ALBERT-LUDWIGS-UNIVERSITÄT
FREIBURG

INSTITUT FÜR INFORMATIK

Arbeitsgruppe Autonome Intelligente Systeme

Prof. Dr. Wolfram Burgard

Robotalk im Deutschen Museum

Studienarbeit

Patrick Pfaff

Mai 2003 – Juli 2003

Inhaltsverzeichnis

Kapitel 1

Einleitung / Aufgabenstellung

Das Jahr 2003 ist ein besonderes Jahr - zumindest für das Deutsche Museum in München. In diesem Jahr feiert es nämlich sein einhundert jähriges Bestehen mit einem Museumsmeilenfest in den Museen in München und Bonn. Als besondere Attraktionen sollten die Roboter Albert und Rhino die Besucher beider Museen durch die Abteilung Kommunikation fuehren und dabei so über das Internet kommunizieren, dass die Besucher die Möglichkeit haben, Exponate in München und Bonn gleichzeitig zu bewundern. In der Vergangenheit wurden auf dem Gebiet der robotergeführten Museumstouren bereits reichlich Erkenntnisse gesammelt. Die Vorreiter auf diesem Gebiet waren sicher die Serviceroboter RHINO[1, 2] und MINERVA[3] die im Deutschen Museum in Bonn bzw. im Smithsonium National Museum of American History in Washington D.C. als Museumsführer eingesetzt wurden. Das von der EU geförderte TOURBOT-Projekt[4] befasst sich ebenfalls mit der Entwicklung eines Museumsroboters. Eine erste komerzielle Nutzung dieser Forschungen wird derzeit im WEBFAIR-Projekt[5] entwickelt. Dieses Projekt hat das Ziel, mit Hilfe eines Roboters einen Messerundgang vornehmen zu können, ohne selbst vor Ort zu sein. Diese Projekte und die Verbindung zu dieser Arbeit werden genauer in Kapitel 2 beschrieben. Die Roboter-Demonstration sollte vom 19.6 bis 22.6.2003 stattfinden. Meine Aufgabe hierbei sollte sein, ein Programm zu entwickeln und implementieren, das die Führung der Besucher und die Kommunikation zwischen den Robotern steuert.

1.1 Zielsetzung / Problembeschreibung

Die Zielsetzung dieser Arbeit war es, eine zuverlässige Software zu entwickeln, die ein Graphical UserInterface (GUI) zur Verfügung stellt, dass vom Museumbesucher leicht durchschaut und angewendet werden kann. Ist der Roboter dann erst einmal unterwegs, sollte er selbstständig eine vorher definierte Tour abfahren und mehrere Exponate beschreiben, was zum Einen über die graphische Darstellung via Text und Bild und zum Anderen durch eine gut hörbare Sprachausgabe verwirklicht werden sollte. Ausserdem sollte, sobald einer der beiden Roboter an einem Exponat angekommen ist, der andere Roboter stoppen, dem Museumsbesucher das Exponat erklären, an dem sich sein Kollege im anderen Museum soeben befindet und zusätzlich ein Live-Bild via Internet übertragen.

1.2 Aufbau dieser Arbeit

Zuerst werde ich verwandte Projekte vorstellen, die Verbindung zu diesem Projekt herstellen und dadurch deutlich machen, was genau beim "Robotalk" geschehen sollte (Kapitel 2). Weiter werde ich die Planung des Projektes und den Aufbau der Software beschreiben, in die die neue Software eingebettet werden muss (Kapitel 3). Dann werde ich die Implementierung und den konkreten Aufbau meiner Software beschreiben (Kapitel 4). Danach werde ich mich den Tests im Vorfeld des Roboter-Events (Kapitel 5) und dem Verlauf des Events selbst widmen (Kapitel 6). Abschließend werde ich nocheinmal alle gemachten Erfahrungen zusammenfassen und ein Fazit ziehen (Kapitel 7).

Kapitel 2

Verwandte Projekte

2.1 MINERVA[3] und RHINO[1, 2]

Die Projekte mit RHINO im Deutschen Museum in Bonn und MINERVA im Smithsonium National Museum of American History in Washington D.C. fanden 1997 und 1998 statt. RHINO war der erste Roboter, der Besucher durch ein Museum führte. MINERVA war ein Jahr später die Weiterentwicklung dieser Technologie. Beide Systeme wurden von einer Software gesteuert, die zum Einen den Roboter steuerte und zum Anderen menschliche Interaktion ermöglichte, was folgende Probleme zu lösen aufgab:

1. **Steuerung des Roboters in dynamischen Umgebungen:**
   Da an öffentlichen Orten viele Menschen sind, die sich nicht immer unbedingt kooparativ mit einem Roboter zeigen, musste eine Software entwickelt werden, die den Roboter sicher und effektiv durch eine Meschenmenge hindurchmanövrieren kann.

2. **Steuerung des Roboters in unveränderten Umgebungen:**
   Die Roboternutzung macht es notwendig, dass sich die Umgebung nicht verändert.

3. **Design der menschlichen Interaktionmöglichkeiten:**
   Die Software wurde entwickelt, um mit Menschen kontakt aufzunehmen, die vorher nochnie einen Roboter gesehen haben. Um die Intuition der Leute anzusprechen, benutzt das Interface des Roboters Interaktionsmuster, die der zwischenmenschlichen Kommunikation ähnlich sind.

4. **Einbeziehen des Internets:**
   Das Web-Interface ermöglicht es Menschen auf der ganzen Welt, den Roboter zu überwachen, seine Bewegungen zu steuern und seine Live-Bilder zu sehen. Es wurde dem Internetbenutzer ermöglicht, die Kontrolle über den Roboter zu bekommen, um sich so einen Einblick ins Museum verschaffen zu können. Es gab hierzu verschiedene Versionen der Internetschnittstelle. Bei der ersten Version steht die Steurung des Roboters im Fordergrund. So hat der Internetbenutzer nur einmal kurz die Möglichkeit in das Geschehen einzugreifen, indem er einen Zielpunkt auswählt, den der Roboter ansteuern soll. Die zweite Version, bei der die Virtuelle Museumstour im Mittelpunkt steht, ermöglicht es dem Internetbenutzer, verschiedene Exponate auszuwählen und somit an einen Rundgang durch das Museum auf der daraus resultierenden Tour teilzunehmen. Die dritte Version fasst alles, was dem Internetbenutzer zugänglich ist, auf einer Webseite zusammen, denn es war bisher so, dass man mehrere Browserfenster benötigte. Außerdem schränkt es die Möglichkeiten des einzelnen Benutzers stark ein, da jeder Benutzer lediglich für eine Tour stimmen kann. Ob diese dann ausgeführt wird, hängt davob ab, ob die Mehrheit der anderen Internetbenutzer auch für diese Tour gestimmt hat.

2.2 TOURBOT[4]

Das Projekt TOURBOT wird von der Europäischen Union gefördert und befasst sich mit der Entwicklung eines Museumsroboters. Dabei hat sich dieses Projekt folgende Ziele gesteckt:

1. Entwicklung eines mobilen Roboters, der die Fähigkeit besitzt, kollisionsfrei zu navigieren, um sich autonom in einem Museum fortzubewegen.

2. Entwicklung einer geeigneten Internet-Benutzerschnittstelle zu diesem Roboter, die es entfernten Personen ermöglicht, durch den "Roboter-Avatar" im Museum präsent zu sein und sich die vorhandenen Ausstellungsstücke anzuschauen.

3. Entwicklung einer geeigneten Benutzerschnittstelle an Board des Roboters, die den Museumsbesuchern vor Ort die Teilnahme an interaktiven Führungen durchs Museum ermöglicht.

2.3 WEBFAIR[5]

Das WEBFAIR-Projekt bezeichnet ein interaktives Tele-Präsentationssystem und soll einzelnen Internetbenutzern durch mobile Agenten/Roboter einen Internetzugang zu großen Ausstellungen und komerziellen Handelsmessen ermöglichen. Die Ziele des Projekts sind dabei folgendermaßen definiert:

1. Entwicklung des mobilen Agenten auf Basis bestehender Technologien, die die Roboternavigation und -Kontrolle, Webinterfaces und die Media-Communication umfassen.

2. Entwicklung einer geeigneten Internet-Benutzerschnittstelle zu diesem Roboter, die es entfernten Personen ermöglicht, durch den "Roboter-Avatar" auf der Messe präsent zu sein und sich die vorhandenen Ausstellungsstücke anzuschauen. Der Internetbenutzer sieht mit den Augen(Kameras) und hört mit den Ohren(Mikrofone) des Roboters. Es soll möglich sein an Austellungen und Präsentationen teilzunehmen. Weiter ist das Ziel, die Kontaktaufnahme zu den Ausstellern und den Informationsfluss zum Internetbenutzer zu vereinfachen. Die beiden großen Schlagworte sind e-commerce und e-advertisement.

2.4 Vergleich der Projekte

Die Projekte mit RHINO[1, 2] und MINERVA[3] waren die Vorreiter der Projekte TOURBOT[4] und WEBFAIR[5], denn die Programme, die einen Roboter sicher und autonom durch menschengefüllte Räume steuern, wurden im Rahmen dieser Forschungen entwickelt. Die Steuerung stand also am Anfang weit mehr im Fordergrund, als das bei den darrauffolgenden Projekten TOURBOT und WEBFAIR der Fall ist. Diese befassen sich mehr mit der Anwendung dieser Technologien und dem Transport von Informationen via das World Wide Web. WEBFAIR verfolgt hierbei im Vergleich zu allen anderen Projekten das Ziel, die kommerzielle Nutzung der Roboter vorranzutreiben. In der Mobilen Robotik hat sich also im Laufe der Zeit ein Wandel von der Grundlagenforschung hin zu einer mehr anwendungs- und nutzenorientierten Forschung vollzogen.

2.5 Verbindung mit ROBOTALK

Das Projekt "Robotalk im Deutschen Museum" passt mit in die Reihe der letzten beiden Projekte[4, 5], da man vom derzigen Stand der Forschung ausging(*State of the Art*), um ein Programm für die Museumsbesucher in München und Bonn zu entwickeln. Im Vergleich zu den Projekten davor sollte aber das Internet eine andere Bedeutung zugemessen bekommen, denn das Programm USERINTEFACE sollte in erster Linie die Menschen in der direkten Umgebung des Roboters ansprechen und den Internetbenutzer nur als Zuschauer integrieren. Es stand also ähnlich wie bei TOURBOT[4] die Erschaffung einer Benutzerschnittstelle an Bord des Roboters im Mittelpunkt, die den Museumbesuchern eine Teilnahme an den Führungen ermöglichen soll. Das Internet hat im Vergleich zu den früheren Projekten aber noch eine zusätzliche Aufgabe: nämlich die Kommunikation zwischen zwei baugleichen Robotern in unterschiedlichen Museen. Das Ziel war, die Menschen im anderen Museum mit in die Museumstour einzubeziehen, indem man immer genau dann, wenn der Roboter ein Exponat erreicht hat, den anderen Roboter über das Internet benachrichtigt. Dieser zeigt dann in einer Art Videokonferenz das Kamerabild seines entfernten Kollegen an und erklärt den umstehenden Besuchern, was darauf zu sehen ist. Im Internet ist das Videobild beider Roboter zu verfolgen und man bekommt eine Information über die Aktuelle position des Roboters. Es ist leider nicht möglich im Internet an den Führungen Teil zu nehmen, da es dort keine Exponatsbeschreibungen gibt. Man sieht dort lediglich den Roboter fahren. Das ist im Nachhinein zu bedauern, da man das in der Vergangenheit bereits realisiert[6] hat und sich der Aufwand daher sicher in Grenzen gehalten hätte. Der Schwerpunkt des Robotalk war also die Entwicklung eines Programmes, das den Ablauf einer Museumstour steuert, mit einem weiteren Roboter kommuniziert, Videobilder überträgt und eine für den Benutzer intuitiv verständliche Benutzeroberfläche aufweist.

Kapitel 3

Projektplanung

3.1 Modularität der TCX-Architektur und BEE Soft

Um den Aufbau meines Programmes besser verstehen zu können, stelle ich noch einmal kurz den Aufbau der BEE-Soft Software und die alles verbindende TCX-Architektur vor. Der prinzipielle Aufbau der Robotersoftware ist modular. Verschiedene Module kontrollieren die Funktionen des Roboters. Es gibt eine ganze Reihe an verschiedenen Modulen, die Paralel arbeiten. So Beispielsweise die Programme der Bee-Soft Sofwtware zur Pfadplanung und Kollisionsvermeidung, das Programm zur Lokalisierung des Roboters, die Sprachausgabe oder eben das von mir programmierte Modul **USERINTERFACE**. Die verschiedenen Module können sich gegenseitig Nachrichten schicken. Dieses Versenden und und empfangen von Nachrichten wird auf einem Server-Modul, dem TCX-Server, organisiert. Die Grundoperationen sind send(Message) und receive(Message). Jedes Modul braucht hierfür einen festgelegten Namen, um von den anderen erreicht werden zu können. Jedes Modul besitzt Handler für jede Nachricht, die es empfangen kann. Diese werden aufgerufen, wenn eine Nachricht, die mit diesem Handler assoziiert wird, empfangen wird. Der Nachrichten-Handler, macht dann das, was für diese Nachricht festgelegt wurde. Die Bee-Soft stellt natürlich eine Vielzahl solcher Handler zur Verfügung, die extrem wichtig für die Funktion meines Programmes sind. Die Bee-Soft, die die Pfadplanung und Kollisionsvermeidung auf einer Museumstour steuert, sendet beispielweise eine Nachricht, sobald ein Ziel bzw. Exponat auf der Tour durchs Museum erreicht worden ist. Diese Tatsache macht es natürlich relativ einfach, eine Tour zu planen,

da man einfach so lange fährt, bis man eine Nachricht bekommt, dass der Roboter sein Ziel erreicht hat. Dazu aber später mehr, denn hier soll lediglich ein kurzer Einstieg in die Modularität der TCX-Architektur gegeben werden.

3.2 Aufteilung des Projektes

Das Modul USERINTERFACE gliedert sich in 4 Teile, deren genauer Aufbau und Funktion in Kapitel 3 ausführlich beschrieben wird. Der wichtigste Teil, sozusagen das Herzstück des Moduls, steuert den Ablauf einer Tour, gibt Kommandos zum Fahren und Sprechen und kommuniziert via TCX mit den Modulen zur Fortbewegung, Sprachausgabe und Pantiltsteuerung. Dieser Teil des Programmes ist als endlicher Automat konzipiert. Der zweite Teil des Programmes ist die Benutzeroberfläche auf dem Touchscreen, das GUI. Die beiden anderen nicht so umfangreichen Teile sind zum Einen ein Image-Client, der für das Anzeigen der Live-Bilder auf dem Screen verantwortlich ist und zum Anderen ein Client-Server-Programm, das für die Kommunikation der beiden Roboter zuständig ist.

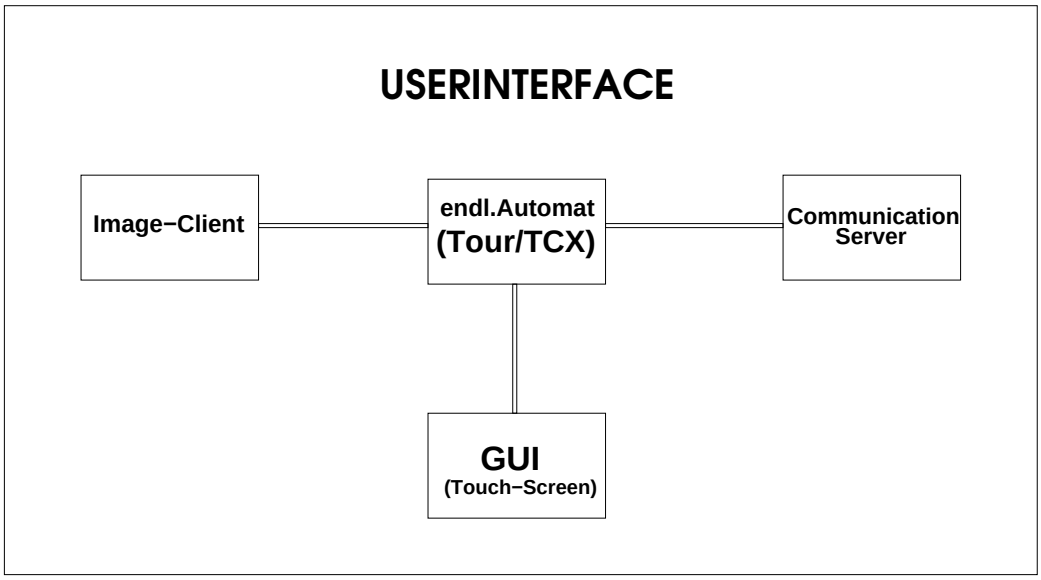

Abbildung 3.1: Aufbau des USERINTERFACEs

3.3 Eigenschaften der Benutzerschnittstelle

Die Vorgaben zur Modellierung der graphischen Benutzeroberfläche lassen sich kurz und knapp auf den Punkt bringen. Der Benutzer sollte verstehen, wie er den Roboter zum Fahren veranlässt und den Programmablauf danach möglichst wenig stört. Die Buttons auf dem Touch-Sreen sollten gut sichtbar und leicht vom Benutzer zu drücken sein. Dem Benutzer sollte darüber hinaus die Möglichkeit gegeben werden, selbst zu interagieren. So wurden Buttons zum Starten und Beenden von Touren, zum Unterbrechen und Fortsetzen einer Tour und zum Überspringen einzelner Exponate vorgesehen. Wie Sinnvoll diese Festlegungen im Nachhinein waren, betrachte ich unter Kapitel 5 Abschnit 2.

3.4 Allgemeinheit und Editierbarkeit

Um das Programm möglichst allgemein zu halten, wurde eine Vielzahl an Variablen in einem *"ini-File"* initialisiert. So kann man alle wichtigen Werte immmer wieder den Gebenheiten anpassen, ohne den Quellcode ändern zu müssen. So wurden beispielsweise die Anzahl der Exponate, die gesprochenen und angezeigten Texte und die Host- und Porteigenschaften auf diese Weise variabel gehalten. Das *"ini-File"* wird beim Starten des USERINTERFACEs eingelesen. Möchte man also einen Wert ändern, genügt es das Programm zu beenden und nach dem Ändern der Daten wieder neu zu starten. Dieser Umstand erwies sich auch im Verlauf des Robotalk, wie in Kapitle 4 beschrieben ist, als sehr Hilfreich, da es das korrigieren der dort auftretenden Probleme erheblich erleichterte.

Für die verschiedenen Teile des Moduls USERINTERFACE wurden folgende Variablen geschaffen:

TCX-Tour:

- Anzahl der Exponate
- Positionsdaten der Exponate
- Exponatstexte für Sprach- und Textausgabe
- Zusammenstellung der verschiedenen Touren
- diverse Wartezeiten zur Feinabstimmung der zeitlichen Abläufe
- Verwendung von Pantilt, ja oder nein

GUI:

- Bildschirmauflösung und Hintergrundbild
- Festlegung der anzuzeigenden Buttons
- Schriftgrößen

Image-Client:

- Bildgröße der übertragenen Bilder
- Host- und Porteigenschaften der des Image-Servers

Communication-Server:

- Host- und Porteigenschaften des QServerSockets

Kapitel 4

Implementierung des Moduls USERINTERFACE

4.1 Museumstour als endlicher Automat

Der zentrale Teil des USERINTERFACEs besteht aus einem endlichen, deterministischen Automaten, der alle Zustände beinhaltet, die während einer Museumstour auftreten können.

Zu Beginn, also unmittelbar nach Starten des Programmes, ist der Automat im Start- und Terminalzustand, der durch doppelte Umrandung gekennzeichnet ist. Diesen kann er verlassen, wenn ein Museumsbesucher eine Tour startet, oder wenn der andere Roboter soeben ein Exponat erreicht hat, das zu Praesentieren ist. Die Tour ist aber formal nicht gestartet, da der Roboter noch steht und noch keinen Zielpunkt erhalten hat. Deshalb ist die globale Variable TOUR = false gesetzt. Abbildung 4.1 zeigt diesen Fall.

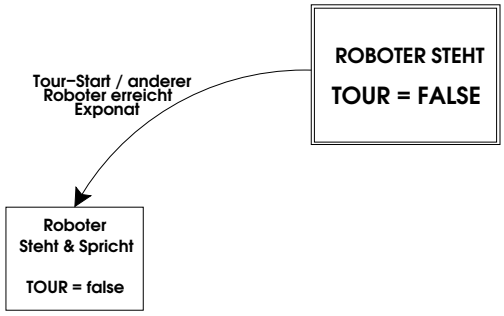

Abbildung 4.1: Terminalzustand als Ausgangszustand

Befindet sich der Automat nun in diesem Zustand, weil die Tour durch eine Benutzerinteraktion gestartet wurde, dann startet der Automat die Tour, sobald der Begrüßungstext vom Roboter gesprochen wurde, TOUR wird "true" gesetzt und der Roboter beginnt zu fahren. Andernfalls befindet sich der Roboter im Zustand "Roboter steht und spricht" mit TOUR = false, weil er ein Exponat vom Roboter im anderen Museum Praesentiert hat. In diesem Fall kehrt der Automat nach Abschluss der Praesentation in den Start -und Endzustand zurück, was durch Abbildung 4.2 dargestellt wird.

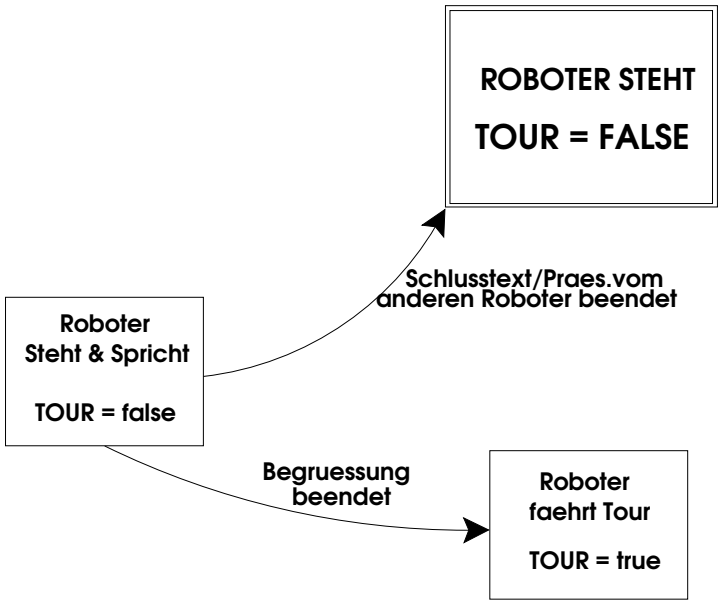

Abbildung 4.2: Tourstart

Abbildung 4.3 beschreibt folgendes: solange der Roboter die Tour fährt und TOUR = true gilt, befindet sich der Automat im entsprechenden Zustand. Kommt einer der Roboter an einem Exponat an, wechselt der Automat seinen Zustand und beide Roboter praesentieren den Museumsbesuchern das Exponat. Ist die Praesentation beendet, kehrt der Automat in seinen vorheringen Zustand zurück und die beiden Roboter fahren weiter auf der Tour.

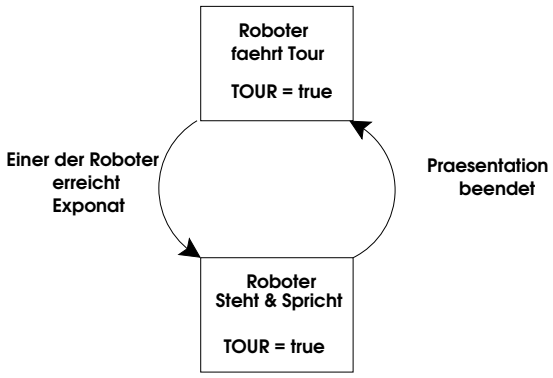

Abbildung 4.3: Wechsel von Fahren und Sprechen

Von den beiden Zuständen aus, die in der Abbildung 4.3 gezeigt werden, kommt man irgendwann in den Zustand "Roboter fährt zurück zum start". Enweder dadurch, dass ein Museumsbesucher auf den Button "Zurück zum start" drückt oder dadurch, dass die soeben beendete Praesentation die des letzten Exponats auf einer Tour war. Die Globale variable TOUR ist hier immernoch "true". Siehe Abbildung 4.4

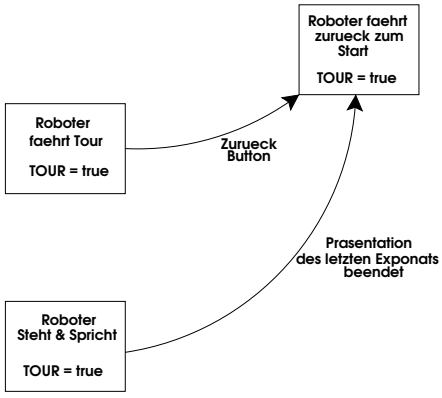

Abbildung 4.4: Roboter fährt zurück zum Start

In diesem Zustand verweilt der Automat solange, bis der Roboter den Ausgangspunkt, andem die Tour gestartet wurde, wieder erreicht hat. Dann wechselt er wieder in den Zustand "Roboter steht und spricht" wobei TOUR = false gilt. Der Roboter spricht dann den Abschlusstext, mit dem er die Besucher verabschiedet. Danach geht der Automat in den Terminalzustand über und die Tour ist beendet, was in Abbildung 4.5 dargestellt wird.

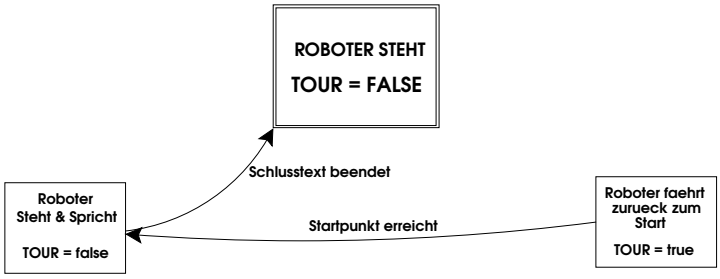

Abbildung 4.5: Ende einer Tour

Ist der Roboter unterwegs und der Automat befindet sich demnach in einem der beiden Zustände, in denen der Roboter fährt, kann der Museumsbesucher jeder Zeit alles stoppen, indem er der "Stop-Button" auf dem Touchscreen drückt. Der Roboter bleibt dann sofort stehen, alle verwendeten Variablen werden in den Ursprungszustand zurückgesetzt und der Automat geht in seinen Terminalzustand über. Man betrachte hierzu Abbildung 4.6.

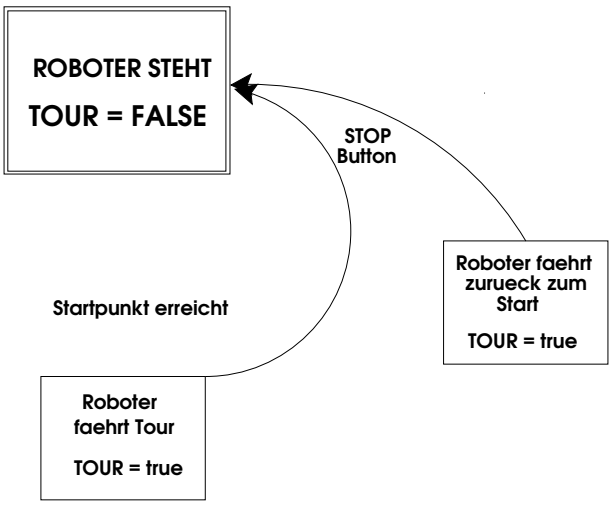

Abbildung 4.6: Abbruch durch Benutzer

Zusammengefasst sieht der endliche, deterministische Automat dann aus wie in Abbildung 4.7.

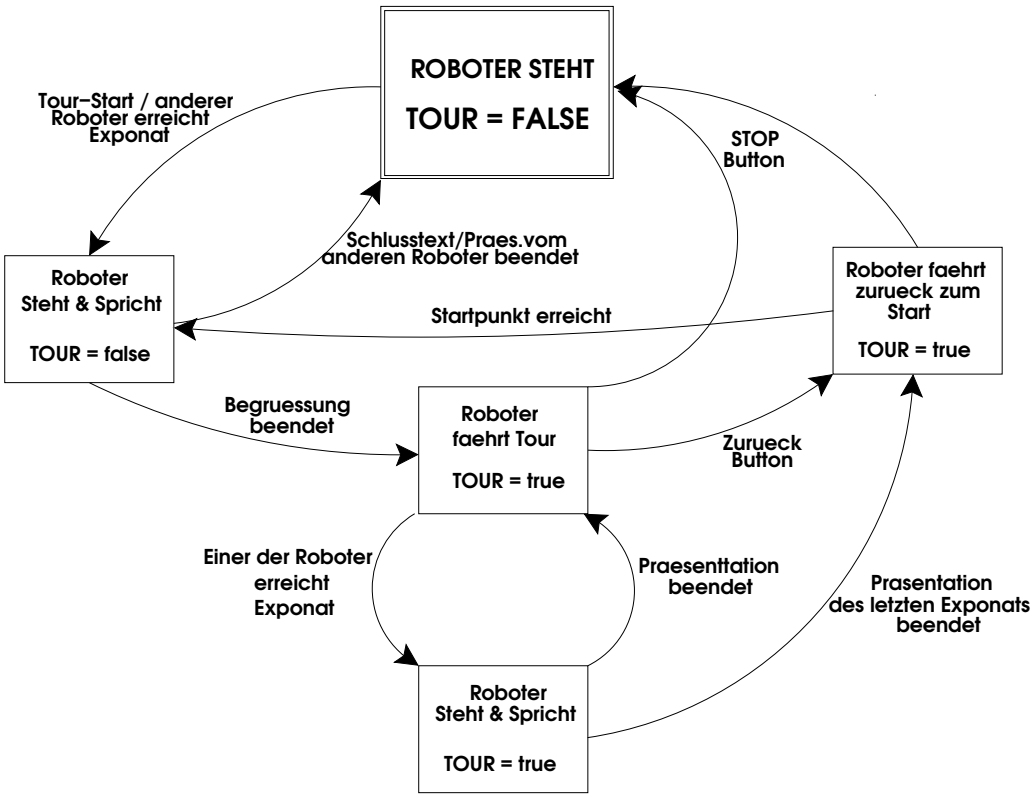

Abbildung 4.7: endlicher, deterministischer Automat

4.2 Programmierung des endlichen Automaten und Einbettung in TCX

Die Programmierung des endlichen Automaten und die Einbettung in die TCX-Architektur hängen sehr eng zusammen, da die Initiatoren, die den Automaten vom einen in den anderen Zustand wechseln lassen, sehr oft TCX-Nachrichten von anderen Modulen sind. Prinzipiell wurde der endliche Automat mit Hilfe von globalen Variablen programmiert, die jederzeit von jedem anderen Programmteil einsehrbar sind. Die wichtigste der globalen Variablen ist hierbei die Variable "Tour". Ist sie "true" gesetzt, dann ist klar, dass der Roboter unterwegs auf einer Museumstour ist. In einer Schleife wird diese Variable ständig abgefragt und die Tour wird gestartet, sobald die Variable durch Benutzerinteraktion auf "true" gesetzt wurde. Es kommt dann auch innerhalb des endlichen Automatens zu einem Zustandswechsel. Es gibt natürlich noch weitere Variablen, damit gewährleistet ist, dass sich alle Zustände des endlichen Automaten logisch unterscheiden, und es nicht zu Verwechslungen kommen kann.

Zur Reaslierung der Fahr -und Pantiltbefehle wurde das Modul "HLI" verwendet. Dieses Modul liefert bereits vorgefertigte Routinen zum Anfahren eines Zielpunktes, Hindrehen und Ausrichten des Pantilts in Richtung eines anderen Punktes. Der Vorteil dieses Moduls ist, dass es eine Reihe von Handlern zur Verfügung stellt, die für die Funktion des Automaten von entscheidendner Bedeutung sind. Das HLI schickt beispielsweise eine Nachricht an das Modul USERINTERFACE und damit an den Automaten, wenn der Roboter einen Zielpunkt erreicht und sich zum Exponat hingedreht hat. Diese Tatsache erspart es einem, eine ständige Ortsabfrage durchzuführen, um zu wissen, wann der Roboter sein Ziel erreicht hat. Darüber hinaus kommt man wie von selbst, im endlichen Automaten in den nächsten Zustand, weil man in diesem Fall nur warten muss, bis man die entsprechende Nachricht vom Modul HLI erhält.

Dann kann mit der Präsentation des Exponats begonnen werden. Dies geschieht zum Einen über eine visuelle Darstellung des Exponats im GUI (siehe 3.3) und zum Anderen durch die Sprachausgabe über das Modul SPEAK. Dieses Modul kann einen Text-String direkt und "on the fly" in Sprache generieren, die über die Lautsprecher des Roboters ausgegeben wird. Dieses Modul schickt wiederum eine Nachricht, sobald die Sprachausgabe beendet ist, was den Automaten wiederum in einen anderen Zustand überführt.

Betrachtet man das Automatenmodell aus Abschnitt 3.1, sieht man sofort, dass sich jeder Zustand außer dem Terminalzustand dadurch auszeichnet,

dass der Roboter entweder fährt oder Spricht. Es ist also immer gesichert, dass das Modul USERINTERFACE von einem anderen Modul eine TCX-Nachricht erhält, die den Automaten in einen anderen Zustand überführt. Jetzt ist lediglich noch darauf zu achten, dass man durch die geeignete Wahl von Variablen den Zustandsraum so konzipiert, dass auf jeden Zustand der richtige Zustand folgt. Aus diesem Grund, muss auch ständig getestet werden, ob die Module SPEAK und HLI aktiv sind, da ein Betrieb des USERINTERFACEs sonst nicht möglich wäre. Die Nutzung den Pantilts wurde variabel gehalten. D.h. es ist möglich, auf Pantiltbewegungen zu verzichten, ohne dass der Programmablauf gestört wird. Dies erschien im Vorfeld sehr wichtig, da es bei der Nutzung des Pantiltes in der Vergangenheit oft Schwierigkeiten gab.

In Abbildung 4.8 wird der Aufbau der Robotersoftware und den damit verbunden Kommunikationsstrukturen zwischen den Modulen dargestellt. Hier noch eine kurze Erklärung des Schaubildes: das Modul USERINTERFACE kommuniziert mit SPEAK und HLI. Das HLI-Modul gibt die Kommandos an das PANTILT und den BEE-Soft-Planner weiter. Dieser berechnet einen zu fahrenden Pfad und gibt Fahrkommandos an die BASE weiter, welche ihre Koordinaten an LOCALIZE schickt. LOCALIZE vergleicht diese Daten mit den Daten des Laser-Range-Scanners und gibt die korrigierten Positionsdaten zusammen mit den Enfernungsmessungen des Lasers an den COLLI-Server weiter. Dieser generiert daraus die Aktuelle Situation bezüglich von Hindernissen auf dem Pfad des Roboters, die er wiederum an den BEE-Soft-Planner zur erneuten Pfadplanung weitergibt.

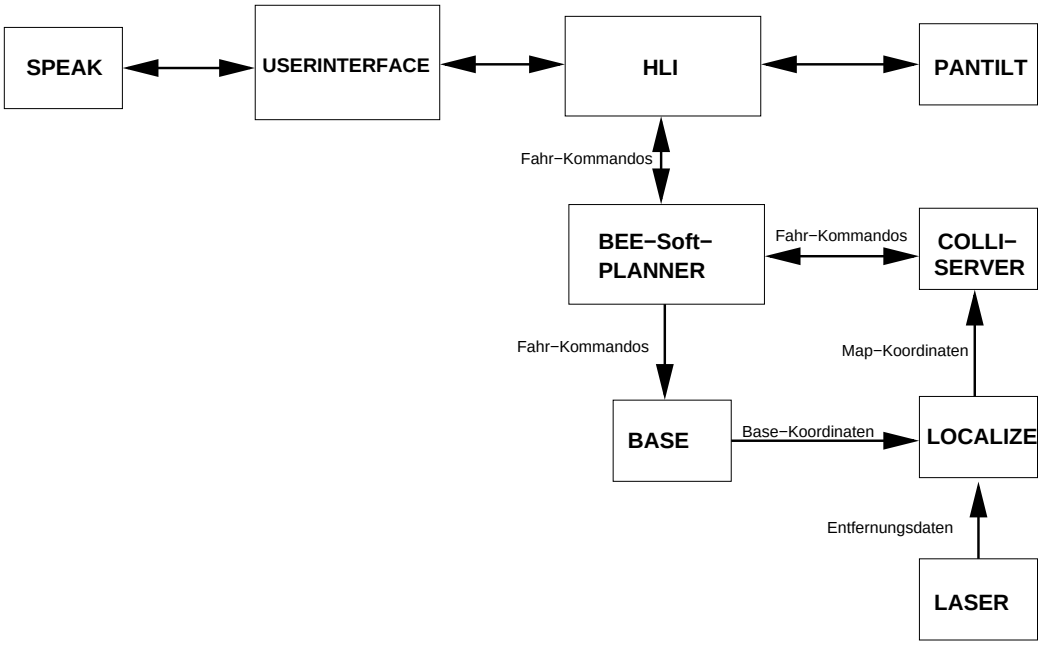

Abbildung 4.8: Kommunikation der Module

4.3 Programmierung des GUI

Bei der Programmierung des GUI wurde vor allem auf maximale Variabilität und Allgemeinheit wertgelegt. So ist beispielsweise die Anzahl der Touren, die dem Museumsbesucher zur Auswahl gestellt werden zwischen 1 und 6 frei wählbar. Außerdem kann man die Kommando-Buttons auswählen, die angezeigt werden sollen. Dies sind Buttons für die Kommandos "Stop", "Pause", "Tour fortsetzten" und "Beenden/Stop". Weiter ist es möglich die Bildschirmauflösung frei zu wählen, wobei die optisch ansprechendste Lösung nur unter den Auflösungen 1064x768 Pixel und 800x600 Pixel zu nutzen ist. Dieser Umstand liegt darin begründet, dass die Kameras der beiden Roboter nur Bilder in der festen Größe 320x240 Pixel übertragen, was die Variabilität des GUI deutlich einschränkt. Zur Programmierung wurde QT3 verwendet. Wie Abbildung 4.3 zeigt, sind oben in einer Reihe die Buttons zur Tourauswahl, darunter zwei Videofenster und wiederum darunter das Textfeld zum Anzeigen von Texten. Ganz unten befinden sich dann ebenfalls in einer Reihe die Kommando-Buttons. Die Buttons sind QPushButton Objekte und die beiden Videofenster sind vom Typ QImage. Das Textfeld ist ein QLabel. Damit das Textfeld und die Videofenster immer das gewünschte anzeigen, laufen hierfür Testschleifen, die den Zustand des Automaten prüfen und dementsprechend die Anzeige steuern. Prinzipiell gibt es 3 Fälle:

1. Der Roboter ist unterwegs auf einer Museumstour oder steht noch an seinem Ausganspunkt für die Tour und der Automat befindet sich noch im Terminalzustand. In diesem Fall zeigt, das linke Videofenster das Videobild an, das von der eigenen Kamera aufgenommen wird. Auf dem rechten Videofenster erscheint ein Standbild.

2. Dieser Fall tritt auf, wenn der Roboter an einem Exponat angelangt ist. Dann wird zusätzlich im rechten Videofenster ein Katalogbild des Exponats und im Textfeld eine kurze Beschreibung angezeigt.

3. Wenn der Roboter ein Exponat des anderen Roboters präsentiert, wird im linken Videofenster das Katalogbild und im rechten Videofenster das Livebild aus dem anderen Museum angezeigt, das via Internet übertragen wird.

In Abbildung 4.3 sieht man den 1.Fall mit einem "Tour-Start-Button" in der oberen Reihe.

Abbildung 4.9: GUI

4.4 Programmierung des Image-Clients

Der Image-Client ist für das Anzeigen der Livebilder verantwortlich. Er empfängt die Bilder vom Imageserver, der die Bilder der Roboterkamera zur Verfügung stellt. Der Imageserver verwendet den Host des Roboters und sendet die Bilder über ein QSocket-Objekt von QT. Das QSocket-Objekt des Clients *horcht* auf dem Port, auf den der Server die Bilder ablegt. Es gibt folgende Basis-Operationen, die der Image-Client ausführen kann. *"Connect", "Disconnect", "start request", "stop request"*. Connect und Disconnect erklären sich von selbst. Start request wird benutzt, wenn man in einem Videofenster, indem bis jetzt ein Standbild angezeigt wurde, ein Livebild anzeigen möchte. Umgekehrt verwendet man stop request, um vom Livebild auf ein Standbild umzuschalten, ohne die Socketverbindung zu trennen. Da es manchmal zu Problemen bei der Übertragung der Bilder sowohl auf Server als auch auf

Clientseite kommt, ist es sinnvoll, ständig zu prüfen, ob noch Bilder empfangen bzw. gesendet werden und falls nicht, die Verbindung zu trennen und neu aufzubauen. Mann führt also hintereinander einen Disconnect und einen Connect mit dem Image-Server aus.

4.5 Kommunikation der beiden Roboter

Fuer die Kommunikation der beiden Roboter benötigt man jeweils einen Host und einen Port. Die Kommunication erfolgt hier ebenfalls über einen QT QServerSocket. Jeder Roboter horcht den Port des anderen Roboters ab und kann so Nachrichten vom anderen Roboter empfangen. Der Roboter, der an einem Exponat angekommen ist, sendet in Form einer Integer-Zahl die Nummer des Exponates an den anderen Roboter. Die Exponatsdaten, wie Bilder, angezeigte und gesprochene Texte werden auf beiden Robotern parallel in einem ini-file gespeichert, so dass der Roboter, der die Meldung erhält, sofort "bescheid weis", an welchem Exponat sich der andere Roboter zu diesem Zeitpunkt befindet. In Abbildung 4.5 sieht man den Fall, dass Roboter 1 gerade ein Exponat erreicht hat und eine Nachricht an Roboter 2 schickt, woraufhin beide Roboter eine Präsentation des Exponats durchführen.

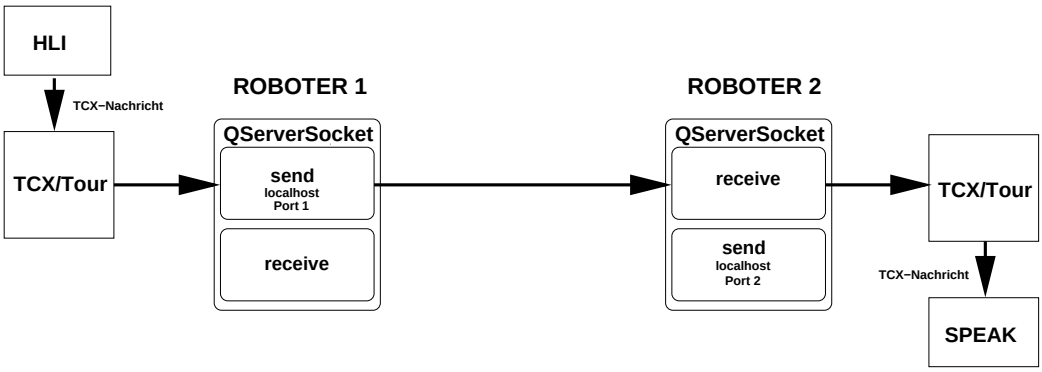

Abbildung 4.10: Kommunikation zwischen den Robotern

Kapitel 5

Verlauf der Tests

5.1 Suboptimales Verhalten des endlichen Automaten

Im Vorfeld der Roboterpräsentation im Deutschen Museum musste die neue Software natürlich ausgiebig getestet werden. Dabei fiel besonders auf, dass man bei der Programmierung eines endlichen Automaten sehr viele Zustände berücksichtigen muss, die bei der Planung und Implementierung, was in Kapitel 3 und Kapitel 4 beschrieben wird, nicht direkt auffallen. Kommt ein Zustandswechsel durch eine Benutzerinteraktion zustande, so ist es beispielsweise notwendig, sich immer zu merken, durch welche Benutzerinteraktion der Zustandswechsel bewirkt wurde. Diese Tatsache muss man dann in speziellen Variablen speichern, damit es zum Beispiel nicht passiert, dass mehrere verschiedene Touren gleichzeitig gestartet werden können. Derartige Probleme traten in den Tests häufiger auf, da durch die hohe Anzahl der Buttons, die TCX-Nachrichten und die Möglichkeit der Kommunikation der beiden Roboter ein sehr großer Zustandsraum geschaffen war, indem noch nicht alle Fälle abgefangen wurden. Es stellte sich in diesem Zusammenhang heraus, dass es von Vorteil ist, wenn der Automat weis, in welchen Zuständen er war, bevor sich sein Zustand abermals ändert, da dies den Möglichen Zustandsraum deutlich minnimiert. Merkt sich der Automat zum Beispiel, dass eine Tour gestartet wurde, so kann es nicht passieren, dass eine zweite Tour gestartet wird, bevor die letzte Beendet wurde. Das sind jetzt nur sehr einfache Beispiele, aber die Zusammenhänge der komplizierteren Probleme zu beschreiben, würde eindeutig zu weit führen.

Ein weiteres Beispiel suboptimalen Verhaltens war zum Beispiel der Effekt,

dass der Roboter beim Erhalt der Nachricht, er solle ein Exponat des anderen Roboters Präsentieren, seinen aktuellen Zielpunkt verlor und einen ganz anderen versuchte anzufahren. Dies lag daran, dass ein Integer-Wert mit der Exponatennummer des zu zeigenden Exponat empfangen wurde, die sich dann im Automaten als die Exponatennummer des aktuell anzufahrenden Exponates einschlich. Dies wurde ebenso dadurch gelöst, dass man sich einfach die Nummer des letzten anzufahrenden Exponates merkt und sich nach der Präsentation des Fremdexponat einfach wieder an diesen alten Zustand zurückerinnert.

Weiter gab es Probleme, wenn Exponate sehr nahe zusammenlagen, so dass innerhalb kurzer Zeit zwei Nachrichten an den anderen Roboter gesendet wurden. Nachdem er das erste Exponat praesentiert hatte, fing dieser an loszufahren und während der Fahrt vom zweiten Exponat zu berichten. Alle diese Fehler traten auf, weil es meistens Fälle waren, mit denen man so im Vorfeld nicht gerechnet hatte. Sie waren zum Glück überwiegend leicht zu beheben, da das Grundgerüst des Automaten korrekt implementiert war und so häufig nur noch weitere meist boolsche Variablen eingefügt werden mussten.

5.2 Kommunikationstest

Die Kommunikationstests in der Zeit vor dem Event sollten sich als schwieriger und anstrengender herausstellen als erwartet. Da das Computernetz in beiden Museen durch Firewalls geschützt war, ging man davon aus, die Verbindung ins weltweite Netzwerk und damit zum anderen Roboter via SSH-Tunnels herstellen zu müssen. Der glückliche Unstand war, dass die Kollegen in Bonn bereits einige Wochen vor dem Event in das Museum konnten, um genau diese Technik zu testen. Wäre dies nicht der Fall gewesen, wäre das Projekt vermutlich gescheitert. Es stellte sich nämlich im Verlauf mehrer Test heraus, dass diese Technik einfach zu instabil war und der Tunnel während eines mehrstündigen Tests des öfteren zusammenbrach. Das Tunneling musste also verworfen werden, was die Kommunikation erheblich stabiler machte. Die Kommunikation funktionierte daraufhin sofort und lief sehr zuverlässig ohne weitere Probleme.

5.3 Fahrtest

Ein ebenfalls sehr wichtiger Test war der Fahrtest. Er beeinhaltete Tests der HLI, BASE und PLAN-Software zur direkten generierung von Fahrkommandos und tests der Kollisionsvermeidung, die in Umgebungen mit vielen Menschen unausweichlich ist. Zu Beginn des Test gingen bei der Pfadplanung immer wieder Zwischenwerte verloren, da aus nicht ganz erklärlichen Gründen der Rechner des Roboters Albert etwas überfordert war. Der Planer gab die Meldung "Busy" aus und schaffte es nicht in annehmbarer Zeit einen Pfad um Hindernisse herum zu generieren. Trat ein Hinderniss vor den Roboter, so verweilte er zuerst an Ort und Stelle und begann dann auf das Hindernis zuzufahren. Diese Probleme konnten dadurch behoben werden, dass im ini-file der Kollisionsvermeidung, "collimodes.ini", das Beschleunigungs- und Bremsverhalten angepasst wurde. Die Probleme traten nämlich deswegen auf, weil die Bremsverzögerung zu niedrig war. Der Roboter schaffte es also nicht, rechtzeitig zu bremsen, wenn ein Hinderniss auftauchte oder wenn er versuchen wollte, durch einen engen Korridor hindurchzufahren. Er stand dann Praktisch schon mitten "in" seinem Hinderniss und musste dann den Pfad zu seinem Zielpunkt umplanen, was auch die Überlastung des Pfadplanungsprogramms erklärt. Als die Test abgeschlossen waren, fuhr der Roboter mit dieser Software recht passabel.

Man muss sich allerdings die Frage gefallen lassen, warum überhaupt auf den BEE-Soft planer zurückgegriffen wurde und nicht der "Planner5d" von Cyrill Stachniss verwendet wurde, mit dem der Roboter wesentlich runder und zügiger um Hindernisse herummaneuvriert. Die Antwort ist folgende: zur Realisierung des endlichen Automaten war es nötig, oder zumindest von großem Vorteil, TCX-Nachrichten zu erhalten, sobald der Roboter einen Zielpunkt erreicht hat. Ausserdem war es nötig den Roboter zu einem Exponat hindrehen zu können und wiederum eine TCX-Nachricht am ende des Vorganges zurückzubekommen. Diese erleichternden Features wurden aber nur vom HLI-Modul bereitgestellt, das zur Pfadplanung den BEE-Soft-Planer verwendet.

Kapitel 6

Verlauf des Robotalk

6.1 Beobachtungen und Benutzerverhalten

Im Verlauf des Robotalks konnte eine Vielzahl von Beobachtungen gemacht werden, die sich auf die Planung weiterer solcher Events durchaus auswirken dürfte. Es wurde schnell klar, dass bei den Museumsbesuchern zwar durchaus Interesse an den Exponaten besteht, die der Roboter Präsentiert, es aber wesentlich unterhaltsamer ist, den Roboter beim Fahren zu beobachten. Diese Tatsache war den Verantwortlichen des Museums im Vorfeld so wohl nicht ganz klar und deshalb waren die Texte, die von den Robotern gesprochen wurden, oft einfach zu lang. Die Besucher verloren vermutlich auch schnell das Interesse daran, den Beschreibungen des Roboters zu folgen, da diese zum Teil doch sehr schwer verständlich waren und die Stimme des Roboters durchaus gewöhnungsbedürftig ist. Ausserdem schien es vor allem Kinder viel mehr zu faszinieren, sich dem Roboter in den Weg zu stellen, und abzuwarten, wie er wohl darauf reagieren würde. Da der Roboter in der Lage ist durch Sprache und Bewegungen in seinem Gesicht, Emotionen zu zeigen, war dieses "Testen" des Roboters die scheinbar spannendste Art sich dem fremden Objekt zu nähern. Dies sieht man auch auf dem Bild 6.1.

Abbildung 6.1: Kinder beim Testen des Roboters

Weiterhin konnte man beobachten, dass die meisten Besucher mit den Interaktionsmöglichkeiten, die das Userinterface bietet, überfordert waren. Dies führte häufig zu unkontrolliertem Betätigen von Knöpfen und Ratlosigkeit, wenn dies nicht den gewünschten Erfolg brachte. Auf dieses Problem werde ich aber noch im nächsten Abschnitt und in Abschnitt 6.3.1, der die Auswertung behandelt, genauer eingehen.

6.2 Probleme und Modifikationen

Wie oben bereits angesprochen war eines der Hauptprobleme, dass die Museumsbesucher durch ihr Eingreifen auf dem Touch-Sreen den Betrieb des USERINTERFACEs zeitweise nahezu lahm legten. Dies hatte entweder zur Folge, dass der Roboter oft irgendwo im Museum herumstand, da jemand eine Tour abgebrochen aber keine neue begonnen hatte oder dass der Roboter nur immer den Anfang einer Tour fuhr. Dies geschah, weil die Besucher offensichtlich Spass daran hatten, die Funktion aller Buttons auf dem Touch-Sreen auszuprobieren und die Tour meist nach wenigen Metern schon wieder abbrachen. Also wurden diese Buttons kurzerhand deaktiviert und es war dem Besucher nurnoch möglich, die Tour zu starten. Dies verbesserte den Betrieb erheblich da die Touren ab sofort immer zu Ende gefahren wurden. Die Versuche der Besucher in den Betrieb einzugreifen wurden aber dennoch weiterhin dockumentiert und gezählt. Dies ist in Abschnitt 6.3.1 dokumentiert.

Ein anderes Problem konnte während des Robotalks allerdings nicht behoben werden. So ergaben sich erhebliche Schwierigkeiten aus der Tatsache, dass die Grundrisse der beiden Museumsabteilungen doch sehr unterschiedlich waren. Konkret heist das, dass die Abteilung in München viel kleiner war und die Wegstrecken zwischen zwei Exponaten sehr gering war. In Bonn war es genau umgekehrt. Dort standen die Exponate sehr weit auseinander. Außerdem waren die Exponatstexte, die in München vorgetragen wurden, um ein Vielfaches länger als die in Bonn. Dies führte dazu, dass der Roboter Albert in München seine Touren mehr oder wenig unbehelligt fahren konnte, da er ja selbst meisten am "sprechen" war. Sein Kollege Rhino in Bonn hingegen wurde ständig auf den langen zurückzulegenden Wegstrecken aufgehalten und kam garnicht dazu, die eigenen Exponate zu Präsentieren. Beachtet man zusätzlich den Umstand, dass sehr viele Leute im Museum in Bonn unterwegs waren, was die Fortbewegung Rhinos zusätzlich verlangsamte, kann man sich vorstellen, wie lange eine Museumstour in Bonn wohl gedaurte haben muss. Dieser Fehler konnte nur zum Teil behoben werden, indem die Texte von München in der Länge denen von Bonn angepasst wurden.

Zusätzlich zu diesen Problemen, gab es gelegendlich auch noch Schwierigkeiten in der Art, wie sie bereits beim Testen aufgefallen waren. So gab es Probleme, die von Inkonsistenzen innerhalb des endlichen Automaten herrührten, wie sie bereits in Abschnitt 5.1 beschrieben wurden. Dies kam daher, dass durch die ständigen Probleme bei der Kommunikation im Verlauf der Tests, die Tests für den endgültigen Betrieb praktisch ausgefallen sind. Dies kam zum einen durch die ständigen Tunnelprobleme beim testen und zum

anderen durch die mangelhafte Vorbereitung seitens des Deutschen Museums in München, weswegen der erste Tag, der eigentlich zum Test der Software gedacht war, ausfiel.

6.3 Auswertung

Um nach Ablauf des Events eine Auswertung vornehmen zu könnnen, mussten selbstverständlich ersteinmal eine ganze Menge an Daten gesammelt werden. Dies wurde auf zwei unterschiedliche Art und Weisen getan. Zum Einen wurden alle HLI-Daten durch einen Recorder aufgezeichnet. Diese Recorder-Daten beinhalten Localize-Korrektur-Parameter, Positionsdaten, die Länge der Zeitintervalle zwischen den einzelnen Messungen und die Entfernungen des Laser-Range-Scans. Zum Anderen wurde vom USERINERFACE eine Meldung in ein Logfile geschrieben, wenn es zu Zustandsänderungen des endlichen, deterministischen Automaten kam.

6.3.1 Auswertung der HLI-Recorder-Daten

Zur Auswertung der HLI-Recorder-Daten war es notwendig, ein kleines Programm zu implentieren, das aus den Korrektur-Parametern von LOCALIZE und den BASE-Koordinaten die Map-Koordinaten generiert, damit die Positionen des Roboters in die Karten der Museen eingezeichnet werden können.

Deutsches Museum München

Wie man anhand der Karte des Deutschen Museums sehen kann, war es in München sehr eng und der Roboter hatte relativ kurze Wege zurückzulegen. Außerdem waren im rechten Teil der eingzeichneten Pfade mehrere Exponate, so dass der Roboter in diesem Bereich auf einer Tour mehrmals an den selben Stellen vorbei fuhr. Deshalb ist die Karte weiter rechts auch etwas schwärzer.

Donnerstag

Wie man in Abbildung 6.2 sehen kann, ist der Ausgangspunkt der Touren rechts auf der Karte und der Roboter arbeitet sich Korridor für korridor durch die Abteilung Telekomunikation hindurch. Er fährt vom ersten in den zweiten, dann wieder in den ersten, nocheinmal in den zweiten Korridor um dann

von dort aus in den dritten korridor hineinzufahren, wo er zum Abschluss eine Blau-Wand-Demonstration erklärte.

Abbildung 6.2: Map Donnerstag

Auf Bild 6.3 kann man schön sehen, wie Albert ein Exponat anschaut und erklärt.

Abbildung 6.3: Albert erklärt ein Exponat

Freitag

Wie man auf der Karte in Abbildung 6.4 sehen kann, fuhr der Roboter einmal direkt in die Blauwand-Demonstration hinein, wo dann das Foto entstand, das unter der Karte zu sehen ist. Auf der Karte ist diese Stelle oben links und man erkennt, dass der Roboter dort nur einmal gewesen sein kann, eben zur Aufnahme des Bildes 6.5.

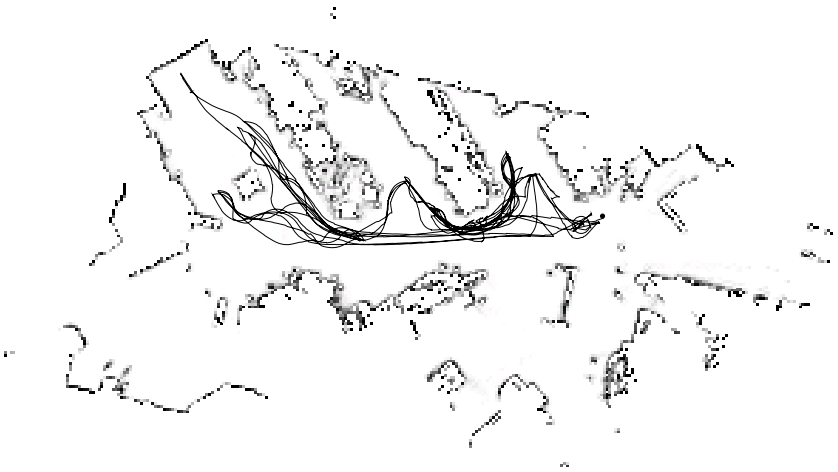

Abbildung 6.4: Map Freitag

Abbildung 6.5: Albert vor der Blauwand

Samstag

Am Samstag und Sonntag waren in München aufgrund des schönen Wetters nicht mehr so viele Museumsbesucher vor Ort wie noch am Donnerstag. Deshalb ist der Bereich, indem sich der Roboter bewegt hat, deutlich schmäler, da er nur wenigen Hindernissen ausweichen musste. Siehe Abbildung 6.6 und Abbildung 6.7.

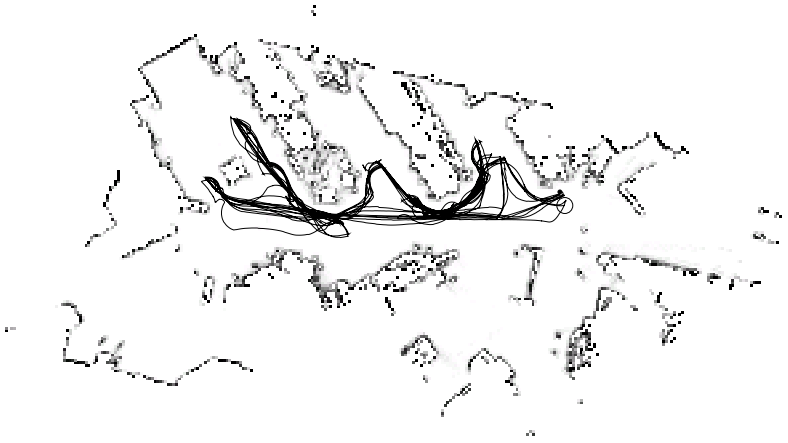

Abbildung 6.6: Map Samstag

Sonntag

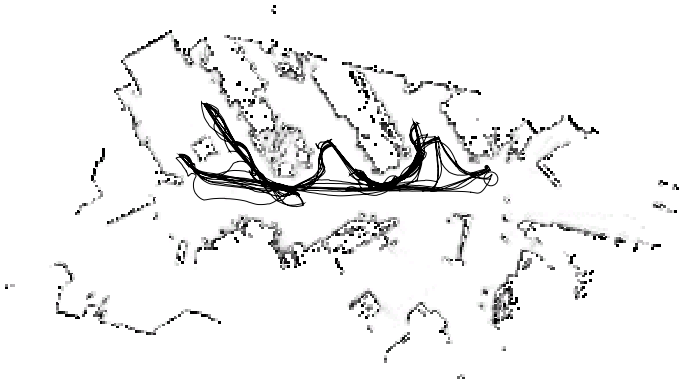

Abbildung 6.7: Map Sonntag

Deutsches Museum Bonn

Hier sieht man nocheinmal deutlich die Unterschiede zwischen den beiden Abteilungen, indenen sich die beiden Roboter bewegen. Während in München lediglich enge Korridore zur Verfügung stehen, hat man hier in Bonn einen weitläufigen Gebäudetrakt und weite Wege zurückzulegen.

Mittwoch

Wie man auf Abbildung 6.8 sieht, war Mittwoch der Tag des ersten Tests. Es wurden nochnicht sehr viele Touren gefahren und die einzelnen Pfade sind noch deutlich zu erkennen.

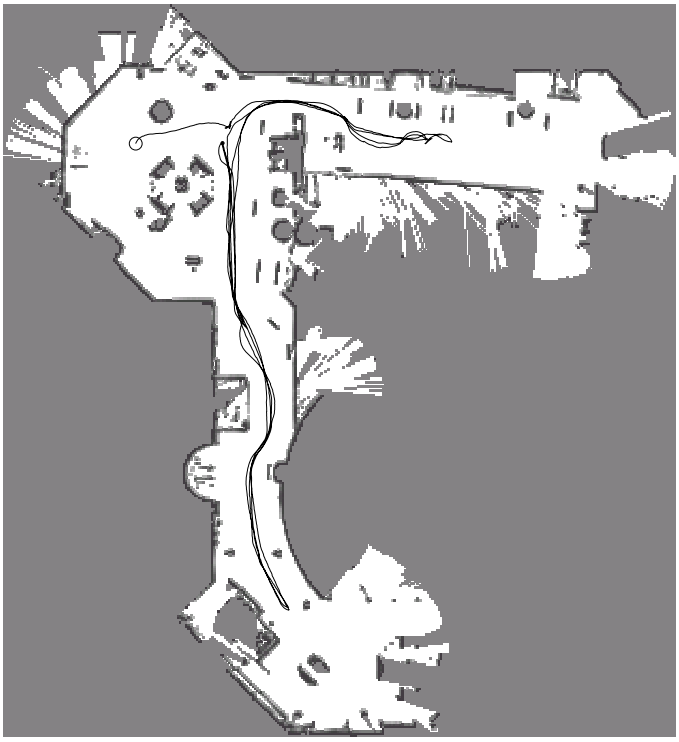

Abbildung 6.8: Map Mittwoch

Donnerstag

Am Donnerstag war auch in Bonn der größte Besucherstrom zu vermelden, wie man auf Abbildung 6.9 deutlich erkennen kann.

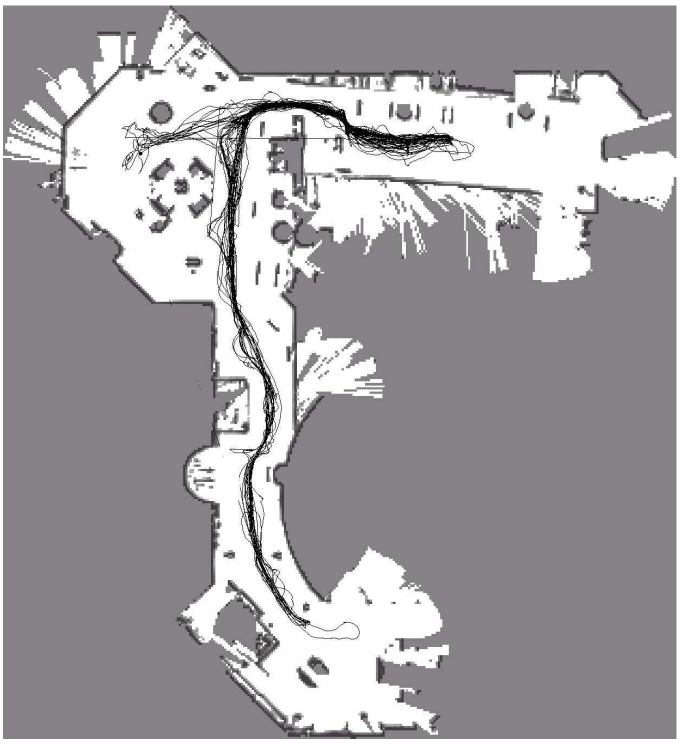

Abbildung 6.9: Donnerstag Map

Freitag

Zu den Aufzeichnungen vom Freitag bleibt nicht so viel zu sagen. Man sieht, dass etwas weniger Besucher im Museum waren als noch am Donnerstag, da in Abbildung 6.10 etwas weniger Pfade eingezeichnet sind.

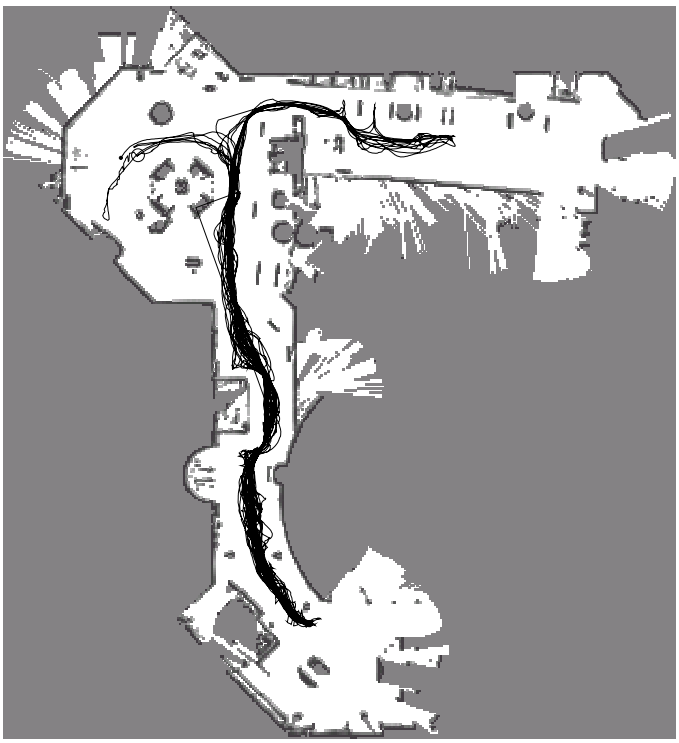

Abbildung 6.10: Map Freitag

Sonntag

Sonntag war der Tag, an dem die Demonstration von Rhino ein abruptes Ende nahm. Unter einem lauten Knall blieb der Roboter stehen und schwarze Rauchwolken begannen, aus dem Antriebsteil emporzusteigen. Aus diesem Grund sind auch nur wenige Pfade in die Karte 6.11 eingezeichnet.

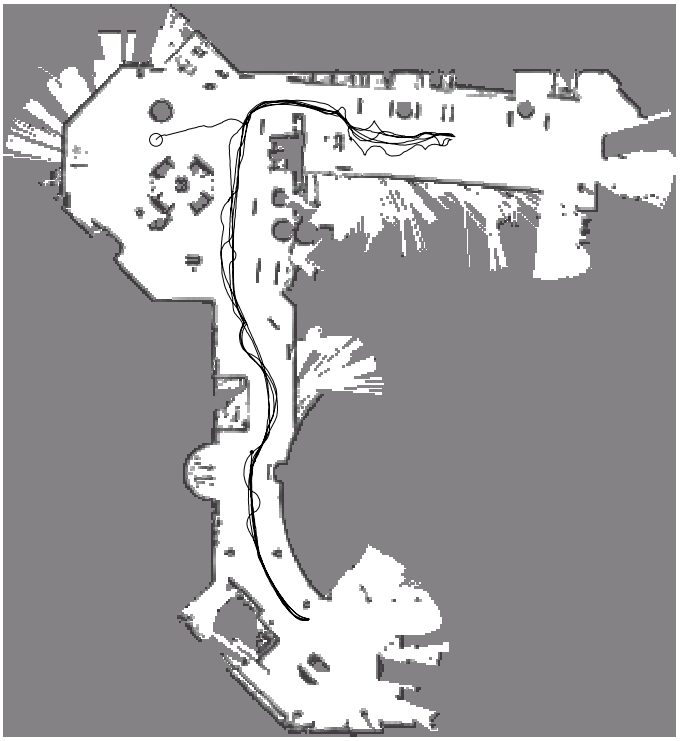

Abbildung 6.11: Map Sonntag

6.3.2 Auswertung der USERINTERFACE-Logs

Um den gesamten Ablauf des Robotalks und das Verhalten der Benutzer im Nachhinein nachvollziehen und beurteilen zu können, wurden alle Zustandsänderungen, des endlichen, deterministischen Automaten innerhalb des USERINTERFACEs registriert und zusammen mit der aktuellen Zeit aufgezeichnet.

Deutsches Museum München

Die Auswertung ergab folgende Werte. Die Tour wurde an 4 Tagen 256 mal gestartet und 60 mal komplett gefahren. Die Differenz zwischen Starts und Ankünften am Endpunkt kommt daher, dass es zu Beginn für den Besucher noch möglich war, die Touren abzubrechen. Ein weiterer Grund war, dass die Tour auch häufig mit Absicht beendet wurde, wenn keine Besucher mehr in der Nähe waren, um die Batterien des Roboters wieder laden zu können. 204 mal wurde ein Exponat vom anderen Roboter akzeptiert und entsprechend Präsentiert. 20 mal musste ein Exponat vom anderen Roboter zurückgewiesen werden, weil der Roboter in dem Fall Albert selbst gerade am sprechen war. Es kam 548 mal zu sinnlosen Besucherinteraktionsversuchen, die deswegen sinnlos waren, weil ab dem zweiten der vier Tage die Funktionen der Buttons auf dem Touch-Screen deaktiviert waren.

Deutsches Museum Bonn

In Bonn ergab die Auswertung die folgenden Ergebnisse. Die Tour wurde an 4 Tagen 292 mal gestartet und 75 mal komplett gefahren. 435 mal wurde ein Exponat vom anderen Roboter akzeptiert und entsprechend Präsentiert. 6 mal musste ein Exponat vom anderen Roboter zurückgewiesen werden, weil der Roboter in dem Fall Rhino selbst gerade am sprechen war. Es kam 924 mal zu sinnlosen Besucherinteraktionsversuchen.

Vergleich München - Bonn

Im Vergleich fällt auf, das Bonn mehr als doppelt so viele Exponate aus München präsentiert hat als ungekehrt. Das lag an den bereits beschriebenen differenzen zwischen den beiden Lokalitäten und Exponatsbeschreibungen der Museen in Bonn und München. Diese Auswertung bestätigt nocheinmal die Beobachtungen und Probleme, wie sie in Abschnitt 6.1 und 6.2 dargestellt sind. Auch das beobachtete Verhalten der Museumsbesucher hat sich mehr als bestätigt. So war es doch meistens so, dass die Museumsbesucher zuerst einmal ohne besonderes Nachdenken einige Buttons drückten, die sie vermutlich nach ihrer Lieblingsfarbe ausgewählt hatten.

Kommunikation der Roboter

Die Abbildungen 6.13 und 6.12 zeigen folgende Konstellation: Albert in München und Rhino in Bonn fahren zur gleichen Zeit eine Museums-Tour. In Abbildung 6.12 sieht man die Stellen, an denen sich die Exponate befinden. Sie wurden der Reihe nach von oben nach unten entland des eingezeichneten Pfades angefahren.

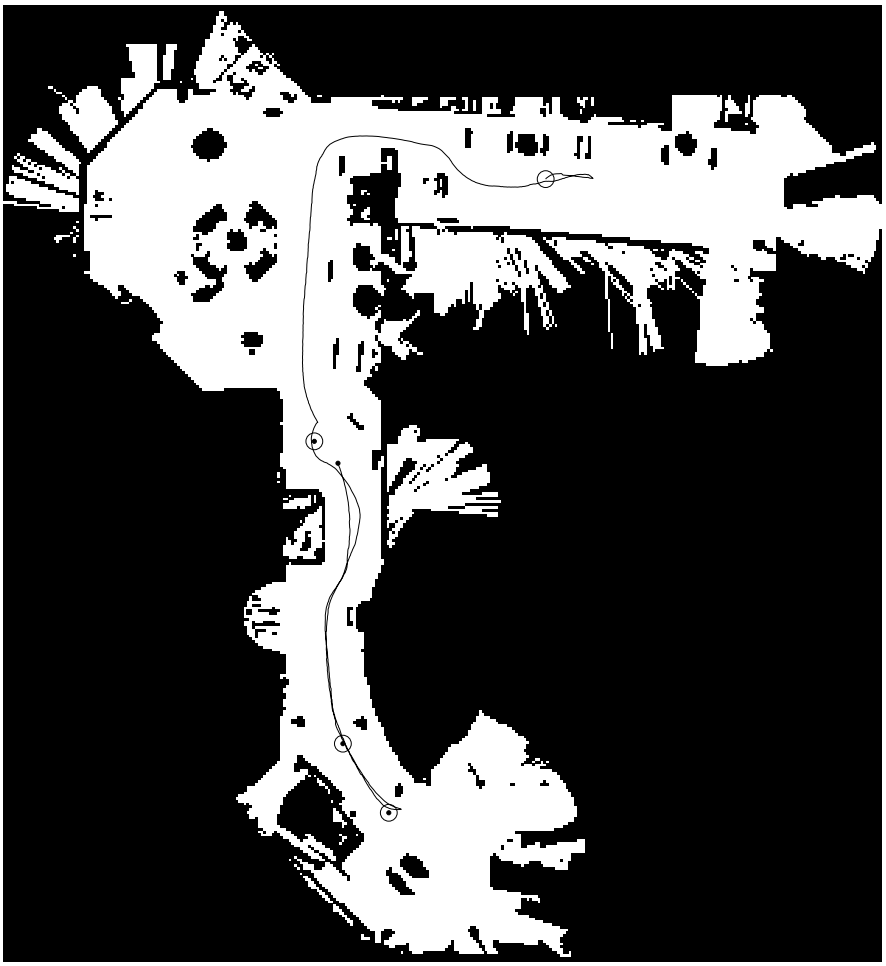

Abbildung 6.12: Communication Map Bonn

In Abbildung 6.13 hingegen sind die Roboterpositionen von Albert eingezeichnet, an denen er eine Nachricht von Rhino erhalten hat, dass dieser sich gerade an einem Exponat befindet. An diesen Stellen hat Albert daraufhin gestoppt und die Exponate aus Bonn präsentiert. Die am weitesten rechts eingezeichnetet Position war die erste Stelle. Die Reihenfolge der anderen beiden wird klar, wenn man den eingezeichneten Pfad verfolgt.

Abbildung 6.13: Communication Map München

Kapitel 7

Zusammenfassung

7.1 Erfahrungen und Lehren

Die Zeit vor, während und nach dem Roboterevent war sehr lehrreich für mich und ich denke, dass mir die gemachten Erfahrungen in Zukunft entscheidend weiterhelfen können. Eine der Haupterfahrung, die ich aus dem Projekt gezogen habe, ist die Tatsache, dass man nie früh genug anfangen kann, eine neue Software zu testen, da man nie weis welche Probleme auftreten werden. In dem Fall war es ja sogar so, dass derartige Netzwerkprobleme auftraten, dass das Testen fast unmöglich wurde. Alles in allem war ich aber nach Abschluss des Projektes überaus zufrieden, da es mein erstes derartiges Projekt war und ich anfangs garnicht wusste, was auf mich zukommt. Darüber hinaus wurde mir klar, wie schwer es ist, ein mögliches Benutzerverhalten zu kalkulieren und dass es eigentlich am besten ist, dem Benutzer nur so wenige Chancen wie möglich zu geben, das Geschehen zu beeinflussen. Will man es anderst machen, muss man dem Benutzer, in dem Fall dem Museumsbesucher wohl erst das Programm erklären. Zum guten Schluss möchte ich noch anmerken, dass die Zustände, die wir bei der Ankunft im Deutschen Museum vorfanden erbärmlich waren. Es hat mir einmal mehr gezeigt, dass man sich bei der Organisation eines solchen Events nicht auf Menschen ausserhalb der eigenen Abteilung verlassen sollte. Falls man doch darauf angewiesen ist, sollte man genügend Zeit mitbringen, damit man vor Ort nicht so unter Zeitdruck gerät, dass man keine Zeit für seine eigentlichen Vorhaben hat.

7.2 Dank

Mein Dank gilt in erster Linie der Abteilung für Autonome Intelligente Systeme mit Prof. Dr. Wolfram Burgard, Dirk Hähnel, Maren Bennewitz, Cyrill Stachniss und Dirk Ziterell, ohne deren tatkräftige Hilfsbereitschaft und den Willen Fragen zu beantworten die Verwirklichung dieses Projektes sicher nicht möglich gewesen wäre. Außerdem gilt mein Dank Mark Moors von der Universität in Bonn. Danken möchte ich auch den beiden Hauptdarstellern Albert und Rhino, die mit großem Einsatz mitgeholfen haben, dieses Projekt zu verwirklichen.

7.3 Team-Bilder

Als kleinen Anhang füge ich hier noch ein paar Bilder der fleisigen Teammitglieder ein, die selbst in den Museen vor Ort waren.

Abbildung 7.1: von links: Albert, ich selbst und Prof.Burgard

Abbildung 7.1: Das Münchner Team

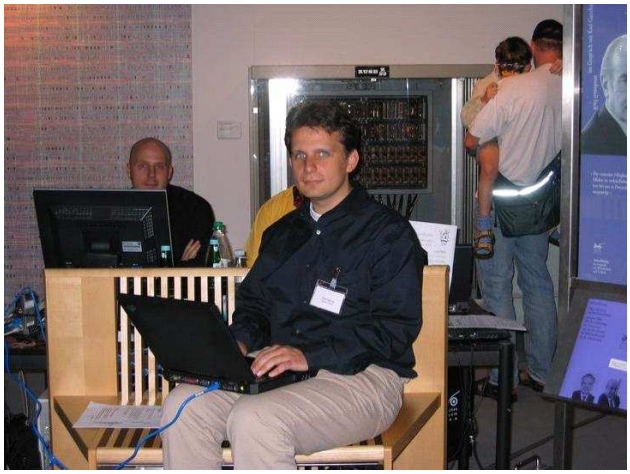

Abbildung 7.2: Dirk Hähnel in Bonn

Abbildung 7.3: von links: Mark Moors und Dirk Zitterel in Bonn

Literaturverzeichnis

[1] W. Burgard, A.B. Cremers, D. Fox, D. Haehnel, G. Lakemeyer, D. Schulz, W. Steiner, S. Thrun.
Experiences with an interactive museum tour-guide robot.
In *Artificial Intelligenz*, 114(1-2), 2000.

[2] W. Burgard, A.B. Cremers, D. Fox, D. Haehnel, G. Lakemeyer, D. Schulz, W. Steiner, S. Thrun.
The interactive museum tour-guide robot.
In *Proc. of the National Conference on Artificial Intelligenz*, 1998.

[3] S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte and D. Schulz.
MINERVA: A Second-Generation Museum Tour-Guide Robot.
In *Proc. of the IEEE International Conference on Robotics Automation(ICRA'99)*, 1999.

[4] The TOURBOT project. http://www.ics.forth.gr/tourbot/.

[5] The WEBFAIR project. http://www.ics.forth.gr/webfair/.

[6] D. Schulz, W. Burgard, A.B. Cremers, D. Fox, S. Thrun.
Web interfaces for mobile robots in public places.
In *IEEE Robotics and Automation Magazine*, 7(1), 2000

Abbildungsverzeichnis