Natürlichsprachliche Bedienung technischer Systeme

Bernd Ludwig

Lehrstuhl für Künstliche Intelligenz
Friedrich-Alexander-Universität Erlangen-Nürnberg
Bernd.Ludwig@informatik.uni-erlangen.de

Abstract: Komplexe elektronische Systeme spielen eine immer größere Rolle im Alltag. Videorekorder zählen dabei ebenso zu solchen Systemen wie Navigationssysteme im Auto oder sehr fortgeschrittene Softwareprogramme, die auf den neuesten Computern installiert sind. Da die Elektronik immer leistungsfähiger wird, und Softwarekomponenten die Kontrolle übernehmen, steigen auch ständig die Anforderungen an Benutzer bei der Bedienung derartiger Systeme. Im Normalfall wird der überwiegende Anteil der angebotenen Funktionen sehr selten benutzt; eine Konsequenz aus dieser Tatsache ist, dass Benutzer meist nicht wissen, wie und zu welchem Zweck bestimmte Systemfunktionen aktiviert werden können.

1 Bedienschnittstellen für technische Geräte

1.1 Ein Bedienvorgang aus Anwendersicht

In Abbildung 1 ist eine beispielhafte natürlichsprachliche Interaktion zwischen einem Nutzer und seinem TV-Gerät zu sehen; bei der *menügeführten Bedienung* werden letztlich die Funktionen der Knöpfe auf der Fernbedienung durch Kommandowörter ersetzt; statt zu

Menügeführte Bedienung

Welches Gerät wollen Sie bedienen?
– Fernseher
Welche Funktion wollen Sie aktivieren?
– Programmliste
Zu welcher Uhrzeit suchen Sie Sendungen?
– jetzt
Welches Genre?
– Nachrichten *Wählen Sie einen Kanal aus!*
– Kanal 2

„Spontane“ Bedienung

Ich möchte jetzt Nachrichten sehen.

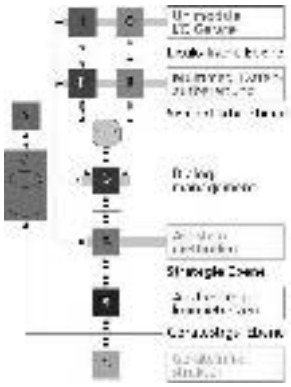

Abbildung 1: Menübasierte versus spontane Bedienung

Taste	Kontext	Wirkung	Zweck
Fensterheber	Tür zu Zündung aus	hoch oder runter (ruckweise)	andere Fensterstellung
	Tür zu Zündung an, kurzer Druck	Aktivierung der Automatik (hoch oder runter)	
	Tür zu, Zündung an	hoch oder runter (fließend)	

Abbildung 2: Ein elektrischer Fensterheber kann unterschiedliche Wirkungen haben.

wissen, welche Knöpfe er in welcher Reihenfolge drücken muss, um Nachrichten sehen zu können, muss er eben Schlüsselwörter lernen, damit ihn das TV-Gerät „verstehen“ kann.

Die Wirkung einzelner Schlüsselwörter hängt jedoch oft auch vom aktuellen Gerätezustand ab – genauso wie Bedienknöpfe je nach Gerätezustand unterschiedliche Wirkungen haben. Abbildung 2 stammt aus der Bedienungsanleitung eines aktuellen PKW-Modells und gibt einen Überblick darüber, was der Fahrer mit dem Fensterheber alles bewirken kann. Fraglich ist dabei jedoch, ob jeder Fahrer überhaupt weiß, welche Funktionen er mit dem Fensterheber aktivieren kann und ob jedem Fahrer immer zur rechten Zeit einfällt, was er tun muss. Wer einmal versucht hat, einem Kopierer klarzumachen, wie er eine nicht standardkonforme Vorlage kopieren soll, kann ein Lied davon singen.

1.2 Sachwissen in die Software! – ein Schritt zu einfacheren Bedienschnittstellen

Das Beispiel der *spontanen Bedienung* in Abbildung 1 verdeutlicht, wie die Bedienung eines TV-Geräts über Sprache für den Nutzer einfacher gestaltet werden kann. Ein derartiges Bedienprinzip technisch so umzusetzen, dass ein Bediensystem an die Anforderungen verschiedenster Geräte ohne großen Aufwand angepasst werden kann, stellt eine erhebliche Herausforderung bei der Entwicklung des Bediensystems selbst, aber auch für die Softwarekomponenten des zu bedienenden Geräts dar. Abbildung 1 zeigt ein Schema für den architektonischen Aufbau des natürlichsprachlichen Dialogsystems NESTOR (*naturally easy system to operate devices*) [Lud04a]. Seine Funktionsweise wird in diesem Beitrag an einem Zukunftsszenario und an Beispielen aus realisierten Anwendungen vorgestellt.

Im Forschungsprojekt EMBASSI (`http://www.embassi.de`) wurde folgende Systemarchitektur entworfen: Die unimodalen I/O-Geräte übersetzen Signale (wie Sprache, Gesten oder Messwerte) in *tokens*, die von der multimodalen Datenaufbereitung interpretiert werden. Die Bedeutung dieser Interpretation im Zusammenhang eines ganz bestimmten Bedienvorgangs zu verstehen, ist die Aufgabe von Dialogmanagement und Assistenzmethoden. Hierin liegt der fundamentale Perspektivenwechsel von der menügeführten zur spontanten Bedienung: im Dialogmanagement von NESTOR wird einer Benutzereingabe ein Ziel zugeordnet, das den Wunsch des Benutzers erfasst; die Assistenzebene sucht nach einer Möglichkeit, das Ziel durch einzelne Schritte zu erfüllen. Jeder einzelne Schritt wird auf der Ausführungsebene erledigt; die Geräte – ein zentraler praktischer Aspekt – melden Fehler an die Assistenz. Dort kann eine andere, durchführbare Lösung gesucht werden.

Abbildung 3: Armaturenbrett eines Cockpits mit modernem Mensch-Maschine-Interface (MMI) und Detailansicht des *display* (Quelle: `http://www.bmw.de`).

1.3 Anforderungen an die Realisierung des technischen Geräts

Die Entwicklung von Assistenzmethoden ist alles andere als trivial [Hub04]. Man stelle sich beispielsweise vor, in einem KFZ mit einem modernen farbigen hochauflösenden *display* zu sitzen (siehe Abbildung 3). Der Detailblick auf das moderne *display* verrät, dass mit Hilfe der modernen *MMI* vor allem die Vielzahl an Tasten und Knöpfen am Armaturenbrett reduziert werden soll, um den Bedienkomfort des Automobils zu erhöhen. In den neuen Modellen sehr vieler Automobilhersteller ist auch eine Bedienung über Sprache möglich. In der Regel ermöglichen diese Sprachschnittstellen die *menügeführte Bedienung* der über das MMI steuerbaren Funktionen. Aus der Sicht der inhaltlichen Verarbeitung erheblich komplexere Benutzerziele wie

> *Ich möchte zur Systems fahren; aber ich will nicht mit dem Auto durch die Stadt. Wie komme ich am schnellsten hin?*

sind zwar für die Spracherkennung und zum Teil auch für die Sprachanalyse keine sehr schwierigen Aufgaben, aber die Gerätesteuerung ist extrem komplex: die Benutzeräußerung enthält viele implizite Anweisungen. So eröffnet die Aussage *nicht mit dem Auto durch die Stadt* mehrere Optionen für die Routenplanung: einen Weg mit dem KFZ um die Stadt herum suchen, oder den öffentlichen Nahverkehr mit einbeziehen und dann eine Route auch mit der U-Bahn. Diese Optionen stehen aber nur dann zur Verfügung, wenn die Routenplanung auch andere Datenbanken abfragen und deren Ergebnisse integrieren kann. Natürlich kann eine KFZ-Navigation dies nicht leisten; deshalb wäre diese Integration eine typische Aufgabe für eine Assistenzmethode eines innovativen *MMI*.

2 Ein Zukunftsszenario

Die Integration von Einzelfunktionalitäten, die geeignet miteinander kombiniert werden müssen, um komplexe Bedienvorgänge zu unterstützen, ist eine Herausforderung für die

Forschung auf dem Gebiet multimodaler Bedienschnittstellen. Sie ermöglicht es, komplexe Bedienvorgänge zu vereinfachen. Man stelle sich folgendes Anwendungsszenario vor: In einem großen Werk eines Automobilherstellers gibt es mehrere Produktionshallen, in denen Teams von Mechanikern Fahrzeuge montieren, reparieren, umbauen und warten. Teile, die sie in die Fahrzeuge verbauen, bestellen sie per Telefon beim zentralen Lagerhaltungs- und Warenwirtschaftssystem, das natürlichsprachliche Bestellungen verstehen und bearbeiten kann. Wenn es den Auftrag

(1) *Team 3 in Halle 5 braucht 10-Xenon-Einbausätze für Modell B5!*

erhält, stellt das System fest, in welchem der zahlreichen, unabhängigen Lager auf dem Fabrikgelände die gesuchten Teile zu finden sind, prüft deren Verfügbarkeit und leitet den Auftrag an das Lagerkontrollsystem weiter. In jeder der Lagerhallen fahren wie Roboter gesteuerte Gabelstapler autonom umher und führen Bestellaufträge wie den eben beschriebenen aus. Bei diesen Abläufen kann viel passieren, wenn ein Gabelstapler ein Bauteil nicht richtig greift, so dass es auf den Boden fällt, oder wenn ein Transportkasten nicht richtig auf der Gabel aufsitzt, so dass er aus großer Höhe herunterfallen kann. Um solche Schäden zu verhindern, führt in jeder Lagerhalle eine Person Aufsicht, die über Monitore das Geschehen in den Regalgassen überwachen und die Gabelstapler steuern kann, indem sie ihnen natürlichsprachliche Kommandos erteilt. Die Aufsichtsperson kontrolliert zudem, dass die eingehenden Aufträge korrekt erledigt werden. Sie kann die Aufträge auf einem Kontrollbildschirm einsehen, und die Gabelstapler erstatten auch Bericht, welche Aufgaben sie gerade durchführen und welche sie bereits erfüllt haben.

(2) *Stapler 2 verlädt Xenon-Einbausätze für Team 3.*

(3) *Stapler 3 verlädt Nebelscheinwerfer für Team 1.*

Die Aufsichtsperson kann auf die Arbeit der Gabelstapler Einfluss nehmen:

(4) *Alle Transportkästen für Halle 5 auf Waggon 2!*

Diese Anweisung findet ab sofort Berücksichtigung:

(5) *Stapler 2 setzt Transportkasten auf Waggon 2 ab. Auftrag erledigt.*

(6) *Stapler 4 bringt Transportkasten zu Waggon 2.*

Gleich nach dieser Meldung beobachtet die Aufsichtsperson auf einem Kontrollmonitor, dass der Transportkasten von Stapler 3 schwankt und zu kippen droht.

(7) *Den Kasten schnell ins Regal stellen!*

Beim hektischen Versuch, ein Herunterfallen des Kastens zu vermeiden, formuliert die Aufsichtsperson nicht präzise: Welcher Stapler ist eigentlich „angesprochen“? Der Spracherkenner bemerkt auch die erhöhte Sprechgeschwindigkeit und die unübliche Lautstärke und informiert das Dialogsystem, dass die Aufsichtsperson sehr angespannt ist. Aus den

Rückmeldungen der Stapler ist dem System bekannt, welche Aktionen die Stapler gerade durchführen. Aus dem Inhalt der Äußerung und der Einschätzung der Stimmungslage der Aufsichtsperson erkennt das System, dass Gefahr im Verzug ist. Die Operationen der meisten Stapler sind unkritisch, Stapler 3 und 5 jedoch heben gerade Lasten in die Höhe. Die beiden Stapler werden beauftragt, die Lasten auf schnellstem Weg auf einem Regalboden abzustellen, um die Gefahr zu bannen.

(8) *Stapler 3 setzt Ladung in Regal 2, Ebene 4 ab.*

(9) *Stapler 5 setzt Ladung in Regal 3, Ebene 4 ab.*

Die Aufsichtsperson merkt an der letzten Meldung, dass auch Stapler 5 seine Ladung absetzt, was gar nicht beabsichtigt war. Mit dem Kommando

(10) *Außer Stapler 3 alle weiterarbeiten!*

sorgt sie dafür, dass der Betrieb nicht mehr als nötig gestört wird und macht sich auf den Weg zu Regal 2, um dort das Problem selbst zu beheben. Team 3 wartet währenddessen auf die Xenon-Einbausätze und fragt beim Lagerhaltungssystem nach:

(11) *Wo bleiben die Xenon-Einbausätze?*

Das System identifiziert an der Bauteilbezeichnung den Auftrag und kann beim Lagerkontrollsystem erfragen, dass der Transportkasten schon verladen ist und nur noch abgeholt werden muss. Vom Warentransportsystem kommt die Information, dass der Transportkasten um 13:47 abgeholt wird. Jetzt kann das Team 3 informiert werden.

(12) *Die Xenon-Einbausätze kommen in 7 Minuten.*

3 Quellen der Komplexität bei der Kommunikation zwischen Mensch und Maschine

In dieser Beschreibung sind einige zentrale Aufgabenstellungen enthalten, die für einen konfigurierbaren Ansatz für zweckrationale Dialoge, essentiell sind:

3.1 Herausforderungen an die Dialoganalyse

Um die Benutzeranforderung in Äußerung (1) verarbeiten zu können, sind Pläne zu erstellen, wie das System adäquat reagieren kann [Lud04b]. Einzelne Handlungen in einem Plan sind dabei entweder Sprechakte oder Aktionen von Komponenten eines Gesamtsystems.

Der Erfolg einer Reaktion wird dadurch festgestellt, dass während der Ausführung eines Plans protokolliert wird, welche Schritte schon abgearbeitet sind. Dazu müssen die Systemkomponenten mit dem Bediensystem Informationen austauschen. Dies wird dadurch

ermöglicht, dass Bediensystem und Komponenten über eine gemeinsame (formale) Sprache verfügen, in der anwendungs- und situationsspezifisches Wissen formuliert werden kann. Die Äußerungen (2), (3), (4), (6) und (7) zeigen, wie die ausgetauschten Informationen an den Benutzer in natürlicher Sprache weitergeleitet werden.

Unter Umständen wird auch von einem Bediensystem erwartet, dass es aus Äußerungen Information über Regeln oder Prozeduren extrahieren kann wie in der Äußerung (5). Dieser Äußerung entspricht ja nicht ein einziges Kommando an irgendeine Systemkomponente; vielmehr beschreibt die Äußerung eine Regel, die beim Planen von Handlungsabläufen zu berücksichtigen ist, und verändert somit die Wissensbasis des Bediensystems.

Eine wesentliche Aufgabe eines interaktiven Bediensystems ist es, den Gesprächsverlauf zu verwalten, der einem Bediendialog zugrunde liegt. Dabei ist zu berücksichtigen, dass die bedienenden Personen in einer sich ändernden Umwelt agieren und Wahrnehmungen machen, die nicht aus dem bisherigen Bediendialog gefolgert werden können. Äußerung (8) ist hierfür beispielhaft: Die Aufsichtsperson beobachtet einen Vorgang beim Blick auf den Monitor, nicht indem sie Systemmeldungen mithört. Die Beobachtung veranlasst die Aufsichtsperson aber zu einer Äußerung. Das Dialogsystem steht nun vor der Aufgabe, einen sachlichen Bezug zwischen dieser Äußerung und dem bisherigen Dialogverlauf zu herzustellen. Dies wird nur dann gelingen, wenn das Dialogmodell erlaubt, dass sich Äußerungen auch auf nicht-sprachlichen Handlungen, Vorgänge und Wahrnehmungen in der aktuellen Situation beziehen können.

Schließlich hängen adäquate Reaktionen des Bediensystems nicht nur davon ab, logisch Korrektes aus der Bediensituation zu erschließen, sondern auch davon, wie man aus mehreren denkbaren Optionen das Relevante herausfindet. Dies ist insbesondere dann entscheidend, wenn bestimmt werden muss, welche Objekte in der Bediensituation von einer Äußerung referenziert werden. Liest man die Äußerungen (6), (7) und (8) ohne Kenntnis der Beobachtung der Aufsichtsperson, wird man davon ausgehen, dass mit der Formulierung *den Kasten* in (8) der Kasten aus (6) oder der aus (7) gemeint sein muss.

Die hier am Beispiel beschriebenen Aufgabenstellungen werden von den Dialogsystemen, die als Dialogmodell vorab definierte und daher nicht abänderbare Ablaufgraphen einsetzen, nicht thematisiert. Auch andere, komplexere Ansätze behandeln nur in einzelnen Aspekten das oben umrissenen Spektrum an Forschungsfragen [Mor97, All95, CSPC00].

3.2 Lösungsansätze

Das Dialogsystem NESTOR hingegen basiert auf der Grundidee, dass es nur dann möglich ist, zweckrationale Dialoge zu interpretieren, wenn ein Dialog nicht nur anhand struktureller Aspekte, die sich an den Äußerungen des Dialogs ablesen lassen, analysiert wird. Denn der Dialog ist der Phänotyp der diskurs- und situationspragmatischen Vorgänge, die tatsächlich stattfinden. Diese Vorstellung führt dazu, dass jede zu steuernde Applikation durch geeignete Operatoren in einer Planungssprache beschrieben wird. So kann die pragmatische Bedeutung einer Äußerung in einem Dialog durch einen Plan beschrieben werden, dessen Erfüllbarkeit im Lauf der Planausführung verifiziert wird.

In einer prototypischen Anwendung, die an den Lehrstühlen für Mustererkennung und Künstliche Intelligenz der Universität Erlangen-Nürnberg untersucht wird, geht es darum, komplexe Logistikaufgaben zu lösen. Eine scheinbar einfache Äußerung wie

Ich möchte gerne eine Tasse Kaffee.

kann eine komplexe Handlungskette auslösen. Der aktuelle Systemzustand

```
(:init (at engine1 stationB)   (at waggon1 siding)
       (cup-state cup1 empty)  (stored-on cup1 stack1)
       (at cm1 stationC)       (at cup1 stationC)
       (coffee-machine-state cm1 off))
```

hat darauf Einfluss, welche Aktionen ausführbar sind. Als Ziel wird der Äußerung

```
(:goal (at cup1 stationA))
```

zugeordnet. Mit Hilfe eines Planers kann nun eine Folge von Handlungen bestimmt werden, die vom aktuellen zum erwünschten Systemzustand führt:

```
0: switch-on(cm1)  put-below-spout(cup1,cm1,stack1)
   compute-route(stationB,siding,engine1)
1: go(engine1,stationB,siding)  fill-cup(cm1,cup1)
2: connect(waggon1,engine1,siding)
   compute-route(siding,stationC,engine1)
3: go(engine1,siding,stationC)
4: put-on-waggon(engine1,waggon1,cup1,CM1,stationC)
   compute-route(stationC,stationA,engine1)
5: go(engine1,stationC,stationA)
```

Ein derartiger Plan beschreibt, wie der Benutzerwunsch möglicherweise erfüllt werden kann. Ob dies tatsächlich geschieht, kann überprüft werden, indem die einzelnen Schritte des Plans ausgeführt werden. Dazu nutzt der Planer sein Wissen, welche Vorbedingungen ein Schritt hat, und welche Veränderungen er im System hervorruft. In der Planungssprache PDDL kann man diese Information so darstellen, wie es das Beispiel unten zeigt:

```
(:action faster
  :parameters  (?e1 ?u)
  :precondition (not (maxspeed ?e1))
  :effect (and (when (likesFast ?u) (maxspeed ?e1))
               (when (likesSlow ?u) (mediumspeed ?e1))))
```

Dieser Operator ist einfacher zu erklären als diejenigen aus dem obenstehenden Plan und soll daher zur Illustration der Idee dienen, wie die Ausführung von Planschritten verfolgt werden kann. Der Operator erhöht, wenn er angewendet wird, die Geschwindigkeit des Zugs `?e1` um einen relativen, aber vom Nutzer `?u` abhängigen Wert. Je nachdem, ob `?u`

hohe oder eher langsamere Geschwindigkeiten präferiert, wird die Geschwindigkeit von `?e1` auf `maxspeed` oder `mediumspeed` gesetzt.

Der Planer baut den Operator nur dann in eine Handlungssequenz ein, wenn es möglich ist, dass vor der Ausführung des Operators die Geschwindigkeit von `?e1` kleiner als `maxspeed` sein wird. Dass dies der Fall ist, muss bei der Planausführung sichergestellt werden. Vor Ausführung des Operators `faster` wird also geprüft werden, ob die Geschwindigkeit kleiner als `maxspeed` ist. Falls ja, wird der Operator ausgeführt werden. Andernfalls wird dies dem Benutzer mitgeteilt werden. Diese Form der Simulation erlaubt es, ein Dialogmodell zu konzipieren, das den komplexen Anforderungen an Dialogkohärenz gerecht wird, wie sie in Bediensituationen auftreten.

Äußerungen sind jedoch nicht immer das Resultat der Simulation von Handlungen. Es gibt für die Dialogteilnehmer auch eine Reihe anderer Motive, dem Gegenüber etwas mitzuteilen. Es gibt auch Motive, gerade dies zu unterlassen. Ein Beispiel hierfür bietet das eben beschriebene Logistikszenario: Für den Benutzer kann es von Interesse sein, zu erfahren, welche Züge auf ihrer Fahrt welche Positionen erreichen. NESTOR meldet dem Benutzer jede Positionsänderung jedes Zugs in einer Mitteilung wie dieser:

Zug 23 fährt auf Gleis 10.

Wieviele solcher Mitteilung das System machen muss, hängt davon ab, wieviele Züge mit welcher Geschwindigkeit unterwegs sind. Irgendwann ist eine Schwelle überschritten, ab der kein Benutzer mehr in der Lage ist, die vielen neuen Mitteilungen aufzunehmen. Aspekte situationsabhängiger Kommunikationsstrategien überlagern also die Ergebnisse der Plansimulation. Eine Mitteilung soll nicht nur sachlich korrekt, sondern auch der Situation angemessen sein. Strategien, die eine derartige flexible Dialogplanung erlauben, lassen sich in NESTOR implementieren. Beispielsweise kann man das Informationsvolumen schätzen, das durch eine Mitteilung auf den Hörer zukommt, indem man misst, wieviel Zeit benötigt wird, um die Äußerung zu sprechen und wieviele Inhaltselemente die Mitteilung enthält. Steigen diese Werte in einem fixen Zeitintervall zu stark an, werden nur noch Mitteilungen höchster Priorität geäußert. Auf diesem Weg lassen sich Kooperationsmaximen wie die GRICEsche Maxime der Quantität und Relevanz umsetzen.

Abbildung 4: Welche beiden ICEs meint die Äußerung *„Beide Züge anhalten!“*?

Die Relevanz einer Äußerung ist nicht nur für die Generierung von System-, sondern auch für die Analyse von Benutzeräußerungen wichtig. Sind mehrere Züge unterwegs, beispielsweise die drei aus Abbildung 4, ist die Äußerung

Beide Züge anhalten!

für sich alleine genommen missverständlich, weil nicht klar ist, welche zwei der drei Züge gemeint sein könnten. In einem gegebenen Kontext kann aber jedem der sechs kombinatorisch möglichen Paare von Zügen eine Relevanzbewertung zugeordnet werden, die – idealerweise – eine eindeutige Identifikation der gemeinten Züge ermöglicht.

Nicht in jeder Situation aber wird eine klare Entscheidung möglich sein. NESTOR sieht für alle solchen Fälle vor, mit dem Benutzer Klärungsdialoge führen. Dies gilt nicht nur für Fragen der Semantik, sondern auch für kritische Fälle bei der Herstellung eines Gesprächszusammenhangs, bei der syntaktischen Analyse und der Verarbeitung von Worthypothesengraphen, die der Spracherkenner liefert. Wie für pragmatische Mehrdeutigkeiten oder Konflikte kann NESTOR mit Dialogstrategien konfiguriert werden, die versuchen, eine Systemreaktion zu planen und umzusetzen. Sie soll dem Benutzer zunächst intuitiv verdeutlichen, welche Art von Missverständnis vorliegt. Dann kann versucht werden, in einem Klärungsdialog die entdeckten Schwierigkeiten und Widersprüche aus dem Weg zu räumen und den Benutzerwunsch zu erfüllen. Die Details der dafür entwickelten Verfahren sind ein Kapitel für sich und würden den Rahmen dieses Beitrags sprengen (Details finden sich in [Lud04a], Kapitel 5).

4 Bilanz und Perspektiven

Das Dialogsystem NESTOR realisiert ein Dialogmodell, das eine Alternative zu den vorherrschenden Ansätzen, Dialoge mit Hilfe endlicher Automaten oder Varianten von ihnen zu interpretieren, darstellt. NESTOR ist so konzipiert, dass die Funktionalität des Dialogverstehens und -generierens durch allgemeine Algorithmen gewährleistet wird, die für sich genommen nicht von anwendungsspezifischen Annahmen oder Heuristiken Gebrauch machen. Stattdessen kann das anwendungstypische Wissen in formalen Sprachen mit wohlverstandenen komplexitäts- und berechenbarkeitstheoretischen Eigenschaften konfiguriert werden. Dies erleichtert die Aufgabe der Wissensrepräsentation, die für die Adaption eines Dialogsystems an eine neue Anwendung unumgänglich ist, erheblich.

Das Prinzip, Benutzeräußerungen mit einem Plan in Zusammenhang zu stellen, der einen Benutzerwunsch in einer konkreten Situation umsetzen soll, wird auch auf die Planung der Dialogführung übertragen. Dadurch lassen sich auch Dialoge erklären, in denen nicht nur die Durchführung des von einem Benutzerwunsch induzierten Plans, sondern auch akustische, syntaktische und semantische Verständnisprobleme thematisiert werden.

Viele Forschungsprobleme sind noch ungelöst. So ist es unklar, welche Faktoren einen Dialogkontext vollständig charakterisieren und wie sie so formalisiert werden können, dass sie effektiv und möglichst in Echtzeit in ein Dialogsystem integriert werden können. Dazu zählen auch viele nicht-sprachliche Phänomene, wie z.B. Hintergrundgeräusche, die

die Spracherkennung stören, emotionale Befindlichkeiten, soziale und konventionelle Gesprächsgewohnheiten und Stereotypen, die Benutzer aus der Interaktion zwischen Menschen auf die Mensch-Maschine-Interaktion übertragen. Da sie nie explizit erwähnt werden, stellen sie jedes Dialogsystem vor Herausforderungen. Bei Dialogen, die geführt werden, während die Benutzer oder sogar technische Komponenten (z.B. Roboter) Aktionen durchführen, reagieren Benutzer oft auf während des Dialogs gemachte Beobachtungen, die das System aber mangels geeigneter Sensoren nicht wahrnehmen kann. Solche Reaktionen plausibel in einen Dialogverlauf integrieren zu können, wird die Forschung in der automatischen Dialogverarbeitung noch lange beschäftigen.

Literatur

[All95] James F. Allen. *Natural Language Understanding*. Benjamin/Cummings Publishing Company, Redwood City, 1995.

[CSPC00] Justine Cassell, Joseph Sullivan, Scott Prevost und Elizabeth Churchill, Hrsg. *Embodied Conversational Agents*. MIT Press, Cambridge, 2000.

[Hub04] Alexander Huber. *Interaktive agentenbasierte Bedienassistenz – Konzeption und Implementation eines Agents für die Bedienung von technischen Systemen*. Dissertation, Universität Erlangen-Nürnberg, 2004.

[Lud04a] Bernd Ludwig. *Ein konfigurierbares Dialogsystem für Mensch-Maschine-Interaktion in gesprochener Sprache*. Dissertation, Universität Erlangen-Nürnberg, 2004.

[Lud04b] Bernd Ludwig. A Pragmatics-First Approach to the Analysis and Generation of Dialogues. In Susanne Biundo, Rhom Frühwirth und Günther Palm, Hrsg., *Proc. KI-2004 (27th Annual German Conference on AI (KI-2004)*, Seiten 82–96, Berlin, 2004. Springer.

[Mor97] Renato De Mori, Hrsg. *Spoken Dialogues with Computers*. Signal Processing and its Applications. Academic Press, San Diego, 1997.

Bernd Ludwig studierte von 1992 bis 1997 Informatik mit einem Schwerpunkt in Theoretischer Informatik und Nebenfach Gräzistik an der Universität Erlangen-Nürnberg. Von 1997 bis 2004 war er wissenschaftlicher Mitarbeiter am Lehrstuhl Informatik 5 (Mustererkennung) der Universität Erlangen-Nürnberg. In dieser Zeit erfolgte auch seine Promotion zum Thema Dialogsysteme, und er war währenddessen ab 2001 stellvertretender Teilprojektleiter für das Arbeitspaket des Bayerischen Forschungszentrums für Wissensbasierte Systeme (FORWISS) im Rahmen des MTI-Leitprojekts EMBASSI, stellvertretender Projektleiter des Teilprojekts SIPaDIM im Bayerischen Forschungsverbund für Situierung, Individualisierung und Personalisierung in der Mensch-Maschine-Interaktion (FORSIP) und seit 1.1.2003 stellvertretender Leiter der Forschungsgruppe Wissensverarbeitung am FORWISS. Seit Mai 2004 ist er wissenschaftlicher Assistent am Lehrstuhl Informatik 8 und dabei unter anderem Leiter verschiedener Forschungsprojekte des Lehrstuhls (`http://www8.informatik.uni-erlangen.de`).