bedienbarkeit des Systems wesentlich verbessert und möchten unter keinen Umständen darauf verzichten. Sämtliche Versuchsteilnehmer finden die Möglichkeit,

gleichzeit mit drei Modalitäten kommunizieren zu können völlig ausreichend. Immerhin noch 84% sind der Meinung, dass auch eine bimodale Kommunikation genügt. Bei der Auswahl der jeweiligen Modalitäten unterscheiden sich die Benutzer jedoch beträchtlich. Dieses Ergebnis bestätigt die quantitativen Werte aus Tabelle 4.3.

Interessanterweise sind Kopfgesten von fast allen Versuchsteilnehmern äußerst schlecht bewertet worden. Dieses Ergebnis steht im klaren Widerspruch zu den quantitativen Auswertungen, bei denen die Kombination von Sprache und Kopfgestik bei der freien multimodalen Interaktion zumindest bei den Experten im Mittel immerhin einen Anteil von fast 30% ausmacht und damit wesentlich häufiger verwendet wurde als die Kombination von Sprache und Handgestik mit einem Anteil von nur 19%. Eine mögliche Erklärung für die negative Bewertung ist die Tatsache, dass Kopfgesten für ausgewählte Funktionen wie beispielsweise das Kippen der Darstellung nach links und rechts (RRO) von vielen VPen unbewusst zur Unterstützung anderer Kommunikationsmöglichkeiten eingesetzt wird.

Bezogen auf die allgemeinen Bemerkungen am Schluss des Versuchs haben viele Teilnehmer explizit den Wunsch nach weiteren Navigationsmöglichkeiten geäußert. Ein häufiger Vorschlag bestand in der Anregung, in einer weiterentwickelten Version des Testsystems einen kontinuierlichen Navigationsmodus zu integrieren, bei dem sich der virtuelle Avatar so lange in eine, beispielsweise per Sprache oder Handgestik vorgegebene Richtung bewegt und auf etwaige Richtungsänderungen reagiert, bis die Aktion vom Benutzer abgebrochen wird. Das Testinterface verfügt wegen der strikt diskret umgesetzten Navigationsart über den Nachteil, dass das ständige Wiederholen einzelner Sprachbefehle, beispielsweise die horizontale Bewegung nach vorne (TVR) in langen Tunnelpassagen, auf den Benutzer sehr ermüdend wirken kann.

Darüber hinaus wurde von den Versuchspersonen angeregt, in der Applikation kontextspezifisches Wissen für die Steuerung des Avatars zu integrieren, um speziell per Sprache oder durch Anzeigen am Bildschirm Objekte aus der virtuellen Szene referenzieren zu können. Mit dieser Möglichkeit wäre eine wesentlich intuitivere und auch einfachere Navigation möglich, da man beispielsweise komplexe Befehle formulieren könnte wie ´Bewege Dich zu dieser Kiste´ oder ´Gehe vor bis zur Wand´.

4.4 Automobilumgebung (AIA)

An der Versuchsreihe zur Evaluierung von Infotainment-Applikationen in der Automobilumgebung (AIA) haben insgesamt $N_{VP} = 28$ Personen teilgenommen. Mit 23 männlichen und nur fünf weiblichen Versuchspersonen ist der Anteil der Männer deutlich größer als in dem DVA-Versuch. Das Durchschnittsalter der Teilnehmer beträgt $\bar{J}_{VP} = 28,6$ Jahre mit einer Standardabweichung von $\delta_J = 7,6$. Bezogen auf die allgemeinen Kenntnisse und Erfahrungen mit Informationssystemen und Eingabegeräten hat die Anfangsbefragung eine a priori Klassifikation in acht Anfänger, 13 Normalnutzer und sieben Experten ergeben. Die prozentualen Verteilungen der Benutzerklassen sind damit in beiden Versuchen sehr ähnlich (siehe auch Tabelle A.7). Als Schnittmenge haben fünf Personen an beiden Versuchsreihen teilgenommen. In begrenztem Umfang ist somit auch ein intrapersoneller Vergleich der Interaktionsmuster in den jeweiligen Domänen möglich.

In Anlehnung an den DVA-Versuch konzentriert sich die folgende Auswertung ebenfalls maßgeblich auf den fünften Versuchsblock mit der freien Modalitätenwahl. Bei der automati-

AIA	PLY	RAD	TEL	LIS	CTL
ANF	18,6	3,9	16,1	31,3	30,1
NOR	19,8	4,3	17,2	36,6	22,1
EXP	20,6	4,2	17,2	33,6	24,5
AVP	20,0	4,2	16,7	34,7	24,3

(a)

AIA	A^{VK}	A^{TK}
ANF	96,3	3,7
NOR	97,6	2,4
EXP	91,8	8,2
AVP	96,5	3,5

(b)

Tab. 4.10: (a) Prozentuale Verteilung der Kommando-Typen bezogen auf die Systemfunktionalitäten des AIA-Testsystems und (b) prozentuale Verteilung der Interaktionen bezogen auf Vollkommandos (A^{VK}) und Teilkommandos (A^{TK}) für die einzelnen Benutzerklassen $U \in \mathcal{U}$

schen Datenwertung und der anschließenden Diskussion der einzelnen Ergebnisse wird absolut identisch vorgegangen, um eine strukturelle Vergleichbarkeit der beiden Domänen zu gewährleisten. An verschiedenen Stellen wird zudem direkt auf die Resultate der DVA-Umgebung Bezug genommen. Zu ausgewählten Werten befinden sich darüber hinaus in Anhang A mehrere Tabellen mit den unmittelbaren Vergleichsdaten. Der Ergebnisteil mit den subjektiven Benutzererfahrungen wird im AIA-Versuch etwas detaillierter behandelt.

4.4.1 Kommandostruktur

Im Vorfeld der Untersuchungen wird zunächst die Struktur der Systeminteraktionen analysiert. Bezogen auf die dritte Versuchsphase sind von sämtlichen Versuchspersonen eine Gesamtanzahl von 1358 Systemkommandos in den Logfiles festgehalten. Pro Versuchsperson x mit $x \in \{1..N_{VP}\}$ variiert die Anzahl der Interaktionen (N_x^{BE}) von 26 zu 80 deutlich weniger als im DVA-Versuch. Die über alle Teilnehmer gemittelte Anzahl der Systeminteraktionen beträgt $\bar{N}^{BE} = 48,5$ mit einer Standardabweichung von $\sigma_{BE} = 9,98$.

Tabelle 4.10(a) zeigt die prozentuale Verteilung der unterschiedlichen, in Abschnitt 3.4.1 eingeführten Kommando-Typen für die Interaktionen mit dem Infotainmentsystem. Die Werte sind jeweils gemittelt für die einzelnen Benutzerklassen $U \in \mathcal{U}$ angegeben. Da die Aufgaben den Versuchspersonen im Test strikt vorgegeben sind, ist erwartungsgemäß die Variationsbreite unter den verschiedenen VPen relativ gering. Lediglich in Bezug auf Kontrollfunktionen (CTL) und Listenmanipulationen (LIS) unterscheiden sich die Benutzer. Dieses Ergebnis resultiert aus der Tatsache, dass beispielsweise die Lautstärke während des Versuchsdurchlaufs von einigen VPen öfter geändert wird oder zum Auffinden eines speziellen Eintrags die komplette Anzeigeliste mehrfach zyklisch durchsucht wird. Insgesamt lässt sich feststellen, dass sich die Benutzer bezogen auf die Verteilung der verwendeten Kommandos im AIA-Szenario weniger stark unterscheiden als in dem DVA-Versuch.

Während sich im DVA-Szenario ein Systemkommando immer aus zwei unterschiedlichen semantischen Komponenten zusammensetzt, gilt dieser Zusammenhang im AIA-Szenario nicht. In dem reduzierten Vokabular des Testsystems lässt sich jede Funktionalität (siehe auch Tabelle 3.2) bereits durch eine einzige, vollständig bedeutungstragende Komponente A^F hinreichend genau spezifizieren. Nur in wenigen Ausnahmesituationen sind zusätzliche Parameter A^P in Form von Zahlenangaben denkbar, jedoch nicht zwingend nötig. Beispielsweise bei der Navi-

gation in einer Listenstruktur kann der Benutzer angeben, dass sich der Listenfokus um eine Zahl $n \in \{1..10\}$ in eine spezielle Richtung bewegen soll. Eine weitere Möglichkeit besteht beim Vor- und Zurückspringen (*Skipping*) einzelner Lieder oder beim Anheben oder Absenken der Lautstärke um diskrete Stufen. Im Unterschied zu Gleichung 4.31 ergibt sich im AIA-Szenario die folgende, erweiterte Beziehung für Vollkommanods (A^{VK}):

$$< A^{VK} > \ ::= < A^{F} > \ | \ \ < A^{F} >< A^{P} > \tag{4.36}$$

$$< A^{TK} > \ ::= < A^{P} > \tag{4.37}$$

Beim Vergleich der Anteile an Vollkommandos (A^{VK}) und Teilkommandos (A^{TK}) in Tabelle 4.10(b) fällt auf, dass die Klasse der A^{VK} die Interaktionen mit dem Testsystem eindeutig dominieren. Dieses Ergebnis lässt sich vollständig mit der bereits oben beschriebenen Struktur der Systemfunktionalitäten erklären. Obwohl in der Trainingsphase alle Benutzer in gleichem Maße auf die prinzipielle Möglichkeit der Parameterspezifikation hingewiesen worden sind, haben lediglich die Experten davon verstärkten Gebrauch gemacht. Anfänger und Normalnutzer verzichten weitgehend auf eine zusätzliche Angabe von Zahlenparametern.

4.4.2 Modalitätennutzung

Die Grundlage für eine Diskussion der Eingabesmuster bildet eine detaillierte Analyse der unimodalen und multimodalen Interaktionsanteile. Von den ingesamt 1358 in Phase III protokollierten Ereignissen können bezogen auf alle Versuchspersonen bei 28 Interaktionen multimodale Benutzereingaben beobachtet werden. Das entspricht lediglich einem Anteil von ca. 2,1%. Dieses Ergebnis widerspricht klar den initialen Ewartungen, ist aber zum einen ebenfalls eine logische Konsequenz aus der prinzipiellen Kommandostruktur und wird zum anderen noch durch den unterschiedlichen Versuchsaufbau in der Automobilumgebung beeinflusst. Da die Versuchspersonen durch die parallel ablaufende Fahraufgabe kognitiv zusätzlich gefordert sind, können sie nur einen gewissen Anteil ihrere Aufmerksamkeit für die Bedienung des Testsystems verwenden.

Die Bedienaufgaben sind in einer Weise aufgebaut, dass mit einfach strukturierten Interaktionen die gewünschte Funktionalität unmittelbar ausgelöst werden kann. Im Rahmen dieser Benutzerstudie werden dazu durchgängig von allen Benutzergruppen unimodale Eingaben klar präferiert. Nur 13 der 28 Benutzer nutzen überhaupt die Möglichkeit einer kombinierten Eingabe. Bei acht VPen aus dieser Gruppe kann jeweils nur eine einzige multimodale Interaktion beobachtet werden. Drei VP setzen zwei multimodale Interaktionen ein und nur zwei Personen im Test kombinieren die Eingabemodalitäten häufiger.

Modalitätenverteilung und Kombinationen

Die prozentuale Verteilung der unimodalen Systeminteraktionen E^{UM} auf die einzelnen Modalitäten E^{UM}_{m} mit $m \in \mathcal{M}_{IN}$ ist in Tabelle 4.11(a) dargestellt. Bezogen auf die taktilen Interaktionen wird die Tastenkonsole sowohl von den Anfängern als auch von den Normalnutzern stärker frequentiert als der Touchscreen. Der Unterschied ist speziell bei den Normalnutzern mit etwas über 2% sehr gering. Experten bevorzugen den Touchscreen mit geringem Abstand vor dem Tastenblock. Sprache wird im AIA-Szenario von allen Benutzergruppen wesentlich häufiger eingesetzt als im DVA-Versuch. Bei den Anfängern stellt sie mit einem Anteil von fast

AIA	E_T^{UM}	E_A^{UM}	E_H^{UM}	E_K^{UM}	E_S^{UM}
ANF	24,9	31,1	3,5	1,0	39,5
NOR	38,7	41,1		0,1	20,0
EXP	36,2	34,9	4,8	0,4	23,6
AVP	31,9	34,4	2,8	0,9	30,0

(a)

AIA	E_T^{MM}	E_A^{MM}	E_H^{MM}	E_K^{MM}	E_S^{MM}
ANF			5	3	8
NOR			6	2	8
EXP	1	1	9	1	12
AVP	1	1	20	6	28

(b)

Tab. 4.11: (a) Prozentuale Verteilung der unimodalen (E_m^{UM}) und (b) absolute Verteilung der multimodalen (E_m^{MM}) Benutzereingaben jeweils mit der Modalität $m \in \mathcal{M}_{IN}$ im AIA-Szenario für die dritte Versuchsphase mit der freien Interaktion, aufgeteilt nach den unterschiedlichen Benutzergruppen

40% die primäre Interaktionsform dar. Für Experten und Normalnutzer ist Sprache dagegen erst die dritte Wahl. Die beiden anderen semantisch höherwertigen Modalitäten Hand- und Kopfgestik kommen eher wenig zum Einsatz. Sie sind aber auch vom prinzipiellen Design des Interfaces her nicht für jede Systemfunktionalität in gleichem Maße geeignet (siehe Tabelle 3.2). Für ausgewählte Kommandos, beispielsweise zur Aktivierung der einzelnen Betriebmodi, ist eine Eingabe per Gestik gar nicht möglich. Obwohl die Versuchspersonen alle fünf Modalitäten frei kombinieren können, zeigt eine detaillierte Analyse des Datenmaterials eine Beschränkung auf zumeist zwei Modalitäten pro Person. Im Mittel sind dies Sprache und Touchscreen.

Da der Anteil der multimodalen Benutzereingaben extrem klein ist, wird in Tabelle 4.11(b) anstatt der sonst üblichen relativen Verteilung die absolute Verteilung der benutzten Modalitäten angegeben. Als erstes fällt auf, dass die haptischen Bedienelemente bei kombinierten Interaktionen im AIA-Szenario kaum verwendet werden. Lediglich eine VP hat jeweils einmal den Touchscreen und den Tastenblock mit der Sprache kombiniert. In beiden Fällen wurde per taktiler Eingabe ein Skip-Befehl initiiert und zusätzlich per Sprache eine Zahlenangabe gemacht. Ansonsten werden ausschließlich semantisch höherwertige Modalitäten untereinander kombiniert. Sprache ist im Kontext der multimodalen Benutzereingaben im Automobil an jeder Interaktion beteiligt und stellt damit zumindest für diese Benutzerstudie eine unverzichtbare Eingabemöglichkeit dar. Handgestik ist im Kontext der Lautstärkeänderung wichtig, Kopfgestik wird statt dessen nur für Skip-Befehle und für Telefonfunktionen mit der Sprache kombiniert.

Bezogen auf die funktionale Zusammensetzung multimodaler Interaktionen können bei 24 der 28 kombinierten Benutzerinteraktionen redundante Eingaben (E^{RED}) beobachtet werden. Ein typisches Beispiel für eine solche Eingabe ist eine co-verbale Untermalung einer Spracheingabe mittels entsprechender Hand- oder Kopfgestik. So wird beispielsweise beim Anheben der Lautstärke neben dem eigentlichen, an sich logisch vollständigen Sprachbefehl zusätzlich noch die Hand gehoben oder bei der Ablehnung eines Telefonanrufs zusätzlich der Kopf geschüttelt. Rivalisierende Eingaben (E^{RIV}) sind bei den Benutzern im Test nicht aufgetreten.

Nur bei vier kombinierten Interaktionen ergänzen sich die Informationsanteile der unterschiedlichen Modalitäten. Komplementäre Eingaben (E^{CMP}) stellen damit im AIA-Szenario nur eine untergeordnete Interaktionsform dar. Sie beziehen sich ausschließlich auf eine Spezifikation der Funktionalität durch Handgestik und die zusätzliche Angabe eines Zahlenparameters über die Spache. Im Sinne der in Abschnitt 4.2.3 eingeführten Notation handelt es sich damit um harte Komplementarität (E^{HAC}). Kombinationen der funktionalen Klassen (E^{REC}, E^{RIC}) treten im AIA-Szenario nicht auf.

AIA	T_t	A_t	H_t	K_t	S_t
T_s	**44,94**	17,72	5,70	1,90	29,75
A_s	15,41	**64,78**	4,09	2,20	13,52
H_s	22,67	24,02	**8,07**	0	45,33
K_s	13,64	9,09	13,64	**0**	63,64
S_s	16,69	7,18	5,84	1,34	**68,95**

(a)

AIA	ANF	NOR	EXP	AVP
T	0,75	0,76	0,83	0,81
A	1,67	1,60	1,88	1,77
H	0,50	0,71	0	0,61
K	0	0	0	0
S	6,68	2,43	2,01	3,58

(b)

Tab. 4.12: (a) Prozentuale Verteilung der relativen Modalitätenübergänge $N^{RMU}_{m_s,m_t}$ im AIA-Szenario von der Ausgangsmodalität m_s auf die Zielmodalität m_t mit $m_s, m_t \in \{\mathcal{M}_{IN}\}$, gemittelt über alle Versuchsteilnehmer; Identitätsübergänge ($m_s \equiv m_t$) sind gesondert hervorgehoben; (b) gemittelte modalitätenspezifische Verweildauer $N^{VD}_{U,m}$ für die einzelnen Eingabemodalitäten $m \in \mathcal{M}_{IN}$ und Benutzergruppen $U \in \mathcal{U}$

Modalitätenübergänge

Eine Analyse der Modalitätenübergänge offenbart deutliche Unterschiede zu den Ergebnissen in dem DVA-Versuch. Tabelle 4.12(a) zeigt im Detail die über alle Versuchspersonen gemittelten relativen Modalitätenübergänge ($N^{RMU}_{m_s,m_t}$) von der Ausgangsmodallität m_s auf die Zielmodalität m_t mit $m_s, m_t \in \mathcal{M}_{IN}$. Identische Übergänge stellen nur bei der Sprache und bei den haptischen Interaktionen den größten Anteil dar. Im Fall einer Eingabe mittels Hand- oder Kopfgestik wird in der nachfolgenden Interaktion meistens die Sprache benutzt, interessanterweise aber niemals die Kopfgestik. Entgegen den initialen Erwartungen ist vor allem die niedrige Übergangsquote bei der Handgestik. Anstatt hintereinander mehrere Eingaben mit dieser Modalität zu machen wird in mehr als neun von zehn Fällen für die nächste Eingabe eine andere Modalität benutzt.

Bei der Betrachtung der durchschnittlichen gruppenspezifischen Verweildauern N^{VD}_U mit $U \in \mathcal{U}$ in Tabelle 4.12(b) fällt zunächst auf, dass sämtliche Benutzerklassen am längsten an der Sprache als Eingabemodalität festhalten. Besonders deutlich zeigt sich dieses Verhalten für die Gruppe der Anfänger, ide mit einem Wert von 6,68 Interaktionen im Durchschnitt dreimal länger bei der Spracheingabe bleiben als die Experten mit einem Wert von 2,01 Interaktionen. Mit Werten zwischen 1,6 und 1,88 Interaktionen wird auch die Tastenkonsole von allen Benutzern häufig mehrfach hintereinander genutzt. Handgestik und Touchscreen werden dagegen mit Werten unter eins wesentlich häufiger nach jeder Eingabe gewechselt. Keine der Versuchspersonen im Test hat zweimal direkt hintereinander eine Eingabe mittels Kopfgestik gemacht.

Analysiert man die Modalitätenübergänge nicht nur im lokalen Kontext zweier direkt aufeinander folgenden Benutzereingaben sondern im Rahmen ganzer Aufgaben, so lässt sich feststellen, dass die einzelnen Modalitäten häufig mit wechselnden komplementären Anteilen benutzt werden. Beispielsweise drückt eine VP in der ersten Interaktion am Touchscreen auf den Button für die Auswahl des Radiomodus und sagt dann in der nächsten Interaktion 'weiter' oder 'nächster', um von Sender zu Sender zu skippen. Wechselnd komplementär bedeutet in diesem globalen Zusammenhang, dass die Art und Weise, wie sich diese Informationsanteile zusammensetzen, unter den einzelnen Versuchsteilnehmern stark variiert. Für die Modusauswahl und den anschließenden Befehl werden verschiedene Strategien gewählt und es lässt sich kein eindeutiges, prototypisches Benutzerverhalten nachweisen.

AIA	unimodal E^{UM}			multimodal E^{MM}			allgemein E^{BE}		
	E_H^{UM}	E_K^{UM}	E_S^{UM}	E_H^{MM}	E_K^{MM}	E_S^{MM}	E_H^{BE}	E_K^{BE}	E_S^{BE}
ANF	1,11	1,31	0,98	0,96	1,34	1,01	1,01	1,15	0,98
NOR	0,84	1,14	0,93	0,97	1,17	0,87	0,98	1,15	0,90
EXP	1,28	1,27	1,09	1,32	1,22	1,17	1,32	1,30	1,05
AVP	1,04	1,19	0,98	1,01	1,25	0,96	1,06	1,16	0,96

Tab. 4.13: Durchschnittliche Interaktionsdauern in [sec] der semantisch höherwertigen Modalitäten H, K, S im AIA-Szenario, aufgeteilt nach den einzelnen Benutzerklassen $U \in \mathcal{U}$ und nach unimodalen (E^{UM}), multimodalen (E^{MM}) und allgemeinen Benutzereingabe (E^{BE})

4.4.3 Zeitrelationen

In der Automobilumgebung wurde für die Diskussion der Interaktionslängen und Zeitrelationen ebenfalls repräsentatives Datenmaterial ausgewählt und wie in Abschnitt 4.2.4 erläutert manuel annotiert. Von den insgesamt 1358 in der Phase III protokollierten Interaktionen sind 466 Interaktionen mit den semantisch höherwertigen Eingabemodalitäten von 16 ausgewählten Personen zeitlich erfasst worden. Unter diesen Versuchspersonen befinden sich sechs Anfänger, sechs Normalnutzer und vier Experten. Bezogen auf die einzelnen Modalitäten wurden 64 Handgesten, 21 Kopfgesten und 381 Spracheingaben analysiert. Die Interaktionen verteilen sich auf 412 unimodale und 27 multimodale Eingaben. Bei der Auswahl der Personen und unimodalen Interaktionen stand wiederum eine möglichst homogene Struktur der Daten im Vordergrund. Da die Datenmenge für multimodale Interaktionen ohnehin schon stark beschränkt ist, wurden in diesem Bereich sämtliche Kombinationen untersucht.

Die allgemeinen Interaktionszeiten liegen mit Ausnahme der Sprache leicht höher als in dem DVA-Versuch. Tabelle 4.13 zeigt die Interaktionsdauern in Sekunden für die einzelnen Benutzergruppen $U \in \mathcal{U}$ und die Modalitäten $m \in \{H, K, S\}$. Aufgrund der klaren Dominanz der unimodalen Eingaben sich die entsprechenden Werte nur unwesentlich von den allgemeinen Zeiten. Die bimodalen Zeiten liegen alle unter dem Durchschnitt. Bezeichnenderweise handelt es sich hier auch maßgeblich um rein redundante Interaktionen. Bei insgesamt nur 27 multimodalen Eingaben dienen speziell die Werte für E^{MM} lediglich als Anhaltspunkte, denen nur eine begrenzte statistische Relevanz zugeordnet werden kann. Auch im AIA-Szenario fällt auf, dass die Experten gerade bei Sprachinteraktionen die längsten Interaktionszeiten aufweisen. Analog zu den Ergebnissen im DVA-Versuch lässt sich dies auf eine höhere Verbosität zurückführen. Experten benutzen mehr Worte für ein Kommando als die beiden anderen Gruppen.

Die meisten kombinierten Interaktionen sind gestaffelt (E^{STA}) oder eingebunden (E^{NES}) überlagert. Analysiert man den Typ der Einbindung (siehe Abbildung 4.3), so stellt sich heraus, dass E^{NE3} und E^{NE4} jeweils zu gleichen Anteilen vorkommen und die anderen beiden Arten nicht auftreten. Sequentielle Interaktionen kommen nur bei vier der 27 multimodalen Interaktionen vor. Die durchschnittliche Verzögerungszeit zwischen den beiden Eingaben (T_{v4}) ist wesentlich größer als der vergleichbare Wert im DVA-Versuch. Simulate Eingaben (E^{SIM}) waren im Testmaterial nicht enthalten. Bezogen auf die Reihenfolge der einzelnen Modalitäten hat sich ergeben, dass bei allen Kombinationen von Sprache mit Hand- oder Kopfgestik die Sprache zeitlich erst an zweiter Stelle benutzt wird.

DVA	ANF	NOR	EXP	AVP
$\bar{T}^{BE}$	1214	823	1185	1006
$\bar{T}^{SYS}$	2454	1684	2078	1968
$\bar{T}^{SR}$	43	44	46	45

(a)

AIA	ANF	NOR	EXP	AVP
$\bar{T}^{BE}$	2074	1659	1426	1720
$\bar{T}^{SYS}$	4167	2956	3957	3353
$\bar{T}^{SR}$	32	31	26	29

(b)

Tab. 4.14: Durchschnittliche Zeiten in [ms] zwischen zwei Benutzerereignissen ($\bar{T}^{BE}$), zwei Systemereignissen ($\bar{T}^{SYS}$) und die durchschnittliche Reaktionszeit des Systems auf eine Benutzereingabe ($\bar{T}^{SR}$) für (a) das DVA-Szenario und (b) das AIA-Szenario aufgeteilt nach den Benutzerklassen $U \in \mathcal{U}$

Bei der Analyse der durchschnittlichen Zeiten zwischen den einzelnen Benutzerereignissen ($\bar{T}^{BE}$) und den Systemereignissen ($\bar{T}^{SYS}$) fällt unmittelbar auf, dass beide Zeiten signifikant länger sind als die vergleichbaren Werte im DVA-Versuch. Tabelle 4.14 enthält eine direkte Gegenüberstellung der einzelnen Zeiten in den beiden Domänen. In den Ergebnissen spiegelt sich der unterschiedliche Versuchsaufbau. Obwohl die Bedienaufgaben im AIA-Versuch vom Anspruch her einfacher sind, folgen die Interaktionen mit dem Testsystem nicht so dicht aufeinander, da die Versuchspersonen gleichzeitig auch mit der Fahraufgabe beschäftigt sind. Erwartungsgemäß ist die Reaktionszeiten des Systems relativ konstant und mit ca. 30ms sogar noch etwas kürzer als im DVA-Versuch. Die Tendenz, dass $\bar{T}^{SYS}$ im Mittel ungefähr doppelt so groß ist wie $\bar{T}^{BE}$ bestätigt sich auch in der Automobilumgebung.

4.4.4 Subjektive Benutzererfahrungen

Im AIA-Versuch haben die Versuchsteilnehmer nach den einzelnen Phasen das Testinterface nach denselben Kriterien bewertet, die auch im DVA-Versuch verwendet wurden. Tabelle 4.15 zeigt, bezogen auf alle Benutzer die Mittelwerte und die Standardabweichungen für die fünf semantischen Differentiale. Im Folgenden werden die Bewertungen im Kontext der Modalitäten, die in den einzelnen Versuchsblöcken zur Verfügung stehen, diskutiert. Dabei wird in Ergänzung zu dem vorherigen Abschnitt auch auf kombinierte Eingaben in der Phase II eingegangen.

Die Bedienung mit den Modalitäten Touchscreen und Tastenblock (K_{TA}) in Block I des Versuchs wird als relativ einfach, angenehm und effektiv bewertet. Bei der Gegenüberstellung der Adjektivpaare SD_4 (entlastend vs. belastend) und SD_5 (sicher vs. unsicher) können die meisten Versuchspersonen keine eindeutige Entscheidung treffen. Dies ist hauptsächlich darauf zurückzuführen, dass die Bedienung mit dem Tastenblock als ablenkend und unsicher eingestuft wird. Der Blick ist dabei zu weit von der Fahrbahn abgelenkt, was sich durch eine nachträgliche Analyse des Videomaterials bestätigen lässt (siehe dazu auch [14, 93]). Zudem wird die mangelnde Ergonomie im Design und auch in der Position des einfachen Bedienelementes kritisiert. Auf der anderen Seite haben die Versuchsteilnehmer mehrheitlich herausgestellt, dass der Tastenblock nach einer Gewöhnungsphase absolut sicher bedient werden kann. Eine direkte Kombination der Modalitäten Sprache und Tastenblock konnte nicht beobachtet werden, eine isolierte Aufgabe wird prinzipiell immer mit einer Modalität gelöst.

Bei dem Einsatz von Touchscreen und Sprache (K_{TS}) im zweiten Testblock sind sämtliche Bewertungen eindeutig besser ausgefallen. Speziell bei den Adjektivpaaren SD_2 und SD_3 ist

AIA	Phase II				Phase III	
Bewertung (1)	TA	TS	SH	SK	ALL	Bewertung (6)
angenehm (SD_1)	2,5 (1,04)	1,9 (0,93)	2,4 (1,23)	2,7 (1,08)	1,4 (0,69)	unangenehm
einfach (SD_2)	2,6 (1,20)	1,8 (0,84)	2,6 (1,29)	2,7 (1,20)	1,5 (0,64)	umständlich
effektiv (SD_3)	2,5 (1,23)	1,7 (0,82)	2,5 (1,45)	2,9 (1,18)	1,7 (0,85)	hinderlich
entlastend (SD_4)	3,4 (1,23)	2,2 (1,20)	2,4 (1,29)	2,6 (1,39)	1,8 (0,77)	belastend
sicher (SD_5)	3,6 (1,37)	2,1 (1,07)	2,2 (1,23)	2,3 (0,94)	1,9 (0,94)	unsicher

Tab. 4.15: Durschnittliche Bewertungen und die dazugehörigen Standardabweichungen für die Modalitätenkombinationen (TA, TS, SH, SK) in Phase II des AIA-Versuchs und für die freie Interaktion (ALL) in Phase III anhand von fünf semantischen Differentialen auf einer sechsstufigen Skala

auch die Streuung unter den Versuchspersonen wesentlich geringer. Dieses Ergebnis ist allerdings weniger auf die Kombination der beiden Modalitäten zurückzuführen als vielmehr auf die Tatsache, dass zum ersten Mal eine Sprachsteuerung benutzt werden konnte. Ein großer Teil der Versuchspersonen nutzt die Möglichkeit der Spracheingaben, um komplexe, mehrstufige Kommandos direkt durch eine Eingabe ausdrücken zu können. Die Bedienung per Sprache ermöglicht einen uneingeschränkten Blick auf die Fahrbahn. Der Touchscreen als zusätzliche Kombinationsmöglichkeit wird in dieser Versuchsphase häufig gar nicht benutzt. Teilweise werden die Modalitäten im globalen Kontext komplementär benutzt, d.h. beispielsweise Anwahl eines speziellen Eintrags in der Telefonliste per Sprache und die anschließende Bestätigung der Wahl mittels Touchscreen.

Auch im dritten Versuchsblock, der Kombination von Sprache und Handgestik (K_{HS}), wird ein grosser Teil der Bedienaufgaben ausschließlich mit der Sprache durchgeführt. Die Bewertungen bewegen sich allgemein im Rahmen der Ergebnisse von K_{TA}, nur bezogen auf die SD_4 und SD_5 erhält K_{HS} eine wesentlich bessere Beurteilung. Handgesten werden gezielt für einfache Richtungsangaben eingesetzt: beispielsweise eine horizontale Winkbewegung zum Springen in der Wiedergabeliste oder eine vertikale Handbewegung zum Ändern der Lautstärke. Generell kann eine deutliche Vergrößerung der Streuung beobachtet werden. Dieses Ergebnis ist auf die unterschiedlichen Haltungen der Versuchspersonen gegenüber der Versuchsdurchführung zurückzuführen. Obwohl die Testpersonen möglichst intuitiv interagieren sollen, gibt es eine Gruppe von Benutzern, die einen großen Teil ihrer Aufmerksamkeit und Konzentration auf die Steuerung per Gestik verwendet haben. Diese werten die Bedienung mehrheitlich als umständlicher und belastender im Vergleich zu der Kombination von Sprache und Touchscreen. Der andere Teil der Versuchspersonen, bei denen Handgestik im Mittel deutlich weniger eingesetzt wird, empfinden hingegen die Bedienung als sehr einfach und sicher. Bezogen auf direkte Kombinationen werden Handgesten vereinzelt redundant zur Sprache eingesetzt, um gesprochenen Kommandos Nachdruck zu verleihen.

Der Einsatz von Sprache und Kopfgestik im vierten Versuchsblock (K_{KS}) wird häufig als umständlicher und weniger effektiv als die anderen Kombinationen beschrieben. Die Bewertungen sind daher auch durchgängig schlechter als in den anderen Blöcken. Für ausgewählte Funktionen, beispielsweise zum Annehmen oder Ablehnen eines eingehenden Anrufs, werden aber von einem Viertel der Versuchspersonen auschließlich Kopfgesten verwendet. Als sehr gut

wird in diesem Zusammenhang die Tatsache bewertet, dass mittels Kopfgesten einfache Eingaben gemacht werden können, ohne die Aufmerksamkeit vom Fahren abzulenken.

Verglichen mit den Ergebnissen aus der zweiten Versuchsphase mit den fest vorgeschriebenen Modalitäten wird die Bedienung des Systems in der dritten Phase bei freier Modalitätenwahl in fast allen Punkten signifikant besser bewertet. Lediglich die Kombination Sprache und Touchscreen erhält, bezogen auf das Adjektivpaar SD_3 die gleiche Bewertung. Auch die Streuung der Ergebnisse ist wesentlich geringer. Im direkten Vergleich mit den Ergebnissen aus dem DVA-Versuch erhält die freie Interaktion ebenfalls wesentlich bessere Beurteilungen.

Der Nutzen der Multimodalität im Automobilbereich liegt offensichtlich nicht in der Option, Befehle aus unterschiedlichen Modalitäten direkt miteinander zu kombinieren. Im Gespräch mit den Versuchspersonen hat sich herausgestellt, dass die vielfachen Eingabemöglichkeiten durchaus positiv bewertet werden. Vor allem der Touchscreen und die natürlichsprachliche Bedienung werden sehr geschätzt. Auch das kooperative und tolerante Verhalten des Versuchsleiters dürfte ein Grund für die Bevorzugung der Unimodalität sein. In anderen Arbeiten im Rahmen des FERMUS-Projektes hat sich herausgestellt, dass bei einer simulierten Erkennung mit partiell eingestreuten Fehlklassifikationen sowohl die Modalitäten viel stärker gewechselt werden als auch die Anzahl der redundanten Interaktionen ansteigen [93, 97].

Abschließend müssen die Benutzer die Eignung der einzelnen Modalitäten in Abhängigkeit von den zu erfüllenden Aufgaben bewerten. Dazu werden nach dem sog. forced-rating Verfahren [113] jeder Funktionalität explizite Modalitätenpräferenzen zugeordnet, d.h. die Modalität, die sich nach Meinung der Versuchsperson am besten für eine Funktionalität eignet erhält Platz eins, die zweite kommt auf Platz zwei usw. bis zu der Modalität, die sich am wenigsten eignet und Platz fünf einnimmt. Dabei dürfen einzelne Plätze nicht mehrfach vergeben werden. Tabelle A.6 zeigt die über alle Versuchspersonen gemittelten Präferenzen und die dazugehörigen Standardabweichungen.

Die Sprache als Eingabemodalität ist mit Abstand die am stärksten präferierte Modalität und wird prinzipiell für alle Funktionalitäten eingesetzt. Touchscreen wird sehr häufig für die Modus-Auswahl oder das Skippen verwendet. Obwohl man auch mit dem separaten Tastenblock das Interface vollständig bedienen kann, werden hauptsächlich die beiden Dreh-Drucksteller benutzt, um die Lautstärke zu regulieren und den aktuellen Eintrag in der Auswahlliste zu ändern. Handgesten werden neben dem Touchscreen ebenfalls zum Skippen benutzt und vereinzelt, um das Annehmen oder Ablehnen eines Telefonanrufs zu signalisieren. Einen besonders interessanten Fall bieten die Kopfgesten. Allgemein werden sie von der Mehrheit der Versuchspersonen stark abgelehnt. Ein paar VPen (8 von 28) haben allerdings für die Funktionalität ´Anruf ablehnen´ der Eingabemöglichkeit mittels Kopfgestik Platz eins zugeordnet. Dies macht sich auch in der Statistik bemerkbar, bei der im Schnitt die Bewertungen für die Kopfgestik bezogen auf diese Funktionalität signifikant besser sind als bei den anderen Funktionalitäten.

Sehr deutlich im AIA-Versuch ist die extensive Nutzung der Sprache. Es wird von dem System erwartet, dass es in der Lage ist, auch über Kontextwissen aus der Anwendungsdomäne zu verfügen. Dazu gehört, dass die Applikation Informationen über die aktuell erreichbaren Radiosender besitzt und diese auch direkt über den Sendernamen in Sprachkommandos referenziert werden können. Viele Versuchspersonen haben zudem ein intelligentes System erwartet, welches aus dem erteilten Kommando selbst den zugehörigen Modus erkennt. Obwohl die korrekte Funktionsweise des Systems zu Beginn der Versuche extensiv erklärt und trainiert wurde, haben viele VPen direkt vorausgesetzt, dass das System bei dem Kommando 'Wähle Nummer xxx' selbständig einen impliziten Moduswechsel aus dem Audio- in den Telefonmodus vornimmt.

4.5 Zusammenfassender Domänenvergleich

Trotz der strukturellen Gemeinsamkeiten bei der Konzeption der Versuche ist ein direkter Vergleich der Ergebnisse aus den beiden Testdomänen Desktop-Virtual-Reality Applikation (DVA) und automotive Infotainment Applikation (AIA) wegen der unterschiedlichen Versuchsaufbauten und den damit verbundenen Rahmenbedingungen nur begrenzt möglich. In dem DVA-Szenario kann sich der Benutzer vollständig auf die Navigation in den virtuellen Welten konzentrieren. Dagegen müssen die Versuchspersonen im AIA-Szenario neben der Bedienaufgabe, dem Steuern von verschiedenen Infotainment-Anwendungen, gleichzeitig noch eine Fahraufgabe erfüllen. Die prinzipielle Struktur der Systemaktionen ist ebenfalls verschieden. Während sich in dem DVA-Versuch eine Aktion immer aus zwei Komponenten zusammensetzt (Bewegungsart und Richtung), bestehen in dem AIA-Versuch alle Aktionen nur aus einer einzigen Komponente, die optional noch durch Zahlenparameter ergänzt werden können.

Die Zielapplikation beeinflusst die Wahl der Eingabemodalitäten. Gestik eignet sich beispielsweise sehr gut zur Beschreibung von räumlichen Zusammenhängen. In der zwischenmenschlichen Kommunikation wird sie dafür oft eingesetzt und daher auch von den Benutzern im DVA-Versuch intuitiv benutzt. Zur Steuerung von Audio- und Kommunikationsgeräten im Automobil muss die Anwendung der Gestik erst motiviert und erlernt werden. Aus der Nutzung automotiver Endgeräte im privaten Bereich haben die Benutzer bereits bestimmte Gewohnheiten angenommen, die auf das Testsystem übertragen werden und zu einer verstärkten Nutzung der haptischen Eingabegeräte führen. Dennoch wird in diesem Abschnitt domänenübergreifend auf verschiedene Charakteristika multimodaler Interaktionsmuster Bezug genommen.

Die generelle Verteilung der einzelnen Modalitäten für beide Domänen wird, jeweils gemittelt über sämtliche Versuchspersonen, in Abbildung 4.6 veranschaulicht. In dieser Darstellung wird nicht zwischen unimodalen und multimodalen Interaktionen unterschieden. Pro Eingabemodalität $m \in \mathcal{M}_{IN} = \{T, A, H, K, S\}$ repräsentiert die linke Säule die prozentualen Anteile in dem DVA-Versuch und die rechte Säule entsprechend die Anteile in dem AIA-Versuch. Die einzelnen Buchstaben stehen jeweils abkürzend für eine Eingabemodalität mit T=Touchscreen, A=Tastatur, H=Handgestik, K=Kopfgestik und S=Sprache.

Im Automobilbereich dominiert eindeutig die Sprache, wohingegen im DVA-Versuch die taktilen Interaktionen mittels Touschscreen und Tastatur die beiden größten Anteile an allen Eingaben besitzen. Fasst man die Anteile der taktilen Eingaben und die der Sprache zu einem Block zusammen, so ergibt sich in beiden Domänen ein ähnlicher Wert. Handgestik und Kopfgestik kommen im Auto nur selten zum Einsatz, zur Navigation in virtuellen Welten werden sie demgegenüber sehr gut von den Benutzern angenommen. Das Verhältnis der semantisch höherwertigen Modalitäten ist dadurch im DVA-Versuch wesentlich ausgeglichener.

Diese Ergebnisse stellen keine allgemein gültigen Aussagen dar, sondern müssen vor dem Hintergrund der entsprechenden Versuchsaufbauten sorgfältig interpretiert werden. Bei den von Geiger [50] durchgeführten Benutzerstudien kommt beispielsweise der Gestik auch im Automobil eine weitaus größere Bedeutung zu. Anders als der in Abschnitt 3.4.1 vorgestellte Prototyp ist sein graphisches Benutzer-Interface speziell für die Bedienung mit Gesten optimiert.

In beiden Domänen hat sich gezeigt, dass Experten dazu tendieren, die haptischen Eingabegeräte zu bevorzugen. Anfänger und Normalnutzer sind in der Wahl ihrer Eingabemöglichkeiten etwas variabler. Sprache wird am häufigsten von der Gruppe der Anfänger verwendet. Betrach-

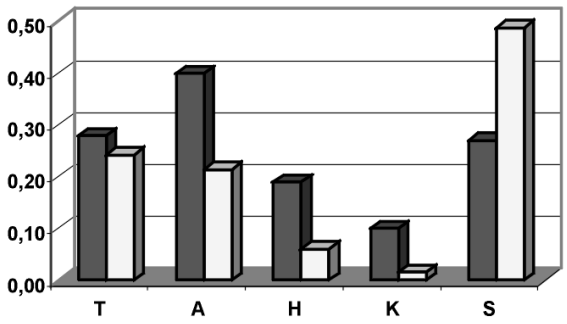


Abb. 4.6: Prozentuale Verteilung der Eingabemodalitäten $m \in \mathcal{M}_{IN}$ in den beiden Domänen DVA (linke Säule) und AIA (rechte Säule) gemittelt über alle Benutzer; in der Darstellung sind sowohl unimodale als auch multimodale Interaktionen mit dem Testsystem enthalten; die Modalitätenbezeichnungen stehen für T=Touchscreen, A=Tastatur, H=Handgestik, K=Kopfgestik und S=Sprache

tet man zusätzlich die Modalitätenverteilungen bei den einzelnen Testpersonen, so lässt sich bei der Mehrheit der Benutzer im DVA-Versuch eine Konzentration auf drei Modalitäten feststellen, wohingegen im Automobilbereich zumeist nur maximal zwei unterschiedliche Modalitäten verwendet werden. Der zur Verfügung stehende Interaktionsraum wird in dem DVA-Szenario wesentlich stärker ausgenutzt.

Beim Vergleich der unimodalen und multimodalen Benutzereingaben zeigt sich, dass der Anteil der kombinierten Interaktionen im DVA-Szenario mit ca. 20% gegenüber 2% im AIA-Versuch deutlich überwiegt. Aufgrund der extrem geringen Datenmenge an multimodalen Interaktionen im Automobilbereich ist ein qualitativer Vergleich mit den umfangreichen Auswertungsergebnissen im DVA-Versuch kaum zu realisieren. In beiden Domänen hat sich aber die Tendenz gezeigt, dass die Gruppe der Anfänger eher als die beiden anderen Gruppen dazu neigen, redundante Eingabe zu machen. Während im DVA-Szenario komplementäre Anteile überwiegen, können im AIA-Szenario überwiegend redundante Eingaben beobachtet werden. Dieses Ergebnis lässt sich direkt auf die unterschiedliche Struktur der Systemkommandos zurückführen. Bei den bimodalen Interaktionen ist die Kombination von Handgestik und Sprache in beiden Anwendungsbereichen sehr beliebt. In den meisten Fällen folgt dabei zeitlich die Sprachäußerung auf die Handgeste.

Bezogen auf die modalitätenspezifischen Interaktionszeiten liegt die durchschnittliche Dauer der Spracheingaben in beiden Domänen dicht beieinander. Der zeitliche Aufwand für die Hand- und Kopfgestik ist dagegen im Automobilbereich deutlich länger. Gleiches gilt für die Verzögerungszeiten im Rahmen von bimodalen Interaktionen. Insgesamt liegen die Interaktionszeiten für die semantisch höherwertigen Modalitäten im interpersonellen Vergleich nur in Ausnahmefällen über 1,5 Sekunden. Dieses Ergebnis kann als heuristisches Kontextwissen unmittelbar in den Integrationsprozess einfließen.

Vergleicht man die Daten der fünf Versuchspersonen, die an beiden Versuchen teilgenommen haben, so fällt auf, dass der Einsatz der Sprache im Automobilbereich deutlich intensiviert wird. Bezogen auf die taktilen Interaktionen kann zudem eine Verschiebung der Eingabeanteile von der Tastatur im DVA-Versuch zum Touchscreen im Automobilbereich beobachtet werden. Obwohl drei dieser fünf Probanden im DVA-Szenario zu den häufigsten Nutzern multimodaler

Interaktionen gehören, agieren sie im AIA-Versuch fast ausschließlich unimodal. Das Ergebnis legt die Vermutung nahe, dass der konkrete Interaktionsstil weniger durch den Benutzer selbst, als vielmehr durch das Testsystem und die Bedienumgebung geprägt wird.

Als wichtiges qualitatives Ergebnis hat die Analyse der Fragebögen ergeben, dass in beiden Domänen die freie Interaktion in der dritten Versuchsphase bis auf eine Ausnahme besser bewertet wird als die vorgegebenen Modalitätenkombinationen in der zweiten Versuchsphase. Benutzer in beiden Domänen haben betont, dass sie wesentlich einfacher und effektiver mit dem Testsystem umgehen können, wenn sie die freie Wahl zwischen mehreren Eingabemöglichkeiten haben und ihnen nicht ein spezieller, wenig flexibler Interaktionsstil von dem System fest vorgeschrieben wird. Dieses Ergebnis bestätigt die initialen Erwartungen, dass funktionale Redundanz in den Eingabemodalitäten hoch geschätzt wird und insgesamt zu einer besseren Bedienbarkeit und zu einer höheren Akzeptanz des technischen Systems führt. Die Versuchspersonen sind darüber hinaus mehrheitlich der Meinung, dass durch die multimodalen Interaktionsmöglichkeiten der Umgang mit dem System einfacher und schneller erlernt werden kann.

Bei der Auswertung der Versuchsdaten stand eine möglichst umfassende Charakterisierung des Benutzerverhaltens unter der Berücksichtigung verschiedener Aspekte im Vordergrund. Es wurde bewusst darauf verzichtet, diese Auswertung aus Sicht der Integrationsaufgabe vorzunehmen, um a priori eine einseitige Betrachtung zu vermeiden. Daher erfolgt eine Untersuchung der einzelnen Ergebnisse im Hinblick auf mögliche Implikationen für das Design einer multimodalen Basisarchitektur erst in den nächsten beiden Kapiteln.

KAPITEL 5

Multimodale Basisarchitektur

In diesem Kapitel wird die Konzeption und die Umsetzung einer Software-Architektur für die Integration multimodaler Informationsanteile beschrieben. Der Fokus liegt auf der Interpretation symbol-orientierter Benutzereingaben und Kontextinformationen. In einer initialen Analysephase werden etablierte Integrationsarchitekturen betrachtet und zentrale Anforderungen für die Entwicklung eines generischen Architekturansatzes identifiziert. Anschließend erfolgt im Rahmen eines Systementwurfs eine Partitionierung des Systems. Der Rest des Kapitels ist maßgeblich geprägt von einer Beschreibung der einzelnen Komponenten. Dabei wird detailliert auf den zugrundeliegenden Kommunikationsformalismus und die unterschiedlichen Integrationsphasen eingegangen. Den Abschluss des Kapitels bilden einige Bemerkungen in Bezug auf den Informationsaustausch und die systemtechnische Umsetzung der einzelnen Komponenten.

5.1 Analysephase

Die Konzeption eines multimodalen Mensch-Maschine-Systems ist ein stark software-lastiges Entwicklungsprojekt. Wissenschaftlich bezeichnet man die Disziplin zur Spezifikation, Implementierung und Evolution solcher Systeme als *Software-Engineering* (SE) [153]. Im Zentrum stehen Methoden und Hilfsmittel für den Prozess zur Herstellung von umfangreichen Software-Systemen. Das Ziel besteht darin, auf Basis vorgegebener funktioneller, zeitlicher und ökonomischer Randbedingungen qualitativ hochwertige Produkte zu erzeugen. Software-basierte Systeme sind in dem Sinne hochgradig abstrakt, dass sie über keine unmittelbare physikalische Form verfügen. Vor diesem Hintergrund unterscheidet sich das Aufgabengebiet von Software-Engineering deutlich von anderen Ingenieur-Disziplinen.

Angelehnt an das klassische Wasserfall-Modell [22] wird im Folgenden auf die initiale Phase im Software-Entwicklungsprozess - die Analysephase - detaillierter eingegangen. Auf Basis einer strukturierten Betrachtung ausgewählter, existierender Interaktionssysteme und den Erfahrungen aus den eigenen Benutzerstudien erfolgt eine Identifikation der maßgeblichen Ziele für die Konzeption einer generischen, applikationsinvarianten multimodalen Basisarchitektur und der zur Umsetzung benötigten Systemfunktionalitäten. Die Analyse ist in einer etwas allgemeineren Form dargestellt. Eine stärker formal ausgerichtete Beschreibung erfolgt in den Abschnitten 5.3 und 5.4.

5.1.1 Existierende Architekturansätze

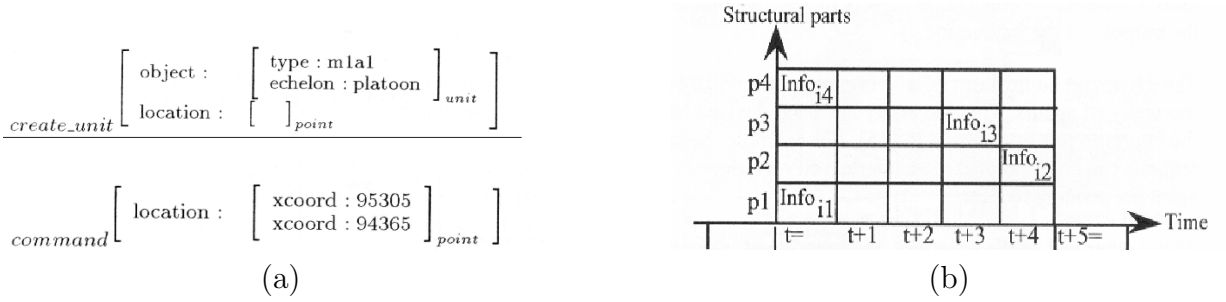
Die Darstellung der folgenden Konzepte zur Realisierung multimodaler Interaktionssysteme bezieht sich maßgeblich auf die bereits in Abschnitt 2.4 vorgestellten Demonstratorumgebungen Quickset [36], MATIS [115] und SGIM [87, 86]. Im Mittelpunkt der Diskussionen stehen ausschließlich Systemansätze, die nach dem Design-Schema von Nigay und Coutaz (siehe Abbildung 2.4 in Abschnitt 2.3.1) multimodale Informationsanteile auf einer rein semantischen Ebene miteinander verbinden. Zur formalen Beschreibung dieser Systeme werden häufig Konstrukte verwendet, die aus der klassischen Wissensmodellierung abgeleitet sind. Die Integration der einzelnen Datenströme wird speziell im Hinblick auf Late-Semantic-Fusion überwiegend mit regelbasierten Verfahren realisiert.

Quickset-System

Im Quickset-System werden zur Modellierung der multimodalen Benutzereingaben sog. *type feature structures* verwendet [36]. Dabei handelt es sich um hierarchisch verschachtelte Strukturen, die sich auf einen Pool von globalen Variablen beziehen. Die explizite Form der Notation hängt vom Typ der modellierten Benutzereingabe ab und ist normalerweise hochgradig domänenspezifisch. Abbildung 5.1(a) zeigt die Modellierung für einen Sprachbefehl (`create_unit`), bei dem die fehlende Positionierungsangabe durch eine referenzierende Geste ergänzt wird (`location`). Neben kategorisierten Benutzerbefehlen können innerhalb dieser Struktur auch zusätzliche Angaben wie Modalität, Zeit und Konfidenzmaß abgelegt werden.

Der Fusionsalgorithmus im Quickset-System basiert auf dem Prinzip der *Unifikation*. In diesem Ansatz wird jeweils eine Gruppe von Benutzereingaben, die aufgrund der zeitlichen Relationen von einer vorverarbeitenden Stufe assoziiert wurden, auf strukturelle und semantische Kompatibilität untersucht. Dabei wird geprüft, ob die Typen der einzelnen Befehle überhaupt zueinander passen und ob durch die Kombination ein syntaktisch korrektes Kommando im Sinne der Applikation entsteht. Im Erfolgsfall findet eine Vereinigung statt und aus den Einzelbefehlen wird eine gemeinsame Datenstruktur gebildet. An dieser Stelle kann zusätzlich noch auf ein statistisches Verfahren zurückgegriffen werden [172]. Die Unifikation findet dann nicht nur einmal statt, sondern für alle sinnvollen Kombinationen aus den Ergebnislisten der einzelnen Erkenner. Durch den zusätzlichen Vergleich der individuellen Lösungen lassen sich insgesamt besser abgesicherte Fusionsergebnisse realisieren.

Die eigentliche Systemarchitektur wurde in Form einer offenen Agentenstruktur umgesetzt [34]. Dabei handelt es sich um ein System von verteilten Agenten, die jeweils aus einer Kommunikationsschicht und einem funktionellen Kern bestehen. Unter Vermittlung eines *Facilitators* tauschen die einzelnen Systemkomponenten Nachrichten in einer eigens festgelegten Meta-Sprache aus. Durch die verteilte Struktur können verschiedene Verarbeitungsprozesse auch parallel ausgeführt werden. Der Vorteil einer solchen Architektur liegt in der einfachen Erweiterbarkeit sowie in der Austauschbarkeit der einzelnen Komponenten. Darüber hinaus können die Agenten in unterschiedlichen Programmiersprachen realisiert werden und auf verschiedenen Hardware-Plattformen laufen. Als einzige Voraussetzung müssen die einzelnen Module über eine geeignete Schnittstelle zu der zentralen Verbindungskomponente verfügen. Der Nachteil der Agentenstruktur besteht darin, dass die Koordination des Nachrichtenstroms einen zusätzlichen Aufwand verursacht.



(a) (b)

Abb. 5.1: Beispiele für eine abtrakte Darstellung multimodaler Informationsanteile mit (a) *type feature structures* aus dem Quickset-System [36] und (b) *melting pots* aus dem MATIS-System [115]

MATIS-SYSTEM

Im MATIS-System [115] werden Benutzerbefehle ebenfalls in eine einheitliche, hier als *melting pot* bezeichnete Framestruktur eingeordnet. Die Modellierung besteht aus mehreren Zellen, die auch als *slots* bezeichnet werden. In diese Zellen werden die einzelnen semantischen Informationsanteile mit einem sog. *slot-filling Mechanismus* eingetragen. Abbildung 5.1(b) veranschaulicht die Struktur an einem abstrakten Beispiel. Die strukturelle Zugehörigkeit der einzelnen slots ist auf der Ordinate ablesbar. Diese Einordnung wird in Abhängigkeit vom semantischen Gehalt der Information bereits im Vorfeld bestimmt. Auf der Abszisse wird der entsprechende Zeitstempel dargestellt. Bei den allgemein mit p1, p2, etc. angegebenen strukturellen Anteilen kann es sich im Rahmen der Anwendung beispielsweise um den Abflugort, den Zielflughafen oder den Typ der angeforderten Information handeln. Die genaue Zuordnung wird extern deklariert und dem System vor Anwendungsbeginn zugeführt.

Das MATIS-System verwendet ebenfalls eine Architektur von verteilten Agenten. Die einzelnen Komponenten sind sternförmig direkt untereinander vernetzt. Auf Basis der Modellierung in melting pots findet die Fusion der Benutzereingaben im Vergleich zu Quickset auf einer höheren Abstraktionsebene statt. Dabei werden sowohl zeitliche als auch semantische Relationen betrachtet. Eine Integrationsstruktur besteht aus einem oder mehreren Zeitfenstern gleicher Länge, deren Anzahl von der Dauer des Befehls abhängt. Jeder dieser Strukturen wird eine feste Zeitspanne zugeordnet (T), in der sie Informationen aufnehmen kann.

Bei der Fusion zweier Benutzereingaben findet zunächst eine Prüfung auf Redundanz statt. Im Erfolgsfall wird einer der beiden Befehle gelöscht. Besteht zwischen den zwei Befehlen eine strukturelle Komplementarität, so findet in Abhängigkeit von ihrer zeitlichen Relation anschließend eine temporale Fusion statt. Dabei werden sowohl zeitlich direkt überlagerte Eingaben als auch sequentielle Eingaben betrachtet, die innerhalb der Zeitspanne T auftreten. Die Informationen aus den einzelnen Quellen werden dann in eine gemeinsame Integrationsstruktur eingetragen. Für mehrere Datenquellen wird dieser Vorgang so lange iteriert, bis nur noch eine einzige Integrationsstruktur vorhanden ist bzw. keine weitere Fusion mehr möglich ist. Durch die Verzögerung der Ausführung um die Dauer T wird dem System die Möglichkeit gegeben, auf verzögert ankommende Benutzerbefehle adäquat reagieren zu können. In einem Auskunftssystem wie MATIS, das durch den selektiven Zugriff auf einzelne Informationsanteile charakterisiert ist, kann eine solche Verzögerung toleriert werden, in Domänen mit strikten Realzeitanforderung führt dies allerdings zu erheblichen Akzeptanzproblemen.

SGIM-System

Das SGIM-Projekt [87] verfolgt ein Konzept namens *Hypothesis*, bei dem den einzelnen Befehlsanteilen jeweils ein Symbol zusammen mit einer Lebensspanne und einem Konfidenzmaß zugeordnet wird. Im Gegensatz zu den anderen Modellierungen bestehen hier keine starren Strukturen. Jedes Symbol ist jedoch innerhalb eines streng hierarchisch geordneten Superkonzepts eingeordnet. Je höher die hierarchische Stufe des Symbols ist, desto höher liegt der Abstraktionsgrad. Auf der untersten Stufe wird den einzelnen Benutzereingaben ein Aktionssymbol zugeordnet. Erst auf den übergeordneten Ebenen wird dann entschieden, ob die mit dem Symbol assoziierte Systemfunktionalität überhaupt im Zusammenhang mit der Anwendung steht und real umgesetzt werden kann.

Beim SGIM-System kommt im Gegensatz zu den anderen Systemen kein direkter slot-filling-Mechanismus zum Einsatz. Die einzelnen Symbole werden hier bereits innerhalb einer Modalität nach festgelegten Regeln kombiniert. Eine solche Kombination wird als Hypothese bezeichnet. Fusionen können auf unterschiedlichen Abstraktionsstufen durchgeführt werden. Bei Befehlen verschiedener Modalitäten geschieht dies auf der höchsten Abstraktionsstufe, da erst an dieser Stelle eine einheitliche Repräsentation der Daten realisiert wird. Um die Komplexität einzuschränken, werden den einzelnen Symbolen in Abhängigkeit von ihrem semantischen Gehalt Lebensdauern zugeordnet, nach deren Ablauf sie aus dem Speicher gelöscht werden.

Die Realisierung dieses Konzeptes erfolgte in Form eines Expertensystems, wobei die Wissensbasis auf die einzelnen hierarchischen Stufen verteilt ist. Auf jeder Integrationsstufe kann nur auf das Wissen zugegriffen werden, das an dieser Stelle lokal verfügbar ist. Zur Modellierung der Inferenzmaschine wird CLIPS[1] [133] eingesetzt. Der zugrundeliegende Mechanismus setzt die im Speicher liegenden Symbole mit den jeweils vorhandenen Integrationsregeln zueinander in Bezug und löst im Falle einer Übereinstimmung eine entsprechende Aktion aus. Auf dieser Basis erfolgt eine Hypothesenbildung und eine Weiterleitung der Informationen an die jeweils übergeordnete hierarchische Instanz. Der Vorteil eines solchen Expertensystems liegt in der einfachen Erweiterbarkeit. Ein Nachteil besteht darin, dass die Anzahl der Ausführungsschritte exponentiell mit der Anzahl der im Speicher liegenden Symbole steigt. Je nach Umfang des abzuarbeitenden Regelsatzes können dadurch selbst auf performanten Systemen erhebliche Verzögerungen auftreten.

5.1.2 Identifikation zentraler Anforderungen

Auf Basis der Betrachtung verschiedener Integrationsarchitekturen im letzten Abschnitt und den Erfahrungen aus den eigenen Benutzerstudien (siehe Kapitel 4) lässt sich eine Reihe von Anforderungen an das Konzept einer generischen multimodalen Basisarchitektur ableiten. Dabei wird generell zwischen zwei Kategorien unterschieden [153]. Mittels *funktionaler Anforderungen* kann beschrieben werden, welche Funktionalitäten ein System bereitstellen soll und in welcher Weise auf verschiedene Eingabeereignisse reagiert werden muss. Darüber hinaus können Grenzen in Bezug auf die Leistungsfähigkeit des Systems spezifiziert werden. In diesen Anforderungen kann beispielsweise auch enthalten sein, was das System explizit nicht leisten soll.

Demgegenüber beschreiben *nicht-funktionale Anforderungen* existierende Randbedingungen für den Betrieb des Systems. Sie stellen Restriktionen für die Freiheiten während der System-

[1] CLIPS steht für C Language Integrated Production System

entwicklung dar. Im Einzelnen werden durch diese Kategorie von Anforderungen Kriterien für spezielle Systemeigenschaften wie Zuverlässigkeit, Antwortzeiten und Speicherkapazitäten spezifiziert. Zusätzlich sind Informationen über eine adäquate interne Repräsentation der Daten enthalten. Einige nicht-funktionale Anforderungen beschreiben eher Prozess- als Produkteigenschaften. Durch die Einhaltung von Qualitätsstandards bestimmen sie wichtige Richtlinien für den Planungs- und Umsetzungsprozess.

Beide Arten von Anforderungen spezifizieren das externe System-Verhalten. Sie sollten daher weder spezifische Design-Charakteristika enthalten noch durch die Verwendung einzelner Implementationsmodelle geprägt sein. Nur wenn sowohl funktionale als auch nicht-funktionale Anforderungen bereits im Systemdesignprozess hinreichend berücksichtigt werden, ist eine performante Koordination der einzelnen Informationskanäle innerhalb der realen multimodalen Bedienumgebung möglich. Im Folgenden werden einige ausgewählte Anforderungen näher erläutert. Diese Anforderungen sind dabei nach Prioritäten in Bezug auf die Relevanz für die Konzeption und die Umsetzung eines multimodalen Prototypen geordnet.

- **Freie Wahl der Interaktionsmodalität**
  Als generelle, übergeordnete Anforderung gilt, dass der Benutzer jederzeit die freie Wahl zwischen mehreren Modalitäten haben soll. Dieser Ansatz impliziert bereits in der Planungsphase des Systems eine konsequente Ausrichtung nach dem Äquivalenz-Prinzip (siehe Abschnitt 2.3.1). Im Hinblick auf die einzelnen Systemfunktionalitäten wird dem Benutzer weder eine spezifische unimodale Interaktionsart noch eine explizite multimodale Kommunikationsweise vorgeschrieben. Mehrere Möglichkeiten können alternativ genutzt werden. Das System muss zu diesem Zweck über ein Reservoir an Statusinformationen verfügen, um eindeutig feststellen zu können, zu welchem Zeitpunkt der Benutzer mit welcher der verfügbaren Modalitäten kommuniziert.

  Aus der obigen Beschreibung leitet sich direkt eine weitere Anforderung ab. Das System muss von seiner Struktur und den verwendeten Schnittstellen prinzipiell für mehrere Modalitäten offen sein. Eine explizite Beschränkung auf Sprache und Gestik, wie in vielen aktuellen Demonstrationssystemen, ist im Vorfeld wenig sinnvoll, da potentielle Nutzer bereits a priori in ihrem Interaktionsverhalten deutlich eingeschränkt würden.

- **Zeitliche Assoziation einzelner Informationsanteile**
  Im Hinblick auf die Kombination unterschiedlicher Informationsquellen muss der Integrator in der Lage sein, zeitlich versetzt auftretende Erkennerergebnisse zueinander in Beziehung zu setzen und im Kontext des Gesamtsystems sinnvoll zu interpretieren. Im allgemeinen Fall kann davon ausgegangen werden, dass die einzelnen Daten sequentiell nacheinander mit unterschiedlichen Abständen bei der zentralen Integrationskomponente eintreffen. Diese Verzögerungen können zum einen direkt durch den Benutzer verursacht sein, zum anderen aber auch auf unterschiedliche Erkennerlaufzeiten zurückgeführt werden. Der genaue Zeitpunkt, an dem der Benutzer eine spezifische Eingabe macht, ist für den Integrationsvorgang von fundamentaler Bedeutung. Aus diesem Grund müssen alle Komponenten des multimodalen Systems untereinander synchronisiert sein.

- **Funktionale Assoziation einzelner Informationsanteile**
  In Ergänzung zu der Interpretation der zeitlichen Relationen muss das System auch mit verschiedenen Konstellationen in Bezug auf die semantische Zusammensetzung der einzelnen Daten umgehen können. Dazu ist es wichtig, dass der verwendete Integrationsmechanismus zunächst strukturelle Informationsanteile aus dem Datenstrom extrahiert und im Hinblick auf die aktuelle Applikationsdomäne kategorisiert. Im nächsten Schritt können redundante, komplementäre und rivalisierende Benutzereingaben identifiziert werden. Schließlich müssen die einzelnen Informationsanteile verbunden und auf eine sinnvolle Systemaktion abgebildet werden, die den Erwartungen des Benutzers entspricht.

- **Behandlung unimodaler Benutzereingaben**
  Neben der Behandlung multimodaler Benutzereingaben besteht eine weitere Anforderung darin, dass das zu entwickelnde Interaktionssystem auch unimodale Benutzereingaben verarbeiten kann. Bezogen auf das Design-Schema von Nigay und Coutaz [114] stellen atomare Benutzereingaben, bei denen sämtliche relevanten Informationen bereits durch eine Modalität in vollständiger Form übertragen werden, vom Komplexitätsgrad her eine untergeordnete Problemklasse dar. Im Hinblick auf die Benutzbarkeit und die Akzeptanz des multimodalen Prototypen kommt diesen Eingaben jedoch eine außerordentliche Bedeutung zu. Unterschiedliche Benutzerstudien (siehe Abschnitt 2.3.2) haben klar gezeigt, dass auch in einer multimodalen Systemumgebung zu einem großen Anteil unimodal mit dem technischen System interagiert wird.

- **Beurteilbarkeit der Integrationsergebnisse**
  Die komplexe Struktur der multimodalen Informationsanteile impliziert im Allgemeinen eine Menge von mehreren möglichen Systemreaktionen. Um möglichst optimal die ursprüngliche Intention des Benutzers bestimmen zu können, muss eine Aussage über die Qualität der einzelnen Lösungsalternativen getroffen werden. Der Entscheidungsprozess gestaltet sich erheblich einfacher, wenn ein quantitatives Maß zur Beurteilung der potenziellen Integrationsergebnisse existiert. In die Entwicklung eines geeigneten Bewertungsverfahrens fließen unterschiedliche Informationen ein. Neben den einzelnen Erkennungsergebnissen und evtl. vorhandenen Konfidenzmaßen müssen auch Meta-Informationen aus dem multimodalen Fusionsprozess und die aktuelle Bediensituation berücksichtigt werden.

- **Benutzer- und Kontextadaption**
  Eine der wichtigsten Erfahrungen aus den Benutzerstudien besteht darin, dass sich ein allgemeines, für alle Benutzer stereotypisches multimodales Kommunikations-Verhalten nicht feststellen lässt. Aus diesem Grund muss das System prinzipiell über die Möglichkeit verfügen, dass es explizit an einzelne Benutzer oder Benutzerklassen angepasst werden kann. Auf der anderen Seite muss das System auch flexibel auf etwaige Änderungen im Systemsetup und in der Umgebung reagieren. Das Potenzial zur Adaption des Systems muss für beide Bereiche in gleichem Maße in das Design der Integrationsalgorithmen einfließen.

- **Situationsspezifisches Fehlermanagement**
  Im Kontext der Bedienung von komplexen Informationssystemen können an verschiedenen Stellen Fehler auftreten. Diese Fehler können auf der einen Seite durch den Benutzer, beispielweise durch eine fehlerhafte Eingabe verursacht werden. Auf der anderen Seite ist

das System selbst eine entsprechende Fehlerquelle. So kann bei der Verwendung statistischer Klassifikatoren immer nur mit einer gewissen Wahrscheinlichkeit auf die Richtigkeit der erkannten Benutzerintention zurückgeschlossen werden. Ein benutzergerechtes multimodales Integrationssystem muss mit beiden Fehlerquellen umgehen können. Dazu sollten geeignete Strategien entwickelt werden, um potenzielle Fehler im Mensch-Maschine-Dialog bereits im Vorfeld zu vermeiden und im realen Systembetrieb auftretende Fehlersituationen zu entdecken und geeignete Maßnahmen zur Fehlerauflösung einzuleiten.

- **Portabilität und Erweiterbarkeit der Systemmodule**
  Um die verschiedenen Anforderungskategorien überhaupt erfüllen zu können, muss die Architektur prinzipiell so konzipiert werden, dass die einzelnen Komponenten sowohl leicht modifiziert und erweitert als auch gegen andere Module ausgetauscht werden können. Die Systemkomponenten müssen dazu über systemweit einheitliche Schnittstellen verfügen. Darüber hinaus muss ein abstrakter Kommunikationsformalismus entwickelt werden, der dieser Schnittstellenspezifikation genügt und den Austausch der Informationen zwischen den Systemkomponenten regelt.

  Im Hinblick auf die Entwicklung einer generischen Basisarchitektur, die flexibel in verschiedenen Systemumgebungen eingesetzt werden kann, besteht eine weitere Anforderung darin, applikationsspezifisches Wissen in das Gesamtsystem integrieren zu können. Dazu ist es nötig, die Spezifikation der Systemabläufe von der reinen Codierungs-Ebene zu abstrahieren. Auf diese Weise können zumindest in eingeschränktem Maße auch Nicht-Entwickler die zugrundeliegenden Interaktionskonzepte auf andere Anwendungsszenarien und auf weitere Eingabe- und Ausgabemodule erweitern.

- **Effektive, effiziente und robuste Systembedienung**
  Vor dem Hintergrund einer realen, weitestgehend echtzeitfähigen Applikation muss schließlich sichergestellt werden, dass der zusätzliche systemtechnische Aufwand, der durch die Verfolgung einer multimodalen Interaktionsstrategie entsteht, nicht zu Lasten der Systemleistung geht. Es gilt, den Trade-Off zwischen den Vorteilen einer insgesamt intuitiveren und fehlerrobusteren Systembedienung auf der einen Seite und den existierenden Mehrkosten an Zeit und Rechenleistung auf der anderen Seite zu identifizieren und realistisch zu bewerten. Das Ziel für ein benutzergerechtes Interaktionssystem kann nicht darin bestehen, Multimodalität als unantastbares Paradigma über existierende ökonomische Randbedingungen zu stellen.

5.2 Systementwurf

Unter Beachtung der identifizierten Anforderungen leitet sich die Konzeption einer multimodalen Mensch-Maschine-Schnittstelle aus den klassischen, bereits in Abschnitt 2.2.2 diskutierten Architekturmodellen ab. Die Eingaben des Benutzers werden über eine externe Sensorik erfasst, vom System im aktuellen Bedienkontext interpretiert und auf eine spezifische Applikationsfunktionalität abgebildet. Als Ergebnis dieses Verarbeitungsprozesses erhält der Benutzer eine entsprechende Rückmeldung auf seine Eingaben. Die Verbindung der eingehenden Informationen und die Steuerung der zugrundeliegenden Applikation sind im Allgemeinen funktional entkoppelt und laufen als sequentielle Prozesse nacheinander ab.

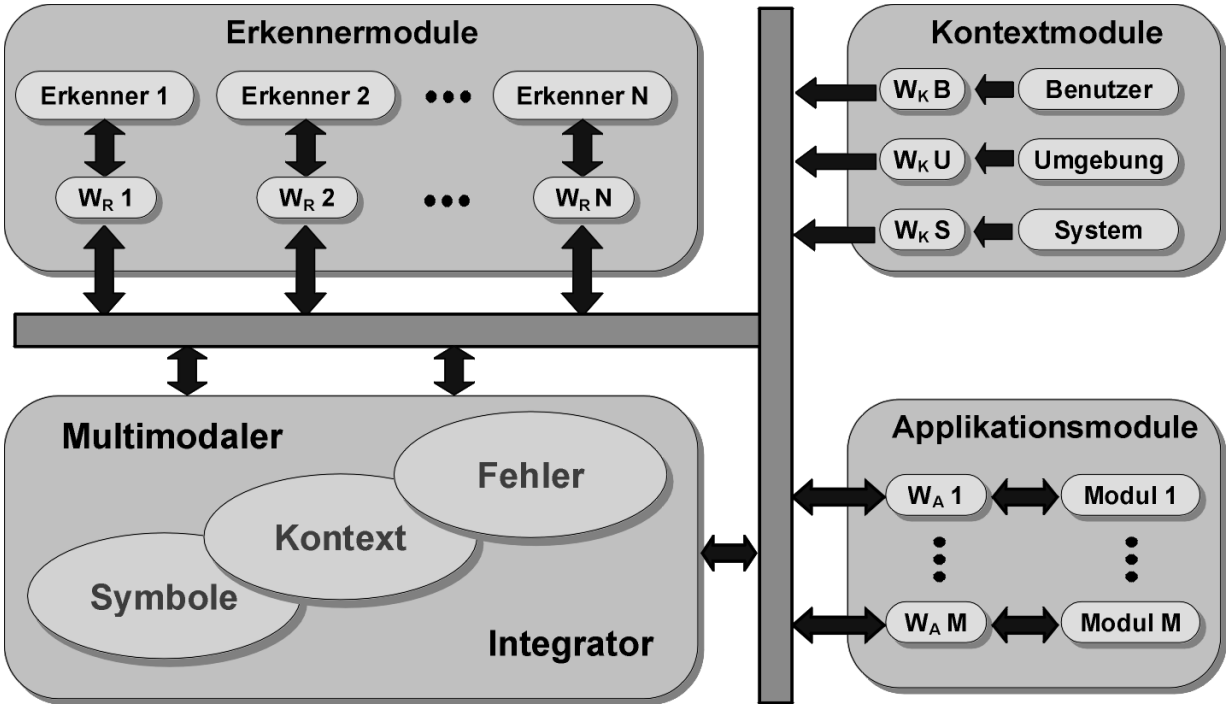


Abb. 5.2: Module der multimodalen Basisarchitektur

Im Mittelpunkt dieser Arbeit steht die Entwicklung einer generischen Software-Architektur und entsprechender algorithmischer Verfahren für die Integration multimodaler Benutzereingaben. Vor dem Hintergrund wechselnder Anwendungsdomänen und unterschiedlicher Eingabegeräte besteht ein potenzieller Lösungsansatz für einen Systementwurf in einer streng modularen, funktionsorientierten Partitionierung des Gesamtsystems. Abbildung 5.2 verdeutlicht den strukturellen Aufbau einer solchen Architektur und zeigt den groben Informationsfluss zwischen den einzelnen Komponenten auf. Es lassen sich im Wesentlichen vier logisch abgeschlossene Gruppen identifizieren: *Erkennermodule*, *Kontextmodule*, *Integratormodule* und *Applikationsmodule*. Im Folgenden wird nur überblicksweise auf diese Module eingegangen. Eine detaillierte Beschreibung erfolgt sukzessive in den restlichen Abschnitten des Kapitels.

5.2.1 Eingabemodule

Nach dem Prinzip der semantischen Fusion wird bei den Erkennermodulen auf bereits existierende Komponenten zurückgegriffen, die jeweils eine Menge von in sich abgeschlossenen Eigenschaften besitzen und vollständig unabhängig voneinander betrachtet werden können. Sind diese Anforderungen erfüllt, so spricht man auch von einer *paramodalen Repräsentation* der Eingabedaten [119]. Die Erkennermodule sind im Vorfeld weder in Bezug auf die Anzahl noch in Bezug auf den Typ näher bestimmt. Das Spektrum reicht von einfachen Elementen, wie Tastatur- oder Maustreibern über aufwendige Klassifikationsmodule zur Interpretation natürlich-sprachlicher Äußerungen oder dynamischer Handgesten bis hin zu komplexen Überwachungssystemen, die in sich bereits eine Vielzahl an multimodalen Eingaben und Kontextinformationen verbinden.

Durch die paramodale Repräsentation der Informationen können im System auch mehrere Komponenten auf den gleichen Interaktionskanal zugreifen. Die Möglichkeit der Co-Existenz ähnlicher Erkennermodule ist ein besonderes Merkmal der Architektur. Formal lässt sich die Gruppe der Erkennermodule als Menge $\mathcal{M}_R$ beschreiben. Die Anzahl der Erkennerkomponenten wird mit $\rho(R) \in \mathbb{N}$ angegeben.

$$\mathcal{M}_R = \{M_{R,1}, M_{R,2}, \ldots, M_{R,\rho(R)}\} \tag{5.1}$$

Die Gruppe der Kontextmodule stellt die zweite wichtige Datenquelle für den Integrationsprozess dar. In diesen Modulen werden zusätzliche Informationen verarbeitet, die über die intentionalen Eingaben des Benutzers hinausgehen. Analog zu den Erkennermodulen können auch die Kontextmodule als Menge $\mathcal{M}_K$ beschrieben werden. Die Kardinalität dieser Menge wird mit $\rho(K) \in \mathbb{N}$ angegeben.

$$\mathcal{M}_K = \{M_{K,1}, M_{K,2}, \ldots, M_{K,\rho(K)}\} \tag{5.2}$$

Kontextinformationen lassen sich in Benutzer-, System- und Umgebungskontext unterteilen. Die Module für die einzelnen Kontextklassen werden jeweils abkürzend mit $M_{K(B)}$, $M_{K(S)}$ und $M_{K(U)}$ bezeichnet. Entsprechend geben $\rho(K(B))$, $\rho(K(S))$ und $\rho(K(U))$ die Zahl der jeweiligen Systemmodule an. Insgesamt gilt die Beziehung $\rho(K) = \rho(K(B)) + \rho(K(S)) + \rho(K(U))$. Der Benutzerkontext subsumiert Informationen über die körperliche und emotionale Verfassung des Benutzers, individuelle Präferenzen und empirisches Wissen, beispielsweise aus Benutzerstudien oder vorhergegangenen Bedienabläufen. Im Systemkontext sind detaillierte Informationen über die einzelnen Systemmodule abgelegt, wie die temporäre Auslastung, Fehlermeldungen oder Statusmeldungen. Der Umgebungskontext umfasst zusätzliche Informationen, die nicht primär im Zusammenhang mit dem Mensch-Maschine-System stehen. So fließen beispielsweise aufgabenbedingte Kontextinformationen ebenfalls in den Umgebungskontext ein.

Die Erkenner $\mathcal{M}_R$ und die Kontextmodule $\mathcal{M}_K$ werden zur Vereinfachung der Darstellung zu einer übergeordneten Kategorie zusammengefasst. Da beide Gruppen den Integrator mit Eingabedaten versorgen, bezeichnet man die Vereinigungsmenge von $\mathcal{M}_R$ und $\mathcal{M}_K$ als Eingabemodule und referenziert sie über das Symbol $\mathcal{M}_E$. Die Gesamtanzahl $\rho(E)$ ergibt sich aus der Summe der einzelnen Erkenner- und Kontextmodule.

$$\mathcal{M}_E = \mathcal{M}_R \cup \mathcal{M}_K \quad \text{und} \quad \rho(E) = \rho(R) + \rho(K) \tag{5.3}$$

Zusammenfassend werden diese Eingabedaten analog zu der Terminologie aus Kapitel 4 als abstrakte *Ereignisse* bzw. *Events* bezeichnet. Die Menge aller möglichen Ereignisse wird formal mit $\mathcal{E}$ angegeben. Diese Menge setzt sich aus den Ereignisspektren der einzelnen Eingabemodule zusammen. Ist die Herkunft eines Ereignisses $E \in \mathcal{E}$ für die weitere Verarbeitung von Bedeutung, so wird die Kategorie des Moduls als hochgestellter Index mitgeführt. Erkennergestützte Benutzereingaben werden mit E^R und Kontextinformationen mit E^K abgekürzt. Die dazugehörigen Ereignismengen werden entsprechend mit $E^R \in \mathcal{E}^R$ und $E^K \in \mathcal{E}^K$ bezeichnet. Bei der Modellierung wird strikt zwischen den einzelnen Modulklassen unterschieden. In der Praxis stellen verschiedene Systemteile jedoch sowohl Erkenner- als auch Kontextfunktionalitäten zur Verfügung und werden häufig in einer einzigen Komponente zusammengefasst.

Die Architektur ist bereits vom Ansatz her darauf ausgelegt, eine Vielzahl unterschiedlich strukturierter Informationsquellen zu verarbeiten. Als zentrales Element beinhaltet der Systementwurf einen generischen Kommunikationsformalismus (siehe Abschnitt 5.3), mit dem die Daten abstrakt beschrieben und zwischen den einzelnen Modulen ausgetauscht werden können. Jeder Systemkomponente wird dazu ein speziell angepasstes Konvertermodul nachgeschaltet. Da die Konvertermodule unterschiedliche Informationsdarstellungen ineinander überführen, werden sie auch *Wrapper* genannt und mit W abgekürzt.

Die Integration der einzelnen Informationsanteile erfolgt unabhängig von den proprietären Geräteformaten auf einer konzeptionellen Ebene. Ein essentieller Vorteil dieses Ansatzes besteht darin, dass jederzeit neue Komponenten in das System integriert werden können. Um mit den anderen Systemmodulen Daten austauschen zu können, müssen sie lediglich über eine entsprechende Schnittstellenspezifikation verfügen.

5.2.2 Applikationsmodule

Die komplette Anwendungslogik ist in den Applikationsmodulen enthalten. Sie kommunizieren ebenfalls über entsprechende Konvertermodule W_A bilateral mit dem multimodalen Integrator und den anderen Systemkomponenten. In Bezug auf den Informationsfluss zwischen Mensch und Maschine stellen die Applikationsmodule die komplementäre Ergänzung zu den Eingabemodulen dar. Durch diese Module können die verschiedenen Ausgabegeräte angesteuert und Informationen an den Benutzer übertragen werden. Analog zu den anderen Systemmodulen lassen sich auch die Applikationsmodule als Menge $\mathcal{M}_A$ darstellen, wobei $\rho(A) \in \mathbb{N}$ die Anzahl der Module in dieser Klasse angibt:

$$\mathcal{M}_A = \{M_{A,1}, M_{A,2}, \ldots, M_{A,\rho(A)}\} \tag{5.4}$$

Es existiert eine strikte funktionale Trennung zwischen den einzelnen Applikationsmodulen. Die Kopplung der Module erfolgt nur über das Integratormodul. Jedes Anwendungsmodul $M_{A,i}$ mit $i \in \{1..\rho(A)\}$ verfügt über eine Reihe von Funktionalitäten, die durch den Integrator angesteuert werden können. Diese Funktionalitäten lassen sich ebenfalls als Menge $\mathcal{A}_i = \{A_{i,1}, A_{i,2}, ..., A_{i,\alpha(i)}\}$ beschreiben, wobei $\alpha(i)$ die Anzahl der gekappselten, modulspezifischen funktionalen Eigenschaften angibt. Im Hinblick auf die Applikation werden die einzelnen Elemente in dieser Menge als *Aktionen* bezeichnet. Die Menge aller möglichen Aktionen $\mathcal{A}$ ergibt sich aus der Vereinigungsmenge der individuellen Funktionsspektren:

$$\mathcal{A} = \bigcup_{i=1}^{\rho(A)} \mathcal{A}_i \quad \text{und} \quad \alpha = \sum_{i=1}^{\rho(A)} \alpha(i) \tag{5.5}$$

Eine wichtige Eigenschaft des Gesamtsystems ist die Abgeschlossenheit der Eingabemodule bzgl. der zur Verfügung stehenden Applikationsfunktionalitäten. In dem multimodalen Bedienumfeld muss jede Systemfunktionalität entweder durch eine einzelne unimodale Eingabe, durch eine Sequenz von unimodalen Interaktionen oder durch eine Kombination unterschiedlicher Eingaben ausgelöst werden können. Nach dem Äquivalenzansatz besteht das Ziel darin, soweit möglich immer mehrere Interaktionswege zur Verfügung zu stellen. Der Benutzer kann dann individuell nach seinen Präferenzen aus den Alternativen wählen.

5.2.3 Generischer Integrationskern

Der multimodale Integrator stellt die zentrale Komponente der Architektur dar. An dieser Stelle werden die einzelnen Benutzereingaben und Kontextinformationen zeitlich und semantisch assoziiert. Als Ergebnis des Integrationsprozesses wird eine Sequenz von Systemaktionen ausgelöst und dem Benutzer über verschiedene Ausgabegeräte angezeigt. Der Integrator besteht aus einem generischen Kern, der ohne grossen Aufwand an verschiedene Anwendungsdomänen und Randbedingungen angepasst werden kann. Die domänenspezifische Konfiguration des Integrators erfolgt über eine standardisierte Schnittstelle von außerhalb. Daher ist für die Aktivierung einer Parametermodifikation auch keine Recompilation des Programmcodes nötig.

Zusammenfassend kann das Problem der semantischen Integration einzelner Informationsanteile wie folgt beschrieben werden. Auf Basis einer Menge von n zusammengehörigen Ereignissen $\mathcal{C}_E = \{E_1, E_2, \ldots, E_n\}$ mit $E_i \in \mathcal{E}$ für alle $i \in \{1..n\}$ bildet der Intentionsdekoder Φ die Ereignisse auf eine isolierte Systemaktion ab. Für eine formal korrekte Darstellung dieser Abildung benötigt man die Potenzmenge $\mathcal{E}^*$. Sie enthält alle möglichen Ereignismengen, die auf der Basis von $\mathcal{E}$ gebildet werden können. In der Funktion Φ wird einem speziellen Element aus dieser Menge $\mathcal{C}_E \in \mathcal{E}^*$ eine Systemaktion $A \in \mathcal{A}$ zugeordnet.

$$\Phi : \mathcal{E}^* \rightarrow \mathcal{A} \quad \text{mit} \quad A = \Phi(\mathcal{C}_E) \quad \text{und} \quad A \in \mathcal{A},\ \mathcal{C}_E \in \mathcal{E}^* \tag{5.6}$$

Zur Vereinfachung der Darstellung werden in dieser Notation auch Sequenzen einzelner Aktionen $A_x A_y A_z$ als eine eigenständige Aktion A_{xyz} kodiert. Um darüber hinaus die aufwendige, indirekte Schreibweise über die Potenzmenge, die getrennte Funktionsangabe und die Ausweisung der Elementbeziehungen zu vermeiden, wird für die Verknüpfung mehrerer Elemente im Hinblick auf den Integrationsprozess ein spezieller Operator $\oplus$ eingeführt. Unter dieser Voraussetzung lässt sich Gleichung 5.6 alternativ auch wie folgt beschreiben:

$$\Phi : E_1 \oplus E_2 \oplus ... \oplus E_n \rightarrow \mathcal{A} \quad \text{mit} \quad E_i \in \mathcal{C}_E \tag{5.7}$$

Der gesamte Integrationsprozess gliedert sich in mehrere Phasen. Zuerst müssen in dem multimodalen Informationsstrom zusammengehörige Ereignisse identifiziert werden. Diese initiale Phase bezeichnet man als *Segmentierung* oder in Anlehnung an die Terminologie zu Objekterkennungsverfahren aus der Bildverarbeitung als *Spotting*. Erst in der nachfolgenden *Fusion* bildet der Intentionsdekoder die durch das Spotting zeitlich und semantisch assoziierten Ereignisse auf eine entsprechende Systemaktion ab. Die Funktionsweise des Integrators wird detailliert in Abschnitt 5.4 beschrieben.

Charakteristisch für die Architektur ist der modulare Aufbau des Integrators aus drei in sich verschachtelten Ebenen (siehe Abbildung 5.2). Das Abstraktions-Niveau, auf dem die einzelnen Informationsanteile verarbeitet werden, steigt dabei von der reinen Symbolverarbeitung über die Kontextevaluation bis hin zum Fehlermanagement kontinuierlich an. Höhere Ebenen greifen auf die Resultate der unteren Ebenen zu. Auf diese Weise fungieren die jeweils übergeordneten Ebenen zum einen als kontrollierende und zum anderen als revalidierende Instanzen. Die Integrationsstrategie kann als maßgeblich ereignisgetriebener Bottom-Up Ansatz beschrieben werden. Eine einzelne Stufe baut zwar extrem auf den Verarbeitungen in den vorangegangenen Phasen auf, funktioniert aber zumindest mit eingeschränktem Leistungsspektrum auch ohne die nachfolgenden Auswertungen.

Die explizite Trennung der einzelnen Integrationsphasen stellt eine effektive Möglichkeit dar, um den Integrator an wechselnde Arbeitsbedingungen anzupassen. Auf der untersten Ebene werden lediglich die semantisch abstrahierten Benutzereingaben verbunden. Darauf aufbauend werden in der nächsten Ebene aktuelle Kontextinformationen über den Benutzer, das System und die Umwelt in den Integrationsprozess einbezogen. Dieses Vorgehen ermöglicht eine bessere und robustere Interpretation der Benutzereingaben in der jeweils aktuellen Bediensituation. Auf der höchsten Ebene können auftretende Fehler im Systemablauf und in der Bedienung identifiziert, klassifiziert und situationsspezifisch behandelt werden.

5.3 Kommunikationsformalismus

Eine wesentliche Problematik bei der Integration multimodaler Benutzereingaben besteht darin, die heterogene Struktur der unterschiedlichen Informationsquellen sinnvoll zueinander in Beziehung zu setzen. Aus diesem Grund werden die modalitätenspezifischen Datenformate durch entsprechende Konvertermodule zunächst in eine gemeinsame, geräteunabhängige Darstellung transformiert. Durch diesen formalen Zwischenschritt kann die Entwicklung von Fusionsalgorithmen erheblich vereinfacht werden. Der Abstraktionsansatz wird analog auch zur Anbindung der Kontextmodule und zur Ansteuerung der Applikationsmodule eingesetzt.

5.3.1 Formale Grundlagen

Der verwendete Darstellungsformalismus geht auf die Theorie der formalen Sprachen zurück. Anhang B.1 beinhaltet eine knappe Rekapitulation der grundlegenden Begriffsterminologien. Ausgehend von einem *Alphabet* an unterscheidbaren Symbolen $\Sigma = \{S_1, S_2, ..., S_\sigma\}$ kann eine formale Sprache $\mathcal{L}$ als Menge aller syntaktisch korrekten *Wörter* über Σ definiert werden. Symbole repräsentieren die elementaren Komponenten einer formalen Sprache und Wörter entstehen durch die Verkettung von Symbolen. Formal gilt $\mathcal{L} \in \Sigma^*$, d.h. $\mathcal{L}$ ist eine Teilmenge der Potenzmenge von Σ. Als zentrale Referenz für eine detaillierte Einführung dient [138].

Bei großen Wortmengen wird $\mathcal{L}$ normalerweise durch eine formale *Grammatik* definiert. Grammatiken bestehen aus einer Menge von Elementen, die durch Regeln zueinander in Beziehung gesetzt werden. In der allgemeinen Form werden die Elemente auf der linken Seite einer Regel durch die Elemente auf der rechten Seite ersetzt. Die Regeln einer Grammatik sind zumeist nicht deterministisch. Einer Symbolfolge auf der linken Seite einer Regel stehen oftmals mehrere Ersetzungsmöglichkeiten auf der rechten Seite gegenüber. Die einzelnen Regelelemente setzen sich aus Symbolen des zugrundeliegenden Alphabets (*Terminalsymbole*) und aus Variablen (*nicht-terminale Symbole*) zusammen, die nicht zum Umfang der Sprache gehören. Zusätzliche strukturierende Elemente (*Meta-Variablen*) stellen die Verbindung zwischen den Symbolen und den Variablen her.

Mit Hilfe einer formalen Grammatik können syntaktisch korrekte Wörter einer Sprache erzeugt werden. Bei dieser *Generierung* eines Wortes w_1 wird beginnend mit einem Startsymbol jede Variable Schritt für Schritt ersetzt, bis nur noch Symbole der Sprache übrig sind. Aus diesem Grund bezeichnet man die Regeln auch als *Produktionen*. Mittels einer Grammatik kann aber auch ein vorgegebenes Wort w_2 daraufhin überprüft werden, ob es ein syntaktisch korrektes Element der Sprache $\mathcal{L}$ ist. Diesen zweiten Vorgang nennt man *Parsing*. Zur Unterstützung beider Prozesse kann auf eine Vielzahl von Standardalgorithmen zurückgegriffen werden.

Nach Chomsky [138] lassen sich formale Grammatiken unter anderem in *kontextfreie* (CFG, context free grammar) und *kontextsensitive* (CSG, context sensitive grammar) Grammatiken einteilen. Eine kontextfreie Grammatik besteht aus Regeln, bei denen eine Variable auf der linken Regelseite unabhängig vom Kontext, in dem diese Variable auftritt, ersetzt werden kann. Demgegenüber hat bei einer kontextsensitiven Grammatik die lokale Umgebung einer Variablen einen signifikanten Einfluss auf deren Interpretation. In der vorliegenden Arbeit werden zur formalen Beschreibung verschiedener struktureller Zusammenhänge ausschließlich kontextfreie Grammatiken verwendet. Für die Darstellung wird analog zu der Beschreibung der Benutzerereignisse in Kapitel 4 auf die erweiterte Backus-Naur-Form (EBNF) [171] zurückgegriffen.

5.3.2 Abstrakte Repräsentation

In der Systemarchitektur erfolgt die Modellierung der einzelnen Informationsanteile mittels einer kontextfreien Grammatik. Die Terminalsymbole dieser Grammatik repräsentieren die kleinste signifikante Informationseinheit im Hinblick auf den semantischen Fusionsprozess. Sie bilden die atomare semantische Basiseinheit. Man bezeichnet sie daher als *Semune* und verwendet die Abkürzung S als das entsprechende Referenzsymbol. Der Begriff Semun steht abkürzend für *semantic unit* und ist usprünglich in den Arbeiten von Müller und Stahl [104, 157] im Rahmen der Interpretation natürlich-sprachlicher Äußerungen eingeführt worden. Mittels einzelner Semune bzw. darauf aufbauenden Semunfolgen lassen sich sowohl einfache Ereignisse als auch die Inhalte komplexer Datenstrukturen modellieren.

Im Sinne der formalen Sprachen werden die einzelnen Informationsanteile vollständig durch eine kontextfreie Sprache beschrieben. Aus der Sicht der Integrationsaufgabe weisen die Daten jedoch strukturelle Abhängigkeiten auf. So hängt beispielsweise im Rahmen eines Entscheidungsdialoges die auszuführende Aktion nicht nur von der Benutzereingabe sondern auch vom aktuellen Systemzustand ab. Diese Abhängigkeit wird als „Kontextsensitivität im Sinne der Funktionalität“ bezeichnet. Die beiden Arten der Kontextsensitivität stehen zwar in einem gewissen Zusammenhang, implizieren sich jedoch nicht notwendigerweise gegenseitig.

Die Interpretation multimodaler Daten ist hochgradig kontextspezifisch. Abstrahiert man von dem eigentlichen Fusionsprozess, dann kann aufgrund der paramodalen Repräsentation der einzelnen Informationsanteile die komponentenspezifische Beschreibung der Daten weiterhin durch eine kontextfreie Grammatik erfolgen. Die formale Sprache dient lediglich als Kommunikationsformalismus, um in einer einheitlichen Form Daten zwischen den einzelnen Systemmodulen auszutauschen. Da die kontextsensitiven Sprachen eine Obermenge der kontextfreien Sprachen bilden, kann die kontextfreie Grammatik auch in einer kontextsensitiven Form notiert werden. Algorithmisch und notationstechnisch ist die kontextfreie Form jedoch deutlich einfacher zu behandeln.

Übertragen auf die Systemarchitektur repräsentieren Semune auf der Eingabeseite modalitätenspezifische Benutzereingaben, Änderungen der Kontextdaten und Zustandsinformationen des Integrator-Moduls. Während ein einzelnes Semun eine in sich abgeschlossene Informationseinheit beschreibt, stellt ein Wort eine logisch zusammenhängende Struktur von mehreren Semunen dar, die den syntaktischen Anforderungen der zugrunde liegenden Grammatik genügt. Auf der Ausgabeseite modellieren Semune Funktionalitäten der Zielapplikationen, die durch den multimodalen Fusionsprozess angesteuert werden können.

5.3.3 Konvertermodule

Im Folgenden wird ein System aus insgesamt ρ einzelnen Modulen betrachtet, die an einen zentralen Integrator Φ angeschlossen sind. Mit Bezug auf Abbildung 5.2 können diese Module drei Kategorien zugeordnet werden: Erkennermodule ($\mathcal{M}_R$), Kontextmodule ($\mathcal{M}_K$) und Applikationsmodule ($\mathcal{M}_A$). Der spezielle Typ eines Moduls $M_i \in \mathcal{M}$ mit $\mathcal{M} = \mathcal{M}_R \cup \mathcal{M}_K \cup \mathcal{M}_A$ mit $i \in \{1..\rho\}$ spielt in der Betrachtung jedoch nur eine untergeordnete Rolle. Die Informationen müssen zunächst in ein gemeinsames Datenformat transformiert werden, damit das Integratormodul sie überhaupt verarbeiten kann.

Jedes Modul M_i verfügt über ein individuelles, proprietäres Vokabular $\mathcal{V}_i$ an Funktionalitäten, die von außen sichtbar sind und über eine dedizierte Schnittstelle den anderen Modulen zur Verfügung gestellt werden können. Der Umfang des Vokabulars für das Modul M_i wird mit υ_i bezeichnet und kann von Modul zu Modul zum Teil deutlich divergieren. Durch speziell angepasste Konvertermodule W_i wird das modulspezifische Vokabular $\mathcal{V}_i$ auf eine kontextfreie formale Sprache $\mathcal{L}_i$ abgebildet, wobei ς_i die Anzahl der Wörter und σ_i die Anzahl der Semune in $\mathcal{L}_i$ angibt. Die entsprechende Abbildungsfunktion wird mit Ω bezeichnet:

$$\Omega_i : \mathcal{V}_i \rightarrow \mathcal{L}_i \quad \text{mit} \quad i \in \{1..\rho\} \tag{5.8}$$

Dieser modulspezifische Konvertierungsmechanismus funktioniert nur in eine Richtung eindeutig. Zu jedem Wort $w \in \mathcal{L}_i$ existiert ein Element $v \in \mathcal{V}_i$ mit $\Omega_i(v) = w$ (Surjektivität). Durch eine Verkleinerung der Zielmenge $\mathcal{V}_i$ ist es immer möglich, die Abbildung Ω_i surjektiv zu machen. Eine weitere Eigenschaft der Konvertierung ist, dass unterschiedliche Elemente $v_1, v_2 \in \mathcal{V}_i$ mit $v_1 \neq v_2$ nicht notwendigerweise auf verschiedene Wörter $w_1, w_2 \in \mathcal{L}_i$ mit $w_1 \neq w_2$ der kontextfreien Grammatik abgebildet werden müssen, d.h. Ω_i ist nicht injektiv. Der Umfang des proprietären Vokabulars stimmt daher zumeist nicht mit der Anzahl der Wörter in der formalen Sprache überein und es gilt $\upsilon_i \geq \varsigma_i$. Da sich Wörter allgemein aus Symbolen zusammensetzen gilt zudem die Beziehung $\sigma_i \leq \varsigma_i$ für alle $i \in \{1..\rho\}$.

Der multimodale Integrator operiert ausschließlich auf der Abstraktionsebene der kontextfreien formalen Sprachen $\mathcal{L}_{CFG}$. Im einfachsten Fall ist die übergeordnete formale Sprache $\mathcal{L}_I$ des Integrators für alle Systemmodule identisch, d.h. es gilt $\mathcal{L}_I = \mathcal{L}_i = \mathcal{L}_j$ für alle $i, j \in \{1..\rho\}$ mit $i \neq j$. Da die einzelnen Module neben unterschiedlich großen Vokabularien zumeist auch über verschiedene Leistungscharakteristika verfügen, ergibt sich $\mathcal{L}_I$ allgemein als Vereinigungsmenge der einzelnen formalen Sprachen. Die Anzahl der zulässigen Wörter bzw. Symbole in $\mathcal{L}_I$ bewegt sich je nach Überlagerungsgrad zwischen einem modulbezogenen Maximum und der Summe sämtlicher modulspezifischen Anteile.

$$\mathcal{L}_I = \bigcup_{i=1}^{\rho} \mathcal{L}_i \quad \text{und} \quad max\left(\varsigma_i\right) \leq \varsigma_I \leq \sum_{i=1}^{\rho} \varsigma_i \quad \text{und} \quad max\left(\sigma_i\right) \leq \sigma_I \leq \sum_{i=1}^{\rho} \sigma_i \tag{5.9}$$

Systemtechnisch wird die Abbildung Ω_i durch eine extern spezifizierbare Auflistung der Funktionalitäten und der dazugehörigen Wörter der formalen Sprache realisiert. Jedem $v_x \in \mathcal{V}_i$ wird direkt ein Wort $w_y \in \mathcal{L}_i$ zugeordnet. Wegen der tabellarischen Anordnung bezeichnet man diese Struktur als Lookup-Tabelle (LUT). Die Erstellung dieser Tabellen stellt den wesentlichen Schritt bei der Anbindung neuer Systemmodule dar. Der abstrakte Zugriff bietet einen weiteren Vorteil. Bereits durch die Modifikation der Tabellen kann das Systemverhalten auf einfache Weise verändert werden, ohne dass in den Programm-Code eingegriffen werden muss.

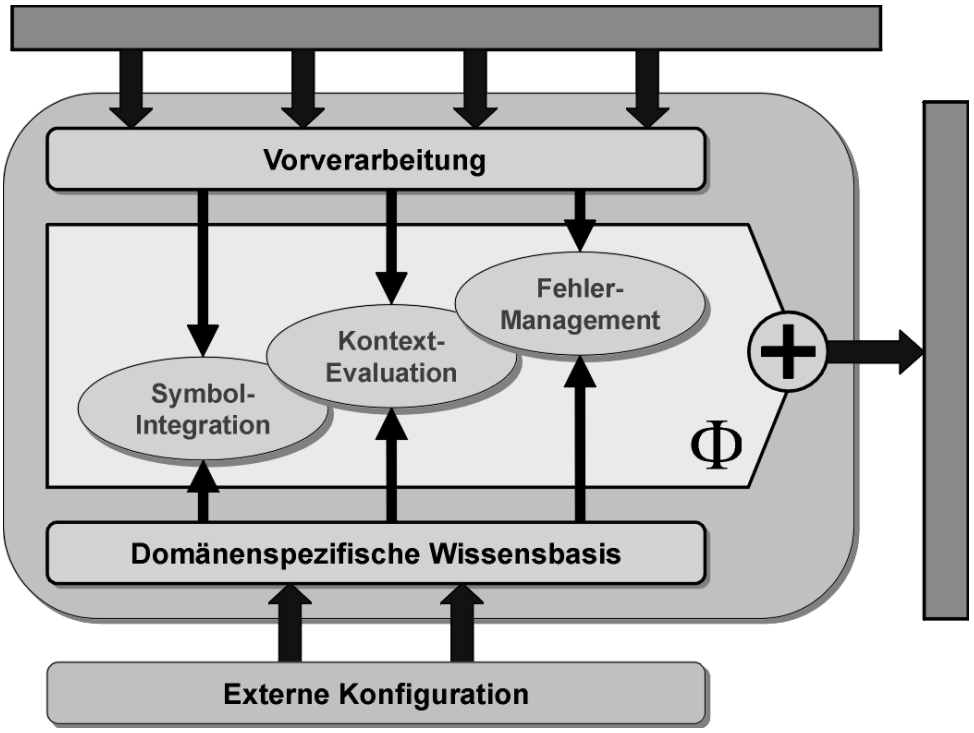


Abb. 5.3: Komponenten des multimodalen Integratormoduls

5.4 Komponenten des Integrators

Die Integration der multimodalen Informationsanteile erfolgt auf einem hohen semantischen Abstraktionsniveau. Abbildung 5.3 visualisiert die strukturellen Komponenten des multimodalen Integratormoduls. Der Intentionsdekoder Φ setzt sich aus verschiedenen, sequentiell ablaufenden Prozessen zusammen: Vorverarbeitung (Φ_V), Symbolfusion (Φ_S), Kontextevaluation (Φ_K) und schließlich Fehlermanagement (Φ_F). Ein besonderes Merkmal der Architektur ist, dass der algorithmische Kern des Integrators domänenübergreifend verwendet werden kann. Die Anpassung an eine konkrete Anwendung und an die in dieser Umgebung existierenden Randbedingungen erfolgt über die Konfiguration einer domänenspezifischen Wissensbasis, auf die die einzelnen Integrationskomponenten zugreifen.

5.4.1 Vorverarbeitung

Der erste Schritt im Rahmen der Integration besteht darin, in dem kontinuierlichen multimodalen Informationsstrom für den Fusionsprozess relevante Ereignisse zu identifizieren (Spotting-Phase). Aus den durchgeführten Benutzerstudien geht klar hervor, dass zusammengehörige Benutzereingaben zum einen innerhalb einer klar definierten Zeitspanne auftreten und zum anderen strukturelle Abhängigkeiten aufweisen.

Analog zu den Ausführungen in Abschnitt 4.2.4 kann mit jedem Ereignis $E \in \mathcal{E}$ eine komponentenspezifische Startzeit $t_s(E)$ und eine Endzeit $t_e(E)$ assoziiert werden. Die zeitliche Dauer von E wird als $T(E)$ bezeichnet und berechnet sich gemäß Gleichung 4.20 aus der Differenz zwischen $t_e(E)$ und $t_s(E)$. Zu Beginn der Anwendung werden alle Systemkomponenten auf eine

global einheitliche Systemzeit synchronisiert. Die Zeitstempel zweier Ereignisse $E_a, E_b \in \mathcal{E}$ mit $a \neq b$ aus unterschiedlichen Informationsquellen können damit absolut verglichen werden. Der Prozess der Synchronisation wird in Abschnitt 5.5.2 erklärt.

Das Ergebnis des Spottings ist ein Intervall $\mathcal{C}$ von zusammengehörigen Ereignissen. Diese Menge bezeichnet man auch als *Integrationscluster* oder kurz als *Cluster*. Im Folgenden wird ein Intervall aus drei Ereignissen $\mathcal{C} = \{E_a, E_b, E_c\}$ betrachtet. Die einzelnen Ereignisse sind nach ihren Startzeiten geordnet, d.h. $t_s(E_a) \leq t_s(E_b) \leq t_s(E_c)$. Sei $t_e(\mathcal{C}) = max(t_e(E_a), t_e(E_b), t_e(E_c))$ absolut betrachtet die Endzeit des Clusters. Auf diese Weise kann C auch durch die beiden Zeitmarken $t_s(C) = t_s(E_a)$ und $t_e(\mathcal{C})$ beschrieben werden. Die Dauer des Intervalls wird mit $T(C)$ angegeben und ergibt sich mit Bezug zu Gleichung 4.21 aus der Zeitdifferenz der absoluten Zeitmarken:

$$T(\mathcal{C}) = t_e(\mathcal{C}) - t_s(\mathcal{C}) = t_e(\mathcal{C}) - t_s(E_a) \tag{5.10}$$

Betrachtet wird im Folgenden ein spezifisches Intervall C_E mit den beiden absoluten Zeitmarken t_a und t_b und $t_a < t_b$. Sei $\mathcal{C}_E(t_1, t_2) = \{E_1, E_2, \ldots, E_n\}$ eine Menge von n zeitlich geordneten, unterscheidbaren Ereignissen mit $E_i \in \mathcal{E}$ für alle $i \in \{1..n\}$. Im Spottingprozess werden aus dieser Menge m Elemente $\{E'_1, E'_2, \ldots, E'_m\}$ mit $E'_a \neq E'_b$ für alle $a \neq b$ und $E'_a, E'_b \in \mathcal{C}_E$ ausgewählt und in das Cluster $\mathcal{C}_S$ übertragen. Jedes Ereignis $E \in \mathcal{C}_E$ wird dahingehend überprüft, ob es in Relation zu den anderen Ereignissen verschiedene zeitliche und semantische Kriterien erfüllt. Für den weiteren Integrationsprozess ist es nicht immer notwendig, sämtliche Ereignisse aus $\mathcal{C}_E$ nach $\mathcal{C}_S$ zu übertragen. Das Intervall $\mathcal{C}_S$ ist eine Teilmenge von $\mathcal{C}_E$, wobei aufgrund der Vielzahl an Informationen im Allgemeinen von der Beziehung $m < n$ ausgegangen wird. Mathematisch lässt sich der Spottingprozess ebenfalls als eine Abbildung beschreiben.

$$\Phi_V : \mathcal{C}_E(t_a, t_b) \rightarrow \mathcal{C}_S(t_a, t_b) \quad \text{mit} \;\; t_a < t_b \;\; \text{und} \;\; |\mathcal{C}_S(t_a, t_b)| \leq |\mathcal{C}_E(t_a, t_b)| \tag{5.11}$$

Die Länge eines problemadäquaten Integrationsintervalls ist von mehreren Faktoren abhängig, z.B. der Anwendungsdomäne, der Systemumgebung und der Art und Anzahl der verfügbaren Eingabemodalitäten. Als grober Richtwert kann aufgrund verschiedener Untersuchungen (siehe Abschnitt 2.3.2 und Kapitel 4) ein oberer Grenzwert von vier Sekunden angesetzt werden. Ein dynamischer, anwendungsspezifischer Grenzwert kann durch eine detaillierte Auswertung des System- und des Benutzerkontextes bestimmt werden. Für den Fall, dass das entsprechende Kontextwissen nicht in expliziter Form vorliegt, wird ein konstanter Wert angenommen.

Bei der Entwicklung des Spotting-Verfahrens muss berücksichtigt werden, dass die einzelnen Informationsanteile mit unterschiedlichen Verzögerungen am Integrator ankommen. Die Zeitunterschiede werden beispielsweise durch divergierende Kanallaufzeiten oder durch Aufwände vorverarbeitender Klassifikationsmodule verursacht. Daher muss in Ergänzung zu $\mathcal{C}_S(t_a, t_b)$ zusätzlich ein weiteres Zeitfenster $\mathcal{C}_I(t_c, t_d)$ mit $t_a \leq t_c$ und $t_b \leq t_d$ bestimmt werden, innerhalb dessen die im Laufe von $\mathcal{C}_S$ erzeugten Ereignisse am Integrator eintreffen. Man bezeichnet $\mathcal{C}_I$ als ein systemspezifisches *Integratorcluster*. In Abbildung 5.4 sind auf der oberen Zeitachse die Ereignisse in der Form dargestellt, wie sie Integrator eintreffen ($\mathcal{C}_I$). Im Vergleich dazu sind auf den unteren Achsen die Ereignisse mit den Anfangs- und Endzeiten im realen Spottingcluster $\mathcal{C}_S$ zu sehen.

Betrachtet man das Ereignis E_3, so stellt man fest, dass der Endzeitpunkt vor dem von E_1 und E_2 liegt. Am Integrator trifft es zeitlich aber erst nach den beiden anderen Ereignissen ein, obwohl E_2 sogar erst nach dem Endzeitpunkt von E_3 generiert wurde. Dies ist darauf

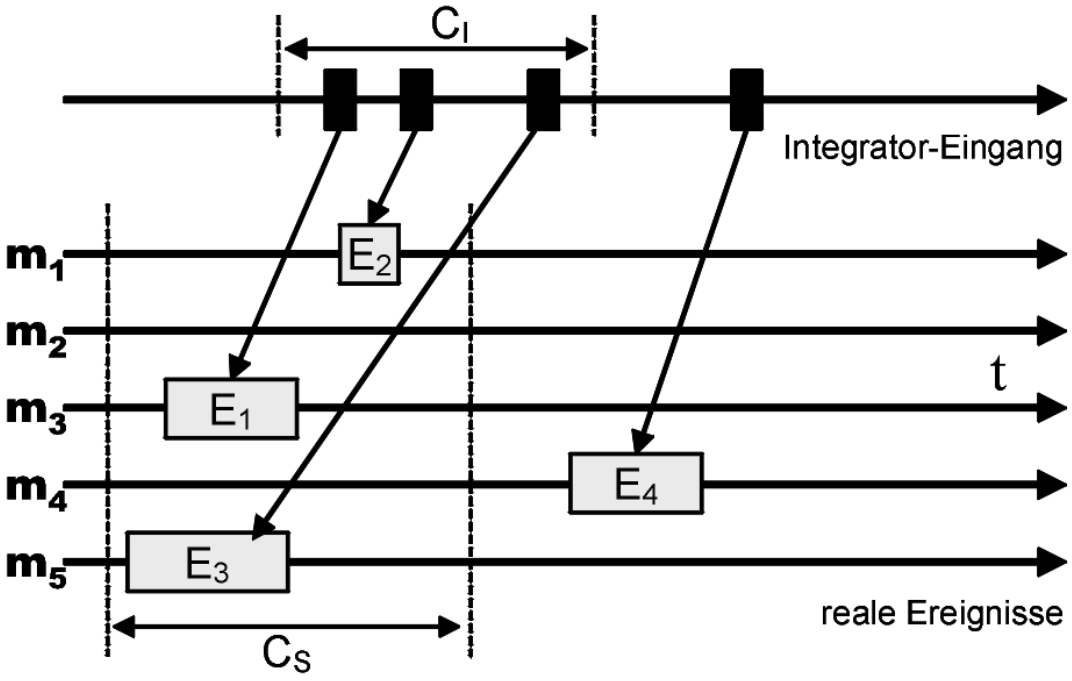


Abb. 5.4: Zusammenhang der entstehenden Verzögerungen im Integratorcluster $\mathcal{C}_I$ (obere Zeitleiste) bezogen auf das reale Cluster $\mathcal{C}_S$ (untere Zeitachsen) im Spottingprozess

zurückzuführen, dass in diesem fiktiven Beispiel das verwendete Erkennermodul die höchste Verarbeitungszeit aufweist. Bei dem Ereignis E_2 fällt demgegenüber kaum Verarbeitungszeit an und das Ereignis wird lediglich um die Kanallaufzeit verzögert. Dieses Beispiel weist auf ein zentrales Problem bei der zeitlichen Segmentierung von Realdaten hin. Das reale Spottingcluster $\mathcal{C}_S$ muss durch entsprechende Anpassungen an die aktuelle Anwendungsumgebung auf das systemspezifische Integratorcluster $\mathcal{C}_I$ umgerechnet werden.

Für das Spotting sind prinzipiell immer nur die Ereignisse von Interesse, die innerhalb des realen Zeitclusters $\mathcal{C}_S$ liegen. Bei einem zu kurz gewählten Intervall $\mathcal{C}_I$ könnten aber einzelne Ereignisse keine Beachtung finden. Wenn beispielsweise in Abbildung 5.4 das Intervall $\mathcal{C}_I$ vor dem Auftreten des Ereignisses E_3 enden würde, könnten hier nur E_1 und E_2 berücksichtigt werden. Aus diesem Grund muss $\mathcal{C}_I$ unter Einbezug der modulspezifischen Verarbeitungs- und Kanallaufzeiten so dimensioniert werden, dass zumindest für einen Großteil der auftretenden Situationen sämtliche innerhalb des realen Intervalls $\mathcal{C}_S$ abgegebenen Befehle in den Spottingprozess integriert werden können.

Diese Betrachtungen sind vor allem bei der Implementierung des realen Systems von essentieller Bedeutung. In der Entwicklungsphase wird zur Vereinfachung angenommen, dass die einzelnen Benutzerbefehle verzögerungsfrei und in der richtigen Reihenfolge am Integrator eintreffen, d.h. $\mathcal{C}_S$ und $\mathcal{C}_I$ stimmen sowohl bzgl. der Längen ($T(\mathcal{C}_S) = T(\mathcal{C}_I)$) als auch bzgl. der Anfangs- und Endzeiten vollständig überein ($t_a = t_c$ und $t_b = t_d$). Die Länge von $\mathcal{C}_S$ wird in der Praxis durch die Auswertung von empirischem Wissen aus Benutzerstudien bestimmt und an den Integrator in Form von Kontextinformationen übermittelt.

5.4.2 Symbolfusion

Das Spektrum der unterschiedlichen Eingabe-Ereignisse, die vom Integratormodul verarbeitet werden können, setzt sich aus zwei Kategorien zusammen. Auf der einen Seite steht die erkennergestützte Auswertung von intentionalen Benutzereingaben (E^R) und auf der anderen Seite die Berücksichtigung von zusätzlichen Informationen aus den Kontextmodulen (E^K). Nach dem modularen Ansatz aus Abbildung 5.3 laufen die dazugehörigen Integrationsprozesse sequentiell nacheinancher ab. Die Struktur der Ereignisse lässt sich wie folgt ausdrücken:

$$<E> \;::=\; <E^R> \;|<E^K> \tag{5.12}$$

$$<E^K> \;::=\; <E^{K(B)}> \;|<E^{K(S)}> \;|<E^{K(U)}> \tag{5.13}$$

Durch die Trennung von wichtigen und weniger wichtigen Ereignissen stellt das Spotting im Hinblick auf den gesamten Integrationsprozess die erste Stufe einer Informationsreduktion dar. In den nachfolgenden drei Fusionsstufen werden die in dem Spottingcluster C_S zeitlich und semantisch assoziierten Ereignisse auf potentielle Systemaktionen abgebildet. Ein solches Integrationsergebnis wird als *Aktionshypothese* bezeichnet und, bezogen auf die Symbolebene mit H_S abgekürzt. Die einzelnen Integrationsstufen zeichnen sich durch den Einbezug unterschiedlicher Informationen aus.

Auf der untersten Ebene Φ_S werden ausschließlich Benutzereingaben verarbeitet. Sie stellen im Hinblick auf die Interpretation der Benutzerintention die wichtigsten Informationsanteile dar. In vielen Fällen kann bereits aufgrund einer Kombination der erkennergestützten Benutzereingaben eine adäquate Systemreaktion erzeugt werden. Durch die Ausblendung der Kontextinformationen wird die Komplexität des Integrationsraums weiter reduziert. Dies ermöglicht in reduziertem Umfang eine extrem performante Umsetzung des Intentionsdekoders.

Sei $\mathcal{C}_S = \{E_1, E_2, ..., E_{\gamma(E)}\}$ mit $E_i \in \mathcal{E}$ für alle $i \in \{1..\gamma(E)\}$ eine Menge von $\gamma(E)$ Ereignissen in einem Spottingcluster. Als Vorbereitung für die Symbolfusion wird aus dieser Menge ein Cluster $\mathcal{C}_R$ erzeugt, in dem ausschließlich erkennergestützte Benutzereingaben enthalten sind. Genügt ein Element $E_i \in \mathcal{C}_S$ der Bedingung $E_i \in \mathcal{E}^R$, so wird es in die Menge $\mathcal{C}_R$ übertragen. Das Cluster $\mathcal{C}_R = \{E'_1, E'_2, ..., E'_{\gamma(R)}\}$ mit $E'_j \in \mathcal{E}^R$ und $j \in \{1..\gamma(R)\}$ ist eine Teilmenge von $\mathcal{C}_S$ mit insgesamt $\gamma(R)$ Elementen, wobei $\gamma(R) \leq \gamma(E)$ ist. Formal lässt sich der symbol-orientierte Teil des Intentionsdekoders wie folgt beschreiben:

$$\Phi_S : E'_1 \oplus E'_2 \oplus ... \oplus E'_{\gamma(R)} \rightarrow \mathcal{H}_S \quad \text{mit} \quad E'_j \in \mathcal{C}_R \;\wedge\; j \in \{1..\gamma(R)\} \tag{5.14}$$

Für die Realisierung der Abbildung Φ_S können sowohl regel-basierte als auch stochastische Verfahren eingesetzt werden [94]. Bei beiden Ansätzen entstehen im Rahmen des Integrationsprozesses mehrere Aktionshypothesen. Die Auflistung der einzelnen Alternativen $\mathcal{H}'_S$ ist ein Element der Potenzmenge von $\mathcal{H}_S$ und läßt sich durch eine eigene Menge beschreiben:

$$\mathcal{H}'_S = \{H_{S,1}, H_{S,2}, \ldots, H_{S,\lambda(S)}\} \quad \text{mit} \quad \mathcal{H}'_S \in \mathcal{H}^*_S \quad \text{und} \quad \lambda(S) \in \mathbb{N} \tag{5.15}$$

Dabei gibt $\lambda(S)$ die Anzahl der generierten Hypothesen auf der Symbolebene an. Zur besseren Beurteilung wird für jedes $H_S \in \mathcal{H}'_S$ ein Konfindenzmaß $Q_S(H_S)$ berechnet, mittels dem sich die Qualität der individuellen Ergebnisse quantitativ vergleichen lässt. Die Elemente in $\mathcal{H}_S$ sind nach diesen Bewertungen absteigend sortiert, d.h. es gilt die Beziehung $Q_s(H_{S,k}) \geq Q_S(H_{S,l})$ für alle $k < l$ und $H_{S,k}, H_{S,l} \in \mathcal{H}_S$.

Der Intentionsdekoder kann extern parametrisiert werden. Vergleichbar mit den Konvertermodulen in Abschnitt 5.3.3 erfolgt die Konfiguration über nachladbare Tabellen. Der Inhalt dieser Tabellen wird als *domänenspezifische Wissensbasis* bezeichnet, durch die der generisch ausgelegte Integrationskern an die aktuelle Anwendungsdomäne und die entsprechenden Randbedingungen angepasst werden kann. Die einzelnen Parameter sind für einen effizienten und effektiven Betrieb des Integrators unerlässlich.

Im Hinblick auf die Integration der Benutzerereignisse besteht die externe Konfiguration im Wesentlichen aus drei unterschiedlichen Listen. In der ersten Liste sind in Form einer kontextfreien Grammatik die vorkommenden Ereignisse und die strukturellen Zugehörigkeiten der einzelnen Semune aufgeschlüsselt. Die zweite Liste beinhaltet eine Aufzählung der gültigen Kombinationsmöglichkeiten und in der dritten Liste sind Informationen darüber abgelegt, ob einzelne Aktionshypothesen in reale Aktionen umgewandelt und von den Applikationsmodulen ausgeführt werden können.

5.4.3 Kontextevaluierung

In der Symbolfusion Φ_S werden lediglich intentionale Benutzereingaben betrachtet. Eine Revalidierung der Aktionshypothesen vor dem Hintergrund der aktuell verfügbaren Kontextinformationen erfolgt erst auf der zweiten Ebene des Intentionsdekoders Φ_K. Diese Integrationsstufe besteht aus zwei getrennten Phasen. Zuerst werden die abstrakten Eingabeereignisse in den jeweiligen Kontexten interpretiert und in applikationsspezifische Meta-Ereignisse transformiert. Diese internen Ereignisse werden dann zu den symbol-orientierten Aktionshypothesen in Beziehung gesetzt und auf semantisch höherwertige Aktionshypothesen $\mathcal{H}_K$ abgebildet.

Im Folgenden wird beispielhaft für den Systemkontext $K(S)$ die situative Interpretation der Eingabeereignisse erklärt. Sei $\mathcal{Z}_S = \{Z_{S,1}, Z_{S,2}, ..., Z_{S,\zeta(S)}\}$ mit $\zeta(S) \in \mathbb{N}$ eine Auflistung der möglichen Systemzustände und $E \in \mathcal{C}_S$ ein beliebiges Eingabeereignis aus dem Spottingintervall. Durch die Abbildung $\Omega_{Z(S)}$ wird dem abstrakten Ereignis E in Abhängigkeit von dem aktuellen Systemzustand $Z_S \in \mathcal{Z}_S$ ein systemspezifisches Meta-Ereignis zugewiesen. Das Ergebnis dieser Abbildung bezeichnet man als *Integrationstoken* und kürzt es mit X_S ab. Die Menge aller systembezogenen Integrationstokens wird mit $\mathcal{X}_S$ angegeben.

$$\Omega_{Z(S)} : E \oplus Z_S \rightarrow \mathcal{X}_S \quad \text{mit} \quad E \in \mathcal{C}_S \tag{5.16}$$

Systemtechnisch wird die Abbildung $\Omega_{Z(S)}$ durch eine extern spezifizierte Wissensbasis realisiert, die auf einen internen Zustandsautomaten des Integrator-Moduls zugreift. Die funktionalen Zusammenhänge sind in Form einer Matrix $D(\Omega_{Z(S)})$ kodiert, bei der in der ersten Zeile die potenziellen Eingabeereignisse und in der ersten Spalte die möglichen Systemzustände repräsentiert sind. Dabei gibt η die Gesamtanzahl aller möglichen Ereignisse an.

$$D(\Omega_{Z(S)}) = \begin{pmatrix} & E_1 & E_2 & \cdots & E_n \\ Z_{S,1} & X_S(E_1, Z_{S,1}) & X_S(E_2, Z_{S,1}) & \cdots & X_S(E_\eta, Z_{S,1}) \\ Z_{S,2} & X_S(E_1, Z_{S,2}) & X_S(E_2, Z_{S,2}) & \cdots & X_S(E_\eta, Z_{S,2}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ Z_{S,\zeta(S)} & X_S(E_1, Z_{S,\zeta(S)}) & X_S(E_2, Z_{S,\zeta(S)}) & \cdots & X_S(E_\eta, Z_{S,\zeta(S)}) \end{pmatrix} \tag{5.17}$$

Die restlichen Matrixelemente enthalten die systemspezifischen Integrationstokens. Vom Abstraktionsgrad her sind sowohl die Zustände als auch die Tokens systemspezifisch, d.h. sie haben zumeist ein direktes Pendant als abstrakte Datentypen im Integrator. Im Gegensatz dazu stellen die Ereignisse eher eine konzeptuelle Beschreibungsebene dar.

Analog zu Gleichung 5.16 existieren natürlich auch Abbildungen $\Omega_{Z(B)}, \Omega_{Z(U)}$ und entsprechende Matrizen $D(\Omega_{Z(B)})$ und $D(\Omega_{Z(U)})$ für die Interpretation der Eingabeereignisse vor dem Hintergrund des aktuellen Benutzerkontextes $K(B)$ und des Umweltkontextes $K(U)$. Sei $\mathcal{Z}_B = \{Z_{B,1}, Z_{B,2}, ..., Z_{B,\zeta(B)}\}$ mit $\zeta(B) \in \mathbb{N}$ eine Auflistung der möglichen Benutzerkontexte und $\mathcal{Z}_U = \{Z_{U,1}, Z_{U,2}, ..., Z_{U,\zeta(U)}\}$ mit $\zeta(U) \in \mathbb{N}$ eine Beschreibung für die Menge der umweltspezifischen Kontextinformationen. Die Transformation der abstrakten Ereignisse $E \in \mathcal{C}_S$ lässt sich wie folgt beschreiben:

$$\Omega_{Z(B)} : E \oplus Z_B \rightarrow \mathcal{X}_B \quad \text{mit} \quad E \in \mathcal{C}_S \tag{5.18}$$

$$\Omega_{Z(U)} : E \oplus Z_U \rightarrow \mathcal{X}_U \quad \text{mit} \quad E \in \mathcal{C}_S \tag{5.19}$$

In Abhängigkeit von der jeweiligen Kontextkategorie werden die Ergebnisse der obigen Abbildungen mit $X_B \in \mathcal{X}_B$ und $X_U \in \mathcal{X}_U$ bezeichnet. Die Menge aller Integrationstokens ergibt sich aus der Vereinigung der einzelnen, kontextspezifischen Integrationstokens.

$$\mathcal{X} = \mathcal{X}^S \cup \mathcal{X}^B \cup \mathcal{X}^U \tag{5.20}$$

Da die einzelnen Matrizen auch im laufenden Betrieb jederzeit selektiv nachgeladen werden können, bieten sie einen ersten Ansatzpunkt, um dynamische Anpassungen des Interaktionsvokabulars vorzunehmen zu können. Eine einfache Anwendung wären beispielsweise zwei unterschiedliche Matrixversionen für die Intepretation der Tastenbelegung einer haptischen Konsole. Je nachdem, über welche Erfahrungen der aktuelle Benutzer im Umgang mit dem System verfügt, könnten unterschiedliche Versionen erzeugt werden. Für einen unerfahrenen Nutzer wird jede Taste unabhängig vom aktuellen Bedienkontext belegt und bei Betätigung immer die gleiche Systemaktion ausgelöst. Demgegenüber könnte den einzelnen Tasten für einen Experten jeweils eine kontextspezifische Bedeutung zugewiesen werden. Diese zweite Version verlangt zwar einen gewissen Trainingsaufwand und eine entsprechende Repräsentation der Funktionalität im mentalen Modell des Benutzers, ermöglicht aber effektivere Systeminteraktionen.

In der zweiten Phase der Kontextevaluation werden für jedes Ereignis $E_i \in C_S$ mit $i \in \{1..\gamma(E)\}$ des ursprünglichen Spottingclusters die transformierten Integrationstokens X_S, X_B und X_U zu den verschiedenen symbol-orientierten Aktionshypothesen $\mathcal{H}'_S$ in Beziehung gesetzt und auf eine semantisch höherwertige Aktionshypothese H_K abgebildet.

$$\Phi_K \ : \ \mathcal{X}_S \oplus \mathcal{X}_B \oplus \mathcal{X}_U \oplus \mathcal{H}'_S \rightarrow \mathcal{H}_K \tag{5.21}$$

Das Ergebnis der Abbildung Φ_K besteht wiederum aus einer Menge von mehreren Aktionshypothesen $\mathcal{H}'_K = \{H_{K,1}, H_{K,2}, \ldots, H_{K,\lambda(K)}\}$ mit $\mathcal{H}'_K \in \mathcal{H}^*_K$, die entsprechend ihrer Bewertungsmaße $Q_K(H_{K,i})$ mit $i \in \{1..\lambda(K)\}$ angeordnet werden. Eine Interpretation der Aktionshypothesen $\mathcal{H}_S$ und $\mathcal{H}_K$ im Hinblick auf potenzielle System- und Benutzerfehler erfolgt auf der dritten Integratorstufe Φ_F (siehe nächster Abschnitt). Letztendlich werden die Aktionshypothesen der einzelnen Integrationsphasen miteinander verbunden und in eine Sequenz von realen Aktionen umgewandelt. Diese Aktionen werden dann durch die Konvertermodule W_A in modulspezifische Funktionen umgewandelt und an die Ausgabemodule übertragen.

5.4.4 Fehlermanagement

Die Fehlerrobustheit eines technischen Systems hat einen wesentlichen, von der Domäne unabhängigen Einfluss auf die Akzeptanz des Systems und die Zufriedenheit des Benutzers. Bezogen auf die modulare Architektur in Abbildung 5.2 baut das Fehlermanagement auf den Aktionshypothesen der Symbolfusion und der Kontextevaluation auf und stellt damit die höchste Form der Informationsverarbeitung dar. Verglichen mit den beiden anderen Integrationsphasen wird nur kurz auf dieses Modul eingegangen. Eine umfassende, theoretisch fundierte Behandlung von Fehlerszenarien im multimodalen Mensch-Maschine-Dialog erfolgt in der Dissertation von McGlaun [93], die in Bezug auf die einzelnen Integrationsebenen eine komplementäre Ergänzung zu der vorliegenden Arbeit darstellt. Im Hinblick auf die Bedienung von Infotainmentsystemen im Automobil ist die Konzeption eines fehlerrobusten Systems zudem extensiv im Kooperationsprojekt FERMUS [82] untersucht worden.

Ein effektives Fehlermanagement besteht aus zwei wesentlichen Bereichen. Auf der einen Seite muss durch geeignete Design-Maßnahmen in der Konzeptionsphase eines Systems potenziellen Benutzer- und Systemfehlern vorgebeugt werden (*apriori-Fehlermanagement*). Im Vorfeld der eigentlichen Nutzung können so bereits eine Vielzahl von unterschiedlichen Fehlern präventiv behandelt und das Fehlerpotenzial insgesamt reduziert werden. Auf der anderen Seite müssen im realen Betrieb des Systems auftretende Fehler erkannt und im Kontext der aktuellen Bediensituation interpretiert werden. Um eine fehlerhafte Situation zu bereinigen, müssen anschließend geeignete Auflösungsstrategien eingeleitet werden (*aposteriori-Fehlermanagement*).

Das Fehlermanagementmodul ist aus den drei Submodulen *Merkmalsextraktion*, *Fehlerklassifikation* und *Fehlerbehandlung* aufgebaut. Die Aufteilung gewährleistet eine funktionale Trennung der Grundfunktionen Informationsbereitstellung, Fehleranalyse sowie die Auswahl einer geeigneten Fehlerauflösungsstrategie. Zwischen den Phasen sorgen spezifizierte Systemschnittstellen für einen reibungslosen Datentransfer. Die einzelnen Submodule sind lokal erweiterbar. Sie können aber auch vor dem Hintergrund wechselnder Anwendungsszenarien vollständig gegen andere Module ausgetauscht werden.

Für die exakte Definition eines Fehlers im Rahmen der Bedienung interaktiver Mensch-Maschine-Systeme existieren unterschiedliche Angaben. Nach der im FERMUS-Projekt verwendeten Terminologie spricht man allgemein von einem Fehler, wenn ein erwünschtes Ziel nicht erreicht wird und dafür kein Zufall verantwortlich ist. Dabei muss grundsätzlich zwischen benutzer- und systemseitigen Fehlern unterschieden werden. Die Ursachen für systemseitige Fehler sind vielfach. Typische Beispiele sind eine fehlerhafte Interpretation der Benutzereingabe, Timingprobleme der Erkennermodule, mangelnder Kontextbezug oder eine fälschliche Aktivierung einzelner Systemfunktionen. Menschliche Fehler sind im Vergleich dazu wesentlich schwieriger zu modellieren, da sie sowohl von benutzerspezifischen Einflüssen als auch von Einflüssen aus der Umgebung abhängig sind.

Im Rahmen der Merkmalsextraktion $F(M)$ werden die ursprünglichen Ereignisse im Spottinginterval $\mathcal{C}_S$, die transformierten Integrationstoken $\mathcal{X}$ sowie die Aktionshypothesen aus der Symbolfusion $\mathcal{H}'_S$ und der Kontextevaluation $\mathcal{H}'_K$ strukturell analysiert. Das Resultat dieser Phase besteht in der Bestimmung charakteristischer Fehlermerkmale $\mathcal{F}_M$, die in Form einer abstrakten Datenbasis den nachfolgenden Klassifikations- und Strategiemodulen zur Verfügung gestellt wird. Die Informationsanteile sind dazu in einer anwendungsspezifischen formalen Sprache $\mathcal{L}_{F(M)}$ definiert, die analog zu den anderen Systemkomponenten extern parametrisiert werden

kann. Typische Fehlermerkmale sind beispielsweise die Zeit zwischen einzelnen Benutzereingaben, die Anzahl der Befehlswiederholungen oder die Frequenz ausgewählter Zustandsänderungen. Für die Automobildomäne finden sich weitere Beispiele in [152]. Formal kann die Merkmalsextraktion wie folgt dargestellt werden:

$$\Phi_{F(M)} \; : \; \mathcal{C}_S \oplus \mathcal{X} \oplus \mathcal{H}'_S \oplus \mathcal{H}'_K \rightarrow \mathcal{F}_M \tag{5.22}$$

In der nachfolgenden Fehleranalyse $F(K)$ werden die einzelnen Merkmale zu den verschiedenen Fehlerkategorien in Beziehung gesetzt. Das Ziel besteht darin, eine quantitative Aussage generieren zu können, ob die gegebene Merkmalsfolge $\mathcal{F}'_M \in \mathcal{F}^*_M$ einen entsprechenden Fehlertyp modelliert. Neben den Merkmalen fließen auch die Eingangsdaten der Merkmalsextraktion direkt in die Fehlerbestimmung ein. Das Ergebnis der Fehleranalyse besteht aus einer Menge $\mathcal{F}'_K \in \mathcal{F}^*_K$, in der die anwendungsspezifischen Fehler nach ihrer Auftrittswahrscheinlichkeit geordnet sind. Die Bewertung eines Fehlers wird als *Error-Score* bezeichnet.

$$\Phi_{F(K)} \; : \; \mathcal{C}_S \oplus \mathcal{X} \oplus \mathcal{H}'_S \oplus \mathcal{H}'_K \oplus \mathcal{F}_M \rightarrow \mathcal{F}_K \tag{5.23}$$

Die einzelnen Fehlerkategorien sind hochgradig anwendungsspezifisch und können ebenfalls mit einer kontextfreien Grammatik beschrieben werden. Für jeden Fehlertyp gibt es eine spezialisierte Systemkomponente (*Error-Sentinel*), die den Ereignisraum nach charakteristischen Fehlermerkmalen untersucht und das Potenzial für das Vorliegen der Fehlerklasse $F(K) \in \mathcal{F}_K$ bestimmt. Dabei werden sowohl regel-basierte, als auch stochastische und hybride Klassifikationsansätze verwendet [152, 167]. Die Grundlage für die Fehleranalyse bilden die unterschiedlichen Eingabedaten aus den Erkenner- und Kontextmodulen sowie Meta-Informationen aus dem Integrationsprozess. Um vor dem Hintergrund dieser immensen Datenmenge unnötige Mehrfachberechnungen zu vermeiden sind die einzelnen Error-Sentinels untereinander verknüpft.

In der abschließenden Fehlerbehandlung wird für jeden Error-Score ein anwendungs- und situationsspezifischer Schwellwert festgelegt. Eine Überschreitung dieses Wertes bewirkt die Initiierung einer geeigneten Auflösungsstrategie. In Bezug auf den Schwellwert können mehrere Fehlerbehandlungsstufen unterschieden werden. Die Einstufung erfolgt wiederum durch eine externe Parametrisierung [152]. Auf diese Weise ist es möglich, bestimmte Error-Sentinels in der Fehlerauflösung zu priorisieren. Treten beispielsweise mehrere Fehler gleichzeitig auf, dann kann über die variablen Schwellwerte gesteuert werden, welche Fehlerauflösungsstrategie primär verfolgt werden soll.

Formal kann die dritte Stufe des Fehlermanagements wie folgt beschrieben werden. Auf Basis der extrahierten Fehlermerkmale $\mathcal{F}'_M$ und den gewichteten Fehlerkategorien $\mathcal{F}'_K$ wird situationsspezifisch eine geeignete Fehlerauflösungsstrategie $H_F \in \mathcal{H}_F$ ausgewählt. Im Hinblick auf die beiden vorhergehenden Integrationsstufen Φ_S und Φ_K bezeichnet man das Element aus dem Bildbereich ebenfalls als eine Aktionshypothese. Das Ergebnis der Abbildung besteht aus einer Menge $\mathcal{H}'_F \in \mathcal{H}^*_F$ mit $\mathcal{H}'_F = \{H_{F,1}, H_{F,2}, \ldots, H_{F,\lambda(F)}\}$ von $\lambda(F)$ alternativen Strategien, die wiederum nach Bewertungen absteigend sortiert sind.

$$\Phi_{F(S)} \; : \; \mathcal{F}_M \oplus \mathcal{F}_K \rightarrow \mathcal{H}_F \tag{5.24}$$

Bei den Fehlerauflösungsstrategien wird prinzipiell zwischen einstufigen und mehrstufigen Dialogstrategien unterschieden. Während bei einer einstufigen Strategie das System eine Nachfrage oder einen Hinweis generiert, auf den der Benutzer mit einer einzelnen Eingabe reagieren

kann, wird im Rahmen einer mehrstufigen Strategie ein komplexer Rückfragedialog initiiert, in dem der Benutzer schrittweise durch den Auflösungsprozess geführt wird. Speziell der zweite Ansatz bietet enormes Potenzial für eine Adaption an den aktuellen Benutzer und die momentane Umgebungssituation [158].

5.5 Modulkommunikation

5.5.1 Client-Server basierter Informationsaustausch

Der multimodale Informationsstrom wird von einem zentralen Integratormodul interpretiert und in entsprechende Systemaktionen umgesetzt. Auf physikalischer Ebene fungiert dieses Modul als zentraler Message-Broadcast-Server, über den die einzelnen Systemkomponenten bilateral kommunizieren. Angelehnt an eine klassische Client-Server Architektur werden die modulspezifischen Informationsanteile durch den Integrator weitergeleitet und an die relevanten Komponenten verteilt. Dieser modulare Systemaufbau und die Verwendung von plattformunabhängigen Implementierungsstandards ermöglichen eine Nutzung des Servers und der Clients in einer verteilten, heterogenen Rechnerumgebung.

Der Informationsautausch funktioniert über Text-Nachrichten, die in einem speziellen Format über eine Netzwerkschnittstelle übertragen werden. Die Kodierung der Nachrichten erfolgt auf Basis modulspezifischer, kontextfreier Grammatiken (siehe Abschnitt 5.3.3). Dabei werden die Informationen der angeschlossenen Erkenner-, Kontext- und Applikationsmodule durch spezielle Konvertermodule auf einer formalen Metasprache abstrahiert und als Text-Nachrichten dargestellt. Das Format einer Nachricht STR entspricht der Form:

$$<STR> \ ::= \ <TargetID><SourceID><MSG> \tag{5.25}$$

Dabei steht $TargetID$ für eine alphanumerische Zeichenfolge, über die das Zielmodul der Nachricht identifiziert werden kann. Analog wird in der Zeichenfolge $SourceID$ der Absender der Nachricht kodiert. Der eigentliche Inhalt der übermittelten Information ist in MSG enthalten. Die strukturellen Inhalte sind jeweils durch Leerzeichen getrennt. Jede Nachricht verfügt über eine eindeutige Identifikationsnummer und einen globalen Zeitstempel. Diese beiden Informationsanteile sind nicht explizit in STR kodiert, da sie den einzelnen Nachrichten erst beim Durchlauf des Integratormoduls zugeordnet werden.

Die Zugehörigkeit einer Nachricht zu einem Erkenner-, Kontext- oder Applikationsereignis kann prinzipiell durch einen Vergleich der Quellenangabe $SourceID$ mit einer applikationsspezifischen Wissensbasis ermittelt werden. Wie bereits weiter oben erwähnt wurde, werden in der Praxis zum Teil unterschiedliche Modultypen in einer einzelnen Komponente integriert. Um vor diesem Hintergrund eine effiziente Klassifikation des speziellen Nachrichtentyps vornehmen zu können wird die Struktur von MSG feiner aufgeschlüsselt.

$$<MSG> \ ::= \ <MSG\_Typ><MSG\_Content> \tag{5.26}$$

Durch die explizite Mitführung eines Nachrichtentyps MSG_Typ kann die Struktur des folgenden Nachrichteninhaltes $MSG\_Content$ stärker eingeschränkt werden. Ein wesentlicher Vorteil dieser Nomenklatur ist, dass der Aufwand zum Parsen der Nachrichten minimiert wird.

Es muss nicht mehr die komplette Zeichenfolge verarbeitet werden. Auf diese Weise können kategorisierte Informationseinheiten zwischen den einzelnen Modulen ausgetauscht werden. Eine Erweiterung um neue Informationsformate ist durch die Integration neuer Nachrichtentypen jederzeit möglich. Im Folgenden werden einige ausgewählte, bereits vordefinierte Nachrichtenformate näher erläutert.

$$< MSG > ::= < MS1 > | < MS2 > | < MS3 > | < MS4 > |... \quad (5.27)$$

$$< MS1 > ::= tue < TimeStamp >< UserEvent > \quad (5.28)$$

$$< MS2 > ::= tce < TimeStamp >< ContextEvent > \quad (5.29)$$

$$< MS3 > ::= ts < TimeStamp > [\ sync \,|\, status\ ] \quad (5.30)$$

$$< MS4 > ::= nd < Content > \quad (5.31)$$

In den ersten beiden Beispielen $MS1$ und $MS2$ werden jeweils Benutzer- bzw, Kontextereignisse gekoppelt mit einem Zeitstempel übertragen. Diese Zeitstempel beinhalten zusätzliche Informationen über die reale Erzeugungszeit der Ereignisse. Das dritte Beispiel $MS3$ veranschaulicht die Nachrichtenstruktur zum Auf- und Abbau der Netzwerkverbindungen bzw. zur Synchronisation der einzelnen Komponenten. Durch das letzte Beispiel $MS4$ wird der allgemeine Fall einer Nachricht dargestellt. Der Typ ist nicht explizit definiert und der Inhalt muss daher einen vollständigen Parsing-Prozess durchlaufen.

5.5.2 Synchronisation der Komponenten

Da bei der Integration multimodaler Informationsanteile aus unterschiedlichen Datenquellen die zeitliche Reihenfolge der einzelnen Ereignisse von fundamentaler Bedeutung ist, werden vor Beginn der Anwendung alle Systemkomponenten auf eine global einheitliche Systemzeit synchronisiert. Die Anfangs- und die Endzeitpunkte der einzelnen Interaktionen können bei genauer Kenntnis der Erkennungsdauer sowie der Laufzeiten der Nachrichten aus den Ankunftszeitpunkten am Integrator bestimmt werden.

Beim Start einer Systemkomponente wird die aktuelle Systemzeit t_{sys} des Rechners, auf dem der Client ausgeführt wird, als programminterne Startzeit t_{ini} festgelegt. Die aktuelle Programmzeit t_{ct} ergibt sich aus der Differenz von t_{ini} und t_{sys}. Um die zeitliche Synchronisation aller angeschlossenen Clients und deren Nachrichten zu ermöglichen, wird in einer Synchronisationsphase nach dem Start jedes Clients dessen Startzeit so angepasst, dass die aktuelle Programmzeit dieses Clients mit der des zentralen Integrators übereinstimmt. Der Integrator übernimmt dazu die Funktion eines globalen Synchronisations-Servers. Es besteht jederzeit die Möglichkeit, die Synchronisation der einzelnen Module erneut durchzuführen.

In dem momentan implementierten Synchronisationsverfahren wird die kritische Laufzeit der Antwort vom Server zum Client als unbekannte Zeit-Differenz nicht behandelt. Die Evaluierung der Nachrichten-Laufzeiten innerhalb verschiedener Systemumgebungen hat ergeben, dass bei einer normalen Auslastung in einem autarken Netzsegment die durchschnittlichen Laufzeiten in einem verteilten System deutlich unterhalb von 200ms liegen [152, 117]. Diese Annahme hat sich sowohl für homogene Windows-Architekturen als auch für heterogene Windows-/Linux-Architekturen bestätigt.

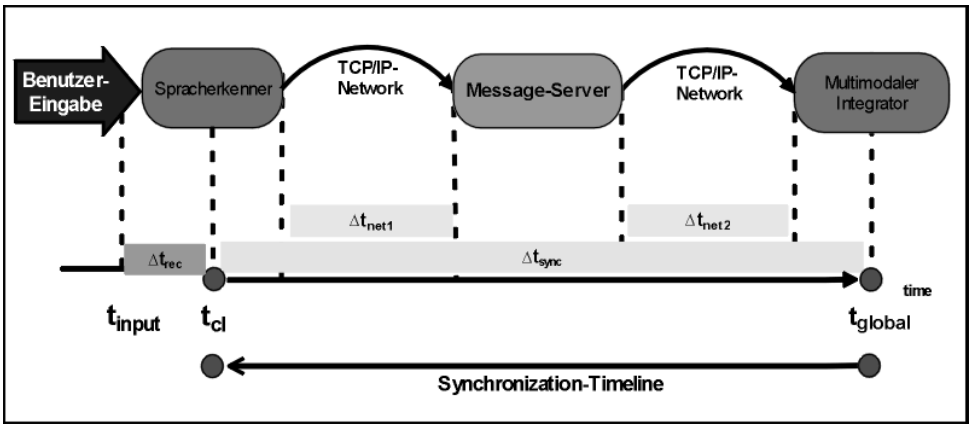


Abb. 5.5: Zeitabschnitte für die Übertragung einer Nachricht

Für den Synchronisationsprozess müssen unterschiedliche Laufzeiten betrachtet werden. Abbildung 5.5 veranschaulicht die Übermittlung einer Nachricht von einem Spracherkenner bis zum Empfang am Integratormodul. Die Client-Zeit t_{cl} ist aber auf Grund der durch ein Client-Polling angeforderten Synchronisation um die Synchronisations-Differenz Δt_{sync} kleiner als die globale Serverzeit t_{global}. Diese Differenz setzt sich aus den programminternen Verarbeitungszeiten und aus den Netzwerklaufzeiten Δt_{net1} und Δt_{net2} der Nachrichten zusammen. Wird beispielsweise vom Spracherkenner zum Zeitpunkt t_{cl} eine Benutzereingabe erkannt und an den Server geleitet, dann kann der genaue Eingabezeitpunkt t_{input} mit der Nachricht mitgeschickt werden. Für die Berechnung des Eingabezeitpunkts spielt auch die Vorverarbeitungszeit t_{rec} des Erkenners eine Rolle und muss entsprechend von der Client-Zeit abgezogen werden.

$$t_{input} = t_{cl} - \Delta t_{rec} = t_{global} - \Delta t_{rec} - \Delta t_{sync} \tag{5.32}$$

Zur Abschätzung der Synchronisationsdifferenz kann die Laufzeit der Nachrichten während der Synchronisationsphase benutzt werden. Diese entspricht der Hälfte der Zeit zwischen Anforderung und Antwort. Im optimalen Fall (Synchronisationsdifferenz ist bekannt) können nach der Synchronisation die Zeiten t_{cl} und t_{global} wechselseitig ineinander überführt werden. Vor der Synchronisation unterlagen diese beiden Zeiten keiner spezifischen Relation.

5.6 Systemtechnische Umsetzung

Im Hinblick auf die in Abschnitt 5.1.2 geforderte Performanz müssen die einzelnen Hardware- und Softwarekomponenten einer verteilten Anwendung optimal aufeinander abgestimmt werden. Eine besondere Bedeutung kommt der Kommunikation zwischen den einzelnen Modulen bzw. dem eingesetzten Kommunikationsprotokoll zu. In der vorliegenden Arbeit tauschen die Module ausschließlich symbol-orientierte Informationen aus. Um die Komponenten auch auf verteilten Systemen betreiben zu können, wird zu Lasten einer geringeren Effizienz auf systemgebundene Kommunikationsparadigmen wie Pipelining oder den Datenaustausch über Shared-Memory vollständig verzichtet. Statt dessen werden die Informationen über eine standardisierte Netzwerkschnittstelle übertragen.

Im einfachsten Fall setzt die Kommunikationsschnittstelle direkt auf TCP/IP-Sockets auf [41]. Jede Systemkomponente verfügt über eine spezielle Prozedur, mit der Text-Nachrichten beliebiger Länge auf den Socket geschrieben bzw. vom Socket gelesen werden können. Um die Netzauslaustung möglichst gering zu halten, können die einzelnen Nachrichten in Blöcke einer frei definierbaren Größe aufgeteilt und getrennt übertragen werden. Zusätzlich besteht die Möglichkeit, die Informationen nach verschiedenen Verfahren auf der Senderseite zu komprimieren und auf der Empfängerseite wieder verlustfrei zu rekonstruieren [117]. Im Laufe der Systementwicklung sind Schnittstellen sowohl für höhere Programmiersprachen wie C, C++, Java und Basic als auch für Rapid-Prototyping Umgebungen und Skriptsprachen wie Perl, Tcl/TK, Python und VRML entstanden. Die Portierung des Basissystems auf unterschiedliche Software-Umgebungen kann dadurch extrem vereinfacht werden.

Als Alternative zu dem direkten Zugriff auf TCP/IP-Sockets ist eine Middleware auf Basis von CORBA entwickelt und prototypisch umgesetzt worden [4]. CORBA ist eine Abkürzung für Common Object Request Brokerage Architecture [127, 151]. In der zugrunde liegenden Spezifikation wird ein offener Standard für die Kommunikation in heterogenen Systemen beschrieben. Der Ansatz besteht darin, dass mehrere Server in einer Netzwerkstruktur mehreren Clients verschiedene Objekte zur Verfügung stellen, auf die jeweils ohne detaillierte Kenntnisse der Lokalisation zugegriffen werden kann. In der vorliegenden Arbeit wird die C++ basierte Implementierung *ommiORB* [89] verwendet.

Um die aufwändige Startprozedur und Koordination der einzelnen Systemkomponenten zu erleichtern, wurde ein zusätzliches Hilfsmodul entwickelt (*LaunchPad* [142]), welches nach dem Laden einer zentralen Konfigurationsdatei sämtliche Komponenten mit den passenden Parameterkonfigurationen startet und untereinander verbindet. Das graphische Tool verfügt über eine Reihe von verschiedenen Funktionalitäten. Einzelne Komponenten können beispielsweise durch einen einfachen Button-Click gestartet bzw. gestoppt werden. In einem optionalen Protokoll-Fenster lässt sich die Übertragung der einzelnen Nachrichten verfolgen und nach unterschiedlichen Kriterien filtern. Als wichtiges Hilfsmittel für Systementwickler bietet das LaunchPad eine Konsole, in der einzelne Befehlsequenzen direkt in der formalen Syntax der spezifischen kontextfreien Grammatiken eingegeben und übertragen werden können. Die Konsole verfügt zudem über eine History-Funktion, eine Auswahl an frei konfigurierbaren Nachrichten-Templates und einen Zugang über programmierbare Tastenkürzel.

KAPITEL 6

Hybride genetische Integration

Dieses Kapitel bildet neben den Darstellungen der Benutzerstudien und der Konzeption der Basisarchitektur einen weiteren Schwerpunkt in der vorliegenden Arbeit. Es beschreibt ein innovatives Verfahren zur Integration semantisch dekodierter, multimodaler Benutzereingaben, welches auf den Prinzipien der natürlichen Evolution beruht. Dazu werden in einer knappen Einführung zunächst die grundlegenden Elemente und Verarbeitungsprozesse in einem genetischen Algorithmus vermittelt. Der zweistufige Integrationsansatz besteht aus einer regelbasierten Vorverarbeitung und einer nachgeschalteten genetischen Datenfusion. Beide Phasen werden in gesonderten Abschnitten detailliert beschrieben. Um die Ausführungen insgesamt im Rahmen zu halten, wird bei den praktischen Beispielen zumeist ausschließlich auf das DVA-Szenario eingegangen. Den Abschluss des Kapitels bilden einige Anmerkungen zu der systemtechnischen Umsetzung des Ansatzes.

6.1 Grundlagen evolutionärer Algorithmen

Die Simulation natürlicher Evolutionsvorgänge ist ein inhärentes Charakteristikum evolutionärer Verfahren. Bezogen auf das spezifische Programmierkonzept kann strukturell zwischen *genetischen Algorithmen* (GA) [52], *genetischer Programmierung* (GP) [79] und *evolutionären Strategien* (ES) [131, 147] unterschieden werden. Zusammenfassend handelt es sich dabei um adaptive statistische Methoden zur Lösung von komplexen Optimierungsproblemen, bei denen nach dem Prinzip der Evolution verschiedene Lösungsvarianten miteinander konkurrieren. Durch den Austausch von Teilinformationen zwischen den einzelnen Individuen einer Lösungspopulation kristallisieren sich über mehrere Generationen insgesamt stabile und gute Lösungen heraus. Obwohl nicht generell garantiert werden kann, dass immer eine optimale Lösung gefunden wird, können durch die Verwendung von evolutionären Verfahren sehr gute Ergebnisse in akzeptabler Laufzeit erzielt werden [23].

Den Schwerpunkt der folgenden Beschreibung bilden die maßgeblich von Holland [66] in den 70er Jahren untersuchten genetischen Algorithmen in ihrer ursprünglichen Form. Es wird kurz auf die biologischen Hintergründe, auf fundamentale algorithmische Prinzipien und auf potenzielle Anwendungsgebiete und Systeme eingegangen. Für eine ausführliche Diskussion der zugrunde liegenden Theorien und Konzepte sei an dieser Stelle auf die umfangreiche Literatur

verwiesen. So findet der interessierte Leser beispielsweise eine funktionstheoretische Einführung in [161], eine umfassende Diskussion existierender algorithmischer Variationen in [7] und weiterführende Informationen in [46, 102, 139].

Biologischer Hintergrund

Die Basis der biologischen Evolution bildet das von Charles Darvin entdeckte Überlebens-Prinzip [39]. Charakteristisch für den Evolutionsprozess ist das permanente Streben organischer Strukturen nach Anpassung an sich ändernde Lebensbedingungen. In der Natur konkurrieren die einzelnen Individuen einer Spezies um limitierte Ressourcen, wie beispielsweise Wasser oder Nahrung. Starke bzw. gut an die Umgebungssituation angepasste Mitglieder einer Population können sich im Hinblick auf die verfügbaren Ressourcen einen Vorteil verschaffen und haben auf diese Weise auch eine deutlich höhere Chance zu überleben als schwache Mitglieder der Spezies. Darüber hinaus haben sie mehr Möglichkeiten, unter den anderen Individuen einen Partner zur Fortpflanzung auszuwählen und Nachkommen zu erzeugen. Im globalen Kontext werden daher die Gene dieser Individuen stärker in der gesamten Spezies verteilt.

Manchmal erschafft die Kombination guter Gene einzelne Nachkommen, die noch besser an die Natur angepasst sind als ihre Eltern und sich gegenüber den anderen Mitgliedern der Spezies leichter durchsetzen können. Bisweilen verändern sich auch die Gene ausgewählter Individuen durch äußere Einflüsse, wodurch vollkommen neue Eigenschaften und Fähigkeiten entstehen können. Über viele Generationen hinweg entwickelt sich auf diese Weise eine Spezies, die sich kontinuierlich an die gegebene Lebenssituation adaptiert.

Genetische Informationen werden in der Desoxyribonukleinsäure (DNS) gespeichert. Die Hauptbestandteile dieser Struktur sind die vier Nukleotide Adenin (A), Guanin (G), Thymin (T) und Cytosin (c), die sich lediglich durch ihre Stickstoffbasen unterscheiden [134]. Alle Einheiten sind über Zucker- und Wasserstoffbrücken miteinander verbunden und bilden eine organische Molekülkette (DNS-Doppelhelix). Über die Abfolge der Nukleotide in diesem Molekülstrang können die spezifischen Charakteristika eines Individuums vollständig kodiert werden. Zudem lässt sich die Struktur beispielsweise im Rahmen einer Zellteilung identisch reproduzieren und bildet damit die Grundlage für eine verlustfreie Übertragung des Erbmaterials.

Basisprinzip

Die Funktionsweise evolutionärer Algorithmen orientiert sich unmittelbar an dem Vorbild der Natur. Stark vereinfacht ausgedrückt lässt sich ein simulierter Evolutionsprozess als iteratives Zusammenspiel von Bewertung, Selektion und Rekombination genetischer Informationen darstellen. Abbildung 6.1 gibt einen strukturellen Überblick über die fundamentalen Elemente eines genetischen Algorithmus.

Jedes Mitglied einer Population repräsentiert eine Lösungsvariante im Hinblick auf ein vorgegebenes Problem. Als Ausgangsbasis wird eine initiale Population entweder zufällig oder unter Einbezug von Heuristiken und situativem Kontextwissen erzeugt. Um die einzelnen Lösungen quantitativ vergleichen zu können, wird ein spezifisches Bewertungsverfahren eingeführt. Dieses Maß bezeichnet man als die *Fitness* eines Individuums. Basierend auf den Ergebnissen wählt man hochwertige Individuen aus (*Selektion*) und kombiniert sie durch einen partiellen Austausch einzelner Lösungsparameter (*Rekombination*).

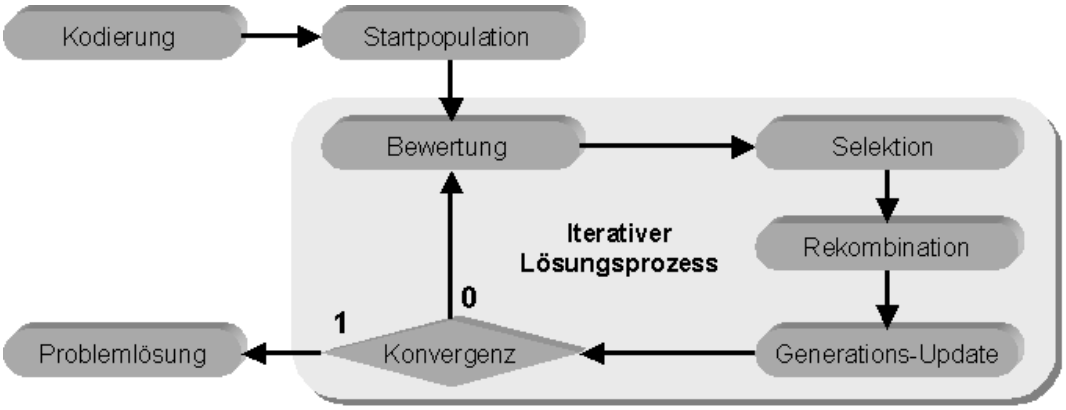


Abb. 6.1: Struktureller Ablauf eines genetischen Algorithmus (nach [52])

Zusätzlich werden in geringerem Maße zufällig markierte Teilkomponenten durch eine externe Instanz manipuliert (*Mutation*). Letztendlich wird aus den Individuen der Ursprungspopulation und den neuen, durch den Rekombinationsprozess entstandenen Individuen eine neue Generation von Lösungsalternativen abgeleitet. Diese werden im nächsten Iterationsschritt wiederum bewertet und variiert. Bei einer adäquaten Wahl der strukturierenden genetischen Elemente konvergiert die Gesamtpopulation nach mehreren Iterationen und führt zu einer annähernd optimalen Lösung des Ausgangsproblems. Im Folgenden wird auf die zentralen Komponenten eines genetischen Algorithmus näher eingegangen.

Anwendungen

Genetische Algorithmen sind erfolgreich für unterschiedliche Optimierungsprobleme eingesetzt worden. Viele forschungsbezogene und kommerzielle Software-Produkte nutzen die GA-Methodik zur Lösungs-Approximation. Ausgewählte Anwendungen werden im Folgenden kurz angesprochen. Ein guter Überblick zu den vielfältigen technischen Anwendungen ist in [19] enthalten.

Der Bereich der *numerischen Funktions-Approximation* war eines der ersten Anwendungsgebiete von genetischen Algorithmen. Beispielsweise in [140] konnte eindrucksvoll nachgewiesen werden, dass GA sowohl auf Basis von binärer, als auch auf natürlicher Kodierung traditionellen Optimierungstechniken wie Gradientenverfahren oder Simulated Annealing mit Bezug auf die Gesamtlaufzeit und Qualität der erzielten Lösung deutlich überlegen sind.

Ein weiteres bedeutendes Einsatzgebiet von genetischen Algorithmen bilden die *kombinatorischen Optimierungsprobleme*. Ein populäres Beispiel dieser Klasse ist das Problem des Handlungsreisenden (Traveling Salesman Problem [18]). Auf Basis von n vorgegebenen Punkten (Städten) in einer virtuellen Ebene besteht die Aufgabe darin, eine Rundreise (Tour) zu finden, die jeden Punkt besucht und eine minimale Gesamtlänge aufweist. Dieses Problem ist NP-vollständig. Die triviale deterministische Lösung hat eine Laufzeit von $O((n-1)!/2)$ und ist daher für große n sehr zeitaufwändig. In [101] wird eine Lösung mittels evolutionärer Algorithmen auf Basis einer natürlichen Kodierung vorgestellt, mit der sich in vergleichsweise kurzer Rechenzeit zufriedenstellende Lösungsapproximationen produzieren lassen.

Weitere Anwendungsgebiete von genetischen Algorithmen mit unmittelbarem technischen Nutzen finden sich beispielsweise beim Design von Platinen [49], dem Routing von Internet-Verbindungen [38] oder auch bei der partiellen Steuerung von Atomreaktoren [20].

6.2 Struktureller Systemüberblick

Motivation

In Abschnitt 2.3.3 sind verschiedene Ansätze zur Integration multimodaler Benutzereingaben vorgestellt worden. Die Verfahren unterscheiden sich sowohl in der spezifischen Integrationsmethode (temporal, regel-basiert oder statistisch) als auch im Abstraktionsgrad des Fusionsmechanismus (Merkmalsfusion, Modellfusion oder semantische Fusion). Um von den individuellen Vorteilen dieser Ansätze optimal profitieren zu können, sind zudem verschiedene hybride Strategien entwickelt worden, die einzelne Technologien vor dem Hintergrund der aktuellen Applikationsdomäne situativ kombinieren (siehe auch Abschnitt 5.1.1).

Für die Erzielung hervorragender Integrationsergebnisse müssen quantitative und qualitative Erfahrungen aus umfangreichen Benutzerstudien im Designprozess berücksichtigt werden. Die Umsetzung dieser Anforderung impliziert einen enormen ökonomischen und entwicklungstechnischen Aufwand. Bei einer Portierung des Systems auf eine andere Anwendungsdomäne sind die Ergebnisse nur bedingt übertragbar. Oftmals resultiert die Portierung eines existierenden Systems daher in einem vollständigen Redesign des Integrationsmoduls.

Die in Abschnitt 5.1.2 identifizierten Anforderungen gelten in gleichem Maße auch für die Entwicklung des Integrationskerns. Keiner der bisher diskutierten Ansätze erfüllt alle Kriterien an eine optimale Integrationskomponente. Systeme mit guten Erkennungsraten sind im Allgemeinen hochgradig domänenspezifisch, basieren auf umfangreichen Benutzerstudien, sind kaum erweiterbar und lassen sich nur bedingt auf andere Anwendungsszenarien übertragen. Auf der anderen Seite verfügen skalierbare, domäneninvariante Ansätze oftmals nur über bedingt akzeptable Erkennungsraten.

In einem grundlegenden Punkt stimmen die unterschiedlichen Ansätze vollständig überein. Das Problem der Integration multimodaler Informationsanteile wird ausschließlich in expliziter Form modelliert. Im Gegensatz dazu wird in der vorliegenden Arbeit ein Verfahren vorgestellt und diskutiert, bei dem die Kombination von Benutzereingaben und Kontextinformationen vollständig in impliziter Form erfolgt. Durch die Anwendung eines evolutionären Ansatzes lässt sich ein Großteil der Anforderungen an eine optimale Integrationsmethode erfüllen.

Zweistufiger Integrationsansatz

Die algorithmische Umsetzung des Integrationsprozesses gliedert sich in zwei strukturell getrennte Phasen. In der Vorverarbeitung werden gemäß der abstrakten Beschreibungen in Abschnitt 5.4.1 die eintreffenden Ereignisse strukturell analysiert und zeitlich und semantisch assoziierte Informationsanteile identifiziert. Die anschließende genetische Datenfusion setzt auf diesen Ergebnissen auf und bildet die einzelnen Events auf eine Systemaktion ab. Aufgrund dieser zweiteiligen Struktur mit regel-basierten und stochastischen Anteilen wird der gesamte Prozess als *hybride Integration* bezeichnet. Bezogen auf die Partitionierung des Integrators in Abbildung 5.3 konzentrieren sich die folgenden Darstellungen maßgeblich auf eine Auswertung der ersten beiden Integrationsebenen (Symbolfusion und Kontextevaluation).

Das Spotting ist analog zu den Integrationsphasen ebenfalls in mehreren Ebenen organisiert. Die Anpassung der zugrundeliegenden Regeln an die jeweilige Anwendungsdomäne ist zwar möglich, aber nicht eine zwingende Voraussetzung. Grundsätzlich besteht der Regelkorpus

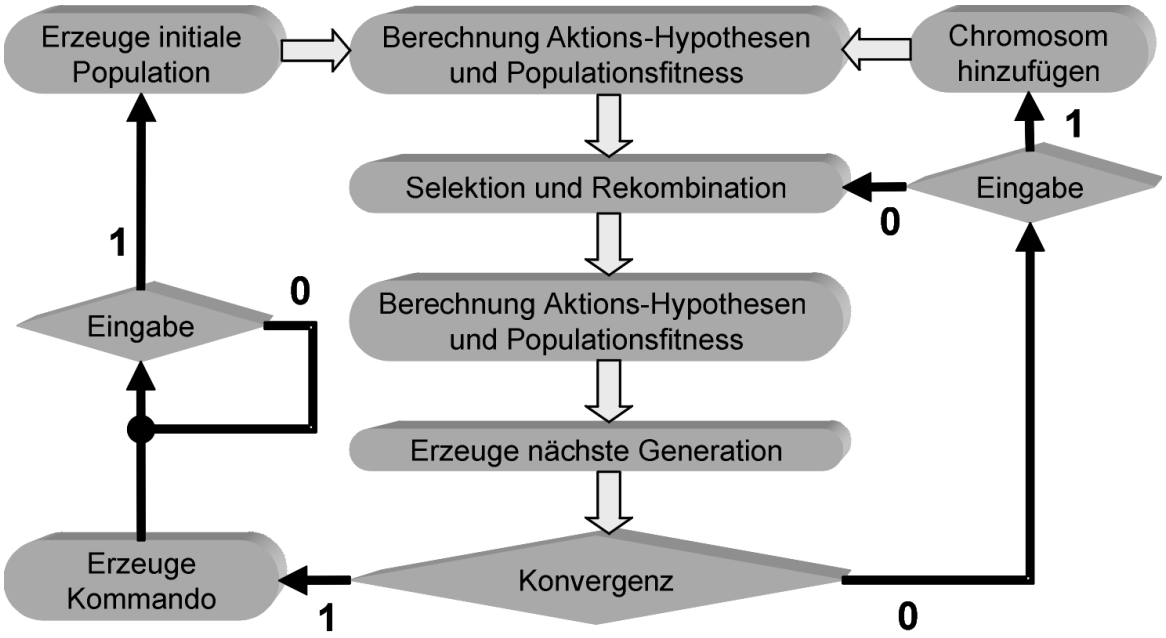


Abb. 6.2: Strukturelle Elemente des angepassten genetischen Algorithmus für die Integration multimodaler Informationsanteile unter Verwendung einer natürlichen Kodierung

aus allgemein gültigen Regeln und domänenspezifischen Erweiterungen. Bei einer Portierung des Systems auf ein anderes Anwendungsszenario kann zumindest ein Teil der Regeln direkt übertragen werden, so dass die prinzipielle Funktionsweise des Systems sichergestellt ist. Die regelbasierte Vorverarbeitung wird detailliert in Abschnitt 6.3 vorgestellt.

Im letzten Abschnitt wurden evolutionäre Berechnungsverfahren mit einem Schwerpunkt auf genetische Algorithmen vorgestellt. Diese Methoden werden bereits seit längerer Zeit erfolgreich für verschiedene Klassen von Such- und Optimierungsproblemen eingesetzt. Genetische Algorithmen verhalten sich darüber hinaus äußerst robust gegenüber variierenden Randbedingungen. Obwohl die Auffindung einer optimalen Lösung nicht garantiert werden kann, finden genetische Algorithmen gute Ergebnisse vor dem Hintergrund eines akzeptablen Zeithorizontes [23]. Aus diesen Gründen sind sie auch für die Integration multimodaler Informationsanteile von besonderem Interesse.

Die strukturellen Elemente des problemangepassten evolutionären Fusionsverfahrens beziehen sich unmittelbar auf das grundlegende Verarbeitungsmodell eines genetischen Algorithmus. Abbildung 6.2 veranschaulicht die wesentlichen Phasen in diesem Prozess. Ein neuer Integrationsdurchlauf wird genau dann initiiert, wenn das Spotting-Modul ein Intervall C_S von zusammengehörigen Ereignissen zur Verfügung stellt. Der erste Schritt besteht darin, eine initiale Population von Individuen zu erzeugen. Den einzelnen Individuen lassen sich entsprechende Integrationsergebnisse zuordnen, die nach der Terminologie aus Kapitel 5 als *Aktions-* bzw. *Kommando-Hypothesen* bezeichnet werden.

Jedes Individuum wird gemäß seiner genetischen Fitness bewertet, wobei die semantischen Repräsentationen der einzelnen Benutzereingaben, die temporalen Beziehungen der Ereignis-Sequenzen, der Status der einzelnen Systemmodule, empirische Benutzerdaten und vorherige Integrationsergebnisse mit einbezogen werden. Im nächsten Schritt werden vielversprechende

Integrationsergebnisse für einen Rekombinationsprozess ausgewählt und durch speziell angepasste genetische Operatoren manipuliert. Aus den existierenden und den neuen Individuen werden im Hinblick auf die Aktionshypothesen die am besten bewerteten Lösungen ausgewählt und in eine neue Population übertragen. Dieser Prozess wird solange iteriert, bis entweder im Hinblick auf die Qualität der Lösungen ein Konvergenzkriterium erfüllt ist oder eine frei definierbare Obergrenze für die Anzahl der durchgeführten Iterationen erreicht ist.

Für den Fall, dass während des Fusionsprozesses weitere Ereignisse auftreten, die zwar nicht zu dem Integrationsintervall C_S gehören, für die Integration aber wichtig sind, können den einzelnen Individuen zusätzliche Informationen hinzugefügt werden. Auf diese Weise können beispielsweise neue Informationsquellen ad hoc in den Integrationsprozess eingebunden werden. Zusätzlich bietet dieser Mechanismus die Möglichkeit, die strikte Trennung von Vorverarbeitung und Datenfusion aufzubrechen. Die Konzeption der genetischen Datenfusion wird in den Abschnitten 6.4 und 6.6 detailliert beschrieben.

6.3 Regelbasierte Vorverarbeitung

Grundlage für die Entwicklung des Spotting-Verfahrens sind die Ergebnisse aus den Benutzerstudien in Kapitel 4 sowie die Erfahrungen aus der Analyse existierender Fusionsansätze. Auf Basis dieser beiden Datenquellen werden im Folgenden stufenweise verschiedene Regeln ermittelt, die in ihrer Gesamtheit einen effektiven und effizienten Spotting-Algorithmus realisieren. Jede Entwicklungsphase wird als eigene Ausbaustufe des Spotting-Moduls konzipiert. In der Praxis muss der Umfang des Regelwerks an die spezifischen Anforderungen der einzelnen Anwendungsdomänen sowie an die Anzahl der verfügbaren Informationsquellen angepasst werden. Um die Darstellung kompakt zu halten, beschränken sich die domänenspezifischen Beispiele hauptsächlich auf das DVA-Szenario.

6.3.1 Funktionale Dekomposition

Der erste Schritt im Rahmen der Segmentierung multimodaler Informationsanteile besteht in einer inhaltlichen Analyse der Eingabedaten. Ohne Beschränkung der Allgemeinheit wird im Folgenden eine Partitionierung der Ereignisse in drei strukturelle Teile vorgenommen: *Modus*, *Funktion* und *Parameter*. Die jeweiligen Informationsanteile bezeichnet man als *funktionale Ereignis-Slots*. Sie entsprechen den Semunen der modulspezifischen kontextfreien Grammatiken und werden daher auch mit S abgekürzt. Analog zu den Benutzerereignissen in Abschnitt 4.3.1 lässt sich die funktionale Zusammensetzung modul-invarianter Informationsanteile in Form einer BNF-Regel darstellen:

$$\begin{aligned} <E> &::= <S^M>\ [<S^F>]\ \{<S^P>\} \\ &::= [<S^M>]\ <S^F>\ \{<S^P>\} \\ &::= [<S^M>]\ [<S^F>]\ <S^P>\ \{<S^P>\} \end{aligned} \tag{6.1}$$

Ein syntaktisch korrektes Ereignis E von einem der angeschlossenen Module enthält eine nichtleere Teilmenge der Strukturkategorien $\mathcal{S} = \{S^M, S^F, S^P\}$. Die einzelnen Slots können jeweils für sich alleine oder in Kombination mit den anderen Einträgen auftreten. Dabei können

pro Ereignis jeweils nur eine Modusinformation und eine Funktionsinformation enthalten sein. Demgegenüber ist die Anzahl der Parameterdaten pro Ereignis nicht limitiert. Die Reihenfolge der einzelnen Informationsanteile innerhalb eines Ereignisses ist nicht von Bedeutung, d.h. neben den in Gleichung 6.1 dargestellten Ereigniszusammensetzungen sind auch sämtliche, daraus resultierende Permutation der Symbolanordnungen zulässig. Darüber hinaus müssen auch die einzelnen Parameteranteile nicht notwendigerweise direkt gruppiert sein. Sie können an beliebigen Positionen innerhalb der Ereignisstruktur stehen.

In Bezug auf Benutzereingaben bezeichnet man die mit dem Ereignis verknüpften Systemreaktionen als Aktionen bzw. Kommandos. In diesem Zusammenhang spricht man von einem *Voll-Kommando* A^{VK}, wenn alle drei Informationsanteile innerhalb eines Ereignisses auftreten. Fehlt einer der Anteile, so bezeichnet man die Aktion als *Teil-Kommando* A^{TK}. Im Rahmen der inhaltlichen Analyse wird bestimmt, welcher Strukturkategorie die einzelnen Informationsanteile zugeordnet werden können. Dies geschieht durch die Auswertung anwendungs- und modulspezifischer Wissensbasen, die vor dem Start des Systems geladen werden.

Bei der Anwendung im DVA-Szenario steht S^M beispielsweise für den übergeordneten Betriebsmodus (walk, fly, examine, etc.), im Automobilbereich AIA wird an dieser Stelle das angesprochene Gerät angegeben (Radio, Telefon, CD-Player, etc.). Im zweiten Eintrag S^F wird der funktionelle Teil des Befehls spezifiziert. Das entspricht im DVA-Szenario der Bewegungsart (rot, trans) und im AIA-Szenario der auszuführenden Funktion (play, skip, etc.). Unter dem dritten Eintrag S^P werden sämtliche Funktionsparameter subsumiert, wobei auch mehrere Angaben zulässig sind. Dies betrifft in dem DVA-System neben der Bewegungsrichtung (left, forward, etc.) auch die Anzahl der durchzuführenden Bewegungsschritte. Für die automotive Infotainmentapplikation handelt es sich bei den Parametern z.B. um einen Verweis auf einen speziellen Listeneintrag.

Die spezifische Kenntnis der Ereignisstruktur ist für den nachfolgenden Segmentierungsprozess von fundamentaler Bedeutung. Im DVA-Szenario ist beispielsweise bekannt, dass Teilbefehle, die von keinem weiteren Befehl überlagert werden, eine höhere Interaktionswahrscheinlichkeit haben als Vollbefehle. Zudem fallen die Mittelwerte der Verzögerungszeiten bei diesem Befehlstyp deutlich höher aus. Aus diesem Grund wird bei Teilbefehlen ein breiteres Zeitfenster gewählt als bei Vollbefehlen. Darüber hinaus weisen die Interaktionswahrscheinlichkeit und die Verzögerungszeit Abhängigkeiten von den Modalitäten der Ereignisse auf. Infolgedessen kann die Länge des Integrationsclusters T_I in Abhängigkeit von der semantischen Struktur und den Modalitäten der einzelnen Ereignisse festgesetzt werden. Bei der Dimensionierung von $\Delta(T_I)$ muss berücksichtigt werden, dass auch Interaktionen mit höheren Verzögerungswerten noch innerhalb des Integrationsintervalls liegen.

6.3.2 Auswertung temporaler Relationen

Im Folgenden werden im Hinblick auf den Spotting-Prozess zunächst nur die zeitlichen Aspekte betrachtet. Die Dauer eines isolierten Ereignisses E ergibt sich gemäß Gleichung 4.21 aus der Differenz des Startzeitpunktes $t_s(E)$ und dem Ende $t_e(E)$ des Ereignisses. Als Bezeichner für diese Dauer wird entweder $T(E)$ oder alternativ auch $\Delta(E)$ verwendet. An vielen Stellen kann die Modalität eines Ereignisses für die Interpretation der Information, die an das Ereignis gekoppelt ist, von entscheidender Bedeutung sein. Aus diesem Grund wird mit $M(E)$ die Datenquelle des Ereignisses referenziert.

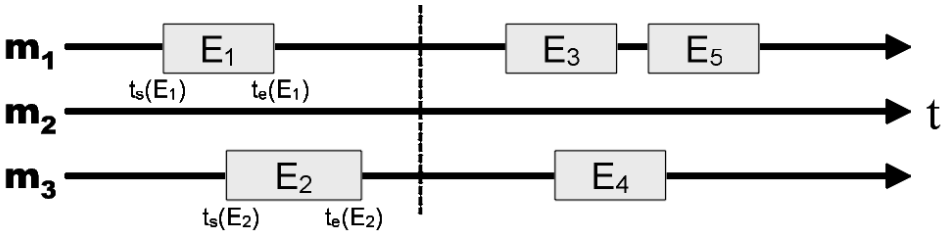


Abb. 6.3: Darstellung zeitlich überlagerter Ereignissequenzen

Die temporalen Beziehungen zweier Ereignisse E_a und E_b können eindeutig über eine Analyse der beiden Startzeiten $t_s(E_a)$ und $t_s(E_b)$ und der beiden Endzeiten $t_e(E_a)$ und $t_e(E_b)$ bestimmt werden. Abbildung 4.1 und die dazugehörigen Gleichungen 4.22 bis 4.24 weisen für zwei überlagerte Events verschiedene Zeitabschnitte aus, die sich aus unterschiedlichen Zeitdifferenzen ergeben. Eine mathematisch exakte Beschreibung der möglichen Überlagerungsarten erfolgt im Gleichungskomplex 4.25 bzw. anschaulich in Abbildung 4.2. Die zeitliche Überlappung von zwei Ereignissen wird mit T_U bezeichnet. Mit Bezug auf die Terminologie in Abbildung 4.1 berechnet sich die Gesamtdauer T_G aus der Summe der einzelnen Zeitabschnitte.

$$T_U(E_a, E_b) = T_{v2}(E_a, E_b) \tag{6.2}$$

$$T_G(E_a, E_b) = T_{v1}(E_a, E_b) + T_{v2}(E_a, E_b) + T_{v3}(E_a, E_b) \tag{6.3}$$

Zeitliche Überlagerung

Für das zeitbasierte Spotting sind in erster Näherung besonders die gestaffelt überlagerten Ereignisse E^{STA} von Interesse. Mit Bezug auf Abbildung 6.3 (links) wird eine Sequenz aus zwei Ereignissen E_1 und E_2 als zeitlich überlagert betrachtet, falls die entsprechenden Zeitmarken der folgenden Bedingung genügen: $t_s(E_1) < t_s(E_2) < t_e(E_1)$. In dieser sowie in allen folgenden Betrachtungen werden die einzelnen Ereignisse in Abhängigkeit von ihren Anfangszeitpunkten indiziert, d.h. für $a < b$ folgt $t_s(E_a) < t_s(E_b)$.

Aus den Versuchsdaten konnte ermittelt werden, dass große Teile der multimodalen Interaktionen zeitlich überlagert sind. Im DVA-Szenario lag dieser Anteil bei mehr als 90%. Aus diesem empirischen Ergebnis wird die erste Spotting-Regel $\Gamma_S(1)$ abgeleitet:

> **Spotting-Regel** $\Gamma_S(1)$:
> Zeitlich überlagerte Eingabeereignisse sind Teil einer multimodalen Interaktion.

Ein System, das nur auf Basis dieser Regel arbeitet, würde alle zeitlich überlagerten Ereignisse in einem Intervall C_E zusammen gruppieren und jeweils in ein gemeinsames Spottingcluster C_S einordnen. Unimodale Benutzerinteraktionen würden jedoch nicht mit aufgenommen werden. Daraus ergeben sich unmittelbar zwei potentielle Fehlerquellen. Zum einen werden sequentielle Interaktionen nicht erkannt und zum anderen werden auch voneinander unabhängige Benutzereingaben als multimodal eingestuft, falls diese zeitlich überlagert sind.

Für die zweite Fehlerquelle kann direkt auf Basis der Benutzerstudien eine geeignete Behandlungsstrategie vorgeschlagen werden. Zwei oder drei zeitlich überlagerte Eingaben unterschiedlicher Modalität, die von keinen weiteren Symbolen überlagert werden, bilden zumeist eine zusammengehörige, multimodale Interaktion. Bei Sequenzen von drei oder mehr Ereignissen,

die paarweise zeitlich überlagert sind, handelt es sich dagegen niemals um eine einzige, zusammenhängende Interaktion, wenn zwei Semune innerhalb einer Sequenz der gleichen Modalität zugeordnet werden können. Das einfachste unter diesen Voraussetzungen denkbare Szenario ist in Abbildung 6.3 auf der rechten Seite dargestellt. In diesem Beispiel stimmen die Modalitäten der Ereignisse E_3 und E_5 überein und beide sind jeweils mit dem Ereignis E_4 zeitlich überlagert. Es handelt sich um eine bimodale Interaktion aus E_3 und E_4 sowie um eine anschließende unimodale Eingabe E_5. Der bisher entworfene Algorithmus würde aber eine trimodale Interaktion aus allen drei Ereignissen erkennen. Zunächst wird eine Regel formuliert, durch die ein solches Szenario korrekt behandelt werden kann:

Spotting-Regel $\Gamma_S(2)$:
Trimodale Interaktionen enthalten niemals Symbole der gleichen Modalität

Durch die Regel $\Gamma_S(2)$ wird ein falsches Spotting in dem oben beschriebenen Sonderfall vermieden. Es muss aber noch ein Entscheidungsverfahren entwickelt werden, mit dessen Hilfe eine korrekte Einordnung der Ereignisse vorgenommen werden kann. Zu diesem Zweck wird ein Überlagerungsfaktor $u_{a,b}$ definiert, der den Anteil der zeitlichen Überlagerung zweier Ereignisse E_a und E_b beschreibt:

$$u_{a,b} = \frac{T_U(E_a, E_b)}{T_G(E_a, E_b)} \tag{6.4}$$

Eine genauere Untersuchung der Versuchsdaten hat ergeben, dass $u_{a,b}$ bei zwei zusammenhängenden Ereignissen E_a, E_B bis auf wenige Ausnahmen wesentlich größer ist als bei isolierten Ereignissen. Aus diesem empirischen Befund kann eine weitere Regel abgeleitet werden, die in kritischen Fällen die einzelnen Überlagerungsfaktoren berechnet und auf dieser Basis eine entsprechende Einordnung vornimmt. Das Verfahren wurde so konzipiert, dass es nicht nur in einfachen Fällen wie in Abbildung 6.3 (rechts), sondern auch in Situationen, in denen mehrere Signale vieler verschiedener Datenquellen beteiligt sind, optimale Ergebnisse liefert.

Spotting-Regel $\Gamma_S(3)$:
Sei $C_E = \{E_1, E_2, ...E_n\}$ das zu untersuchende Intervall an Eingabeereignissen. Paarweise überlagerte Ereignisse E_x, E_y mit $x, y \in \{1..n\}$ und $x \neq y$ werden solange in ein temporäres Cluster C'_S eingefügt, bis zwei Ereignisse E_x, E_y derselben Modalität ($M(E_x) = M(E_y)$) in C'_S liegen. Danach findet eine Berechnung von $u_{a,b}$ für alle unmittelbar benachbarten Ereignisse E_a, E_b mit $|a - b| = 1$ statt. Ab der Stelle, an der $u_{a,b}$ den niedrigsten Wert einnimmt, werden alle weiteren Events aus dem temporären Cluster entfernt und aus den verbleibenden Ereignissen wird das finale Intervall C_S gebildet.

Die Vorgehensweise wird anhand des in Abbildung 6.4 auf der linken Seite dargestellten Beispiels erläutert. Hierbei handelt es sich um sechs paarweise überlagerte Ereignisse, die innerhalb eines Intervalls C_E liegen. Nach den bisherigen Regeln würden zuerst alle sechs Ereignisse in ein temporäres Cluster C'_S überführt. Da die Datenquellen jedoch zum Teil übereinstimmen, können nicht alle Ereignisse Teil einer einzigen multimodalen Interaktion sein. Diese Erkenntnis kann bereits aufgrund der Einschränkung in Regel $\Gamma_S(2)$ nach der Betrachtung von E_4 gemacht werden. Das Auffüllen von C'_S wird daher an dieser Stelle abgebrochen. Berechnet man die einzelnen Überlagerungsfaktoren $u_{1,2}$, $u_{2,3}$, $u_{2,3}$ und $u_{3,4}$, so ergibt sich in diesem Beispiel für $u_{3,4}$ der kleinste Wert. Daraus wird geschlossen, dass E_4 nicht zu den anderen Ereignissen gehören kann und deshalb auch aus dem temporären Intervall entfernt werden muss.

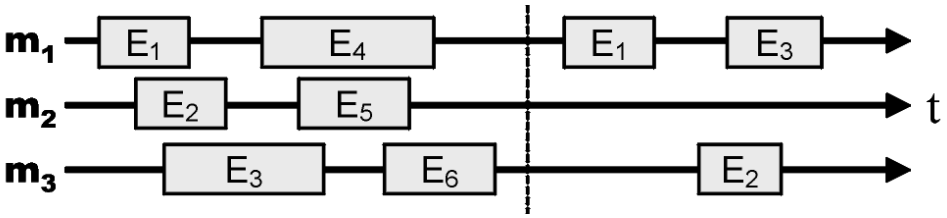


Abb. 6.4: Komplexe Anordnung zeitlich überlagerter Ereignisse (links) und typische Konstellation für Ereignissequenzen mit sequentiellen und überlagerten Ereignisrelationen (rechts)

Zeitliche Sequentialität

Neben zeitlich überlagerten Ereignissen müssen auch sequentielle Benutzereingaben im Spotting-Prozess berücksichtigt werden. Die Verzögerung zwischen zwei direkt aufeinanderfolgenden Ereignissen E_a, E_b wird mit $T_L(E_a, E_b)$ bzw. nach Gleichung 4.26 mit T_{v4} bezeichnet. Befinden sich innerhalb eines zu untersuchenden Integrationsintervalls C_E ausschließlich zwei Ereignisse, die zueinander sequentiell sind, so ist es unmöglich, allein aufgrund der zeitlichen Betrachtung eine Entscheidung bzgl. des Zusammenhangs zwischen den beiden Ereignissen zu treffen. Eine solche Entscheidung kann nur unter Einbezug der semantischen Relationen getroffen werden und wird daher erst im nächsten Abschnitt behandelt. Im Folgenden werden zwei Regelerweiterungen vorgestellt, die genau dann Anwendung finden, wenn innerhalb eines Integrationsintervalls sowohl sequentielle als auch überlagerte Ereignisse auftreten.

Spotting-Regel $\Gamma_S(4)$:
Treten innerhalb eines Integrationsintervalls C_E drei Ereignisse E_a, E_b, E_c auf, bei denen zwei Ereignisse aus der gleichen Quelle stammen (z.B. $M(E_a) = M(E_c)$) und das andere Ereignis E_b entweder mit E_a oder E_c zeitlich überlagert ist, dann bilden die überlagerten Ereignisse eine zusammengehörige bimodale Interaktion und das andere Ereignis muss in ein separates Cluster übertragen werden.

Spotting-Regel $\Gamma_S(5)$:
Bei einer trimodalen Interaktion ($M(E_a) \neq M(E_b) \neq M(E_c)$) kann maximal ein Ereignis zu den anderen beiden Ereignissen sequentiell sein.

Mit Hilfe dieser Regeln kann das in Abbildung 6.4 auf der rechten Seite dargestellte Szenario korrekt interpretiert werden. Zunächst werden alle drei Events in ein temporäres Cluster C'_S übertragen. Aufgrund der identischen Datenquellen von E_1 und E_3 kann es sich hier nicht um eine trimodale Interaktion handeln. Regel $\Gamma_S(4)$ besagt, dass in diesem Fall die Ereignisse E_2 und E_3 strukturell zusammengehören. Daher werden aus C'_S zwei Integrationsintervalle abgeleitet. Im ersten Cluster $C_1 = \{E_1\}$ befindet sich mit E_1 nur ein einziges Ereignis. Die beiden anderen Ereignisse werden einem separaten Integrationsintervall $C_2 = \{E_2, E_3\}$ zugeordnet. Wäre E_3 ein Ereignis einer anderen Modalität, das sequentiell zu E_2 liegt, so kann bezogen auf die Erfahrungen aus den Benutzerstudien eine trimodale Interaktion ausgeschlossen werden. Wie ein solcher Fall zu behandeln ist, wird im anschließenden Abschnitt erläutert.

6.3.3 Auswertung semantischer Relationen

Im letzten Abschnitt wurde ein reduziertes Regelsystem vorgestellt, das ein Spotting alleine aufgrund einer Betrachtung der zeitlichen Zusammenhänge sowie der Datenquellen der einzelnen Ereignisse ermöglicht. Die Leistungsfähigkeit dieses Ansatzes kann durch die zusätzliche Auswertung der semantischen Relationen zwischen den Ereignissen weiter gesteigert werden.

Da der Algorithmus in dieser Ausbaustufe noch nicht auf Kontextwissen zurückgreifen kann, müssen die aufgestellten Regeln domänenübergreifend gelten. Durch den Einbezug der Semantik soll vor allem die Frage geklärt werden, ob zwei Befehle überhaupt sinnvoll miteinander kombiniert werden können. Zu diesem Zweck werden die semantischen Relationen zwischen den einzelnen Ereignissen analog zu der formalen Darstellung in Gleichung 4.18 als redundant, rivalisierend oder komplementär bezeichnet. Die Gruppe der komplementären Beziehungen lässt sich noch feiner aufschlüsseln. Nach Gleichung 4.19 wird zwischen redundant-komplementären, rivalisierend-komplementären und harten komplementären Beziehungen unterschieden. Für eine detaillierte Beschreibung dieser Begriffsterminologien sei auf die entsprechenden Ausführungen in Abschnitt 4.2.3 verwiesen.

Grundsätzlich sind zwei Ereignisse nicht ohne weiteres kombinierbar, wenn sie zueinander rivalisierend oder komplementär-rivalisierend sind. In allen anderen Fällen können die Events in ein gemeinsames Spottingintervall übertragen werden. Für die beiden rivalisierenden Fälle kann eine Kombination aber auch nicht völlig ausgeschlossen werden, da die Möglichkeit eines Erkenner- oder Bedienfehlers in Betracht gezogen werden muss. Es macht wenig Sinn, den Spottingprozess allein auf Basis der semantischen Informationen durchzuführen. In der Praxis werden die semantischen und die zeitlichen Relationen immer gemeinsam evaluiert.

Bei zeitlich überlagerten Ereignissen wird der semantische Zusammenhang nur in Fällen, in denen das Überlagerungsverfahren aus Regel $\Gamma_S(4)$ Anwendung findet, als zusätzliche, höher priorisierte Entscheidungshilfe herangezogen. Kann in einer solchen Situation aufgrund der semantischen Relationen bereits entschieden werden, dass zwei Ereignisse nicht miteinander interagieren, so muss das Überlagerungskriterium nicht mehr überprüft werden. Treten die Ereignisse sequentiell auf, so kommt ihrer semantischen Relation eine entscheidende Bedeutung zu. Da in dieser Phase aufgrund der fehlenden Kontextinformationen noch keine adaptiven Zeitgrenzen gesetzt werden können, ist eine Entscheidung über die Zusammengehörigkeit der Ereignisse nur über die Auswertung der semantischen Informationen möglich.

> **Spotting-Regel** $\Gamma_S(6)$:
> Seien E_a, E_b zwei sequentielle Ereignisse und dazu die beiden einzigen Elemente in einem temporären Cluster C'_S mit $M(E_a) \neq M(E_b)$. Sind die Ereignisse zueinander rivalisierend oder komplementär-rivalisierend, so wird C'_S in zwei Cluster aufgespaltet, andernfalls wird aus C'_S das finale Spottingintervall C_S gebildet. Ereignisse gleicher Modalität ($M(E_a) = M(E_b)$) können dagegen nur dann miteinander kombiniert werden, wenn sie untereinander komplementär oder rivalisierend sind.

Ereignisse mit rivalisierenden Informationsinhalten, die aus einer einzigen Datenquelle stammen, werden im Spotting als *korrigierende Fehler* bezeichnet, d.h. es besteht die Möglichkeit, dass der Benutzer eine Anweisung gibt und diese gleich im Anschluss wieder korrigiert. In diesem Fall werden zwar beide Ereignisse in ein gemeinsames Cluster übertragen, zusätzlich wird aber ein Fehler-Flag gesetzt, dass vom Fehlermanagement Φ_F ausgewertet werden kann.

Bei sequentiellen Ereignissen aus unterschiedlichen Modalitäten, die sich entweder rivalisierend oder komplementär-rivalisierend zueinander verhalten, kann es aufgrund der harten Entscheidung in Regel $\Gamma_S(6)$ in manchen Situationen zu einem falschen Spotting kommen (z.B. im Falle eines Erkennerfehlers). Diese Situationen können aber in der späteren Fehlerbehandlung problemlos aufgelöst werden.

Die Regel $\Gamma_S(6)$ kann dahingehend erweitert werden, dass auch mehrere, innerhalb eines Intervalls sequentiell zueinander liegende Ereignisse behandelt werden können. Dabei wird die Erkenntnis aus den Benutzerstudien benutzt, dass sequentielle Interaktionen überwiegend komplementär sind, gelegentlich redundant-komplementären Charakter haben und sich nur in wenigen Fällen aus rein redundanten Informationsanteilen zusammensetzen. Für den Fall, dass drei sequentielle Ereignisse E_a, E_b, E_c innerhalb eines temporären Intervalls C'_S liegen, spricht man von einem *doppelt sequentiellen Szenario* und geht wie folgt vor:

Spotting-Regel $\Gamma_S(7a)$:
Wenn $M(E_a) \neq M(E_b) \neq M(E_c)$ gilt und zwischen E_a und E_b nicht die gleiche semantische Relation besteht wie zwischen E_b und E_c, so wird eine komplementäre Relation generell einer redundanten Relation vorgezogen. Bei einer rivalisierenden Relation wird auf die Unabhängigkeit der einzelnen Ereignisse geschlossen. Bei gleichen semantischen Relationen zwischen den beiden Symbolpaaren muss eine Entscheidung aufgrund von Kontextkriterien getroffen werden.

Spotting-Regel $\Gamma_S(7b)$:
Wenn $M(E_a) = M(E_b)$ und $M(E_b) \neq M(E_c)$ gilt und sich E_a und E_b zueinander rivalisierend verhalten, so liegt ein korrigierender Fehler vor, d.h. die beiden Ereignisse sind Teil einer gemeinsamen Interaktion. Ansonsten wird wie oben vorgegangen, bei gleichen semantischen Relationen werden allerdings E_b und E_c als zusammengehörig eingestuft.

Falls von den einzelnen Erkennernmodulen nach Konfidenzmaßen geordnete Listen mit Erkennungsergebnissen zur Verfügung gestellt werden, so findet in der aktuellen Systemrealisierung nur eine Betrachtung des ersten Eintrags statt. Aufgrund der vielfachen Kombinationsmöglichkeiten wäre das Spotting bei Betrachtung aller Einträge nur mit Hilfe deutlich aufwendigerer Algorithmen zu realisieren. Die Ergebnislisten fließen aber als wichtiges Merkmal in die Fehlerbehandlung ein.

6.3.4 Auswertung kontextueller Informationen

Analog zu dem Integrationsprozess profitiert auch das Spottingverfahren von einem Einbezug der verfügbaren Kontextinformationen. Durch die Kenntnis der Kombinationsoptionen der einzelnen Ereignisse kann die im letzten Abschnitt durchgeführte Analyse der semantischen Informationsanteile weiter verfeinert werden. Als neue semantische Relation wird die *funktionale Unabhängigkeit* eingeführt, die genau dann eintritt, wenn zwei Symbole zwar strukturell komplementär sind, die einzelnen Semune aber nicht zu einem syntaktisch korrekten Wort der kontextfreien Grammatik miteinander verbunden werden können.

Diese neue Relation ist vor allem in Domänen von großem Nutzen, in denen komplementäre Interaktionen zwischen den Benutzerinteraktionen aufgrund der Randbedingungen weitestgehend ausgeschlossen sind. Ein typisches Beispiel mit relativ einfachen Funktionen ist der in den Benutzerstudien verwendeten Prototyp für den AIA-Versuch (siehe Abschnitt 3.4).

Für die Unabhängigkeitsrelation gelten prinzipiell die gleichen Regeln wie für rivalisierende Beziehungen. Es können aber keine korrigierenden Fehler mehr behandelt werden. Treten innerhalb eines Spotting-Clusters beispielsweise zwei Parameter-Semune mit unterschiedlichem Inhalt auf, so wird nicht mehr automatisch eine rivalisierende Relation angenommen, sondern es wird zuerst überprüft, ob unter den Parameterwerten eine komplementäre Relation besteht. Die Information bezüglich der Semune, die miteinander kombinierbar sind, werden durch externe Tabellen zur Verfügung gestellt.

Die zweite Art von Kontextwissen, die in den Algorithmus einbezogen wird, sind die Informationen aus dem Benutzerkontext. Bei genauerer Betrachtung basieren auch sämtliche bisher entwickelten Regeln auf Benutzerwissen, da sie aus Rückschlüssen aus dem Verhalten der Testpersonen in den Benutzerstudien resultieren. Dabei wurden aber nur die Ergebnisse einbezogen, die tendenziell auch eine domänenübergreifende Gültigkeit besitzen. Als Kontextwissen werden in der dritten Ausbaustufe des Spottingprozesses dagegen nur die Informationen betrachtet, die für die aktuelle Domäne und den momentanen Benutzer charakteristisch sind. Dabei handelt es sich im Idealfall um Daten, die sich nur auf den gerade aktiven Benutzer beziehen. Es kann sich aber auch um die Daten einer spezifischen Benutzergruppe handeln.

Eine der Einsatzmöglichkeiten des Benutzerwissens ist die dynamische Anpassung der Länge des Integrationsintervalls $T(C_S)$. Zu diesem Zweck muss aus den Testdaten der maximale zeitliche Abstand zwischen dem Anfangszeitpunkt des ersten Ereignisses einer zusammengehörigen Interaktion und dem Beginn des letzten damit assoziierten Ereignisses ermittelt werden. Dies geschieht in Abhängigkeit von der Modalität und dem semantischen Inhalt des ersten Events. Die entsprechenden Zeitwerte werden vor Anwendungsbeginn an den Integrator übermittelt. Auf diese Weise kann bei jedem Cluster auf eine situationsspezifische Voreinstellung von $T(C_S)$ zurückgegriffen werden.

Ein Algorithmus, der lediglich die bisher eingeführten Regeln nutzt, weist unter Umständen eine hohe Fehlerrate bei der Einordnung sequentiell zueinander liegender Ereignisse auf. In dem in Abschnitt 6.3.2 betrachteten Fall von zwei sequentiellen Ereignissen E_a, E_b wird angenommen, dass die Events strukturell zusammengehören, wenn sie innerhalb von C_S liegen und keine semantischen Widersprüche auftauchen. Auf diese Weise werden jedoch viele Interaktionen, die eigentlich unimodal sind, fälschlicherweise als bimodal eingeordnet. Der Grund dafür liegt zum einen darin, dass die semantischen Regeln zu allgemein gehalten sind. Zum anderen ist die Intervalllänge als einzige Zeitgrenze in vielen Fällen deutlich zu hoch. Bei der Wahl von $T(C_S)$ müssen auch Szenarien mit mehreren Interaktionen und überdurchschnittlich langen Interaktionszeiten berücksichtigt werden.

Abhilfe bringt auch in diesem Fall die Verwendung von Benutzerwissen, wobei zeitliche und semantische Informationen miteinander kombiniert werden. Aus den Versuchsdaten der jeweiligen Domäne werden dazu die durchschnittlichen Verzögerungszeiten aller sequentiellen Interaktionen $\bar{T}_L$ in Abhängigkeit von der Modalität, der Reihenfolge und der semantischen Relationen der einzelnen Ereignisse bestimmt. Bei der Unterteilung der komplementären Interaktionen wird die Erkenntnis einbezogen, dass bei komplementären, sequentiellen Interaktionen meistens beim ersten Ereignis ein strukturell höherstehender Eintrag gefüllt wird als beim zwei-

ten Event. In dem DVA-Szenario erfolgte beispielsweise die Funktionsangabe (Bewegungsart) fast immer vor der Parameterangabe (Bewegungsrichtung). Der umgekehrte Fall ist in den Benutzerstudien insgesamt nur drei mal aufgetreten. In diesem Fall weisen die Slots zudem noch wesentlich kürzere Verzögerungszeiten auf.

Mit Hilfe dieser empirischen Informationen kann der Spotting-Algorithmus um eine variable Zeitgrenze $\epsilon(C_E)$ ergänzt werden. In Abhängigkeit von den Modalitäten und der semantischen Relationen der Ereignisse in C_E wird aus den durchschnittlichen Verzögerungszeiten $\bar{T}_L(C_E)$ und deren Standardabweichungen $\delta\bar{T}_L(C_E)$ eine Summe gebildet und $\epsilon(C_E)$ in erster Näherung zugewiesen:

> **Spotting-Regel** $\Gamma_S(8)$:
> Zwei semantisch kombinierbare Ereignisse E_a und E_b, die als einzige innerhalb eines temporären Spotting-Intervalls C_E liegen und für die die Beziehung $\bar{T}_L \leq \epsilon(E_a, E_b)$ gilt, werden als strukturell zusammenhängende Ereignisse betrachtet und in ein finales Cluster C_S übertragen.

Die Verzögerungszeit zweier Ereignisse E_a und E_b in einem Intervall C_E wird mit $\epsilon(E_a, E_b)$ verglichen. Gemäß Regel $\Gamma_S(8)$ wird dann eine Entscheidung getroffen. Befinden sich zwei Ereignisse in dem Ereignisintervall, für die aus den Versuchsdaten keine Werte für $\bar{T}_L$ und $\delta\bar{T}_L(C_E)$ ermittelt werden konnten, weil das entsprechende Szenario in den Versuchsdaten nicht vorkommt, so wird ϵ auf Null gesetzt und E_1 und E_2 können nicht sequentiell miteinander interagieren. Diese Maßnahme ist nur dann sinnvoll, wenn davon ausgegangen werden kann, dass die Versuchsdaten alle innerhalb einer Domäne vorkommenden Szenarien vollständig erschöpfend beschreiben. Ansonsten kann für solche Fälle auch ein Default-Wert festgelegt werden.

Regel $\Gamma_S(8)$ kann auch bei doppelt sequentiellen Szenarien angewendet werden, wenn beispielsweise aufgrund der Semantik keine eindeutige Entscheidung möglich ist. Zuerst werden die Werte ϵ_1 (für E_1 und E_2) und ϵ_2 (für E_2 und E_3) berechnet. Überschreitet ein Ereignispaar diese Grenze, so wird ein weiterer Zusammenhang ausgeschlossen. Dieses Vorgehen erweist sich in der Praxis vor allem deswegen als sehr wirksam, weil in vielen Fällen zwischen den einzelnen Ereignispaaren unterschiedliche Werte für ϵ berechnet werden können. Wird die jeweilige Grenze von beiden Paaren überschritten, so werden alle drei Ereignisse als unimodale Interaktionen eingestuft. Überschreitet keines der Ereignispaare diese Grenze, so kann in letzter Instanz aufgrund der Interaktionslänge entschieden werden. Dabei wird wiederum von der Annahme Gebrauch gemacht, dass sequentielle Einzelbefehle im Mittel kürzer sind als unimodale. Es findet ein Vergleich der Interaktionslängen von E_1 und E_3 statt und das Ereignis mit der kürzeren Dauer bildet eine bimodale Interaktion mit E_2.

Die Regeln könnten noch weiter ausgebaut werden. Beispielsweise könnte man die jeweiligen Interaktionslängen nicht nur untereinander, sondern auch mit den spezifischen Durchschnittswerten vergleichen und so präzisere Erkenntnisse erhalten. Die verwendete Form hat den Vorteil, dass auf jeden Fall eine Entscheidung getroffen wird. Nähere Betrachtungen der Szenarien, in denen Regel $\Gamma_S(8)$ überhaupt zum Einsatz kommt, haben ergeben, dass selbst eine falsche Zuordnung in vielen Fällen nach der Fusion zu den gleichen Systemaktionen führt wie eine vollständig korrekte Interpretation. Aus diesem Grund kann für den extrem selten auftretenden Fall, dass E_1 und E_3 identische Interaktionslängen aufweisen, ohne weitere Analysen eine bimodale Interaktion von E_1 und E_2 angenommen werden.

6.4 Problemadäquate natürliche Kodierung

6.4.1 Formale Beschreibung

Der erste Schritt im Rahmen der Konzeption eines genetischen Algorithmus besteht in der Auffindung einer abstrakten Modellierung der Problemdomäne. Dieser Prozess wird allgemein als *Kodierung* bezeichnet. Die folgenden Betrachtungen gehen von einer Gesamtpopulation $\mathcal{P}(q)$ mit $q \in \mathbb{N}^0$ aus. Der Bezeichner q wird als Zähl-Indikator für die Identifikation einer speziellen Generation verwendet. Eine Population besteht jeweils aus ι Individuen mit $\iota \geq 1$:

$$\mathcal{P}(q) = \{\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_\iota\} \tag{6.5}$$

Innerhalb einer Spezies kann ein einzelnes Individuum $\mathcal{I}_i$ mit $i \in \{1..\iota\}$ formal als Menge von μ Parameterwerten dargestellt werden. Die Elemente $g_{i,j}$ mit $j \in \{1..\mu\}$ eines Individuums $\mathcal{I}_i$ bezeichnet man als *Gene*.

$$\mathcal{I}_i = \{g_{i,1}, g_{i,2}, ..., g_{i,\mu}\} \quad \text{mit} \quad \mu \geq 1 \tag{6.6}$$

Gene stellen die atomaren Informationsentitäten dar, mit denen die einzelnen Problemlösungsparameter vollständig spezifiziert werden können. Zusammenfassend lassen sich die genetischen Informationen in einer Population alternativ zu Gleichung 6.5 auch als Matrix der Dimension $\iota \times \mu$ schreiben. Die Zeilen der Matrix repräsentieren jeweils ein Individuum der Population und die Spalten modellieren die Variationsbreite bzgl. eines speziellen Parameters.

$$\mathcal{P}(q) = \begin{pmatrix} \mathcal{I}_1 \\ \mathcal{I}_2 \\ \vdots \\ \mathcal{I}_\iota \end{pmatrix} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,\mu} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,\mu} \\ \vdots & \vdots & \ddots & \vdots \\ g_{\iota,1} & g_{\iota,2} & \cdots & g_{\iota,\mu} \end{pmatrix} \quad \text{mit} \quad \iota,\, \mu \geq 1 \tag{6.7}$$

Bei der allgemeinen Diskussion der genetischen Struktur eines Individuums sind die konkreten Informationsinhalte oftmals nicht von Bedeutung. In diesem Fall werden sowohl das Individuum als auch die einzelnen Gene ohne den Laufindex i zur Identifikation eines spezifischen Individuums $\mathcal{I}_i$ innerhalb der Population kodiert, d.h. $\mathcal{I} = \{g_1, g_2, ..., g_\mu\}$ bezieht sich auf ein abstraktes Individuum der Population.

Zusammengehörige Parameterwerte bilden logische Cluster, die man im Hinblick auf den biologischen Hintergrund als *Chromosome* bezeichnet. Als allgemeines Referenzsymbol wird $\mathcal{G}$ verwendet. Ein Chromosom besteht aus einer fest definierten Ansammlung von Genen, die sich aus einer Teilmenge der Gene eines Individuums $\mathcal{I}$ zusammensetzen. Die einzelnen Chromosomstrukturen sind untereinander disjunkt. Sei ν mit $\nu \geq 1$ die Anzahl der Chromosome in $\mathcal{I}$ und $\mu(x)$ mit $1 \leq \mu(x) \leq \mu$ die Anzahl der Gene in einem spezifischen Chromosom $\mathcal{G}_x$ mit $x \in \{1..\nu\}$. Gilt zusätzlich die Beziehung $\mu = \sum_{x=1}^{\nu} \mu(x)$, so kann ein Individuum $\mathcal{I}$ alternativ zu Gleichung 6.6 auch als Partitionierung der Chromosome beschrieben werden:

$$\mathcal{I} = \bigcup_{x=1}^{\nu} \mathcal{G}_x \quad \text{und} \quad \mathcal{G}_a \cap \mathcal{G}_b = \emptyset, \quad \forall\, a \neq b, \quad a, b \in \{1..\nu\} \tag{6.8}$$

Im Gegensatz zu der Natur ist die Zusammensetzung der Chromosome in einem GA für alle Individuen einer Population identisch. Für eine mathematisch eindeutige Darstellung der einzelnen Individuen, Chromosome und Gene müsste jeweils noch der Laufindex q zur Identifikation einer speziellen Generation $\mathcal{P}(q)$ mitgeführt werden. Aus Gründen der Übersichtlichkeit wird jedoch darauf verzichtet, wenn aus dem Kontext klar ersichtlich ist, um welche Generation es sich aktuell handelt.

In technische motivierten Optimierungsproblemen ist der Suchraum zumeist durch eine Menge von realen Objekten definiert, wie beispielsweise in einem industriellen Fertigungsprozess Antriebe, Heizelemente, Pumpen und Ventile. Die Verarbeitungseinheiten verfügen über spezifische Charakteristika wie Energieaufnahme, Geschwindigkeit, Durchflusskoeffizienten, durchschnittliche Funktionsdauer und viele weitere Mess- und Erfahrungswerte. Vor diesem Hintergrund besteht eine Optimierung in der Bestimmung einer Parameterkonfiguration, mit der sich die Gesamtkosten der Produktion minimieren lassen. Die Menge aller theoretisch möglichen Parametervariationen bildet den *Phänotyp* ($\mathcal{Z}$).

Auf systemtechnischer Ebene arbeiten die genetischen Operatoren auf abstrakten mathematischen Objekten. Die genetische Zusammensetzung eines Individuums aus den verschiedenen Chromosomen bezeichnet man als *Genotyp* ($\mathcal{Y}$). In dem Genotyp sind sämtliche Informationen enthalten, um eine konkrete Lösung für ein vorgegebenes Problem formal zu beschreiben. Eine zentrale funktionale Komponente in jedem evolutionären Algorithmus ist die *genetische Dekodierfunktion* Ψ_D. Sie bildet die Daten im Genotyp auf eine Struktur im Phänotyp ab und realisiert damit die Ableitung einer spezifischen Problemlösung auf Basis der einzelnen genetischen Informationen.

$$\Psi_D : \mathcal{Y} \rightarrow \mathcal{Z} \tag{6.9}$$

In der ursprünglichen Form der genetischen Algorithmen werden die einzelnen Gene eines Individuums in einem binären Alphabet kodiert. Zu dieser Art der Kodierung sind vielfältige theoretische und empirische Ergebnisse verfügbar, die das Design von Selektions-, Crossover- und Mutationsfunktion extrem vereinfachen [52]. Auf der anderen Seite benötigt man eine komplexe Dekodierfunktion, um die individuellen Parameterkonfigurationen für ein vorgegebenes Problem adäquat modellieren zu können. Über diese Abbildung können unter Umständen verschiedene Nicht-Linearitäten entstehen, mit denen der Problemlösungsprozess zusätzlich erschwert wird [19].

Demgegenüber erlaubt die von Michalewicz [101] vorgeschlagene *natürliche Kodierung* eine problemangepasste Repräsentation der einzelnen Gene. Auf diese Weise ist das genetische Vokabular nicht auf ein binäres Alphabet begrenzt, sondern es können im Prinzip beliebige abstrakte Datentypen als atomare Informationseinheiten verwendet werden. Darüber hinaus erlaubt diese Art der Kodierung eine typ-polymorphe Struktur innerhalb der einzelnen Chromosome eines Individuums.

Die Verwendung der natürlichen Kodierung ermöglicht die Konzeption einfacher genetischer Zielfunktionen, da bereits der Genotyp eine problemangepasste Struktur aufweist. Im Gegensatz zu einer rein binären Kodierung existieren jedoch keine Standardverfahren zur Ableitung der genetischen Operatoren. Sie müssen speziell an die Datenstruktur angepasst werden und erfordern daher sowohl Erfahrung im Umgang mit genetischen Algorithmen als auch spezifische Kenntnisse aus der Problemdomäne.

6.4.2 Chromosom-Zusammensetzung

Für die Modellierung der multimodalen Informationsanteile wird eine natürliche Kodierung verwendet, die speziell an die Integrationsproblematik angepasst ist. Durch die Nutzung abstrakter Datentypen kann der Aufwand bei der Entwicklung einer geeigneten Datenrepräsentation und entsprechender Zugriffsmethoden erheblich reduziert werden. Es wird direkt auf die existierenden Datenobjekte zurückgegriffen. Darüber hinaus besteht die Möglichkeit, problemspezifische Anforderungen in die Konzeption der genetischen Operatoren einfließen zu lassen. Im Hinblick auf Optimierungs- und Kombinationsprobleme lassen sich mit einer natürlichen Kodierung insgesamt schnellere, konsistentere und genauere Ergebnisse erzielen als mit einer Standard-Kodierung [73]. Daher bietet sich die Nutzung auch im Rahmen eines evolutionären Integrationsverfahrens an.

Der Fusionsalgorithmus basiert auf der Auswertung unterschiedlicher Datenquellen. Die relevanten Informationen können verschiedenen Kategorien zugeordnet werden, die jeweils einer abgeschlossenen Chromosomstruktur entsprechen. Jedes Individuum $\mathcal{I}$ besteht prinzipiell aus vier unterschiedlichen Chromosomanteilen. Die Anzahl der Chromosome innerhalb der einzelnen Strukturen ist zum Teil von der aktuellen Anwendungsdomäne und den spezifischen Randbedingungen in dieser Umgebung abhängig.

$$\mathcal{I} = (\mathcal{G}_M, \mathcal{G}_R, \mathcal{G}_K, \mathcal{G}_A) \tag{6.10}$$

Im Folgenden wird auf diese Kategorien näher eingegangen. Grundlegende Angaben zur Konzeption der angepassten Chromosome sind in [8] enthalten. Die genetische Struktur sowie die systemtechnische Umsetzung der dazugehörigen abstrakten Datentypen werden ausführlich in [5, 6] behandelt. Ein Schwerpunkt liegt auf einer semi-formalen Beschreibung der individuellen Manipulationsmöglichkeiten.

Speicher-Chromosom

Der erste Chromosomblock $\mathcal{G}_M$ besteht lediglich aus einem einzigen Element, das als *Speicher-Chromosom* bezeichnet wird. Diese Struktur beinhaltet sämtliche Informationen über die letzten Integrationsergebnisse mit den dazugehörigen Konfidenz- und Zeitwerten, eine Liste von alternativen Aktionshypothesen, eine detaillierte Angabe von Mutationsmöglichkeiten sowie erwartungsbezogene Angaben zu möglichen Folgeaktionen und komplementären Ergänzungen. Für neue Individuen wird $\mathcal{G}_M$ automatisch mit den Daten aus dem letzten Integrationsergebnis gefüllt. Auf diese Weise verfügen alle Individuen in einer Startpopulation über absolut identische Speicherchromosome.

Im Laufe des Integrationsprozesses kann das Speicherchromosom durch den Mutationsoperator manipuliert werden. Die Wahrscheinlichkeit, mit der $\mathcal{G}_M$ einer Mutation unterzogen wird, hängt von den Bewertungen der gespeicherten Integrationsergebnisse ab. Gute und stabile Ergebnisse werden weniger oft mutiert und verteilen sich daher stärker unter den Individuen einer Population als schlechter bewertete Aktionshypothesen. Durch die Mutation besteht die Möglichkeit, fehlerhafte Integrationsergebnisse rückwirkend zu korrigieren, wodurch ein evtl. negativer Einfluss auf den aktuellen Integrationsprozess vermindert wird.

Die Ergebnisse der vorherigen Integrationsphasen sind in Form von einzelnen Semunen bzw. Semunfolgen in $\mathcal{G}_M$ abgelegt. In einem speziellen Gen werden pro Semun die möglichen Mutati-

onswerte gespeichert. Auf diese Weise kann sichergestellt werden, dass im Mutationsprozess ausschließlich Semune erzeugt werden, die genau wie das Ausgangssemun ein syntaktisch korrektes Element der zugrundeliegenden kontextfreien Grammatik darstellen. Durch die Einschränkung der Mutationsmöglichkeiten wird die Umsetzung einer nachfolgenden Korrekturfunktion (engl. *repair function*) überflüssig gemacht. Die Parametrisierung des Speicherchromosoms beruht auf problemspezifischem Domänenwissen und erfolgt analog zu den Darstellungen in Abschnitt 5.4 in Form von LUTs durch externe Datenbasen.

Erkenner-Chromosome

Der zweite Chromosomblock $\mathcal{G}_R$ enthält eine Repräsentation der einzelnen Benutzereingaben E_R in einem vorsegmentierten Spotting-Intervall C_S. Mit Bezug auf die Modulstruktur in Abbildung 5.2 werden die Chromosome in dieser Kategorie als *Erkenner-Chromosome* bezeichnet. Jedes Individuum einer Population verfügt über die gleiche Anzahl an Erkennerchromosomen $\nu(R)$. Diese Anzahl kann für unterschiedliche Integrationsprozesse variieren, speziell wenn sich im laufenden Betrieb des Systems die Konfiguration der Erkennermodule ändert, neue Module hinzukommen oder bestehende Module abgekoppelt werden.

Zu Beginn einer Integration wird für jedes Erkennermodul $M_{R,i}$ mit $i \in \{1..\rho_R\}$ ein spezifisches Erkennerchromosom im Genotyp erzeugt. Diese Basismenge bleibt in ihrer Zusammensetzung über die Integrationsdurchläufe hinweg konstant. Die Elemente werden als *Standard-Erkenner-Chromosome* bezeichnet und mit $\mathcal{G}_{R(S)}$ abgekürzt, die Kardinalität der Menge wird durch $\nu(R(S))$ angegeben. Sind in einem System beispielsweise zwei haptische Eingabegeräte, ein Spracherkenner und ein Kopfgestenerkenner angeschlossen, dann sind in $\mathcal{G}_R$ zunächst fünf Chromosome enthalten ($\nu(R)=\nu(R(S))=\rho_R=5$).

In einem Chromosom $G_{R(S)} \in \mathcal{G}_{R(S)}$ werden unterschiedliche Informationen über das Erkennermodul und die erkannte Benutzereingabe abgelegt. Ein wichtiger Bestandteil der Datenstruktur ist beispielsweise die Identifikation der Erkennerkomponente. Systemtechnisch erfolgt dies über eine Auswertung des Eintrags $< SourceID >$ in den Modulnachrichten (siehe Abschnitt 5.5.1). Zusätzlich werden der Typ und eine generelle Erkennungssicherheit der Eingabekomponente abgespeichert. Bezogen auf das aktuelle Erkennungsergebnis enthält $G_{R(S)}$ ein Semun der entsprechenden kontextfreien Grammatik, detaillierte Zeitinformationen, Konfidenzwerte und eine Liste von alternativen Resultaten. Für jedes Semun beinhaltet $G_{R(S)}$ darüber hinaus Angaben zu der spezifischen Mutationsrate und den entsprechenden Variationsmöglichkeiten. Dabei wird jeweils zwischen redundanten, komplementären und rivalisierenden Informationsanteilen unterschieden.

Eine Änderung der Modulkonfiguration entspricht einer Modifikation von $\mathcal{G}_{R(S)}$. Für jede neue Erkennerkomponente wird ein dediziertes Erkenner-Chromosom hinzugefügt bzw. für abgeschaltete Module das entsprechende Element aus der Basismenge entfernt. Sämtliche Erkennermodule werden, wie in Abschnitt 5.3.2 beschrieben, durch einen einheitlichen Formalismus modelliert. Der Integrationskern operiert auf dieser allgemeinen Struktur und kann daher optimal an wechselnde Anwendungsszenarien angepasst werden.

Neben den standardmäßigen Erkennerchromosomen $\mathcal{G}_{R(S)}$ verfügt jedes Individuum zusätzlich noch über eine gewisse Anzahl $\nu(R(E))$ an *Extra-Erkenner-Chromosomen* $\mathcal{G}_{R(E)}$. Bezogen auf die modellierten Informationsinhalte stimmen die Elemente aus $\mathcal{G}_{R(E)}$ mit denen aus $\mathcal{G}_{R(S)}$ exakt überein. Sie werden immer dann automatisch im Genotyp erzeugt, wenn ein Erkennermodul eine Sequenz von mehreren Semunen liefert.

Sei $E_R = S_a S_b S_c$ das transformierte Ergebnis einer fiktiven Benutzereingabe, dann wird beispielsweise S_a in einem Standard-Chromosom und S_b und S_c jeweils in einem eigenen Extra-Chromosom abgelegt. Da die beiden Chromosom-Kategorien algorithmisch vollkommen gleich behandelt werden, ist es nicht von Bedeutung, an welcher Stelle die einzelnen Semune gespeichert werden. Die Aufteilung der Semune vereinfacht den anschließenden Fusionsmechanismus und wirkt sich positiv auf das Konvergenzverhalten aus. Darüber hinaus ermöglicht diese Struktur die Durchführung genetischer Operatoren auf den einzelnen Semunen einer Benutzereingabe. Die übrigen Elemente eines Ereignisses bleiben unverändert.

Im Gegensatz zu $\nu(R(S))$ ist die Anzahl der Extra-Chromosome $\nu(R(E))$ nicht notwendigerweise konstant und kann sich sogar während eines laufenden Integrationsprozesses verändern. Eine zentrale Anforderung besteht jedoch darin, dass $\nu(R(E))$ für alle Individuen einer Population identisch ist. Die Menge der Erkennerchromosome ergibt sich aus der Summe der standardmäßigen und der zusätzlichen Erkennerchromosome.

$$\nu(R) = \nu(R(S)) + \nu(R(E)) \tag{6.11}$$

Kontext-Chromosome

Der dritte Chromosomblock, die *Kontext-Chromosome* $\mathcal{G}_K$ sind direkt von den Erkennerchromosomen abgeleitet. Bezogen auf die Datenstruktur stimmen die beiden Kategorien vollständig überein. Ein Element $G_K \in \mathcal{G}_K$ beinhaltet Informationen über die Kontextereignisse, diverse Zeitinformationen, alternative Erkennungsergebnisse und detaillierte Angaben zu dem entsprechenden Kontextmodul. Analog zu der Aufteilung in Abschnitt 5.2.1 lassen sich die Kontextchromosome im Hinblick auf die Modellierung des Benutzerkontextes $\mathcal{G}_{K(B)}$, des Systemkontextes $\mathcal{G}_{K(S)}$ und des Umgebungskontextes $\mathcal{G}_{K(U)}$ aufteilen. An dieser Stelle werden u.a. auch die verschiedenen Zustandsinformationen $\mathcal{Z}_S$, $\mathcal{Z}_B$ und $\mathcal{Z}_U$ für die kontextspezifischen Abbildungen in den Funktionen 5.16, 5.18 und 5.19 gespeichert.

Die Anzahl der Elemente in den einzelnen Kategorien $\nu(K(B))$, $\nu(K(S))$, $\nu(K(U))$ hängt von der aktuellen Konfiguration des Systems und der momentanen Bediensituation ab. Vergleichbar mit den zusätzlichen Erkennerchromosomen $\mathcal{G}_{R(E)}$ kann der Umfang von $\mathcal{G}_K$ zwischen verschiedenen Integrationsläufen variieren, ist aber wiederum für alle Individuen einer Population identisch. Die Gesamtanzahl der Kontextchromosome $\nu(K))$ ergibt sich durch Summation der kategoriespezifischen Umfänge.

$$\nu(K) = \nu(K(B)) + \nu(K(S)) + \nu(K(U)) \tag{6.12}$$

Die explizite Trennung der Eingabeereignisse in Erkenner- und Kontextchromosome resultiert zum einen aus dem modularen Aufbau der multimodalen Basis-Architektur und zum anderen aus der Anforderung einer phasen-orientierten Datenfusion. Kontextinformationen werden erst auf der zweiten Stufe im Integrationsprozess berücksichtigt. Soll ausschließlich eine Symbolfusion durchgeführt werden, bleiben die Kontextchromosome leer und üben auf diese Weise keinen Einfluss auf die Ableitung der Aktionshypothesen aus. Die einzelnen Integrationsstufen lassen sich wesentlich einfacher realisieren, wenn die unterschiedlichen Informationsanteile separat voneinander modelliert werden.

Aktions-Chromosom

Der vierte Chromosomblock $\mathcal{G}_A$ beinhaltet eine Modellierung des aktuellen Integrationsergebnisses. Jedem Individuum einer Population können potenzielle Aktionshypothesen zugeordnet werden, die gemeinschaftlich durch ein entsprechendes Chromosom repräsentiert werden. Im Hinblick auf die Intentionsfunktion Φ aus Gleichung 5.6 wird dieses Chromosom daher auch als *Aktions-Chromosom* bezeichnet. Neben einer präferierten Systemaktion enthält $\mathcal{G}_A$ auch eine quantitative Bewertung der Aktionshypothese, eine Liste mit alternativen Integrationsergebnissen sowie zusätzliche Informationen über evtl. benötigte komplementäre Informationsanteile und entsprechende Einschränkungen für das dazugehörige Zeitfenster.

Das Aktionschromosom hängt von den jeweiligen Inhalten der anderen Chromosome ab. Sowohl die Aktionen als auch die Semune sind Elemente der im Integrator benutzten formalen Sprache $\mathcal{L}_I$. Bezogen auf die kontextfreie Notation lässt sich eine eindeutige strukturelle Abhängigkeit nachweisen. Ein Wort A_a setzt sich jeweils aus einem einzelnen Semun S_a oder einer Verkettung von mehreren Semunen $(S_{a,1}S_{a,2}...)$ zusammen.

Sei $\mathcal{S}$ die Menge aller zulässigen Semune in $\mathcal{L}_I$ und $\mathcal{A}$ die Menge der syntaktisch korrekten Aktionen, die sich unter Beachtung der Regeln einer kontextfreien Applikationsgrammatik aus der Potenzmenge $\mathcal{S}^*$ bilden lassen. Die Semune werden nach der Zugehörigkeit zu den einzelnen Chromosomanteilen unterschieden, wobei $\mathcal{S}_M$, $\mathcal{S}_R$ und $\mathcal{S}_K$ jeweils die Menge der Semune im Speicherchromosom $\mathcal{G}_M$, in den Erkennerchromosomen $\mathcal{G}_R$ bzw. in den Kontextchromosomen $\mathcal{G}_K$ repräsentieren. Auf Basis dieser Modellierung kann die allgemein formulierte genetische Dekodierfunktion Ψ_D aus Gleichung 6.9 wie folgt konkretisiert werden:

$$\Psi_D : \mathcal{S}_M \oplus \mathcal{S}_R \oplus \mathcal{S}_K \rightarrow \mathcal{A} \tag{6.13}$$

Für die Bestimmung von $\mathcal{G}_A$ werden verschiedene Regeln angewendet. Im einfachsten Fall stimmen die Semune der einzelnen Erkenner- und Kontextchromosome vollständig überein, d.h., es gilt $S = S_i$ für alle $i \in \{1..(\nu(R) + \nu(K))\}$ und eine potenzielle Systemaktion $A_x \equiv S$ kann direkt aus diesem Semun gebildet werden. Wenn ein spezielles Semun S_a eindeutig den Phänotyp dominiert, dann wird S_a, evtl. in Kombination mit einem dominierenden komplementären Semun S_b als Aktion ausgewählt. Falls zu einem Semun S_a mehrere komplementäre Semune S_c, S_d existieren, berechnet der Algorithmus eine Bewertung für die gültigen Kombinationen S_aS_c und S_aS_d und wählt daraus das beste Ergebnis aus. Der Prozess zur Ableitung einer Aktionshypothese wird detailliert in Abschnitt 6.5 diskutiert.

Im Gegensatz zu den anderen Chromosomen eines Individuums wird $\mathcal{G}_A$ nicht durch genetische Operatoren manipuliert. Jedes Mal, wenn sich eine genetische Information im Genotyp ändert, muss das Aktionschromosom neu bestimmt werden. Es ist das einzige Chromosom in einem Individuum, das von den einzelnen Chromosomen der anderen Kategorien abhängt. Aus diesem Grund ist wenig sinnvoll, Teilinformationen zu mutieren oder in einem Crossover-Prozess mit anderen Individuen auszutauschen. Ein Eingriff an dieser Stelle könnte den Integrationsprozess unter Umständen völlig verfälschen. Das Aktionschromosom repräsentiert auf abstrakter Ebene ein vollständiges Individuum. In der iterativen Datenfusion wird hauptsächlich auf dieses Chromosom zugegriffen. Die Aktionschromosome in einer Population bilden die Datenbasis für die Auswahl einzelner Individuen in den spezifischen Selektions- und Rekombinationsprozessen. Darüber hinaus fließen die Bewertungen der Aktionshypothesen in die Überprüfung der Konvergenzkriterien und in die letztendliche Bestimmung des Integrationsergebnisses ein.

Eintrag	Beschreibung
$L(S,0)$	Semun S, Referenzbasis für die folgenden Einträge
$L(S,1)$	Liste von möglichen vorangegangenen Aktionen
$L(S,2)$	Liste von komplementären Semunen
$L(S,3)$	Liste von redundanten Semunen
$L(S,4)$	Liste von rivalisierenden Semunen
$L(S,5)$	Liste von benötigten Kombinations-Semunen
$L(S,6)$	Liste von möglichen Folge-Semunen mit Auftrittswahrscheinlichkeiten
$L(S,7)$	Liste von Mutationsmöglichkeiten mit spezifischen Wahrscheinlichkeiten

Tab. 6.1: Struktur einer domänenspezifischen Lookup-Tabelle $L(S)$ für ein einzelnes Semun S

6.4.3 Domänen-Konfiguration

Der generisch konzipierte Integrationskern kann durch eine geeignete Parametrisierung an die aktuelle Anwendungsdomäne angepasst werden. Diese Adaption ist eine fundamentale Voraussetzung, um im praktischen Einsatz des Systems effektive Integrationsergebnisse zu erzielen und die Erwartungen der Benutzer hinsichtlich einer einfachen und fehlerrobusten Bedienung zu erfüllen. Die Konfiguration erfolgt analog zu den Konvertermodulen aus Abschnitt 5.3.3 in Form von externen Lookup-Tabellen. In diesen Tabellen werden die einzelnen Semune der verschiedenen Applikationsmodule funktional zueinander in Beziehung gesetzt. Für die Erstellung einer Lookup-Tabelle L ist zum Teil domänenspezifisches Wissen nötig. Eine Portierung des Gesamtsystems besteht im Wesentlichen in der Bestimmung neuer, problemadäquater Lookup-Tabellen L_{neu}. Im Gegensatz dazu bleiben die Struktur und die Funktionsweise des Integrationskerns in der Grundversion unverändert.

Für jedes Semun aus der kontextfreien Sprache der Applikationsmodule $S \in \mathcal{L}(\mathcal{A})$ existiert eine eigene Lookup-Tabelle $L(S)$. Die grundlegende Struktur ist in Tabelle 6.1 dargestellt. Der erste Eintrag $L(S,0)$ beinhaltet das Referenzsemun S. Alle weiteren Einträge beziehen sich ausschließlich darauf. Als nächstes werden Informationen zu möglichen vorhergegangenen Aktionen $L(S,1)$ abgelegt. Generell sind hier alle Semune erlaubt. In gewissen Situationen kann die Menge der Aktionen jedoch eingeschränkt werden. Wenn das Semun S beispielsweise immer nur in einer der Sequenzen S_aS, S_bS oder S_cS auftritt, so besteht $L(S,1)$ aus der Menge $\{S_a, S_b, S_c\}$. Für diese Einschränkung ist empirisches Wissen aus der Anwendungsdomäne nötig. Ist dieses Wissen nicht explizit verfügbar, muss auf Standardeinstellungen zurückgegriffen werden.

Die nächsten drei Einträge der Lookup-Tabelle $L(S,2)$, $L(S,3)$, und $L(S,4)$ beinhalten eine Auflistung der komplementären, redundanten und rivalisierenden Semunen. Ohne eine explizite Angabe dieser Informationen wäre eine Bewertung der Aktionshypothesen im Rahmen des genetischen Fusionsprozesses nicht möglich. Die Werte gehen als zentrale Merkmale in die Berechnung der genetischen Fitness ein (siehe nächster Abschnitt).

Das sechste Datenfeld $L(S,5)$ enthält eine Menge an Semunen $\{S_x, S_y, S_z, ...\}$, die im Zusammenhang mit dem aktuellen Semun S benötigt werden, um syntaktisch korrekte Wörter $\{SS_x, S_xS, SS_y, ...\}$ der Applikationsgrammatik zu erzeugen. An dieser Stelle können auf Basis der quantitativen Erfahrungen aus den Benutzerstudien zusätzlich noch Informationen über

spezifische Zeitfenster abgelegt werden, in denen nach den jeweiligen Semunen gesucht werden soll. Der vorletzte Eintrag $L(S, 6)$ beschreibt eine Liste von möglichen Folge-Semunen, jeweils mit einer spezifischen Angabe zu der Auftrittswahrscheinlichkeit. Ähnlich zu $L(S, 1)$ sind hier zumeist sämtliche Semune der erzeugenden Grammatik zulässig.

Im letzten Eintrag $L(S, 7)$ werden jeweils die einzelnen Möglichkeiten, zu denen sich das aktuelle Semun S im Mutationsprozess verändern kann, in expliziter Form angegeben. Diese Liste ist nach den Auftrittswahrscheinlichkeiten absteigend sortiert. Die Mutationsmöglichkeiten werden direkt in den Chromosomen gespeichert, um die eventuelle Erzeugung ungültiger Semune bereits im Vorfeld vollständig zu vermeiden. Generell kann der Fusionsmechanismus durch den Einbezug von Kontextwissen aus vorangegangenen Benutzerstudien wesentlich besser an das aktuelle Anwendungsszenario angepasst werden. Das Wissen kann in unterschiedlichen Granularitätsstufen vorliegen.

6.5 Quantitative Bewertung

6.5.1 Individuelle Fitness

In Analogie zu dem Vorbild aus der Natur weisen die einzelnen Individuen einer Population von Integrationsergebnissen normalerweise stark unterschiedliche Charakteristika auf. Um die Qualität der verschiedenen Aktionshypothesen bewerten und vergleichen zu können, wird ein quantitatives Maß zur Beurteilung der einzelnen Parameterkonfigurationen eingeführt. Die entsprechende Berechnungsfunktion nennt man *Genetische Zielfunktion*.

$$\Psi_G : \mathcal{I} \rightarrow \mathbb{R} \tag{6.14}$$

Das Ergebnis wird im Allgemeinen auf den Wertebereich der reellen Zahlen abgebildet und als *individuelle genetische Fitness* $\omega(\mathcal{I})$ bezeichnet. Nach der bei genetischen Algorithmen üblichen Konvention fällt die individuelle Bewertung $\omega(\mathcal{I})$ umso höher aus, je besser das Individuum $\mathcal{I}$ eine optimale Lösung der Integrationsaufgabe approximiert. Die Funktion Ψ_G sollte einen gleichmäßig stetigen Wertebereich besitzen, so dass ähnliche Parameterkonfigurationen auch auf eine ähnliche Bewertung abgebildet werden.

Für verschiedene Fragestellungen ist eine lokale Betrachtung der Ergebnisse nur bedingt aussagekräftig. Aus diesem Grund wird auf Basis der individuellen Bewertungen in einem nachfolgenden Berechnungsschritt eine *populationsbezogene genetische Fitness* $\omega_P(\mathcal{P})$ ermittelt. Dieser Wert erlaubt in erster Näherung eine zusammenfassende Beurteilung der unterschiedlichen Integrationsalternativen in einer Population. Sei $\mathcal{P}$ eine Population mit insgesamt ι Elementen, dann ergibt sich $\omega_P(\mathcal{P})$ aus der Summe der individuellen Bewertungen.

$$\omega_P(\mathcal{P}) = \sum_{i=1}^{\iota} \omega(\mathcal{I}_i) \tag{6.15}$$

Ein Nachteil dieser Bewertung ist, dass vereinzelt auftretende Extremwerte das Gesamtergebnis überproportional stark beeinflussen. Um die Qualität der individuellen Bewertungen auch über mehrere Generationen hinweg besser vergleichen zu können, wird die individuelle

Fitness häufig in einer normierten Form angegeben. Die *standardisierte individuelle genetische Fitness* $\bar{\omega}(\mathcal{I}_i)$ ergibt sich aus dem Quotienten der einzelnen Fitnesswerte $\omega(\mathcal{I}_i)$ mit $i \in \{1..\iota\}$ und der Bewertung für die gesamte Population $\omega(\mathcal{P})$.

$$\bar{\omega}(\mathcal{I}_i) = \frac{\omega(\mathcal{I}_i)}{\omega(\mathcal{P})} \quad \forall\, \mathcal{I}_i \in \mathcal{P} \tag{6.16}$$

Bei numerischen Maximierungs- bzw. Minimierungsaufgaben kann die Bewertung eines Individuums $\mathcal{I}$ relativ einfach ermittelt werden, da $\omega(\mathcal{I})$ zumeist mit dem Ergebniswert der untersuchten Funktion übereinstimmt. Für allgemeine Parameteroptimierungsprobleme ist dies weitaus schwieriger. Eine direkte Angabe von $\omega(\mathcal{I})$ durch einen vollständig differenzierbaren analytischen Ausdruck ist hier nur in wenigen Ausnahmefällen möglich. Die Auffindung einer aussagekräftigen Bewertungsfunktion gestaltet sich als zentrales Problem beim Design genetischer Algorithmen. Es ist ein kreativer Prozess, in dem sowohl Wissen aus der Problemdomäne als auch Erfahrung im Umgang mit evolutionären Berechnungsmethoden benötigt wird.

Im Rahmen einer evolutionären Integration müssen die in den Individuen modellierten multimodalen Informationsanteile zu den potenziellen Systemaktionen in Beziehung gesetzt werden. Die Bewertung der Chromosomanteile wird auch als *Score* bezeichnet und repräsentiert ein Maß für die Konsistenz des Phänotyps. Je konsistenter der Phänotyp ist, umso sicherer kann davon ausgegangen werden, dass das Integrationsergebnis mit der Intention des Benutzers übereinstimmt. Dieser Schluss ist allerdings nur dann gültig, wenn das System, wie in der vorliegenden Arbeit, maßgeblich nach dem Äquivalenz-Prinzip entwickelt wurde (siehe Abschnitt 5.1.2).

Im weiteren Sinne kann ein Score auch als Wahrscheinlichkeit interpretiert werden. Die Bewertungen bedürfen jedoch einer sorgfältigen Interpretation, da sie keine Wahrscheinlichkeiten im klassischen mathematischen Sinne darstellen. Sie eignen sich definitiv nicht für den Vergleich der Integrationsergebnisse aus unterschiedlichen Anwendungsszenarien, weil die Ergebnisse von Ψ_G massiv von den konkreten Randbedingungen abhängen. Ein Wert von 100% würde bedeuten, dass zum einen sämtliche Erkennerchromosome gleichzeitig ein identisches Semun enthalten und zum anderen die dahinter stehenden individuellen Erkennungsprozesse jeweils mit absoluter Sicherheit erfolgten. Im allgemeinen Fall ist diese Konstellation eher untypisch. Eine wichtige Erfahrung aus den Benutzerstudien ist beispielsweise, dass speziell in einem System mit mehreren Erkennerkomponenten zumeist nur mit einer gewissen Auswahl von zwei oder maximal drei Modalitäten gleichzeitig interagiert wird. Die Bewertungen fallen im Mittel daher umso niedriger aus, je mehr Komponenten in dem System enthalten sind.

6.5.2 Integrationsspezifischer Berechnungsansatz

Auf die Bestimmung der individuellen genetischen Fitness $\omega(\mathcal{I})$ wird im Folgenden näher eingegangen. Den Ausgangspunkt bilden die benutzerbezogenen Eingabeereignisse $E_R \in \mathcal{E}_R$ in einem spezifischen Spottingintervall $\mathcal{C}_S$. Jedem $E_R \in \mathcal{C}_S$ kann, wie in Abschnitt 5.3.3 beschrieben, eindeutig ein Wort w einer übergeordneten kontextfreien Sprache $\mathcal{L}_I$ zugeordnet werden. Ein solches Wort w_x besteht aus einem oder mehreren Semunen $w_x = \{(S_x), (S_{x,1}S_{x,2}), \dots\}$, die zusammen mit verschiedenen Zusatzinformationen in den Erkennerchromosomen $\mathcal{G}_R$ eingetragen werden. Die Menge der einzelnen Semune in $\mathcal{G}_R$ wird abkürzend mit $\mathcal{S}_R$ bezeichnet. Diese Menge enthält sowohl die primären Erkennungsergebnisse als auch mögliche Alternativen, soweit sie von den Erkennermodulen in Form von *n-best* Listen zur Verfügung gestellt werden.

Sei $\mathcal{I}$ ein beliebiges Individuum aus einer Population $\mathcal{P}$ und $\mathcal{S}_R(\mathcal{I})$ eine Beschreibung der transformierten Benutzereingaben, die in den Erkennerchromosomen von $\mathcal{I}$ modelliert sind. Zur Vereinfachung der Darstellung wird die Abkürzung $\mathcal{S}_R := \mathcal{S}_R(\mathcal{I})$ verwendet. Für jedes Semun $S \in \mathcal{S}_R$ existiere darüber hinaus ein Maß für die Güte des jeweiligen Erkennungsprozesses. Dieser Wert wird als *semunbezogene Basisbewertung* $p_b(S)$ bezeichnet und repräsentiert das Konfidenzmaß einer Erkennung. Sämtliche Bewertungen werden auf das Standard-Intervall [0..1] abgebildet. Sind keine quantitativen Aussagen über die Qualität der Erkennungsergebnisse verfügbar, so wird $p_b(S)$ mit dem Wert 1 angesetzt.

Sei $\breve{M}_R(S)$ eine Bezeichnung für dasjenige Erkennermodul $M_{R,i}$ mit $i \in \{1..\rho(R)\}$, von dem das Semun S erzeugt wurde. Die erste Berechnungsoperation besteht darin, für jedes Semun $S \in \mathcal{S}_R$ die Güte des semunbezogenen Erkennungsprozesses $p_b(S)$ mit der allgemeinen Erkennungssicherheit des entsprechenden Eingabemoduls $p_D(\breve{M}_R(S))$ zu multiplizieren. Der Wert $p_D(\breve{M}_R(S))$ stammt aus einer anwendungsspezifischen Lookup-Tabelle. Sind keine detaillierten Informationen vorhanden, wird $p_D(\breve{M}_R(S))$ analog zu $p_b(S)$ mit 1 vorbelegt. Das Ergebnis ist eine *erweiterte semunbezogene Basisbewertung* $p_m(S)$, die sowohl von den individuellen Erkennungsergebnissen als auch von den Typen der jeweiligen Erkennungsmodule abhängt.

$$p_m(S) = p_b(S)\; p_D(\breve{M}_R(S)) \quad \forall\, S \in \mathcal{S}_R \tag{6.17}$$

Die Einführung der Gewichtung in der ersten Stufe der Bewertung ermöglicht später einen qualitativ höherwertigeren Vergleich der einzelnen Integrationshypothesen. Beispielsweise haben taktile Interaktionen über eine Tastenkonsole im Allgemeinen eine weitaus bessere Erkennungssicherheit als natürlichsprachliche Eingaben. Um das Leistungspotenzial dieses Ansatzes aber vollständig ausschöpfen zu können, müssen für die einzelnen Faktoren in Gleichung 6.17 entweder alle beteiligten Erkennermodule Konfidenzwerte liefern, oder der entsprechende Faktor wird generell auf 1 gesetzt.

Zeitlicher Gewichtungsfaktor

Benutzereingaben, die in relativ kurzen Zeitabständen nacheinander erfolgen, gehören mit einer höheren Wahrscheinlichkeit zusammen, als Benutzereingaben, die zeitlich gesehen weit auseinander liegen. Der experimentelle Nachweis dieser Beobachtung ist in Benutzerstudien von unterschiedlichen Forschungsinstituten erbracht und mehrfach bestätigt worden, z.B. in [11, 121, 154]. Um diesen empirischen Befund auch in dem evolutionären Fusionsansatz integrieren zu können, wird ein *zeitlicher Gewichtungsfaktor* θ eingeführt, der bei Bedarf auf die modifizierten Bewertungen $p_m(S)$ in Gleichung 6.17 angewendet werden kann.

Im Hinblick auf das Integrationsproblem ist speziell der Auftrittszeitpunkt eines Semuns S innerhalb des Spottingintervalls $\mathcal{C}_S$ von Interesse. Sei $t(S)$ der absolute Zeitpunkt von S und $t_s(\mathcal{C}_S)$ und $t_e(\mathcal{C}_S)$ der Anfang bzw. das Ende von $\mathcal{C}_S$ mit $t_s(\mathcal{C}_S) \leq t(S) < t_e(\mathcal{C}_S)$. Der normierte relative Startpunkt $t'(S)$ ergibt sich aus der Differenz von $t(S)$ und $t_s(\mathcal{C}_S)$ mit Bezug auf die Dauer des Spottingintervalls. Diese Zeiten werden für jedes Ereignis $E \in \mathcal{C}_S$ bereits während des Spottings automatisch erzeugt und in den Chromosomen abgelegt.

$$t'(S) = \frac{t(S) - t_s(\mathcal{C}_S)}{t_e(\mathcal{C}_S) - t_s(\mathcal{C}_S)} \quad \forall\, S \in \mathcal{S}_R \tag{6.18}$$

Zur Simulation der zeitlichen Abhängigkeitseffekte kann zwischen verschiedenen Funktionen gewählt werden. Im praktischen Einsatz wird jedoch zumeist ein exponentieller Gewichtungsfaktor bevorzugt, da dieser flexibel an die anwendungsspezifischen Randbedingungen angepasst werden kann. Das Ausmaß des zeitlichen Abfalls wird zusätzlich über einen domänenspezifischen Reduktionsparameter f_e bestimmt.

$$\theta_S(S) = e^{(1-t'_s(S))\, f_e} \quad \text{mit} \quad 0 < f_e \leq 1 \tag{6.19}$$

6.5.3 Ableitung einer Aktionshypothese

Im nächsten Schritt werden die modifizierten Semunbewertungen zu den möglichen Integrationsergebnissen in Beziehung gesetzt. Sei $\mathcal{A} = \{A_1, A_2, \ldots, A_\alpha\}$ eine vollständige Auflistung der Systemaktionen und $\sigma(\mathcal{I})$ die Gesamtanzahl der Semune in $\mathcal{I}$. Weiterhin sei $\sigma(A)$ mit $\sigma(A) \leq \sigma(\mathcal{I})$ die Anzahl der Semune in $\mathcal{I}$, die eine spezielle Systemaktion $A \in \mathcal{A}$ modellieren. Die dazugehörige Menge $\mathcal{S}_R(A) = \{S_{A,1}, S_{A,2}, \ldots, S_{A,\sigma(A)}\}$ beschreibt eine Teilmenge von $\mathcal{S}_R$, deren Elemente der Relation $S_{A,x} \in \mathcal{S}_R$ und $S_{A,x} \equiv A$ für alle $S_{A,x} \in \mathcal{S}_R(A)$ unterliegen. Für jedes Semun $S_{A,x} \in \mathcal{S}_R(A)$ wird nach Gleichung 6.17 eine semunbezogene Basisbewertung ermittelt und in Abhängigkeit von der relativen zeitlichen Position innerhalb des Spottingintervalls mit $\theta_S(S_{A,x})$ aus Gleichung 6.19 zeitlich gewichtet. Die Summe der Produkte resultiert in einer *aktionsspezifischen Grundbewertung*, die mit $p_1(A)$ bezeichnet wird.

$$p_1(A) = \frac{1}{\sigma(\mathcal{I})} \sum_{i=1}^{\sigma(A)} \theta_S(S_{A,i})\, p_m(S_{A,i}) \quad \forall\, A \in \mathcal{A} \tag{6.20}$$

Die zwischenmenschliche Kommunikation ist bewusst mit redundanten Anteilen versehen. Speziell diese Anteile machen den Informationsaustausch zwischen den einzelnen Partnern sicher und robust gegenüber veränderlichen Randbedingungen. Aus den Ergebnissen der Benutzerstudien in Kapitel 4 kann entnommen werden, dass der Mensch zum Teil auch bei der Kommunikation mit einem technischen System redundant interagiert. Gleichung 6.20 bietet eine einfache Möglichkeit, diese Beobachtung bei der quantitativen Beurteilung der einzelnen Integrationslösungen zu simulieren. Indem die Bewertungen der jeweils redundanten Semune addiert werden, ergibt sich für jedes $A \in \mathcal{A}$ ein Score, der umso höher ausfällt, je mehr redundante Anteile in dem Spottingintervall enthalten sind. Die aktionsspezifische Grundbewertung wird daher auch als *redundante Grundwertung* bezeichnet.

Eine weitere Erfahrung aus den Benutzerstudien ist, dass komplementäre Informationsanteile die Sicherheit eines Integrationsergebnisses erhöhen und rivalisierende Eingaben die Sicherheit entsprechend verringern. Um diese beiden Effekte in den Bewertungsprozess integrieren zu können, wird zunächst für die jeweilige Kategorie der Informationszusammensetzung ein spezifischer Score ermittelt. Dabei wird auf die bereits in Gleichung 4.18 verwendete Terminologie zurückgegriffen.

Sei $\mathcal{S}_R(A_K) = \{S_{K,1}, S_{K,2}, \ldots, S_{K,\sigma(A_C)}\}$ eine Teilmenge aus $\mathcal{S}_R$ mit insgesamt $\sigma(A_K)$ Elementen. Die Menge $\mathcal{S}_R(A_K)$ enthält ausschließlich Semune, die komplementär zu der Referenzaktion A sind. Entsprechend sei $\mathcal{S}_R(A_R) = \{S_{R,1}, S_{R,2}, \ldots, S_{R,\sigma(A_R)}\}$ eine Teilmenge aus $\mathcal{S}_R$ der Kardinalität $\sigma(A_R)$ mit Semunen, die in einer rivalisierenden Relation zu der Aktion A stehen.

Die Umfänge der einzelnen Semunkategorien unterliegen der Beziehung $\sigma(A)+\sigma(A_K)+\sigma(A_R) \leq \sigma(\mathcal{I})$. Unter Anwendung des Summantionsverfahrens aus Gleichung 6.20 ergeben sich auf Basis der jeweiligen Mengen $\mathcal{S}_R(A_K)$ und $\mathcal{S}_R(A_R)$ in Abhängigkeit von den einzelnen Aktionen $A \in \mathcal{A}$ eine *komplementäre Grundbewertung* $p_2(A)$ und eine *rivalisierende Grundbewertung* $p_3(A)$.

$$p_2(A) = \frac{1}{\sigma(\mathcal{I})} \sum_{j=1}^{\sigma(A_K)} \theta_S(S_{K,j})\, p_m(S_{K,j}) \quad \forall\, A \in \mathcal{A} \tag{6.21}$$

$$p_3(A) = \frac{1}{\sigma(\mathcal{I})} \sum_{k=1}^{\sigma(A_R)} \theta_S(S_{R,k})\, p_m(S_{R,k}) \quad \forall\, A \in \mathcal{A} \tag{6.22}$$

Der erste Schritt auf dem Weg zu einer übergeordneten Semunbeurteilung besteht darin, die Bewertungen für die redundanten und die komplementären Informationsanteile zu addieren und davon die Bewertung der rivalisierenden Informationsanteile zu subtrahieren. Die resultierende Bewertung wird als *multimodale Basisbewertung* bezeichnet und mit $p_{mmb}(A)$ abgekürzt. Zusätzlich werden drei kategoriespezifische Gewichtungsfaktoren $\beta_1, \beta_2, \beta_3 \in \mathbb{R}^+$ eingeführt, mit denen die Bedeutung der einzelnen Informationsanteile individuell an die jeweilige Applikationsdomäne angepasst werden kann. Die Gewichtungsfaktoren werden zum Programmstart aus einer zentralen Datei eingelesen, können aber auch während eines Integrationslaufs als systeminterne Kontextparameter verändert werden. Standardmäßig wirken sich die drei Faktoren mit einer Vorbelegung von $\beta_1 = \beta_2 = \beta_3 = 1$ nicht auf das Ergebnis aus.

$$p_{mmb}(A) = \beta_1 p_1(A) \,+\, \beta_2 p_2(A) \,-\, \beta_3 p_3(A) \quad \forall\, A \in \mathcal{A} \tag{6.23}$$

Überprüfung auf Vollständigkeit und Kompatibilität

Anschließend wird für jede Aktion $A \in \mathcal{A}$ überprüft, ob die jeweilige Aktion überhaupt aus den aktuell in den Erkennerchromosomen vorhandenen Semunen gebildet werden kann. Manche Aktionen bestehen lediglich aus einem einzelnen Semun, andere wiederum aus einer verschachtelten Struktur von mehreren Informationsanteilen. Zu diesem Zweck durchlaufen die Semune $S \in \mathcal{S}_R$ im Hinblick auf die einzelnen Wörter der formalen Applikations-Sprache $w \in \mathcal{L}_A$ einen zielgerichteten Generierungsprozess. Dabei werden die Regeln der dazugehörigen kontextfreien Grammatik CFG_A solange schrittweise expandiert, bis entweder die Aktion A abgeleitet werden kann oder keine weiteren Regelanwendungen mehr möglich sind.

Stellen die Semune in $\mathcal{I}$ ein syntaktisch korrektes Wort A der Zielgrammatik dar, so bleibt die Bewertung in ihrer ursprünglichen Form erhalten. Andernfalls wird $p_{mmb}(A)$ mit der Zeit skaliert. Dazu wird in Abhängigkeit von dem Generationszähler q ein weiterer zeitlicher Gewichtungsfaktor $\theta_G(q)$ mit $0 < \theta_G(q) \leq 1$ eingeführt. Auf diese Weise kann zu einem frühen Zeitpunkt der Integration auch eine noch unvollständige Aktion eine hohe Bewertung erzielen. Demgegenüber werden zu einem späteren Zeitpunkt die Bewertungen von unvollständigen Aktionen durch die Multiplikation mit $\theta_G(q)$ deutlich verringert. Kann A auf Basis der Semune in $\mathcal{I}$ abgeleitet werden, so gilt $\theta_G(q) = 1$.

In die Bewertung der potenziellen Systemaktionen fließt zusätzlich noch der kontextuelle Bezug zu den vorherigen Integrationsergebnissen ein. Die entsprechenden Informationen dazu sind in dem Speicherchromosom $\mathcal{G}_M$ abgelegt. Sei $A_{pre} \in \mathcal{A}$ das Resultat der letzten Fusion, die real in eine Systemaktion umgesetzt wurde. Für die Bestimmung der funktionalen Relation

zwischen A_{pre} und der aktuellen Aktion A wird das Kriterium der *Kompatibilität* eingeführt. Dieses Kriterium ist genau dann erfüllt, wenn sich A und A_{pre} entweder strukturell zu einem weiteren Wort aus $\mathcal{L}_A$ ergänzen oder A in der Liste der möglichen Folgeaktionen von A_{pre} eingetragen ist. Zur formalen Darstellung der Kompatibilität zwischen zwei Aktionen A_a und A_b wird die Symbolik $A_a \bowtie A_b$ verwendet.

Sei $\theta_{pre}(A_{pre})$ die zeitlich gewichtete Gesamtbewertung der vorherigen Systemaktion. Für jedes $A \in \mathcal{A}$ wird jetzt überprüft, ob A und A_{pre} kompatibel zueinander sind. Ist dies der Fall, dann wird $\theta_{pre}(A_{pre})$ zur Unterstützung des potenziellen Integrationsergebnisses positiv berücksichtigt. Besteht zwischen den beiden Aktionen jedoch keine funktionale Kompatibilität, so wirkt sich $\theta_{pre}(A_{pre})$ negativ auf die aktuelle Bewertung aus.

$$p_{pre}(A) := \begin{cases} \theta_{pre}(A_{pre}) & \text{falls} \quad A_{pre} \bowtie A \\ -\theta_{pre}(A_{pre}) & \text{andernfalls} \end{cases} \tag{6.24}$$

Zusammengesetzte Gesamtbewertung

Auf Basis der einzelnen Komponenten, die in den Gleichungen 6.17 bis 6.24 und den dazugehörigen Textabschnitten beschrieben wurden, lässt sich eine zusammenfassende Beurteilung für jede potenzielle Systemaktion $A \in \mathcal{A}$ ableiten. Dieser finale Wert wird als *aktionsspezifische multimodale Gesamtbewertung* bezeichnet und mit $p_{mm}(A)$ referenziert. In Abhängigkeit von der verwendeten Integrationsstufe können einzelne Elemente von der Berechnung ausgeschlossen bzw. mit Standardwerten belegt sein.

Zunächst wird für jedes Semun $S \in \mathcal{S}_R$ eine Basisbewertung ermittelt, in der das Konfidenzmaß der entsprechenden Erkennung und die allgemeine Erkennungssicherheit des dazugehörigen Systemmoduls eingehen. Diese Basisbewertungen fließen in die übergeordnete Bewertung der funktionalen Informationszusammensetzung ein. Die redundante und die komplementäre Grundbewertung werden addiert und davon die rivalisierende Grundwertung subtrahiert. Das Ergebnis wird auf syntaktische Vollständigkeit überprüft und in Bezug auf die aktuelle Lösungsiteration q mit $\theta_G(q)$ zeitlich skaliert. Abschließend wird in Abhängigkeit von der Kompatibilitätsrelation zwischen der untersuchten Aktion A und dem vorherigen Integrationsergebnis A_{pre} die zeitlich gewichtete Bewertung von A_{pre} entweder addiert oder subtrahiert. Unter Anwendung der eingeführten Abkürzungen ergibt sich die Gesamtbewertung $p_{mm}(A)$ für das aktuell untersuchte Individuum $\mathcal{I}$ wie folgt:

$$p_{mm}(\mathcal{I}, A) = p_{mmb}(A)\ \theta_G(q) + p_{pre}(A) \tag{6.25}$$

Sei $\mathcal{I}_i$ mit $i \in \{1..\iota\}$ ein beliebiges Individuum aus einer Population $\mathcal{P}(q)$ mit dem Generationsindex q. Weiterhin sei $\mathcal{A} = \{A_1, A_2, \ldots, A_{\alpha(A)}\}$ eine Modellierung der möglichen Systemaktionen. Die statistisch gewichtete Summe der einzelnen Chromosomanteile in Gleichung 6.25 repräsentiert im erweiterten Sinne ein Maß für die Wahrscheinlichkeit, dass ein Individuum $\mathcal{I}_i \in \mathcal{P}(q)$ eine spezifische Aktion $A \in \mathcal{A}$ modelliert. Mit $\omega(\mathcal{I}_i, A) := p_{mm}(\mathcal{I}_i, A)$ stellt diese Bewertung eine problemangepasste Berechnung der individuellen Fitness dar. Als lokales Ergebnis für die Integration wird diejenige Aktion ausgewählt, die auf Basis der Parameterkonfiguration in $\mathcal{I}_i$ die beste Beurteilung erhält. Nach der in Abschnitt 5.4 eingeführten Terminologie bezeichnet man dieses Ergebnis als eine Aktionshypothese H.

$$H(\mathcal{I}_i) = \text{argmax}_A\ \omega(\mathcal{I}_i, A) \quad \forall\ A \in \mathcal{A},\ \forall\ i \in \{1..\iota\} \tag{6.26}$$

Aus den individuellen Bewertungen wird, wie in Gleichung 6.15 beschrieben, eine aktionsspezifische Populationsfitness $\omega(\mathcal{P})$ berechnet. Das Ergebnis eines Integrationsdurchlaufs mit dem Zählindex q ist schließlich genau die Aktion $A \in \mathcal{A}$ mit der besten akkumulierten Bewertung $\omega(\mathcal{P}(q), A)$. Diese Aktion stellt im Hinblick auf die Abbildungsfunktion in Gleichung 5.14 eine Aktionshypothese auf der Symbolebene dar.

$$H_S(\mathcal{P}(q)) = \text{argmax}_A\, \omega(\mathcal{P}(q), A) \quad \forall\, A \in \mathcal{A},\ \forall\, i \in \{1..\iota\} \tag{6.27}$$

Zur besseren Vergleichbarkeit der individuellen Bewertungen für unterschiedliche Integrationsdurchläufe q_x und q_y mit $q_x \neq q_y$ werden die lokalen Bewertungen analog zu Gleichung 6.16 bezogen auf die Populationsfitness normiert und zentral in einer Tabelle abgespeichert. Die standardisierten Beurteilungen bilden die Basis für die im weiteren Verlauf wichtigen Untersuchungen zum Konvergenzverhalten der einzelnen Integrationsalternativen.

6.6 Iterative genetische Datenfusion

Das wesentliche Element eines evolutionären Integrationsansatzes ist die iterative Erzeugung und Variation von mehreren unterschiedlichen Aktionshypothesen. Ein einzelner Integrationsdurchlauf orientiert sich unmittelbar an den in Abbildung 6.1 dargestellten Standardprozessen in einem genetischen Algorithmus. Während die Bewertung einzelner Lösungen bereits im letzten Abschnitt ausführlich erläutert wurde, stehen im Folgenden der Austausch von Teilinformationen zwischen ausgewählten Individuen einer Generation und die Überprüfung der Lösungskonvergenz im Zentrum der Beschreibung.

Die iterative genetische Datenfusion setzt an, wenn ein Spotting-Prozess vollständig abgeschlossen ist. Durch diese Vorverarbeitung werden relevante Ereignisse zu einem Integrationscluster $\mathcal{C}_S$ zusammengefasst. Auf Basis der einzelnen Elemente in $\mathcal{C}_S$ wird eine initiale Population $\mathcal{P}(q) = \{\mathcal{I}_{q,1}, \mathcal{I}_{q,2}, \ldots, \mathcal{I}_{q,\iota}\}$ mit einer festen Anzahl von ι Individuen erzeugt. Diese Population wird als *Ursprungspopulation* bezeichnet und durch den Zähler $q = 0$ identifiziert.

Zu Beginn des Fusionsprozesses sind sämtliche Individuen vollkommen identisch. Der Inhalt des Speicherchromosoms $\mathcal{G}_M$ wird aus dem vorherigen Integrationsergebnis übernommen. Die Erkenner- und Kontextchromosome $\mathcal{G}_R$, $\mathcal{G}_K$ repräsentieren die vorhandenen Eingabedaten. Sie stammen größtenteils direkt aus den jeweiligen Erkennungsprozessen und werden durch die Integrator-Nachrichten übertragen. Eine zusätzliche Informationsquelle sind die domänenspezifischen Lookup-Tabellen. Das Aktionschromosom $\mathcal{G}_A$ bleibt zunächst leer.

6.6.1 Selektion

Auf Basis der quantitativen Bewertung werden einzelne Individuen einer Population nach unterschiedlichen Verfahren für den folgenden Rekombinationsprozess ausgewählt. Sei $\mathcal{P} := \mathcal{P}(q)$ eine beliebige Generation mit $q \geq 0$. Die ausgewählten Elemente werden in eine temporäre *Paarungspopulation* $\mathcal{P}_P$ kopiert. Diese Population besteht aus π Individuen $\mathcal{P}_P = \{\mathcal{I}_1^P, \mathcal{I}_1^P, \ldots, \mathcal{I}_\pi^P\}$ mit $\mathcal{I}_j^P \in \mathcal{P}$ für alle $1 \leq j \leq \pi$. Man bezeichnet diesen Auswahlprozess im Hinblick auf den biologischen Hintergrund auch als *Paarungsselektion*.

Anders als in der Ausgangspopulation $\mathcal{P}$, deren Elemente nominell unterscheidbar sind, können in $\mathcal{P}_P$ auch Individuen $\mathcal{I}_a^P, \mathcal{I}_b^P$ mit $a \neq b$ enthalten sein, die beide von einem einzigen Individuum $\mathcal{I}_x$ stammen. Die mit dem Paarungsprozess assoziierte Auswahlfunktion Π ist zwar surjektiv, aber nicht injektiv.

$$\Pi : \mathcal{P}(q) \rightarrow \mathcal{P}_P(q) \quad \text{mit} \quad q \in \mathbb{N}^+ \tag{6.28}$$

Individuen mit einer guten Fitness haben tendenziell eine bessere Chance, in den Paarungsprozess mit einbezogen zu werden als schlecht bewertete Aktionshypothesen. Die Anzahl π der Elemente in $\mathcal{P}_P$ ist ein wichtiges strukturierendes genetisches Element, das speziell an die konkrete Anwendung angepasst werden muss. In vielen praktischen Szenarien wird zunächst von einer geraden Zahl im Bereich von $2 \leq \pi \leq 2\iota$ ausgegangen.

Die einfachste Möglichkeit, ein Individuum aus $\mathcal{P}$ auszuwählen, besteht darin, die aktionsbezogene normierte Fitness als Wahrscheinlichkeit für die Aufnahme in die Paarungspopulation zu verwenden. Man bezeichnet dieses Selektionsverfahren auch als *Standard-Selektion* (Π_S). Problematisch an diesem Vorgehen ist die Tatsache, dass im Verlauf der Evolution die Unterschiede in den Genen durch die präferierte Selektion qualitativ hochwertiger Lösungen sehr schnell extrem klein werden. Die teilweise zufällige Initialisierung der Ausgangspopulation kann dazu führen, dass einige Individuen über ungewöhnlich hohe Bewertungen verfügen. Man bezeichnet diese speziellen Lösungen als *superfitte* Individuen. Die Existenz eines solchen speziellen Individuums kann dazu führen, dass nach mehreren Iterationen die Gesamtpopulation maßgeblich durch dieses eine superfitte Individuum dominiert wird. Der Algorithmus konvergiert daraufhin unter Umständen lediglich auf ein lokales Maximum [52].

Um dieses Problem zu umgehen sind in der Literatur verschiedene alternative Selektionsverfahren vorgeschlagen und diskutiert worden [21]. Bei der *Rang-Selektion* (Π_R) werden die Individuen der Ausgangspopulation gemäß der berechneten Fitness absteigend sortiert. Jedem Platz in dieser Anordnung wird eine vorgegebene Wahrscheinlichkeit zugeordnet, mit der das entsprechende Individuum in die Paarungspopulation gewählt wird. Auf diese Weise kann zwar die Existenz von superfitten Individuen vermieden werden, auf der anderen Seite gehen aber auch Informationen über die relativen Unterschiede in den Bewertungen verloren.

Im praktischen Einsatz wird normalerweise die *Turnier-Selektion* (Π_T) verwendet [53]. Bei diesem Verfahren wird eine feste Anzahl von τ zufällig ausgewählten Individuen in eine spezielle Turnierpopulation $\mathcal{P}_T$ kopiert. Das Individuum mit der besten Fitness innerhalb dieser Menge wird dann in die Paarungspopulation $\mathcal{P}_P$ übertragen. Dieser Prozess wird insgesamt π-mal wiederholt. Die Turnierselektion ist zwar rechentechnisch aufwendiger, umgeht aber die Nachteile der beiden anderen Verfahren. Der Selektionsdruck, mit dem vorrangig gut bewertete Lösungen ausgewählt werden, ist direkt proportional zu τ.

6.6.2 Rekombination

Im Rahmen des Rekombinationsprozesses wird das vorhandene genetische Material variiert. Dazu werden auf die Menge der selektierten Lösungsalternativen zwei genetische Operatoren angewendet. Der erste Operator kombiniert Teilkomponenten einzelner Individuen und erzeugt auf diese Weise vollkommen neue Lösungen, deren strukturelle Zusammensetzung sich unter Umständen signifikant von den ursprünglichen Individuen unterscheiden kann. Dieser interne Prozess wird als *Crossover* bezeichnet und stellt die maßgebliche Methode zur schnellen Exploration des Suchraums dar. Der zweite Operator verändert zufällig bestimmte Gene einzelner

Individuen innerhalb streng vorgegebener Toleranzgrenzen. In direkter Analogie zu der Natur bezeichnet man diesen externen Variationsprozess daher auch als *Mutation*. Auf beide Prozesse wird im Folgenden näher eingegangen.

Crossover

Die Paarungspopulation $\mathcal{P}_P$ bildet die Ausgangsbasis für die Rekombination. In der folgenden Darstellung werden lediglich Kombinationen von genau zwei Individuen betrachtet. Zur Identifikation der einzelnen Rekombinationsprozesse wird ein diskreter Zähler l mit $l = 0..(\frac{\pi}{2} - 1)$ eingeführt, wobei $\mathcal{P}_P(0)$ mit der initialen Paarungspopulation $\mathcal{P}_P$ übereinstimmt. Für eine spezielle Paarung mit dem Index l werden ohne Beschränkung der Allgemeinheit jeweils zwei Individuen $\mathcal{P}'_P(l) = \{\mathcal{I}_a^P, \mathcal{I}_b^P\}$ mit $a \neq b$ und $a, b \in \{1..\pi\}$ zufällig aus der Menge $\mathcal{P}_P(l)$ ausgewählt, rekombiniert und anschließend aus der temporären Population entfernt. Für die Bildung der reduzierten Paarungspopulation $\mathcal{P}''_P(l)$ gilt:

$$\mathcal{P}''_P(l) = \mathcal{P}_P(l) \setminus \mathcal{P}'_P(l) \quad \forall\, l \in \{0..(\frac{\pi}{2} - 1)\} \tag{6.29}$$

Die um zwei Individuen reduzierte Population $\mathcal{P}''_P(l)$ bildet die Ausgangsmenge für den nächsten Rekombinationsschritt und lässt sich rekursiv durch den Ausdruck $\mathcal{P}_P(l+1) = \mathcal{P}''_P(l)$ angeben. Der gesamte Vorgang wird solange iteriert, bis die virtuelle Paarungspopulation mit $\mathcal{P}''_P(l) = \emptyset = \mathcal{P}_P(\frac{\pi}{2} - 1)$ vollkommen leer ist. Formal kann der Crossover-Prozess als surjektive Abbildung Ψ_C dargestellt werden, bei der auf Basis von zwei existierenden Individuen $\mathcal{I}_a$ und $\mathcal{I}_b$ zwei neue Lösungsalternativen $\mathcal{I}_c$ und $\mathcal{I}_d$ entstehen. Zur speziellen Kennzeichnung der Verbindung wird das Operatorsymbol $\diamond$ eingeführt. Im Zusammenhang mit dem Rekombinationsprozess werden die beiden Individuen $\mathcal{I}_a$ und $\mathcal{I}_b$ als *Stammindividuen* oder *Eltern* bezeichnet. Entsprechend nennt man die beiden neuen Lösungen $\mathcal{I}_c$ und $\mathcal{I}_d$ *Kinder* oder *Nachkommen*.

$$\Psi_C : \mathcal{I}_a \diamond \mathcal{I}_b \rightarrow \{\mathcal{I}_c, \mathcal{I}_d\} \quad \text{mit} \quad \mathcal{I}_a, \mathcal{I}_b \in \mathcal{P}_P \tag{6.30}$$

Beim Crossover wird die genetische Information der Eltern-Individuen aufgebrochen und wechselseitig auf die beiden Nachkommen verteilt. Man bezeichnet die entsprechenden Positionen in der genetischen Struktur als *Kreuzungspunkte*. Die Anzahl der Kreuzungspunkte κ mit $1 \leq \kappa < \mu$ ist ein weiteres strukturierendes Element in jedem genetischen Algorithmus, das an die aktuelle Problemdomäne angepasst werden muss. Insgesamt werden in Abhängigkeit von κ jeweils $2 \leq (\kappa + 1) \leq \mu$ Elemente zwischen den beiden Stammindividuen ausgetauscht.

In der ursprünglichen Form der genetischen Algorithmen wird mit $\kappa = 1$ von einem einzelnen Kreuzungspunkt ausgegangen. Speziell bei komplexen Optimierungsproblemen haben sich jedoch auch Ansätze mit mehreren Kreuzungspunkten $\kappa \geq 2$ bewährt [44]. Die Positionen der Kreuzungspunkte innerhalb der genetischen Struktur eines Individuums $\mathcal{I}$ werden durch die Menge $\mathcal{R}_K(\mathcal{I})$ modelliert. In dieser Menge sind die einzelnen Elemente $k \in \mathcal{R}_K(\mathcal{I})$ in Bezug auf die referenzierten Kreuzungspositionen streng monoton ansteigend geordnet.

$$\begin{aligned} \mathcal{R}_K(\mathcal{I}) &= \{k_1, k_2, ..., k_\kappa\} \\ &\text{mit} \quad k_i < k_j \quad \forall \;\; k_i, k_j \in \{1..\mu\}, \;\; i < j, \;\; i, j \in \{1..\kappa\} \end{aligned} \tag{6.31}$$

Bei mehreren Kreuzungspunkten $\kappa \geq 2$ werden zwischen den festgelegten Positionen jeweils zufällig die genetischen Informationen eines Stammindividuums auf eines der beiden Kinder

übertragen. Der andere Nachkomme ergibt sich analog aus dem Komplement der initial gewählten Kreuzungs-Struktur. Sei $\mathcal{I}_a = \{g_{a,1}, g_{a,2}, ..., g_{a,\mu}\}$ die genetische Zusammensetzung von $\mathcal{I}_a$, $\mathcal{I}_b = \{g_{b,1}, g_{b,2}, ..., g_{b,\mu}\}$ entsprechend die Struktur von $\mathcal{I}_b$ und k mit $1 \leq k < \mu$ ein ausgewiesener Kreuzungspunkt, dann ergeben sich die Nachkommen $\mathcal{I}_c$ und $\mathcal{I}_d$ wie folgt:

$$\mathcal{I}_c = \{g_{a,1}, g_{a,2}, ..., g_{a,k}, g_{b,k+1}, g_{b,k+2}, ..., g_{b,\mu}\} \tag{6.32}$$

$$\mathcal{I}_d = \{g_{b,1}, g_{b,2}, ..., g_{b,k}, g_{a,k+1}, g_{a,k+2}, ..., g_{a,\mu}\} \tag{6.33}$$

Die Nachkommen aus den einzelnen Crossover-Prozessen bilden eine temporäre Population $\mathcal{P}_C$ von neuen Lösungshypothesen, deren Größe mit der Kardinalität π der temporären Paarungspopulation $\mathcal{P}_P$ übereinstimmt. Zusammen mit den ι Individuen der Ausgangspopulation ergeben die neu erzeugten Individuen eine Gesamtpopulation $\mathcal{P}_G$. Die Größe dieser Menge resultiert aus der Summe von ι und π und wird mit o angegeben.

$$\mathcal{P}_G = \mathcal{P} \cup \mathcal{P}_C \quad \text{und} \quad o = \iota + \pi \tag{6.34}$$

Mutation

Durch den zweiten genetischen Operator kann einer ungewollten, zu schnellen Konvergenz der Population auf ein lokales Minimum wirksam entgegengewirkt werden. Der Mutationsoperator bezieht prinzipiell jeden Bereich im Lösungsraum mit einer gewissen Wahrscheinlicht in den Lösungsprozess mit ein und unterstreicht damit den Charakter eines global ausgelegten Optimierungsverfahrens. Im Gegensatz zum Crossover wird die Mutation im Allgemeinen jedoch nur auf wenige Individuen angewendet.

Quantitativ wird der Mutationsprozess durch zwei zentrale Größen beschrieben. Der erste Parameter ϕ mit $0 \leq \phi \leq o$ gibt die Anzahl der zu mutierenden Individuen an. Aus der Gesamtpopulation $\mathcal{P}_G$ werden zufällig ϕ Individuen ausgewählt und in eine temporäre *Mutationspopulation* $\mathcal{P}_M = \{\mathcal{I}_1^M, \mathcal{I}_2^M, ..., \mathcal{I}_\phi^M\}$ kopiert. Der zweite Parameter ξ bewegt sich in einem Wertebereich zwischen 1 und μ und spezifiziert die Anzahl der Gene, die pro Individuum $\mathcal{I} \in \mathcal{P}_M$ einer Variation unterzogen werden. Analog zu Gleichung 6.31 wird eine Menge $\mathcal{R}_M(\mathcal{I}) = \{m_1, m_2, \ldots, m_\xi\}$ eingeführt, die bezogen auf ein spezifisches Individuum $\mathcal{I} \in \mathcal{P}_M$ die einzelnen Mutationspositionen auflistet.

$$\begin{aligned} \mathcal{R}_M(\mathcal{I}) &= \{m_1, m_2, ..., m_\xi\} \\ &\text{mit} \quad m_i < m_j \quad \forall \; m_i, m_j \in \{1..\mu\}, \;\; i < j, \;\; i, j \in \{1..\xi\} \end{aligned} \tag{6.35}$$

Im Gegensatz zu der Natur ist in einem genetischen Algorithmus der Wert ϕ für alle Generationen identisch. Ebenso bleibt ξ für sämtliche Individuen sowohl einer Generation als auch über verschiedene Generationen hinweg konstant. Die Indizes der mutierten Gene $\mathcal{R}_M(\mathcal{I}_a)$, $\mathcal{R}_M(\mathcal{I}_b)$ sind für unterschiedliche Individuen $\mathcal{I}_a \neq \mathcal{I}_b$ jedoch zumeist verschieden.

Formal kann die Mutation als Funktion Ψ_M beschrieben werden, bei der für jede Position $m \in \mathcal{R}_M(\mathcal{I}_x)$ in der genetischen Struktur eines ausgewählten Individuums $\mathcal{I}_x^M \in \mathcal{P}_M$ das entsprechende Gen $g_{x,m}$ auf ein modifiziertes Gen abgebildet wird. Bei einer binären Kodierung bewirkt die Mutation die Invertierung einzelner Bits. Im Rahmen der natürlichen Kodierung ist die Mutation wesentlich aufwendiger und benötigt extensives Kontextwissen über die Problemdomäne. Ein Gen entspricht jeweils einem Semun $S \in \mathcal{S}$.

$$\Psi_M : \mathcal{S} \rightarrow \mathcal{S} \tag{6.36}$$

Die Mutationsmöglichkeiten sind pro Semun in einer zentralen Lookup-Tabelle abgelegt. Auf diese Weise kann sichergestellt werden, dass im Mutationsprozess ausschließlich Semune entstehen, die ebenfalls zu der kontextfreien Sprache $\mathcal{L}_I$ gehören. Durch die Beschränkung der Variationsmöglichkeiten wird eine nachträgliche Anpassungsfunktion überflüssig gemacht. Als Ergebnis entsteht eine weitere temporäre Population $\mathcal{P}_{M'} = \{\mathcal{I}_1^{M'}, \mathcal{I}_2^{M'}, ..., \mathcal{I}_\phi^{M'}\}$ von Integrationslösungen mit leicht veränderten Eigenschaften, deren Größe mit der Kardinalität der Mutationspopulation $\phi = |\mathcal{P}_M| = |\mathcal{P}_{M'}|$ übereinstimmt.

Obwohl die beiden genetischen Parameter ϕ und ξ explizit an die jeweilige Problemdomäne angepasst werden müssen, bewegen sich die Werte in vielen praktischen Anwendungen mit $\phi \ll o$ und $\xi \ll \mu$ in einem vergleichbar engen Bereich. Allgemein kann nicht davon ausgegangen werden, dass in jedem Iterationsdurchlauf immer mindestens ein Individuum der Gesamtpopulation mutiert wird. Unter dieser Voraussetzung müsste der Parameter ϕ daher auch Werte aus den reellen Zahlen annehmen können.

Für Werte von ϕ zwischen 0 und 1 wird eine *Mutationsrate* $\varphi \in \mathbb{R}^+$ mit $0 \leq \varphi \leq 1$ eingeführt, die für jedes Individuum der Gesamtpopulation $\mathcal{I} \in \mathcal{P}_G$ die Wahrscheinlichkeit angibt, dass die Gene von $\mathcal{I}$ einer Mutation unterzogen werden. Mutierte Lösungen werden dann in die Menge $\mathcal{P}_{M'}$ übertragen. Anders als bei der ganzzahligen Variante mit $\phi \in \mathbb{N}$ kann in Abhängigkeit von φ die Anzahl der Elemente in $\mathcal{P}_{M'}$ auch zwischen den einzelnen Integrationsläufen variieren.

6.6.3 Generations-Update

Durch die Variation des genetischen Materials im Rekombinationsprozess entsteht eine Menge von neuen Lösungshypothesen. Aus der Vereinigungsmenge der ursprünglichen Individuen einer Population $\mathcal{P}$, der im Crossover neu erzeugten Individuen ($\mathcal{P}_C$) und der mutierten Individuen $\mathcal{P}'_M$) werden einzelne Lösungen selektiert und zu einer neuen Generation zusammengefügt. Diesen zweiten Auswahlprozess bezeichnet man als *Generationsselektion* Π_G. Mit $q \geq 0$ als Generationsindex lässt sich Π_G formal als surjektive Abbildung darstellen.

$$\Pi_G : \mathcal{P}(q) \cup \mathcal{P}_C(q) \cup \mathcal{P}'_M(q) \rightarrow \mathcal{P}(q+1) \tag{6.37}$$

Die Größe einer ausgewiesenen Generation $\mathcal{P}(q)$ wird mit $\iota(q)$ bezeichnet. In genetischen Algorithmen stellt diese Größe standardmäßig für alle $q \in \mathbb{N}$ einen konstanten Wert dar. Eine abgeleitete Population $\mathcal{P}(q+1)$ beinhaltet demnach mit $\iota(q) = \iota(q+1)$ die gleiche Anzahl an Integrationshypothesen wie die Ausgangspopulation $\mathcal{P}(q)$. Dieses artifizielle Verhalten steht im Gegensatz zu der natürlichen Evolution, bei der mit $\iota(q) \leq \iota(q+1)$ normalerweise ein Reproduktionsüberschuss beobachtet werden kann.

Zur Auswahl der einzelnen Individuen für die nächste Iteration sind unterschiedliche Methoden untersucht worden [24]. Die einfachste Möglichkeit besteht darin, die alten Individuen vollständig durch die Nachkommen zu ersetzen. Im Rahmen einer gemischten Generation aus alten und neuen Lösungen können aus $\mathcal{P}(q)$ und $\mathcal{P}(q+1)$ entweder zufällig oder nach der individuellen Fitness gewichtet einzelne Mitglieder ausgewählt und in die nächste Generation übertragen werden. Die Wahl der speziellen Selektionstechnik hängt von der Struktur der Problemlösungsdomäne ab.

6.6.4 Konvergenzkriterien

Bei einer adäquaten Wahl der verschiedenen genetischen Parameter konvergiert die Gesamtpopulation nach mehreren Iterationen. Die strukturierenden Elemente müssen dazu hinsichtlich unterschiedlicher Kriterien in einem iterativen Anpassungsverfahren aufeinander abgestimmt werden. Da eine optimale Parameterkonfiguration zur Lösung des Ausgangsproblems im Vorfeld normalerweise nicht bekannt ist, wird ein Abbruch-Kriterium eingeführt, mit dem eine Aussage über die Qualität der Lösungspopulationen getroffen werden kann. Dieses Kriterium überprüft die Konvergenzeigenschaften einer Population. Im Kontext der natürlichen Evolution wird Konvergenz als das inhärente Streben der Individuen nach einer kontinuierlich wachsenden Gleichförmigkeit definiert [23].

Vor dem Hintergrund eines simulierten Evolutionsprozesses stehen im Wesentlichen zwei verschiedene Ansätze für die Bestimmung der Konvergenz einer Population $\mathcal{P}$ zur Verfügung. Die dazugehörigen Überprüfungsverfahren sind jeweils durch einen spezifischen *Konvergenztyp* gekennzeichnet, der mit $\Xi \in \{\Xi_1, \Xi_2, \Xi_3\}$ bezeichnet wird. Nach dem ersten Ansatz ($\Xi := \Xi_1$) wird eine Lösung genau dann akzeptiert, wenn ein vordefinierter Grenzwert ω_Ξ für die Populationsfitness $\omega(\mathcal{P})$ erreicht ist. Da die absolute Bewertung der einzelnen Lösungen sowohl von der Anzahl als auch von dem speziellen Typ der angeschlossenen Erkennermodule abhängt, kann $\omega(\mathcal{P})$ zwischen einzelnen Integrationsläufen beträchtlich schwanken.

Die beiden anderen Ansätze sind für das Problem der multimodalen Integration weitaus besser geeignet. Für die Überprüfung der Konvergenz werden die Bewertungen der einzelnen Individuen zueinander in Relation gesetzt. Als Vergleichsmaßstab wird beim zweiten Konvergenztyp $\Xi := \Xi_2$ die durchschnittliche individuelle Fitness $\bar{\omega}(\mathcal{P}) := \frac{\omega(\mathcal{P})}{\iota}$ herangezogen. Liegen die Bewertungen von einem vorgegebenen Prozentsatz ψ der Individuen über $\bar{\omega}(\mathcal{P})$, dann ist das Konvergenzkriterium erfüllt. Als Integrationsergebnis wird die Aktionshypothese des am besten bewerteten Individuums ausgewählt. Der dritte Ansatz $\Xi := \Xi_3$ ist eine leichte Variation dieses Verfahrens. Anstatt der durchschnittlichen Bewertung $\bar{\omega}(\mathcal{P})$ wird die lokal beste Bewertung $\omega_{best} := MAX\{\omega(I_1), \omega(I_2), \ldots, \omega(I_\iota))\}$ als Vergleichsmaßstab benutzt. Eine Population gilt genau dann als konvergiert, wenn ein vorgegebener Prozentsatz ψ der Individuen die Gene des lokal besten Individuums innerhalb einer Toleranzgrenze von einem Prozent teilt.

Die Überprüfung der Akzeptanzschwelle ψ entspricht der Einführung einer *Konvergenzrate*, die den populationsbezogenen Mindestanteil einer spezifischen Bewertung angibt. Analog zu den anderen genetischen Parametern hängt die Konfiguration von ψ von der aktuellen Problemdomäne ab. Um eine zufällige Konvergenz in den ersten Iterationsläufen zu vermeiden, wird zusätzlich eine untere Schranke für die Mindestanzahl an Iterationen eingeführt. Diese Schranke wird mit χ bezeichnet und stellt ein weiteres strukturierendes Element bei der Konzeption eines genetischen Algorithmus dar.

6.7 Implementierung

Der in den letzten Abschnitten formal beschriebene Integrationsansatz ist die zentrale Komponente innerhalb der multimodalen Basisarchitektur. Alle laufzeittechnisch relevanten Teile des Algorithmus sind vollständig in ANSI C++ implementiert. Für die Umsetzung eines graphischen Front-Ends zur einfachen Konfiguration der einzelnen Parameter wird zusätzlich die

Skriptsprache Tcl/Tk benutzt. Im weiteren Kontext wichtige Komponenten wie beispielsweise die Realisierung einer prototypischen Simulationsumgebung und dazugehörige automatische Test- und Auswertungsverfahren basieren zum einen auf JAVA und zum anderen ebenfalls auf Tcl/Tk-Code. Sämtliche Module verfügen über eine einheitliche Kommunikationsschnittstelle, durch die textbasierte Nachrichten, wie in den Abschnitten 5.5 und 5.6 beschrieben, mittels TCP/IP-Sockets ausgetauscht werden können.

Das regelbasierte Spotting und die genetische Datenfusion sind als eigenständige, programmtechnisch unabhängige Komponenten realisiert. Diese funktionale Trennung ermöglicht eine variable Konfiguration und Modifikation der einzelnen Integrationsstufen. In einer rein simulierten Bedienumgebung wird darüber hinaus die Evaluierung des Intentionsdekoders extrem vereinfacht. Extensive Testläufe mit unterschiedlichen Parameterbelegungen können jeweils pro Modul isoliert durchgeführt werden. Beide Integratorkomponenten sind vollständig in die LaunchPad-Umgebung eingebunden und können somit analog zu den anderen Systemmodulen extern kontrolliert werden.

Der genetische Fusionsalgorithmus ist primär unter dem Betriebssystem Linux entwickelt und getestet worden. Die Auswahl von C++ als Programmiersprache ermöglicht ein objektorientiertes Design des Intentionsdekoders, wobei die einzelnen Verarbeitungsstufen jeweils in sich geschlossene Module darstellen. Sowohl das Prinzip der Datenkapselung als auch das Konzept der Klassenhierarchien sind während des gesamten Entwicklungsprozesses konsequent angewendet worden. Der Programmcode kann einfach erweitert und auch auf andere Systemplattformen portiert werden.

Im Rahmen einer systemtechnischen Umsetzung werden neben problemspezifischen, abstrakten Datentypen auch verschiedene grundlegende Strukturen benötigt, die in ANSI C++ nicht standardmäßig enthalten sind. Speziell für die Repräsentation mathematischer Objekte wie typpolymorphen Vektoren und Matrizen und den damit verbundenen Berechnungsoperationen wird auf die Klassenbibliothek LEDA [100] zurückgegriffen. Einige numerische Routinen benutzen zudem Implementierungen aus dem Software-Paket Numerical Recipes [128].

An einigen Punkten stellt die Implementierung eine Vereinfachung der beschriebenen Verfahren dar. Im Hinblick auf die Darstellung der einzelnen Integrationsphasen in Abbildung 5.3 konzentriert sich die Umsetzung des genetischen Fusionsansatzes maßgeblich auf die unterste Integrationsebene Φ_S. Insbesondere die Auswertung von kontextuellen Informationen ist nur für wenige ausgewählte Szenarien umgesetzt. Der erzeugte Programmcode hat zudem eher einen prototypischen Charakter, mit dem vorrangig die prinzipielle Umsetzbarkeit des hybriden Integrationsverfahrens demonstriert werden soll. Eine Überarbeitung des Programmablaufs hinsichtlich der Optimierung von Laufzeiten und Speicherverbrauch ist bisher nur in Ansätzen erfolgt. So wurde beispielsweise auf eine Parallelverarbeitung vollständig verzichtet. Spezifische Details zu der konkreten Implementierung der Systemkomponenten können in Bezug auf das Spotting der Diplomarbeit von Renz [132] und in Bezug auf die genetische Fusion der Studienarbeit von Al-Hames [5] entnommen werden.

KAPITEL 7

Evaluierung des Gesamtsystems

In diesem Kapitel werden die Ergebnisse zur Evaluierung der entwickelten Systemkomponenten diskutiert. Um die jeweiligen Funktionen möglichst praxisorientiert testen und bewerten zu können, werden verschiedene Szenarien aus den Versuchsreihen in einer simulierten Bedienumgebung identisch reproduziert. Für diesen Zweck ist eigens eine spezielle Simulationsumgebung entwickelt worden, die am Anfang kurz vorgestellt wird. Die Bewertung des Gesamtsystems erfolgt getrennt nach den beiden Phasen des im letzten Kapitel vorgestellten, hybriden Integrationsansatzes. Zunächst wird auf das regelbasierte Spotting eingegangen. Vor dem Hintergrund realer und fiktiver Szenarien werden die verschiedenen Verfahren zur zeitlichen Segmentierung der Benutzereingaben auf ihre Praxistauglichkeit hin überprüft. Der zweite Bewertungsblock geht ausführlich auf die evolutionäre Datenfusion und Konfiguration der einzelnen genetischen Strukturparameter ein. Dabei werden sowohl das Konvergenzverhalten als auch die Laufzeit des Algorithmus näher betrachtet. Als Resultat ergibt sich eine Parameterkonfiguration, die sich besonders gut für den Einsatz im realen Intentionsdekoder eignet. Die Diskussion der Ergebnisse erfolgt maßgeblich anhand des DVA-Szenarios. Den Abschluss des Kapitels bilden einige zusätzliche Bemerkungen im Hinblick auf die extern wahrgenommene Erkennungsleistung des entwickelten Systems im AIA-Szenario.

7.1 Simulationsumgebung

Der korrekte Betrieb des Gesamtsystems mit einer Vielzahl von real angeschlossenen Erkenner-, Kontext- und Applikationsmodulen verursacht einen enormen Konfigurations- und Koordinationsaufwand. In Abschnitt 5.6 wurde mit dem LaunchPad bereits ein wichtiges Hilfsmittel vorgestellt, über das die einzelnen Komponenten zentral parametrisiert, selektiv gestartet bzw. gestoppt und zeitsynchron untereinander verbunden werden können. Um die benötigten Ressourcen im Rahmen der Systemevaluierung zu reduzieren ist als Alternative zu den realen Eingabemodulen darüber hinaus ein zentraler, rein software-basierter Simulator entwickelt und in die Basisarchitektur integriert worden.

Die Simulationsumgebung besteht im Wesentlichen aus einem frei parametrisierbaren Ereignisgenerator, der durch eine Reihe von automatischen Protokollierungs- und Auswertungskomponenten ergänzt wird. Abbildung 7.1 vermittelt einen Eindruck über das dazugehörige gra-

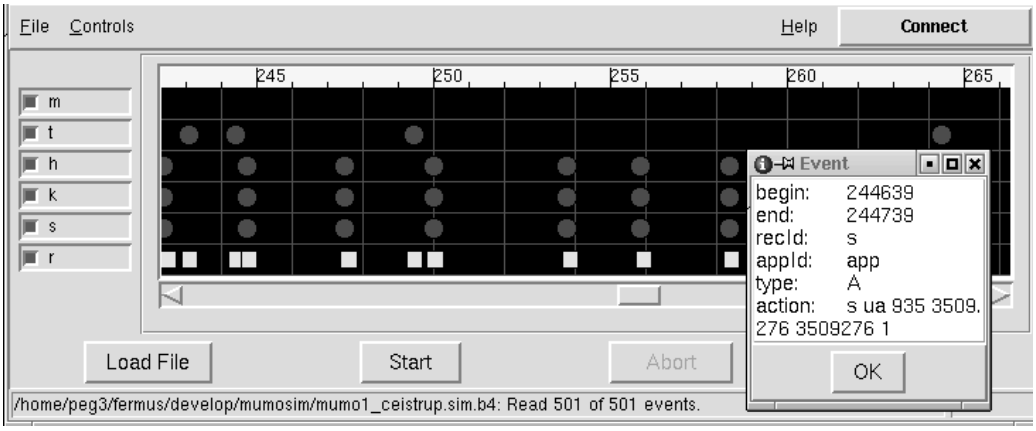


Abb. 7.1: Graphisches Front-End für den multimodalen Ereignissimulator

phische Front-End. Mit Hilfe des Simulators können verschiedene Bediensituationen abstrakt modelliert und beliebig oft identisch reproduziert werden. Auf diese Weise ist es möglich, extrem große Integratorlasten zu generieren und die Leistungsfähigkeit des Systems bei der Übertragung komplexer Datenmengen zu untersuchen, ohne immer direkt die einzelnen Erkenner- und Kontextmodule anschließen zu müssen. Zur Feinabstimmung der Integrationsstufen können die Latenzzeiten zwischen der Erkennung der Benutzereingaben und der eigentlichen Übertragung der dazugehörigen Nachrichten für jede Komponente individuell variiert werden.

Für eine sinnvolle, quantitative Bewertung des Gesamtsystems muss die Leistungsfähigkeit von unterschiedlichen Parameterkonfigurationen in Bezug auf eine Auswahl an prototypischen Bediensituationen überprüft werden. Neben einem erheblichen geringeren Zeit- und Kostenaufwand bietet eine Software-Simulation zusätzlich die Möglichkeit, das Verhalten des Integratormoduls in Grenzsituationen auszutesten, die in der Realität nur sehr schwer bzw. extrem selten erreicht werden. Darüber hinaus können ebenfalls unter minimalem Aufwand verschiedene Szenarien provoziert werden, in denen unterschiedliche Arten von Fehlern auftreten. Speziell diese Fähigkeit ist für die Evaluierung des Fehlermanagements von besonderer Bedeutung.

Ein weiterer Vorteil einer Simulation ist die Einhaltung konstanter Randbedingungen. In einem Testlauf mit realen Erkennermodulen kann das Ergebnis neben möglichen Schwankungen in der Erkennungsleistung auch von unvorhersehbaren Unterschieden im Verhalten der Testpersonen abhängig sein und auf diese Weise einer objektiven Beurteilung entgegenwirken. Der Simulator zeichnet sich demgegenüber durch deterministische Ausgaben aus. Die domänenspezifische Konfiguration der Simulationsumgebung erfolgt analog zu den anderen Systemmodulen über eine nachladbare Textdatei. An dieser Stelle können neben der Anzahl der zu simulierenden Komponenten und den jeweils dazugehörigen Ereignissequenzen u.a. auch kontextabhängige Meta-Ereignisse und Kontrollausgaben spezifiziert werden. Für detaillierte Informationen zu der systemtechnischen Umsetzung des Simulators sei auf [4] verwiesen.

Bei der folgenden Evaluierung werden lediglich diskrete Kontextdaten verarbeitet, die an einzelne Status-Nachrichten gekoppelt sind. Nach diesem vereinfachten Ansatz ist ein spezifisches Kontextereignis $E_K \in \mathcal{E}_K$ solange gültig, bis es durch eine anders lautende Status-Nachricht aufgehoben oder auf einen neuen Wert gesetzt wird. In Bezug auf die Benutzerereignisse werden pro Eingabe jeweils zwei Nachrichten über das Netzwerk versendet, die den Beginn

bzw. das Ende des Erkennungsprozesses markieren. Aufgrund der geringen Laufzeiten zwischen Simulator und Integrator können die Ankunftszeitpunkte der Nachrichten als verzögerungsfrei betrachtet werden. Die Reihenfolge der einzelnen Nachrichten entspricht der tatsächlichen Abgabereihenfolge der Befehle. Innerhalb einer Nachricht werden dann keine zeitlichen Informationen mehr übertragen.

Die Plattform für die Evaluierung ist ein Standard-PC mit einem Pentium III Prozessor, 750 MHz CPU und 256 MByte Hauptspeicher. Als Betriebssystem wird Linux in der Kernelversion 2.4.18 basierend auf der SUSE-Distribution 8.1 eingesetzt. Während der verschiedenen Messungen laufen auf diesem Rechner ausschließlich der Intentionsdekoder und grundlegende Systemprozesse. Die zusätzlich benötigten Komponenten wie die Simulationsumgebung, der Broadcast-Server und verschiedene Auswertungsprogramme werden auf einem anderen System betrieben und über TCP/IP-Sockets angesteuert. Dabei wird die bereits in Abschnitt 5.5 beschriebene Schnittstellentechnologie verwendet.

7.2 Zeitliche Segmentierung

Die erste Phase im Integrationsprozess besteht aus einer Identifikation zusammengehöriger Benutzereingaben und Kontextinformationen. Abschnitt 6.3 beinhaltet eine detaillierte Beschreibung der entwickelten Verfahren. Für den zugrundeliegenden Spotting-Algorithmus stehen verschiedene Ausbaustufen zur Verfügung, die getrennt voneinander bewertet werden. Im Hinblick auf den praktischen Einsatz sind an verschiedenen Stellen leichte Änderungen gegenüber der theoretischen Systembeschreibung vorgenommen worden. Das Testmaterial für die zeitliche Segmentierung besteht aus protokollierten Benutzereingaben sowie daraus abgeleiteten, künstlich erzeugten Bediensequenzen. In den Auswertungen wird auf beide Anwendungsdomänen eingegangen. Aufgrund der strukturell höheren Komplexität liegt der Fokus jedoch eher auf dem DVA-Szenario.

Die Evaluierung findet in fünf Ebenen statt. Auf der ersten Stufe ($\Phi_{V(1)}$) werden nur zeitlich überlagerte Befehle als multimodal betrachtet. Sequentielle Interaktionen innerhalb eines a priori festgelegten, konstanten Zeitfensters können erst in der nächsten Phase ($\Phi_{V(2)}$) interpretiert werden. Die dritte Ebene ($\Phi_{V(3)}$) kennzeichnet sich durch die Hinzunahme des Überlagerungsverfahrens zur Analyse komplex überlagerter Ereignissequenzen. Während die ersten drei Stufen ausschließlich zeitliche Informationen auswerten, werden in der nachfolgenden Ebene ($\Phi_{V(4)}$) zusätzlich die semantischen Relationen der Ereignisse und vorhandenes Kontextwissen betrachtet. Die letzte Stufe ($\Phi_{V(5)}$) untersucht speziell den Zusammenhang zwischen taktilen und semantisch höherwertigen Benutzereingaben, da hier aufgrund der systemtechnischen Unterschiede ein verstärktes Maß an Kontextwissen benötig wird.

7.2.1 Realdaten

In den Protokolldateien sind für einzelne Eingaben $E \in \mathcal{E}^{\mathcal{ZBE}}$ von ausgewählten Benutzern $B \in \mathcal{B}_{ZVP}$ die jeweils dazugehörigen Interaktionszeiten $t_s(E)$ und $t_e(E)$ für alle $E \in \mathcal{E}^{\mathcal{ZBE}}$ nachträglich manuell erfasst worden (siehe Abschnitt 4.2.4). Die Menge der zeitlich annotierten Ereignisdaten kann direkt in die Simulationsumgebung geladen werden und ermöglicht auf diese Weise eine exakte Rekonstruktion des realen Versuchsablaufs.

DVA		allgemein		gruppenspezifisch		personenspezifisch	
		korrekt	falsch	korrekt	falsch	korrekt	falsch
unimodal	gesamt	457	6	461	2	461	2
bimodal	gesamt	365	23	367	21	374	14
	sequentiell	77	10	79	8	86	1
	überlagert	288	13	288	13	288	13
trimodal	gesamt	55	0	55	0	55	0
	sequentiell	3	0	3	0	3	0
	überlagert	52	0	52	0	52	0

Tab. 7.1: Evaluierung des Spotting-Verfahrens auf der höchsten Stufe ($\Phi_{V(5)}$) anhand der semantisch höherwertigen Interaktionen E^{HKS} im DVA-Szenario; aufgeteilt nach unimodalen, bimodalen und trimodalen Eingabesequenzen mit den jeweiligen Subkategorien und bezogen auf eine allgemeine, eine gruppenspezifische und eine personenspezifische Konfiguration des Algorithmus

Zu Evaluierungszwecken werden zwei Indizes, die bereits im Rahmen der Post-Klassifikation der Wizard-Events (siehe Abschnitt 4.2.3) gesetzt worden sind, in jede Benutzernachricht automatisch eingefügt. Die erste Markierung i mit $i \in \{1..N_B^{ZBE}\}$ enthält für den jeweiligen Versuchsteilnehmer $B \in \mathcal{B}_{ZVP}$ den Index des entsprechenden Eingabeereignisses $E_i \in \mathcal{E}_B^{ZBE}$. Dieser Wert ergibt sich direkt aus der zeitlichen Reihenfolge der auflaufenden Eingabeereignisse. Demgegenüber beschreibt die zweite Markierung j die Zuordnung einer einzelnen, modalitätenspezifischen Eingabe E_i zu einem speziellen Integrationscluster $\mathcal{C}_{I,j}$.

Diese beiden Markierungshilfen werden im Auswertungsmodul zur Überprüfung der Ergebnisse benutzt. Bei einem korrekt durchgeführten Spotting ist der zweite Index j jeweils für alle Ereignisse E_i in dem Integrationscluster $\mathcal{C}_{I,j}$ identisch und unterscheidet sich von den entsprechenden Werten der benachbarten Intervalle $\mathcal{C}_{I,j-1}$, $\mathcal{C}_{I,j+1}$, $\mathcal{C}_{I,j-2}$, $\mathcal{C}_{I,j+2}$. Ist eine der beiden Bedingungen nicht erfüllt, so liegt ein Fehler vor. Mit Hilfe der beiden Indizes können falsch segmentierte Ereignisse zudem eindeutig lokalisiert werden.

Sämtliche Versuchsdaten, die im Rahmen des SHM-Labelings nachbearbeitet worden sind, können in die Evaluierung einbezogen werden. Insgesamt beläuft sich das Testmaterial auf 906 Eingaben von 19 Versuchspersonen im DVA-Szenario und auf 404 Eingaben von 16 Versuchspersonen aus dem AIA-Szenario. Die Eingaben verteilen sich auf taktile und semantisch höherwertige Interaktionen. Bei den folgenden Betrachtungen wird das Spotting-Modul ausschließlich auf der höchsten Stufe ($\Phi_{V(5)}$) betrieben. Das verwendete Benutzerwissen ergibt sich jeweils als Mittelwert über die Teilnehmer der betrachteten Benutzerklasse.

Die Leistungsfähigkeit des Spotting-Verfahrens kann primär anhand einer Untersuchung der semantisch höherwertigen Interaktionen beurteilt werden. Bezogen auf das DVA-Szenario sind die Ergebnisse in Tabelle 7.1 dargestellt. In den einzelnen Spalten wird zwischen einer allgemeinen, einer gruppenspezifischen und einer personenspezifischen Konfiguration unterschieden und jeweils die Anzahl der korrekten und der fehlerhaft durchgeführten Ereignis-Klassifikationen ausgewiesen. Neben den Angaben für die Klasse der unimodalen Interaktionen sind die Ergebnisse im Hinblick auf die bimodalen und die trimodalen Eingabeereignisse zusätzlich noch nach den jeweiligen zeitlichen Relationen aufgeschlüsselt.

Insgesamt beeindruckt die geringe Fehlerquote bei der Auswertung der Realdaten. Bereits auf der Basis einer allgemeinen Konfiguration stehen 877 korrekt segmentierte Interaktionen lediglich 29 falsch eingeordneten Benutzereingaben gegenüber. Trimodale Interaktionen werden vollständig korrekt klassifiziert. Eine detaillierte Betrachtung der Fehler liefert im Wesentlichen zwei Fehlerquellen. Durch eine ungenügend angepasste Voreinstellung für den Zeitabstand $T_L(E_a, E_b)$ zwischen zwei zusammengehörigen sequentiellen Ereignissen E_a und E_b werden E_a und E_B jeweils als unimodale Eingaben eingestuft. Diese Problematik ist für vier der sechs unimodalen Fehlklassifikationen verantwortlich. Als zweite Fehlerquelle kann die dominierende Rolle der zeitlichen Überlagerung innerhalb des Regelwerks identifiziert werden. Bei den 13 fehlerhaften Fällen handelt es sich ausnahmslos um Interaktionen, die trotz einer Überlagerung voneinander unabhängig sind.

Durch die Ausnutzung von benutzerspezifischem Kontextwissen (siehe rechte Spalte von Tabelle 7.1) lässt sich die erste Fehlerquelle weitestgehend ausschließen. Im Vergleich zu der allgemeinen Konfiguration können die vier oben erwähnten unimodalen und neun der 23 bimodalen Fehlklassifikationen vermieden werden. Wie die mittlere Spalte von Tabelle 7.1 zeigt, reicht der Einbezug von gruppenspezifischem Kontextwissen hier noch nicht aus. Für eine sinnvolle Anwendung sind die Schwankungen bzgl. T_L innerhalb der einzelnen Benutzergruppen $U \in \{ANF, NOR, EXP\}$ noch zu groß. Daher können nur vier unimodale und zwei bimodale Fehlerkennungen umgangen werden.

Bei der zweiten Fehlerquelle handelt es sich zumeist um einzelne Interaktionen, die innerhalb von längeren, zeitlich überlagerten Eingabesequenzen auftreten. Aufgrund der stark unterschiedlichen Auftrittsformen sind diese Fehler nur mit einem immensen Integrationsaufwand zu vermeiden. Im Vergleich zu der ersten Fehlerquelle lässt sich weder mit gruppenspezifischem noch mit personenspezifischem Kontextwissen ein deterministisches, prototypisches Verhalten nachweisen, mit dem jeweils mehr als eine Fehlklassifikation ausgeschlossen werden kann. Aus diesem Grund bleibt der zweite Fehlertyp unbehandelt.

Benutzereingaben mit den beiden taktilen Modalitäten Maus (E^T) und Tastatur (E^A) stellen genau dann eine zusammengehörige multimodale Interaktion dar, wenn sie ebenfalls die zeitlichen Kriterien hinsichtlich Überlagerung und Sequentialität erfüllen. Ansonsten werden sie als unimodale Interaktion interpretiert und direkt ausgeführt. In dem zeitlich erfassten Testmaterial sind insgesamt 2658 ausschließlich unimodale taktile Interaktionen enthalten. Diese Zahl steht lediglich 310 Kombinationen mit anderen Modalitäten gegenüber. In allen Eingabekonstellationen funktioniert das Spottingverfahren bereits mit einer allgemeinen Konfiguration vollständig korrekt.

Die multimodalen Interaktionen unter Beteiligung der taktilen Modalitäten bestehen in insgesamt 206 Fällen ausschließlich aus bimodalen Eingabesequenzen mit Maus und Tastatur. In Bezug auf die Verteilung der strukturellen Informationsanteile werden bei 89 Interaktionen zunächst die Navigationsart durch Sprache oder Gestik spezifiziert und erst anschließend die Richtung als zusätzlicher Parameter mittels Maus oder Tastatur angegeben. Umgekehrt folgen nur 15 Parametereingaben mit den semantisch höherwertigen Modalitäten auf taktile Funktionsangaben. Im Zusammenhang mit taktilen Eingaben treten korrigierende Fehler in einer besonderen Form auf. Bei insgesamt 77 Eingabesequenzen werden Funktionsangaben mit Maus oder Tastatur direkt von einer divergierenden Funktionsangabe mittels derselben Modalität gefolgt. Mit einem Anteil von 57 Beobachtungen stammt dieses Interaktionsmuster von zwei Benutzern, die zu der Gruppe der Anfänger gehören. Das Verhalten lässt auf eine deutliche Unsicherheit in Bezug auf die im Versuch benötigten Navigationsarten schließen.

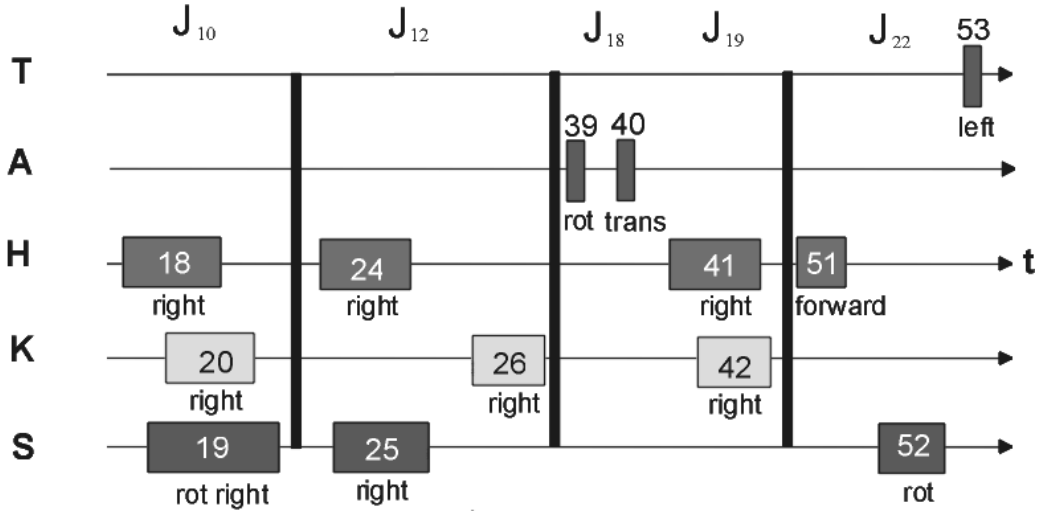


Abb. 7.2: Darstellung ausgewählter Simulationsszenarien $\{J_{10}, J_{12}, J_{18}, J_{19}, J_{22}\}$ aus der DVA-Domäne; bezogen auf die semi-formalen Beschreibungen in den beiden Tabellen 7.2 und 7.3

7.2.2 Ausgewählte Szenarien

Im Laufe der Untersuchungen sind verschiedene Simulationsdaten entstanden, mit denen Bedienszenarien dokumentiert werden, die im Hinblick auf eine zeitliche Segmentierung der real zusammengehörigen Ereignisse zu Problemen und Fehlklassifikationen führen. Ein Auswahl prototypischer Daten ist in einer zentralen Datei zusammengestellt. Anhand dieser Szenarien kann die Funktionsweise des Spotting-Moduls auf den einzelnen Auswertungsebenen exemplarisch demonstriert und bewertet werden.

Tabelle 7.2 und Tabelle 7.3 enthalten eine semi-formale Beschreibung von insgesamt 26 Auswertungsszenarien $\mathcal{J} = \{J_1, J_2, \ldots, J_{26}\}$. Die einzelnen $J \in \mathcal{J}$ können jeweils vollständig unabhängig voneinander betrachtet werden. Während in der zweiten Spalte die Zusammensetzung der Ereignisse erläutert wird, können der dritten Spalte die Elemente der verschiedenen Integrationscluster $\mathcal{C}_I \in \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{36}\}$ entnommen werden. Bei einer korrekten Interpretation der Ausgangsdaten lassen sich genau 36 Interaktionen ableiten.

Die 17 Szenarien in Tabelle 7.2 repräsentieren typische Eingabesequenzen, die in den Benutzerstudien in genau dieser oder zumindest in ähnlicher Form mehrfach aufgetreten sind. Demgegenüber stellen die neun Szenarien in Tabelle 7.3 konstruierte Sonderfälle dar, mit denen der Spotting-Algorithmus in Extremsituationen ausgetestet werden kann. Die beiden Abbildungen 7.2 und 7.3 enthalten für ausgewählte Szenarien aus den Tabellen zusätzlich eine Darstellung der zeitlichen Relationen, der zugleich auch die funktionale Zusammensetzung der einzelnen Informationsanteile entnommen werden kann.

Tabelle 7.4 weist, jeweils in Abhängigkeit von der verwendeten Ausbaustufe des Spotting-Moduls $\Phi_{V(x)}$ mit $x \in \{1..5\}$, die Anzahl der korrekt bzw. der falsch klassifizierten Szenarien aus. Die ersten vier Verarbeitungsstufen $\Phi_{V(1)}$ bis $\Phi_{V(4)}$ verfügen noch nicht über die Möglichkeit, taktile Interaktionen erkennen und verarbeiten zu können. Eine korrekte Einordnung der neun Szenarien J_{14} bis J_{22} ist daher in diesen Auswertungsebenen noch nicht möglich. Beispielsweise resultieren alle neun Einordnungsfehler mit Bezug auf $\Phi_{V(4)}$ aus dieser Beschränkung.

	Beschreibung	Zuordnung
J_1	unimodale Interaktion E_1^S, Vollbefehl	$\mathcal{C}_1 = \{E_1^S\}$
J_2	gestaffelt überlagerte Eingabe, bestehend aus Vollbefehl E_2^S und einzelner Parameterangabe E_3^H, redundant-komplementär	$\mathcal{C}_2 = \{E_2^S, E_3^H\}$
J_3	Parameterangabe E_5^H, die zeitlich in einen Vollbefehl E_4^S eingebettet ist; Interaktionen sind rivalisierend-komplementär	$\mathcal{C}_3 = \{E_4^S, E_5^H\}$
J_4	Funktionsangabe E_6^H und Parameter E_7^S, die zeitlich gestaffelt überlagert sind; Informationsanteile komplementär zueinander	$\mathcal{C}_4 = \{E_6^H, E_7^S\}$
J_5	Funktion E_8^S und komplementärer Parameter E_9^K sequentiell nacheinander mit einer zeitlichen Verzögerung von $T_L = 520$ms	$\mathcal{C}_5 = \{E_8^S, E_9^K\}$
J_6	E_{10}^S, E_{11}^K analog zu J_5 mit zeitlicher Verzögerung $T_L = 1420$ms	$\mathcal{C}_6 = \{E_{10}^S, E_{11}^K\}$
J_7	Parameter E_{12}^S und E_{13}^K sequentiell nacheinander mit einer zeitlichen Verzögerung von $T_L = 520$ms; Interaktionen redundant	$\mathcal{C}_7 = \{E_{12}^S\}$ $\mathcal{C}_8 = \{E_{13}^K\}$
J_8	Vollbefehl E_{14}^S und Vollbefehl E_{15}^H sequentiell nacheinander mit $T_L = 980$ms; Informationsanteile rivalisierend-komplementär	$\mathcal{C}_9 = \{E_{14}^S\}$ $\mathcal{C}_{10} = \{E_{15}^H\}$
J_9	zwei sprachliche Vollbefehle E_{16}^S und E_{17}^S sequentiell nacheinander mit $T_L = 300$ms, rivalisierende Informationsanteile, Spezialfall eines korrigierenden Fehlers (siehe Abschnitt 6.3.3)	$\mathcal{C}_{11} = \{E_{16}^S, E_{17}^S\}$
J_{10}	trimodale Interaktion mit den Parametern E_{18}^H, E_{20}^K und dem Vollbefehl E_{19}^S, redundant-komplementär nach Abbildung 7.2	$\mathcal{C}_{12} = \{E_{18}^H, E_{19}^S, E_{20}^K\}$
J_{11}	analog zu Szenario J_{10}, wobei die beiden Parameter E_{21}^H und E_{23}^K rivalisierend zueinander sind, dominiert aber durch E_{22}^S	$\mathcal{C}_{13} = \{E_{21}^H, E_{22}^S, E_{23}^K\}$
J_{12}	trimodale Interaktion mit den Parametern E_{24}^H, E_{25}^S und und E_{26}^K mit $T_L(E_{25}^S, E_{26}^K) = 340$ms, redundante Informationsanteile	$\mathcal{C}_{14} = \{E_{24}^H, E_{25}^S, E_{26}^K\}$
J_{13}	analog zu Szenario J_{12} mit dem Unterschied, dass die Parameter E_{27}^H und E_{28}^S rivalisierend zu dem Parameter E_{29}^H sind	$\mathcal{C}_{15} = \{E_{27}^H, E_{28}^S\}$ $\mathcal{C}_{16} = \{E_{29}^K\}$
J_{14}	taktile, gestaffelt überlagerte Interaktion mit Funktion E_{30}^T und Parameter E_{31}^A gefolgt von einer unimodalen Parameterangabe E_{32}^T mit zeitlicher Verzögerung von $T_L(E_{31}^A, E_{32}^T) = 600$ms	$\mathcal{C}_{17} = \{E_{30}^T, E_{31}^A\}$ $\mathcal{C}_{18} = \{E_{32}^T\}$
J_{15}	Funktionsangabe E_{33}^A vor Parameter E_{34}^S mit $T_L = 650$ms	$\mathcal{C}_{19} = \{E_{33}^A, E_{34}^S\}$
J_{16}	Parameter E_{35}^K überlagert von Funktionsangabe E_{36}^T	$\mathcal{C}_{20} = \{E_{35}^K, E_{36}^T\}$
J_{17}	Funktionsangabe E_{38}^S nach Parameter E_{37}^T mit $T_L = 240$ms	$\mathcal{C}_{21} = \{E_{37}^T, E_{38}^S\}$

Tab. 7.2: Beschreibung ausgewählter Szenarien $\mathcal{J} = \{J_1, J_2, \ldots, J_{17}\}$ für die ebenenspezifische Evaluierung des Spottingmoduls; abgeleitet aus den Protokollen der Benutzerstudien; die dritte Spalte enthält jeweils die korrekte Zuordnung der einzelnen Ereignisse $E \in \{E_1, E_2, \ldots, E_{38}\}$ zu den entsprechenden Integrationsintervallen $\mathcal{C}_I \in \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{21}\}$; für eine Darstellung der beiden Szenarien J_{10} und J_{12} siehe auch Abbildung 7.2

	Beschreibung	Zuordnung
J_{18}	zwei rivalisierende Funktionsangaben E^A_{39} und E^A_{40} mit einer zeitlichen Verzögerung von $T_L = 170$ms, korrigierender Fehler	keine Clusterbildung
J_{19}	Weiterführung von J_{18} mit zwei redundanten Parameterangaben E^H_{41} und E^K_{42}, die zeitlich gestaffelt überlagert sind	$\mathcal{C}_{22} = \{E^A_{40}, E^H_{41}, E^K_{42}\}$
J_{20}	gestaffelte, redundant-komplementäre Überlagerung eines Vollbefehls E^S_{43} und einer Parameterangabe E^H_{44}, gefolgt von einer isolierten Parameterangabe E^T_{45} mit $T_L(E^H_{44}, E^T_{45}) = 800$ms	$\mathcal{C}_{23} = \{E^S_{43}, E^H_{44}\}$ $\mathcal{C}_{24} = \{E^T_{45}\}$
J_{21}	Funktionsangabe E^S_{46} gefolgt von vier gleichen Parameterangaben E^A_{47}, E^A_{48}, E^A_{49} und E^A_{50}; Parameter werden gezählt	$\mathcal{C}_{25} = \{E^S_{46}, E^A_{50}\}$
J_{22}	Parameter E^H_{51} gefolgt von einer Funktionsangabe E^S_{52} und einem weiteren Parameter E^T_{53}, Interaktionen sequentiell zueinander	$\mathcal{C}_{26} = \{E^H_{51}\}$ $\mathcal{C}_{27} = \{E^S_{52}, E^T_{53}\}$
J_{23}	Beispiel für die Anwendung des Überlagerungsverfahrens mit den beiden Parametern E^H_{54} und E^H_{56} und einer Funktionsangabe E^S_{55}	$\mathcal{C}_{28} = \{E^H_{54}, E^S_{55}\}$ $\mathcal{C}_{29} = \{E^H_{56}\}$
J_{24}	analog zu Szenario J_{23} mit den Parametern E^H_{57} und E^H_{59}, aber einem sprachlichen Vollbefehl $E^S_{58} = trans\ \ left$	$\mathcal{C}_{30} = \{E^H_{57}\}$ $\mathcal{C}_{31} = \{E^S_{58}, E^H_{59}\}$
J_{25}	Überlagerungsverfahren für trimodale Interaktionen; siehe Abbildung 7.3 und für die Zusammensetzung der Informationsanteile	$\mathcal{C}_{32} = \{E^H_{60}, E^S_{61}, E^K_{62}\}$ $\mathcal{C}_{33} = \{E^H_{63}, E^K_{64}, E^S_{65}\}$
J_{26}	komplex überlagerte Interaktionssequenz mit unterschiedlichen Modalitäten und Informationsanteilen, für eine Darstellung der Zusammenhänge siehe Abbildung 7.3	$\mathcal{C}_{34} = \{E^K_{66}, E^S_{67}, E^H_{68}\}$ $\mathcal{C}_{35} = \{E^S_{69}, E^K_{70}\}$ $\mathcal{C}_{36} = \{E^H_{71}, E^K_{72}, E^S_{73}\}$

Tab. 7.3: Beschreibung zusätzlicher, künstlich erzeugter Szenarien $\mathcal{J} = \{J_{18}, J_{19}, \ldots, J_{26}\}$ für die weiterführende ebenenspezifische Evaluierung des Spottingmoduls; analog zu Tabelle 7.2 enthält die dritte Spalte die korrekte Zuordnung der einzelnen Ereignisse $E \in \{E_{39}, E_{40}, \ldots, E_{73}\}$ zu den entsprechenden Integrationsintervallen $\mathcal{C}_I \in \{\mathcal{C}_{22}, \mathcal{C}_{23}, \ldots, \mathcal{C}_{36}\}$; siehe auch Abbildungen 7.2 und 7.3

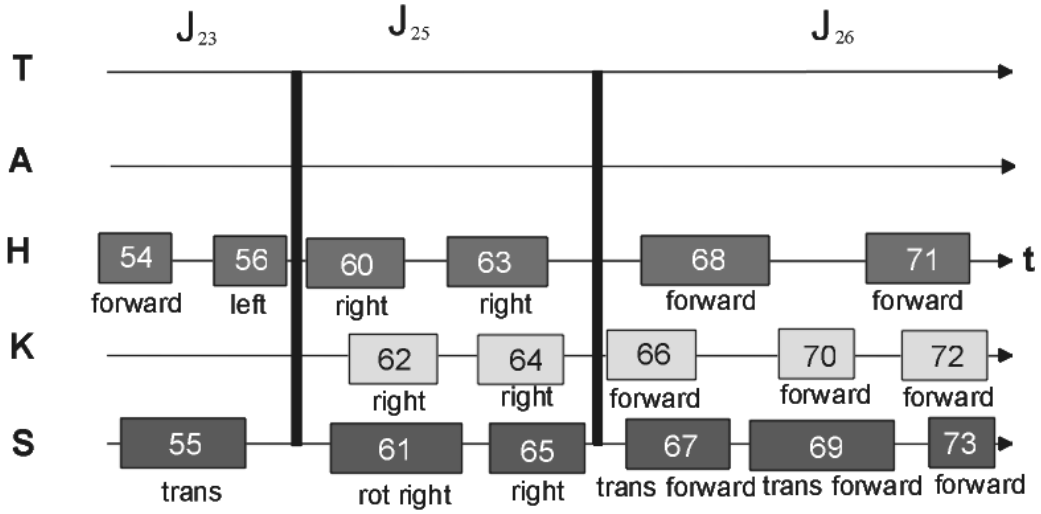


Abb. 7.3: Weitere Simulationsszenarien $\{J_{23}, J_{25}, J_{26}\}$ bezogen auf Tabelle 7.3

DVA	$\Phi_{V(1)}$	$\Phi_{V(2)}$	$\Phi_{V(3)}$	$\Phi_{V(4)}$	$\Phi_{V(5)}$
korrekt	9	8	11	17	26
falsch	17	18	15	9	0

Tab. 7.4: Ergebnisse für die stufenweise Evaluierung des Spotting-Verfahrens in der DVA-Domäne anhand der 26 konstruierten Bedienszenarien aus den Tabellen 7.2 und 7.3; dargestellt ist jeweils die Anzahl der korrekt bzw. der falsch bearbeiteten Szenarien bezogen auf die verwendete Ausbaustufe des Spotting-Moduls $\Phi_V \in \{\Phi_{V(1)}, \Phi_{V(2)}, \Phi_{V(3)}, \Phi_{V(4)}, \Phi_{V(5)}\}$

Auf der ersten Stufe $\Phi_{V(1)}$ können nur zeitlich überlagerte Benutzereingaben miteinander kombiniert werden. Die Ereignisse in den Szenarien J_1 bis J_4 werden korrekt in die entsprechenden Integrationscluster $\mathcal{C}_1$ bis $\mathcal{C}_4$ eingeordnet. Bei den Szenarien J_5 bis J_9 erfolgt hingegen immer eine Klassifikation in unimodale Eingabeblöcke. In J_{10} und J_{11} findet wiederum ein korrektes Spotting statt. Bei den simulierten Sequenzen in J_{12} und J_{13} wird jeweils das sequentielle Ereignis als unimodale Interaktion eingeordnet, wodurch in einem Fall ein Fehler entsteht. Da in dem Spotting-Modul auf $\Phi_{V(1)}$ keine Regeln zur Behandlung zeitlich überlagerter Befehlssequenzen hinterlegt sind, werden die Szenarien J_{23} bis J_{26} nicht korrekt erfasst.

Angesichts der zahlreichen Bedienszenarien, die in $\Phi_{V(1)}$ noch nicht richtig behandelt werden können, scheint sich diese Ausbaustufe des Spotting-Moduls nicht besonders für den praktischen Einsatz zu eignen. Die Simulationsdatei enthält jedoch vor allem eine Ansammlung von Szenarien, die in dieser Vielfalt bezogen auf einen einzelnen Benutzer nur äußerst selten auftreten. Es ist auf der anderen Seite interessant zu beobachten, dass bei fünf der 19 ausgewerteten Versuchsteilnehmer aus dem DVA-Szenario die Erkennungsleistung im Hinblick auf die semantisch höherwertigen Modalitäten in $\Phi_{V(1)}$ bereits genauso hoch ist wie in $\Phi_{V(5)}$.

Die zweite Ausbaustufe $\Phi_{V(2)}$ ermöglicht bereits eine korrekte Einordnung von zusammengehörigen, sequentiellen Ereignissen innerhalb eines fest vorgegebenen Bereiches. Für die Auswertungen ist dieses Zeitfenster auf 1400ms gesetzt worden. Auf diese Weise können die beiden Szenarien J_5 und J_{12} in $\Phi_{V(2)}$ richtig eingeordnet werden. Aufgrund der fehlenden semantischen Informationen werden aber die Ereignisse in J_7, J_8 und J_{13} fälschlicherweise auch als multimodale Interaktionen klassifiziert. Da in J_6 die Verzögerungszeit $T_L(E_{10}^S, E_{11}^K)$ größer als die voreingestellte Akzeptanzschwelle von 1400ms ist, stuft der Spotting-Algorithmus die Ereignisse als unimodale Interaktionen ein. Der korrigierende Fehler in J_{13} kann ebenfalls nicht korrekt interpretiert werden. Insgesamt liefert $\Phi_{V(2)}$ ein etwas schlechteres Ergebnis als $\Phi_{V(1)}$. Dies liegt vor allem daran, dass in der Simulationsdatei überdurchschnittlich viele Szenarien mit sequentiellen Ereignissen enthalten sind. Speziell in diesen Bediensituationen reicht eine Betrachtung der Verzögerungszeit mit einem konstanten Zeitfenster als alleiniges Entscheidungskriterium für die Annahme einer multimodalen Interaktion oftmals nicht aus.

Auf der dritten Ausbaustufe des Spotting-Moduls ($\Phi_{V(3)}$) durchlaufen die multimodalen Ereignissequenzen zusätzlich das in den Regeln $\Gamma_S(3)$ bis $\Gamma_S(5)$ beschriebene Überlagerungsverfahren. Es können erstmals auch die Szenarien J_{23} bis J_{26} untersucht werden, wobei lediglich in J_{24} noch eine fehlerhafte Einordnung vorgenommen wird. Der Fehler liegt an der Begrenzung des Regelkorpus auf die Auswertung von zeitbezogenen Relationen zwischen den Ereignissen. Analog zu $\Phi_{V(2)}$ werden dadurch einige sequentielle Interaktionen fälschlicherweise als unimodale Eingaben interpretiert.

Gegenüber den ersten drei Ebenen zeichnet sich die vierte Stufe des Spotting-Moduls ($\Phi_{V(4)}$) zum einen durch die Auswertung der semantischen Relationen zwischen den einzelnen Ereignissen und zum anderen durch die zusätzliche Betrachtung von verschiedenen Kontextinformationen aus. Aufgrund der variablen, benutzerspezifischen Zeitgrenzen können erstmals die Szenarien J_6 und J_7 beide richtig eingeordnet werden. Darüber hinaus führt die Analyse der funktionalen Zusammenhänge zu einer korrekten Klassifikation der Ereignisse in den Szenarien J_8, J_{13} und J_{24}. Dadurch kann auch der korrigierende Fehler in J_9 aufgelöst werden.

Auf der letzten Ausbaustufen des Spotting-Moduls ($\Phi_{V(4)}$) werden schließlich auch Regeln zur speziellen Behandlung von taktilen Benutzereingaben in den Auswertungsprozess einbezogen. Gegenüber den semantisch höherwertigen Modalitäten zeichnen sie sich durch extrem kurze Interaktionszeiten aus. Die Szenarien J_{14} bis J_{17} stellen typische Eingabesequenzen dar, die durch die Regelerweiterungen problemlos klassifiziert werden können. In dem Szenario J_{18} ist ein Sonderfall eines korrigierenden Fehlers dargestellt, bei dem explizit kein Integrationscluster erzeugt wird. Sobald der Algorithmus zwei aufeinanderfolgende, taktile Funktionsangaben erkennt wird die erste Angabe verworfen und mit der zweiten Information ein neuer Spotting-Durchlauf gestartet.

Folgt auf eine funktional abgeschlossene Eingabe mit Sprache oder Gestik eine taktile Parametereingabe (J_{20}), so wird diese als unimodale Interaktion eingestuft. Szenario J_{21} zeigt das Potenzial einer weiteren Regelerweiterung. Treten jeweils innerhalb eines benutzerspezifisch anpassbaren Zeitintervalls mehrere, semantisch identische Parameterangaben mit Touchscreen, Maus oder Tastatur auf, dann werden diese Interaktionen nicht einzeln gewertet, sondern zu einem einzigen Kommando zusammengefasst. Die Anzahl der Parameter wird dabei als zusätzliche Kontext-Information an die nachfolgende Fusionsstufe übergeben. Auf diese Weise kann sowohl die Verarbeitungszeit als auch die voreingestellte Länge der Integrationsintervalle reduziert werden. Durch das Zusammenspiel der verschiedenen Regeln lassen sich in $\Phi_{V(4)}$ sämtliche Szenarien vollständig korrekt einordnen.

7.3 Anpassung der genetischen Parameter

Das Prinzip des genetischen Fusionsansatzes ist in den Abschnitten 6.4 bis 6.6 formal eingeführt worden. Die stark theoretisch ausgerichtete Beschreibung beinhaltet die Identifikation unterschiedlicher Variablen, über die die Performanz des Algorithmus maßgeblich beeinflusst werden kann. Man bezeichnet die entsprechenden Werte daher auch als die *Strukturelemente* oder die *Zielparameter* eines evolutionären Optimierungsproblems.

Tabelle 7.5 enthält in komprimierter Form eine Zusammenstellung der verwendeten Bezeichnungen. Im Laufe der Evaluierung werden unterschiedliche Parametereinstellungen verwendet. Um sich unmittelbar auf eine Konfiguration zu beziehen, werden die Strukturparameter in kompakter Form auch als 9-Tupel $\Theta_x = (\iota, \nu, \pi, \Pi, \tau, \varphi, \Xi, \psi, \chi)$ referenziert. Dabei beinhalten die Einträge an den einzelnen Positionen jeweils den Wert des entsprechenden Parameters. Der Bezeichner x wird verwendet, um eine spezifische Konfiguration zu kennzeichnen. Sind einzelne Komponenten für die aktuelle Betrachtung nicht von Bedeutung, so werden die dazugehörigen Stellen mit einem '-' markiert. Beispielsweise ist der fünfte Eintrag τ nur dann wichtig, wenn als Selektionstyp Π die Turnierselektion $\Pi = 2$ eingestellt ist.

Parameter	Beschreibung
ι	Anzahl der Individuen in einer Population
ν	Anzahl der Chromosome in einem Individuum
π	Größe der Paarungspopulation $\mathcal{P}_P$
Π	Typ des Selektionsverfahren, $\Pi \in \{\Pi_S, \Pi_R, \Pi_T\}$ mit Π_S=Standardselektion, Π_R=Rangselektion und Π_T=Turnierselektion
τ	Größe der temporären Auswahlpopulation bei der Turnierselektion
φ	Mutationsrate
Ξ	Konvergenztyp, $\Xi \in \{\Xi_1, \Xi_2, \Xi_3\}$
ψ	Konvergenzrate
χ	Mindestanzahl an Iterationsläufen

Tab. 7.5: Zusammenstellung der essentiellen strukturellen Zielparameter in einem genetischen Algorithmus, aufgeschlüsselt nach der Bezeichnung und einer kurzen Beschreibung; abkürzend werden einzelne Konfigurationen auch über das 9-Tupel $\Theta = (\iota, \nu, \pi, \Pi, \tau, \varphi, \Xi, \psi, \chi)$ referenziert; für eine ausführliche Erklärung der Parameter sei auf die Beschreibungen in Kapitel 6 verwiesen

Die Evaluierung der genetischen Datenfusion erfolgt in zwei getrennten Schritten. In der ersten Phase wird die zeitliche Performanz vor dem Hintergrund unterschiedlicher Parameterkonfigurationen untersucht. Das konkrete Ergebnis der Integration spielt dafür zunächst keine Rolle. Die erste Phase dient lediglich dazu, die Wertebereiche der möglichen Parameterkonfigurationen im Hinblick auf zeitliche Randbedingungen für den praktischen Einsatz einzuschränken. Erst in der zweiten Phase wird auch das Ergebnis der genetischen Fusion betrachtet. Anhand von fünf prototypischen Integrationsaufgaben wird eine Auswahl von 23 Konfigurationen sowohl auf die Korrektheit der Integrationsläufe als auch auf die benötigte Zeit hin überprüft. Aus den Ergebnissen der beiden Evaluierungsphasen kann dann eine mögliche Parameterkonfiguration für realen Betrieb des Intentionsdekoders abgeleitet werden.

7.3.1 Laufzeitverhalten

Der Fusionsalgorithmus operiert auf abstrakten Elementen einer formalen Sprache. Zunächst wird überprüft, in welcher Art und Weise sich die zugrundeliegenden kontextfreien Grammatiken auf das Laufzeitverhalten des Algorithmus auswirken. Dazu werden insgesamt vier unterschiedliche Grammatiken getestet. Die erste Grammatik $G1$ stammt noch aus der Konzeptionsphase des genetischen Algorithmus und enthält lediglich eine limitierte Auflistung typischer multimodaler Interaktionsmuster. Demgegenüber modellieren die anderen drei Grammatiken jeweils das Vokabular real existierender Applikationen. Während $G2$ die Aktionen des AIA-Testszenarios beschreibt stellen $G3$ und $G4$ zwei Versionen für das DVA-Szenrio dar. Dabei modelliert $G4$ die volle Funktionalität des VRML-Browsers und $G3$ eine eingeschränkte Version, die auch in den Benutzertests angewendet wurde. Anhang B.3 enthält eine formale Angabe der einzelnen Grammatiken in Form von BNF-Regeln.

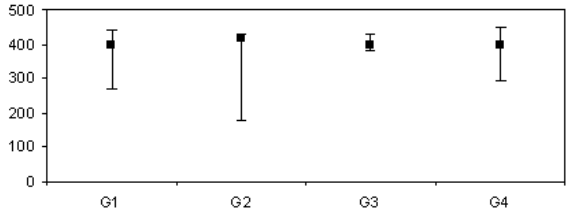


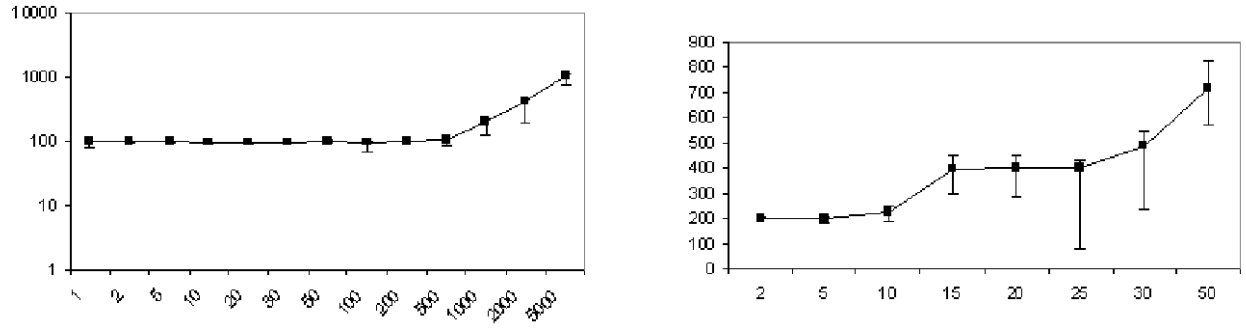
Abb. 7.4: Zeitliche Performanz für die Standardeinstellung $\Theta_S = (100; 20; 100; \Pi_T; 10; 0,001; \Xi_2; 0,8; 9)$ der genetischen Strukturparameter bezogen auf vier unterschiedliche, kontextfreie Testgrammatiken $G1, G2, G3$ und $G4$; für eine Definition der Grammatiken in BNF sei auf Anhang B.3 verwiesen

Abbildung 7.4 veranschaulicht die Laufzeiten des Fusionsalgorithmus für eine Standardeinstellung der genetischen Strukturparameter mit $\Theta_S = (100; 20; 100; \Pi_T; 10; 0,001; \Xi_2; 0,8; 9)$, jeweils bezogen auf die einzelnen Grammatiken G mit $G \in \{G1, G2, G3, G4\}$. Obwohl sowohl die Größe als auch die strukturelle Komplexität der einzelnen Grammatiken zum Teil erheblich divergieren, ergeben sich bei konstanter Wahl der einzelnen Parameterwerte vergleichbare Laufzeiten. Aufgrund dieses Ergebnisses kann die Bedeutung der konstruierenden Grammatik im weiteren Verlauf der Evaluierung vernachlässigt werden. Vor dem Hintergrund der in Abschnitt 5.1.2 aufgestellten Anforderungen unterstützt das den generischen Anspruch des Gesamtsystems und erleichtert die Portierung auf andere Anwendungsdomänen. Struktur und Größe der Grammatik haben keinen besonderen Einfluss auf die Performanz.

Im Folgenden wird der Einfluss der einzelnen genetischen Parameter auf die Laufzeit untersucht. Dabei wird immer genau ein Strukturelement aus Tabelle 7.5 innerhalb vorgegebener Grenzen variiert, während die anderen Werte konstant bleiben. Um die Ergebnisse der verschiedenen Bewertungen untereinander besser vergleichen zu können, werden die dazugehörigen Parametereinstellungen explizit angegeben. Ein Test für eine spezifische Konfiguration Θ besteht aus jeweils 100 nacheinander durchgeführten Evaluierungsläufen des Algorithmus unter identischen Randbedingungen. Neben den durchschnittlichen Laufzeiten werden dabei auch die Minima und Maxima protokolliert, um eine Aussage über die Schwankung der Performanz zu erhalten. Das Ziel der Laufzeitmessungen besteht darin, erste Anforderungen für eine optimale Parameterkonfiguration abzuleiten, die im praktischen Einsatz verwendet werden kann.

Abbildung 7.5(a) zeigt die Integrationszeit in Abhängigkeit von der Anzahl der Individuen ι in einer Population. Die Einstellung der Strukturparameter Θ_ι stimmt fast vollständig mit Θ_S überein. Lediglich für die Größe der Paarungspopulation wurde mit $\pi = 20$ ein deutlich kleinerer Wert gewählt. Für Populationsgrößen bis zu 500 Individuen bleibt die Laufzeit des Algorithmus annähernd konstant. Darüber steigt die benötigte Zeit kontinuierlich an. Als Empfehlung für die Parameterkonfiguration im realen Einsatz sollte für ι ein Wert kleiner als 500 gewählt werden, soweit die Erkennungsleistung mit dieser Einstellung ausreicht.

Der Einfluss der Chromosomanzahl ν ist in Abbildung 7.5(b) dargestellt. Die dazugehörige Parameterkonfiguration Θ_ν leitet sich unmittelbar aus der Standardeinstellung Θ_S ab. Für bis zu zehn Chromosome pro Individuum ergeben sich auf dem Testsystem annähernd gleiche Laufzeiten. Im Bereich zwischen 15 und 25 Chromosomen ist die Integrationszeit ebenfalls fast konstant, jedoch ungefähr doppelt so hoch wie auf dem ersten Plateau. Bei einer größeren Anzahl von Chromosomen kann eine deutliche Steigerung in der Laufzeit beobachtet werden. Aus diesem Grund sollte der Parameter ν im Hinblick auf die benötigte Integrationszeit möglichst nur kleine Werte annehmen.



(a) (b)

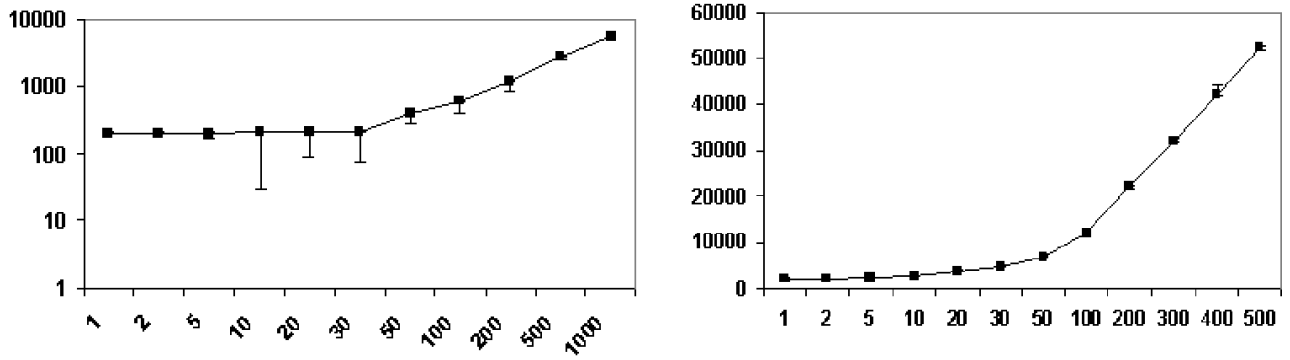
Abb. 7.5: Integrationszeit in [ms] in Abhängigkeit von (a) der Anzahl der Individuen ι in einer Population mit der Parametereinstellung $\Theta_\iota = (\iota; 20; 10; \Pi_T; 10; 0,001; \Xi_2; 0,8; 9)$ und (b) der Anzahl der Chromosome ν in einem Individuum mit $\Theta_\nu = (100; \nu; 100; \Pi_T; 10; 0,001; \Xi_2; 0,8; 9)$

Das Diagramm in Abbildung 7.6(a) veranschaulicht die Abhängigkeit der Integrationszeit von der Anzahl π der Individuen in der Paarungspopulation $\mathcal{P}_P$. Die verwendeten Strukturparameter $\Theta_\pi = (1000; 20; \pi; \Pi_T; 10; 0,001; \Xi_2; 0,8; 9)$ zeichnen sich gegenüber der Standardeinstellung Θ_S insbesondere durch eine deutlich erhöhte Zahl der Individuen in der Ausgangspopulation aus. In direkter Analogie zu der Evaluierung der Populationsgröße ι ergibt sich auch für π ein annähernd konstantes Zeitverhalten für einen gewissen Bereich. Für mehr als 30 Individuen in $\mathcal{P}_P$ steigt die Integrationszeit kontinuierlich an. Verglichen mit den Werten in Abbildung 7.5(a) fällt dieser Anstieg jedoch deutlich geringer aus. Für den praktischen Einsatz stellt eine Paarungspopulation von 30 bis 50 Individuen daher einen sinnvollen Startwert dar.

Mit Bezug auf das verwendete Selektionsverfahren Π ergeben sich unter Anwendung von $\Theta_\Pi = \Theta_S$ für die Standardselektion ($\Pi := \Pi_S$) und die Rangselektion ($\Pi := \Pi_R$) mit 198ms bzw. 197ms im Mittel fast identische Integrationszeiten. Auf Basis einer temporären Auswahlpopulation $\mathcal{P}_T$ in der Größe von $\tau = 10$ Individuen benötigt die Turnierselektion ($\Pi := \Pi_T$) demgegenüber mit durchschnittlich 402ms etwas mehr als die doppelte Zeit. Aus diesem Grund sollte aus rein zeitlicher Sicht auf die Turnierselektion weitgehend verzichtet werden. Der Einsatz von Π_T lohnt sich nur dann, wenn sich mit diesem Selektionstyp in Bezug auf die Erkennungsleitung signifikant bessere Ergebnisse ableiten lassen.

Bei Anwendung der Turnierselektion hängt die Laufzeit zusätzlich von der Größe τ der temporären Auswahlpopulation $\mathcal{P}_T$ ab. Für $\Theta_\tau = (1000; 20; 500; \Pi_T; \tau; 0,001; \Xi_2; 0,8; 9)$ ergeben sich die in Abbildung 7.6(b) dargestellten gemittelten Integrationszeiten. Bei relativ kleinen Werten mit $\tau \leq 10$ bleiben die Laufzeiten ungefähr auf einem Niveau. In dem Bereich $10 \leq \tau \leq 50$ lassen sich bereits deutliche Steigerungen beobachten. So ist die Laufzeit bei $\tau = 50$ auf dem Testsystem mit durchschnittlich 6905ms mehr als doppelt so hoch im Vergleich zu 2813ms bei $\tau = 10$. Für Werte ab $\tau = 100$ ergibt sich schließlich ein linearer Anstieg der Integrationszeiten. Für den praktischen Einsatz sollten Turnierpopulationen von mehr als 10 Individuen aufgrund des hohen Zeitaufwandes unbedingt vermieden werden.

Abbildung 7.7 zeigt die Abhängigkeit der Integrationszeit von der eingestellten Mutationsrate φ bezogen auf die Standardeinstellung $\Theta_\varphi = \Theta_S = (100; 20; 100; \Pi_T; 10; \varphi; \Xi_2; 0,8; 9)$ der genetischen Strukturparameter. Dabei ist φ in Werten zwischen $0,0001$ und 0,1 variiert



(a) (b)

Abb. 7.6: Integrationszeit in [ms] in Abhängigkeit von (a) der Anzahl der Individuen π in der Paarungspopulation $\mathcal{P}_P$ bezogen auf die Parametereinstellung $\Theta_\pi = (1000; 20; \pi; \Pi_T; 10; 0,001; \Xi_2; 0,8; 9)$ und (b) der Anzahl der Individuen τ in der temporären Auswahlpopulation $\mathcal{P}_T$ bei Anwendung der Turnierselektion und $\Theta_\tau = (1000; 20; 500; \Pi_T; \tau; 0,001; \Xi_2; 0,8; 9)$

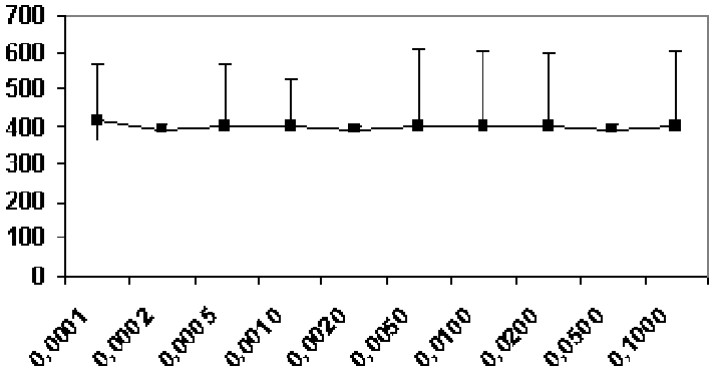


Abb. 7.7: Integrationszeit in [ms] in Abhängigkeit von der Mutationsrate φ und bezogen auf die Standardeinstellung $\Theta_\varphi := \Theta_S = (100; 20; 100; \Pi_T; 10; \varphi; \Xi_2; 0,8; 9)$ der genetischen Strukturparameter

worden. Die Ergebnisse der Evaluierungen bestätigen eindeutig die allgemeine Erfahrung aus anderen genetischen Algorithmen, dass die Einstellungen der Mutationsrate keinen Einfluss auf die Laufzeit des Algorithmus haben. Im Gegensatz zu den anderen genetischen Parametern unterliegt φ daher keinerlei Beschränkungen im Hinblick auf die Integrationszeit.

Abschließend wird noch der Einfluss des gewählten Konvergenztyps auf die Laufzeit untersucht, wobei Ξ_1 aufgrund der in Abschnitt 6.6.4 beschriebenen strukturellen Defizite von der Evaluierung ausgeschlossen ist. Basierend auf der Standardeinstellung $\Theta_\Xi = \Theta_S$ ergeben sich für Ξ_2 und Ξ_3 keine signifikanten Unterschiede. Der dritte Konvergenztyp benötigt in jeder Iteration etwas mehr Zeit, da jeweils das am besten bewertete Individuum gesucht werden muss. An der deutlich erhöhten Varianz der Laufzeiten für Ξ_3 zeigt sich, dass der zusätzliche Suchprozess unter Umständen zu einer Erhöhung der Integrationszeiten führt. Durch eine geschickte Implementierung lässt sich dieser Mehraufwand jedoch fast vollständig vermeiden. Im Hinblick auf den praktischen Einsatz ist der zweite Konvergenztyp zunächst vorzuziehen.

set	ι	π	Π	τ	φ	Ξ	ψ	χ
Θ_1	10	10	Π_T	10	0.001	Ξ_2	0.8	9
Θ_2	50	50	Π_T	10	0.001	Ξ_2	0.8	9
Θ_3	100	100	Π_T	10	0.001	Ξ_2	0.8	9
Θ_4	500	100	Π_T	10	0.001	Ξ_2	0.8	9
Θ_5	1000	100	Π_T	10	0.001	Ξ_2	0.8	9
Θ_6	100	10	Π_T	10	0.001	Ξ_2	0.8	9
Θ_7	100	50	Π_T	10	0.001	Ξ_2	0.8	9
Θ_8	100	10	Π_T	5	0.001	Ξ_2	0.8	9
Θ_9	100	10	Π_T	20	0.001	Ξ_2	0.8	9
Θ_{10}	100	10	Π_T	50	0.001	Ξ_2	0.8	9
Θ_{11}	100	10	Π_S	-	0.001	Ξ_2	0.8	9
Θ_{12}	100	10	Π_R	-	0.001	Ξ_2	0.8	9

set	ι	π	Π	τ	φ	Ξ	ψ	χ
Θ_{13}	100	10	Π_T	10	0.002	Ξ_2	0.8	9
Θ_{14}	100	10	Π_T	10	0.005	Ξ_2	0.8	9
Θ_{15}	100	10	Π_T	10	0.01	Ξ_2	0.8	9
Θ_{16}	100	10	Π_T	10	0.001	Ξ_2	0.9	9
Θ_{17}	100	10	Π_T	10	0.001	Ξ_2	0.7	9
Θ_{18}	100	10	Π_T	10	0.001	Ξ_3	0.7	9
Θ_{19}	100	10	Π_T	10	0.001	Ξ_3	0.8	9
Θ_{20}	100	10	Π_T	10	0.001	Ξ_3	0.9	9
Θ_{21}	100	10	Π_T	10	0.001	Ξ_2	0.8	20
Θ_{22}	100	10	Π_T	10	0.001	Ξ_2	0.8	30
Θ_{23}	100	10	Π_T	10	0.001	Ξ_2	0.8	50

Tab. 7.6: Konfiguration der einzelnen Strukturparameter für die Evaluierungen in der zweiten Phase mit ι=Größe der Ausgangspopulation, π=Größen der Paarungspopulation, Π=Selektionstyp, τ=Turniergröße, φ=Mutationsrate, Ξ=Konvergenztyp, ψ=Konvergenzrate und χ=Konvergenzstart

7.3.2 Konvergenzverhalten

Die Evaluierungen im letzten Abschnitt haben im Hinblick auf zeitliche Randbedingungen verschiedene Einschränkungen für die strukturierenden genetischen Parameter identifiziert. Auf Basis dieser Erfahrungen werden in der zweiten Bewertungsphase insgesamt 23 ausgewählte Konfigurationen anhand von fünf konstruierten, prototypischen Integrationsaufgaben detailliert getestet. Die Evaluierung bezieht sich dabei sowohl auf die Erkennungsleistung als auch auf die benötigte Zeit. Letztendlich besteht das Ergebnis der Untersuchungen in der Ableitung einer Parameterkonfiguration, die sich vor allem auch für den praktischen Betrieb des Intentionsdekoders eignet.

Die konkreten Einstellungen der einzelnen Strukturparameter $\Theta_E = \{\Theta_1, \Theta_2, \ldots, \Theta_{23}\}$ kann der Aufstellung in Tabelle 7.6 entnommen werden. Für sämtliche Integrationsläufe wird die Anzahl ν der Chromosome pro Individuum a priori auf zwölf begrenzt. Neben dem Speicherchromosom $\mathcal{G}_M$ und dem Aktionschromosom $\mathcal{G}_A$, die fest in jedem Individuum enthalten sind, bleiben damit zehn Chromosome übrig, mit denen Erkenner- und Kontextdaten modelliert werden können. In der folgenden Evaluierung wird die Bedeutung der Kontextdaten vernachlässigt, so dass die volle Anzahl der freien Chromosome für die Repräsentation der verschiedenen Erkennerergebnisse zur Verfügung stehen.

Als Selektionstyp Π wird trotz der größeren Laufzeiten hauptsächlich auf die Turnierselektion Π_T zurückgegriffen, da sich damit allgemein weitaus bessere Konvergenzeigenschaften und qualitativ höherwertige Ergebnisse realisieren lassen [21, 46, 53]. Um die Integrationszeit dadurch jedoch nicht überproportional zu erhöhen, wird die Größe τ der temporären Auswahlpopulation $\mathcal{P}_T$ in den meisten Fällen auf zehn Individuen beschränkt.

Um die Erkennungsleistung des Algorithmus vor dem Hintergrund realistischer Bediensituationen bewerten zu können, werden die einzelnen Parameterkonfigurationen $\Theta \in \Theta_E$ jeweils mit fünf konstruierten Integrationsaufgaben $\mathcal{Y}_I = \{Y_1, Y_2, Y_3, Y_4, Y_5\}$ getestet. Die Evaluierung erfolgt auf Basis der Testgrammatik $G1$ (siehe Anhang B.3), kann aber auch beliebig auf die anderen Grammatiken übertragen werden. Es werden typische Bediensituationen simuliert. Neben einer unimodalen Eingabe sind dabei vier verschiedene multimodale Informationszusammensetzungen abgedeckt. Formal lassen sich die Aufgaben wie folgt beschreiben:

- **Integrationsaufgabe** Y_1
  Das System wird mit einem einzelnen Semun $< semun >$ getestet. Dieser Fall entspricht einer unimodalen Eingabe ohne irgendwelche Zusatzinformationen. Als Ergebnis der Integration sollte wiederum das Semun erzeugt werden.

- **Integrationsaufgabe** Y_2
  Die zweite Aufgabe simuliert eine multimodale Eingabe, bei der eine Information durch ein komplementäres Element ergänzt wird. In dem Spottingintervall $\mathcal{C}_S$ sind die beiden Semune $< semun >$ und $< komplement\_1 >$ enthalten. Das korrekte Integrationsergebnis ist die Verkettung der beiden Elemente zu einem Wort $w =< semun >< komplement\_1 >$.

- **Integrationsaufgabe** Y_3
  In der dritten Integrationsaufgabe soll ein Semun mit einem Komplement und einer widersprüchlichen Eingabe verbunden werden. Das Spottingintervall besteht aus den Elementen $< semun >$, $< komplement\_1 >$ und $< komplement\_4 >$. Da sich der erste und der letzte Eintrag widersprüchlich zueinander verhalten, besteht das korrekte Integrationsergebnis wiederum aus dem Wort $w =< semun >< komplement\_1 >$.

- **Integrationsaufgabe** Y_4
  Bei der vierten Aufgabe wird der Algorithmus mit einem Semun $< semun >$ und zwei gültigen Komplementen $< komplement\_1 >$ und $< komplement\_2 >$ konfrontiert. Das Szenario entspricht der Eingabe eines Befehls und zwei Parametern, die untereinander redundant sind. Als Ergebnis sollte sich entweder das Wort $w_1 =< semun >< komplement\_1 >$ oder das Wort $w_2 =< semun >< komplement\_2 >$ ergeben, wobei w_1 und w_2 in Bezug auf die kontextfreie Grammatik des Zielsystems dieselbe Aktion modellieren.

- **Integrationsaufgabe** Y_5
  Die letzte Integrationsaufgabe besteht aus einem Integrationsintervall mit den drei Elementen $< semun\_1 >$, $< semun\_2 >$ und $< komplement\_1 >$. Das richtige Ergebnis soll anhand der vorhandenen Komplemente ermittelt werden. Da in diesem Fall das dritte Semun $< komplement\_1 >$ zu dem ersten Semun $< semun\_1 >$ komplementär ist, zu dem zweiten Semun $< semun\_2 >$ aber keinerlei Beziehung aufweist, ergibt sich das korrekte Integrationsergebnis aus dem Wort $w =< semun\_1 >< komplement\_1 >$.

Analog zu den zeitbezogenen Evaluierungen im letzten Abschnitt wird jede Konfiguration $\Theta \in \Theta_E$ mit jeder Integrationsaufgabe $Y \in \mathcal{Y}_I$ genau 100 Mal getestet. Damit ergeben sich insgesamt 11500 Integrationsläufe. Tabelle 7.7 enthält die entsprechenden Ergebnisse in zusammengefasster Form. Pro Aufgabe $Y_x \in \mathcal{Y}_I$ wird jeweils in der ersten Spalte die Anzahl der

	Y_1		Y_2		Y_3		Y_4		Y_5	
set	N_{I,Y_1}	T_{I,Y_1}	N_{I,Y_2}	T_{I,Y_2}	N_{I,Y_3}	T_{I,Y_3}	N_{I,Y_4}	T_{I,Y_4}	N_{I,Y_5}	T_{I,Y_5}
Θ_1	100	97	99	96	100	95	100	95	100	94
Θ_2	100	112	100	116	100	121	99	122	100	118
Θ_3	100	227	99	220	100	220	100	222	100	242
Θ_4	100	402	99	402	100	403	100	404	100	405
Θ_5	100	599	100	599	100	610	100	604	100	602
Θ_6	100	96	100	100	100	99	100	101	100	98
Θ_7	100	129	97	178	100	194	99	196	100	196
Θ_8	100	96	99	100	100	99	100	101	100	99
Θ_9	100	96	100	100	100	99	100	102	100	99
Θ_{10}	100	103	100	102	100	100	100	102	100	100
Θ_{11}	100	96	100	95	100	99	99	98	100	99
Θ_{12}	99	96	99	96	100	100	100	98	100	99
Θ_{13}	99	96	99	101	100	100	100	101	100	99
Θ_{14}	100	96	96	101	100	99	99	102	100	100
Θ_{15}	97	96	92	101	100	100	97	102	100	100
Θ_{16}	100	96	99	101	100	100	99	101	100	100
Θ_{17}	100	96	100	101	100	100	100	101	100	100
Θ_{18}	100	96	100	101	100	99	100	102	100	99
Θ_{19}	100	96	100	101	100	100	100	102	100	100
Θ_{20}	100	96	99	101	100	101	99	100	100	99
Θ_{21}	100	206	97	211	100	211	100	211	100	211
Θ_{22}	99	306	98	312	100	311	100	306	100	311
Θ_{23}	98	506	96	511	100	511	98	504	100	511

Tab. 7.7: Anzahl der korrekten Integrationsläufe $N_{I,Y}$ von 100 und die dazugehörige durchschnittliche Laufzeit $T_{I,Y}$ (in [ms]) für die einzelnen Parameterkonfigurationen $\Theta \in \{\Theta_1, \Theta_2, \ldots, \Theta_{23}\}$, jeweils bezogen auf die fünf prototypischen Bedienszenarien $Y \in \{Y_1, Y_2, Y_3, Y_4, Y_5\}$

korrekten Integrationsergebnisse N_{I,Y_x} angegeben während die zweite Spalte die durchschnittliche Laufzeit des Algorithmus T_{I,Y_x} (in [ms]) enthält. Die Testplattform stimmt mit der aus dem letzten Abschnitt überein.

Mit sämtlichen Parameterkonfigurationen sind die Integrationsaufgaben Y_3 und Y_5 absolut fehlerfrei gelöst worden. In der Tabelle kann dies daran abgelesen, dass für alle $x \in \{Y3, Y5)$ die jeweilige Spalte $N_{I,x}$ ausschließlich mit dem Wert 100 gefüllt ist. Auch die erste Integrationsaufgabe Y_1, die einer unimodalen Eingabe entspricht, wird von den meisten Konfigurationen korrekt resolviert. Lediglich die beiden Einstellungen Θ_{15} und Θ_{23} sind hier mit 97 bzw. 98 richtigen Ergebnissen etwas schlechter.

Auch im Hinblick auf die zweite Integrationsaufgabe Y_2 liefern fast alle Konfigurationen zufriedenstellende Ergebnisse. Nur die Einstellungen Θ_7, Θ_{14}, Θ_{15}, Θ_{21} und Θ_{23} weisen jeweils mehr als zwei Fehlklassifikationen auf. Die vierte Aufgabe Y_4 kann von den meisten Konfigurationen mit maximal einem Fehler aufgelöst werden. Für die beiden Konfigurationen Θ_{15} und Θ_{23} ergeben sich wiederum etwas schlechtere Resultate.

Insgesamt können mit nahezu jeder Parametereinstellung bei sämtlichen Integrationsaufgaben Ergebnisse im Bereich von 99 oder 100 richtig integrierten Semunfolgen erzielt werden. Die Konfigurationen Θ_{15} und Θ_{23} liefern grundsätzlich die schlechtesten Klassifikationsergebnisse. Bei der Einstellung Θ_{15} ist dies auf die hohe Mutationsrate von einem Prozent zurückzuführen, die dazu führt, dass einzelne Informationen in den Erkennerchromosomen zu schnell verloren gehen. Demgegenüber ist bei der Konfiguration Θ_{23} die extrem große Mindestanzahl χ der Iterationen für das im Verhältnis zu den anderen Konfigurationen schlechte Ergebnis verantwortlich. Eine frühzeitig konvergierte Population kann aufgrund der Iterationsschwelle nicht beenden und driftet in den weiteren Evolutionsschritten durch Rekombination und vor allem Mutation zu falschen Integrationsergebnissen.

Es ist bemerkenswert, dass insgesamt bei elf der 23 verschiedenen Parameterkonfigurationen in allen fünf Integrationsaufgaben jeweils höchstes nur ein falsches Integrationsergebnis beobachtet werden konnte. Die sieben Konfigurationen Θ_5, Θ_6, Θ_9, Θ_{10}, Θ_{17}, Θ_{18} und Θ_{19} haben sogar alle aufgabenbezogenen Semunfolgen vollständig korrekt integriert. Im Hinblick auf die erzielte Erkennungsleistung eignen sich besonders diese Einstellungen für den praktischen Einsatz im realen Intentionsdekoder.

Zusammenfassend bestätigen die Ergebnisse allgemeine Erfahrungen bei der Konzeption evolutionärer Algorithmen. Die Einstellungen für die einzelnen genetischen Parameter haben zumeist nur einen geringen Einfluss auf das erzielte Resultat [23, 24]. Neben der Mutationsrate φ sollte auch die Mindestanzahl der Iterationen χ nicht zu große Werte annehmen, da sich dadurch speziell auch die Integrationszeit signifikant verlängern kann.

Die Ergebnisse der Laufzeiten $T_{I,Y}$ mit $Y \in \mathcal{Y}_I$ bestätigen die initialen Erwartungen, dass die Integrationszeit kaum von der Komplexität der zu bearbeiteten Aufgabe beeinflusst wird. Bei sämtlichen Parameterkonfigurationen liegen die zeitlichen Abweichungen unterhalb von einem Prozent. Beim Vergleich der Laufzeiten in Tabelle 7.7 mit denen aus dem letzten Abschnitt ergeben sich annähernd die gleichen Werte. Besonders stark zeigt sich die Abhängigkeit von der Populationsgröße ι und der Größe der Paarungspopulation π. Lediglich die starke Abhängigkeit von der Turniergröße τ kann nicht bestätigt werden. Die Konfigurationen Θ_9 und Θ_{10} mit $\tau(\Theta_9) = 20$ bzw. $\tau(\Theta_{10}) = 50$ weisen ähnliche Laufzeiten wie die anderen Konfigurationen mit deutlich kleineren Turniergrößen auf. Dieses Ergebnis zeigt, dass im praktischen Einsatz vor allem auf die Einflüsse von ι und π geachtet werden muss.

7.4 Abschließende Bemerkungen

Die Diskussion der Ergebnisse in den letzten Abschnitten erfolgte maßgeblich anhand prototypischer Bediensituationen aus dem DVA-Szenario. Demgegenüber ist der Komplexitätsgrad der multimodalen Eingabesequenzen im AIA-Szenario weitaus geringer. Von den insgesamt 404 zeitlich markierten Interaktionen mit den semantisch höherwertigen Modalitäten Handgestik, Kopfgestik und Sprache werden 399 Eingaben richtig interpretiert. Die fünf Fehler verteilen sich auf vier unimodale Eingaben und eine bimodale Sequenz.

Bei den vier fehlerhaften unimodalen Benutzerereignissen handelt es sich um zwei bimodale Interaktionen, die aufgrund der zu hohen Verzögerung zwischen den einzelnen Eingaben nicht erkannt werden. Die falsch klassifizierte bimodale Interaktion besteht aus zwei funktional unabhängigen Eingabeereignissen, die zeitlich überlagert auftreten. In diesem Fall hat die entsprechende Versuchsperson einen Sprachbefehl zur Annahme eines eingehenden Telefonanrufs gemacht und gleichzeitig über eine Handgeste die Lautstärke der aktuellen Musikwiedergabe verringert. Taktile Interaktionen kommen in 1412 Fällen vor. Sie werden ausschließlich unimodal eingesetzt und sowohl von dem regelbasierten Spotting-Modul als auch von der genetischen Fusionskomponente korrekt erkannt und verarbeitet.

Sämtliche multimodalen Interaktionen im Automobilbereich bestehen ausschließlich aus redundanten Informationsanteilen, die bereits implizit in der Struktur des genetischen Integrationsverfahrens modelliert sind. Bezogen auf die zeitliche Segmentierung der Benutzereingaben ist bereits auf der ersten Ausbaustufe ein nahezu fehlerfreier Betrieb des Systems möglich. Lediglich für den Fall einer sequentiellen Eingabe muss der Algorithmus zusätzlich auf die Regeln der zweiten Stufe zurückgreifen. Die zeitliche Überlagerung mehrerer Eingaben kann nur auf der vierten Ebene korrekt behandelt werden. In den protokollierten Daten aus den Benutzerstudien kommen diese beiden Szenarien mit insgesamt drei beobachteten Eingabesequenzen allerdings extrem selten vor.

KAPITEL 8

Demonstratoren

In diesem Kapitel werden die erarbeiteten Konzepte im Kontext von drei realen Anwendungsszenarien diskutiert. Den Anfang bildet eine kurze Erläuterung der allgemein verwendeten Erkennungstechnologien. Dabei werden jeweils die einzelnen Komponenten zur Auswertung taktiler und semantisch höherwertiger Interaktionen explizit angesprochen. Sämtliche Applikationen basieren auf der gleichen, in Kapitel 5 vorgestellten Systemarchitektur und analysieren und interpretieren multimodale Benutzereingaben nach dem hybriden Integrationsansatz aus Kapitel 6. Während das erste Szenario den Aufbau eines multimodalen VRML-Browsers in einer desktop-orientierten Virtual-Reality Umgebung relativ ausführlich beschreibt, geht das zweite Szenario in etwas kürzerer Form auf die Bedienung von verschiedenen Audio- und Kommunikationsdiensten in einer Automobilumgebung ein. Die beiden, in den Benutzerstudien verwendeten Test-Prototypen leiten sich unmittelbar aus diesen Demonstratoren ab. Darüber hinaus wird in diesem Kapitel eine potenzielle Schnittstelle zwischen den unterschiedlichen Anwendungsdomänen aufgezeigt. Unter Berücksichtigung der existierenden Randbedingungen zeigt der entsprechende Demonstrator die Vorteile dreidimensionaler Anzeigeelemente bei der Bedienung automotiver Infotainment-Applikationen. Zum Abschluss des Kapitels wird noch ein kurzer Ausblick auf eine mögliche medizintechnische Nutzung der entwickelten Lösungsansätze im Rahmen einer präoperativen Tumoranalyse gegeben.

8.1 Verwendete Erkennermodule

Im Hinblick auf die automatische Verarbeitung der Benutzereingaben lassen sich die Erkennermodule im Wesentlichen zwei verschiedenen Klassen zuordnen. Während die erste Kategorie hauptsächlich taktile Interaktionen mit Maus, Tastatur oder ähnlichen Geräten auswertet, werden in der zweiten Kategorie Interaktionen mit semantisch höherwertigen Modalitäten wie Sprache, Gestik oder Mimik untersucht. Für die Realisierung der Eingabemodule werden sowohl industriell verfügbare Software-Produkte als auch spezielle, am Lehrstuhl für Mensch-Maschine-Kommunikation der Technischen Universität München entwickelte Forschungsprototypen eingesetzt. Dabei wird an mehreren Stellen auf die umfangreichen Erfahrungen aus den verschiedenen Vorarbeiten am Lehrstuhl zurückgegriffen [80].

Verarbeitung taktiler Interaktionen

Bis auf wenige Ausnahmen basiert die Mensch-Maschine-Interaktion bei dem Hauptanteil der kommerziell verfügbaren Informationssysteme noch auf rein taktilen Eingabeparadigmen. Vor dem Hintergrund einer möglichst breiten modalitätenspezifischen Redundanz wird diese Eingabeform daher auch in den hier vorgestellten Demonstratoren unterstützt. Neben den beiden klassischen, aus dem täglichen Umgang mit dem Computer vertrauten Geräten Maus und Tastatur können auch zusätzliche Interaktionsmodule eingebunden werden, die speziell auf die Anforderungen der jeweiligen Anwendungsdomäne abgestimmt sind. Systemtechnisch muss dazu lediglich ein Konvertermodul implementiert werden, mit dessen Hilfe sich die gerätespezifischen Funktionalitäten auf eine kontextfreie Grammatik abbilden lassen.

Erkennung sprachlicher Äußerungen

Die Verwendung der Sprache als Eingabemodalität ist ein wichtiger Schritt in Richtung einer natürlichen Mensch-Maschine-Kommunikation. Gegenüber taktilen Eingabegeräten erlaubt die Sprache eine direkte Bedienung von Systemfunktionalitäten. Die Hände stehen vollständig für andere Aufgaben zur Verfügung. Unterschiedliche Benutzer weisen im Allgemeinen eine hohe Varianz in Bezug auf das verwendete Vokabular und die Verbosität auf. In den Demonstratoren werden daher sowohl kommandosprachliche als auch natürlichsprachliche Erkenner eingesetzt. Während die Erkennung einzelner Wörter auf dem kommerziell verfügbaren Spracherkenner von Lernout and Hauspie [88] basiert, gehen die Komponenten zur Auswertung spontaner, natürlichsprachlicher Äußerungen im Kern auf die Arbeiten von Müller und Stahl [104, 157] zurück. Je nach Einsatzzweck kann dabei zwischen zwei verschiedenen Erkennerversionen ausgewählt werden.

Der erste Ansatz ist ein einstufiger, stochastischer Top-Down Parser, auf dem ein applikationsspezifischer Intentionsdekoder aufsetzt [105, 156]. Das Besondere an diesem Verfahren ist, dass die Auswertungen auf der akustischen Ebene vollständig in den Erkennungsprozess eingebunden sind, wodurch die Suche auf der semantischen und der syntaktischen Ebene auf die darunterliegende phonetische Ebene ausgeweitet wird. Das Verfahren ist zwar auf aktuellen Systemplattformen noch nicht ganz in Echtzeit durchführbar, erlaubt auf der anderen Seite aber eine äußerst robuste Erkennung auch von komplexen Äußerungen. Eine Einschränkung besteht darin, dass das verwendete Vokabular a priori bekannt sein muss.

Die alternativ verwendbare zweite Erkennerkomponente basiert auf einem zweistufigen Ansatz, dem ebenfalls ein applikationsspezifischer Intentionsdekoder nachgeschaltet wird [143, 156]. Auf der akustischen Ebene liefert ein entsprechendes Modul die erkannten Wortketten inklusive der Konfidenzmaße für die einzelnen Wörter. Die semantisch-syntaktische Ebene verarbeitet diese Ergebnisse nach einem signalgetriebenen, hauptsächlich regelbasierten Verfahren, wobei auftretende Ambiguitäten nach einem Satz von vordefinierten Wahrscheinlichkeiten aufgelöst werden. Der Ansatz erlaubt die zusätzliche Auswertung von domänenspezifischem Wissen, die Berücksichtigung einer Dialog-Historie und kann in gewissem Maße auch mit Wörtern außerhalb des aktiven Vokabulars umgehen. Verglichen mit dem ersten Ansatz ist die Erkennungsleistung zwar etwas geringer, dafür aber in Echtzeit möglich. Die beiden Ansätze können darüber hinaus auch in einem einzigen System integriert werden, welches situationsspezifisch zwischen den jeweiligen Alternativen umschalten kann.

Videobasierte Erkennung dynamischer Hand- und Kopfgesten

In der zwischenmenschlichen Kommunikation wird die Sprache oftmals durch Gesten ergänzt. Für Benutzer, die durch spezielle Behinderungen in ihren Interaktionsmöglichkeiten eingeschränkt sind, stellt die Gestik zudem eine mächtige Alternative zu taktilen oder sprachbasierten Eingabeformen dar. Das in den Demonstratoren verwendete Erkennermodul baut auf einem von Morguet entwickelten Verfahren zur videobasierten Erkennung dynamischer Handgesten auf [107]. Der Erkennungsprozess basiert auf der Auswertung von Hidden-Markov-Modellen (HMMs), die für die zeitliche Segmentierung und die Klassifikation der einzelnen Bilder in einem kontinuierlichen Video-Stream eingesetzt werden.

Die Bilder werden von einer Standard CCD-Kamera aufgenommen und in das *YUV* Format umgewandelt. Um die Hand vom Hintergrund zu trennen, berechnet der Algorithmus auf Basis eines applikations- und situationsspezifischen Farbhistogramms ein Binärbild, das ausschließlich die Handform enthält. Unter definierten Beleuchtungsbedingungen lassen sich mit Hilfe von zusätzlichen Aufbereitungsverfahren optimale Segmentierungsergebnisse erzielen [108]. Die Bildsequenzen werden anschließend in Merkmalsvektoren transformiert. Für die Klassifikation werden semi-kontinuierliche HMMs mit 256 Prototypen und 25 Zuständen pro Modell verwendet. Die zeitliche Segmentierung (Spotting) der Gesten erfolgt nach einem einstufigen Verfahren, bei dem nach jedem Iterationsschritt die Ausgabewerte der einzelnen HMMs untersucht werden [109]. Aus den lokalen Maxima in den Spektren dieser Werte kann zum einen auf den Abschluss einer zusammenhängenden Interaktion geschlossen und zum anderen die Geste einer spezifischen Klasse zugeordnet werden. Das realzeitfähige System unterscheidet 41 verschiedene Handgesten mit einer durchschnittlichen Erkennungsrate von über 95%.

Für die Erkennung dynamischer Kopfgesten sind zwei eigenständige Verfahren entwickelt worden [96]. Einer extrem einfachen und schnellen regelbasierten Realisierung steht ein flexibles, etwas langsameres stochastisches Klassifikationssystem gegenüber. Bei beiden Ansätzen wird der Kopf durch eine Kombination von farb- und formbasierten Segmentierungsalgorithmen lokalisiert. Für die kontinuierliche Merkmalsextraktion benutzt das regelbasierte Modul ein Template-Matching der Nasenwurzel, während die stochastische Komponente zusätzliche Merkmale auswertet, die aus dem optischen Fluss abgeleitet sind. Beide Ansätze arbeiten in Echtzeit und unabhängig von den spezifischen Bildhintergründen. Insgesamt können sechs bis acht unterschiedliche Kopfgesten erkannt werden. Die Erkennungsrate liegt bei durchschnittlich 94% (regelbasiert) bzw. 97% (stochastisch), wobei der zweite Ansatz weitaus einfacher um weitere Gestentypen erweitert werden kann [12].

8.2 Multimodaler VRML Browser

In dem Einführungskapitel ist die historische Entwicklung von Benutzerschnittstellen bereits kurz thematisiert worden (siehe Abschnitt 2.2.3). Die Anzeigen der Feedbackinformationen beschränkten sich dabei jedoch ausschließlich auf zweidimensionale Darstellungsformen. Durch Virtual-Reality Systeme wird auch diese Beschränkung aufgehoben. Dreidimensionale und zweidimensionale Objekte werden gemeinsam in einem virtuellen Raum platziert und erlauben das Eintauchen des Benutzers in eine vollständig simulierte Umgebung (Immersion), welche bei entsprechend leistungsstarker Hardware die reale Arbeitsumgebung ersetzen kann. Durch die hohe Ähnlichkeit mit der gewohnten Umgebung können häufig auch unerfahrene Benutzer schnell und unkompliziert mit den Funktionalitäten des entsprechenden Systems umgehen [63].

Der erste Demonstrator ist ein solches Virtual-Reality Interface. Das System basiert auf dem VRML-Standard [72], mit dem sich interaktive dreidimensionale Szenarien auf einfache Weise beschreiben lassen. Die Modellierungen der Objekte werden in ASCII-Dateien abgelegt und können über beliebige Netze zwischen unterschiedlichen Plattformen ausgetauscht werden. In Analogie zu dem HTML-Standard bezeichnet man die programmtechnischen Komponenten zur Interpretation dieser Dateien ebenfalls als *Browser*. Ein VRML-Browser ist entweder eine eigenständige Applikation oder ein Plugin, das in herkömmliche HTML-Browser integriert werden kann. Anhang C enthält eine kurze Einführung in die wesentlichen VRML Sprachelemente, das zugrunde liegende Ausführungsmodell und die eingebauten Interaktionsparadigmen. Weiterführende Details zu VRML finden sich beispielsweise in [32, 58].

Multimodale Interaktionsparadigmen ermöglichen dem Anwender allgemein einen höheren Grad an Flexibilität und Natürlichkeit in der Mensch-Maschine-Kommunikation [120]. Durch die Beschränkung auf taktile Interaktionsformen lassen aktuelle VRML-Browser genau diese Vorteile vermissen. Aus diesem Grund wird im Folgenden ein speziell angepasster VRML-Browser vorgestellt, der zusätzlich über verschiedene Module zur Interpretation multimodaler Benutzereingaben verfügt. Dieser Demonstrator ist im Rahmen des lehrstuhlinternen Forschungsprojektes MIVIS[1] entwickelt und implementiert worden [10, 16, 17].

8.2.1 Erweiterte Systemmodule

Systemtechnisch basiert der MIVIS-Demonstrator auf dem bereits existierenden VRML-Browser Blaxxun Contact [28]. Dieser Browser wird auch von verschiedenen Software-Herstellern als zentrales Schnittstellenmodul für verteilte, graphische Anwendungen im Internet genutzt. Auf der Windows-Plattform unterstützt die integrierte Darstellungskomponente direkt verschiedene hardware-beschleunigte Grafik-Chipsätze und ermöglicht auf diese Weise fließende Bewegungen mit einem hohen Grad an Realismus. Im Gegensatz zu dem in den Benutzerstudien verwendeten FreeWRL-Browser ist in Blaxxun Contact der komplette Umfang der VRML97 Spezifikation [72] implementiert. Neben Gravitation und Kollisionsdetektion sind zudem noch besondere Effekte wie konfigurierbare Avatare, Feuer, Schneefall oder Partikel-Systeme enthalten.

Das angepasste Browser-Modul basiert zudem auf der in Kapitel 5 beschriebenen Architektur. Abbildung 8.1 zeigt die applikationsspezifischen Erweiterungen im Hinblick auf die Verarbeitung multimodaler Benutzereingaben. Die nachfolgenden Ausführungen beschränken sich zwar ausschließlich auf eine effiziente Umsetzung der Navigation in beliebigen virtuellen Welten, lassen sich aber in wesentlichen Teilen auch auf andere Interaktionsparadigmen übertragen. Die Anpassung des MIVIS-Systems an eine multimodal steuerbare Selektion und Manipulation dreidimensionaler Objektstrukturen wird ausführlich in [45, 136] behandelt.

Der Benutzer hat prinzipiell immer die Möglichkeit, über unterschiedliche Modalitäten mit dem VRML-Browser zu kommunizieren, wobei er vollkommen frei zwischen einer Vielzahl an haptischen und semantisch höherwertigen Eingabekanälen auswählen kann. Während einzelne Interaktionen mit Sprache oder Gestik von den jeweiligen Erkennermodulen direkt auf diskrete Aktionen abgebildet werden können, müssen die taktilen Interaktionen gesondert interpretiert werden. Die entsprechenden Eingabegeräte erzeugen in erster Linie quasi-kontinuierliche Bewegungsinformationen, die vor dem Hintergrund des aktuell eingestellten Navigationsmodus interpretiert werden müssen.

[1] MIVIS steht für Multimodal Interaction in VIrtual Scenarios

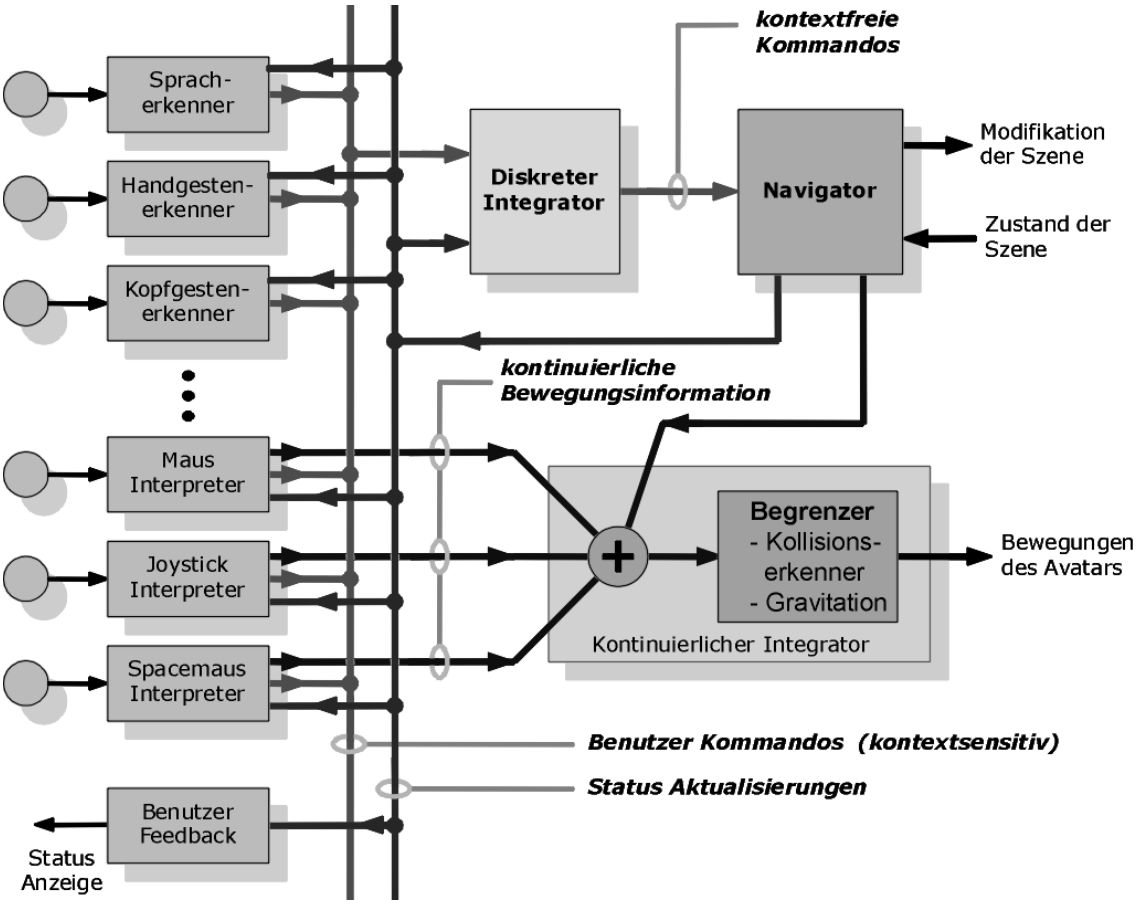


Abb. 8.1: Erweiterte Struktur des multimodalen VRML-Browsers im MIVIS-System

Umgekehrt fungieren ausgewählte taktile Manipulatoren gleichzeitig auch als Ausgabegeräte. Auf diese Weise kann dem Benutzer ein haptisches Feedback vermittelt werden, beispielsweise wenn ein virtueller Avatar mit einer vorgegebenen Geometrie in der Szene kollidiert. Diese Feedback-Information ist besonders für unerfahrene Systemnutzer wichtig, um sich in der virtuellen Umgebung besser orientieren zu können. In Abbildung 8.1 ist zudem noch ein eigenständiges Feedback-Modul angedeutet, über das weitere Status-Informationen an den Benutzer übertragen werden können. Das Feedback-Modul könnte beispielsweise durch eine konventionelle Benutzungsoberfläche realisiert werden, die in einem entkoppelten Darstellungsbereich den aktuellen Navigationsmodus, die Schrittweite, den Namen des aktiven Aussichtspunktes und andere relevante Informationen anzeigt.

Die multimodale Integrationskomponente wird systemtechnisch in drei Teilmodule aufgespaltet. Ein *diskreter Integrator* verarbeitet nach der in Kapitel 6 beschriebenen Vorgehensweise die Eingabedaten, die bereits durch entsprechende Konvertermodule in abstrakte Symbole transformiert worden sind. In dieser Phase werden auch auftretende Mehrdeutigkeiten im Hinblick auf redundante, komplementäre und rivalisierende Informationsanteile aufgelöst. Als Ergebnis liefert diese Komponente eine semantisch abgeschlossene Navigationsanweisung.

Über eine TCP/IP-Verbindung werden die Kommando-Sequenzen als Wörter einer kontextfreien Grammatik an das nachfolgende *Navigator*-Modul übertragen. Der Navigator verwaltet

den Navigationsmodus, die Navigationsgeschwindigkeit, die Liste der aktuellen Aussichtspunkte und das eingestellte Drehzentrum für rotatorische Bewegungen. Das Modul verfügt zudem über einen Speichermechanismus, mit dessen Hilfe einzelne Bewegungsschritte rückgängig gemacht werden können. Veränderungen der einzelnen Navigationsparameter werden sowohl dem diskreten Integrator als auch den angeschlossenen Eingabegeräten rückgemeldet.

Der *kontinuierliche Integrator* kombiniert die diskreten Anweisungen des Navigators mit den kontinuierlichen Daten der haptischen Interpreter und setzt sie in entsprechende Bewegungen des virtuellen Avatars um. Dazu werden die einzelnen Bewegungsinformationen in Form von Geschwindigkeitsvektoren komponentenweise addiert und in die sechs Freiheitsgrade des dreidimensionalen Raums umgerechnet. In Abhängigkeit von dem eingestellten Navigationsmodus, der Gravitationssimulation und der Kollisionsdetektion können einzelne Freiheitsgrade temporär ausgeblendet und die resultierenden Bewegungen unterdrückt werden.

8.2.2 Abstraktes Funktionsspektrum

Die Funktionalität des angepassten VRML-Browsers kann vollständig in Form einer kontextfreien Grammatik angegeben werden. Dabei werden im Wesentlichen vier zusammenhängende Aktionskategorien unterschieden. Bewegungskommandos (`MOV`) enthalten native Navigationsaktionen, Positionskommandos (`POS`) beschreiben referenzierende Bewegungen, Kontrollkommandos (`CTRL`) spezifizieren Parameteränderungen und Statusaktionen (`SFB`) enthalten diverse Feedback-Informationen für den Benutzer. Auf die einzelnen Kategorien wird im Folgenden näher eingegangen. Für die ersten drei Kategorien beinhaltet Abbildung 8.2 eine zusammenhängende Darstellung der grammatikalischen Regeln in BNF.

Die einfachen Navigationsaktionen beschreiben eine abgeschlossene Bewegung der Szenenansicht in der Form `<mode> <type> <direction>`, denen optional noch ein weiteres Terminalsymbol `start` vorangestellt werden kann. Durch diese drei elementaren Variablen wird jeweils der Navigationsmodus, die Art der Bewegung und die Richtung spezifiziert. Ein Kommando entspricht normalerweise genau einem Bewegungsschritt. Die Länge dieses Schrittes kann über ein entsprechendes Kontroll-Kommando eingestellt werden. Mit dem Präfix `start` wird die Bewegung in der vorgegebenen Art und Weise solange durchgeführt, bis der Integrator entweder ein `stop` erkennt oder der Benutzer durch eine weitere Aktion eine neue Bewegung initiiert. Im praktischen Betrieb des Systems kommt es häufig vor, dass eine Benutzeraktion in Bezug auf die zugrunde liegende kontextfreie Grammatik ein unvollständiges Wort darstellt. Diese Daten werden im Integrator zwischengespeichert und im Kontext der vorherigen bzw. nächsten Bewegungsaktionen interpretiert.

Positionsorientierte Kommandos spezifizieren verschiedene Bewegungen jeweils bezogen auf einen dedizierten Bezugspunkt. Bei einigen Aktionen wird dieser Punkt in dem entsprechenden Kommando mitgeliefert, bei anderen muss der Integrator die Ergänzung durch Informationsanteile aus anderen Modalitäten explizit vornehmen. Die drei Kommandoarten `moveto`, `orientto` und `beamto` drücken den Wunsch des Benutzers aus, sich auf einen spezifischen Punkt zuzubewegen bzw. seine Blickrichtung auszurichten. Orientierungen können sowohl in der VRML Notation über eine Drehachse und den dazugehörigen Winkel als auch über die Angabe eines Punktes spezifiziert werden, durch den die optische Achse der virtuellen Kamera gehen soll. Die meisten Aktionen können mit dem Parameter `nearto` ergänzt werden, der bei den Bewegungen die existierenden geometrischen Beschränkungen berücksichtigt.

```
# --- Grundlegende Navigationskommandos -------------------------------------
<MOV>  ::= start <MCMD> | <MCMD> | stop
<MCMD> ::= walk <WLK> | fly <FLY> | examine <EXM>
<WLK>  ::= trans <ADIR> | rot <LR>
<FLY>  ::= trans <ADIR> | rot <LRUD> | roll <LR>
<EXM>  ::= trans <MDIR> | rot <LRUD> | roll <LR>
<ADIR> ::= <MDIR> | <DIAG>
<MDIR> ::= left | right | up | down | forward | backward
<DIAG> ::= lfwd | rfwd | lbwd | rbwd
<LRUD> ::= left | right | up | down
<LR>   ::= left | right

# --- Positionsbezogene Navigationskommandos --------------------------------
<POS> ::= gothere | lookat | exacenter
<POS> ::= indicated geometry fff | indicated pos fff | indicated ori fff
<POS> ::= indicated posori ffffff | orientto pos fff
<POS> ::= moveto pos fff | moveto nearpos fff | moveto ori ffff
<POS> ::= moveto posori ffffff | beamto pos fff | beamto nearpos fff

# --- Zusätzliche Kontrollkommandos -----------------------------------------
<CTRL> ::= ctrl <CCMD> | repeat | undo | quit
<CCMD> ::= mode <SMOD> | stepsize <SPS> | viewpoint <VIP> | sendstatus <SVAR>
<CCMD> ::= light <OOT> | collision <OOT> | gravity <OOT> | straighten | balance
<SMOD> ::= set <MVAR> | restrict <x y z yaw pitch roll>
<MVAR> ::= walk | fly | examine
<SPS>  ::= inc | dec | reset | set f
<VIP>  ::= prev | next | reset | set f
<OOT>  ::= on | off | toggle
<SVAR> ::= all | sstepsize | smode | slight | scollision | sgravity
```

Abb. 8.2: Abstrakte Beschreibung der möglichen Aktionen im MIVIS-System; die Regeln sind in BNF dargestellt, wobei das ´f´ für eine Parameterangabe in Form einer reellen Zahl steht; Meta-Variablen sind in spitze Klammern eingeschlossen und das Raute-Zeichen zeigt eine Kommentar-Zeile an

Neben den unterschiedlichen Bewegungsaktionen kann der Benutzer über eine Reihe von Kontroll-Kommandos die Einstellungen im Navigatormodul des VRML-Browsers verändern. Dazu gehören beispielsweise die Parametrisierung der Beleuchtungsbedingungen, die Ansteuerung vordefinierter Aussichtspunkte, das Ein- und Ausschalten von Gravitation und Kollisionsdetektion und die Umstellung der Navigationsmodi. Darüber hinaus kann die Schrittweite für einzelne Bewegungsschritte an die Leistungsfähigkeit des laufenden Systems angepasst werden und jeweils die gespeicherten Grundeinstellungen zurückgesetzt werden.

8.2.3 Anpassbare Virtual-Reality Applikationen

Durch die Möglichkeit, einzelne Interaktionsstile an die Anforderungen einer speziellen Applikation oder die Profile einer bekannten Benutzergruppe anpassen zu können, eröffnet sich ein breites Spektrum für VRML-basierte Virtual-Reality Systeme. Bei den momentan verfügbaren VRML-Browsern besteht jedoch eine starre, direkt im Browser implementierte Verbindung

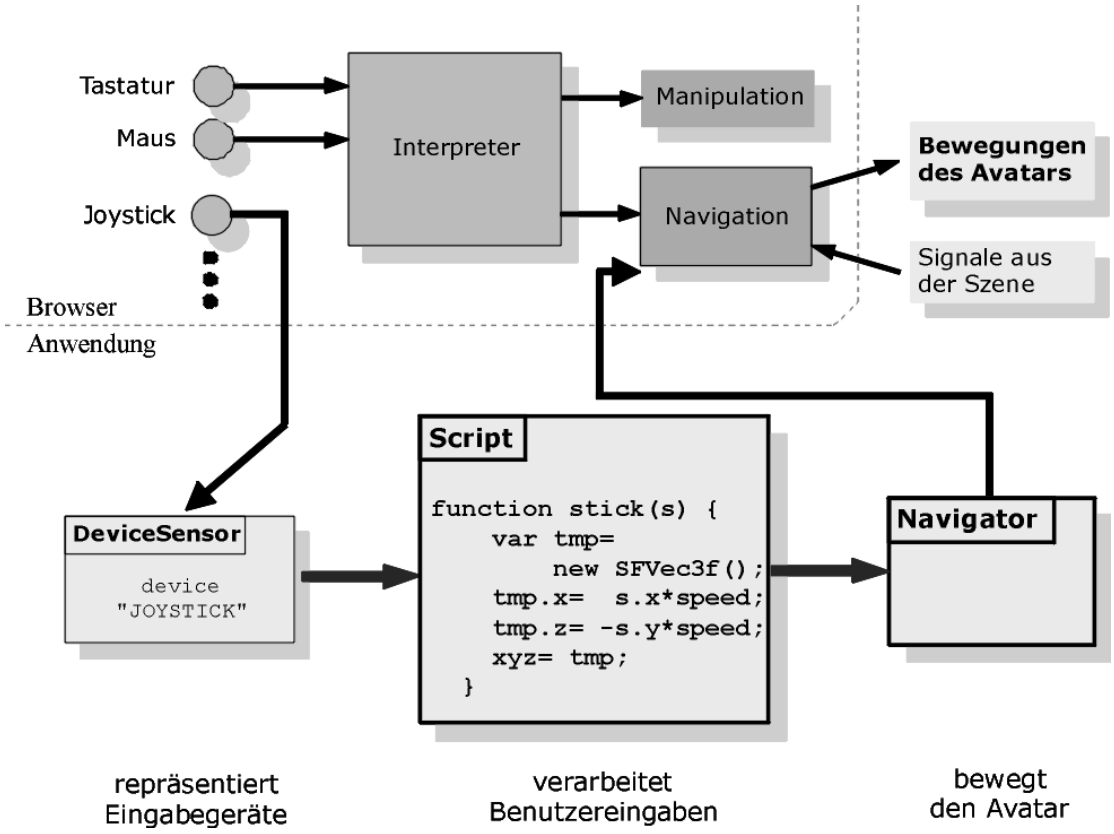


Abb. 8.3: Strukturelle Elemente für eine angepasste, VRML-basierte Virtual-Reality Applikation

zwischen den einzelnen Eingabegeräten und den damit realisierbaren Navigationsbewegungen. In der MIVIS-Architektur wird diese Verbindung mit Hilfe des ROUTE-Mechanismus aufgebrochen und durch die Anwendung umgeleitet. Die dafür benötigten technischen Grundlagen werden in Abschnitt C.3 kurz erläutert.

Die Einführung von zwei neuen VRML Sprachkonstrukten erlaubt dem Programmierer, auf einfache Weise anpassbare 3D Applikationen zu erzeugen, ohne die Implementierung des Browsers ändern zu müssen. Der `DeviceSensor`-Knoten erlaubt eine formal einheitliche Modellierung beliebiger Eingabegeräte. Über den PROTO-Mechanismus kann innerhalb einer Szenenbeschreibung auf das volle Funktionsvokabular der entsprechenden Geräte zugegriffen werden. Durch diese zentrale Schnittstelle kann Verarbeitung multimodaler Benutzereingaben auch durch externe Module erfolgen.

Das zweite neu eingeführte Sprachkonstrukt, der `Camera`-Knoten, erlaubt einen abstrakten Zugriff auf die Steuerung der Navigationsfunktionen. Im Gegensatz zu der umständlichen, in VRML eingebauten Viewpoint-Navigation entfällt bei der Benutzung des `Camera`-Knoten die exakte Spezifikation der sechs-dimensionalen Geschwindigkeitsvektoren. Der Programmierer kann auf einfache Weise verschiedene Pfade durch die virtuelle Szene definieren, das Rotationszentrum für den Untersuchungs-Modus einstellen, zu verschiedenen Ansichten wechseln und die einzelnen Navigationsparameter kontrollieren. Das Zusammenspiel von `DeviceSensor` und `Camera` im Hinblick auf die Umleitung der Benutzereingaben ist in Abbildung 8.3 anhand eines Joysticks prototypisch dargestellt.

8.3 Infotainment-Applikationen im Automobil

Die in Kapitel 5 vorgestellte Architektur bildet auch die Plattform für eine Reihe von Demonstratoren zur multimodalen Bedienung von verschiedenen Kommunikations- und Informationssystemen in einer Automobilumgebung. Auf zwei ausgewählte Systeme, die im Rahmen von kooperativen Forschungsprojekten mit unterschiedlichen Industriepartnern erarbeitet worden sind, soll im Folgenden näher eingegangen werden. Darüber hinaus wird ein alternativer Entwurf für eine Benutzungsoberfläche mit dreidimensionalen Darstellungselementen präsentiert, der eine potenzielle Verbindung zwischen der Nutzung der VRML-Technologie und der Realisierung von automotiven Mensch-Maschine-Systemen aufzeigt.

8.3.1 Beispiele für multimodale Systemschnittstellen

Bimodale Steuerung eines MP3-Players (SOMMIA)

Der erste Demonstrator ist aus einer bilateralen Kooperation mit der Firma SiemensVDO AG entstanden. Die Projektbezeichnung SOMMIA steht abkürzend für *Sprachorientiertes MMI im Automobil*. Vor dem Hintergrund restriktiver Randbedingungen bietet das Testsystem ein generisch erweiterbares Interaktionskonzept für die Bedienung eines MP3-Players [81]. In Bezug auf den Bedienkomfort müssen gleichzeitig hohe Anforderungen erfüllt werden. Die grundlegenden Funktionalitäten des Systems sollen im Rahmen eines simulierten Showroom-Szenarios auch ohne vorherige Einweisung intuitiv zugänglich sein. Als Leitmodalität kommt der Verwendung der Sprache dabei eine besondere Bedeutung zu.

Die anvisierte Zielplattform ist ein kompaktes Gerät, das als Alternative zu dem Audiomodul des Erstausstatters in den genormten Radioschacht eines beliebigen Kraftfahrzeugs eingebaut werden kann. Als geometrische Anforderung wird die effektive Anzeigefläche auf ein zweizeiliges Dot-Matrix-Display mit maximal 16 Zeichen pro Zeile beschränkt. Zusätzlich kann eine begrenzte Menge an LEDs verwendet werden. Die Spracherkennung basiert auf dem von der Firma Philips entwickelten Ganzworterkennermodul HelloIC [62]. Das aktive Vokabular ist pro Ebene auf 30 bis 50 Wörter beschränkt, kann aber dynamisch umgeladen werden. Der Erkennungsprozess wird benutzerinitiiert durch den Druck einer Push-to-Talk Taste aktiviert und systemseitig durch einen Time-Out Mechanismus beendet. Alternativ zu der Sprache können die Grundfunktionen des Systems auch über verschiedene Tasten bedient werden. Als Rückkopplung erhält der Benutzer sowohl ein akustisches als auch ein visuelles Feedback.

In ausgedehnten Benutzerstudien ist zunächst für unterschiedliche Bediensituationen jeweils eine Auswahl der häufigsten Kommandos bestimmt worden [98]. Auf Basis dieses Vokabulars sind in dem Demonstrator zwei Dialogkonzepte umgesetzt. Das *Tree-Structure-Principle (TSP)* lehnt sich an klassische Suchtopologien an, bei denen die Aktionen streng hierarchisch in Menüebenen organisiert sind. Der Wechsel zu einem anderen Knoten in der zugrunde liegenden Graphenstruktur impliziert eine Umladung des aktiven Vokabulars. Im Gegensatz dazu basiert das *Action-Object-Principle (AOP)* auf einer abstrakten Grammatikstruktur, bei der in einer fest vorgegebenen Reihenfolge zuerst eine Aktion und anschließend das Zielobjekt dieser Aktion und eventuell zusätzliche Parameter angegeben werden müssen. Das aktive Vokabular wird bei diesem Ansatz in Abhängigkeit von der erkannten Aktion nachgeladen.

Gemischt-modale Steuerung von Infotainment-Systemen (FERMUS)

Der zweite Demonstrator ist im Rahmen einer mehrjährigen Forschungskooperation des Lehrstuhls für Mensch-Maschine-Kommunikation mit der BMW Group, der DaimlerChrysler AG und der SiemensVDO AG entstanden [82, 83, 84]. Die Projektbezeichnung FERMUS steht abkürzend für *Fehlerrobuste Multimodale Sprachdialoge*. Hauptziel dieses Projektes ist die Identifikation und Bewertung von verschiedenen Fehlerauswertungsstrategien bei der Bedienung von Informations- und Kommunikationssystemen im Kraftfahrzeug. Im Gegensatz zum SOMMIA-Demonstrator kann der Benutzer in einer multimodalen Systemumgebung vollkommen frei aus einem Spektrum an unterschiedlichen taktilen und semantisch höherwertigen Interaktionsmöglichkeiten wählen.

Als Testplattform dient das bereits in Abschnitt 3.4.1 beschriebene Infotainment-System mit einem leicht erweiterten Funktionsumfang. Wesentliches Charakteristikum des Demonstrators ist ein leistungsfähiges Fehlermanagementmodul, das vollständig in den multimodalen Integrationsprozess eingebunden ist. Auf Basis einer theoretisch fundierten Fehlerdefinition können sowohl intern generierte als auch von außen imponierte problematische Situationen adäquat behandelt werden [9]. Grundsätzlich wird dabei zwischen benutzerinitiierten und systeminitiierten Auflösungsdialogen unterschieden.

Speziell in einer multimodalen Bedienumgebung kommt der effektiven Behandlung von Fehlerkennungen eine besondere Bedeutung zu. Widersprüchliche Informationsanteile aufgrund konkurrierender Ergebnisse werden in erster Näherung durch eine adaptive Umschaltfunktion aufgelöst, die kontextspezifisch ausgewählte Modalitäten priorisiert. Unsichere Klassifikationsergebnisse werden zudem durch einen mehrstufigen Rückfragedialog disambiguiert, wobei der Benutzer immer als höchste Entscheidungsinstanz fungiert. Auf diese Weise können Fehlinterpretationen bereits frühzeitig erkannt und vermieden werden [14].

Der FERMUS-Demonstrator passt die Dialogstrategie kontinuierlich an die jeweiligen Benutzereingaben an. Grundlage für diesen Adaptionsprozess sind eine Vielzahl von direkten Messdaten und Meta-Informationen aus den bilateralen Klärungsdialogen [94, 97]. In diesem Zusammenhang spielt die Auswertung von emotionalen Benutzerreaktionen eine Schlüsselrolle. Neben einer dynamischen Anpassung der zugrunde liegenden Benutzermodelle werden sie vor allem in Stresssituationen als zusätzliche Steuergröße für den Integrationsprozess verwendet [95, 145]. Auf diese Weise tragen sie insgesamt zu einer Verbesserung der Systemakzeptanz bei. Die automatische Erkennung der Benutzeremotionen basiert auf der Auswertung von prosodischen und semantischen Merkmalen sprachlicher Äußerungen [144, 146].

8.3.2 Alternative Informationspräsentation

Die meisten der kommerziell verfügbaren automotiven Komfort-Applikationen weisen erhebliche Mängel in Bezug auf eine einfache und robuste Bedienbarkeit auf. Dies liegt zum einen an dem kontinuierlich steigenden Funktionsspektrum und zum anderen an der Beschränkung auf rein taktile Interaktionsmöglichkeiten. Industrielle Produktverbesserungen konzentrieren sich weitestgehend auf eine Optimierung der Eingabegeräte, eine klar strukturierte Informationspräsentation oder eine ergonomisch günstige Platzierung der zentralen Bedienelemente [126, 174]. In verschiedenen Forschungsprototypen wird zusätzlich die Nutzung von Sprache und Gestik als alternativ verwendbare Eingabemodalitäten untersucht [25]. Die visuelle Aufbereitung der Informationen ist jedoch strikt auf zwei Dimensionen limitiert.

Abb. 8.4: MP3-Modus, Telefon-Modus und WAP-Modus des VIRIS-Demonstrators

Aus diesem Grund soll mit einem weiteren Demonstrator eine Verbindung zwischen dem multimodalen VRML-Browser und den Infotainmentsystemen aus dem letzten Abschnitt hergestellt werden. Das Ziel besteht darin, unter Einbezug der speziellen Randbedingungen in der Automobilumgebung die Vorteile von dreidimensionalen Desktop-Interfaces in die Automobildomäne zu übertragen und dadurch eine Verbesserung in der Akzeptanz und der Bedienbarkeit zu erreichen. Eine wichtige Anforderung bei der Entwicklung besteht darin, dass sich die neue Schnittstelle vollständig in die existierende Basisarchitektur integrieren lässt und alternativ zu der FERMUS-Oberfläche verwendet werden kann. Die innovativen Aspekte konzentrieren sich auf eine Verbesserung der visuellen Informationsaufbereitung [13].

Die Projektbezeichnung VIRIS steht abkürzend für *Virtual-Reality Infotainment System*. Im Wesentlichen verfügt der Demonstrator über vier zentrale Betriebsarten. Während sich der MP3-, der Radio- und der Telefon-Modus systemtechnisch direkt auf den Funktionalitäten des FERMUS-Demonstrators abbilden lassen, ist mit einem eigenständigen WAP-Modul auch ein weiterer Betriebsmodus hinzugekommen. Ein weiterer Unterschied besteht darin, dass der explizite Kontroll-Modus aufgehoben ist und die Lautstärkeregelung statt dessen in allen Modi direkt angesprochen werden kann.

Abbildung 8.4 enthält drei verschiedene Momentaufnahmen des VIRIS-Demonstrators. Die Darstellung eines Betriebsmodus nimmt immer die komplette Anzeigefläche ein. Ein Wechsel zwischen den verschiedenen Anwendungen ist mit Hilfe der Icon-Leiste am unteren Bildschirmrand möglich. Um eine möglichst konsistente Darstellung beizubehalten, sind die Ansichten mit Ausnahme des WAP-Browsers jeweils aus identischen Elementen aufgebaut. Funktional vergleichbare Objekte befinden sich immer an der gleichen Position. Eine Unterscheidung der Modi ist zum einen über eine entsprechende Farbkodierung und zum anderen über entsprechende Symbole realisiert.

Als zentrales Element befindet sich eine Listendarstellung in der Mitte des Bildschirms. Je nach Modus können hier Liedtitel, Radiosender oder Telefonnummern angezeigt und ausgewählt werden. Die Listenanzeige hat die plastische Form einer Zylinderoberfläche. Vertikale Bewegungen in der Liste können durch eine Drehung des Zylinders visualisiert werden. Das linke Drehrad ermöglicht Lautstärkeänderungen und das rechte Drehrad dient zur Steuerung des Listenfokus. Die Pfeile über bzw. unter den Drehrädern zeigen die relativen Positionen an. Eine Erweiterung stellt die Statusleiste am oberen Bildschirmrand dar. Neben der Ausgabe wichtiger Informationen ermöglicht sie einen reduzierten Direktzugang zu den wichtigsten Funktionen der entsprechenden Anwendung. Für Experten ermöglicht die Statusleiste einen schnellen Themenwechsel und bei extensiver Nutzung darüber hinaus auch einen modusübergreifenden Parallelbetrieb der angeschlossenen Geräte.

Der FERMUS und der VIRIS Demonstrator sind im Rahmen einer entwicklungsbegleitenden Benutzerstudie qualitativ verglichen worden. Allgemein wird keins der beiden Systeme eindeutig bevorzugt. In Bezug auf ausgewählte Designelemente und den mit der Bedienung assoziierten Spaßfaktor erhält die dreidimensional aufbereitete Darstellung signifikant bessere Bewertungen. Auf der anderen Seite besteht mehrheitlich der Wunsch nach einer einfachen und klar strukturierten Benutzungsoberfläche, die nicht vom eigentlichen Fahren ablenkt. Weiterführende Details zu den Ergebnissen finden sich in [13, 125].

8.4 Präoperative Tumoranalyse

Neben den beiden bisher beschriebenen Anwendungsgebieten Desktop Virtual-Reality Browser und automotive Infotainment-Systeme ist im Rahmen einer internen Kooperation mit der nuklearmedizinischen Klinik des Universitätskrankenhauses Rechts der Isar (NRI) auch in der Medizindomäne ein Demonstrator auf Basis der multimodalen Systemarchitektur entstanden. Ziel dieser Arbeit ist die Entwicklung einer intuitiv bedienbaren Benutzungsoberfläche, um in einem präoperativen Schritt Tumore im menschlichen Körper visualisieren und analysieren zu können. Auf diese Weise können chirurgische Eingriffe besser vorbereitet, alternative Interventionsstrategien geplant bzw. zu Ausbildungszwecken vollständig simuliert werden.

Die Untersuchungsgrundlage liefert ein spezielles, nuklearmedizinisches Schnittbildverfahren, das im Fachjargon als Positronen-Emissions-Tomographie (PET) [168] bezeichnet wird. Mit dieser Technologie können nicht-invasiv verschiedene funktionelle und biochemische Vorgänge im menschlichen Organismus dargestellt werden. Radioaktiv markierte Substanzen (Tracer) werden dem Patienten im Vorfeld der Untersuchung verabreicht. Diese Substanzen reichern sich vor allem in Gewebestrukturen mit speziellen biologischen Eigenschaften wie beispielsweise auch Krebszellen an. Mit einem Positronen-Tomographen werden die von der Tracer-Materie emittierten Gammaquanten registriert, entsprechend aufbereitet und zu koronalen, sagittalen und axialen Schnittbildern verarbeitet.

Für die medizinische Forschung ist es wichtig, Aktivitätsverteilungen in unterschiedlichen Organen quantitativ bestimmen zu können. In einer multimodalen Systemumgebung erlaubt der neu konzipierte Demonstrator eine semi-automatische Markierung und Analyse relevanter Gewebestrukturen. Darüber hinaus können auf Basis eines dreidimensional aufbereiteten VRML-Datensatzes einzelne Elemente selektiert und für eine genauere Betrachtung manipuliert werden. Aus dem zeitlichen Verlauf der Tracer-Aktivitäten während der Untersuchung lässt sich unter anderem auch der Glukosestoffwechsel und die Gewebeperfusion des Patienten berechnen und direkt in der Simulationsumgebung anzeigen. Details zu dem Demonstrator und Ergebnisse zu der Evaluierung des Systems in einer kleinen Benutzerstudie mit ausgewählten Experten finden sich in [57, 103].

KAPITEL 9

Diskussion und Ausblick

Multimodale Mensch-Maschine-Systeme repräsentieren zur Zeit die höchste Evolutionsstufe in Bezug auf die Entwicklung von Benutzerschnittstellen. Verglichen mit klassischen, unimodalen Bedienphilosophien können Informationen auf mehreren, unterschiedlichen Kanälen bilateral zwischen dem Benutzer und dem System ausgetauscht werden. Durch die konsequente Umsetzung eines multimodalen Interaktionsparadigmas lassen sich kürzere Einarbeitungsphasen realisieren sowie die Fehlersicherheit und die Akzeptanz technischer Anwendungen erhöhen.

In der vorliegenden Arbeit wurde auf Basis einer umfangreichen Benutzerstudie ein generisches Konzept für die Verarbeitung multimodaler Benutzereingaben vorgestellt, anhand realer und fiktiver Bedienszenarien evaluiert und vor dem Hintergrund ausgewählter Anwendungen in separaten Demonstratoren prototypisch umgesetzt. Dieses Kapitel geht noch einmal auf die zentralen Phasen des Entwicklungsprozesses ein. Dabei werden die einzelnen Themengebiete jeweils zusammenfassend diskutiert und ein kurzer Ausblick zu potenziell anschließenden Arbeitspaketen gegeben. Eine ausführliche Beschreibung der Vorgehensweisen und der spezifischen Ergebnisse ist in den entsprechenden Kapiteln enthalten.

Nach dem Prinzip eines benutzerzentrierten Entwicklungsprozesses wurde zunächst eine Usability-Untersuchung zur Evaluierung multimodaler Interaktionsparadigmen durchgeführt. Gegenstand dieser Benutzerstudie war die Auswertung von Eingabesequenzen mittels Touchscreen, Tastatur, Handgestik, Kopfgestik und Sprache in zwei verschiedenen Applikationsdomänen. Während sich das erste Szenario auf die Navigation in dreidimensionalen, virtuellen Welten in einer desktop-orientierten Arbeitsumgebung konzentrierte, stand beim zweiten Szenario die Bedienung von verschiedenen Diensten in einer Automobilumgebung im Mittelpunkt. Die Probanden mussten nach einem strukturell identischen Versuchsplan vorgegebene Aufgaben bearbeiten und dabei jeweils verschiedene Modalitätenkombinationen anwenden.

Die Ergebnisse stellen keine allgemein gültigen Aussagen dar, sondern müssen vor dem Hintergrund der existierenden Randbedingungen sorgfältig interpretiert werden. Es hat sich gezeigt, dass die Gruppe der Anfänger im Vergleich zu erfahrenen Nutzern das Spektrum der Eingabemöglichkeiten wesentlich stärker ausnutzten. In Bezug auf die funktionale Zusammensetzung multimodaler Eingaben wurden hauptsächlich redundante und komplementäre Interaktionen beobachtet. Die modalitätenspezifischen Interaktionszeiten lagen in beiden Anwendungen sehr dicht beieinander, wohingegen die intermodalen Zeitrelationen und Überlagerungsmuster unter den Versuchsteilnehmern zum Teil deutlich variierten. Insgesamt dominierten die Sprache im Automobilbereich und taktile Interaktionen in der Virtual-Reality-Umgebung.

Trotz der strukturellen Unterschiede in den beiden Versuchsaufbauten konnten aus den verschiedenen quantitativen und qualitativen Resultaten fundamentale Anforderungen für das Design multimodaler Interaktionssysteme abgeleitet werden. Sämtliche Versuchsteilnehmer haben explizit betont, die freie Wahl zwischen mehreren Eingabekanälen gegenüber einem fest vorgeschriebenen, wenig flexiblen Interaktionsstil deutlich zu bevorzugen. Auf diese Weise können sie in Abhängigkeit von der aktuellen Bediensituation und ihren individuellen Präferenzen eine subjektiv optimale Strategie zur Kommunikation mit der Maschine verfolgen.

Das aufgezeichnete Datenmaterial bietet noch weiteres Potenzial. Aufgrund des enorm hohen Nachbearbeitungssaufwands sind bisher nur die Zeitrelationen von ausgewählten Benutzern und prototypischen Bediensituationen ausgewertet worden. Darüber hinaus könnten durch zusätzliche Benutzerstudien mit weiteren Modalitäten und in anderen Umgebungen, wie beispielsweise einer Untersuchung zum multimodalen Eingabeverhalten im Umgang mit ubiquitären Endgeräten, die bisher gesammelten Erfahrungen erweitert werden.

Ausgehend von einer detaillierten Analyse existierender Architekturansätze und den Ergebnissen aus den Benutzerstudien wurde ein Modell für eine generische, multimodale Basisarchitektur entwickelt. Die Eingaben des Benutzers werden über verschiedene Sensoren erfasst, im aktuellen Bedienkontext interpretiert und auf eine entsprechende Systemaktion abgebildet, die dem Anwender dann über verschiedene Ausgabekanäle in multimedialer Form angezeigt wird. Charakteristisch ist der streng modulare Aufbau mit einer zentralen Integrationseinheit, die vollständig über eine externe Wissensbasis parametrisiert werden kann. Im direkten Vergleich zu einem verteilten Agentensystem mit einer dezentralen Speicher- und Verarbeitungsstruktur lässt sich der Aufwand für die Koordination der einzelnen Systemkomponenten auf ein Minimum reduzieren. Ein Nachteil besteht darin, dass die volle Funktionsfähigkeit des Gesamtsystems nur bei einem korrekten Betrieb des Integratormoduls gewährleistet werden kann.

Der zugrundeliegende Kommunikationsformalismus basiert auf einer abstrakten, paramodalen Modellierung. Die verschiedenen, normalerweise in proprietären Formaten vorliegenden multimodalen Benutzereingaben, Kontextinformationen und Applikationsparameter werden mittels gerätespezifischer Konvertermodule in eine einheitliche Darstellung transformiert und auf die Elemente einer kontextfreien Sprache abgebildet. Im Hinblick auf den Integrationsprozess repräsentieren die Terminalsymbole der dazugehörigen Grammatik die kleinste, semantisch abgeschlossene Basiseinheit. Durch die formale Beschreibung ist der Kern der Architektur vollkommen unabhängig von der Anzahl und dem Typ der verwendeten Ein- und Ausgabemodule.

Die Interpretation multimodaler Informationsanteile ist im Allgemeinen ein hochgradig kontextspezifischer Verarbeitungsprozess. Zur Lösung dieses Problems wurde ein zweistufiger Ansatz verfolgt, der auf einer symbol-orientierten Beschreibung der zur Verfügung stehenden Daten aufsetzt. In der ersten Phase (*Spotting*) werden die auftretenden Ereignisse strukturell analysiert. Das Ziel besteht darin, zeitlich und semantisch assoziierte Informationsanteile zu identifizieren und in ein gemeinsames Integrationsintervall zu übertragen. In der zweiten Phase (*Fusion*) werden die multimodalen Daten zueinander in Beziehung gesetzt und eine entsprechende Systemreaktion erzeugt. Die beiden Phasen der Integration sind funktional entkoppelt und können als getrennte Prozesse sequentiell abgearbeitet werden.

Das Fusionsmodul selbst besteht wiederum aus mehreren, sich partiell überlappenden Verarbeitungsebenen, die nacheinander durchlaufen werden, um die jeweiligen Integrationsergebnisse sukzessive zu verbessern. In einem maßgeblich datengetriebenen Prozess steigt das Abstraktionsniveau, auf dem die einzelnen Informationsanteile verarbeitet werden, von der reinen Symbol-

verarbeitung über die Kontextevaluation bis hin zu einem dedizierten Fehlermanagement kontinuierlich an. In den höheren Ebenen wird zudem wesentlich intensiver auf domänenspezifisches Zusatzwissen zugegriffen. Um einen Basisbetrieb sicherzustellen, sind die Module der einzelnen Verarbeitungsschichten auch ohne die jeweils übergeordneten Ebenen zumindest eingeschränkt funktionsfähig. Die domänenspezifische Konfiguration des Integrators erfolgt extern über eine standardisierte Schnittstelle und ist direkt zur Laufzeit möglich.

Für das Spotting wurde ein deterministisches, regelbasiertes Verfahren eingesetzt. Der zugrundeliegende Regelkorpus besteht aus einer Menge von allgemeinen Abhängigkeitsrelationen und zusätzlichen, domänenspezifischen Erweiterungen. Im Rahmen einer Portierung des Systems kann ein Großteil der Regeln direkt übernommen werden. Bei den beiden betrachteten Anwendungsszenarien hat sich die Verwendung eines regelbasierten Ansatzes als äußerst vorteilhaft erwiesen. Die Auswertung der a priori formulierten Regeln stellt ein probates Mittel dar, um Ambiguitäten im Hinblick auf die temporalen und semantischen Beziehungen zwischen den einzelnen Informationsquellen aufzulösen. Im Vergleich zu statistischen Verfahren fällt zudem die Notwendigkeit zum Sammeln geeigneter Trainings- und Testmuster weg.

Zeitlich überlagerte Benutzereingaben werden von dem Regelsystem bis auf wenige Ausnahmen grundsätzlich als Teil einer multimodalen Interaktion eingeordnet. Bei rein sequentiellen Eingaben wird die Zugehörigkeit zu einem spezifischen Integrationsintervall aufgrund der semantischen Beziehungen zwischen den einzelnen Informationsanteilen getroffen. Ein Problem stellen zur Zeit noch diejenigen Benutzereingaben dar, die bereits von dem Erkennermodul falsch klassifiziert wurden. Aus diesem Grund könnten im Spotting neben dem besten Ergebnis, wie es momentan realisiert ist, in Zukunft auch weitere Resultate mit entsprechenden Konfidenzwerten ausgewertet werden. Eine weitere Verbesserung lässt sich unter Umständen auch durch eine Migration des Spottings und der Fusion zu einem einstufigen Verfahren erzielen.

Den algorithmischen Schwerpunkt der Arbeit bildete die Einführung eines innovativen Verfahrens für die Fusion multimodaler Eingabedaten. Der entwickelte Ansatz zeichnet sich vor allem dadurch aus, dass er die Vorteile eines frei skalierbaren, domäneninvarianten Basissystems mit einer optionalen Auswertung zusätzlich vorhandener Kontextinformationen verbindet. Im Gegensatz zu klassischen, regelbasierten und stochastischen Integrationsansätzen wird die funktionale Zusammensetzung aus redundanten, komplementären und rivalisierenden Informationsanteilen bereits implizit durch die Struktur des Algorithmus unterstützt.

Angelehnt an die Prinzipien der natürlichen Evolution konkurrieren jeweils mehrere Integrationshypothesen miteinander. Unter Anwendung eines iterativen Optimierungsverfahrens werden einzelne Lösungsalternativen selektiert, rekombiniert und durch problem-adäquate genetische Operatoren manipuliert. Dabei tauschen sie in einem simulierten Evolutionsprozess Informationen untereinander aus und vererben sie an die nächste Generation. Auf diese Weise entsteht eine neue Population von Integrationsergebnissen, die im Hinblick auf die Lösung des Integrationsproblems im Allgemeinen verbesserte Eigenschaften aufweist.

Die Ergebnisse der Evaluierung haben gezeigt, dass die einzelnen genetischen Strukturparameter zwar einen beträchtlichen Einfluss auf die Integrationszeit, nicht aber auf das erzielte Integrationsergebnis haben. Darüber hinaus konnte eine weitgehende Unabhängigkeit der Laufzeit von der Komplexität der zu bearbeitenden Integrationsaufgabe nachgewiesen werden. Vor dem Hintergrund realzeitfähiger Anforderungen müssen speziell die Einstellungen für die Größe der Ausgangspopulation und der Paarungspopulation sorgfältig gewählt werden. In Bezug auf die Erreichung stabiler Integrationsergebnisse ist es wichtig, dass eine Mindestanzahl von ca. zehn Iterationsläufen vor der ersten Konvergenzprüfung nicht unterschritten wird.

Das Konvergenzverhalten und die zeitliche Performanz lassen sich an verschiedenen Stellen noch deutlich verbessern. Einen ersten Ansatzpunkt bietet eine heterogene Initialisierung der Startpopulation mit den unterschiedlichen Erkennungsergebnissen aus den einzelnen Eingabemodulen. Darüber hinaus kann durch die Einführung variabler Populationsgrößen zum einen der Suchraum in Abhängigkeit von dem situativen Bedienkontext erweitert und zum anderen die Auffindung geeigneter Integrationshypothesen beschleunigt werden. Eine weitere Verbesserung ist im Rahmen der Einführung co-evolutionärer Fusionsstrategien zu erwarten, bei denen das Ergebnis des Integrationsprozesses nicht nur aus der am besten bewerteten Lösungshypothese besteht, sondern eine Liste von mehreren Alternativen mit entsprechenden Konfidenzmaßen generiert wird. Die einzelnen Modifikationen implizieren jeweils eine Anpassung der Selektionsmechanismen und der Konvergenzüberprüfungen. Schließlich lassen sich auch in Bezug auf die Implementierung eine Reihe von Optimierungen durchführen. So wurde beispielsweise das Potenzial paralleler Berechnungsmöglichkeiten nicht ausgeschöpft.

In Kombination mit der regelbasierten Vorverarbeitung konnten durch den evolutionären Fusionsansatz die initial formulierten Erwartungen an ein generisches Konzept für die Integration multimodaler Informationsanteile weitgehend erfüllt werden. Der Nachweis einer praktischen Nutzbarkeit der entwickelten Systemarchitektur und des darauf aufsetzenden, hybriden Integrationsalgorithmus wurde anhand von mehreren Demonstratoren erbracht. Die in dieser Arbeit propagierten Methoden stellen eine Möglichkeit dar, auf dem Weg der schrittweisen Annäherung an das Vorbild der zwischenmenschlichen Kommunikation den Umgang mit komplexen technischen Systemen einfacher, flexibler und robuster zu gestalten und vielleicht dafür zu sorgen, dass die Bedienung auch im alltäglichen Einsatz mehr Spaß macht.

ANHANG A

Details zu den Benutzerstudien

Dieses Kapitel enhält zusätzliche Informationen im Hinblick auf den Aufbau und die Ergebnisse der in Kapitel 3 und Kapitel 4 beschriebenen Benutzerstudien. Neben einer Auflistung der einzelnen Aufgaben im AIA-Szenario und einer Zusammenstellung der Ereignisklassen werden hier die einzelnen Fragebögen angegeben. Die Darstellung der ergänzenden quantitativen Ergebnisse erfolgt ohne eine explizite textuelle Einordnung, da die entsprechenden Kontexte bereits in den dazugehörigen Textabschnitten eingeführt wurden und auf die hier angegebenen Informationen zudem direkt Bezug genommen wird.

A.1 Versuchsaufbau

Im Folgenden werden kurz die einzelnen Benutzeraufgaben $\mathcal{Y}^A$ jeweils getrennt für die zweite $\mathcal{Y}_2^A$ und für die dritte Phase $\mathcal{Y}_3^A$ des AIA-Versuchs beschrieben.

Beschreibung der Aufgaben $\mathcal{Y}_2^A$ in der zweiten Phase des AIA-Versuchs

$Y_{2,1}^A$	Starten der Wiedergabe im MP3 Modus
$Y_{2,2}^A$	Springen zum übernächsten Titel
$Y_{2,3}^A$	Lautstärke etwas reduzieren
$Y_{2,4}^A$	Ein Lied in der aktuellen Liste zurückspringen
$Y_{2,5}^A$	In den Radiomodus wechseln
$Y_{2,6}^A$	Sender WDR 2 auswählen
$Y_{2,7}^A$	Eingehenden Anruf von Getränkeservice annehmen
$Y_{2,8}^A$	In den Radiomodus wechseln und Sender *Gong 96.3* auswählen
$Y_{2,9}^A$	Eingehenden Anruf von *Martina* ablehnen
$Y_{2,10}^A$	In den MP3-Modus wechseln und Wiedergabe starten

Beschreibung der Aufgaben $\mathcal{Y}_2^A$ in der zweiten Phase des AIA-Versuchs

$Y_{2,11}^A$ In der aktuellen Wiedergabeliste drei Lieder nach vorne springen
$Y_{2,12}^A$ Lautstärke etwas erhöhen
$Y_{2,13}^A$ Teilnehmer Gregor anrufen
$Y_{2,14}^A$ Teilnehmer nimmt den Anruf nicht an, Gespräch beenden
$Y_{2,15}^A$ Titelsong der Mupped-Show auswählen und spielen
$Y_{2,16}^A$ Musikwiedergabe beenden

Beschreibung der Aufgaben $\mathcal{Y}_3^A$ in der dritten Phase des AIA-Versuchs

$Y_{3,1}^A$ Starten der Wiedergabe im MP3 Modus
$Y_{3,2}^A$ Teilnehmer *Karla* in USA anrufen
$Y_{3,3}^A$ Teilnehmer nimmt den Anruf nicht an, Gespräch beenden
$Y_{3,4}^A$ In den MP3-Modus wechseln und das Lied *Computerliebe* auswählen
$Y_{3,5}^A$ Noch einmal die zuletzt gewählte Nummer anrufen
$Y_{3,6}^A$ In den Radiomodus wechseln und einen Schlagersender auswählen
$Y_{3,7}^A$ Lautsärke des Radios etwas erhöhen
$Y_{3,8}^A$ Eingehenden Anruf aus Süd-Schweden annehmen
$Y_{3,9}^A$ Radiosender Bayern3 einstellen
$Y_{3,10}^A$ In den MP3-Modus wechseln und beliebiges Lied auswählen
$Y_{3,11}^A$ Zum übernächsten Titel springen
$Y_{3,12}^A$ Die Lautstärke etwas reduzieren
$Y_{3,13}^A$ Ein Lied der Gruppe *Guano Apes* auswählen
$Y_{3,14}^A$ Eingehenden Anruf von Martina ablehnen, Wiedergabe fortsetzen
$Y_{3,15}^A$ Teilnehmer Martina jetzt anrufen
$Y_{3,16}^A$ Teilnehmer nimmt den Anruf nicht an, Gespräch beenden
$Y_{3,17}^A$ In den MP3-Modus wechseln und ein Lied von *J. Lopez* auswählen
$Y_{3,18}^A$ Wiedergabe in den Pause-Modus setzen
$Y_{3,19}^A$ Pause-Modus wieder aufheben
$Y_{3,20}^A$ Drei Lieder in der Wiedergabeliste zurückspringen
$Y_{3,21}^A$ Den vorletzten Eintrag in der Wiedergabeliste auswählen
$Y_{3,22}^A$ Den ersten Eintrag in der Wiedergabeliste auswählen
$Y_{3,23}^A$ Musikwiedergabe beenden

I. Persönliche Angaben

VPIndex:

Geschlecht: ❑ M ❑ W

Alter:

Beruf:

II. Vorbefragung

Welche Erfahrungen / Kenntnisse haben Sie im Umgang mit ...

	sehr viel					keine
... Computersystemen (allgemein)	❑	❑	❑	❑	❑	❑
... rein tastaturbasierten Systemen	❑	❑	❑	❑	❑	❑
... graphischen Benutzeroberflächen (Windows 95, etc.)	❑	❑	❑	❑	❑	❑
... mit Touchscreens	❑	❑	❑	❑	❑	❑
... Audioeinrichtungen im Auto (Radio, CD, etc.)	❑	❑	❑	❑	❑	❑
... Telefonanwendungen im Auto	❑	❑	❑	❑	❑	❑
... Navigationssystemen	❑	❑	❑	❑	❑	❑
... sprachbedienbaren Systemen (allgemein)	❑	❑	❑	❑	❑	❑
... kommandosprachlichen Systemen	❑	❑	❑	❑	❑	❑
... natürlichsprachlichen Systemen	❑	❑	❑	❑	❑	❑
... gestenbasierten Systemen (Handgesten)	❑	❑	❑	❑	❑	❑
... gestenbasierten Systemen (Kopfgesten)	❑	❑	❑	❑	❑	❑
... gestenbasierten Systemen (Ganzkörpergesten)	❑	❑	❑	❑	❑	❑
... der Kombination mehrerer Eingabegeräte (allgemein)	❑	❑	❑	❑	❑	❑
... der Kombination Touchscreen und Sprache	❑	❑	❑	❑	❑	❑
... der Kombination Sprache und Handgestik	❑	❑	❑	❑	❑	❑
... der Kombination Sprache und Kopfgestik	❑	❑	❑	❑	❑	❑
... der Kombination Touchscreen und Kopfgestik	❑	❑	❑	❑	❑	❑

	JA	NEIN
Haben Sie bereits an einem Versuch zur Bedienung von Audio- und Kommunikationseinrichtungen im Auto teilgenommen ?	❑	❑
Ist dieser Versuch Ihr erster multimodaler Usability Versuch ?	❑	❑

Abb. A.1: Anfangsfragebogen im AIA-Versuch (Phase I)

Fragen nach Block I:

VPIndex:

Teil I: Bewertungen der Kombination: Touchscreen + Tastenblock

Wie haben Sie die Bedienung empfunden ?								Anmerkungen:
angenehm	❑	❑	❑	❑	❑	❑	unangenehm	
einfach	❑	❑	❑	❑	❑	❑	umständlich	
effektiv	❑	❑	❑	❑	❑	❑	ineffektiv	
entlastend	❑	❑	❑	❑	❑	❑	belastend	
sicher	❑	❑	❑	❑	❑	❑	unsicher	

Teil II: Bewertungen der Kombination: Touchscreen + Sprache

Wie haben Sie die Bedienung empfunden ?								Anmerkungen:
angenehm	❑	❑	❑	❑	❑	❑	unangenehm	
einfach	❑	❑	❑	❑	❑	❑	umständlich	
effektiv	❑	❑	❑	❑	❑	❑	ineffektiv	
entlastend	❑	❑	❑	❑	❑	❑	belastend	
sicher	❑	❑	❑	❑	❑	❑	unsicher	

Teil III: Bewertungen der Kombination: Sprache + Handgestik

Wie haben Sie die Bedienung empfunden ?								Anmerkungen:
angenehm	❑	❑	❑	❑	❑	❑	unangenehm	
einfach	❑	❑	❑	❑	❑	❑	umständlich	
effektiv	❑	❑	❑	❑	❑	❑	ineffektiv	
entlastend	❑	❑	❑	❑	❑	❑	belastend	
sicher	❑	❑	❑	❑	❑	❑	unsicher	

Teil IV: Bewertungen der Kombination: Sprache + Kopfgestik

Wie haben Sie die Bedienung empfunden ?								Anmerkungen:
angenehm	❑	❑	❑	❑	❑	❑	unangenehm	
einfach	❑	❑	❑	❑	❑	❑	umständlich	
effektiv	❑	❑	❑	❑	❑	❑	ineffektiv	
entlastend	❑	❑	❑	❑	❑	❑	belastend	
sicher	❑	❑	❑	❑	❑	❑	unsicher	

Abb. A.2: Modalitätenspezifische Zwischenbefragung im AIA-Versuch (Phase II)

Fragen nach Block II: VPIndex:

Bewertungen der freien kombinierten Bedienung

Wie haben Sie insgesamt die Bedienung empfunden ?								Anmerkungen:
angenehm	❑	❑	❑	❑	❑	❑	unangenehm	
einfach	❑	❑	❑	❑	❑	❑	umständlich	
effektiv	❑	❑	❑	❑	❑	❑	ineffektiv	
entlastend	❑	❑	❑	❑	❑	❑	belastend	
sicher	❑	❑	❑	❑	❑	❑	unsicher	

Was halten Sie von ...	sehr viel					wenig
... dem parallelen Einsatz von 2 Modalitäten (allgemein)	❑	❑	❑	❑	❑	❑
... dem parallelen Einsatz von 3 Modalitäten (allgemein)	❑	❑	❑	❑	❑	❑
... dem parallelen Einsatz von mehreren Modalitäten (3+)	❑	❑	❑	❑	❑	❑
... der Kombination Touchscreen und Tastenblock	❑	❑	❑	❑	❑	❑
... der Kombination Touchscreen und Sprache	❑	❑	❑	❑	❑	❑
... der Kombination Sprache und Handgestik	❑	❑	❑	❑	❑	❑
... der Kombination Sprache und Kopfgestik	❑	❑	❑	❑	❑	❑
... der Kombination Sprache und Tastenblock	❑	❑	❑	❑	❑	❑
... der Kombination Tastenblock und Kopfgestik	❑	❑	❑	❑	❑	❑
... der Kombination Touchscreen und Kopfgestik	❑	❑	❑	❑	❑	❑
... der Kombination Handgestik und Kopfgestik	❑	❑	❑	❑	❑	❑

Welche Funktionen möchten Sie bevorzugt steuern mit...

... dem Tastenblock?	... dem Touchscreen?
... Sprache?	... Kopfgestik?
... Handgestik?	Anmerkungen:

Abb. A.3: Seite eins der Abschlussbefragung im AIA-Versuch (Phase III)

VPIndex:

Mit welchen Modalitäten möchten Sie folgende Funktionen bevorzugt steuern?

	Tasten	Touchscr.	Sprache	Kopfgestik	Handgestik
Modusauswahl	❑	❑	❑	❑	❑
Durch Liste scrollen	❑	❑	❑	❑	❑
Aus Liste auswählen	❑	❑	❑	❑	❑
Lautstärke hoch/ runter	❑	❑	❑	❑	❑
Play (Player)	❑	❑	❑	❑	❑
Stop (Player)	❑	❑	❑	❑	❑
Pause (Player)	❑	❑	❑	❑	❑
Skip forward (Player)	❑	❑	❑	❑	❑
Skip backward (Player)	❑	❑	❑	❑	❑
Annehmen (Telefon)	❑	❑	❑	❑	❑
Ablehnen (Telefon)	❑	❑	❑	❑	❑
Wählen (Telefon)	❑	❑	❑	❑	❑
Wahlwdh. (Telefon)	❑	❑	❑	❑	❑

Was war gut an der Bedienung des Systems ?	Was war schlecht ?
Was hat Ihnen am besten gefallen ?	Was hat Ihnen überhaupt nicht gefallen ?
Was hätten Sie gerne anders ?	Fehlt etwas ?
Weitere Anmerkungen:	

Abb. A.4: Seite zwei der Abschlussbefragung im AIA-Versuch (Phase III)

A.2 Ereignisklassen

Name	Beschreibung
E	allgemeines Ereignis
E^{BE}	Benutzereingabe, $E_{BE} \in \{E^T, E^A, E^H, E^K, E^S\}$
E^{WIZ}	von Versuchsleiter ausgelöste Interaktion
E^{WC}	Kontrolleingriff des Versuchsleiters
E^{SYS}	Systemfeedback
E^{EXT}	Ereignis eines externen Moduls
E^{TA}	direkte, haptische Benutzereingabe, $E^{TA} \in \{E^T, E^A\}$
E^{HKS}	indirekte Benutzereingabe, $E^{HKS} \in \{E^H, E^K, E^S\}$
E^T	Eingabe des Benutzers mittels Touchscreen
E^A	Eingabe des Benutzers mittels Tastenkonsole
E^H	Eingabe des Benutzers mittels Handgestik
E^K	Eingabe des Benutzers mittels Kopfgestik
E^S	Eingabe des Benutzers mittels Sprache
E^{UM}	unimodale Benutzereingabe
E^{MM}	multimodale Benutzereingabe

Tab. A.1: Zusammenhängende Aufstellung der allgemeinen Ereignisklassen

Name	Beschreibung
E^{STA}	gestaffelt überlagerte multimodale Benutzereingabe
E^{SEQ}	sequentiell überlagerte multimodale Benutzereingabe
E^{NES}	eingebundene überlagerte multimodale Benutzereingabe
E^{SIM}	simultan überlagerte multimodale Benutzereingabe
E^{NE1}	vorlaufend eingebundene multimodale Benutzereingabe
E^{NE2}	mittig eingebundene multimodale Benutzereingabe
E^{NE3}	nachlaufend eingebundene multimodale Benutzereingabe
E^{NE4}	annähernd simultan eingebundene multimodale Benutzereingabe
E^{NEC}	längere, eine andere vollständig enthaltene Benutzereingabe
E^{NEN}	kürzere, in der anderen vollständig eingebundene Benutzereingabe

Tab. A.2: Zusammenhängende Aufstellung der zeitbezogenen Ereignisklassen

Name	Beschreibung
E^{RED}	redundante multimodale Benutzereingabe
E^{RIV}	rivalisierende multimodale Benutzereingabe
E^{CMP}	komplementäre multimodale Benutzereingabe
E^{REC}	redundant-komplementäre multimodale Benutzereingabe
E^{RIC}	rivalisierend-komplementäre multimodale Benutzereingabe
E^{HAC}	harte komplementäre multimodale Benutzereingabe
E^{BMM}	bimodale Benutzereingabe
E^{TMM}	trimodale Benutzereingabe
E^{KMM}	komplex multimodale Benutzereingabe

Tab. A.3: Zusammenhängende Aufstellung der multimodalen Ereignisklassen

A.3 Weitere Ergebnisse

	Block 1 (K_{TA})		Block 2 (K_{TS})		Block 3 (K_{SH})		Block 4 (K_{KS})	
DVA	E^{UM}	E^{MM}	E^{UM}	E^{MM}	E^{UM}	E^{MM}	E^{UM}	E^{MM}
ANF	97,1	2,9	86,7	13,3	70,9	29,1	67,6	32,4
NOR	82,3	17,7	82,3	17,7	58,0	42,0	65,5	34,5
EXP	90,6	9,4	87,0	13,0	60,2	39,8	68,0	32,0
AVP	92,9	7,1	84,5	15,5	61,7	38,3	66,6	33,4

Tab. A.4: Prozentuale Verteilung der unimodalen (E^{UM}) und multimodalen (E^{MM}) Benutzereingaben im DVA-Szenario bezogen auf die ersten vier Blöcke in Phase II des Versuchs; K_{TA}=Kombination von Touchscreen und Tastatur, K_{TS}=Kombination von Touchscreen und Sprache, K_{SH}=Kombination von Sprache und Handgestik und K_{SK}=Kombination von Sprache und Kopfgestik

	Kopfgestik E^K				
DVA	E_K^{STA}	E_K^{SEQ}	E_K^{NEC}	E_K^{NEN}	E_K^{SIM}
ANF	1,12	0,57	1,44	0,98	1,00
NOR	1,26	0,95	1,54		0,90
EXP	1,29		1,36		1,28
AVP	1,14	0,91	1,50	1,08	0,98

Tab. A.5: Durchschnittliche Interaktionsdauern in [sec] mit Bezug auf die verschiedenen Überlagerungstypen speziell für die Kopfgestik (E^K), aufgeteilt nach den einzelnen Benutzerklassen $U \in \mathcal{U}$

AIA	E^T	E^A	E^S	E^K	E^H
Modusauswahl ($A_{19}..A_{22}$)	1,7 (0,66)	3,4 (1,10)	1,4 (0,63)	4,8 (0,48)	4,5 (0,58)
Liste bewegen (A_{16}, A_{17})	2,8 (1,17)	2,1 (1,21)	2,4 (1,20)	4,8 (0,65)	4,1 (1,03)
Eintrag auswählen (A_{18})	2,3 (1,17)	2,6 (1,23)	1,6 (0,79)	4,7 (0,77)	4,3 (0,90)
Lautstärke ändern ($A_{22}..A_{25}$)	3,6 (0,84)	1,5 (1,07)	2,1 (0,98)	4,7 (0,72)	3,6 (1,29)
MP3 Player ($A_1..A_3$)	2,1 (1,03)	3,2 (0,98)	1,3 (0,55)	4,6 (0,87)	3,9 (1,17)
MP3 Skipping (A_4, A_5)	2,1 (0,88)	2,6 (1,32)	2,1 (1,22)	4,7 (0,72)	3,4 (1,39)
Radio Skipping (A_4, A_5)	2,3 (0,81)	2,6 (1,21)	1,9 (1,13)	4,8 (0,56)	3,5 (1,31)
Wählen/Beenden (A_{11}, A_{12})	2,4 (0,87)	3,1 (0,92)	1,3 (0,80)	4,8 (0,83)	4,1 (1,97)
Annehmen/Ablehnen (A_{13}, A_{14})	2,4 (0,96)	4,0 (0,98)	1,8 (1,06)	3,1 (1,64)	3,4 (1,47)

Tab. A.6: Durchschnittliche Bewertungen und Standardabweichungen der Modaliäten $\{T, A, H.K, S\}$ in Abhängigkeit von den einzelnen Systemfunktionalitäten im AIA-Szenario. Nach einem forced-rating Verfahren wird Platz eins für die am besten geeignete und Platz fünf für die am wenigsten geeignete Modalität vergeben; die Bezeichnungen für die Systemaktionen beziehen sich auf Tabelle 3.2

	DVA				AIA			
	ANF	NOR	EXP	AVP	ANF	NOR	EXP	AVP
N_{VP}	11	20	9	40	8	13	7	28
N_{VP} (rel.)	0,275	0,500	0,225		0,286	0,464	0,250	

Tab. A.7: Absolute und relative Verteilung der Versuchsteilnehmer auf die unterschiedlichen Benutzerklassen ANF, NOR und EXP, basierend auf einer Auswertung des Anfangsfragebogens

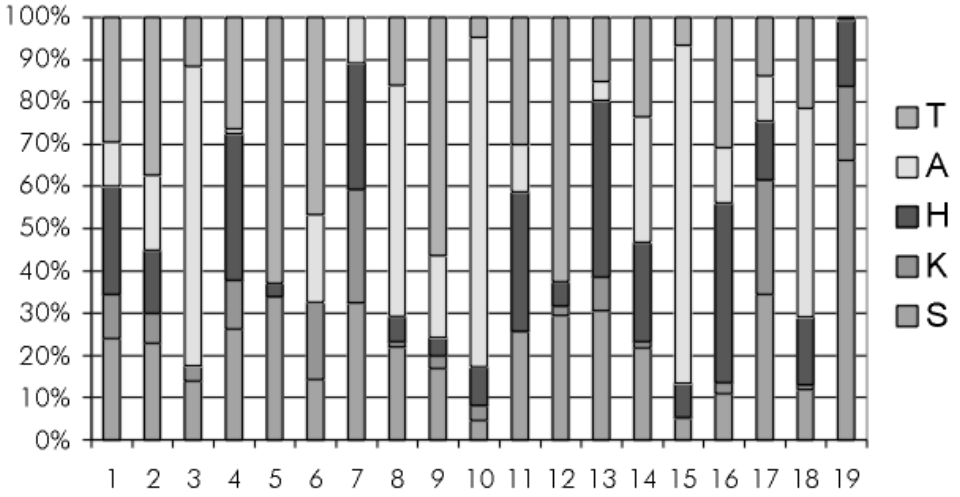


Abb. A.5: Interpersonelle Verteilung der verwendeten Modalitäten (T=Touchscreen, A=Tastenkonsole, H=Handgestik, K=Kopfgestik und S=Sprache) bei den 19 augewählten Versuchspersonen zur Evaluierung der Zeitrelationen in der Phase III des DVA-Versuchs

ANHANG B

Formale Grammatiken

In diesem Abschnitt werden die zentralen Begriffsterminologien im Kontext formaler Sprachen kurz rekapituliert. Eine detaillierte Beschreibung ist beispielsweise in [138, 162] enthalten. Die Kenntnis der verwendeten Darstellungsformalismen ist eine Grundvoraussetzung für das allgemeine Verständnis der Arbeit, speziell im Hinblick auf die Konzeption der Basisarchitektur und die Diskussion des hybriden Integrationsalgorithmus.

B.1 Grundlegende Definitionen

Symbole, Alphabete und Wörter

Sei Σ eine endliche, nichtleere Menge von Symbolen $\Sigma = \{S_1, S_2, ..., S_\sigma\}$. Die Kardinalität von Σ wird mit $\sigma := |\Sigma|$ angegeben. Die einzelnen Symbole können zwar unterschieden, nicht aber weiter zerlegt werden. Sind diese Voraussetzungen erfüllt, so bezeichnet man Σ auch als ein *Alphabet*. Die folgende Darstellung enthält zwei mögliche Beispiele. Im ersten Fall repräsentieren die elementaren Symbole jeweils einzelne Zeichen aus dem ASCII-Code. Demgegenüber bestehen die Elemente des zweiten Alphabetes bereits aus mehreren Zeichen.

$$\begin{aligned} \Sigma_1 &= \{\text{a,b,c,d,} \ldots, \text{z}, 0, 1, \ldots, 9\} \\ \Sigma_2 &= \{\text{walk, fly, examine, trans, rot, left, right, up, down}\} \end{aligned}$$

Sei Σ ein Alphabet. Eine Folge $s_1 s_2 \ldots s_n$ $(n \geq 0)$ von n Symbolen aus Σ repräsentiert ein *Wort* über Σ der Länge n. Wörter entstehen durch die Verkettung einzelner Symbole, können aber auch mit einem isolierten Symbol übereinstimmen. Die Länge eines Wortes w wird abkürzend mit $|w|$ bezeichnet. Das einzige Wort der Länge 0 heißt das *leere Wort* und wird allgemein durch das besondere Symbol ε angegeben. Beispiele für Wörter mit Bezug auf die beiden oben angegebenen Alphabete sind im Folgenden aufgeführt. Zur besseren Kennzeichnung sind die Wörter in runde Klammern eingeschlossen:

$$\begin{aligned} \text{Wörter aus} \quad \Sigma_1 : \quad & \{(\text{d i e}), (\text{a n t w o r t}), (\text{i s t}), (4\ 2), \ldots\} \\ \text{Wörter aus} \quad \Sigma_2 : \quad & \{(\text{walk trans left}), (\text{right}), (\text{rot up}), \ldots\} \end{aligned}$$

Wenn ein Symbol mehrfach hintereinander in einem Wort vorkommt, dann kann abkürzend auch die in der Mathematik übliche Potenzschreibweise verwendet werden. Auf diese Weise lässt sich das Wort $w_1 = \text{abba}$ auch durch $w_1 = \text{ab}^2\text{a}$ darstellen. Sei Σ wiederum ein Alphabet einzelner Symbole. Dann bezeichnet Σ^* die Menge aller Wörter (einschließlich ε) über Σ. Diese Menge wird *Stern von* Σ genannt. Die folgende Darstellung enthält eine Aufzählung der möglichen Wörter über einem binären Alphabet $\Sigma_B = \{0, 1\}$:

$$\Sigma_B^* = \{\varepsilon, 0, 1, 00, 01, 10, 000, 010, 100, 001, 011, 111, 0000, \dots\}$$

Formale Sprachen und Grammatiken

Eine formale Sprache $\mathcal{L}$ über einem Alphabet Σ von Symbolen ist eine Teilmenge von Σ^*. Für einfach strukturierte Sprachen kann die Definition in verständlicher und nachvollziehbarer Form noch durch die Angabe einer expliziten Aufzählung der gültigen Wörter über Σ beschrieben werden. Bei größeren Wortmengen (oft mit unendlicher Dimension) erfolgt die Spezifikation der Sprache mittels einer abstrakten Grammatik. Über einen impliziten Regelmechanismus können entweder syntaktisch korrekte Wörter einer Sprache erzeugt oder ein vorgegebenes Wort daraufhin überprüft werden, ob es sich um ein syntaktisch korrektes Element der Sprache handelt. Der erste Prozess wird als *Generierung* und der zweite als *Parsing* bezeichnet.

Grammatiken werden allgemein durch Regeln in der Form: *linkeSeite* $\rightarrow$ *rechteSeite* beschrieben. Die Regeln sind zumeist nicht deterministisch, das heißt eine linke Seite kann durch verschiedene rechte Seiten ersetzt werden. Bei den Elementen der Grammatik wird strikt zwischen *terminalen Symbolen* (Symbole aus dem zugrundeliegenden Alphabet) und *nichtterminalen Symbolen* (Variablen zur Beschreibung der Sprachstruktur) unterschieden.

Formal kann eine Grammatik G als 4-Tupel $G = (\Phi, \Sigma, R, S)$ definiert werden. Die Bedeutung der einzelnen Elemente wird im Folgenden kurz erklärt:

Φ : endliche Menge von Variablen (nicht-terminale Symbole)

Σ : Alphabet von Terminalsymbolen, wobei $\Sigma \cap \Phi = \emptyset$

R : endliche, nichtleere Menge von Produktionen
$R \subset (\Gamma^* \cdot \Phi \cdot \Sigma^*) \times \Gamma^*$, mit dem Gesamtalphabet $\Gamma = \Phi \cup \Sigma$

S : Startsymbol, wobei $S \in \Phi$

Nach Chomsky [138] lassen sich Grammatiken in vier Typen einteilen:

- Allgemeine Grammatik (*Typ 0-Grammatik*):
  Jede Grammatik ist automatisch vom *Typ 0*. Man spricht in diesem Fall von *allgemeinen Phrasenstrukturgrammatiken*. Die einzige Einschränkung für die Regeln dieser Grammatiken besteht darin, dass auf der linken Seite der Produktionen mindestens eine Variable vorkommen muss.

- Kontextsensitive Grammatik (*Typ 1-Grammatik*):
  Eine Typ 0-Grammatik $G = (\Phi, \Sigma, R, S)$ ist vom *Typ 1* oder *kontextsensitiv*, falls alle Regeln von R in der Form $uAv \rightarrow uwv$ notiert werden können mit $u, v, w \in \Gamma^*$, $A \in \Phi$ und $w \neq \varepsilon$ falls $A \neq S$. Zusätzlich darf das Startsymbol S nicht auf der rechten Seite einer Regel vorkommen. Wegen $w \neq \varepsilon$ gilt: $|uAv| \leq |uwv|$, d.h. die Länge der Wörter ist monoton steigend. Man sagt daher auch: die Grammatik ist *expansiv*.

- Kontextfreie Grammatik (*Typ 2-Grammatik*)
  Eine Typ 1-Grammatik ist vom *Typ 2* oder *kontextfrei*, wenn bei allen Regeln auf der linken Seite immer nur eine einzelne Variable vorkommt. Für die Regeln gilt: $R \subset \Phi \times \Gamma^*$

- Reguläre Grammatik (*Typ 3-Grammatik*)
  Eine Typ 2-Grammatik ist vom *Typ 3* oder *regulär*, wenn für alle Regeln $w_1 \rightarrow w_2$ gilt: $w_1 \in \Phi$ und $w_2 \in \Sigma \cup \Sigma\Phi$, d.h. die rechten Seiten der Regeln sind entweder einzelne Terminalsymbole oder ein Terminalsymbol gefolgt von einer Variablen.

Kontextfreie und kontextsensitive Grammatiken unterscheiden sich folgendermaßen: Eine kontextfreie Regel $A \rightarrow w$ bewirkt, daß A unabhängig vom Kontext, in dem A steht, durch w ersetzt wird. Bei einer kontextsensitiven Grammatik können auch Regeln der Form $uAv \rightarrow uwv$ vorkommen. In diesem Fall wird A nur dann durch w ersetzt, wenn A im Kontext zwischen u und v steht. Eine Sprache heißt kontextfrei, wenn sie durch eine kontextfreie Grammatik erzeugt wird. Entsprechendes gilt für die anderen Grammatik-Typen.

B.2 Darstellungskonventionen

Kontextfreie Grammatiken spielen eine zentrale Rolle bei der formalen Notation der multimodalen Informationsanteile. Zur formalen Beschreibung der grammatikalischen Beziehungen eignen sich beispielsweise graphisch-orientierte Synatxdiagramme oder die stärker text-orientierte Auflistung der Regeln in Backus-Naur-Form (BNF) [77]. Beide Beschreibungsformalismen lassen sich direkt ineinander überführen. In der vorliegenden Arbeit wird maßgeblich die BNF in der erweiterten Version (EBNF) [138] benutzt. Um bei der Zusammensetzung der Elemente besser zwischen terminalen und nicht-terminalen Symbolen unterscheiden zu können, haben sich verschiedene Darstellungskonventionen etabliert.

Variablen werden durch den Einschluss in spitze Klammern textuell speziell gekennzeichnet. Dadurch können neben einzelnen Zeichen auch zusammenhängede, vom Namen her sinngebende Zeichenfolgen zur Notation von Nicht-Terminalsymbolen verwendet werden. Anstatt mit der Pfeilzuweisung werden in der BNF die linke und die rechte Regelseite durch ein spezielles Zuweisungssysmbol (::=) getrennt. Existieren zu einer linken Regelseite mehrere Alternativen, so können die einzelnen Optionen auf der rechten Seite, jeweils durch die Meta-Variable | getrennt, direkt hintereinander geschrieben werden ohne die linke Seite jedes Mal explizit wiederholen zu müssen. Als Beispiel werden im Folgenden die BNF-Syntaxregeln für die textuelle Zusammensetzung eines Bezeichners ($VNAME$) in der Hochsprache PASCAL [74] angegeben:

$$
\begin{aligned}
<VNAME> &::= <CHAR> \mid <CHAR><DIGIT> \\
<DIGIT> &::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\
<CHAR> &::= A \mid B \mid C \mid D \mid ... \mid Z \\
<STRING> &::= <CHAR> \mid <DIGIT> \mid \\
&::= <CHAR><STRING> \mid <DIGIT><STRING>
\end{aligned}
$$

Nach dieser Definition setzt sich ein Bezeichner aus einer Zeichenkette zusammen, die mit einem Buchstaben beginnen muss. In der Länge ist die Zeichenkette nicht begrenzt. Syntaktisch korrekte Wörter dieser Grammatik sind z.B X, X42, TFG, nicht aber 010 oder 5f.

Gegenüber einer standardmäßigen BNF sind bei der EBNF zusätzlich verschiedene Meta-Symbole erlaubt, die vor allem die Lesbarkeit und den Kompaktheitsgrad der Darstellung erhöhen. Einzelne Symbole oder Symbolfolgen, die in einer Regeldefinition auch weggelassen werden können, lassen sich abkürzend in eckige Klammern [...] darstellen. Darüber hinaus können auf der rechten Seite einer Regel Symbole oder Symbolfolgen, die beliebig oft wiederholt oder auch komplett weggelassen werden können, in geschweifte Klammern {...} eingeschlossen werden. Innerhalb dieser Klammerstrukturen lassen sich wiederum durch | getrennt mehrere Alternativen angeben. Die Definition eines Bezeichners kann mit dem erweiterten Formalismus deutlich einfacher dargestellt werden:

$$<VNAME> ::= <CHAR> \{ <CHAR> \mid <DIGIT> \}$$

B.3 Test- und Demonstratorsysteme

Dieser Abschnitt enthält eine Spezifikation des Funktionsvokabulars von drei verschiedenen Systemen, die in der vorliegenden Arbeit verwendet werden. Die Beschreibungen beschränken sich auf eine Angabe der entsprechenden BNF-Regeln. Zur Verbesserung der Lesbarkeit wurden einzelne Kommentare eingefügt, die jeweils durch eine Raute gekennzeichnet sind.

Abstrakte Evaluierungsgrammatik

```
<S>         ::= <CMD_SEQ>
<CMD_SEQ>   ::= <CMD> | <CMD> <CMD_SEQ>
<CMD>       ::= <S> | <S1>

<S>          ::= Semun | Semun <Komplement> | Semun <Komplement> <Komplement>
<S1>         ::= Semun_1 | Semun_1 <Komplement_zu_1> | Semun_1 <STRING>

<Komplement>       ::= Komplement_1 | Komplement_2 | Komplement_3
<Komplement_zu_1>  ::= Komplement_4 | Komplement_5
```

Infotainmentsystem im Automobil

```
<EVENT> ::= <BASICS>   # verschiedene Basisfunktionalitäten
        ::= <MNAVIG>   # Menüauswahl und allgemeine Navigation
        ::= <AUDIO>    # alles zu den Audiofunktionalitäten
        ::= <COM>      # Kommunikationsgeräte
        ::= <BEST>     # Spezielle Best-Auswahlmöglichkeiten
        ::= <SET>      # diverse Einstellungsmöglichkeiten
        ::= <LIST>     # Listen und Canvas bezogene Funktionalitäten
        ::= <HELP>     # Aktive und passive Hilfsfunktionalitäten
        ::= <ERROR>    # Fehleridentifikation und Fehlermangement
        ::= <SCTRL>    # Sound-Control, Einstellungen
        ::= <DIV>      # diverse Zusatzfunktionalitäten
```

```
# --- Details zu verschiedenen Basisfunktionalitäten ----------------
<BASICS>::= switchon   # schaltet das Gerät ein

# --- Details zu Menü-Navigation und Auswahl ------------------------
<MNAV>  ::= audio      # aktiviert das Audio-Menü
        ::= com        # aktiviert das Kommunikationsmenü
        ::= best       # aktiviert das Settingsmenü
        ::= set        # aktiviert das Einstellungsmenü
        ::= back       # Sprung zurück in der State-History
        ::= redo       # wiederholt den letzten Befehl

# --- Details zu den Audioevents -------- --------------------------
<AUDIO>  ::= <PLYCTL>   # generelles player control für alle audio devices
         ::= <MP3>      # spezielle Steuerung für MP3-Device
         ::= <CD>       # spezielle Steuerung für CD-Gerät
         ::= <RADIO>    # spezielle Steuerung des Radio-Teils

<PLYCTL> ::= stop          # hält aktuelle Wiedergabe an
         ::= pause         # toggle für klassische Pause Funktion
         ::= backward      # springt ein Lied zurück
         ::= backward <no> # springt <no> Lieder zurück
         ::= forward       # springt ein Lied vor
         ::= forward <no>  # springt <no> Lieder vor
         ::= play_clt      # spielt den aktuellen Eintrag der Liste

<RADIO>  ::= radio         # aktiviert Radio-Menü, zeigt Liste an
         ::= play_rs <i>   # wählt direkt Radiostation an [1..n]

<MP3>    ::= mp3player          # aktiviert MP3 Basis-Auswahlmenü
         ::= play_mp3track <ti> # spielt track aus Liste ALLER MP3s
         ::= tagselect          # spezifische Filterroutine
         ::= allmp3             # zeigt direkt die Liste aller MP3s

<CD>     ::= cdplayer               # aktiviert CD Auswahlmenü
         ::= cdplay_it <cdid> <ti>  # spielt track <ti> von cd <cdid>

# --- Details zu den Kommunikationsevents ---------------------------
<COM>   ::= <TEL>      # Steuerung der Telefonfunktionalitäten
        ::= <WAP>      # Steuerung der WAP-Funktionalität
        ::= <SMS>      # Steuerung der SMS-Funktionalität

<SMS>   ::= smsmode         # aktiviert den SMS Modus
        ::= sms_char <c>    # fügt Charakter in aktuelle Darstellung ein
<WAP>   ::= wapmode         # aktiviert den WAP Modus
```

```
<TEL>   ::= telmode          # aktiviert den Telefon-Modus
        ::= call_accept      # aktuell eingehenden Anruf annehmen
        ::= call_deny        # aktuell eingehenden Anruf ablehnen
        ::= call_end         # aktuelle Verbindung beenden
        ::= show_adrind <i> # zeigt Eintrag i der Adressliste an [0..n-1]
        ::= tel_npad         # zeigt Nummernwahlmenü an
        ::= tel_clists       # zeigt Menü mit Ruflisten an

# --- Details zu Bestevents ----------------------------------------
<BEST>  ::= autotop <att>              # wählt spezielle Best-Auswahl
<att>   ::= MP3 | RADIO | WAP | TEL    # Auswahlmöglichkeiten

# --- Details zu Listenevents --------------------------------------
<LIST>  ::= up             # springt einen Eintrag in aktueller Liste hoch
        ::= up <no>        # springt <no> Einträge hoch
        ::= select         # wählt aktuellen Eintrag aus
        ::= down           # springt einen Eintrag in aktueller Liste runter
        ::= down <no>      # springt <no> Einträge runter

# --- Details zu Hilfe-Events --------------------------------------
<HELP>  ::= help           # kontextspezifische Hilfe
        ::= thanks         # Beispiel für Dialogabschluss

# --- Details zu Fehler-Events -------------------------------------
 <ERROR> ::= fem error     # Initiierung kontextspezifisches Fehlermanagement
         ::= sp_nrec       # Mitteilung der Spracheingabe nicht erkannt
         ::= usr_error     # Benutzerinitiierter Fehlerdialog

# --- Details zu SoundControl --------------------------------------
<SCTRL> ::= voldown        # Verringern der Lautstärke, Defaultinkrement
        ::= volup          # Erhöhen der Lautstärke
        ::= volmute        # Muten, Toggle auf 10% der Lautstärke

# --- Details zu Zusatzfunktionalitäten ----------------------------
<DIV>   ::= number <no>    # spezifiziert Zahleninformation, beispielsweise
                           # für multiples Skippen in Tracks, Listen, etc.
```

Multimodaler VRML-Browser

```
# --- Grundlegende Event-Spezifikation ---------------------------------
<EVENT>     ::= <Continous> | <Discrete> | <ContextSens>
            ::= <Referral> | <Control> | <STATUS>

# --- Bewegungsmodi und Basiskommandos ---------------------------------
<Continous> ::= start <Discrete> | stop
<Discrete>  ::= walk <Walk> | fly <Fly> | examine <Examine>
            ::= move <Move> | scale <Scale> | color <Color>

<Walk>      ::= trans <AllButY> | rot <XY>
<Fly>       ::= trans <AllDirs> | rot <XY> | roll <X>
<Examine>   ::= trans <XYZ> | rot <XY> | roll <X>
<Move>      ::= trans <AllDirs> | rot <XY> | roll <X>
<Scale>     ::= trans <iXiYiZS>
<Color>     ::= step <iHiSiViT> | goto <H_S_V>

# --- Richtungsdefinitionen --------------------------------------------
<AllDirs>   ::= <XYZ> | <Diag>
<AllButY>   ::= <XZ> | <Diag>
<Diag>      ::= lfwd | rfwd | lbwd | rbwd
<XYZ>       ::= <X> | <Y> | <Z>
<iXiYiZS>   ::= <iX> | <iY> | <iZ> |<S>
<XZ>        ::= <X> | <Z>
<XY>        ::= <X> | <Y>
<X>         ::= left | right
<Y>         ::= up | down
<Z>         ::= forward | backward

# --- Änderungen -------------------------------------------------------
<iXiYiZS>   ::= <iX> | <iY> | <iZ> | <S>
<iHiSiViT>  ::= <iH> | <iS> | <iV> | <iT>
<iX>        ::= incX | decX
<iY>        ::= incY | decY
<iZ>        ::= incZ | decZ
<S>         ::= bigger | smaller
<iH>        ::= incH | decH
<iS>        ::= incS | decS
<iV>        ::= incV | decV
<iT>        ::= incT | decT
<H_S_V>     ::= <FLOAT_VALUE> <FLOAT_VALUE> <FLOAT_VALUE>
```

```
# --- Kontextsensitive Kommandos ---------------------------------------
<ContextSens> ::= ks <CS_type>
<CS_type>     ::= <CS_mode> <CS_Trans> | <CS_Rot> | <CS_Roll> | <CS_Dirs>
<CS_Mode>     ::= walk | fly | examine
<CS_Trans>    ::= trans <AllDirs> | trans
<CS_Rot>      ::= rot <XY> | rot
<CS_Roll>     ::= roll <X> | roll

<CS_Dirs>     ::= <AllDirs>
<Referral>    ::= <RefActions> | <Indications> | <RefResolved>
<RefActions>  ::= gothere
<RefActions>  ::= lookat
<RefActions>  ::= setexacenter
<Indications> ::= indicated <Coordinates>
<Indications> ::= indicated geometry <Position>

<RefResolved> ::= orientto pos <Position>
<RefResolved> ::= orientto dir <Direction>
<RefResolved> ::= moveto <Coordinates>
<RefResolved> ::= beamto pos <Position>
<RefResolved> ::= beamto watch <Position>
<RefResolved> ::= exacenter set <Position>
<Coordinates> ::= pos <Position>
<Coordinates> ::= ori <Orientation>
<Coordinates> ::= posori <Position> <Orientation>
<Coordinates> ::= posdir <Position> <Direction>
<Coordinates> ::= dir <Direction>

<Position>    ::= <FLOAT_VALUE> <FLOAT_VALUE> <FLOAT_VALUE>
<Orientation> ::= <FLOAT_VALUE> <FLOAT_VALUE> <FLOAT_VALUE> <FLOAT_VALUE>
<Direction>   ::= <Position>

# --- Steuer-Kommandos -------------------------------------------------
<ControlCmd> ::= mode set <Mode> | mode restrict <DOFs>
<ControlCmd> ::= stepsize <IncDec> | stepsize set <FLOAT_VALUE>
<ControlCmd> ::= speed <IncDec> | speed set <FLOAT_VALUE>
<ControlCmd> ::= light <OnOff>
<ControlCmd> ::= viewpoint <PrevNext> | viewpoint tour
<ControlCmd> ::= viewpoint set <INT> | viewpoint set <STRING>
<ControlCmd> ::= collision <OnOff> | gravity <OnOff>
<ControlCmd> ::= straighten | balance | select
<ControlCmd> ::= repeat | undo
<ControlCmd> ::= sendstatus <StatusGroups>
<ControlCmd> ::= quit
```

```
# --- Zusätzliche Hilfsvariablen ---------------------------------------
<Mode>          ::= walk | fly | examine
<IncDec>        ::= inc | dec | reset
<OnOff>         ::= on | off | toggle
<PrevNext>      ::= prev | next | reset
<StatusGroups> ::= all | stepsize | mode | light
<StatusGroups> ::= viewpoint | collision | gravity
<Walk_Fly_Ex>   ::= walk | fly | examine
<DOFs>          ::= <DOF> <DOFs> | end_list
<DOF>           ::= x | y | z
                ::= yaw | pitch | roll
                ::= phi | omega | radius | rho

# --- Statusänderungen -------------------------------------------------
<STATUS>        ::= status <StatusMsg>
<StatusMsg>     ::= mode mode <Walk_Fly_Ex>
<StatusMsg>     ::= mode continous <On_Off>
<StatusMsg>     ::= mode restricted <DOFs>
<StatusMsg>     ::= stepsize <FLOAT_VALUE>
<StatusMsg>     ::= speed <FLOAT_VALUE>
<StatusMsg>     ::= light <On_Off>
<StatusMsg>     ::= collision <On_Off>
<StatusMsg>     ::= gravity <On_Off>
<StatusMst>     ::= viewpoint activated <INT> <INT> <STRING>
<StatusMst>     ::= viewpoint list_changed <INT> <Viewpoints>
<StatusMsg>     ::= isbeaming <On_Off>
<StatusMsg>     ::= collided <CollisionInfo>
<StatusMsg>     ::= undoing nothingstored
<StatusMsg>     ::= undoing <UnDoType>
<UndoType>      ::= light | collision | gravity
                ::= mode | position | viewpoint
```

ANHANG C

Einführung in VRML

Dieses Kapitel enthält eine kompakte Einführung in die grundlegenden Elemente und Verarbeitungsmechanismen der VRML Sprachtechnologie. Darüber hinaus werden das Ausführungsmodell zur Interpretation von beliebigen VRML-Dateien, der ereignisbasierte Informationsaustausch und die bereits eingebauten Interaktionsparadigmen erläutert. Ausführliche Informationen zu VRML finden sich beispielsweise in [32, 58] sowie in einer Vielzahl von Tutorials mit interaktiven Beispielen, die direkt über das Internet verfügbar sind.

C.1 Technologische Grundlagen

Die Abkürzung VRML steht für Virtual-Reality Modeling Language [72]. Es handelt sich um eine Sprache, durch die beliebige dreidimensionale Objekte in einer virtuellen Welt spezifiziert werden können. Darüber hinaus ermöglicht VRML die Beschreibung unterschiedlicher Navigations-, Selektions- und Manipulationsmechanismen. Gegenüber der ursprünglichen Form gestattet der aktuelle Standard VRML97 auch die Definition von interaktiven Beziehungen zwischen verschiedenen Objekten, multimediale Zusatzeffekte sowie die Festlegung grundsätzlicher Szene-Eigenschaften wie Gravitation und Kollisionsdetektion. Die Objekt- und Szenenmodellierungen werden in Form von ASCII-Daten abgelegt. Ein weiterer Teil der VRML-Spezifikation beschreibt den Umgang mit diesen Daten. In Anlehnung an Softwaremodule zur Interpretation des HTML-Standards bezeichnet man die programmtechnischen Komponenten zur Verarbeitung von VRML-Daten auch als *Browser*. Ein solcher Browser stellt eine eigenständige, von der konkreten Anwendung logisch unabhängige, visuelle Präsentationseinheit dar.

Ein VRML-Browser besitzt ein Software-Interface für externe Module. Diese Programmierschnittstelle wird als *External Authoring Interface (EAI)* bezeichnet und erlaubt den anderen Software-Komponenten, entweder die Szene direkt zu verändern oder auf Ereignisse zu reagieren, die innerhalb der simulierten Welt auftreten. Über diese Module lassen sich applikationsspezifische Funktionen realisieren. Auf Webseiten werden dafür häufig Java-Applets verwendet. Typischerweise wird eine Szenenbeschreibung zusammen mit entsprechenden Texturen, Audio- und Videodaten auf einem Server abgelegt. Als offener Standard für den Gebrauch im Internet unterstützt die VRML-Spezifikation den Datenaustausch zwischen verschiedenen Plattformen und eine dezentrale Speicherstruktur der einzelnen Daten. In Bezug auf Grafikformate und Übertragungsprotokolle wird auf existierende Technologien zurückgegriffen.

```
Background
{  skyAngle    [1.2, 1.57]
   skyColor    [0 0 1, .5 .5 1, .8 .5 .2]
   groundAngle 1.57
   groundColor [.3 .2 .1, .7 .8 .3]
}
NavigationInfo { headlight FALSE }
Transform
{ translation 0 0 -3
  children
  [ DEF Kegel Transform
    { children
      [ Shape
        { geometry Cone
          { bottomRadius 5
            height       8      }
          appearance Appearance
          { material DEF Kegelfarbe Material
            { diffuseColor 1 0 0}
]}]}}}
```

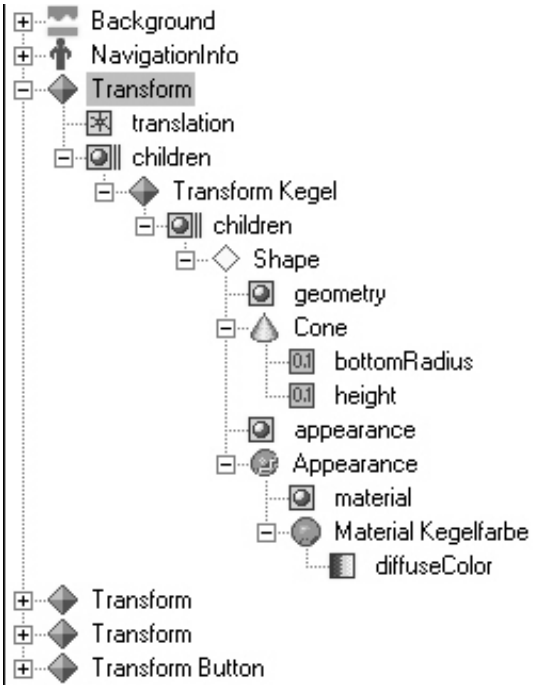


Abb. C.1: Beispiel für eine VRML-Szenenbeschreibung und den dazugehörigen Szenengraph

C.2 Wesentliche Sprachelemente

Der VRML-Standard lässt sich formal auf eine deklarative Sprache abbilden. Diese Sprache besteht im Wesentlichen aus *Knoten*, *Feldern* und *Werten*, die durch einen streng hierarchisch organisierten *Szenengraphen* zueinander in Beziehung gesetzt werden. Dabei stellen die Knoten die zentralen Strukturelemente dar. Ein Knoten verfügt jeweils über eine gewisse Anzahl an kontextspezifisch belegbaren Feldern. Auf ein Feld folgt entweder eine Reihe von Werten oder ein hierarchisch untergeordneter Knoten.

Eine Szenenbeschreibung in VRML ist ein gerichteter, zyklenfreier Graph, der im Allgemeinen aus mehreren, nicht notwendigerweise zusammenhängenden Bäumen besteht. In dieser Baumstruktur wird ein Knoten in Abhängigkeit von seiner Position gemäß der englischen Fachtermini entweder als `Parent` oder als `Child` bezeichnet. Durch spezielle Konstrukte können identische Teilbäume an verschiedenen Stellen der Graphenstruktur wiederverwendet werden. Die Blätter des Graphen beschreiben die visuelle Darstellung einzelner Objekte, deren geometrische Form, akustische Signale und den Zusammenhang mit anderen Objekten. Abbildung C.1 enthält auf der linken Seite ein Beispiel für ein Textfragment für die abstrakte Beschreibung einer Szene und auf der rechten Seite den dazugehörigen Szenengraph.

Der Szenengraph dient vor allem dazu, zwischen den einzelnen Knoten eine Transformationshierarchie aufzubauen. Jeder Knoten verfügt über ein lokales, knotenspezifisches Koordinatensystem. Standardmäßig wird dieses Koordinatensystem zunächst von dem übergeordneten `Parent`-Knoten übernommen. Ausgewählte Knoten können die Position, Orientierung und Skalierung der assoziierten Objekte verändern und das modifizierte Koordinatensystem an die untergeordneten Knoten vererben. Auf diese Weise lassen sich beliebige Koordinatentransfor-

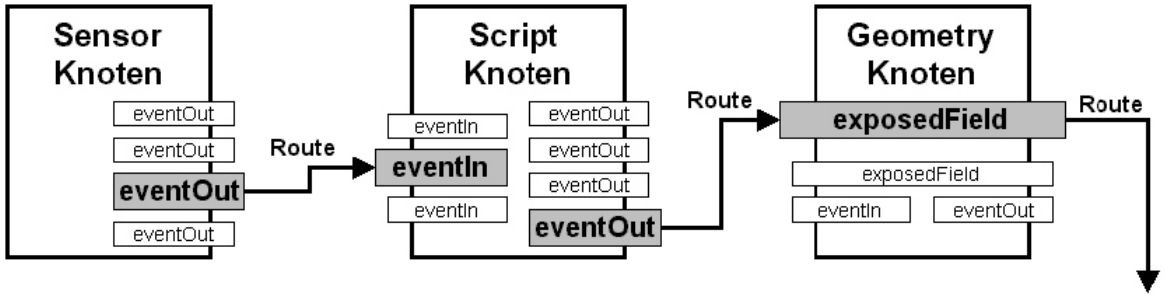


Abb. C.2: Abstrakte Darstellung des Routing-Mechanismus, mittels dem einzelne Informationsanteile zwischen unterschiedlichen Knoten in einer VRML-Szene ausgetauscht werden können

mationen realisieren. Ein wesentlicher Vorteil dieser impliziten Modellierung ist, dass einzelne Objekte unabhängig von ihrer späteren Verwendung erzeugt und mit Teilkomponenten von anderen Objekten kombiniert werden können. Die Bewegung eines Objektes resultiert in VRML aus der zeitabhängigen Änderung der einzelnen Koordinatentransformationen.

Zur Verwirklichung großer virtueller Szenarien ist es möglich, den zugrunde liegenden VRML-Code in mehrere Dateien aufzuspalten sowie externe Objektdefinitionen in die simulierte Welt einzubinden. Mittels des `INCLUDE`-Mechanismus können komplette VRML-Dateien als Knoten in eine andere Welt integriert werden. Dies stellt eine einfache Möglichkeit dar, auf bereits existierende Objekte zuzugreifen. Ein Nachteil besteht darin, dass Modifikationen an den eingebundenen Objekten nur in der Original-Datei vorgenommen werden können. Der `PROTO`-Mechanismus ermöglicht dagegen eine größere Flexibilität. Einzelne Objekte können als `PROTOs` definiert und mit einer Schnittstelle versehen werden, über die die Eigenschaften des Objektes festgelegt werden können. Dieser deutlich weiterführende Ansatz ermöglicht eine modulare Verwendung von Objekten sowie eine dynamische Modifikation einzelner Eigenschaften.

C.3 Ereignisbasierter Informationsaustausch

Neben dem Szenengraph existiert in der VRML-Spezifikation mit dem *Routengraphen* eine weitere, gerichtete Graphenstruktur, über die verschiedene Knoten Informationen austauschen können. Eine Kante dieses Graphen wird auch als *Route* bezeichnet. Durch eine Route werden immer zwei kompatible Felder miteinander verbunden. Diese Verbindung stellt die Grundlage für die Übertragung von Ereignissen innerhalb einer Szene dar.

Für die Erkennung und Verarbeitung der unterschiedlichen Ereigniskategorien existieren jeweils spezielle Sensoren. Zeitabläufe können beispielsweise über einen `TimeSensor`, Interaktionen des Benutzers über einen `ButtonSensor` erfasst werden. In Abhängigkeit von der Position des virtuellen Avatars können durch den `VisibilitySensor` und den `ProximitySensor` darstellungsrelevante Einstellungen vorgenommen werden. Bei einer Aktivierung sendet ein Sensor-Knoten über das `eventOut`-Feld ein entsprechendes Ereignis aus, das von dem Zielknoten über das `eventIn`-Feld aufgenommen wird. Mit `exposedFields` können Ereignisse gleichzeitig gesendet und empfangen werden. Der Routing-Mechanismus ist in Abbildung C.2 anhand eines einfachen Beispiels dargestellt.

Um auch komplexere Ereignisbehandlungen modellieren zu können, existieren in VRML sogenannte *Scriptknoten*. Diese Knoten repräsentieren selbst keine Objekte, sondern implementieren die Auswertungen der übertragenen Ereignisse. Ohne Scriptknoten wären nur extrem einfache Ereignisse wie z.B. das Starten oder Stoppen von vorgegebenen Aktionen realisierbar. Ein Scriptknoten besteht jeweils aus zwei Teilen. Im Kopf werden die Ereignisse mit den dazugehörigen Datentypen deklariert und der Rumpf enthält die eigentliche Verarbeitungslogik. Als Skriptsprache wird zumeist JAVA oder JAVAScript verwendet.

Einige Knoten beschreiben globale Darstellungscharakteristika. Der `Viewpoint`-Knoten gibt beispielsweise die Perspektive an, aus der die simulierte Welt für den externen Betrachter sichtbar wird und über den `Background`-Knoten kann eine generelle Hintergrundfarbe definiert werden. Diese Eigenschaften können auch an verschiedenen Stellen der Szene variieren. Dazu werden die globalen Knoten in einem LIFO-Speicher abgelegt, der durch den Routengraph manipuliert werden kann. Die Aktivierung einer Einstellung, die mit einem speziellen Knoten assoziiert wird, bezeichnet man als das *Binden* eines Knotens.

C.4 Fest eingebaute Interaktionsparadigmen

In der VRML-Spezifikation sind zudem bereits Konzepte für die beiden Interaktionsparadigmen Navigation und Manipulation enthalten. Auf die entsprechenden Umsetzungen wird im Folgenden kurz eingegangen. Kommunikation als eigenständige Interaktionsmetapher wird nur indirekt unterstützt. Der Austausch von Textinformationen kann beispielsweise durch ein spezielles Java-Applet kontrolliert werden.

Das Interaktionsparadigma der Manipulation wird in VRML durch eine Reihe von speziellen Sensor-Knoten realisiert, die die Benutzerinteraktionen mit einzelnen Objekten erfassen. Diese Sensoren sind ausschließlich auf die Benutzung von Zeigegeräten ausgelegt, mittels derer die betreffenden Objekte referenziert werden. Die Objekt-Bewegungen werden dann in Form von variierenden Koordinaten an die Szene weitergeleitet und dort durch einen `Transform`-Knoten in reale Bewegungen umgesetzt. Der `CylinderSensor` überträgt Bewegungen eines Eingabegerätes auf eine rotatorische Bewegung um eine vorgegebene Achse. Demgegenüber lassen sich durch den `PlaneSensor` beliebige Rotationen in einer Ebene realisieren. Mit dem `SphereSensor` lassen sich schließlich beliebige Drehungen um einen Punkt beschreiben.

In diesem Zusammenhang hat der `TouchSensor` die Aufgabe, immer dann ein binäres Signal auszulösen, wenn ein Objekt durch ein Zeigegerät aktiviert wird. Dieser Knoten fungiert als Schalter für die Initiierung einer Manipulationssequenz. Der `Anchor`-Knoten führt keine neue Interaktionsmöglichkeit ein. Er ist vielmehr ein Komfortknoten, der die Schalter-Funktionalität des `TouchSensor`-Knotens um die Möglichkeit erweitert, nach dem Auslösen des Ereignisses einen Hyperlink im Internet aufzurufen oder einen speziellen Aussichtspunkt anzuspringen.

Darüber hinaus existieren verschiedene Sensoren, die durch die Bewegungen des Benutzers ausgelöst werden. Der `Collision`-Knoten stellt fest, wenn der virtuelle Avatar mit Objekten der Szene kollidiert. Mit dem `ProximitySensor` kann die Position und die Orientierung des Avatars festgestellt werden, wobei der relevante Reaktionsbereich einstellbar ist. Zudem zeigt dieser Sensor an, wenn der Avatar in diesen Bereich eintritt bzw. ihn wieder verlässt. Durch den `VisibilitySensor` kann schließlich überprüft werden, ob von der aktuellen Position der virtuellen Kamera ein spezifisches Objekt vollständig sichtbar ist.

In Bezug auf die Umsetzung des Navigationsparadigmas macht die VRML-Spezifikation keine genauen Angaben. Aus diesem Grund finden sich oftmals hochgradig browserspezifische Realisierungen. Systemtechnisch stehen für die Einschränkung der Bewegungsfreiheit im Wesentlichen zwei Knoten zur Verfügung. Durch zwei Felder am `Viewpoint`-Knoten können die Position und die Blickrichtung des virtuellen Betrachters relativ zu einem Navigationskoordinatensystem angegeben und modifiziert werden. Der Knoten `NavigationInfo` bietet dem VRML-Autor zusätzlich die Möglichkeit, verschiedene Parameter, wie etwa die Größe des virtuellen Avatars, die nominelle Navigationsgeschwindigkeit oder den Navigationsmodus zu beeinflussen.

Symbolverzeichnis

Allgemeine Bezeichner

A	systemspezifische Aktion $A \in \mathcal{A}$
$\mathcal{A}$	Menge aller systemspezifischen Aktionen
B	spezifischer Benutzer $B \in \mathcal{B}$
$\mathcal{B}$	Menge von zusammengehörigen Benutzern
$\mathcal{C}$	Cluster, Intervall; Menge von zusammengehörigen Ereignissen
D	Matrix für die kontextsensitive Interpretation unterschiedlicher Ereignisse
E	abtraktes Ereignis, Event, $E \in \mathcal{E}$
$\mathcal{E}$	Menge aller Ereignisse
F	Komponente des Fehlermanagement-Moduls
$\mathcal{G}$	Zusammenfassung einzelner Gene zu einem Chromosom
g	atomare genetische Informationseinheit
H	Aktionshypothese, Ergebnis eines Integrationsprozesses, $H \in \mathcal{H}$
$\mathcal{H}$	Menge aller Aktionshypothesen
$\mathcal{I}$	Individuum in einer Population von Lösungen, $\mathcal{I} \in \mathcal{P}$
J	Spezifisches Auswertungsszenario im Hinblick auf den Spotting-Prozess
$\mathcal{J}$	Menge von zusammengehörigen Auswertungsszenarien
K	Kontextkategorie
$\mathcal{K}$	Menge aller Kreuzungspunkte für den Crossover-Prozess
L	Lookup-Tabelle
$\mathcal{L}$	formale Sprache, zumeist durch eine kontextfreie Grammatik definiert
M	einzelnes Modul bzw. Komponente in der multimodalen Systemarchitektur
$\mathcal{M}$	Menge aller angeschlossenen Systemmodule
N	freie Mengenangabe, speziell im Kontext der Benutzerereignisse
$\mathcal{P}$	Population von Lösungsalternativen
p	Hilfsvariable zur Berechnung der genetischen Fitness
$Q(H)$	quantitative Bewertung für eine Aktionshypothese H
q	Zählidentikator zur Identifikation einer speziellen Generation

R	Erkennermodul
$\mathcal{R}$	Menge von genetischen Positionen im Rekombinations-Prozess
r	Kreuzungspunkt oder Mutationsposition im rekombinations-Prozess
S	Semun; unterscheidbare, nicht weiter zerlegbare semantische Basiseinheit, entspricht einem nicht-terminalen Symbol einer kontextfreien Grammatik
T	Zeitdauer, bezogen auf einzelne Ereignisse oder Intervalle
U	spezifische Benutzerklasse, $U \in \mathcal{U}$
$\mathcal{U}$	Menge der unterschliedlichen Benutzerklassen, $\mathcal{U} = \{ANF, NOR, EXP, AVP\}$
u	Überlagerungsfaktor im Spottingverfahren
$\mathcal{V}_i$	proprietäres Vokabular der Funktionalitäten des Systemmoduls M_i
v	spezifische Funktionalität, $v \in \mathcal{V}$
W_i	angepasstes Kovertermodul (Wrapper) für die Systemkomponente M_i
w	Wort einer kontextfreien Sprache, $w \in \mathcal{L}$
X	kontextspezifisches Integrationstoken, Meta-Ereignis
$\mathcal{X}$	Menge aller kontextspezifischen Integrationstokens
Y	Aufgabe
$\mathcal{Y}$	Menge von zusammengehörigen Aufgaben
Z	kontextspezifische Zustandsvariable
$\mathcal{Z}$	Menge der Zustandsvariablen einer spezifischen Kontextkategorie

Spezielle Bezeichner aus dem griechischen Alphabet

α	Anzahl der möglichen Systemaktionen, $\alpha := \lvert\mathcal{A}\rvert$
β	Gewichtungsfaktoren
Γ	Spotting-Regel
γ	Anzahl der Elemente in einem spezifischen Cluster, $\gamma := \lvert\mathcal{C}\rvert$
ϵ	leeres Wort einer formalen Sprache $\mathcal{L}$
ζ	Anzahl der Zustände in einer spezifischen Kontextkategorie
η	Anzahl aller potenziell auftretenden Ereignisse, $\eta := \lvert\mathcal{E}\rvert$
Θ	Konfiguration der genetischen Strukturparameter
θ	zeitlicher Gewichtungsfaktor
ι	Anzahl der Individuen in einer Ausgangspopulation, $\iota := \lvert\mathcal{P}\rvert$
κ	Anzahl der Kreuzungspunkte im Crossover-Prozess bezogen auf ein Individuum
λ	Anzahl der Aktionshypothesen in einer spezifischen Ebene

μ Anzahl der Gene in einem Individuum

ν Anzahl der Chromosome in einem Individuum

o Anzahl der Individuen nach dem Crossover-Prozess

ξ Anzahl der Mutationspunkte bezogen auf ein Individuum

Π Auswahlfunktion, Selektion einzelner Individuen

π Anzahl der Individuen in einer Paarungspopulation, $\pi := \mathcal{P}_P$

ρ Anzahl der Module in einer spezifischen Kategorie, $\rho(x) := |\mathcal{M}_x|$

π Anzahl der Individuen in einer temporären Selektionspopulation

Σ Alphabet, Menge von unterscheidbaren Symbolen

σ Anzahl der Symbole in einem Alphabet Σ bzw. in einer formalen Sprache $\mathcal{L}$

ς Anzahl der Wörter in einer formalen Sprache, $\varsigma := |\mathcal{L}|$

τ Anzahl der Individuen in einer Tunierselektion

υ Anzahl der proprietären Funktionalitäten eines Moduls, $\upsilon := |\mathcal{V}|$

Φ Intentionsdekoder

ϕ Anzahl der zu mutierenden Individuen in einer Population

φ Mutationsrate im Rekombinationsprozess

χ untere Schranke für die Anzahl an Iterationen in einem Integrationsdurchlauf

ψ Konvergenzrate

Ω kontextsensitive Interpretation eines Ereignisses

ω genetische Fitness zur Bewertung der Qualität einer Lösung

Abkürzungsverzeichnis

AIA	Automotive Infotainment Applikation
ANF	Bezeichnung für die Klasse der Anfänger
ATN	Augmented Transition Network
AVP	Bezeichnung für die Klasse aller Versuchspersonen
BNF	Backus-Naur-Form
CFG	Context-free grammar, kontextfreie Grammatik
CSG	Context-sensitive grammar, kontextsensitive Grammatik
CLIPS	C Language Integrated Production System
DVA	Desktop Virtual-Reality Applikation
EBNF	Erweiterte Backus-Naur-Form
EXP	Bezeichnung für die Klasse der Expertennutzer
FERMUS	Fehlerrobuste Multimodale Sprachdialoge
GUI	Graphical User Interface
KLM	Keystroke-Level Model
LUT	Lookup-Tabelle
MIVIS	Multimodal Interaction in Virtual Scenarios
NOR	Bezeichnung für die Klasse der normalen Nutzer
RTN	Recursive Transition Network
SOMMIA	Sprachorientiertes MMI im Automobil
STD	State Transition Diagram, Zustandsübergangsdiagramm
UsaWiz	Usability Wizard, semi-automatische Unterstützung von Benutzerstudien
VP(en)	Versuchsperson(en)

Literaturverzeichnis

[1] *International Conference on Human Factors in Computing Systems (CHI)*, The Association for Computing Machinery (ACM), ACM SIGCHI (New York), 1983-2003.

[2] *International Conference on Human Computer Interaction (HCI)*, 1984-2003.

[3] *International World Multi-Conference on Systemetics, Cybernetics and Informatics (SCI)*, The International Institute of Informatics and Systemics (IIIS), 1995-2003.

[4] Aiglstorfer, G., A. Haase und C. Lewis: *EMMI: Evaluation of multimodal interaction*. Interdisziplinäres Projekt, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2001.

[5] Al-Hames, M.: *Implementierung und Evaluierung eines genetischen Algorithmus für die multimodale Integration*. Studienarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, Januar, 2003.

[6] Al-Hames, M.: *Design of a Genetic Algorithm for Multimodal Integration*. Bachelor-Thesis, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, November, 2001.

[7] Alander, J. T.: *An Indexed Bibliography of Genetic Algorithms*. In: Chambers, L. (Hrsg.): *Practical Handbook of Genetic Algorithms: New Frontiers*. CRC Press, 1995.

[8] Althoff, F., M. Al-Hames, G. McGlaun und M. Lang: *Towards a New Approach for Integrating Multimodal User Input Based on Evolutionary Computation*. In: *Proceedings ICASSP 2002*. IEEE Signal Processing Society, Mai 2002, Orlando, USA.

[9] Althoff, F., K. Geiss, G. McGlaun, B. Schuller und M. Lang: *Experimental Evaluation of User Errors at the Skill-Based Level in an Automotive Environment*. In: *Int. Conf. on Human Factors in Computing Systems CHI 02*, S. 782–783. ACM SIGCHI (New York), April 2002, Minneapolis, USA.

[10] Althoff, F., G. McGlaun und M. Lang: *Using Multimodal Interaction to Navigate in Arbitrary Virtual VRML Worlds*. In: *Workshop on Perceptive User Interfaces*. ACM Digital Library, November 2001. Orlando, XUSA.

[11] Althoff, F., G. McGlaun, M. Lang und G. Rigoll: *Evaluating multimodal interaction patterns in various application scenarios*. In: *Proc. of Gesture Workshop 2003*, April 2003, Genua, Italien.

[12] Althoff, F., G. McGlaun, M. Lang und G. Rigoll: *A real-time demonstrator for video-based recognition of dynamic head gestures, using discrete Hidden Markov Models*. In: *Proc. of World Conference on Systemetics, Cybernetics and Informatics*, Juli 2003, Orlando, USA.

[13] Althoff, F., G. McGlaun, M. Lang und G. Rigoll: *Comparing an innovative 3D and a standard 2D user interface for automotive infotainment applications*. In: *Proc. Mensch Computer 2003*, September 2003, Stuttgart, Germany.

[14] Althoff, F., G. McGlaun, B. Schuller, M. Lang und G. Rigoll: *Evaluating Misinterpretations during Human-Machine-Communication in Automotive Environments*. In: *Proc. of World Conference on Systemetics, Cybernetics and Informatics*, S. 181–185, Juli 2002, Orlando, USA.

[15] Althoff, F., G. McGlaun, G. Spahn und M. Lang: *Combining Multiple Input Modalities for Virtual Reality Navigation - A User Study*. In: *9th Int. Conf. on Human Computer Interaction (HCI2001)*, S. 47–49. Lawrence Erlbaum Ass. (New Jersey), August 2001, New Orleans, USA.

[16] Althoff, F., H. Stocker, G. McGlaun und M. Lang: *A Generic Approach for Interfacing VRML Browsers to Various Input Devices and Creating Custamizable Applications*. In: Wagner, M. und M. Beitler (Hrsg.): *Tagungsband Web3D 2002 Symposium*, S. 67–74. ACM SIGGRAPH, Februar 2002, Tempe, USA.

[17] Althoff, F., T. Volk, G. McGlaun und M. Lang: *A Generic User Interface Framework for Virtual Reality Applications*. In: *9th Int. Conf. on Human Computer Interaction (HCI2001)*, S. 52–55. Lawrence Erlbaum Ass. (New Jersey), August 2001, USA.

[18] Applegate, D., R. Bixby, V. Chvatal und W. Cook: *Finding cuts in the TSP: a preliminary report*. In: *Report 95-05*. DIMACS, Rutgers University, 1995.

[19] Baeck, T., U. Hammel und H. Schwefel: *Evolutionary computation: comments on the history and current state*. IEEE Transactions on Evolutionary Computation, 1(1):3–17, April 1997.

[20] Baeck, T., J. Heistermann, C. Kappler und M. Zamparelli: *Evolutionary algorithms support refueling of pressurized water reactors*. In: *Proceedings of the 3rd IEEE Conference on Evolutionary Computation*, S. 104–108. IEEE Press, 1996.

[21] Baker, J.: *Adaptive selection methods for genetic algorithms*. In: Grefenstette, J. (Hrsg.): *Proceedings of tje First International Conference on Genetic Algorithms*, S. 101–111. Lawrence Erlbaum Associates, 1987.

[22] Balzert, H.: *Lehrbuch der Software-Technik, Band 1 und 2*. Spektrum Akademischer Verlag.

[23] Beasley, D., D. R. Bull und R. R. Martin: *An Overview of Genetic Algorithms: Part 1, fundamentals*. University Computing, 15(2):58–69, 1993.

[24] Beasley, D., D. R. Bull und R. R. Martin: *An Overview of Genetic Algorithms: Part 2, Research Topics*. University Computing, 15(4):170–181, 1993.

[25] Bengler, K., P. Geutner, B. Niedermaier und F. Steffens: *Eyes free - Hands free oder Zeit der Stille. Ein Demonstrator zur multimodalen Bedienung im Automobil*. DGLR-Bericht 2000-02 der 42. Fachausschusssitzung Anthropotechnik, Multimodale Interaktion im Bereich der Fahrzeug- und Prozessführung, S. 299–307, München 2000.

[26] Benoit, C., J.-C. Martin, C. Pelachaud, L. Schomaker und B. Suhm: *Audio-visual and Multimodal Speech Systems*. In: *Handbook of Standards and Resources for Spoken Language Systems - Supplement Volume*. D. Gibbon, I. Mertins, R.K. Moore (Hrsg.), Kluwer International Series in Engineering and Computer Science 2000.

[27] Berthele, P.-J.: *Videobasierte Kopfgestenerkennung im multimodalen Kontext*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2001.

[28] *Blaxxun Contact VRML Browser*. Blaxxun Interactive, www.blaxxun.com, 2004.

[29] Bolt, R.: *Put that there: Voice and gesture at the graphics interface*. ACM Computer Graphics, 14(3):262–270, 1980.

[30] Card, S., T. Moran und A. Newell: *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, 1983.

[31] Card, S. K., T. P. Moran und A. Newell: *The keystroke-level model for user performance time with interactive systems*. Com. of the ACM, 23(7):396–410, 1980.

[32] Carey, R.: *The Annotated VRML 2.0 Reference Manual*. Addison-Wesley, 1997.

[33] Charniak, E. und D. McDermott: *Introduction to Artificial Intelligence*. Addison-Wesley, 1985.

[34] Cheyer, A. und L. Julia: *Designing, Developing and Evaluating Multimodal Applications*. In: *WS on Pen/Voice Interfaces (CHI 99)*, Pittsburgh 1999.

[35] Cohen, P., R. Coulston und K. Krout: *Multiparty Multimodal Interaction: A Preliminary Analysis*. Proceedings of the International Conference on Spoken Language Processing (ICSLP 2002), Denver, CO, September 2002.

[36] Cohen, P., M. Johnston und D. McGee: *Quickset: Multimodal interaction for distributed applications*. Proceedings of the Fifth ACM International Multimedia Conference, S. 31–40, ACM Press, New York 1997.

[37] Cohen, P. R., M. Johnston, D. McGee, I. Smith, J. Pittman, L. Chen und J. Clow: *Multimodal interaction for distributed interactive simulation*. Menlo Park, CA, USA 1997.

[38] Cox, L. und L. Davis: *Dynamic anticipatory routing in circuit-switched telecommunications networks*. In: *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, 1991.

[39] Darwin, C.: *The origin of species by means of natural selection*. Murray, London, 1859.

[40] Dix, A., J. Finlay, G. Abowd und R. Beale: *Human-Computer Interaction*. Prentice Hall, 1993.

[41] Donahoo, M., K. Calvert, K. L. Calvert und M. J. Donahoo: *TCP/IP Sockets in C: Practical Guide for Programmers*. Morgan Kaufmann Pub., 2000.

[42] Dzida, W.: *Das IFIP-Modell für Benutzungsschnittstellen*. Office Management, Sonderheft 31.

[43] Engelmeier, K.-H. et al.: *Virtual reality and multimedia human-computer interaction in medicine*. IEEE WS on Multimedia Signal Processing, S. 88–97, Los Angeles, December 1998.

[44] Eshelmann, L., R. Caruna und J. Schaffer: *Biases in the crossover landscape*. In: *Proc. of the 3rd International Conference on Genetic Algorithms*. Morgan Kaufmann, 1989.

[45] Fischer, J.: *Multimodale Selektion von Objekten in Virtual-Reality*. Industrie-Praktikum, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, Juli, 2003.

[46] Fogel, D.: *Evolutionary Computation*. IEEE Press, 1995.

[47] Foley, J. D.: *Models and Tools for the Designers of User-Computer Interfaces*. In: Earnshaw, R. A. (Hrsg.): *Theoretical Foundations of Computer Graphics and CAD. Proceedings*, S. 1122–1151, Ciocco, Italy, Juli 1988. NATO ASI Series. Vol. F40.

[48] Foley, J. D. et al.: *Computer Graphics, Principles and Practice*. Addison-Wesley, 1992.

[49] Fourman, M.: *Compaction of symbolic layout using genetic algorithms*. In: Grefenstette, J. (Hrsg.): *Proceedings of the First International Conference on Genetic Algorithms*, S. 141–153. Lawrence Erlbaum Associates, 1985.

[50] Geiger, M.: *Berührungslose Bedienung von Infotainment-Systemen im Fahrzeug*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2003.

[51] Geiser, G.: *Mensch-Maschine-Kommunikation*. Oldenbourg-Verlag, München, 1990.

[52] Goldberg, D.: *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley, 1989.

[53] Goldberg, D. und K. Deb: *A comparative analysis of selection schems used in genetic algorithms*. In: Rawlins, G. (Hrsg.): *Foundations of Genetic Algorithms*, S. 69–93. Morgan Kaufmann, 1991.

[54] Gourdol, A. P. J., L. Nigay, D. Salber und J. Coutaz: *Two Case Studies of Software Architecture for Multimodal Interactive Systems: VoicePaint and a Voice-enabled Graphical Notebook*. In: *Engineering for Human-Computer Interaction*, S. 271–284, 1992.

[55] Green, M.: *Report on Dialogue Specification Tools*. In: Pfaff, G. E. (Hrsg.): *User Interface Management Systems: Proceedings of the Workshop on User Interface Management Systems held in Seeheim*, S. 9–20. Springer-Verlag.

[56] Görz, G.: *Einführung in die künstliche Intelligenz*. Addison-Wesley, 1993.

[57] Hartl, A. und F. Prilmeier: *Bildverarbeitungstechnische Aufbereitung medizinischer PET-Datensätze*. Interdisziplinäres Projekt, Lehrstuhl für Mensch-Maschine-Kommunikation und Nuklearmedizinische Klinik am Klinikum Rechts der Isar, Technische Universität München, 2003.

[58] Hartman, J. und J. Wernecke: *The VRML 2.0 Handbook*. Addison-Wesley, 1996.

[59] Hartson, H. R., A. C. Siochi und D. Hix: *The UAN: a user-oriented representation for direct manipulation interface designs*. ACM Transactions on Information Systems (TOIS), 8(3):181–203, 1990.

[60] Hartung, K., S. Münch, L. Schomaker et al.: *MIAMI: Software Architecture, Deliverable Report 4*. Techn. Ber., Report of the ESPRIT Project 8579 MIAMI, 1996.

[61] Heinecke, A.: *Dialog im Dunkeln: Eine Ausstellung zur Entdeckung des Unsichtbaren*. Speicherstadt Hamburg, Internet-Publikation, www.dialog-im-dunkeln.de, 2004.

[62] *HelloIC Spracherkennungssmodul*. Philips Semiconductors, Internet-Publikation, www.semiconductors.philips.com, 2001.

[63] Hennig, A.: *Die andere Wirklichkeit*. Addison-Wesley, 1997.

[64] Hewett, T., R. Baecker, S. Card, T. Carey, J. Gasen, M. Mantei, G. Perlman, G. Strong und W. Verplank: *Curricula for Human-Computer Interaction*. ACM Special Interest Group on Computer-Human Interaction, Curriculum Development Group (1996).

[65] Hill, R. D.: *Supporting concurrency, communication, and synchronization in human-computer interaction: The Sassafras UIMS*. ACM Transactions on Graphics (TOG), 5(3):179–210, 1986.

[66] Holland, J.: *Adaption in Natural and Artificical Systems*. MIT Press, 1975.

[67] Hüttner, J., H. Wandke und A. Rätz: *Benutzerfreundliche Software - Psychologisches Wissen für die ergonomische Schnittstellengestaltung*. Bernd-Michael-Paschke Verlag, Berlin, 1995.

[68] Hunsinger, J.: *Multimodale Erfassung mathematischer Formeln durch einstufig-probabilistische semantische Decodierung*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2003.

[69] Hunsinger, J. und M. Lang: *A Single-Stage Top-Down Probalistic Approach towards Understanding Spoken and Handwritten Mathematical Formulas*. In: *Proceedings ICSLP 2000, Peking, China, 16.–20.10.2000*, Bd. 4, S. 386–389. China Friendship Publish, 2000.

[70] ISO 13407: *Benutzerorientierte Gestaltung interaktiver Systeme*. International Organization for Standardization, Genf, 1998.

[71] ISO 9241 Part 11: *Ergonomic requirements for office work with visual display terminals - Guidance on usability*. International Organization for Standardization, Genf, 1998.

[72] ISO/IEC 14772-1. Specification of VRML97, www.web3d.org/technicalinfo, 1997.

[73] Janikow, C. und Z. Michalewicz: *An experimental comparison of binary and floating point representations in genetic algorithms*. In: Below, R. und L. Booker (Hrsg.): *Proceedings of the 4th International Conference on Genetic Algorithms*, S. 31–36. Morgan Kaufmann, 1991.

[74] Jensen, K. und N. Wirth: *PASCAL User Manual and Report*. Springer, 1985.

[75] Johannsen, G.: *Mensch-Maschine-Systeme*. Springer-Verlag, Berlin, 1993.

[76] Johansson, H.: *Integrating Multimodal Information*, 2001.

[77] Knuth, D. E.: *backus normal form vs. Backus Naur form*. Communications of the ACM, 7(12):735–736, 1964.

[78] Koons, D., C. Sparrell und K. Thorisson: *Integrating simultaneous input from speech, gaze and hand gestures*. In: Maybury, M. (Hrsg.): *Intelligent Multimedia Interfaces*, S. 257–276. MIT Press, 1993.

[79] Koza, J. R.: *Genetic Programming - On the Programming of Computers by Means of Natural Selection*. MIT University Press, 1998.

[80] Lang, M.: *Institute for Human-Machine Communication: Activity Report 1997-2000*. Techn. Ber., Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2000.

[81] Lang, M., G. McGlaun und F. Althoff: *SOMMIA - Final Report (Speech Oriented MMI Automotive)*. Techn. Ber., Lehrstuhl für Mensch-Maschine-Kommunikation, September 2001. Auftragsforschung für die Siemens-VDO Automotive AG.

[82] Lang, M., G. McGlaun, F. Althoff, B. Schuller und K. Geiss: *FERMUS Phase I (Fehlerrobuste multimodale Sprachdialoge)*. Techn. Ber., Lehrstuhl für Mensch-Maschine-Kommunikation, Juli 2001. Auftragsforschung für die BMW AG, die DaimlerChrysler AG und die Siemens-VDO Automotive AG.

[83] Lang, M., G. McGlaun, F. Althoff, B. Schuller und K. Geiss: *FERMUS Phase II (Fehlerrobuste multimodale Sprachdialoge)*. Techn. Ber., Lehrstuhl für Mensch-Maschine-Kommunikation, Oktober 2002. Auftragsforschung für die BMW AG, die DaimlerChrysler AG und die Siemens-VDO Automotive AG.

[84] Lang, M., G. McGlaun, F. Althoff, B. Schuller und K. Geiss: *FERMUS Phase III (Fehlerrobuste multimodale Sprachdialoge)*. Techn. Ber., Lehrstuhl für Mensch-Maschine-Kommunikation, Juli 2003. Auftragsforschung für die BMW AG, die DaimlerChrysler AG und die Siemens-VDO Automotive AG.

[85] Lang, M. K.: *Institute for Human-Machine Communication: Activity Report 1997-2000*. Techn. Ber., Lehrstuhl für Mensch-Maschine-Kommunikation, Technische Universität München, 2001.

[86] Latoschik, M.: *A General Framework for Multimodal Interaction in Virtual Reality Systems: PrOSA*. In: Broll, W. und L. Schäfer (Hrsg.): *The Future of VR and AR Interfaces - Multimodal, Humanoid, Adaptive and Intelligent. Proceedings of the workshop at IEEE Virtual Reality 2001, Yokohama, Japan*, S. 21–25. GMD Report No. 138, GMD-Forschungszentrum Informationstechnik GmbH, Sankt Augustin, März 2001.

[87] Latoschik, M., B. Jung und I. Wachsmuth: *Multimodale Interaktion mit einem System zur Virtuellen Konstruktion*. Informatik '99, 29. Jahrestagung der Gesellschaft für Informatik, Paderborn, S. 88–97, Oktober 1999.

[88] Lernout und Hauspie: *Lernout and Hauspie - Software Developers Kit*. Lernout and Hauspie: The Speech and Language Company N.V, 1998.

[89] Lo, S.-L. und S. Pope: *The Implementation of a High Performance ORB over Multiple Network Transports*. Techn. Ber., Olivetti and Oracle Research Laboratory, 1998.

[90] Martin, J.: *Towards adequate representation technologies for multimodal interfaces*. Proc. of Int. Conf. on Cooperative Multimodal Communication, 1995.

[91] Martin, J.: *Towards intelligent cooperation between modalities: The example of a system enabling multimodal interaction with a map*. Proc. of IJCAI 97 workshop on intelligent multimodal systems, Nagoya, Japan 1997.

[92] Maybury, M. und W. Wahlster: *Intelligent User Interfaces: An Introduction*. Reading in Intelligent User Interfaces, S. 1–13, 1998.

[93] McGlaun, G.: *Entwicklung einer generischen multimodalen Systemarchitektur zur kontextsensitiven fehlerrobusten Verarbeitung von Nutzerinteraktionen (Arbeitstitel)*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2004. Status: Erstellungsphase.

[94] McGlaun, G., F. Althoff und M. Lang: *Ein neuer Systemansatz für die Integration multimodalen Inputs durch Late Semantic Fusion*. In: *VDI/VDE - GMA Fachtagung USEWARE 2002*, S. 181–185. VDI-Verlag, Juni 2002, Darmstadt, Deutschland.

[95] McGlaun, G., F. Althoff, M. Lang und G. Rigoll: *Towards Multimodal Detection and Classification of Emotional Pattern in Human-Machine Interaction - Results of a Baseline Study*. In: *Proc. of World Conference on Systemetics, Cybernetics and Informatics*, S. 181–185, Juli 2002, Orlando, USA.

[96] McGlaun, G., F. Althoff, M. Lang und G. Rigoll: *Robust video-based Recognition of dynamic head-gestures in various domains - Comparing a rule-based and a stochastic approach*. In: *Proc. of Gesture Workshop 2003*, April 2003, Genua, Italien.

[97] McGlaun, G., F. Althoff, M. Lang und G. Rigoll: *Towards Multimodal Error Management: Experimental Evaluation of User Strategies after System Application Faults in an Automotive Environment*. In: *Proc. of World Conference on Systemetics, Cybernetics and Informatics*, Juli 2003, Orlando, USA.

[98] McGlaun, G., F. Althoff, H.-W. Rühl, M. Alger und M. Lang: *A Generic Operation Concept for an Ergonomic Speech MMI under Fixed Constraints in the Automotive Environment*. In: *9th Int. Conf. on Human Computer Interaction (HCI2001)*, S. 288–290. Lawrence Erlbaum Ass. (New Jersey), August 2001, New Orleans, USA.

[99] McGlaun, G., F. Althoff, B. Schuller und M. Lang: *A new technique for adjusting distraction moments in multi-tasking non-field usability tests*. In: *Int. Conf. on Human Factors in Computing Systems CHI 02*, S. 666–667. ACM SIGCHI (New York), April 2002, Minneapolis, USA.

[100] Mehlhorn, K., S. Näher, M. Seel und C. Uhrig: *The LEDA User Manual, Version 4.2*. Max-Planck-Institut für Informatik, Saarbrücken, Germany, 2002.

[101] Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, Berlin, 1999.

[102] Michalewicz, Z.: *The Spirit of Evolutionary Algorithms*. Journal of Computing nad Information Technology, 7, 1999.

[103] Möller, C.: *3D-Visualisierung und Auswertung von PET-Patientendaten unter Verwendung eines VRML-Modells*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation und Nuklearmedizinische Klinik am Klinikum Rechts der Isar, Technische Universität München, 2003.

[104] Müller, J.: *Die semantische Gliederung zur Repräsentation des Bedeutungsinhalts innerhalb sprachverstehender Systeme*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 1997.

[105] Müller, J. und H. Stahl: *Speech Understanding and Speech Translation in various Domains by Maximum a-posteriori Semantic Decoding*. In: *Proc. EIS 98*, S. 256–267, La Laguna, Spain 1998.

[106] Moran, T.: *The Command Language Grammar*. International Journal of Man-Machine Studies, 15:3–50, 1981.

[107] Morguet, P.: *Stochastische Modellierung von Bildsequenzen zur Segmentierung und Erkennung dynamischer Gesten*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2001.

[108] Morguet, P. und M. Lang: *Spotting Dynamic Hand Gestures in Video Image Sequences Using Hidden Markov Models*. In: *Proceedings of ICIP 97, Chicago, USA*, S. III/193–III/197, 1997.

[109] Morguet, P. und M. Lang: *Comparison of approaches to Continuous Hand Gesture Recognition for a Visual Dialog System*. In: *Proceedings of ICASSP 99*, S. 3549–3552, 1999.

[110] Neal, J. und S. Shapiro: *Intelligent Multi-Media Interface Technology*. In: Sullivan, J. und S. Tyler (Hrsg.): *Intelligent User Interfaces*. ACM Press, New York, 1991.

[111] Neuss, R.: *Usability Engineering als Ansatz zum Multimodalen Mensch-Maschine-Dialog*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, Mai, 2001.

[112] Niedermaier, F. B.: *Entwicklung und Bewertung eines Rapid-Prototyping Ansatzes zur multimodalen Mensch-Maschine-Interaktion im Kraftfahrzeug*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2003.

[113] Nielsen, J.: *Usability Engineering*. Morgan Kaufmann Publishers Inc., 1993.

[114] Nigay, L. und J. Coutaz: *A design space for multimodal systems: concurrent processing and data fusion*. In: *Proceedings of the conference on Human factors in computing systems*, S. 172–178. Addison-Wesley Longman Publishing Co., Inc., 1993.

[115] Nigay, L. und J. Coutaz: *A Generic Platform for Adressing the Multimodal Challenge*. Proc. of CHI '95, ACM Press 1995.

[116] Norman, D. A.: *The invisible Computer*. MIT Press, 1999.

[117] Novak, V.: *Kommunikationsansätze für multimodale Interaktionssysteme*. Interdisziplinäres Projekt, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2003.

[118] Oviatt, S. L.: *Designing the User Interface for multimodal Speech and Pen-based Gesture Applications: State-of-the-Art Systems and Future Directions*. Human Computer Interaction, 15(4):263–322, 2000.

[119] Oviatt, S. L.: *Integration and Synchronization of Input Modes during multimodal human-computer interaction*. Proc. of the 6th Int. Conf. on Spoken Language Processing (ICSLP), Beijing, China 2000.

[120] Oviatt, S. L.: *Multimodal Interface Research: A science without borders*. In: Yuan, B., T. Huang und X. Tang (Hrsg.): *Proc. of the 6th International Conference on Spoken Language Processing (ICSLP 2000)*, Bd. 3, S. 1–6. Chinese Friendship Publishers, 2000.

[121] Oviatt, S. L., A. DeAngeli und K. Kuhn: *Integration and Synchronization of Input Modes during Multimodal Human-Computer Interaction*. In: *Proceedings of Conference on Human Factors in Computings Systems: Chi'97*. ACM Press, 1997.

[122] Oviatt, S. L. und R. vanGent: *Error resolution during multimodal human-computer interaction*. In: Bunnell, T. und W. Idsardi (Hrsg.): *Proc. of the International Conference on Spoken Language Processing*, S. 204–207. University of Delaware Press, 1996.

[123] Pavlovic, V., G. Berry und T. Huang: *BattleView: A Multimodal HCI Research Application*. Workshop on Perceptual User Interfaces (PUI 98), November 1998.

[124] Payne, S. und T. Green: *Task-Action Grammars: A model of the mental representation of task languages*. Human-Computer Interaction, 2(2):93–133, 1986.

[125] Peters, J.: *Design und Evaluierung eines 3D-Interfaces zur Steuerung von Komfortapplikationen im Automobil*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2002.

[126] *PITech Phantom User Interface*. PITech, Internet-Pub., www.pitechnology.com, 2003.

[127] Pope, A.: *The CORBA Reference Guide: Understanding the Common Object Request Broker*. Addison-Wesley Pub, 1998.

[128] Press, W. H., B. P. Flannery, S. A. Teukosky und W. T. Vetterling: *Numerical Recipes in C*. Cambridge Univerity Press, 2nd Aufl., 1992.

[129] Raisamo, R.: *Multimodal human-computer interaction: a constructive and empirical study*. Doktorarbeit, University of Tampere, Department of Computer Science, Finnland, 1999.

[130] Rasmussen, J.: *Information Processing and Human-Machine-Interaction*. Noth Holland, Amsterdam, 1986.

[131] Rechenberg, I.: *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, 1973.

[132] Renz, A.: *Untersuchung von Zeitrelationen im multimodalen Kontext*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2003.

[133] Riley, G.: *CLIPS: An expert system building tool*. Proceeding of the Technology, 2001.

[134] Russel, P. J.: *Genetik - Eine Einführung*. Springer-Verlag, Berlin, 1983.

[135] Sayan, m. F.: *Using emotional context information in multimodal systems*. Master Thesis, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2001.

[136] Scheja, C.: *Multimodale Manipulation in Virtual-Reality*. Bachelor-Thesis, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, Dezember, 2003.

[137] Schmidtke, H. et al.: *Handbuch der Ergonomie mit ergonomischen Konstruktionsrichtlinien und Methoden*, Bd. I-V. Bundesamt für Wehrtechnik und Beschaffung, Koblenz, 2. überarbeitete und erweiterte Auflage, 1989.

[138] Schöning, P. U.: *Theoretische Informatik kurz gefasst*. BI-Wissenschaftsverlag, 1993.

[139] Schoenauer, M. und Z. Michalewicz: *Evolutionary Computation at the Edge of Feasibility*. In: Voigt, H.-M., W. Ebeling, I. Rechenberg und H.-P. Schwefel (Hrsg.): *Parallel Problem Solving from Nature – PPSN IV*, S. 245–254, Berlin, 1996. Springer.

[140] Schoeneburg, E., F. Heinzmann und S. Feddersen: *Genetische Algorithmen und Evolutionsstrategien*. Addison-Wesley, Germany, 1994.

[141] Schomaker, L., J. Nijtmanns, C. Camurri, P. Morasso, C. Benoit et al.: *A Taxonomy of multimodal interaction in the human information processing system*. Techn. Ber., ESPRIT III: Basic Research Project 8579, 1995. Multimodal Interface for Advanced Multimedia Interfaces (MIAMI).

[142] Schuller, B., F. Althof, G. McGlaun, M. Lang und G. Rigoll: *Towards Automation of Usability Studies*. In: A. El Kamel, K. Mellouli, P. B. (Hrsg.): *CD-ROM-Proceedings IEEE International Conference on Systems, Man and Cybernetics, SMC 2002*, Oktober, Tunesien 2002.

[143] Schuller, B., F. Althoff, G. McGlaun und M. Lang: *Navigation in Virtual Worlds via Natural Speech*. In: *9th Int. Conf. on Human Computer Interaction (HCI2001)*, S. 19–21. Lawrence Erlbaum Ass., August 2001, New Orleans, USA.

[144] Schuller, B., G. Rigoll und M. Lang: *Hidden Markov Model-based Speech Emotion Recognition*. Proceedings of Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 03), 2003.

[145] Schuller, B., G. Rigoll und M. Lang: *Sprachliche Emotionserkennung im Fahrzeug*. In: *45. Fachausschusssitzung Anthropotechnik, Entscheidungsunterstützung für die Fahrzeug- und Prozessführung*, S. 227–240. DGLR Bericht 2003-04, Universität der Bundeswehr München, 2003.

[146] Schuller, B., G. Rigoll und M. Lang: *Speech Emotion Recognition Combining Acoustic Features and Linguistic Information in a Hybrid Support Vector Machine - Belief Network Architecture*. Proceedings of Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 04), 2004.

[147] Schwefel, H.: *Numerical optimization of computer models*. Wiley&Sons, Chichester, 1981.

[148] Shannon, C. E.: *A mathematical theory of communication - Part I*. Bell System Technical Journal, S. 379–423, Juli 1948. vol. 27.

[149] Shannon, C. E.: *A mathematical theory of communication - Part II*. Bell System Technical Journal, S. 623–656, Oktober 1948. vol. 27.

[150] Shneiderman, B.: *Direct Manipulation: A Step Beyond Programming Languages*. IEEE Computer, 16(8):57–69, August 1983.

[151] Siegel, J.: *CORBA 3 Fundamentals and Programming, 2nd Edition*. John Wiley and Sons, 2000.

[152] Singer, M.: *Konzeption eines Systems zur kontextabhängigen Fehlerdialogführung im Automobil*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2002.

[153] Sommerville, I.: *Software Engineering*. Addison-Wesley, 5. Aufl., 1996.

[154] Sowa, T., M. Fröhlich und M. E. Latoschik: *Temporal Symbolic Integration Applied to a Multimodal System Using Gestures and Speech*. In: *Gesture Workshop*, S. 291–302, 1999.

[155] Sowa, T. und I. Wachsmuth: *Coverbal Iconic Gestures for Object Descriptions in Virtual Environments: An Empirical Study*. Post-Proc. of Int. Conf on Gestures: Meaning and Use, Portugal 2000.

[156] Spahn, G.: *Natürlichsprachliche Navigation in virtuellen Welten*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2000.

[157] Stahl, H.: *Konsistente Integration stochastischer Wissensquellen zur semantischen Decodierung gesprochener Äußerungen*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 1997.

[158] Streit, A.: *Ein Algorithmus zur Lang- und Kurzzeitadaption im multimodalen Mensch-Maschine-Dialog in der Fahrzeugdomäne*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2003.

[159] Suhm, B.: *Empirical evaluation of interactive multimodal error correction*. In: Furui, S., B. Jang und W. Chou (Hrsg.): *IEEE Workshop on Automatic Speech Recognition and Understanding*, S. 583–590, Santa Barbara (CA), USA 1997.

[160] Suhm, B., B. Myers und A. Waibel: *Model-based and empirical evaluation of multimodal interactive error correction*. Proceedings of CHI 99, S. 584–591, May 1999.

[161] Tomassini, M.: *Evolutionary Algorithms*. In: Sanchez, E. und M. Tomassini (Hrsg.): *Towards Evolvable Hardware; The Evolutionary Engineering Approach*, S. 19–47, Berlin, 1996. Springer.

[162] Wachsmuth, P. I.: *Informatik IV (Theoretische Informatik)*. Skript zur Vorlesung im Sommersemester 1994, Universität Bielefeld, 1994.

[163] Wahlster, W.: *User and Discourse Models for Multimodal Communication*. In: Sullivan, J. und S. Tyler (Hrsg.): *Intelligent User Interfaces*. ACM Press, New York, 1991.

[164] Wahlster, W., N. Reithinger und A. Blocher: *EMBASSI: Multimodal Assistance for Infotainment and Service Infrastructures*. In: Wolf, G. und G. Klein (Hrsg.): *Proceedings der Statustagung der Leitprojekte zum Thema Mensch-Technik-Interaktion*, S. 35–44. Projektträger des BMBF für Informationstechnik, Saarbrücken, 2001.

[165] Wahlster, W., N. Reithinger und A. Blocher: *SmartKom: Towards Multimodal Dialogs with Anthropomorphic Interface Agents*. In: Wolf, G. und G. Klein (Hrsg.): *Proceedings der Statustagung der Leitprojekte zum Thema Mensch-Technik-Interaktion*. Projektträger des BMBF für Informationstechnik, Saarbrücken, 2001.

[166] Waibel, A., M. T. Vo, P. Duchnowski und S. Manke: *Multimodal Interfaces*. Artifical Intelligence Review, 10(3-4):299–319, 1996.

[167] Waitzhofer, F.: *Fehlerprävention und Fehlerbehandlung bei multimodaler Interaktion im Fahrzeug*. Diplomarbeit, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 2003.

[168] Weber, W., N. Avril und M. Schwaiger: *Relevance of Positron Emission Tomography (PET) in Oncology*. Techn. Ber., Nuklearmedizinische Klinik der Technischen Universität München, Aktuelles Forum, 1999.

[169] Wiener, N.: *Cybernetics or Control and Communication In the Animal and the Machine*. MIT Press, Cambridge, 1948.

[170] Winkler, H.-J.: *Entwurf und Realisierung eines auf statistischen Ansätzen basierenden Systems zur Erkennung handgeschriebener mathematischer Formeln*. Dissertation, Lehrstuhl für Mensch-Maschine-Kommunikation, Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, 1996.

[171] Wirth, N.: *Compiler Construction*. Addison-Wesley, 1996.

[172] Wu, L., S. L. Oviatt und P. R. Cohen: *Multimodal integration - a statistical view*. IEEE Transactions on Multimedia, 1(4):334–341, December 1999.

[173] *FreeWRL Home Page*. Internet-Pub., www-ext.crc.ca/FreeWRL, 2004.

[174] Zeller, A., A. Wagner und M. Spreng: *iDrive - Zentrale Bedienung im neuen 7er von BMW*. Elektronik im Kraftfahrzeug, VDI-Bericht, 2001.

[175] Zobl, M. et al.: *A usability-study on hand-gesture controlled operation of in-car devices*. In: *9th Int. Conf. on Human Computer Interaction (HCI2001)*. Lawrence Erlbaum Ass. (New Jersey), August 2001, New Orleans, USA.

[176] Zobl, M., M. Geiger, B. Schuller, G. Rigoll und M. Lang: *A Realtime System for Hand-Gesture Controlled Operation of In-Car Devices*. In: *Proceedings ICME 2003, Baltimore, MD, USA*, Bd. 3, S. 541–544. IEEE Multimedia Human Machine Interface and Interaction, 2003.