# Umsetzung und Evaluation eines mit Laserscannern gesicherten Rollstuhls als interaktives Museumsexponat

Diplomarbeit an der Universität Bremen
Fachbereich Mathematik und Informatik
Arbeitsgruppe von Prof. Dr. Bernd Krieg-Brückner

Jost Gollub

Betreuer: Dr. Udo Frese, Dr. Thomas Röfer

Gutachter: Dr. Udo Frese, Prof. Dr. Bernd Krieg-Brückner

# Inhaltsverzeichnis

## Abbildungsverzeichnis

## Tabellenverzeichnis


Abbildung 1: Plakat der Sonderausstellung 'Computer.Medizin'

# 1 Einführung

## 1.1 Aufbau und Inhalte dieser Arbeit

Diese Diplomarbeit beschäftigt sich mit dem Nachbau des Rolland III, welcher die aktuelle Forschungsplattform des Projekts 'Bremer autonomer Rollstuhl' dar stellt. Der Rolland III ist ein elektrischer Rollstuhl, der durch zusätzliche Hardware zu einem Roboter umgebaut ist.

Im Projekt 'Bremer autonomer Rollstuhl' werden Rollstühle entwickelt, die Menschen mit Behinderungen helfen sollen, einen Rollstuhl zu bedienen. Dabei zielt der Einsatzzweck auf Menschen, die durch ihre Behinderung keinen normalen elektrischen Rollstuhl bedienen können. Dies können Menschen sein, die Probleme mit der Feinmotorik haben und einen Rollstuhl nicht kollisionsfrei steuern können, aber auch Menschen, die durch ihre Behinderung gar nicht in der Lage sind, einen Rollstuhl mit einem Joystick zu lenken.

Nachgebaut wurde der Rolland III für die Ausstellung 'Computer.Medizin' (Abbildung 1), die zunächst im Heinz Nixdorf MuseumsForum in Paderborn gastiert (Abbildung 2). Dort wird der Rolland III als interaktives Museumsexponat ausgestellt. Von den Anwendungen des Rolland III wird in der Ausstellung nur die softwaregestützte Kollisionsvermeidung demonstriert.

Weil das Museumsexponat sich in vielen Punkten von dem Rolland III unterscheidet, wird es, zur besseren Unterscheidung, in dieser Arbeit HNF-Rolland (für Heinz Nixdorf MuseumsForums-Rolland) genannt. Der grundsätzliche Aufbau und die verwendeten Sensoren bleiben erhalten. Während der Rolland III jedoch darauf ausgelegt ist, als Prototyp und Forschungsplattform verschiedenen Anwendungen zu dienen, dient der HNF-Rolland nur einer wesentlichen Anwendung, der Kollisionsvermeidung. Beim HNF-Rolland ist, im Gegensatz zum Rolland III, eine starke Robustheit gefordert, da in der Ausstellung 'Computer.Medizin' bis zu 800 Personen täglich den Rollstuhl testen können.


Abbildung 2: Das Heinz Nixdorf MuseumsForum

Die vorliegende Arbeit beschreibt detailliert den Aufbau der Hard- und Software des HNF-Rollands und insbesondere die Unterschiede zum Rolland III. Der Nachbau und die Anpassung an die veränderten Anforderungen stellt den praktischen Teil der Diplomarbeit dar.

Die Arbeit ist in fünf Kapitel gegliedert. Im Anhang befinden sich Schaltpläne, Bauzeichnungen und Quellcode, die für den Aufbau des HNF-Rolland nötig sind.

Beginnend wird (Abschnitt 1.2) ein Überblick über das Projekt 'Bremer autonomer Rollstuhl' gegeben. Dargestellt wird die Entstehung des Projekts und die bisher daraus hervorgegangenen Prototypen.

Anschließend erfolgt in Abschnitt 2 die Beschreibung der Hardware des HNF-Rollands. Hierbei wird besonders auf die Unterschiede zum Rolland III eingegangen. In diesem Abschnitt werden die Sensoren und der eingesetzte Computer, sowie die benötigten Halterungen und die notwendige Stromversorgung dargestellt.

Der folgende Abschnitt 3 stellt die eingesetzte Software vor. Hierbei wird die Software GTRolland beschrieben, das eingesetzte Betriebssystem und dessen Konfiguration. Es wird speziell auf die Unterschiede zum Rolland III eingegangen. Diese resultieren in diesem Abschnitt vor allem aus den eingeschränkten Funktionen des HNF-Rollands, sowie aus der Tatsache, dass HNF-Rolland und Rolland III verschiedene Betriebssysteme benutzen.

Abschnitt 4 stellt die Evaluation des Fahrverhaltens des HNF-Rollands dar. Dieses wurde, im Vergleich zum Rolland III, stark verändert um den Anforde-

rungen als Museumsexponat gerecht zu werden. Die Evaluation war erst nach einigen Monaten des Einsatzes in der Ausstellung 'Computer.Medizin' abgeschlossen.

Abschließend wird in Abschnitt 5 wird das Fazit gezogen. Hierfür wird das Gesamtsystem des HNF-Rollands zusammengefasst. Außerdem werden die Bewertungen der Besucher der Ausstellung 'Computer.Medizin' vorgestellt. Außerdem erfolgt ein Ausblick auf die bevorstehenden Aufgaben des Projekts 'Bremer autonomer Rollstuhl' und das Vorhaben den Rolland zur Serienreife zu führen.

## 1.2 Das Projekt 'Bremer autonomer Rollstuhl'

Das Projekt 'Bremer Autonomer Rollstuhl' wurde 1995 mit der Entwicklung des Rolland I ins Leben gerufen. Ursprünglich als Testplattform für Neuronale Netze von einem Bremer Biologen entwickelt ([Bühlmeier u. a., 1996] und [Bühlmeier und Herwig, 1995]), wuchs aus diesen Anfängen ein Projekt, das es sich zur Aufgabe gemacht hat, ältere Menschen und Menschen mit Behinderungen, die nicht in der Lage sind, einen elektrischen Rollstuhl sicher zu manövrieren, bei der kollisionsfreien Steuerung eines Rollstuhls zu unterstützen. Zu den Anwendungen gehören Kollisionsvermeidung, Ausweichverhalten und das Navigieren auf bekannten Karten. Seit neuestem wird eine Sprachsteuerung entwickelt, die es ermöglicht, Sprachbefehle zu geben, um bekannte Wege zu befahren.

In der Entwicklung befinden sich ebenfalls zur Zeit Steuerungsmethoden, die eine Neigung des Kopfes als Eingabe ermöglichen [Mandel u. a., 2007], sowie ein reaktives Ausweichverhalten.

In den letzten 12 Jahren wurden drei verschiedene Rollstuhlmodelle durch dieses Projekt zu intelligenten Rollstühlen und erhielten den Namen 'Rolland', oder 'Bremer Autonomer Rollstuhl'.

### 1.2.1 Der Rolland I

Der Rolland I [Röfer, 1997] wurde 1995 gebaut (siehe Abbildung 3). Seine erste Funktion war eine Kollisionsvermeidung. Diese hielt allerdings mit großem Abstand zu den Hindernissen an und funktionierte nur bei sehr geringer Geschwindigkeit. Darüber hinaus wurden einfachere Verhalten, wie eine Wandverfolgung, implementiert. Diese Anwendungen werden später ausführlicher erwähnt.

Für den Rolland I wurde der handelsübliche elektrische Rollstuhl Meyra Modell 3.422 eingesetzt. Dieser Rollstuhl hat eine feste Vorderachse, über die er angetrieben wird. Hinten befindet sich eine Lenkachse, über die die Fahrtrich-

Abbildung 3: Der Rolland I (Bild aus [Röfer, 1997])

tung gesteuert wird. Die Kinematik funktioniert damit wie bei einem Auto, das rückwärts fährt.

Als zusätzliche Hardware wurde in den Meyra Modell 3.422 ein Pentium 100 Computer eingebaut. Zur Raumerfassung dienen sechs Infrarotsensoren, zwölf Ultraschallsensoren und eine Kamera. Des weiteren wurden 12 Stoßabweiser montiert. Diese dienen auch als Kontaktsensoren. Für die Bewegungssensorik wird die rollstuhleigene Odometrie eingesetzt, die sich an der Vorderachse befindet. Da es möglich ist, den Ausschlag der Lenkachse auszulesen, ist es dem Computer, bzw. der Software möglich, sowohl Geschwindigkeit als auch die Bewegungsrichtung zu ermitteln.

Zur Befestigung der zusätzlichen Hardware umgibt ein Rahmen den Rollstuhl an allen Seiten, sogar über dem Kopf des Fahrers befindet sich eine Stange zur Befestigung der Kamera. Dadurch werden die Ausmaße des Meyra Modell 3.422 stark zu allen Seiten vergrößert.

Es kommen zwei verschiedene Arten Ultraschallsensoren zum Einsatz: Acht von ihnen haben einen Scanbereich von 80°, vier von ihnen scannen nur einen Bereich von 7°. Die Infrarotsensoren haben nur eine Reichweite von 15 cm und können Hindernisse nur erkennen, nicht aber die Entfernung zu ihnen bestimmen. Da die Ultraschallsensoren mit weitem Winkel keine genauen Richtungsangaben liefern, werden die 7° Ultraschallsensoren und die Infrarotsensoren zur Navigation, die 80° Ultraschallsensoren zur Kollisionsvermeidung genutzt. Im Nahbereich spielen die Infrarotsensoren auch eine Rolle bei der Kollisionsvermeidung.

Die Sensoren sind um den gesamten Rollstuhl angebracht. Auf jeder der vier

Seiten befinden sich zwei Ultraschallsensoren mit einem Scanbereich von 80°. Von den Infrarotsensoren befinden sich je drei auf der vorderen und der hinteren Seite, wobei alle vier Ecken von je einem Infrarotsensor abgedeckt werden. Die Ultraschallsensoren mit 7° Scanbereich befinden sich ausschließlich an der Vorderseite, zwei sind nach vorne gerichtet, je einer nach rechts und links [Röfer und Müller, 1998].

Als Anwendungen wurden Kollisionsvermeidung, Landmarkenerkennung, Wandverfolgung und Türdurchfahrtsverhalten implementiert [Krieg-Brückner u. a., 1998].

Ein wirkliches Ausweichverhalten wurde zwar nicht definiert, jedoch eine Lenkeinschränkung: Durch die Lenkung mit der Hinterachse kann der Rollstuhl hinten erheblich ausschwenken, würde dies zu Kollisionen führen, wird die Lenkbewegung soweit minimiert, dass Kollisionen vermieden werden.

Bei der Raumerkennung gibt es grundsätzliche Probleme mit Ultraschallsensoren: Die bekanntesten sind Reflexionen und Cross-Talk [Röfer und Lankenau, 1999]. Zu Reflexionen kommt es, wenn die Oberfläche eines Hindernisses in einem ungünstigen Winkel zum Sensor steht. In diesem Fall können Objekte übersehen werden. Beim so genannte Cross-Talk empfängt ein Sensor Signale, die nicht er selbst sondern ein anderer Sensor ausgesendet hat. Dadurch können Objekten falsche Positionen bei der Erkennung zugeordnet werden. Darüber hinaus übersehen die hier verwendeten Ultraschallsensoren kleinere Hindernisse leicht bei einer Messung.

Um diese Probleme zu verhindern, oder zu minimieren, werden sogenannte virtuelle Sensoren eingesetzt. Diese liefern Entfernungen zu Hindernissen, genau wie Ultraschallsensoren. Sie messen jedoch nicht in der Realität, sondern auf einer lokalen Hinderniskarte, die von der Software auf Grundlage der Ultraschallmessungen aufgebaut wird. Somit sind diese Sensoren nur in der Software existent und werden daher 'virtuelle Sensoren' genannt.

Die lokale Hinderniskarte ist eine Gitterkarte der Größe 4m x 4m, auf der alle Messungen der letzten 30 Sekunden gesammelt werden. Bewegt sich der Rollstuhl, so wird die Karte entsprechend verschoben. In der Karte werden zwei der virtuellen Sensoren platziert. Diese liefern die Entfernung zum nächsten Hindernis in der Richtung, in die sie gerichtet sind. Sie haben jedoch den Vorteil, alle Messungen der letzten 30 Sekunden zur Verfügung zu haben. Daher werden Objekte erkannt, die bei der letzten Ultraschallmessung übersehen wurden [Röfer und Müller, 1998].

Abbildung 4: Der Rolland II (Bild aus [Röfer, 2002])

### 1.2.2 Der Rolland II

Der Rolland II [Röfer und Lankenau, 1999] wurde 1997 gebaut (siehe Abbildung 4) und ist eine direkte Weiterentwicklung des Rolland I. Er setzt Sensoren ein, die auf der gleichen Messtechnik basieren, wie die des Rolland I. Aufgrund der Erfahrungen mit dem Rolland I wurden jedoch einige weggelassen, wie die Kontakt- und Infrarotsensoren, andere wurden in höherer Qualität und höherer Quantität verbaut. Die bestehenden Anwendungen des Rolland I wurden verbessert und ein echtes Ausweichverhalten hinzugefügt.

Für den Rolland II wurde der handelsübliche elektrische Rollstuhl Meyra Genius 1.522 zu einem Roboter umgebaut. Der Antrieb und die Lenkung werden genau wie beim Rolland I umgesetzt: Die Vorderachse dient dem Antrieb, die Hinterachse dient der Lenkung.

Als zusätzliche Hardware wurde ein Standard-PC mit einem Pentium 233 Prozessor und 64 MB RAM eingesetzt. Zur Bewegungserfassung wird die rollstuhleigene Odometrie verwendet. Wie beim Rolland I ist es für den PC, respektive die Software, möglich, den Stand der Lenkachse zu ermitteln. So kann sowohl die Geschwindigkeit als auch die Bewegungsrichtung erfasst werden. Zur Raumerfassung dienen 27 Polaroid-Ultraschallsensoren, die in einem Oval in 50 cm Höhe um den Rollstuhl herum angebracht sind.

Wie alle Ultraschallsensoren können auch die Polaroid-Sensoren Fehlmessungen liefern. Die häufigsten Gründe hierfür sind die oben schon beschriebenen Phänomene Reflexion und Cross-Talk.

Um Cross-Talk zu vermeiden, wird beim Rolland II eine Feuerstrategie eingesetzt, bei der nur 2 Sensoren gleichzeitig Messungen durchführen. Es wird

dabei beachtet, dass diese Sensoren möglichst weit auseinander liegen und in verschiedene Richtungen messen. So feuert immer ein Sensor auf der rechten und einer auf der auf der linken Seite zugleich. Eine Messung besteht somit aus mehreren Iterationen. Mit 14 Iterationen kann man eine komplette Messung, in die alle Sensoren eingebunden sind, durchführen. Eine solche Messung dauert ca. 0,5 Sekunden. Um die Erkennung der Hindernisse für die Kollisionsvermeidung zu beschleunigen, wird eine Auswahl der zu verwendenden Sensoren anhand der Bewegungsrichtung des Rollstuhls getroffen. Ebenfalls spielt bei der Auswahl eine Rolle, wie alt die letzte Messung des jeweiligen Sensors ist. Darüber hinaus wird forciert, dass in der Richtung eines neu erkannten Hindernisses sofort eine zweite Messung stattfindet, um Fehlmessungen schnell zu korrigieren [Lankenau und Röfer, 2001].

Implementiert wurden, wie beim Rolland I, die Anwendungen Raumerfassung, Kollisionsvermeidung, autonomes Fahren anhand von Karten und Landmarken, sowie die eigene Positionsbestimmung auf den Karten ( [Lankenau und Röfer, 2002] und [Lankenau u. a., 2003]) und Türdurchfahrtsverhalten (siehe [Röfer und Lankenau, 2000] und [Röfer und Lankenau, 2002]). Als zusätzliche Anwendung bietet der Rolland II ein Ausweichverhalten [Lankenau und Röfer, 2001].

Für die Kollisionsvermeidung und das Ausweichverhalten wurde die Strategie der virtuellen Sensoren erweitert. Die virtuellen Sensoren des Rolland I haben eine feste Position und liefern den Abstand zum nächsten Hindernis. Beim Rolland II wird für jede mögliche Geschwindigkeit und Lenkbewegung vorberechnet, welche Zellen der Hinderniskarte die Gefahr einer Kollision bergen. Für jede mögliche Bewegung werden nun die entsprechenden Indizes der Zellen in einer Liste gespeichert. Diese werden so geordnet, dass die nächsten Hindernisse zuerst erkannt werden. Da diese Berechnung nicht während der Laufzeit durchgeführt wird, wird im fahrenden Betrieb immer die kleinst mögliche Anzahl an Zellen durchsucht. In der vorberechneten Liste werden zu jeder Zelle auch die Entfernungen zwischen Rollstuhl und dem Hindernis gespeichert, so dass diese zur Laufzeit nur ausgelesen und nicht berechnet werden müssen. Dieses Vorgehen ermöglicht hohe Genauigkeit bei guter Performance [Lankenau und Röfer, 2001]. Diese Strategie wird, mit leichten Anpassungen, auch beim Rolland III und dem HNF-Rolland eingesetzt.

### 1.2.3 Der Rolland III

Der Rolland III [Mandel u. a., 2005] wurde 2003 gebaut (siehe Abbildung 5) und revolutioniert die Sensorstrategie des Rolland I und II. Statt Ultraschallsensoren werden nun Laserscanner eingesetzt, welche eine größere Genauigkeit

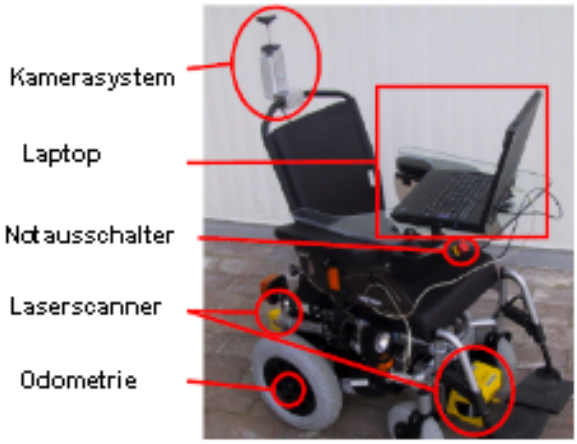



Abbildung 5: Der Rolland III (Bild aus [Mandel u. a., 2005])

der Ortsbestimmung von Hindernissen und Landmarken zulassen. Der Rollstuhl selbst ist durch eine andere Kinematik sehr viel wendiger. Die Anwendungen wurden weiter verbessert, nicht zuletzt durch die höhere Ortsauflösung und eine bessere Odometrie. Bis heute ist der Rolland III die aktuelle Testplattform des Projekts 'Bremer Autonomer Rollstuhl'. Es wurden und werden neue Anwendungen dafür entwickelt und evaluiert. So zum Beispiel die Sprachsteuerung und die Benutzung neuer Eingabemethoden zur Steuerung.

Als Plattform für den Rolland III dient der handelsübliche Rollstuhl Champ Modell 1.594 des Herstellers Meyra. Im Gegensatz zu den, für Rolland I und II verwendeten Rollstühlen, verfügt der Champ über einen Differentialantrieb: Es können die hinteren beiden Räder mit unterschiedlichen Geschwindigkeiten angetrieben werden. Aus der Differenz der Geschwindigkeiten der beiden Räder ergibt sich die Lenkbewegung. Durch diesen Antrieb ist es möglich, den Rollstuhl auf der Stelle zu drehen, wenn sich die Antriebsräder gegenläufig bewegen. Die vorderen Räder sind auf einer senkrechten Achse gelagert, wodurch Lenkbewegungen in jeder Richtung nachgegeben werden kann. Ihre Aufhängung entspricht der Aufhängung von Rädern eines Einkaufswagens. Der Hersteller bezeichnet diese vorderen Räder in seiner Bedienungsanleitung als 'Lenkräder'. In dieser Arbeit wird diese Bezeichnung übernommen. Es sollte jedoch nicht der Irrtum entstehen, dass diese Räder an der gewollten Lenkung beteiligt sind. Sie sind absolut passiv und können nur durch ihren Stand bei Fahrtrichtungsänderungen zu nicht beabsichtigten leichten Richtungsänderungen führen. Der Rolland III besitzt darüber hinaus einen Therapietisch, der nicht zur Standardausrüstung des Champs gehört, jedoch vom Hersteller als Erweiterung an-

geboten wird.

Als zusätzliche Hardware wurden an den Meyra Champ zwei Laserscanner, Siemens LS4, eine Kamera und zwei Odometriesensoren, MiniCoder GEL 248, angebaut. Außerdem wurden zwei Lautsprecher und ein Mikrophon hinzufügt. Die Software läuft, im Gegensatz zu Rolland I und II, nicht auf einem eingebauten Computer, sondern auf einem Laptop, der auf den Therapietisch gestellt wird. So ist es möglich, dass verschiedene Anwendungen von verschiedenen Entwicklern geschrieben werden, die alle ihren eigenen Laptop an den Rolland III anschließen. Außerdem ist es dadurch möglich, die Software im laufenden Betrieb des Rollands zu debuggen, da ein Monitor für den Benutzer sichtbar ist. Um nicht jedes Hardwaremodul einzeln an den Laptop anzuschließen, enthält der Rolland III zusätzlich einen USB-Hub, mit dem jedes Hardwaremodul verbunden ist. Der Benutzer muss demzufolge nur einen USB-Stecker mit seinem Laptop verbinden.

Als Anwendungen gibt es bis jetzt eine Kollisionsvermeidung, Raumerfassung und autonome Navigation auf bekannten Karten [Mandel u. a., 2006]. Als Erweiterung der autonomen Navigation gibt es eine Sprachsteuerung [Ross u. a., 2004].

Mit dieser ist es möglich, sein Fahrtziel anzugeben, zum Beispiel 'Badezimmer' und der Rollstuhl steuert selbstständig dieses Ziel an. Dies funktioniert, wenn dem Sprachbefehl ein Ziel in der bekannten Karte zugeordnet ist.

Zur Zeit wird an einer Steuerung gearbeitet, die es ermöglicht, dass der Fahrer durch das Neigen seines Kopfes den Rollstuhl lenkt [Mandel u. a., 2007]. Außerdem wird an einem reaktiven Ausweichverhalten gearbeitet.

### 1.2.4 Der HNF-Rolland

Der HNF-Rolland (siehe Abbildung 6) wurde im Rahmen dieser Diplomarbeit im Laufe des Jahres 2006 gebaut. Er stellt weitgehend einen Nachbau des Rolland III dar. Weil der Rolland III ein Prototyp für Forschungszwecke ist und der HNF-Rolland als Museumsexponat entwickelt wurde, waren leichte Änderungen erforderlich:

Als Plattform dient weiterhin der Meyra Champ Modell 1.594. Die einzigen äußeren Unterschiede sind ein neues Bedienmodul mit leicht geändertem Protokoll. Das Fehlen des Therapietisches liegt daran, dass der HNF-Rolland nicht mit einem Laptop, sondern, wie bei den Vorgängermodellen Rolland I und II, mit einem eingebauten Computer betrieben wird. Jener ist im Gegensatz zu Rolland I und II, ein eingebetteter Computer, der von Digital Logic speziell für Roboter entwickelt wurde.

Weitere äußere Änderungen dienen dem Schutz der Museumsbesucher. Um

Abbildung 6: Der HNF-Rolland

den Sicherheitsanforderungen des Museums gerecht zu werden, erhielt der HNF-Rolland Radkappen und eine durchsichtige Abdeckhaube für das Bedienmodul. Alle Verstellschrauben, die von Hand und ohne Werkzeug zu bedienen waren, wurden durch Schrauben ersetzt, die Werkzeug erfordern. Außerdem wurden die Fußstützen gepolstert, um bei einer unvermeidbaren Kollision Verletzungen möglichst gering zu halten.

Da der HNF-Rolland nur mit einem Computer betrieben wird, entfällt der USB-Hub, die gesamte Hardware ist direkt mit dem Computer verbunden.

Darüber hinaus wurde die maximale Geschwindigkeit auf 2,0 $\frac{km}{h}$ reduziert, der Rolland III kann hingegen schneller als 7,0 $\frac{km}{h}$ fahren. Auch das Beschleunigungs- und Bremsverhalten wurde deutlich weicher eingestellt als dies beim Rolland III der Fall ist. Dies war zum einen erforderlich, da die Testumgebung im Museum sehr klein ist, zum anderen, da am Rolland III die Maximalwerte für Geschwindigkeit, Bremskraft und Beschleunigung eingestellt sind. Diese Maximalwerte können einen unerfahren Fahrer, selbst in geräumigen Umgebungen, erschrecken.

Als einzige Anwendung wurde die Kollisionsvermeidung implementiert, autonomes Fahren und Ausweichverhalten sowie Sprachsteuerung oder neue Bedienmethoden werden im Museum nicht zur Schau gestellt.

Der HNF-Rolland dient dem Zweck, Museumsbesuchern zu demonstrieren, wieweit heutzutage Methoden der Robotik in der Lage sind, Menschen mit Behinderungen mehr individuelle Mobilität zu verschaffen.

# 2 Hardware

Dieser Abschnitt beschäftigt sich mit der Hardware des HNF-Rollands. Wie in der gesamten Arbeit werden dabei besonders die Unterschiede zwischen Rolland III und HNF-Rolland hervorgehoben.

Beschrieben wird der Rollstuhl, der eingesetzte Computer, sowie die Sensoren und deren Halterungen. Die Stromversorgung und die daraus resultierenden Probleme erhalten eigene Unterabschnitte.

## 2.1 Der Rollstuhl 'Meyra Champ' von Meyra

Bei dem Meyra Champ [Meyra, 2005] handelt es sich um einen elektrischen Rollstuhl, der für die Beförderung von Menschen mit Behinderungen konzipiert wurde, die einen mechanischen Rollstuhl nicht über längere Stecken aus eigener Kraft fahren können. Er ist für den Indoor- und Outdoorbereich konzipiert.

Angetrieben wird dieser Rollstuhl über einen Differentialantrieb der Hinterräder. Das heißt, beide Antriebsräder haben einen eigenen Motor, die Differenz der Bewegungen der einzelnen Räder ergibt die Lenkbewegung. Der Meyra Champ kann eine Höchstgeschwindigkeit von 6 $\frac{km}{h}$ erreichen.

Für die sehr enge Testumgebung im Museum wurde der HNF-Rolland bei ca. 2 $\frac{km}{h}$ abgeriegelt (Kurvenfahrten können Abweichungen zur Folge haben). Hingegen hat der Rolland III ein spezielles Patch der Meyra Software, so dass auch Geschwindigkeiten über 7 $\frac{km}{h}$ erreicht werden können.

Der Meyra Champ kann leichte Steigungen und Gefälle überwinden, sowie Stufen oder Bordsteine hoch und herunter fahren. Der Rolland ist wegen den Umbaumaßnahmen nur für den Indoorbereich geeignet. Das Überfahren von Steigungen, Gefällen und Stufen ist nicht, beziehungsweise nur sehr eingeschränkt, möglich. Die Begrenzung auf den Indoor-Bereich resultiert daraus, dass die Anschlüsse der Laserscanner trotz Spritzwasserschutz empfindlich auf Nässe reagieren können und die Messzahnräder bei häufigem Wasserkontakt rosten. Das Überfahren von Stufen wird eingeschränkt, da die Bodenfreiheit durch die Halterungen der Laserscanner beeinträchtigt wird und somit ein weitgehend ebener Boden unabdingbar ist.

Tabelle 1 zeigt die Auswirkungen der Umbaumaßnahmen auf die Ausmaße und das Gewicht des Rollstuhls. Die Längenangaben sind inklusive Fußstützen, die Breite hängt von den Einstellungen der Armlehnen ab.

Von den Ausmaßen wird lediglich die Länge durch die Umbaumaßnahmen beeinflusst. Der hintere Laserscanner ragt beim HNF-Rolland ca. 1-2 cm über die Grundfläche des Rollstuhls hinaus, der Laserscanner des Rolland III 5-6 cm. Des Weiteren hat der HNF-Rolland eine spezielle Polsterung der Fußstützen, die

| Model | Länge | Breite | Gewicht | max. Gewicht des Nutzers |
|---|---|---|---|---|
| Meyra Champ | 109 cm | 64-66 cm | 106 kg | 120 kg |
| Rolland III | 114-115 cm | 64-66 cm | 112 kg | 114 kg |
| HNF-Rolland | 112-113 cm | 64-66 cm | 112 kg | 114 kg |

Tabelle 1: Ausmaße und Gewicht des Meyra Champ, Rolland III und des HNF-Rollands

weitere 2 cm über die Grundfläche des Meyra Champ ragt.

Die zusätzliche Hardware, die beim Rolland in den Meyra Champ eingebaut ist, wiegt ca. 6 kg. Dieses Gewicht muss entsprechend beim Nutzer eingespart werden, daher verringert sich das maximale Nutzergewicht.

Die Stromversorgung wird über zwei in Reihe geschaltete 12-Volt-Autobatterien realisiert. Diese liefern 24 Volt bei maximal 60 Ampere. Eine Überschreitung dieser Leistung wird durch die Hauptsicherung verhindert. Die gesamte Hardware des Rollands ist ebenfalls an diese Stromversorgung angeschlossen.

### 2.1.1 Bauliche Veränderungen am Meyra Champ zum sicheren Einsatz als Museumsexponat

Um einen sicheren Einsatz als Museumsexponat zu gewährleisten wurden drei Veränderungen am Meyra Champ vorgenommen (siehe Abbildung 7).

Die Tasten des Bedienmoduls sind durch eine Plexiglasscheibe abgedeckt, um zu verhindern, dass Museumsbesucher den Rollstuhl ausschalten, die Lichtanlage betätigen oder die Fahrwerte verändern. Letzteres könnte zu ungewollten Eigenschaften der Kollisionsvermeidung führen, da die Software von den härtesten einstellbaren Fahrwerten ausgeht und nicht auf eine Änderung dieser ausgelegt ist. Es wäre möglich, dass es in diesem Fall zu Kollisionen kommt, weil sich bei weicheren Fahrparametern der Bremsweg verlängert.

An den hinteren Rädern des Meyra Champ sind Radkappen montiert, um Verletzungen von Museumsbesuchern zu verhindern. Im normalen Betrieb des Meyra Champ kommt es nicht vor, dass sich Finger oder andere Körperteile in den Speichen befinden, während der Fahrer losfährt, und auch im Museum sollte dies nicht der Fall sein. Bei interaktiven Exponaten, ist dies jedoch nicht auszuschließen, besonders nicht, da das Exponat nicht ständig unter Aufsicht steht.

An den Fußstützen des Meyra Champ ist eine spezielle Polsterung angebracht worden, um im Fall einer Kollision den Schaden so gering wie möglich zu halten. Zwar ist der Rolland darauf ausgelegt, Kollisionen zu vermeiden, und daher

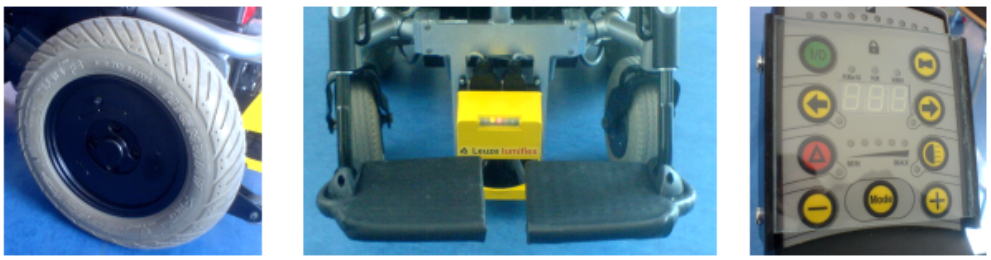


Abbildung 7: Die baulichen Veränderungen für maximale Sicherheit der Besucher
Links: Radkappen, Mitte: Fußstützen, Rechts: Abdeckscheibe

sollte dieses Problem nicht bestehen, es gibt jedoch Materialien, die von den Laserscannern nur schlecht oder gar nicht erkannt werden. Kollisionen mit solchen Objekten sind nicht auszuschließen. Außerdem kann es zu Kollisionen kommen, wenn sich Personen oder Objekte schnell in den Bremsweg des Rollands bewegen, so dass eine sichere Bremsung vor dem Hindernis physikalisch nicht mehr möglich ist.

Die Polsterung der Fußstützen besteht aus einem herkömmlichen Gartenschlauch, den es in jedem Baumarkt zu kaufen gibt. Sein Durchmesser beträgt 2 cm, seine Materialstärke 2,5 mm. Er besteht aus elastischem Kunststoff. Es ist leicht, ihn etwas einzudrücken, jedoch wird eine recht große Kraft benötigt ihn soweit einzudrücken, dass sich die gegenüberliegenden Seiten berühren. Er stellt somit einen guten Puffer dar. Auf die Außenseite der Fußstützen geklebt wurde der Schlauch mit dem Pattex Repair Extreme. Die Beschaffenheit des Klebers wird im Abschnitt 2.5.2 genau beschrieben. Für diesen Einsatz reicht es zu erwähnen, dass der Kleber sehr gute Eigenschaften besitzt, um den Kunststoff des Gartenschlauchs dauerhaft mit dem Plastik der Fußstützen zu verbinden. Aus optischen Gründen und wegen der besseren Stabilität, wurden anschließend die Fußstützen samt Gartenschlauch mit Kunstleder überzogen. Nach viereinhalb Monaten Ausstellungsbetrieb war das Kunstleder sehr abgenutzt und hatte einige Löcher. Die Fußstützen wurden daher erneut bezogen, diesmal mit einer robusten, schwarzen Klebefolie.

Durch diese Lösung sind die relativ scharfen unteren Kanten der Fußstützen durch den Gartenschlauch ummantelt und der relativ harte Kunststoff gepolstert. Dadurch besteht keine Gefahr von Schnittwunden oder Kratzern nach einer möglichen Kollision mit Personen, die von der Software nicht verhindert werden kann. Die Gefahr von leichten Prellungen oder Hämatomen wird verringert.

Abbildung 8: Das Bedienmodul des Meyra Champ

### 2.1.2 Bedienmodul des Meyra Champ

Das Bedienmodul enthält den Joystick zur Steuerung des Rollstuhls sowie Tasten, um die Lichtanlage, Blinker und die Hupe zu betätigen, sowie die Fahrwerte einzustellen. Diese ermöglichen es auch, die Konfiguration des Rollstuhls zu verändern. Da es im Museum nicht erwünscht ist, dass Lichtanlage oder Hupe von Besuchern betätigt werden können und es gefährlich wäre, wenn Besucher die Fahrwerte oder die Konfiguration veränderten, sind die Tasten mit einer Plexiglasscheibe abgedeckt. Die Museumsbesucher können nur den Joystick betätigen.

Das Bedienmodul hat außerdem einen CAN-Bus, über den externe Joysticks angeschlossen werden können. Meyra bietet beispielsweise einen Joystick, der sich mit der Zunge betätigen lässt und einen, der von einer Begleitperson bedient werden kann.

Dieser CAN-Bus ist, im Fall des Rollands, die Schnittstelle, über die der Computer den Rollstuhl ansteuert. Der Computer ist, im Verständnis der Meyra Philosophie, ein externer Joystick. Die Prioritäten müssen dementsprechend so eingestellt werden, dass der externe Joystick eine höhere Priorität erhält als der interne Joystick an dem Bedienmodul. Die Prioritäten sind, im Fall des Rolland, besonders wichtig, da der Benutzer ihn über den internen Joystick bedient. Die gewünschte Fahrtrichtung und Geschwindigkeit wird jedoch von dem Computer auf Kollisionsfreiheit geprüft. Der Computer liest dafür die Werte des internen Joysticks aus und modifiziert diese, bevor die Fahrbefehle an den Rollstuhl zurückgegeben werden. Für den Rollstuhl sind also zwei Joysticks im Einsatz, es werden jedoch nur die Fahrbefehle des Joysticks ausgeführt, der die höhere

| Pin Nr. | Funktion |
|---|---|
| 1 | + 5V, maximal 20mA |
| 2 | ext. Lenksignal (analog Eingang) 2,5 V +- 2V |
| 3 | ext. Fahrsignal (analog Eingang) 2,5 V +- 2V |
| 4 | Gnd, Signal-Masse |
| 5 | Ein/Aus-Taster, gegen Gnd geschaltet |
| 6 | Totmann-Schalter, gegen Gnd geschaltet |
| 7 | TxD, Ausgang für serielle Daten, TTL-Pegel |
| 8 | RxD, Eingang für serielle Daten, TTL-Pegel |

Tabelle 2: Belegung des CAN-Bus beim Meyra-Champ

Priorität hat. Sind die Prioritäten falsch eingestellt, würde der Computer zwar weiterhin die Fahrwerte modifizieren, der Rollstuhl würde jedoch die unveränderten Werte des internen Joysticks ausführen. Dadurch wäre die Kontrolle des Computers praktisch ausgeschaltet.

### 2.1.3 Der CAN-Bus

Der CAN-Bus bietet acht Pins, über die alle Betriebsfunktionen des Rollstuhls betätigt werden können (siehe Tabelle 2). Die Konfiguration der Prioritäten kann nur über die Tasten des Bedienmoduls geändert werden.

Die Pins 2 und 3 werden von dem Computer nicht verwendet. Die Ansteuerung erfolgt über die serielle Verbindung der Pins 7 und 8. Diese Pins in Verbindung mit Pin 1 und 4, für Phase und Masse, reichten beim Rolland III für die Steuerung aus. Für den HNF-Rolland wird noch der Pin 5 zum Ein- und Ausschalten benötigt. Pin 6 wird erst benötigt seit der Entscheidung, dem internen Joystick keine Priorität zu geben. Näheres dazu wird im Abschnitt 2.1.4 beschrieben.

### 2.1.4 Der Totmann-Schalter

Der Totmann-Schalter ist eine Sicherung, die verhindert, dass der Rollstuhl sich in ungewollten Situationen bewegt. Er ist rein elektrotechnisch umgesetzt und dient der Absicherung gegen mögliche Softwarefehler. Sendet die Software Fahrbefehle, ohne dass der Totmann-Schalter die Motoren freigeschaltet hat, so werden diese nicht umgesetzt. Der Totmann-Schalter sichert sowohl die Software des Bedienmoduls als auch die des externen Joysticks. Im Falle des Rollands ist der externe Joystick der Computer.

Der Meyra Champ hat eine interne Bremse, die manuell mit einem Hebel bedient werden kann. Dieser Hebel sollte, nach Herstellerangaben, nur im ausgeschalteten Zustand betätigt werden und stellt den Rollstuhl von Motor- auf Schiebebetrieb.

Die gleiche interne Bremse kann auch vom Totmann-Schalter gelöst werden und ermöglicht, dass die Fahrbefehle von den Motoren ausgeführt werden. Der Totmann-Schalter wird vom internen Joystick geschaltet, wenn sich dieser aus der Null-Postion bewegt. Dies erfolgt jedoch nur, wenn dem internen Joystick auch eine Priorität zugewiesen wurde. Alternativ kann der Totmann-Schalter geschaltet werden, wenn der Pin 6 des CAN-Bus gegen den Pin 4, die Masse, geschaltet wird.

Beim Rolland III wird der Totmann-Schalter benutzt, indem dem internen Joystick eine Priorität zugewiesen wird, die niedriger als die des externen Joysticks ist. Für den Einsatz im Museum wird nur dem externen Joystick eine Priorität zugewiesen. Dies stellt sicher, dass der Rollstuhl nur mit aktiver Kollisionsvermeidung fährt. Hat der interne Joystick eine Priorität, so lässt sich der Rollstuhl, bei ausgeschaltetem Computer, normal fahren. Diese Lösung erfordert, dass der Totmann-Schalter vom externen Joystick, im Fall des HNF-Rolland vom Watchdog, geschaltet wird. Genaueres dazu folgt im Abschnitt 2.2.

## 2.2 Der Watchdog / das Steuerungsmodul

Das Steuerungsmodul zwischen Computer und Bedienmodul des Rollstuhls hat beim HNF-Rolland zwei Aufgaben. Erstens müssen die Pegel der Signalleitungen umgewadelt werden, um die Kommunikation zwischen Computer und Bedienmodul zu ermöglichen, zweitens hat das Steuerungsmodul die Funktion eines Watchdogs, der mögliche Softwarefehler abfangen kann. Zum Schutz vor Softwarefehlern kann der Watchdog den Totmann-Schalter in sicheren Situationen freischalten, beziehungsweise dies in unsicheren Situationen unterlassen. Des Weiteren kann der Watchdog den Rollstuhl bei Bedarf ein- und bei schweren Fehlern ausschalten.

Da die serielle Schnittstelle des CAN-Bus TTL-Pegel verwendet, also +5V als high Pegel und 0V als low Pegel, der Computer jedoch +12V und -12V verwendet, wird ein Pegelwandler benötigt. Beim Rolland III wurde dafür der Chip MAX232 verwendet. Dieser reicht aus, um die serielle Kommunikation zu realisieren.

Beim HNF-Rolland kommt jedoch zusätzlich ein Watchdog zum Einsatz, der die serielle Kommunikation überwacht. Bei Unterbrechungen der Kommunikation soll der Watchdog den Rollstuhl automatisch ausschalten, dadurch werden unsichere Zustände vermieden. Außerdem soll der Watchdog den Totmann-

Schalter betätigen, um die Ausführung von Fahrbefehlen durch den Motor zu ermöglichen. Der Totmann-Schalter soll nur freigeschaltet werden, sobald sich der interne Joystick aus der Null-Position bewegt.

Dies hat den Effekt, dass der Rollstuhl keine Fahrbefehle ausführt, wenn der interne Joystick nicht bewegt wird. Der Benutzer hat so die Möglichkeit, auch bei auftretenden Softwarefehlern den Rollstuhl zu bremsen, indem er den Joystick loslässt. Dieses Verhalten ist wichtig, falls die Software falsche Fahrbefehle sendet. Da der Totmann-Schalter rein elektrotechnisch umgesetzt ist und die GT-Rolland-Software keinen Einfluss auf dessen Schaltung hat, können dadurch Softwarefehler sicher abgefangen werden.

Zur Realisierung dieses Watchdogs wurde Christoph Budelmann, elektrotechnischer Praktikant beim DFKI, beauftragt, eine entsprechende elektrische Schaltung zu entwerfen. Die Schaltung beinhaltet einen Mikrocontroller. Dieser überwacht die gesamte Kommunikation zwischen Computer und Rollstuhl, setzt die Pegel der Signale entsprechend um, liest die Joystickdaten des internen Joysticks aus, betätigt den Totmann-Schalter und schaltet den Rollstuhl ein und aus. Entsprechend der Beschreibung wird der Totmann-Schalter nur betätigt, wenn sich der interne Joystick aus der Null-Position bewegt. Beim Abreißen der Kommunikation wird der Rollstuhl ausgeschaltet. Dies geschieht auch beim Betätigen des Hauptschalters, in diesem Fall wird der Computer ausgeschaltet, die Kommunikation reißt ab und der Watchdog schaltet das Bedienmodul aus. Zum Einschalten des Rollstuhls erwartet der Watchdog eine Initialisierungssequenz über die serielle Verbindung zum Computer. Nach dem Erhalt dieser Sequenz schaltet der Watchdog den Rollstuhl ein und beginnt die Überwachung der Kommunikation und der Joystickstellung. Ein Schaltplan des Watchdogs befindet sich im Anhang A.1.

## 2.3 Der Computer 'MSM855' von Digital-Logic

Der MSM855 [Digital-Logic, 2000] (siehe Abbildung 9) bietet alle Schnittstellen eines Standard-PCs. Er basiert auf Laptop-Technologie und ist darauf ausgelegt, als eingebetteter Computer eingesetzt zu werden. Daher ist er sehr klein und platzsparend. Der MSM855 ist in Etagen aufgebaut, die je ca. 10 x12 cm groß sind. Eine Etage enthält das Netzteil, eine weitere das Mainboard. Es ist möglich, weitere Etagen für zusätzliche Funktionen zu installieren, diese werden über eine PC/104 Schnittstelle mit dem Mainboard verbunden.

Standardmäßig hat der MSM855 vier USB2.0-Anschlüsse, zwei RS323 serielle Schnittstellen, zwei PS2 Anschlüsse für Maus und Tastatur, einen Anschluss für die Netzwerkkarte, einen VGA-Anschluss, einen IDE-Anschluss, einen Anschluss für die Soundkarte und eine PC/104 Schnittstelle.

Abbildung 9: Der MSM855 von Digital Logic (Bild aus [Digital-Logic, 2000])

Zunächst wurde geplant, für den HNF-Rolland eine zusätzliche Etage, das MSMX104, mit 4 weiteren seriellen Schnittstellen, über die PC/104 Schnittstelle anzuschließen. Diese zusätzlichen seriellen Schnittstellen hätten optional mit dem RS323 oder dem RS422 Standard betrieben werden können. Verschiedene Gründe führten jedoch dazu, dies nicht zu tun: So wurde zunächst geplant, alle Sensoren, die Laserscanner und die Odometrie sowie das Steuerungsmodul für den Meyra Champ, über serielle Schnittstellen anzuschließen. Nach einigen Änderungen im Schaltplandesign der Odometrie und des Champ-Steuerungsmoduls, siehe Abschnitte 2.5 und 2.2, war es vorteilhaft, diese über USB- und nicht über serielle Schnittstellen anzuschließen. Daraufhin wurden nur noch zwei serielle Schnittstellen für die beiden Laserscanner benötigt. Da diese auf dem Mainboard verfügbar waren, war das MSMX104 nicht mehr unbedingt erforderlich. Der einzige Vorteil wäre der schnellere RS422 Standard gewesen, der nur auf dem MSMX104, nicht jedoch auf dem Mainboard verfügbar ist.

Des Weiteren wäre der MSM855 mit dem MSMX104 zu hoch für den Batteriekasten des Meyra Champ geworden. Dies hätte erfordert, einen neuen, größeren Deckel für den Batteriekasten zu bauen.

### 2.3.1 Befestigung des MSM855

Die Etagen des MSM855 werden durch Abstandshalter mit M4 Gewinden verbunden. Die untere Etage steht auf diesen Abstandhaltern. Zur Befestigung im Batteriekasten des Rollstuhls wird unter die unteren Abstandhalter, die dadurch die Stützen des MSM855 sind, eine Kunststoffplatte befestigt. Dazu werden, in

den Abständen der Stützen, Löcher in die Kunststoffplatte gebohrt. Die Gewinde der Stützen werden durch die Löcher gesteckt und von unten mit M4 Muttern befestigt.

Im Batteriekasten des Meyra Champ befindet sich eine klappbare Stahlplatte, die die Elektronik von den Batterien trennt. Im vorderen Teil dieser Stahlplatte ist Platz für eine elektrische Höhenverstellung des Sitzes. Diese elektrische Höhenverstellung ist jedoch weder beim HNF-Rolland noch beim Rolland III vorhanden, da sie keinen Platz für den vorderen Laserscanner lassen würde. Auf der Stahlplatte des Batteriekastens sind für die Befestigung der Elektronik der Höhenverstellung Bohrungen vorhanden. Diese werden beim HNF-Rolland genutzt, um die Kunststoffplatte, auf der das MSM855 befestigt ist, im Batteriekasten zu befestigen. Dazu werden weitere Löcher in die Kunststoffplatte gebohrt, gemäß den Abständen der Bohrungen im Batteriekasten. Mit M5 Schrauben wird nun die Kunststoffplatte auf die Stahlplatte im Batteriekasten geschraubt.

### 2.3.2 Belegung der Schnittstellen des MSM855 im Auslieferungszustand

Im Auslieferungszustand des HNF-Rolland ist die Belegung der Schnittstellen des MSM855 wie folgt: Über die seriellen Schnittstellen sind die Laserscanner mit dem Computer verbunden. Die Odometrie und das Champ-Steuerungsmodul sind über die USB-Schnittstellen verbunden. Ein weiterer USB-Steckplatz ist mit einem 2 Gigabyte USB-Stick, der als Festplatte für das Betriebssystem fungiert, belegt. Die Sound- und die Netzwerkkarte sind nicht installiert. Ebenso ist die IDE-Schnittstelle nicht belegt. Die PS2-Stecker, die auf eine Pin-Reihe des Mainboards gesteckt werden müssen, sind ebenfalls nicht im Auslieferungszustand enthalten. Sollten diese bei Wartungsarbeiten benötigt werden, kann man sie aufstecken oder einen USB-PS2-Adapter auf den freien USB-Steckplatz stecken.

## 2.4 Die Laserscanner 'rotoScan ROD-4' von Leuze electronic

Der beim HNF-Rolland eingesetzte Laserscanner rotoScan ROD-4 [Leuze_electronic, 2001] ist baugleich mit dem Laserscanner Siemens LS4 [Siemens, 2003] des Rolland III. Ausmaße und Protokoll sind unverändert.

Als typische Anwendungen gibt der Hersteller Überstandskontrolle auf Transportfahrzeugen, sowie Hinderniserkennung im Fahrbereich vollautomatischer Fahrzeuge, meist Schienenfahrzeuge, an. Weiter werden Konturvermessung, Personenerkennung und -zählung, Raumsicherung, Fassadenüberwachung, Zutrittskontrolle und Volumenmessung angegeben.

Beim Rolland dienen diese Laserscanner der Raumerkennung und der Kollisionsvermeidung. Mit den von ihnen erhaltenen Daten wird eine lokale Hindernisskarte aufgebaut um vor erkannten Hindernissen anzuhalten. Beim Rolland III dienen sie auch der Raumerkennung und der Selbstlokalisation auf Karten [Krieg-Brückner u. a., 2005].

### 2.4.1 Technische Beschreibung der Laserscanner

Der rotoScan ROD-4 ist ein optischer, zweidimensional messender Distanzsensor. Sein Laserstrahl wird über eine Ablenkeinheit periodisch in den verschieden Winkeln des Messbereichs ausgesendet, welcher 190° umfasst. Er sendet in Schritten von 0,36° Lichtimpulse aus. Deren Reflexion an Hindernissen, wie Personen oder Gegenständen wird von dem rotoScan ROD-4 empfangen und ausgewertet. Aus der Lichtlaufzeit und dem aktuellen Winkel der Ablenkeinheit werden die genauen Koordinaten des Hindernisses berechnet. Die maximale Weite einer Messung beträgt 40 m. Objekte der Größe 15 cm x 15 cm werden jedoch nur bis zu einer Entfernung von 15 m sicher erkannt. Probleme bereiten durchsichtige oder stark reflektierende Objekte wie Glasscheiben oder spiegelnde Flächen, sowie Objekte, die Licht stark absorbieren wie matte schwarze Stoffe ohne Konturen. Letzteres tritt jedoch extrem selten auf. Diese Objekte können unter Umständen nicht, oder zumindest nicht bei jedem Scan, erkannt werden.

Für eine Messung im gesamten Messbereich benötigt der rotoScan ROD-4 40 ms. Er kann 25 Messungen in einer Sekunde realisieren und sendet bei jeder Messung 528 Laserstrahlen aus.

Beide Laserscanner werden über serielle RS323 Schnittstellen mit dem Computer verbunden. Sie senden mit einer Baudrate von 109 kBaud. Bei dieser Baudrate werden für die Kommunikation fünf Laserstrahlen zusammen gefasst. Durch eine Änderung der Baudrate wird auch diese Auflösung verändert.

Nach Herstellerangaben ist es nicht sicher, die Laserscanner mit 109 kBaud über eine RS323 Schnittstelle zu betreiben, hierfür wird die RS422 Schnittstelle empfohlen. Bei der Nutzung des Rolland III sowie des HNF-Rolland stellt sich der Betrieb mit einer Übertragungsrate von 109 kBaud über die RS323 Schnittstelle als praktikabel heraus. Dabei gehen keine Scans verloren.

### 2.4.2 Halterungen der Laserscanner

Für die Befestigung der Laserscanner werden Halterungen angefertigt, die am Rahmen des Meyra Champ mit Aluminiummanschetten angebracht werden. Beim Rolland III sind diese vorne wie hinten identisch. An den Manschetten


Abbildung 10: Die Halterungen der Laserscanner, Links: Vorne, Rechts: Hinten

werden die Halterungen, ebenfalls aus Aluminium, senkrecht an der äußeren Kante befestigt. Die vordere Halterung wird etwas tiefer angebracht als die hintere, um unter den Fußstützen hindurch zu scannen. Diese Anordnung kann mit identischen Halterungen realisiert werden, da der Rahmen des Meyra Champ hinten höher ist als vorne. Beim hinteren Laserscanner ragt, durch die Befestigung an der äußeren Kante der Manschette, der Laserscanner ca. 4-6 cm über den Rahmen des Meyra Champ hinaus.

Für den HNF-Rolland wurde das Design der vorderen Halterungen ohne Veränderungen übernommen. Beim hinteren Laserscanner wurde jedoch versucht, das Überstehen über den Grundriss des Meyra Champ zu minimieren. Dafür wurde eine neue Halterung entworfen, die nicht aus Aluminium, sondern aus Stahl besteht. Dies hat den Vorteil, dass bei fast unveränderter Stabilität, die Halterung deutlich schmaler sein kann. Die Halterung wird auch nicht mehr an der äußeren Kante der Manschette montiert sondern an der unteren, wodurch die Halterung so nah wie möglich an den Batteriekasten heranrückt (siehe Abbildung 10).

Infolgedessen ragt der Laserscanner nur noch ca. 1-2 cm über den Grundriss des Meyra Champ.

Die Zeichnungen der Halterungen befinden sich im Anhang A.2.

## 2.5 Die Odometriesensoren 'MiniCoder GEL 248' von Lenord + Bauer

Um die Bewegungen des HNF-Rollands zu erfassen, werden Zahnräder auf den Achsen der Antriebsräder angebracht. Ein Sensor wird so befestigt, dass er, bei Bewegungen des Rollstuhls, die vorbeilaufenden Zähne dieser Zahnräder zählen

kann. Dadurch werden die Bewegungen der Antriebsräder erkannt und ebenso die daraus resultierenden Bewegungen des Rollstuhls.

Dazu werden Magnetsensoren von Lenord + Bauer, 'MiniCoder GEL 248' [Lenord+Bauer, 2002] verwendet. Diese sind nach Herstellerangaben darauf ausgelegt, für Drehzahl- und Positionsmessungen an Getrieben, Motoren, Förderanlagen und Maschinen verwendet zu werden.

Beim Rolland werden für diesen Einsatz auf der hinteren Achse des Meyra Champ auf jeder Seite ein Stirnzahnrad aus Stahl ohne Narbe mit 124 Zähnen und 10 mm Zahnbreite montiert. In einem Abstand von 0,1-0,5 mm werden die MiniCoder angebracht. Da der Meyra Champ über einen Differentialantrieb die Lenkung realisiert, lässt sich mit der Bewegungserfassung beider Antriebsräder die Geschwindigkeit und die Drehung des HNF-Rollands ermitteln. Die Bewegungen des Rollands lassen sich so im Millimeterbereich erkennen.

Durch eine Stellung der vorderen Lenkräder die nicht der Fahrtrichtung entspricht sowie durch rutschige Bodenbeläge, können sich Fehlmessungen ergeben.

### 2.5.1 Technische Beschreibung der Odometriesensoren und deren Signalaufnahme

Für die Ausgangssignale haben die MiniCoder jeweils zwei Signalleitungen, eine zur Geschwindigkeits- und eine zur Richtungserkennung. Die MiniCoder werden über Zahnrädern, Schlitzscheiben oder Zahnstangen montiert. Diese verändern bei Bewegungen das Magnetfeld der Sensoren [Lenord+Bauer, 2002].

Die Elektronik zur Erfassung der Sensorsignale bestand beim Rolland III zunächst aus einer seriellen Maus. Bei dieser wurden die Sensoren zur Erfassung der Mausbewegungen herausgelötet und an dieser Stelle die Signalleitungen der MiniCoder befestigt. Diese Konstruktion erwies sich als nicht langlebig. So war die Mausplatine des Rolland III nach ca. 3 Jahren im Einsatz durchgebrannt und einige Leiterbahnen waren schwarz. Auch im normalen Betrieb ergab sich ein entscheidender Nachteil: Wurden keine Bewegungen erkannt, so wurden, genau wie bei einer bewegungslosen seriellen Maus, auch keine Signale gesendet. Dies ist das gleiche Signal wie beim Ausfall der Sensoren. Dadurch war für die Software nicht zu erkennen, ob der Rolland still steht oder die Odometrie ausgefallen war.

Um dieses Problem zu lösen, wurde Christoph Budelmann, elektrotechnischer Praktikant beim DFKI, damit beauftragt, eine Platine zu entwerfen, die die Signale der MiniCoder aufnimmt und als Integerwerte über eine serielle Schnittstelle sendet. Dabei werden auch 0-Werte gesendet, wenn keine Bewegungen erkannt werden. Diese Konstruktion erwies sich, aus den genannten Gründen,

als besser im Gegensatz zu der Mausplatine. Aus diesem Anlass wurde eine solche neuentwickelte Elektronik auch für den HNF-Rolland angefertigt und die modifizierte serielle Maus nicht verwendet.

### 2.5.2 Unterschiede zwischen Rolland III und HNF-Rolland

Zur Befestigung des Zahnrades auf der hinteren Achse beim Rolland III wurde jenes erhitzt und anschließend auf die Achse geschrumpft. Diese Prozedur dauerte relativ lang und erforderte die entsprechenden Werkzeuge. Auch gab es für die Ausrichtung des Zahrades nicht sehr viel Zeit.

Beim HNF-Rolland wird das Zahnrad mit einem Spezialkleber aufgeklebt. Dazu wird ein spezieller Kleber, Pattex Repair Extreme Power-Kleber, verwendet. Dieser hält nach Herstellerangaben Dehnung, Vibration, Wasser und Temperaturen zwischen -50°C und +120°C stand. Außerdem ermöglicht dieser Kleber fünf Minuten lang Korrekturen der Position. Sollte diese Zeit nicht ausreichen, ist es möglich, den aufgetragenen Kleber zu entfernen und die Ausrichtung von neuem zu beginnen, da erst nach zwei Stunden eine gute Festigkeit und erst nach 48 Stunden die endgültige Festigkeit erreicht wird.

Durch diese Klebeeigenschaften ist es möglich, eine sehr exakte Ausrichtung ohne weitere Werkzeuge zu ermöglichen. Dazu lässt man die Achse drehen und beobachtet das Zahnrad aus verschiedenen Perspektiven. Dadurch werden mit bloßem Auge Ungenauigkeiten im und unterhalb des Millimeterbereichs sichtbar.

Um die MiniCoder über den Zahnrädern zu befestigen, wurden beim Rolland III Löcher in den Rahmen des Meyra Champ gebohrt. Mit vielen Unterlegscheiben und einer Schraubenmutter werden die MiniCoder in der richtigen Position befestigt. Exaktes Arbeiten ist hierbei schwierig, da der Rahmen aus einem sehr harten Stahl gefertigt ist und durch die Größe des Meyra Champ ein Einspannen in die vorhandenen Präzisionsbohrer fast unmöglich ist. Durch die Vielzahl an Unterlegscheiben ist eine ungewollte Änderung der Positionen der MiniCoder im laufenden Betrieb nicht auszuschließen. So kommt es vor, dass durch Vibrationen die MiniCoder das Zahnrad berührten. Dabei kann es zu falschen Messungen oder sogar zu Beschädigungen der Sensoren und der Zahnräder kommen.

Beim HNF-Rolland werden Aluminiumplatten mit M5 Schraubengewinden gefertigt. Bei diesen kleinen Aluminiumplatten ist es möglich, mit einem Präzisionsbohrer die Gewinde exakt in den Abständen des MiniCoders zu fertigen.

Diese Aluminiumplatten werden mit dem Pattex Repair Extreme Power-Kleber auf den Rahmen des Meyra Champ geklebt. Dabei ist eine exakte Ausrichtung, dank der Eigenschaften des Klebers, leicht zu realisieren. Um die ge-

naue horizontale Position zu erreichen, werden maximal 2 Unterlegscheiben benötigt. Dies schließt eine Veränderung der Position und damit ein Schleifen auf den Zahnrädern, aus.

Beim HNF-Rolland haben die MiniCoder, im Gegensatz zum Rolland III, einen offenen Kollektor, besser bekannt unter dem englischen Ausdruck 'Open Collector'. Dies ändert die Schaltstruktur des Elektronikmoduls leicht. So werden keine Dioden an den Sensorausgängen der MiniCoder benötigt, weil sich die Signalausgänge eines offenen Kollektors problemlos parallel schalten lassen.

Die MiniCoder des HNF-Rollands werden bewusst mit offenem Kollektor bestellt, da sich hierdurch die Schaltung vereinfacht. Dies hat jedoch den Nachteil, dass für den Rolland III und den HNF-Rolland nicht dieselbe Schaltung verwendet werden kann. Im Nachhinein wäre es von Vorteil gewesen, die Schaltung genau wie beim Rolland III zu gestalten, da jeder Unterschied eine mögliche Fehlerquelle darstellt.

Der Schaltplan der neuen Elektronik zur Aufnahme der Sensorsignale und deren Umsetzung befindet sich im Anhang A.3.

## 2.6 Stromversorgung und Inbetriebnahme der Hardware

Alle Strom benötigenden Module sind, soweit möglich, innerhalb des Batteriekastens befestigt. Ausnahmen stellen die Laserscanner dar. Die Odometriesensoren sind, wie in Abschnitt 2.5 beschrieben, hinter den Rädern angebracht. Sie erhalten ihre Stromversorgung durch die Schaltung zur Erfassung der Sensorsignale. Dieses Modul befindet sich innerhalb des Batteriekastens. Abbildung 11 zeigt die Anordnung der Module innerhalb des Batteriekastens. Alle Kabel, die außerhalb verlaufen, sind mit Kabelbindern eng und soweit wie möglich auf der Innenseite des Rahmens des Meyra Champ verlegt. Dies ist besonders wichtig, da bei der hohen Benutzerzahl im Museum davon ausgegangen werden kann, dass sich einige Besucher an Stellen des Rollstuhls an denen dies möglich ist, mit ihren Kleidern verfangen. Da ein solcher Fall nach vier Monaten des Ausstellungsbetriebes nicht aufgetreten ist, kann davon ausgegangen werden, dass es nicht möglich ist, sich an den Kabeln des HNF-Rollands zu verfangen.

Die Stromversorgung des HNF-Rolland basiert auf einer Parallelschaltung aller Module, die Strom benötigen. Dazu werden die Phase und die Masse der Batterien zu einem Stromverteiler aus Lüsterklemmen geleitet. Alle so entstandenen Paare aus Masse und Phase sind über die Hauptsicherung des Meyra Champ abgesichert. Auf dem Weg zwischen Batterien und Stromverteiler kann die Phase durch einen Schalter unterbrochen werden. Dieser ist der Hauptschalter des HNF-Rolland. Alle Module, die Laserscanner, die Odometrie und der Computer werden über diesen Schalter gleichzeitig an- und ausgeschaltet. Der Rollstuhl

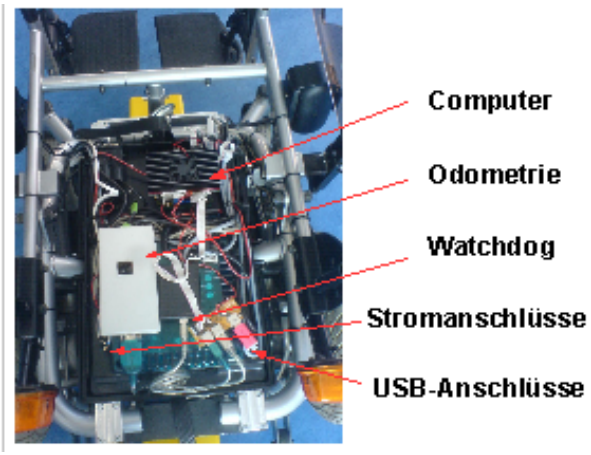



Abbildung 11: Die Raumaufteilung innerhalb des Batteriekastens

selbst wird über das Steuerungsmodul zwischen Computer und Rollstuhlbedienmodul softwaregesteuert ein- und ausgeschaltet (siehe Abschnitt 2.2).

Nach dem Betätigen des Hauptschalters ist die Odometrie sofort einsatzbereit. Die Laserscanner führen zunächst einen Selbsttest durch. Zeitgleich bootet der Computer: Er fährt das Betriebssystem hoch und startet anschließend die GTRolland-Software. Diese startet nun den Rollstuhl, der seinerseits zunächst einen Selbsttest durchführt.

Hat die Software eine Verbindung zu allen Hardwaremodulen aufgebaut und melden diese keine Fehler, lässt sie die Hupe des Meyra Champ für eine Zehntelsekunde ertönen. Dies ist das Zeichen für einen erfolgreichen Start.

Sollten beim Prozess der Inbetriebnahme des HNF-Rollands Fehler aufgetreten sein, werden diese durch Fehlercodes signalisiert, um mögliche Fehlerquellen aufzuzeigen. Bei Problemen mit den Laserscannern blinken die Blinklichter des Champs, links für den vorderen, rechts für den hinteren Scanner. Bei Problemen mit der Odometrie blinkt die Lichtanlage. Nach dem Auftreten eines Fehlers werden alle Fahrbefehle aus Sicherheitsgründen unterbunden.

Diese Fehlercodes wurden definiert, um Museumsmitarbeitern ein Beheben von einfachen Fehlern ohne Rücksprache mit DFKI-Mitarbeitern zu ermöglichen, beziehungsweise um DFKI-Mitarbeitern eine grobe Fehleranalyse zu ermöglichen, ohne vor Ort zu sein. Die Bedienungsanleitung, die den Mitarbeitern der Ausstellung 'Computer.Medizin' mitgeliefert wurde, findet sich im Anhang A.4.

## 2.7 Masseschleife als besondere Schwierigkeit

Die Tatsache, dass, resultierend aus der Parallelschaltung aller Module, jedes Modul, das an die Stromversorgung angeschlossen ist, dieselbe Masse besitzt, birgt Risiken und erschwert zum Teil die Fehleranalyse. Dazu kommt, dass jedes Modul über Signalleitungen mit dem Computer verbunden ist. Jede dieser Signalleitungen besitzt ebenfalls eine Masse, die jeweils nur eine parallelgeschaltete Masse der Batterie ist. Besonders die Signalleitungen sind nur auf sehr geringe Stromstärken ausgelegt. Somit gibt es in den meisten Modulen eine Querverbindung zwischen der Masse der Signalleitungen und der Masse der Stromversorung, die logisch gesehen alle die Masse der Batterie darstellen. Es bleibt zu erwähnen, dass wenn jedes Modul richtig angeschlossen ist, keine Probleme entstehen.

Es trat jedoch der Fall auf, dass während der Aufbauphase des Rollstuhls eine Lötstelle brach. Über diese Lötstelle war der Computer an die Masse der Rollstuhlbatterien angeschlossen. Dieser Fehler wurde zunächst nicht erkannt, da der Computer einwandfrei funktionierte. Es sollte vermutet werden, dass, ohne Anschluss an die Masse, der Stromkreis des Computers unterbrochen ist, und er sich somit nicht mehr betreiben ließe. Durch die Masseschleife des Systems war es dem Computer jedoch möglich, die benötigte Masse durch den USB-Port und das Steuerungsmodul zwischen Computer und Bedienmodul zu beziehen. So lief der Strom nicht wie vorgesehen über die Phase des Netzteils in den Computer und über die Masse des Netzteils wieder hinaus, sondern über die Phase des Netzteils herein, jedoch dann über den USB-Port direkt in die Masse des Bedienmoduls.

Dieser Fehler wurde erst erkannt, als festgestellt wurde, dass das Steuerungsmodul und der Watchdog nicht mehr funktionierten. Auch dann benötigte es einige Zeit, bis der Fehler genau analysiert war. Erst als der USB-Stecker des Steuerungsmoduls im laufenden Betrieb herausgezogen wurde und daraufhin der Computer ohne Vorwarnung ausging, konnte das Problem erkannt werden. Das Steuerungsmodul und der Watchdog wurden dabei beschädigt. Die Masse des USB-Ports ist zwar auf Stromversorgung ausgelegt, nicht jedoch die Masse einer seriellen Verbindung. Diese ist nur dazu gedacht, den Stromkreis der Signalleitungen zu schließen. Da im Steuerungsmodul ein USB-Seriell-Konverter eingebaut ist, lief zuviel Strom über die Masse der seriellen Verbindung. Das Modul musste daraufhin ausgetauscht werden.

Ein anderes Problem, welches sich ebenfalls auf die Masseschleife zurück-

führen lässt, bestand beim Odometriemodul. Durch einen Fehler im Schaltplan wurde die Masse der seriellen Verbindung des Computers mit der Phase der Stromversorgung des Odometriemoduls verbunden. Dieses Modul wurde lange an Labornetzteilen getestet, bevor es in den HNF-Rolland eingebaut wurde. Da die Masse des Labornetzteils galvanisch von der Masse des Computers, und damit von der Masse des seriellen Ports, isoliert ist, und es somit im Test zwei verschiedene Massen für Computer und Stromversorgung des Odometriemoduls gab, verursachte dieser Fehler im Test keine Probleme.

Im Rolland verursachte dieser Fehler jedoch einen Kurzschluss, der dazu führte, dass die Spannungswandler nach wenigen Sekunden sehr stark überhitzten.

Auch dieser Fehler war schwer zu finden, da die Spannung vor und hinter den Spannungswandlern exakt richtig war. Als zur genaueren Analyse das Modul wieder an Labornetzteile angeschlossen wurde, trat dieser Fehler natürlich nicht mehr auf. Erkannt wurde er erst, als jede Lötstelle des Moduls einzeln überprüft wurde.

Auch das Odometriemodul musste hiernach ausgetauscht werden, weil durch den Kurzschluss Teile der Elektronik beschädigt worden waren.

Die Laserscanner und der Meyra Champ, samt Bedienmodul, blieben zum Glück von Schäden, die aus Masseschleifen resultierten, verschont. Das ist besonders wichtig, da diese Elemente die teuerste Hardware des HNF-Rollands darstellen. Es ist jedoch auch denkbar, dass diese Module in ähnliche Probleme verwickelt werden. Festzuhalten bleibt, dass bei allen Problemen, bei denen die Stromversorgung mit im Spiel ist, die Masseschleife des Systems bedacht werden sollte. Dies gilt für jedes Rollandmodell.

# 3 Software

In diesem Abschnitt wird die gesamte Software, die auf dem HNF-Rolland zum Einsatz kommt, beschrieben. Dabei wird nicht nur auf die Software GTRolland und das eingesetzte Betriebssystem eingegangen, sondern auch auf die Entstehung der GTRolland-Software, sowie die notwendigen betriebssystemabhängigen Softwareanpassungen.

Die eingesetzte Kollisionsvermeidung wird in einem eigenen Unterabschnitt behandelt.

## 3.1 Das Betriebssystem

### 3.1.1 Windows, QNX oder Linux

Beim Rolland III läuft die GTRolland-Software unter Microsoft Windows XP. Dies birgt einige Risiken. So ist Windows kein RealTime-Betriebssystem und sichert nicht zu, zeitkritische Berechnungen zur gewünschten Zeit zu erledigen. Dies führt dazu, dass möglicherweise Windows gerade Teile seines Arbeitsspeichers in die Auslagerungsdatei schreibt, beziehungsweise die Auslagerungsdatei ausliest, während der Rolland auf eine Wand zufährt. So können Gefahren übersehen werden, und die Software wird unsicher aufgrund des Betriebssystems.

Um diese Probleme beim HNF-Rolland zu vermeiden, wurden verschiedene Betriebssysteme in Erwägung gezogen. So zum Beispiel mehrere Linux-Distributionen, mit und ohne den RealTime-Linux-Kernel, und QNX, bei dem es sich um ein reines RealTime-Betriebssystem handelt.

Die Entscheidung fiel auf die Linux-Systeme, da die GTRolland-Software nicht darauf ausgelegt ist, RealTime-Schnittstellen zu benutzen, und Linux eine höhere Hardware-Kompatibilität aufweist als QNX. Darüber hinaus gibt es eine umfassende Dokumentation zu einer Vielzahl von Problemen mit Linux im Internet. QNX hat auf Grund der geringeren User-Zahlen keine so umfassende Dokumentation.

### 3.1.2 RealTimeLinux

Obwohl die GTRolland-Software keine RealTime-Schnittstellen benutzt, ist es dennoch eine zeitkritische Software, eine RealTime Anwendung. Um die Möglichkeit zu wahren, die Software in diese Richtung auszubauen, wurde erwogen, den RealTime-Linux-Kernel zu verwenden. Dieser lässt sich mit allen Linux-Distributionen kombinieren. Er stellt einen normalen Linux-Kernel in fast jeder gewünschten Version dar und bietet die Möglichkeit, RealTime-Schnittstellen als Kernelmodule zu installieren. Darüber hinaus laufen Anwendungen parallel

zum Betriebssystem und können eine höhere Priorität haben als das Betriebssystem selbst. Darunter läuft nur noch der Scheduler. Sollte eine Anwendung, deren Priorität höher ist als die des Betriebssystems, an einem Punkt des Programms oder dauerhaft, sehr rechenintensiv sein, entsteht der Eindruck eines stehenden Betriebssystems. Die Anwendung jedoch läuft stabil weiter.

Aufgrund dieser Vorteile kommt der RealTime-Linux-Kernel auf dem HNF-Rolland zum Einsatz.

### 3.1.3 Die Wahl der Linux Distribution

Bei der Wahl der Linux-Distribution ist es entscheidend, dass es einfach ist, alle nicht benötigten Dienste abzuschalten. Das Ziel ist es, ein möglichst schlankes Betriebssystem zu haben und maximale Leistung für die GTRolland-Software zu gewährleisten.

Die meisten Distributionen bieten ein Grundsystem an, welches während der Installation viele Möglichkeiten zum Ausbau der gelieferten Dienste und Programme stellt, aber kaum zum Abbau von nicht benötigten Diensten. Minimal-Linux-Systeme haben meist einen bestimmten Zweck, den sie erfüllen sollen, zum Beispiel Router-Funktion für ein lokales Netzwerk. Diese Minimal-Linux-Systeme haben den Nachteil, dass ihre Dokumentationen nur auf den eigentlichen Zweck abzielen, beziehungsweise, aufgrund von geringen User-Zahlen, nicht alle Möglichkeiten dokumentiert sind.

Eine Ausnahme unter den Linux-Distributionen stellt Gentoo-Linux dar. Dies ist eine volle Distribution, die alle standardisierten Softwarepakete liefert und sich an den Linux-Standard hält. Diese stellt keine grafische Installation bereit, die Standard-Dienste und Programme automatisch installiert. Stattdessen wird eine Anleitung geliefert, wie man Schritt für Schritt die Installation manuell realisiert. Nicht nur die Installation, sondern alle Probleme, die mit der Distribution auftreten können, sind umfassend dokumentiert und werden im Gentoo-Forum diskutiert.

In der Installationsdokumentation wird genau beschrieben, welche Dienste und Programme erforderlich sind, und welche optional sind. So ist es leicht möglich, ein minimales Linux-System nach eigenen Bedürfnissen zu erstellen.

Die genannten Eigenschaften von Gentoo-Linux führten zu der Entscheidung, diese Distribution als Betriebssystem zu verwenden. Nur der Kernel wurde nicht von Gentoo bezogen. Wie in Abschnitt 3.1.2 beschrieben, kommt der RealTime-Linux Kernel in der Version 2.6.9 zum Einsatz.

### 3.1.4 Konfiguration des Betriebssystems

Das Betriebssystem wird für den HNF-Rolland auf einem 2-GB-USB-Flash-Stick von SanDisk installiert. Als erstes werden zwei Partitionen erstellt, eine für den Bootmanager und eine Systempartition. Die Bootpartition ist 32 MB groß. Der übrige Speicherplatz des USB-Sticks enthält die Systempartition. Es wird bewusst keine SWAP-Partition erstellt. Diese gehört normalerweise zum Standard bei Linuxsystemen und stellt Festplattenspeicher für die Auslagerungen aus dem Arbeitsspeicher bereit. Durch das Fehlen der SWAP-Partition wird das Betriebssystem gezwungen, keine Daten aus dem Arbeitsspeicher auszulagern, was, wie bereits beschrieben, zu Problemen führen kann.

Beide Partitionen werden mit dem Dateisystem ReiserFS formatiert. Dies ist ein Journaling-Dateisystem, was den Vorteil hat, dass bei jedem schreibenden Dateizugriff ein Journal geschrieben wird, welches die Zugriffe dokumentiert. Nach einem Systemabsturz müssen folglich nur die Dateien auf Konsistenz geprüft werden, die zur Zeit des Absturzes geöffnet waren. Da das System des HNF-Rolland nach dem Betrieb nicht heruntergefahren wird sondern mit dem Hauptschalter auch der Computer ausgeschaltet wird, ist diese Funktion ein wichtiger Sicherheitsfaktor. Zusätzlich wird die Systempartition nach dem Booten als read-only neu eingebunden, so dass sich die Zeit eines möglichen Datenverlustes auf den Bootvorgang beschränkt.

Auf alle optionalen Systemtools wird bei der Gentoo-Installation für den HNF-Rolland verzichtet. So enthält das System keinen System-Logger zum Protokollieren von Systemaktivitäten und keinen Cron-Deamon zum geplanten Ausführen von Tasks. Es wird jedoch ein SSH-Deamon installiert, um den HNF-Rolland leichter über ein Netzwerk zu warten. Außerdem wird der Start der GTRolland-Software in den Bootvorgang eingebunden.

Als Bootmanager wird Lilo installiert, weil es sich als leichter herausstellte, mit Lilo von einem USB-Stick zu booten als mit Grub. In der Lilo-Konfiguration gibt es nur eine Bootoption, die nach einem Timeout von 0 Sekunden geladen wird.

Eine detalliertere Beschreibung der Konfiguration des Betriebssystems befindet sich im Anhang A.5.

### 3.1.5 Konfiguration des RTLinux-Kernels

Der RTLinux-Kernel wird nur mit dem Minimum benötigter Module konfiguriert, um ihn möglichst schlank zu halten. Es werden im Folgenden nicht alle Module einzeln aufgezählt, sondern beschrieben, welche Unterstützungen unabdingbar sind und welche Standardmodule eines normalen Linux-Kernels nicht

installiert werden.

Es wird keine Unterstützung der Soundkarte, des CD-, beziehungsweise des DVD-Laufwerks und keine Plug and Play Unterstützung installiert. Im Gegensatz zu normalen Linux-Systemen wird nur das ReiserFS-Dateisystem und keine weiteren Linux-, CD-, oder Microsoft-Dateisysteme unterstützt.

Des Weiteren werden unterstützt: SCSI-Disk und -Emulation Unterstützung, USB-Dateisystem und -Massenspeicher, alles um auf den USB-Stick zugreifen zu können; Unterstützung von seriellen Verbindungen und USB-Seriell-Konvertern; von den Konvertern werden mehr unterstützt als benötigt werden, um Probleme bei einem eventuellen Austausch zu verhindern; Netzwerkunterstützung mit den entsprechenden Treibern für die Netzwerkkarte und Power-Management-Unterstützung, auch wenn Letzteres nicht unbedingt erforderlich wäre. Als Prozessor ist, wie im System vorhanden, der Pentium 4 Celeron angegeben.

Alle Module werden fest in den Kernel und nicht als nachladbare Module kompiliert.

Eine detalliertere Beschreibung der Konfiguration des Kernels befindet sich im Anhang A.6.

## 3.2 Die Software GTRolland

Der Software GTRolland liegt ein Framework zu Grunde, welches es ermöglicht, für Aufgaben mehrere Lösungsvarianten zu liefern. Die verschiedenen Lösungsvarianten, im folgenden Module genannt, können im laufenden Betrieb an- und ausgeschaltet werden. So ist es möglich, dass Entwickler an verschiedenen Aufgaben und Szenarien arbeiten und jeweils die für ihre Aufgabe wichtigen Module laden, ohne die Architektur der Software zu verändern. Darüber hinaus ist es möglich, verschiedene Lösungsansätze für dasselbe Problem zu entwickeln. Diese kann man abwechselnd im laufenden Betrieb testen, um den stärksten Ansatz zu erkennen.

### 3.2.1 GT200X

Das Framework GT200X stammt ursprünglich aus den Arbeiten mehrerer Robocup-Teams. 2002 wurde es für die Weltmeisterschaft der 'Sony legged League' in Fukuoka unter dem Namen GT2002 [Röfer, 2003] vom GermanTeam entwickelt. Es wurde benötigt, da sich für die Weltmeisterschaft fünf deutsche Universitäten zusammenschlossen. Bei den German Open 2002 waren sie noch gegeneinander angetreten. Dieses Phänomen besteht weiter, die deutschen Universitäten schließen sich für Weltmeisterschaften zusammen und

treten in kleineren Turnieren gegeneinander an, auch wenn sich inzwischen einzelne Universitäten aus dem GermanTeam verabschiedeten und ihr Glück auf eigene Faust suchen.

Die Software GT2002 wurde mit dem Ziel entwickelt, dass die stärksten Module der einzelnen Universitäten miteinander kombinierbar werden, um die beste Performance bei der Weltmeisterschaft zu erzielen. Es wurden feste Schnittstellen zwischen den Modulen und ein allgemeines Weltmodell definiert. So ist es möglich, die Module für Sensorik, Aktuatorik und Verhaltenssteuerung der verschiedenen Universitäten zu kombinieren und auszutauschen. Des Weiteren wurde ein Simulator und eine Debug-Queue definiert, die es ermöglichen, Code ohne Roboter auf einem PC zu testen, beziehungsweise Debug-Ausgaben zu machen. Außerdem wird ermöglicht, Log-Dateien zu schreiben, sowohl während des Betriebs als auch in der Simulation. Diese Log-Dateien ermöglichen ein Log-File-Replay im Simulator, um entstandene Situationen zu reproduzieren, zu debugen und mit neuem Code zu testen.

Das Framework erlaubt es, in den verschiedenen Schichten der Software beliebig viele Prozesse zu erzeugen. So wird den Entwicklern ein höchstmögliches Maß an Freiheit gegeben ohne die Kompatibilität zu gefährden.

Das Framework wird jedes Jahr weiterentwickelt und erscheint jeweils unter dem Namen, der die aktuelle Jahreszahl beinhaltet. Das Framework GT2004 diente als Grundlage für die beim HNF-Rolland verwendete Software. Zurzeit wird das Framework GT2007 zur Benutzung im Projekt 'Bremer autonomer Rollstuhl' umgeschrieben.

### 3.2.2 Änderungen der GT2004 Software zur GTRolland Software

Für den Gebrauch der GT2004 Software beim Rolland III mussten viele Module der Sensorik, der Aktuatorik, des Weltmodells und der Verhaltenssteuerung geändert, beziehungsweise neu geschrieben werden. Dies liegt zum einen an dem grundlegend anderen Design des Roboters sowie an den Anwendungen, die ein völlig anderes Ziel verfolgen. Der Rolland III hat im Gegensatz zu den Sony Aibo Robotern keine Beine sondern Räder, ist bedeutend größer und nutzt andere Sensoren. Die Anwendungen der Sony Aibo Roboter zielen auf ein gemeinschaftliches Fußballspiel, die des Rolland III auf das sichere Manövrieren eines Rollstuhls hin.

Viele Module konnten jedoch auch ohne, oder nur mit leichten Veränderungen, übernommen werden, so zum Beispiel die Netzwerkkommunikation, die Simulation, die Log-Dateien inklusive Log-File-Replay, die Debug-Queues und die Interprozesskommunikation.

## 3.3 Betriebssystemsabhängige Softwareanpassung

Da die GTRolland Software unter Windows XP geschrieben wurde und für den Rolland III auch unter Windows XP eingesetzt wird, war es notwendig, Softwareanpassungen zu machen, um die Software unter Linux zu verwenden. Die Änderungen betreffen die serielle Kommunikation, den Dateizugriff, die Debug-Ausgaben und Systemaufrufe, um Zeitangaben und Hostnamen abzufragen sowie die Prozessverwaltung. Die Netzwerkkommunikation, die in den Dateien TCPConnection.h und TCPHandler.h und den dazugehörigen .cpp-Dateien definiert wird, wurde so umgeschrieben, dass sie von beiden Betriebssystemen verwendet werden kann.

Die Softwareanpassungen sind nur auf den Einsatz beim HNF-Rolland ausgerichtet. Als Anwendung wurde allein die Kollisionsvermeidung, die auf dem HNF-Rolland läuft, berücksichtigt. Es ist nach den Softwareanpassungen nicht möglich, jede beliebige Anwendung des Rolland III unter Linux laufen zu lassen. Dies folgt aus der Tatsache, dass auf dem HNF-Rolland nur ein Prozess läuft. Die Prozessverwaltung der Linuxversion ist folglich auch nur auf die Verwaltung eines Prozesses ausgelegt. Andere Anwendungen benötigen mehrere Prozesse und deren Verwaltung. Des Weiteren wurden viele Klassen, die windowsspezifische Programmteile enthalten, aus der Linuxversion einfach entfernt, da sie für den Einsatz in der Ausstellung 'Computer.Medizin' nicht relevant waren. Außerdem gibt es unter Linux keine Unterstützung für einen Simulator und kein Log-File-Replay.

Sollten die Vorteile des Linuxsystems auch für die Forschung auf dem Rolland III zum Tragen kommen, so müssen weitere umfassende Anpassungen der Linuxversion der GTRolland-Software folgen.

## 3.4 Kollisionsvermeidung

Die meisten Kollisionsvermeidungsalgorithmen in der Robotik basieren auf einem Ausweichverhalten. Sie werden eingesetzt, um Robotern, die autonom Wege befahren, eine Möglichkeit zu geben, einen neuen Weg zu berechnen, wenn Hindernisse in ihrem ursprünglichen Weg stehen.

Um unerwarteten Hindernissen ausweichen zu können, müssen sensorbasierte reaktive Methoden verwendet werden. Zu diesen Zählen die 'Vektorfeld Histogram Methode' [Borenstein und Koren, 1991], der Einsatz von Potentialfeldern ([Khatib, 1986] und [Khatib und Chatila, 1995]), die 'Curvature-Velocity Method' [Simmons, 1996] und der 'Dynamic Window Approach' [Fox u. a., 1997].

Viele Ansätze setzen Roboter ein, welche eine runde Grundform haben, um Kollisionen zur Laufzeit anhand von einfachen mathematischen Formeln zu be-

rechnen([Fox u. a., 1997], [Stachniss und Burgard, 2002] und [Siegwart u. a., 2003]).

Das 'NavChair Project' [Levine u. a., 1999] entwickelte eine Kollisionsvermeidung für Rollstühle, die ein Vektorfeld Histogram einsetzt [Bell u. a., 1994]. An der Universität von Shiga wurde auf einem Rollstuhl eine Kollisionsvermeidung eingesetzt, die auf einem neuronalen Netz basiert [Yasuda u. a., 2006]. [Lankenau und Röfer, 2000] gibt einen ausführlichen Überblick über intelligente Rollstühle und ihre Applikationen.

Die Kollisionsvermeidung des HNF-Rolland setzt hingegen kein Ausweichverhalten ein, sondern stoppt vor Hindernissen, bevor es zu Kollisionen kommt. Der Benutzer muss selbstständig eine neue Route vorgeben, die das Hindernis umgeht. Des Weiteren ist es nicht möglich, einem Rollstuhl eine runde Form zu geben. Dennoch gibt es Ansätze, die sowohl bei der Kollisionsvermeidung des HNF-Rollands als auch bei anderen Kollisionsvermeidungen mit Ausweichverhalten eingesetzt werden.

Wie schon in Abschnitt 1.2 beschrieben wird, setzt die Kollisionsvermeidung des Rolland III, die mit leichten Veränderungen des Fahrverhaltens auch beim HNF-Rolland eingesetzt wird, die Ansätze des Rolland I und II fort.

Im Folgenden werden Methoden beschrieben, deren Ansätze in Teilen der Kollisionsvermeidung auf dem Rolland II und Rolland III sowie des HNF-Rollands ähneln.

### 3.4.1 Der 'Dynamic Window Approach'

Einer der Ansätze, der auf einem Ausweichverhalten basiert, ist der 'Dynamic Window Approach' [Fox u. a., 1997]. Er wurde für den Roboter 'RHINO' entwickelt und zur Kollisionsvermeidung in Umgebungen mit dynamischen Hindernissen, wie Personen, eingesetzt.

Der Ansatz basiert darauf, dass der Suchraum der nächsten Geschwindigkeit auf sichere Geschwindigkeiten reduziert wird. Im Sinne der Entwickler beinhaltet 'Geschwindigkeit' sowohl die absolute als auch die rotierende. Der Suchraum besteht also aus Paaren von Geschwindigkeit und Lenkbewegung. In diesem verbleiben nur Paare, die nicht zu Kollisionen führen. Des Weiteren werden nur Fahrbefehlspaare berücksichtigt, welche im nächsten Zeitintervall erreicht werden können.

Aus den verbleibenden Paaren im Suchraum wird nun jenes für den nächsten Befehlsschritt ausgewählt, das eine Zielfunktion maximiert. Diese optimiert die Summe aus 'Bewegungsrichtung auf das Hindernis', 'Abstand zum nächsten Hindernis' und 'maximaler Geschwindigkeit'.

Der Ansatz [Stachniss und Burgard, 2002] kritisiert den 'Dynamic Window Approach' für fehlende Weitsicht. Durch die Maximierung der Zielfunktion wird bei weiter Entfernung zum Ziel oft eine sehr hohe Geschwindigkeit präferiert. Diese ist jedoch hinderlich, wenn in der momentanen Situation eine scharfe Kurve, zur Durchfahrt eines Korridors oder einer Tür gefordert ist.

In [Stachniss und Burgard, 2002] wird daher, beschrieben den 'Dynamic Window Approach' nur zu verwenden, um den Suchraum des nächste Geschwindigkeitsbefehls einzuschränken. Für die genaue Berechnung des nächsten Befehls wird ein Unterziel definiert. Wie weitschauend dieses ist, hängt von dem vorhandenen Speicher und der Rechenleistung des eingesetzten Computers ab.

### 3.4.2 'Fast Local Obstacle Avoidance' von C. Schlegel

Der Ansatz in [Schlegel, 1998] setzt Lookup-Tables ein, um eine nicht runde Form eines Roboters zur Laufzeit auf Kollisionen zu prüfen. Auch dieser beinhaltet Wegplanung und Ausweichverhalten eines autonom steuernden Roboters. Ähnlich wie beim 'Dynamic Window Approach' werden nur Bewegungen berücksichtigt, die im nächsten Befehlsschritt erreicht werden können.

Die besondere Aufgabe dieses Ansatzes ist es, dass der eingesetzte Roboter keine runde Grundform besitzt. Geometrische Berechnungen der Entfernungen zu Hindernissen erschweren sich dadurch deutlich und werden daher vorberechnet.

In den Lookup-Tables wird gespeichert, welche Fahrbefehle im nächsten Schritt ausgeführt werden können. Des Weiteren wird eine Gitterkarte aufgebaut, in diese wird zur Laufzeit die Hindernissituation um den Roboter eingetragen. Dank der Vorberechnung kann aus den Zellen ausgelesen werden, wie weit die Hindernisse noch von dem Roboter entfernt sind. Ebenfalls ist es möglich, die Geschwindigkeiten auszulesen, die maximal ausgeführt werden können, ohne eine unsichere Situation in Bezug auf Kollisionen zu erzeugen.

Durch diese Lookup-Tables ist es möglich, eine effiziente Kollisionsvermeidung zu gestalten, ohne dass der Roboter an eine leicht zu berechnende Form gebunden ist.

### 3.4.3 Die virtuellen Sensoren des Rolland II

Der Ansatz, welcher der Kollisionsvermeidung des Rolland III und des HNF-Rollands am meisten ähnelt, ist aus offensichtlichen Gründen die des Rollands II [Lankenau und Röfer, 2001].

Der Rolland II realisiert die Kollisionsvermeidung auf einer lokalen Hinderniskarte, ähnlich wie bei [Schlegel, 1998]. Die globale Hindernissituation, die

von den Sensoren erkannt wird, wird in eine Gitterkarte der Größe 402 $cm \times$ 402 $cm$ übertragen. Eine Zelle dieser Karte ist 3 $cm \times 3$ $cm$ groß. Der Rollstuhl befindet sich immer in der Mitte dieser Karte. Bewegt sich der Rollstuhl, so werden die Hindernisse in der Karte entsprechend der erkannten Bewegung durch die Odometriesensoren verschoben. Bei Drehbewegungen des Rollstuhls werden nicht die Hindernisse gedreht, da dies zu rechenaufwändig wäre, sondern der Rollstuhl wird entsprechend in der Mitte der Karte gedreht.

Durch die Einführung der Gitterkarte wird der Suchraum, in dem nach kritischen Hindernissen gesucht wird, schon deutlich verringert. Es bedeutet dennoch einen zu großen Umstand, alle Zellen der Gitterkarte zur Laufzeit zu durchsuchen. Ebenso ist es zu zeitaufwändig, die Entfernungen zwischen Rollstuhl und jedem Hindernis zu berechnen, da der Rolland keine runde Form besitzt.

Zur Lösung dieses Problems werden Lookup-Tables eingesetzt. Diese nennen sich 'Virtuelle Sensoren'. Es wird abhängig von Geschwindigkeit und Lenkbewegung vorberechnet, welche Zellen der Gitterkarte nach kritischen Hindernissen durchsucht werden müssen. Dabei wird eine Liste von Zellenindizes aufgebaut, die jene Zellen angibt, welche kritische Hindernisse enthalten könnten. Eine solche Liste wird für jede mögliche Orientierung des Rollstuhls in der Karte, jede mögliche Bewegungsrichtung und jede mögliche Geschwindigkeit berechnet. Die Zellenindizes werden so sortiert, dass beim Überprüfen das dichteste Hindernis als erstes erkannt wird, und somit, bei der ersten Erkennung, die Überprüfung abgebrochen werden kann. Die Speicherung dieser Listen erfolgt in so genannten Kollisionstabellen. In diesen wird zudem zu jeder Zelle der Abstand zum Rollstuhl bei der jeweiligen Orientierung des Rollstuhls gespeichert.

Aus den Kollisionstabellen wird demnach zur Laufzeit ausgelesen, welche Zellen überhaupt geprüft werden müssen und wie groß der mögliche verbleibende Bremsweg zum nächsten Hindernis ist. Zur Laufzeit erfolgt somit nur noch die Auswertung dieser Informationen. Auf dieser Grundlage wird die zu verwendende sichere Geschwindigkeit für den nächsten Geschwindigkeitsbefehl berechnet. Ähnlich wie bei [Fox u. a., 1997] wird nur der nächste Befehlsschritt betrachtet.

Aus der Tatsache, dass diese Kollisionstabellen nicht zur Laufzeit berechnet werden, ergibt sich, dass die Ausmaße des Rollstuhls keine runde Grundform haben müssen, wie dies in anderen Ansätzen der Fall ist. Es ist möglich, die tatsächlichen Ausmaße, so komplex sie auch sind, zu verwenden. Das Ausschwenken von einigen Bereichen des Rollstuhls kann so berücksichtigt werden.

Der Suchraum, der zur Laufzeit verarbeitet werden muss, reduziert sich auf ein Minimum.

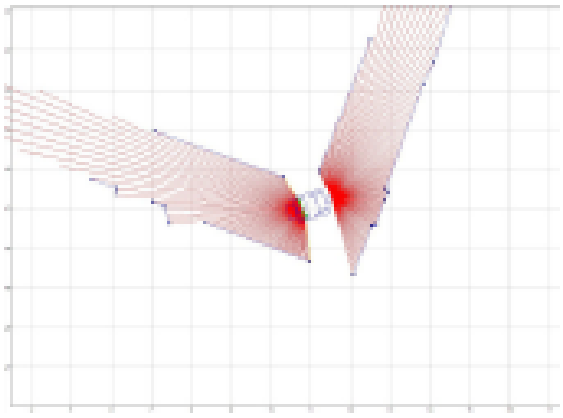


Abbildung 12: Die toten Winkel des Rolland III und des HNF-Rolland (Bild aus [Mandel u. a., 2005])

### 3.4.4 Die Virtuellen Sensoren des Rollands III und des HNF-Rollands

Die beim Rolland III und beim HNF-Rolland eingesetzte Kollisionsvermeidung [Mandel u. a., 2005] ist eine direkte Weiterentwicklung der virtuellen Sensoren des Rollands II (Abschnitt 3.4.3). Dabei wird berücksichtigt, dass der eingesetzte Rollstuhl Meyra Champ einen Differenzialantrieb besitzt. Außerdem werden nicht mehr die Abstände zum nächsten Hindernis in den Kollisionstabellen gespeichert, sondern die vorberechnete maximal sichere Geschwindigkeit.

Zudem muss berücksichtigt werden, dass der Rolland III, ebenso wie der HNF-Rolland, tote Winkel in der Raumerkennung besitzt. Dies resultiert daraus, dass die Sensoren zur Raumerfassung nicht, wie beim Rolland II, einen Ring um den Rollstuhl darstellen, sondern nur vorne und hinten Laserscanner angebracht sind (siehe Abbildung 12). Hindernisse, die zuvor erkannt worden sind, und durch die Bewegung des Rollstuhls in die toten Winkel verschwinden, werden gespeichert. Damit es zu keiner Situation kommt in der jede Bewegung blockiert wird, werden diese gespeicherten Hindernisse jedoch nach einer gewissen Zeit wieder gelöscht.

Um in die unbekannten Bereiche der toten Winkel fahren zu können ohne die Gefahr von Kollisionen mit größerem Schaden zu riskieren, wird für Fahrten in unbekannte Bereiche eine maximale Geschwindigkeit definiert. Dies ist besonders zu Beginn einer Fahrt nötig, da jede Bewegung nach vorne durch unbekannte Bereiche führt. Diese Tatsache resultiert aus den vorderen Lenkrädern, die sich im Scanbereich der vorderen Laserscanner befinden und somit tote Winkel erzeugen.

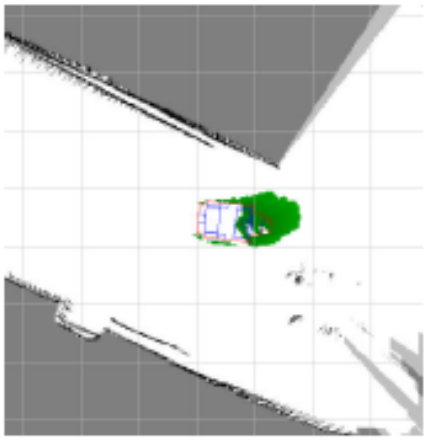


Abbildung 13: Die virtuellen Sensoren des Rolland III bei einer leichten Linkskurve (Bild aus [Mandel u. a., 2005])

Die eingesetzte Gitterkarte zur Darstellung der lokalen Hindernissituation entspricht in Größe und Auflösung der des Rollands II. Auch der Ansatz bei Rotationen des Rollstuhls nicht die Karte, sondern den Rollstuhl innerhalb der Karte, zu rotieren, wurde übernommen.

Anders als beim Rolland II wird in den Kollisionstabellen nicht mehr die Entfernung zwischen Rollstuhl und einem Hindernis gespeichert, sondern eine vorberechnete maximal sichere Geschwindigkeit. Dies erspart einen weiteren Rechenvorgang zur Laufzeit.

Die Kollisionstabellen enthalten die virtuellen Sensoren. Diese geben die Zellen der Gitterkarte an, die, abhängig von Geschwindigkeit und Lenkbewegung des Rollstuhls, nach Hindernissen durchsucht werden müssen. Abbildung 13 zeigt in grün diese Zellen. Die Aufnahme der virtuellen Sensoren wurde auf dem Rolland III gemacht, als dieser eine Linkskurve in mäßiger Geschwindigkeit fuhr.

Wie auch beim Rolland II benötigt der Einsatz der Kollisionstabellen verhältnismäßig viel Arbeitsspeicher. So sind beim HNF-Rolland die Kollisionstabellen 3,31 MB groß. Es ist jedoch eine übliche Vorgehensweise, eine Ersparnis an Berechnungen zur Laufzeit auf Kosten von Arbeitsspeicher zu realisieren.

Für die Berechnung der Kollisionstabellen werden die Ausmaße des Rollstuhls und seine Fahreigenschaften verwendet. Zu diesen gehören unter anderem die maximale Geschwindigkeit, die Drehgeschwindigkeit, die Latenzzeit zwischen Fahrbefehlen und deren Ausführung, die maximale Bremswirkung sowie die maximale Beschleunigung.

Wie bereits erwähnt sind diese Werte beim Rolland III und dem HNF-Rolland

nicht gleich. Der HNF-Rolland hat weichere Fahreigenschaften, aus denen eine kleinere Höchstgeschwindigkeit und eine kleinere Bremswirkung resultieren. Auch die Ausmaße sind nicht identisch, wie in Abschnitt 2.1 beschrieben wird.

Daraus geht hervor, dass die Kollisionstabellen für den HNF-Rolland neu berechnet werden mussten.

Im Prinzip war es nicht nötig, das Programm zu verändern. Wie in Abschnitt 4.5.1 beschrieben wird, gab es jedoch einige Probleme mit den Kollisionstabellen, die vorher nicht aufgetreten waren, so dass leichte Programmänderungen sich nicht umgehen ließen.

Die maximal sichere Geschwindigkeit, die zur Laufzeit aus den Kollisionstabellen ausgelesen wird, wird eingesetzt um den nächsten Fahrbefehl sicher zu gestalten. Das daraus resultierende Fahrverhalten hat verschiedene Ziele beim Rolland III und dem HNF-Rolland. Abschnitt 4 beschäftigt sich im Folgenden ausführlich mit diesen Zielen und dem daraus resultierenden Fahrverhalten.

## 3.5 Änderungen im Protokoll des Bedienmoduls

Bei dem Rolland III und dem HNF-Rolland verhält sich das Protokoll, über welches der Rollstuhl angesteuert wird, divergent. Dies ist jedoch nicht mit einem grundlegend veränderten Protokoll zu erklären. Alle Änderungen sind bereits in der von Meyra bereitgestellten Dokumentation des Protokolls zu finden, die für den Rolland III verwendet wurde. Das Bedienmodul des HNF-Rollands ist jedoch restriktiver. Das für den HNF-Rolland eingesetzte Protokoll funktioniert auch beim Rolland III, das vorher für den Rolland III verwendete jedoch nicht beim HNF-Rolland.

So muss beim HNF-Rolland ein Bit des Protokolls gesetzt werden, damit die Werte des internen Joysticks des Meyra Champs, über die serielle Schnittstelle, an den Computer gesendet werden. Beim Rolland III wurden diese in jedem Fall gesendet.

Außerdem ist das Protokoll so beschrieben, dass der Computer nach dem Einschalten des Rollstuhls sofort ein Datenpaket über die serielle Schnittstelle senden muss, um die Kommunikation aufzubauen. Antwortet der Rollstuhl auf dieses Paket, so ist die Kommunikation hergestellt und der Computer sendet nach jedem empfangenen Paket ein Datenpaket mit den nächsten Fahranweisungen.

Es ist ein Timeout definiert, welches angibt, in welcher Zeit ein Paket beantwortet wird. Wurde dieses überschritten, sendete das Protokoll, das beim Rolland III zum Einsatz kam, ein Default-Paket, um die Kommunikation sofort

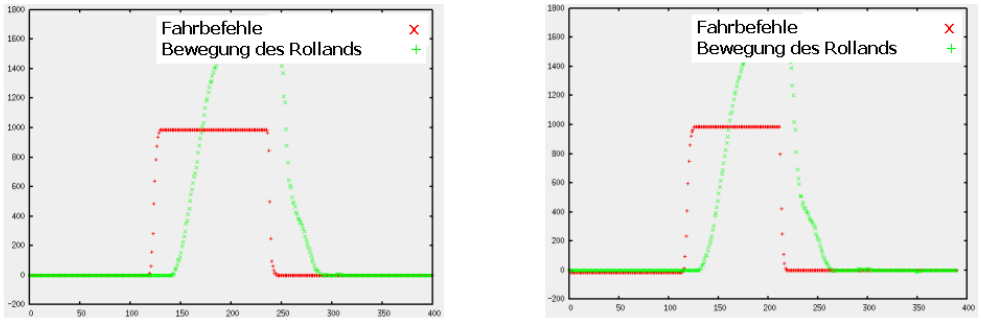



Abbildung 14: Latenz zwischen Fahrbefehlen und Bewegungen des Rollands
Links: große Latenz durch falsche Nutzung des Protokolls
Rechts: kleine Latenz durch optimale Nutzung des Protokolls
X-Achse: Zeit im Aktualisierungstakt der GTRolland-Software
Y-Achse: Geschwindigkeit in $\frac{mm}{sec}$

wiederherzustellen. Es wurden also zwei oder noch mehr Datenpakete gesendet, ohne dass zwischendurch ein Paket vom Rollstuhl empfangen wurde. Beim Einschalten des Rollstuhls kommt es jedoch immer zur Überschreitung des Timeouts, da der Rollstuhl zunächst einen Selbsttest durchführt.

Beim Rolland III verursachte dies keine Probleme, beim HNF-Rolland hingegen scheint es im Bedienmodul des Meyra Champ einen Puffer für diese Datenpakete zu geben. So führt der Rollstuhl nicht den zuletzt empfangenen Fahrbefehl aus, sondern den ältesten im Puffer. Der neue Fahrbefehl wird an letzter Stelle in die Datenschlange im Puffer gelegt. Dies führte zu einer Latenzzeit in der Ausführung der Fahrbefehle. Diese Latenz machte die Steuerung des Rollstuhls unnötig schwierig, auch wenn man sie als Nutzer nur bemerkt, wenn man den direkten Vergleich zwischen einer Fahrt mit Latenz und ohne Latenz hat. Abbildung 14 veranschaulicht diese Latenz in Form eines Diagramms. Darin erkennt man, dass die Latenzzeit sich um mehr als ein Drittel verringert, wenn das Protokoll beim Start über das Timeout hinaus auf die Antwort des Bedienmoduls wartet.

Es bleibt zu erwähnen, dass auf dem Rolland III keine offizielle Firmware zum Einsatz kommt, sondern die Software des Bedienmoduls speziell für das Projekt 'Bremer autonomer Rollstuhl' gepatcht wurde. Dies geschah, um die direkten Fahrwerte (siehe Abschnitt 4.1), welche im normalen Betrieb eines Rollstuhls eher hinderlich wären, umzusetzen. Es ist nicht ausgeschlossen, dass im Zuge

dieses Patches die beschriebenen Änderungen im Protokoll zustande kamen. Der Firmware-Patch wurde von Meyra-Mitarbeitern und nicht vom Projekt 'Bremer autonomer Rollstuhl' erstellt und bereitgestellt.

# 4 Evaluation des Fahrverhaltens

In diesem Abschnitt wird das Fahrverhalten des HNF-Rollands beschrieben. Dies baut, wie alle Module des HNF-Rollands, auf den Erfahrungen des Rolland III auf. Für den Einsatz als interaktives Museumsexponat werden jedoch andere Anforderungen an das Fahrverhalten gestellt als an den Rolland III.

Zunächst wird das Fahrverhalten des Rolland III beschrieben, dann die zu erwartenden Probleme beim Einsatz in einem Museum. Schließlich wird die Evaluation beschrieben, die zu dem eingesetzten Fahrverhalten führte.

Softwaretechnisch wird das Fahrverhalten in der Klasse HNFSafetyLayer umgesetzt. Der Quellcode dieser Klasse findet sich im Anhang A.7.

## 4.1 Die Fahrwerte des HNF-Rollands

Ein Unterschied, welcher für die Entwicklung des Fahrverhaltens entscheidend ist, sind die Fahrwerte des HNF-Rollands. Damit ist die Umsetzung der Fahrbefehle von Seiten des Meyra Champs gemeint.

Für den normalen Gebrauch eines Rollstuhls ist es vorteilhaft, weiche Übergänge zwischen Fahrbefehlen zu realisieren. Im Fall, dass der Benutzer den Joystick schnell vor und zurück bewegt, soll verhindert werden, dass der Rollstuhl sich sehr ruckartig bewegt. Die Leistung der Motoren würde in diesem Fall ausreichen, um eine unvorbereitete Person vom Rollstuhl zu werfen, wenn Vor- und Rückwärtsbewegungen in einem Abstand von unter einer Sekunde ausgeführt würden. Gleiches könnte passieren, wenn der Rollstuhl aus dem Stand die maximale Motorenleistung benutzen würde.

Um die Benutzer eines Rollstuhls nicht zu erschrecken oder gar zu verletzen, werden nur Fahrwerte zur Einstellung angeboten, die weiche Übergänge zwischen Fahrbefehlen realisieren.

In der Robotik ist genau der entgegengesetzte Fall erforderlich. Es ist von Vorteil, wenn Fahrbefehle schnell umgesetzt werden, um der Software eine maximale Kontrolle über den Roboter zu geben.

Da der Rolland III eine Testplattform der Robotik ist, wurden sehr direkte Fahrwerte eingestellt. Dafür war ein spezielles Patch der Fahreigenschaften des Meyra Champs erforderlich, welches der Hersteller bereit stellte. Es ist also möglich, dem Rolland III mit einem Fahrbefehl eine Beschleunigung und im nächsten eine Bremsung zu befehlen. Diese Befehle werden umgehend umgesetzt. Kein normaler Rollstuhl des Herstellers kann diese Fahrwerte einstellen.

Beim HNF-Rolland wurde kein Patch aufgespielt. Es sind also Fahrwerte eines handelsüblichen Meyra Champs eingestellt. Es wurden die direktesten normal einstellbaren Fahrwerte gewählt. Diese sind immer noch deutlich weicher

als beim Rolland III.

Wird beim HNF-Rolland eine Beschleunigung und anschließend eine Bremsung befohlen, so wird vom Rollstuhl ein weicher Übergang realisiert. Diese Fahrwerte wurden für den HNF-Rolland ausgewählt, da er von unerfahrenen Museumsbesuchern bedient werden soll, und diese nicht durch starke Beschleunigungen und Bremsungen erschreckt oder verletzt werden sollen.

Besonders im Nahbereich verursachen diese Fahrwerte Probleme, da der HNF-Rolland die Reaktionen der Software auf die Hindernissituation nicht so schnell umsetzt wie dies beim Rolland III der Fall ist. Als Folge dessen muss deutlich früher in die Geschwindigkeit des Rollstuhls eingegriffen werden. Das heißt, dass im Nahbereich von Hindernissen nur sehr geringe Geschwindigkeiten erlaubt werden können.

Im Verlauf der Evaluation des Fahrverhaltens spielten diese Fahrwerte immer wieder eine entscheidende Rolle.

## 4.2 Fahrverhalten bei der SAMSafetyLayer des Rolland III

Bei der SAMSafetyLayer werden die Werte für die maximal sichere Geschwindigkeit aus den Kollisionstabellen mit den Odometriedaten verglichen. Es werden zwei Grenzen definiert; eine, bei der die Geschwindigkeit reduziert wird und eine, bei der eine Vollbremsung eingeleitet wird. Die erste Grenze wird Softbrakegrenze genannt, die zweite Hardbrakegrenze. Beide liegen unterhalb der maximal sicheren Geschwindigkeit, um sicher Kollisionen zu vermeiden.

Überschreiten die Odometriedaten die Softbrakegrenze, so werden die Fahrbefehle für die Geschwindigkeit nach unten modifiziert. Die Herabsetzung der Geschwindigkeit erfolgt entsprechend stärker, wenn die Softbrakegrenze stark überschritten wird. Sollte die Softbrakegrenze nur leicht überschritten werden, so wird der Geschwindigkeitsbefehl auch nur leicht modifiziert. Somit ist für den Benutzer nicht an einer Stelle der Bewegung eine harte Bremsung zu spüren. Stattdessen wird die eingeleitete Bremsung stärker, je mehr sich der Rollstuhl dem Hindernis nähert. Weicht der Benutzer während der Softbrakephase dem Hindernis aus, so wird die Bremsung unterbrochen, da kein Hindernis mehr im Fahrweg ist.

Wird die Hardbrakegrenze überschritten, leitet sich eine Vollbremsung ein. Während dieser hat der Benutzer keine Kontrolle mehr über den Rollstuhl. Sowohl Geschwindigkeitsbefehle als auch Lenkbefehle werden von der Software auf 0 gesetzt. Die Vollbremsung gilt erst als abgeschlossen, wenn der Rollstuhl stillsteht und der interne Joystick in der Nullstellung ist. Der Joystick kehrt automatisch in diese zurück, wenn er losgelassen wird. Natürlich ist es auch möglich, den Joystick mit der Hand in die Nullstellung zu bringen. Da der Bereich des

analogen Joysticks, der als Nullstellung definiert ist, sehr klein ist, ist es schwer, die Nullstellung mit der Hand zu treffen. Es ist deutlich leichter, den Joystick kurz loszulassen.

Da die Software immer die maximal sichere Geschwindigkeit mit den Odometriedaten vergleicht, wird im Stillstand nach einer Vollbremsung immer noch eine Bewegung in Richtung des Hindernisses erlaubt. Dies folgt daraus, dass die Vollbremsung schon leicht vor der Überschreitung der maximal sicheren Geschwindigkeit eingeleitet wird. Der Rollstuhl hält vor dem Hindernis an, somit ist eine minimale Geschwindigkeit noch möglich. Der Rollstuhl wird erst dann daran gehindert, weiter in Richtung des Hindernisses zu fahren, wenn eine Kollision schon erkannt wurde. Im Nahbereich ist es also möglich, mehrfach den Rollstuhl in Richtung des Hindernisses zu beschleunigen, was dazu führt, dass sofort die nächste Vollbremsung eingeleitet wird. Der Benutzer kann demnach den Rollstuhl ruckartig in Richtung des Hindernisses bewegen bis eine Kollision erkannt wird.

Auf dieses Phänomen geht es zurück, dass eine Vollbremsung nicht unterbrochen wird und eine Nullstellung des internen Joysticks als Bedingung für den Abbruch der Vollbremsung gefordert wird. Würde die Vollbremsung wieder aufgehoben, sobald die Hardbrakegrenze nicht mehr überschritten ist, würde der Rollstuhl während des Bremsvorgangs mehrfach beschleunigen und anschließend wieder eine Vollbremsung einleiten. Dies würde sich für den Benutzer wie eine stotternde Bremsung anfühlen, was nicht erwünscht ist.

Ein ähnliches Phänomen würde auftreten, wenn die Nullstellung des Joysticks nicht gefordert wäre. Zwar würde die Vollbremsung bis zum Stillstand durchgeführt, das eben beschriebene 'Heranruckeln an Hindernisse' würde aber ohne Unterbrechung in einer stotternden Bewegung erfolgen. Dies wäre erst beendet, wenn die Software eine Kollision schon erkannt hat. Da die Software aus Sicherheitsgründen von leicht größeren Ausmaßen des Rollstuhls ausgeht als dies tatsächlich der Fall ist, heißt das nicht, dass wirklich schon eine Kollision stattgefunden hat.

Dieses Verhalten birgt Probleme mit bewegten Hindernissen: Bewegt sich ein Hindernis quer zur Fahrtrichtung des Rollstuhls und leitet die Entfernung zu diesem eine Vollbremsung ein, so wird diese bis zum Stillstand ausgeführt, auch wenn sich das Hindernis längst aus dem Fahrweg des Rollstuhls bewegt hat.

Die weichen Fahrwerte des HNF-Rollands (siehe Abschnitt 4.1) haben zur Folge, dass in der Phase des 'Heranruckelns an Hindernisse' Kollisionen unvermeidbar sind, da die Bremsung nach einer Beschleunigung nicht so direkt erfolgt wie beim Rolland III. Es war also notwendig, dieses Fahrverhalten zu ändern. Die Evaluation des neuen Fahrverhaltens, das noch andere Änderungen für die Benutzerfreundlichkeit beinhaltet, wird in den Abschnitten 4.4 und 4.5

beschrieben.

Ein weiteres bekanntes Problem der 'SAMSafetyLayer' ist das des Festfahrens. Nach einer Vollbremsung, besonders nach anschließendem Heranruckeln an das Hindernis, werden oft die Fahrbefehle, die den Rollstuhl von dem Hindernis wegbewegen sollen, nicht ausgeführt. Dies wird darauf zurückgeführt, dass die Software ein Ausschwenken des Rollstuhls auf der zur Bewegung hinteren Seite des Rollstuhls vermutet. Da die Ausmaße des Rollstuhls, die die Software annimmt, aus Sicherheitsgründen, einige Zentimeter größer sind als in Wirklichkeit, ist für den Benutzer nicht ersichtlich, warum eine Kollision vermutet wird. Besonders stark tritt dieses Problem auf, wenn eine Kollision von der Software schon erkannt wurde. Die virtuellen Sensoren testen nicht nur den vermuteten Fahrweg des Rollstuhl und die Bereiche, in die er ausschwenkt, sondern auch den Bereich auf dem der Rollstuhl steht. Wird also ein Hindernis innerhalb der Ausmaße des Rollstuhls erkannt oder ist dieses noch im 'Gedächtnis' der Hinderniskarte, so wird jeder Fahrbefehl untersagt.

Auch dieses Problem wird durch die weichen Fahrwerte des HNF-Rollands noch verstärkt, da es durch diese leichter zu Kollisionen, beziehungsweise zu erkannten Kollisionen mit den größeren internen Ausmaßen kommt.

## 4.3 Ansätze und Ziele für das Fahrverhaltens im Museum

Wie schon in Abschnitt 4.2 beschrieben ist es erforderlich, für den HNF-Rolland ein neues Fahrverhalten zu realisieren. Zum einen aufgrund anderer Fahrwerte, zum anderen, um ein intuitives, benutzerfreundliches Fahrverhalten umzusetzen. Um das Fahrverhalten des Rolland III nicht zu ändern und weiter eine Kollisionsvermeidung für die Fahrwerte des Rolland III bereitzustellen, wurde für den HNF-Rolland, aufbauend auf der 'SAMSafetyLayer', eine 'HNFSafetyLayer' geschrieben.

Für die Fahrwerte muss es, wie in Abschnitt 4.2 beschrieben, unmöglich sein, nach einer Vollbremsung in Richtung des Hindernisses zu fahren. Um den Benutzer nicht dadurch zu irritieren, dass der Rollstuhl weit vor dem Hindernis anhält, muss der Rollstuhl sich bis kurz vor dem Erkennen einer Kollision auf ein Hindernis zubewegen können. Diese Bewegung muss ohne Unterbrechungen durch eine Vollbremsung oder eine stotternde Bewegung erfolgen. Außerdem müssen andere Kollisionstabellen berechnet werden, da sich die Werte für Beschleunigung, Bremswirkung und Höchstgeschwindigkeit vom Rolland III stark unterscheiden.

Für die Benutzerfreundlichkeit wird vermutet, dass es sinnvoll ist, die Nullstellung des Joysticks nicht als Abbruchbedingung einer Vollbremsung zu fordern. Auch das Durchführen einer Vollbremsung bis zum Stillstand ist fraglich,

wenn sich das Hindernis bereits aus dem Fahrweg entfernt hat. Dafür ist ein früheres Eingreifen in die Geschwindigkeit, beziehungsweise die Fahrbefehle, nicht unerwünscht.

Beim Rolland III war es gefordert, nur gering in die Fahranweisungen einzugreifen, um möglichst viele Fahrbefehle unverändert ausführen zu können. Dies liegt daran, dass die 'SAMSafetyLayer' auch in Verbindung mit anderen Anwendungen zum Einsatz kommen kann, denen eine große Freiheit in der Wahl der Fahranweisungen gelassen werden soll.

Im Museum hingegen kann davon ausgegangen werden, dass die Benutzer zum größten Teil keine Erfahrungen mit der Steuerung eines Rollstuhls haben. Daher ist es wünschenswert, dass nicht erst im letzten Moment gebremst wird, sondern der Bremsvorgang möglichst weich erfolgt. Ein starkes Eingreifen in die Geschwindigkeit ist somit nicht nur möglich, sondern auch empfehlenswert.

Als besonders wichtig wurde die Lösung des Festfahrproblems erachtet. Es wurde davon ausgegangen, dass ein Benutzer keinerlei Verständnis dafür hat, dass eine Bewegung von einem Hindernis weg unterbunden wird. Als einziger Ansatz zur Lösung dieses Problems wurde zu diesem Zeitpunkt eine Änderung der virtuellen Sensoren erdacht.

Für eine Bestätigung dieser Vermutungen wurden Testpersonen beobachtet, die den HNF-Rolland mit dem Fahrverhalten der SAMSafetyLayer testeten. Die Beschreibung der Beobachtungen und die daraus resultierenden Änderungen des Fahrverhaltens finden sich in den Abschnitten 4.4 und 4.5.

## 4.4 Der erste Praxistest im HNF

Um notwendige Änderungen am Fahrverhalten der 'SAMSafetyLayer' zu erkennen, wurden die Schüler zweier Schulklassen beim Praxistest der Ausstellung 'Computer.Medizin' im Heinz Nixdorf MuseumsForum beobachtet. Sie konnten den HNF-Rolland schon vor Beginn der Ausstellung testen. Das eingestellte Fahrverhalten war das der 'SAMSafetyLayer'. Geändert wurden bis zu diesem Zeitpunkt lediglich die Kollisionstabellen, die auf die Fahrwerte des HNF-Rollands umgerechnet wurden. Außerdem wurde das Heranfahren an ein Hindernis nach einer Vollbremsung verhindert. Dazu ging die Software beim Stillstand des Rollstuhls von einer Geschwindigkeit von 30 $\frac{cm}{sec}$ aus. Je nach Stand des Joysticks wurde die Geschwindigkeit als Vorwärts- beziehungsweise Rückwärtsbewegung interpretiert. Durch diesen 'Trick' lieferten die Kollisionstabellen nur einen Wert größer Null, wenn sich keine Hindernisse im Nahbereich befanden. Dennoch waren manchmal nach einer Vollbremsung ruckartige Bewegungen in Richtung des Hindernisses möglich. Diese führten in dem Test jedoch nie zu Kollisionen.

| Beobachtung | Anzahl |
|---|---|
| Personen insgesamt | 27 |
| Probleme mit der Nullstellungbedingung zum Abbruch einer Vollbremsung | 12 |
| Festfahren frontal | 12 |
| Festfahren hinten | 0 |
| Festfahren auf der freien Fläche | 13 |
| Rotationssperre | 5 |
| Rotationskollision versucht | 11 |
| Kollision hinten versucht | 13 |
| Kollision vorne versucht | 19 |
| Kollision mit beweglichen Zielen versucht | 7 |
| Kollisionen beim Rotieren | 1 |
| Kollisionen vorne | 0 |
| Kollisionen hinten | 0 |
| Kollisionen mit beweglichen Zielen | 4 |

Tabelle 3: Beobachtungen bei den Testpersonen im HNF-Rolland

### 4.4.1 Beobachtungen beim Praxistest

Die Schulklassen waren eine 8. Klasse einer Realschule und eine Berufsschulklasse. Insgesamt 27 Schüler testeten den HNF-Rolland beim Praxistest der Ausstellung.

Tabelle 3 zeigt die Probleme, die bei den Testpersonen auftraten. Die wenigsten hatten überhaupt keine Probleme, mit Ausnahme von einigen, die sich nur einige Sekunden in dem HNF-Rolland aufhielten.

Über die Probleme hinaus, die in der Tabelle 3 dargestellt werden, wurde beim Praxistest beobachtet, dass Museumsbesucher gerne Menschen als Testhindernisse benutzen. Wenn sich diese Personen auf den Rolland zubewegen, ist es möglich, dass der berechnete Bremsweg nicht ausreicht und es zu Kollisionen kommt. Besonders besteht dieses Problem, wenn sich Personen schnell von der Seite in den Fahrweg des Rolland bewegen. Dieses Problem wird durch die Reduzierung der Höchstgeschwindigkeit (Abschnitt 4.5.6) minimiert. Ganz kann dieses Problem nicht gelöst werden, denn dazu müssten Bewegungen der erkannten Hindernisse vorausgeahnt werden. Dieses Problem führte maßgeblich zum Entwurf der Polsterung der Fußstützen (siehe Abschnitt 2.1.1).

Dieser Test verdeutlicht, dass es unabdingbar ist, das Fahrverhalten auf den Gebrauch im Museum zu optimieren. Abschnitt 4.4.2 beschreibt die genaue Auswertung dieses Praxistests.

### 4.4.2 Auswertung des Praxistests

Die Tatsache, dass 44,4% der Testpersonen Probleme hatten, intuitiv zu erkennen, dass es nach einer Vollbremsung gefordert ist, den Joystick in die Nullstellung zu bringen, also loszulassen, bestätigte die Vermutung, dass dieses Verhalten nicht der Benutzerfreundlichkeit dient. Des Weiteren ist zu vermuten, dass bei den 55,6% der Testpersonen, die keine Probleme mit diesem Verhalten hatten, bei vielen das Loslassen des Joysticks durch Zufall geschah oder bei ihrem Test keine Vollbremsung erfolgte. Dieses Verhalten musste demnach geändert werden. Ebenso musste die Ursache, die dieses Verhalten notwendig machte, behoben werden.

Das Problem des Festfahrens, das ebenfalls 44,4% der Testpersonen Probleme bereitete, wurde , wie in Abschnitt 4.2 beschrieben, darauf zurückgeführt, dass ein zu starkes Ausschwenken des Rollstuhls von der Software vermutet wurde. Wie vermutet, war es dieses Problem, das die Benutzer am meisten irritierte. Wenn nach dem Versuch einer frontalen Kollision und einer relativ langen Pause mit Loslassen des Joysticks die Rückwärtsbewegung verhindert wurde, vermuteten einige Testpersonen sogar, sie hätten etwas kaputtgemacht. Dieses Problem stellte sich jedoch später als noch komplexer dar. Die Abschnitte 4.5.1 und 4.5.5 befassen sich intensiver mit diesem Problem.

In den Fällen, in denen sich der HNF-Rolland auf freier Fläche festgefahren hat, ist davon auszugehen, dass sich Personen im Fahrweg des Rollstuhls befunden haben und eine Vollbremsung auslösten, die nicht durch die Nullstellung des Joysticks aufgehoben wurde. Nach der Änderung des Fahrverhaltens (Abschnitt 4.5), bei dem auf dieses spezielle Problem nicht eingegangen wurde, ließ sich dieser Fehler nicht mehr reproduzieren.

Die Rotationssperre, bei der Rotationen ohne ersichtliches Hindernis verhindert wurden, wurde darauf zurückgeführt, dass der Blindbereich für die vorderen Lenkräder zu klein war. Dieser Blindbereich ist definiert, damit die Lenkräder vom vorderen Laserscanner nicht als Hindernisse erkannt werden. Bei ungünstigem Stand ragten diese jedoch aus dem Blindbereich heraus. Nach einer Vergrößerung dieses Blindbereichs trat das Problem nicht mehr auf.

Die eine Rotationskollision eröffnete ein neues Problem. Zwar kam es nur zu einer Kollision, es hatten jedoch auch nur fünf Personen versucht, eine Kollision in der Rotationsbewegung zu forcieren. Dieses Problem bestand beim Rolland III nicht und ist daher auf die weichen Fahreinstellungen zurückzuführen. Bei einer genaueren Analyse zeigte sich, dass die weichen Fahrwerte zwar eine starke Beschleunigung in die Rotation zulassen, das Abbremsen jedoch sehr weich erfolgt. Dadurch wird, bei einer Rotation mit vollem Joystickausschlag, mindestens eine sechstel Drehung durchgeführt, auch wenn der Computer direkt

nach der Beschleunigung den Befehl zum Bremsen gibt. Um dieses Problem zu beheben, wurden die Werte für die Drehbewegung, die der Computer maximal sendet, stark reduziert.

Die Kollisionen mit beweglichen Zielen, die in 57.1% der versuchten Fälle auftraten, waren allesamt darauf zurückzuführen, dass Personen sich in den Bremsweg des Rollstuhls bewegten. Der Rollstuhl bremste zwar, die Personen standen jedoch schon zu nah. Dieses Problem ließ sich nicht durch Softwareänderungen lösen. Um den Schaden bei solchen Kollisionen möglichst gering zu halten, wurden die Fußstützen speziell gepolstert. Aus anderen Gründen wurde später auch noch die Höchstgeschwindigkeit reduziert, wie in Abschnitt 4.5.6 beschrieben wird. Seit der Senkung der Höchstgeschwindigkeit und der Polsterung der Fußstützen können Gefahren in dieser Situation ausgeschlossen werden.

## 4.5 Änderungen des Fahrverhaltens nach dem Praxistest

Wie in den vorangegangenen Abschnitten deutlich wurde, hängen die Probleme des Festfahrens, des Heranruckelns an Hindernisse, der Vollbremsung bis zum Stillstand und der geforderten Nullstellung des Joysticks zum Abbruch einer Vollbremsung eng miteinander zusammen. Die zwei letzten Verhalten, das Abbremsen bis zum Stillstand und die Nullstellung des Joysticks, dienen einer gleichmäßigen Bremsbewegung und verhindern stotternde Phasen in dieser. Besonders die stotternde Phase, die nach einer Vollbremsung mehrere ruckartige Bewegungen in Richtung des Hindernisses ermöglichen würde, birgt beim HNF-Rolland, wie im Abschnitt 4.2 beschrieben, die Gefahr von Kollisionen und führt unweigerlich zum Festfahren. Das Problem des Festfahrens bedarf jedoch noch einer besonderen Behandlung.

### 4.5.1 Das Festfahrproblem

Wie in dem Abschnitt 4.3 erwähnt, war zunächst der einzige Lösungsansatz für das Problem des Festfahrens, die virtuellen Sensoren zu ändern. In Abschnitt 4.2 wird beschrieben, dass diese nicht nur in Fahrtrichtung nach Hindernissen suchen, sondern auch ein Ausschwenken auf der zur Bewegung hinteren Seite kontrollieren. Außerdem wird die Fläche innerhalb der Rollstuhlausmaße auf erkannte, oder noch im Gedächtnis der Kollisionskarte befindliche Hindernisse durchsucht. Es wurde daher vermutet, dass die Ausschwenkbewegung als größer erachtet würde als diese tatsächlich ist.

Zur Lösung wurden die virtuellen Sensoren für das Ausschwenken gründlich überprüft und auf ein Minimum reduziert. In der Tat waren diese zuvor größer

als sie für ein sicheres Fahrverhalten sein mussten.

Außerdem wurde es als notwendig erachtet, dass sich der Rollstuhl von einem Hindernis, das sich innerhalb der Ausmaße des Rollstuhls befindet, wegbewegen kann. Wie schon in Abschnitt 4.2 beschrieben, sind die von der Software intern verwendeten Ausmaße des Rollstuhls etwas größer als diese tatsächlich sind. Stellt nun eine Person seinen Fuß oder einen anderen Gegenstand sehr dicht an den Rollstuhl heran, so soll es möglich sein, sich von diesem wegzubewegen.

Zur Lösung wurden die virtuellen Sensoren so verändert, dass sie nur die Hälfte der inneren Ausmaße, die sich in der Richtung der Fahrtbewegung befindet, auf Hindernisse durchsucht.

In Verbindung mit den Änderungen des Bremsverhaltens, die in den Abschnitten 4.5.2 und 4.5.3 beschrieben werden, konnten zunächst keine Festfahrprobleme mehr erkannt werden. Dieses Problem trat jedoch vor der Auslieferung des HNF-Rollands an das Heinz Nixdorf MuseumsForum noch einmal in Erscheinung. Abschnitt 4.5.5 beschreibt dieses aufgetretene Phänomen und dessen Lösung.

### 4.5.2 Das Bremsverhalten

Wie in Abschnitt 4.3 erläutert, ist es beim HNF-Rolland kein Problem, mit der Aufgabenstellung stark in die Geschwindigkeit des Rollstuhls einzugreifen. Dies ist notwendig, um bei den weichen Fahrwerten rechtzeitig bremsen zu können. Im Nahbereich von Hindernissen wird also nur eine sehr geringe Geschwindigkeit erlaubt, bei einer schnellen Fahrt auf Hindernisse zu wird schon sehr früh, und dafür sanft, die Geschwindigkeit reduziert. Die in Abschnitt 4.2 erwähnten Grenzen Softbrake und Hardbrake wurden dafür zunächst um das 1,8-Fache nach unten modifiziert. Als, wie in den Abschnitten 1.2.4 und 2.1 bereits erwähnt und im Abschnitt 4.5.6 genauer beschrieben, die Höchstgeschwindigkeit des HNF-Rollands stark reduziert wurde, stellten sich die Werte der 'SAMSafetyLayer' für die Hardbrake- und Softbrakegrenze wieder als vorteilhafter heraus. Dies wird darauf zurückgeführt, dass nun die maximal befohlene Geschwindigkeit klein genug ist, so dass, trotz der weichen Fahrwerte, Bremsbefehle auch in der Beschleunigungsphase schnell vom Rollstuhl umgesetzt werden. Die Berechnungen für die Softbrakegrenze und die Hardbrakegrenze lauten:

$$v_{soft} = \frac{v_{maxSafe}}{3} \tag{1}$$

$$v_{hard} = \frac{v_{maxSafe}}{1,8} \tag{2}$$

- $v_{soft}$ - Softbrakegrenze
- $v_{hard}$ - Hardbrakegrenze
- $v_{maxSafe}$ - aus den Kollisionstabellen erhaltene maximale sichere Geschwindigkeit

Beide Varianten, die Herabsetzung der Grenzen und die Beschränkung der Höchstgeschwindigkeit, ermöglichen, dass nicht mehr gefordert werden muss, eine Vollbremsung, die beim Überschreiten der Hardbrakegrenze ausgelöst wird, bis zum Stillstand durchzuführen.

Auch im Bereich zwischen der Softbrake- und der Hardbrakegrenze wird stärker in die Geschwindigkeit eingegriffen, also wenn die Ungleichung gilt: $v_{hard} \geq v_{real} > v_{soft}$. Der aktuelle Fahrbefehl wird entsprechend stärker modifiziert, wenn die reale Geschwindigkeit nah an der Hardbrakegrenze liegt. Es wird schon für den Fall, dass $v_{hard} = v_{real}$ ist, in der Softbrakephase eine Vollbremsung eingeleitet. Wird die Softbrakegrenze nur sehr leicht überschritten, wird als Fahrbefehl fast die Geschwindigkeit der Softbrakegrenze erlaubt.

Die zunächst eingesetzte Formel lautet:

$$v_{request} = v_{joystick} \frac{v_{hard} - v_{real}}{v_{hard} - v_{soft}} \tag{3}$$

- $v_{request}$ - nächster Geschwindigkeitsbefehl
- $v_{joystick}$ - durch die Joystickstellung geforderte Geschwindigkeit
- $v_{real}$ - aktuell von den Sensoren gemessene Geschwindigkeit

Aufgrund der weichen Fahrwerte war es nötig, noch stärker in die Geschwindigkeit einzugreifen je näher die aktuelle Geschwindigkeit an die Hardbrakegrenze herankam. Die eben genannte Formel führte zu einer stotternden Bewegung im Übergang zwischen Softbrake- und Hardbrakegrenze. Die daraufhin eingesetzte Formel lautet:

$$v_{request} = v_{joystick} \frac{v_{hard} - v_{real}}{v_{hard} - v_{soft}} \times \frac{v_{soft}}{v_{real}} \tag{4}$$

Im Nahbereich müssen die weichen Fahrwerte noch stärker berücksichtigt werden. Dafür fließt die Stellung des Joysticks, und damit die vom Benutzer

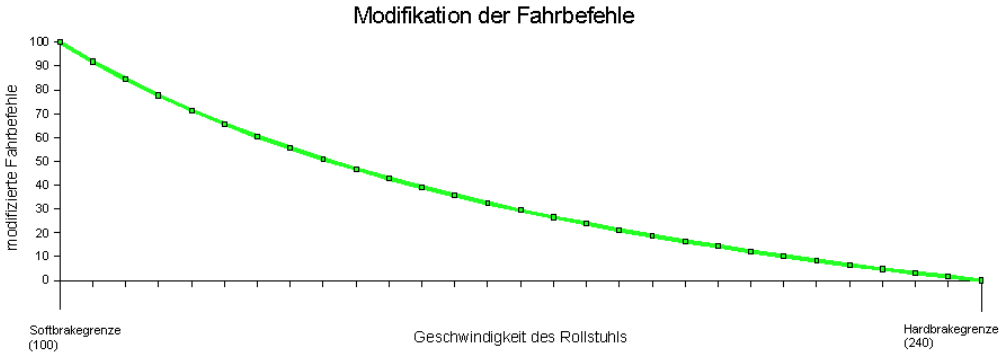



Abbildung 15: Modifikation der Fahrbefehle im Bezug auf die reale Geschwindigkeit alle Angaben sind in $\frac{cm}{sec}$

geforderte Geschwindigkeit, noch stärker in diese Berechnung ein. Die Formel lautet dann:

$$v_{request} = v_{joystick} \frac{v_{hard} - v_{real}}{v_{hard} - v_{soft}} \times \min[\frac{v_{soft}}{v_{real}}, \frac{v_{soft}}{v_{joystick}}] \quad (5)$$

Abbildung 15 zeigt, wie durch diese Formel stärker in die Fahrbefehle eingegriffen wird je näher die Geschwindigkeit des Rollstuhls an der Hardbrakegrenze liegt.

Das über die Formel 5 erreichte Verhalten realisiert ein relativ weiches, gleichmäßiges Bremsverhalten. Der Rollstuhl wird sehr langsam, wenn er sich einem Hindernis nähert. Durch die langsame Geschwindigkeit vor einem Hindernis ist es möglich, trotz der weichen Fahrwerte, eine Bremsung zu realisieren, die erst sehr nah an einem Hindernis endet. Leichte Unregelmäßigkeiten lassen sich, wegen der weichen Fahreinstellungen, nicht völlig vermeiden, werden so aber weitgehend minimiert.

### 4.5.3 Das Problem des Heranruckelns an Hindernisse

In die Formel 5 des Abschnitts 4.5.2 fließt die Stellung des Joysticks mit ein. Ist die von der Joystickstellung geforderte Geschwindigkeit größer als die tatsächliche, wird diese für die Berechnung genutzt. Ähnlich wird auch das Problem des an Hindernisse Heranruckelns gelöst. Dafür wird der in Abschnitt 4.4 beschriebene 'Trick' weiterentwickelt. Um beim Praxistest das Verhalten der 'SamSa-

fetyLayer' nutzen zu können, ohne dass sich der Rollstuhl immerzu festfährt, wurde dem Rollstuhl im Stand eine Geschwindigkeit von 30 $\frac{cm}{sec}$ simuliert. Diese Geschwindigkeit wurde für die Berechnung der maximal sicheren Geschwindigkeit verwendet und mit dieser verglichen.

Da dieser 'Trick' jedoch erst eingesetzt wurde, als der Rollstuhl schon das erste Mal zum Stehen kam und nur eine gerade Bewegung nach vorne oder hinten, ohne Berücksichtigung von Lenkbewegungen, simulierte, traten Probleme auf:

Das erste Problem bestand darin, dass es keinen fließenden Übergang zwischen einer leichten Bewegung des Rollstuhls und dem Stillstand des Rollstuhls gab. Dies führte dazu, dass, wenn sich der Rollstuhl, trotz der simulierten Geschwindigkeit, noch minimal in Richtung des Hindernisses bewegen durfte, die Bewegungen erlaubt wurden. Sobald sich der Rollstuhl bewegte, wurden bei der Berechnung der maximal sicheren Geschwindigkeit wieder die echten Geschwindigkeiten, die die Sensoren ermittelten, zugrunde gelegt. Da diese ermittelte Geschwindigkeit natürlich zunächst geringer als die simulierte war, konnte sich der Rollstuhl ein deutliches Stück dem Hindernis nähern. Hätte der Rollstuhl von vornherein nur einen halben Zentimeter dichter am Hindernis gestanden, hätte die simulierte Geschwindigkeit eine Bewegung unterbunden. Dadurch entstanden Differenzen, wie dicht man an ein Hindernis heranfahren konnte. Diese lagen zwischen 5 und 15 Zentimetern.

Zur Lösung dieses Problems wurde eine Geschwindigkeitsgrenze definiert. Unterschreitet die von den Sensoren gemessene Geschwindigkeit diese Grenze, wird die von dem Joystick geforderte Geschwindigkeit anteilig auf die echte Geschwindigkeit addiert, so dass die simulierte Geschwindigkeit genau der Geschwindigkeitsgrenze entspricht. Dadurch kann die für die Berechnung der maximal sicheren Geschwindigkeit verwendete Geschwindigkeit niemals die definierte Grenze unterschreiten. Für die Berechnung der maximal sicheren Geschwindigkeit wird also die reale Geschwindigkeit verwendet, solange diese oberhalb der definierten Geschwindigkeitsgrenze liegt. Liegt die reale Geschwindigkeit unterhalb der Grenze, wird die simulierte Geschwindigkeit verwendet.

Durch diese Erweiterung des 'Tricks' aus dem Praxistest ergeben sich kaum noch Differenzen, wie nah man an ein Hindernis heranfahren kann.

Das zweite Problem, das sich bei dem 'Trick', der im Praxistest eingesetzt wurde, zeigt, ergibt sich aus der Tatsache, dass die Lenkbewegung in der Simulation nicht berücksichtigt wird. Es werden nur gerade Vorwärts- und Rück-

wärtsbewegungen simuliert. Da es möglich ist, dass der Rollstuhl nicht kollisionsfrei gerade nach vorne fahren kann, aber eine Bewegung nach vorne links möglich ist, kann es passieren, dass so Bewegungen unterbunden werden, die eigentlich kollisionsfrei realisierbar wären.

Schlimmer ist der umgekehrte Fall, in dem es sein kann, dass eine Bewegung gerade nach vorne möglich ist, nicht aber eine geforderte Bewegung nach vorne rechts. Da nur gerade Bewegungen simuliert werden, würde die Rechtskurve, auf Grund der falschen Simulation, zugelassen. Normalerweise sollte diese sofort unterbunden werden, sobald sich der Rollstuhl bewegt und nicht mehr die simulierte Geschwindigkeit zur Berechnung der maximal sicheren Geschwindigkeit verwendet wird. Aufgrund der weichen Fahrwerte des HNF-Rollands sind in diesem Fall jedoch Kollisionen möglich.

Zur Lösung des Problems müssen also die Lenkbefehle des Joysticks mit in die Simulation einfließen. Um dies zu erläutern, muss zunächst klargestellt werden, wie Lenkbewegungen dargestellt werden. Bisher wurden in allen Formeln nur Geschwindigkeiten, nicht jedoch Lenkbewegungen beschrieben. Dies erfolgte der Einfachheit halber. Da der Meyra Champ über einen Differenzialantrieb verfügt, ergibt sich die Lenkbewegung aus der Differenz des linken und des rechten Antriebsrades. Demzufolge gibt es eine rechte und eine linke Geschwindigkeit. Geschwindigkeiten werden daher in der Kollisionsvermeidung immer als Vektoren aus rechter und linker Geschwindigkeit dargestellt. Wann immer Geschwindigkeiten vorkommen, bei denen rechte und linke Geschwindigkeiten nicht erwähnt werden, ist die 1-Norm dieses Vektors gemeint. Die bisher in den Formeln verwendete Geschwindigkeit ergibt sich demnach wie folgt:

$$v_{real} = \left|v_{leftreal}\right| + \left|v_{rightreal}\right| \tag{6}$$

- $v_{leftreal}$ - Geschwindigkeit des linkes Antriebsrades, ermittelt durch einen Odometriesensor
- $v_{rightreal}$ - Geschwindigkeit des rechten Antriebsrades, ermittelt durch einen Odometriesensor

Die anderen Geschwindigkeiten ergeben sich entsprechend aus der Summe der Beträge der Geschwindigkeiten des rechten und des linken Rades, beziehungsweise repräsentieren diese. In den Berechnungen wird eine gerade Bewegung also mit doppelter Geschwindigkeit dargestellt, da die Geschwindigkeiten des rechten und linken Rades addiert werden.

Um die Stellung des Joysticks in diese Rechnung aufzunehmen, muss dessen Stellung in diese Form umgerechnet werden. Trotz des Differenzialantriebs des Meyra Champ besteht das Joysticksignal aus einem Wert für die Geschwindigkeit und einem für die Lenkung. Diese Werte liegen im Signal zwischen -63 und +63. Die Umrechnung in Werte, die denen vom rechten und linken Rad entsprechen, geschieht folgendermaßen:

$$v_{left\,joystick} = scale\left(Sig_{joystickspeed} - Sig_{joysticksteering}\right) \tag{7}$$

$$v_{right\,joystick} = scale\left(Sig_{joystickspeed} + Sig_{joysticksteering}\right) \tag{8}$$

- $v_{left\,joystick}$ - Geschwindigkeit für das linke Antriebsrad, die vom Joystick gefordert wird
- $v_{right\,joystick}$ - Geschwindigkeit für das rechte Antriebsrad, die vom Joystick gefordert wird
- scale - Umrechnungskonstante, um die Joystickwerte in $\frac{mm}{s}$ umzurechnen
- $Sig_{joystickspeed}$- Signalwert des Joysticks für die Geschwindigkeit
- $Sig_{joysticksteering}$ - Signalwert des Joysticks für die Lenkung

Die Berechnung der simulierten Geschwindigkeit wird nun komplexer. Es müssen nun zwei Geschwindigkeiten, für das rechte und das linke Rad, simuliert werden. Die 1-Norm des Vektors der simulierten Geschwindigkeit soll weiterhin der Geschwindigkeitsgrenze entsprechen. Zunächst wird ermittelt welchen Anteil an der simulierten Geschwindigkeit die reale Geschwindigkeit hat und welchen die Geschwindigkeit des Joysticks.

$$part_{real} = \frac{v_{real}}{v_{limit}} \tag{9}$$

$$part_{joystick} = 1 - \frac{v_{real}}{v_{limit}} \tag{10}$$

- $part_{real}$ - Anteil der realen Geschwindigkeit an der simulierten
- $part_{joystick}$ - Anteil der vom Joystick geforderten Geschwindigkeit an der simulierten
- $v_{limit}$ - die Geschwindigkeitsgrenze, deren Unterschreitung zur Simulation der Geschwindigkeit führt
- $v_{real}$ - 1-Norm des Geschwindigkeitsvektors, der aktuell von den Sensoren gemessen wird

Um die 1-Norm des Vektors der simulierten Geschwindigkeit auf dem Wert der Geschwindigkeitsgrenze zu halten, wird die vom Joystick geforderte Geschwindigkeit auf die Geschwindigkeitsgrenze skaliert. Dazu wird ein Faktor berechnet, mit dem sowohl die rechte als auch die linke Geschwindigkeit, die vom Joystick gefordert ist, multipliziert wird.

$$scale_{joystick} = \frac{v_{limit}}{v_{joystick}} \tag{11}$$

- $scale_{joystick}$ - Faktor mit dem die 1-Norm des vom Joystick geforderten Geschwindigkeitsvektors auf $v_{limit}$ skaliert wird
- $v_{Joystick}$ - 1-Norm des Geschwindigkeitsvektors, der vom Joystick gefordert wird

Die Berechnungen der simulierten Geschwindigkeiten für rechtes und linkes Rad ergeben sich damit wie folgt:

$$v_{leftsimulated} = v_{leftreal} + part_{joystick} \times scale_{joystick} \times v_{left\,joystick} \tag{12}$$

$$v_{rightsimulated} = v_{rightreal} + part_{joystick} \times scale_{joystick} \times v_{right\,joystick} \tag{13}$$

- $v_{leftsimulated}$ - für die Berechnung der maximal sicheren Geschwindigkeit eingesetzte, simulierte Geschwindigkeit für das linke Rad
- $v_{rightsimulated}$ - für die Berechnung der maximal sicheren Geschwindigkeit eingesetzte, simulierte Geschwindigkeit für das rechte Rad

Die Berechnung der simulierten Geschwindigkeit nach dieser Formel und deren Einsatz zur Berechnung der maximal sicheren Geschwindigkeit bei Unterschreitung der definierten Geschwindigkeitsgrenze $v_{limit}$ führen zu einem Verhalten, in dem der Rollstuhl sehr nah an Hindernisse heranfahren kann, ohne sich festzufahren. Die Bewegung erfolgt nahezu gleichmäßig und fast ohne eine Phase einer stotternden Bremsung. Leichte Unregelmäßigkeiten in der Bewegung lassen sich aufgrund der weichen Fahreinstellungen des HNF-Rollands nicht völlig vermeiden. In der Bremsbewegung gibt es in der Regel kurz vor dem Hindernis einen Punkt, an dem die Geschwindigkeit stark reduziert wird, bevor der Rollstuhl dann konstant auf den minimalen Abstand zum Hindernis fährt. Eindeutiger Beleg dafür, dass dieses Verhalten an den weichen Fahrwerten liegt, ist, dass dieses Verhalten beim Rolland III nicht auftritt, wenn man ihn mit diesem Fahrverhalten betreibt.

Der Einsatz der Formeln 12 und 13 ermöglicht es, die Bewegungen zu reproduzieren, es besteht also keine ersichtliche Differenz zwischen den Abständen, in denen der Rollstuhl vor einem Hindernis anhält, wenn zweimal auf das gleiche Hindernis zugefahren wird. Aus dem Stand und bei geringen Geschwindigkeiten werden sofort die Bereiche auf Kollisionsfreiheit geprüft, die in Richtung des Joystickausschlages liegen. Dadurch können die zuvor beschriebenen Probleme nicht mehr auftreten. Wird der Rollstuhl durch die Software vor einem

Hindernis zum Stehen gebracht, ist keine Bewegung mehr in Richtung des Hindernisses möglich.

In Verbindung mit den, in Abschnitt 4.5.2 beschriebenen, Änderungen des Bremsverhaltens ist es möglich, auf einen Abstand von unter zwei Zentimetern an ein Hindernis heranzufahren. Dabei entsteht weder die Gefahr von Kollisionen noch tritt das Problem des Festfahrens auf. Für Letzteres sind auch die, in Abschnitt 4.5.1 beschriebenen, Änderungen der virtuellen Sensoren verantwortlich.

### 4.5.4 Die Probleme in der Rotationsbewegung

Wie in Abschnitt 4.4.1 bereits erwähnt wurde, lässt sich das Festfahren in der Rotationsbewegung auf freier Fläche darauf zurückführen, dass der Blindbereich, in dem sich die vorderen Lenkräder befinden, nicht ausreichend groß war. Bei ungünstigem Stand der Lenkräder wurden diese von den Laserscannern als Hindernisse innerhalb der Ausmaße des Rollstuhls erkannt und eine Vollbremsung eingeleitet.

Um dieses Problem zu lösen, wurden die Blindbereiche etwas vergrößert, woraufhin dieses Problem nicht mehr auftrat.

Ein weiteres Problem, das in Abschnitt 4.4.1 beschrieben wird, ist das der Rotationskollisionen. Diese finden statt, da die weichen Fahrwerte des HNF-Rollands in der Rotation deutlich stärkere Beschleunigungen zulassen als bei geraden Fahrbewegungen. Wird eine starke Beschleunigung befohlen und, durch die Erkennung eines Hindernisses, sofort eine Vollbremsung eingeleitet, macht der Rollstuhl in jedem Fall mindestens eine sechstel Drehung, bevor er zum Stehen kommt.

Es war demnach dringend erforderlich, die Rotationsbewegung deutlich langsamer und weicher zu gestalten. Ein zweiter Punkt machte dies ebenfalls erforderlich: Da der Rollstuhl sich bei der Rotationsbewegung immer in tote Winkel, also unbekannte Bereiche, bewegt und für diesen Fall eine Maximalgeschwindigkeit für die maximal sichere Geschwindigkeit vorgesehen ist, verhält sich die Geschwindigkeit in der Rotation immer wellenartig um diese maximale Geschwindigkeit für unbekannte Bereiche. Es war demnach notwendig, dass die Geschwindigkeit in der Rotation immer unterhalb dieser maximalen Geschwindigkeit bleibt, um ein gleichmäßiges Fahrgefühl zu gewährleisten.

Dieses Verhalten wird realisiert, indem die Fahrbefehle für die Lenkung begrenzt werden. Es genügte nicht eine maximale Grenze für die Lenkbefehle zu definieren, da aus dem Stand der Rollwiderstand überwunden werden muss. Dazu ist ein stärkerer Lenkbefehl nötig als später, um die Geschwindigkeit konstant zu halten. Der Rollwiderstand kann sich erheblich vergrößern, wenn die Lenkrä-

der in einer ungünstigen Position stehen.

Die Werte des Lenksignals, wie auch die des Geschwindigkeitssignals, liegen zwischen -63 und 63. Mit steigender Geschwindigkeit muss, wie erwähnt, ein kleineres Signal gesendet werden als im Stand. Im Stand muss es, für den Fall von ungünstig stehenden Lenkrädern, möglich sein, eine größere Leistung der Motoren zu fordern als wenn sich der Rollstuhl bereits bewegt.

In der Realisierung wird, abhängig von der Rotationsgeschwindigkeit, eine maximale Signalstärke berechnet. Der Betrag der vom Joystick geforderten Signalstärke für die Rotation wird damit verglichen. Ist der Befehl des Joysticks größer als die maximale Signalstärke, wird anstelle des Joystickbefehls die maximale Signalstärke gesendet. Fordert der Joystick eine negative Zahl als Signal, wird die maximale Signalstärke vor dem Senden selbstverständlich negiert.

$$Sig_{maxsteering} = 20 + \left( \frac{400 - v_{rotation}}{400} \right) \times 10 \tag{14}$$

$$v_{rotation} = \frac{\left| v_{leftreal} - v_{rightreal} \right|}{2} \tag{15}$$

- $Sig_{maxsteering}$ - maximale Signalstärke für die Lenkung
- $v_{rotation}$ - Rotationsgeschwindigkeit

Durch den Einsatz dieser Formel wird die Bewegung in der Rotation deutlich sanfter und gleichmäßiger. Kollisionen können mit demselben Algorithmus wie in jeder anderen Bewegung vermieden werden.

### 4.5.5 Die letzten Änderungen vor der Auslieferung. Das Festfahrproblem II

Die bisher in Abschnitt 4.5 beschriebene Fahrverhalten löst alle bisher erkannten Probleme. Die Bremsung erfolgt weich, es ist möglich, sehr nah und konstant reproduzierbar an Hindernisse heranzufahren und der Rollstuhl fährt sich nicht fest. Bis zu diesem Zeitpunkt wurde das Fahrverhalten jedoch nur von Personen, die in die Entwicklung des Projektes 'Bremer autonomer Rollstuhl' eingebunden sind, getestet. Das hat den Nachteil, dass diese Personen sehr gut mit der Steuerung des Rollstuhls vertraut sind. Sie kennen die Algorithmen und wissen, auf welche Feinheiten zu achten ist. Dies geschieht bei diesen Personen automatisch, so dass sie nicht in der Lage sind, Probleme zu erkennen, die unerfahrene Benutzer mit dem Rollstuhl haben.

Zwei Tage vor der Auslieferung des HNF-Rollands an die Ausstellung 'Computer.Medizin' wurden Personen zu Tests gebeten, die mit dem Steuern eines

Rollstuhl völlig unvertraut sind. Sie sollten Fehler erkennen, die von Mitarbeitern des Projekts 'Bremer autonomer Rollstuhl' nicht erkannt werden konnten.

Eine Fahrt quer über den Campus der Universität Bremen, die durch Menschenmengen, enge Gänge und enge Fahrstühle führte, verlief ohne Kollisionen. Danach wurden Tests in einem Bereich, der nur etwas größer ist als die Testfläche in der Ausstellung 'Computer.Medizin', durchgeführt. Zum Erstaunen der Entwickler trat nun das Problem des Festfahrens, das schon als gelöst galt, wieder auf.

Nach dem Versuch einer frontalen Kollision mit einer Wand, die erfolgreich verhindert wurde, war es den Testpersonen unmöglich, den Rollstuhl rückwärts von der Wand wegzubewegen.

Die Fehleranalyse stellte sich als schwierig heraus, da die Entwickler diesen Fehler zunächst nicht reproduzieren konnten.

Nach einiger Zeit wurde deutlich, dass dieses Problem auf die Ausschwenkbewegung des Rollstuhls zurückzuführen ist. Wie bereits erwähnt, sind die Ausmaße des Rollstuhls, von denen die Software ausgeht, etwas größer als der Rollstuhl selbst. An der vorderen Seite der Fußstützen sind diese deutlich größer. Dies geht darauf zurück, dass auch die Fußspitzen des Benutzers vor Kollisionen geschützt werden sollen. Da der HNF-Rolland keine Sensoren zur Erkennung der Fußspitzen hat, muss von sehr großen Füßen ausgegangen werden, damit auch Benutzer mit der Schuhgröße 47 keine Kollisionen zwischen Hindernissen und ihren Füßen erleben. Wird nun von der Software erkannt, dass dieser Bereich, in dem die Fußspitzen vermutet werden, dicht an einer Wand ist, darf der Rollstuhl nur eine gerade Bewegung rückwärts machen. Jede Bewegung, die dazu führt, dass der Bereich der Fußstützen und der Fußspitzen ausschwenkt, führt in diesem Fall zu einer möglichen Kollision mit den Fußspitzen. Gerade für unerfahrene Benutzer ist es fast unmöglich, eine genaue und gerade Bewegung zu befehlen. Der analoge Joystick sendet nur in einem Bereich von ca. einem Millimeter eine Null als Wert für die Lenkung. Besonders Benutzern mit kleineren Füßen ist es in diesem Fall nicht ersichtlich, warum eine Bewegung unterbunden wird.

Zur Lösung dieses Problems wurde beschlossen, den Bereich, in dem der Joystick eine Null als Wert für die Lenkung sendet, deutlich zu vergrößern. Dies hat zur Folge, dass, wenn der Benutzer den Joystick relativ gerade nach hinten oder nach vorne bewegt, eine gerade Bewegung ausgeführt wird. So ist es für den Benutzer leicht, eine Bewegung auszuführen, bei der die zur Bewegung hintere Seite nicht ausschwenkt. Zur Modifikation des Joysticksignals wird die Formel 16 eingesetzt. Sollte der Joystick negative Werte fordern, wird der Wert $Sig_{sendsteering}$ selbstverständlich negiert. Liegt der vom Joystick geforderte Wert für die Lenkung zwischen dem Grenzwert $Sig_{zerolimit}$ und dessen Negati-

on $-Sig_{zerolimit}$, so wird eine Null als Wert für die Lenkung an den Rollstuhl gesendet.

Zum Verständnis der Formel 16 bleibt zu erwähnen, dass die Signalwerte für die Lenkung, genau wie die für die Geschwindigkeit, zwischen -63 und 63 liegen.

$$Sig_{sendsteering} = Sig_{zerolimit} + \left|Sig_{joysticksteering}\right| \frac{63 - Sig_{zerolimit}}{63} \qquad (16)$$

- $Sig_{sendsteering}$ - Signalwert für die Lenkung, der als Befehl zum Rollstuhl gesendet wird
- $Sig_{zerolimit}$ - Grenzwert; liegt das Joysticksignal für die Lenkung im Intervall zwischen diesem Grenzwert und seiner Negation, wird Null als Wert für die Lenkung zum Rollstuhl gesendet

Durch diese Modifikation der Signalwerte haben selbst unerfahrene Benutzer nie das Gefühl, dass sich der Rollstuhl festgefahren hätte. Es ist ihnen leicht möglich, gerade Bewegungen zu befehlen, die ohne ungewolltes Ausschwenken ausgeführt werden.

### 4.5.6 Die Änderungen nach zwei Monaten Betrieb in der Ausstellung 'Computer.Medizin'

Nach knapp zwei Monaten des Betriebes in der Ausstellung 'Computer.Medizin' wurden weitere Probleme deutlich. Da dem Projekt 'Bremer autonomer Rollstuhl' bis zur Auslieferung der genaue Aufbau der Testumgebung nicht bekannt war, konnte erst nach der Startphase auf diese Probleme eingegangen werden.

Die Probleme waren zum einen, dass die Testfläche sehr kein ist und in ihr mehrere Hindernisse stehen (siehe Abbildung 16). So stellte sich heraus, dass der Rollstuhl sich fast immer in der Phase des Softbraking (siehe Abschnitte 4.2 und 4.5.2) befand. Nur in wenigen Situationen wurden die vollen Fahrbefehle ausgeführt. Dies bedeutete für den Benutzer, dass der Rollstuhl sich fast immer langsam bewegte, und in nur wenigen Ausnahmen stark beschleunigte.

Es wurde entschieden, dass es besser sei, die Maximalgeschwindigkeit zu reduzieren, so dass der Rollstuhl sich gleichmäßig bewegt, egal ob ein Hindernis in mittlerer Entfernung erkannt wird oder nicht. Es wurde eine ähnliche Formel zur Reduzierung der Maximalgeschwindigkeit verwendet wie zur Reduzierung der Rotationsgeschwindigkeit, die bereits in Abschnitt 4.5.4 beschrieben wurde. Es wird dasselbe Verhalten zur Überwindung des Rollwiderstandes benötigt wie bei der Rotationsgeschwindigkeit. Ebenso kann der Rollwiderstand sich stark vergrößern, wenn die vorderen Lenkräder ungünstig stehen. Es ist also nötig, aus dem Stand einen größeren Fahrbefehl zu senden, mit zunehmender

Abbildung 16: Die Testumgebung für den HNF-Rolland in der Ausstellung 'Computer.Medizin'

Geschwindigkeit sind kleinere Fahranweisungen erforderlich. Wie bei der Rotationsgeschwindigkeit wird eine maximale Fahranweisung berechnet, in diesem Fall für die Geschwindigkeit. Diese hängt von der realen Geschwindigkeit des Rollstuhls ab, die von den Sensoren gemessen wird. Überschreitet die Fahranweisung diesen Wert, so wird er durch den der maximalen Fahranweisung ersetzt. Sollte der Joystick einen negativen Fahrbefehl senden, so wird die maximale Fahranweisung selbstverständlich negiert. Hier wird nur die Berechnung der maximalen Fahranweisung aufgezeigt.

$$Sig_{maxspeed} = 30 + \left( \frac{400 - |v_{real}|}{400} \right) \times 10 \tag{17}$$

- $Sig_{maxspeed}$ - maximale Signalstärke für die Geschwindigkeit

Durch den Einsatz dieser Formel wird die Maximalgeschwindigkeit des Rollstuhls auf ca. 2 $\frac{km}{h}$ reduziert. Dies ist ein guter Wert für die vorhandene sehr kleine Testumgebung, in der viele Hindernisse stehen.

Das zweite Problem, das sich nach dem Einsatz in der Ausstellung darstellte, war etwas komplexer. Es resultierte daraus, dass in der Testumgebung keine Hindernisse mit senkrechten Kanten zum Einsatz kamen, sondern Verkehrspylonen (siehe Abbildung 17), die eine Kegelform besitzen. Das Problem besteht

Abbildung 17: Verkehrspylon als Hindernis in der Testumgebung

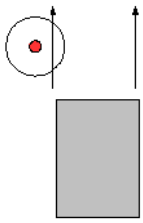


Abbildung 18: Vereinfachte Darstellung des Problems mit Pylonen
Roter Kreis: Erkanntes Hindernis, weißer Kreis: das echte Hindernis auf Bodenhöhe

darin, dass die Laserscanner Hindernisse nur auf einer Höhe von ca. 10 Zentimetern erkennen. Die Verkehrspylonen sind jedoch in 10 Zentimetern Höhe bereits schmaler als auf dem Boden. So hält der HNF-Rolland vor den erkannten Hindernissen an, befindet sich jedoch bereits in einer Kollision mit den tieferen Bereichen der Verkehrspylonen. Abbildung 18 veranschaulicht dieses Problem.

Es wurde nach einer Lösung gesucht, die es trotzdem ermöglicht, eng an Hindernissen vorbeifahren zu können. Es sollten zum Beispiel Türen, die unwesentlich breiter als die Ausmaße des Rollstuhls sind, weiter durchfahren werden können. Als jedoch keine adäquate Lösung gefunden werden konnte, wurde entschieden, die intern gespeicherten Ausmaße des Rollstuhls so zu vergrößern, dass der Abstand zwischen erkannten Hindernissen und dem Bodenbereich der Verkehrspylonen durch die größeren Ausmaße des Rollstuhls abgedeckt ist. Die intern gespeicherten Ausmaße wurden so um knapp zwei Zentimeter zu allen Seiten vergrößert.

Diese Lösung schränkt selbstverständlich die Bewegungsfreiheit des Rollstuhls ein. Wie bereits erwähnt, müssen zum Beispiel Türen, die gerade noch durchfahren werden können, nun vier Zentimeter breiter sein. Da auf der Testfläche, auf der der HNF-Rolland eingesetzt wird, jedoch keine Türen sondern Verkehrspylonen stehen und diese durch diese Lösung nicht mehr mit dem Rollstuhl kollidieren, ist der Ansatz für diesen speziellen Einsatzfall der bestmögliche.

Abbildung 19: Ein Museumsbesucher auf dem HNF-Rolland

# 5 Fazit

In diesem Abschnitt werden die Bewertungen der Museumsbesucher und des Kurators der Ausstellung 'Computer.Medizin' vorgestellt. Es erfolgt ein Ausblick auf weitere Aufgaben und Ziele des Projekts 'Bremer autonomer Rollstuhl. Außerdem wird das Gesamtsystem des HNF-Rollands zusammenfassend dargestellt.

## 5.1 Bewertung der Museumsbesucher und des Ausstellungskurators

Zu zwei Terminen wurden Besucher der Ausstellung zum HNF-Rolland befragt. Die erste Befragung erfolgte bei dem ersten Praxistest der Ausstellung 'Computer.Medizin' (siehe Abschnitt 4.4). Die zweite Befragung fand nach viereinhalb Monaten des Austellungsbetriebs statt. Das bei der ersten Befragung aktive Fahrverhalten war das der SAMSafetyLayer (siehe Abschnitt 4.2). Bei der zweiten Befragung war das Fahrverhalten nach den, in Abschnitt 4.5 beschriebenen, Änderungen aktiv.

Ebenfalls nach viereinhalb Monaten wurde der Kurator der Ausstellung 'Computer.Medizin' nach dem Wert des Exponats befragt.

### 5.1.1 Besucherbefragungen

Es wurden bei beiden Terminen jeweils 20 Personen befragt, hauptsächlich Schüler. Diese Gruppe ist sehr klein, dennoch stellen die Bewertungen einen

| Note | Anzahl |
|---|---|
| Eins | 2 |
| Zwei | 10 |
| Drei | 8 |
| Vier | 0 |
| Fünf | 0 |
| Durchschnitt | 2,3 |

Tabelle 4: (Erster Termin:) „Wie bewerten Sie das Exponat und seine Bedienbarkeit?“

Anhaltspunkt dar und zeigen Tendenzen in der Wirkung des Exponats auf die Besucher.

Die gestellten Fragen bei beiden Terminen sind nicht identisch, dies geht auf den unterschiedlichen Entwicklungsstand des Exponats zurück. Bei beiden Terminen wurden die Fragen 'Wie bewerten Sie das Exponat und seine Bedienbarkeit?' und 'Wie bewerten Sie den Wert eines Rollstuhls mit Kollisionsvermeidung?' gestellt. Die Befragten sollten diese Fragen jeweils mit einer Note zwischen eins und fünf beantworten, wobei eine 'eins' sehr gut und eine 'fünf' sehr schlecht ist. Darüber hinaus gehende Fragen werden im Folgenden bei den Auswertungen beschrieben.

Bei der ersten Befragung wurden hauptsächlich drei Fragen gestellt. Die beiden bereits oben erwähnten Fragen, sowie die Frage, ob sich die Besucher eine ausführlichere Anleitung zur Bedienung wünschten. Eine Frage nach Verbesserungsmöglichkeiten machte zu diesem Zeitpunkt noch wenig Sinn, da die Besucher wussten, dass das Fahrverhalten noch deutlich geändert werden musste. Die in Abschnitt 4.4.1 beschrieben Probleme, die die Besucher bei der Bedienung des Rollstuhls hatten, wurden von diesen auch angemerkt.

Auf die Frage, ob die Anleitung zur Bedienung ausreiche, antworteten 18 Personen, dass die Erklärungen ausreichend seinen. Die zwei Personen mit einer anderen Meinung fanden beide, dass auch weniger Anleitung ausreichend wäre. Aufgrund dieser Aussagen wurden die Anleitungen zur Bedienung nicht verändert.

Tabelle 4 und 5 zeigen die Auswertung der ersten Befragung beim Praxistest der Ausstellung.

Bei der zweiten Befragung wird die Frage nach Verbesserungsmöglichkeiten ergänzt. Da nun das Fahrverhalten bestmöglich verändert worden ist, stellt dies

| Note | Anzahl |
|---|---|
| Eins | 8 |
| Zwei | 12 |
| Drei | 0 |
| Vier | 0 |
| Fünf | 0 |
| Durchschnitt | 1,6 |

Tabelle 5: (Erster Termin:) „Wie bewerten Sie den Wert eines Rollstuhls mit Kollisionsvermeidung?“

| Verbesserungsvorschlag | Anzahl |
|---|---|
| Keine | 7 |
| Bremsung nicht konstant genug | 7 |
| Unterbrechungen in der Lenkbewegung | 1 |
| Rollstuhl sollte näherer an Hindernisse fahren | 2 |
| Bremsung später einleiten | 7 |
| Rollstuhl sollte schneller reagieren | 2 |

Tabelle 6: Verbesserungsvorschläge für den HNF-Rolland

einen Punkt zur Bewertung des Ergebnisses und möglicher anderer Ansätze bei weiteren Projekten mit ähnlichem Ziel dar.

Auch Personen, die das Exponat als sehr gut bewerteten, hatten noch Verbesserungsvorschläge. Tabelle 6 zeigt die Verbesserungswünsche und ihre Häufigkeit.

Das von zwei Besuchern geforderte schnellere Reagieren des Rollstuhls, beziehungsweise das von ihnen erlebte zu langsame Reagieren, sowie die, einmal aufgetretene, Unterbrechung in der Lenkbewegung, sind darauf zurückzuführen, dass die Personen nicht merkten, dass ihre Fahrbefehle unterbunden wurden, da eine Kollision mit dem hinteren Teil des Rollstuhls und den Verkehrspylonen in der Testumgebung drohte. Diese Aussagen tauchen hier der Vollständigkeit halber auf, haben jedoch ihre Ursache in der funktionierenden Kollisionsvermeidung.

Die Tatsache, dass der Rollstuhl nicht nahe genug an Hindernisse heran fährt, tritt nur an der vorderen Seite des Rollstuhls auf. Dies liegt daran, dass die intern gespeicherten Ausmaße des Rollstuhls nach vorne deutlich größer sind, um auch die Füße von großen Menschen vor Kollisionen zu schützen. Menschen, deren Füße nicht über die Fußstützen des Rollstuhl hinausragen, empfinden dies

| Note | Anzahl |
|---|---|
| Eins | 5 |
| Zwei | 12 |
| Drei | 3 |
| Vier | 0 |
| Fünf | 0 |
| Durchschnitt | 1,8 |

Tabelle 7: (Zweiter Termin:) „Wie bewerten Sie das Exponat und seine Bedienbarkeit?“

| Note | Anzahl |
|---|---|
| Eins | 7 |
| Zwei | 11 |
| Drei | 2 |
| Vier | 0 |
| Fünf | 0 |
| Durchschnitt | 1,75 |

Tabelle 8: (Zweiter Termin:) „Wie bewerten Sie den Wert eines Rollstuhls mit Kollisionsvermeidung?“

manchmal als störend. Für den HNF-Rolland ist dieses Verhalten jedoch gewollt, da es störender wäre, wenn einige Museumsgäste mit ihren Füßen an Hindernisse stießen.

Das frühe Einleiten von Bremsungen ist ebenfalls intendiert, um Museumsbesucher nicht dazu zu verleiten, mit dem Joystick selbst vor Hindernissen zu bremsen und um Besucher nicht durch starke Bremsungen zu erschrecken.

Das Problem der nicht konstanten Bremsung wurde, wie in Abschnitt 4.5 beschrieben, soweit gelöst wie es die weichen Fahreinstellungen des Meyra Champ beim HNF-Rolland zulassen. Für eine Weiterentwicklung des Systems, wie in Abschnitt 5.2 beschrieben wird, sollte dieser Punkt auf jeden Fall angegangen werden.

Tabelle 7 und 8 zeigen die Bewertungen der Besucher beim zweiten Termin. Bei der ersten Frage nach dem Exponat und der Bedienbarkeit ergibt sich eine Verbesserung um eine halbe Note zwischen der ersten Befragung und der nach viereinhalb Monaten Ausstellungsdauer. Schlechte Bewertungen sind die Ausnahme, beziehungsweise treten gar nicht auf, wenn die Befragten wissen, dass sie mit den Entwicklern des Exponats sprechen. Außerdem handelte es sich bei den Befragten zu beiden Zeitpunkten nicht um dieselben Personen. Demnach

gab es für die Befragten keine Vergleichswerte. Dennoch bleibt festzuhalten, dass die Bewertungen besser wurden, nachdem die wichtigsten Probleme beseitigt waren.

Die zweite Frage gilt dem generellen Nutzen eines Rollstuhls mit Kollisionsvermeidung. Es spielt hierbei keine Rolle, in welchem Zustand sich das Exponat befindet. Hier sieht man, dass die Bewertungen zwischen den beiden Terminen sich weniger unterscheiden: Der Unterschied liegt bei nur 0,15 Noten. Insgesamt gab es bessere Noten für den Zweck eines Rollstuhls mit Kollisionsvermeidung als für das Exponat selbst. Daran, und an den sehr guten Noten für die Idee des Rollands, erkennt man, dass der praktische Nutzen solcher Rollstühle gut erfasst und verstanden wird.

### 5.1.2 Bewertung des Kurators der Ausstellung 'Computer.Medizin'

Viereinhalb Monate nach Beginn der Ausstellung im Heinz Nixdorf MuseumsForum, zum Termin der zweiten Besucherbefragung, wurde auch der Kurator der Ausstellung um eine Stellungnahme zum HNF-Rolland gebeten.

Er nannte das Exponat eines der Highlights der Ausstellung. Es ist nach seinen Angaben eines der meist genutzten interaktiven Exponate der Ausstellung. Die Besucher nehmen es gerne an und interessieren sich für den Hintergrund.

Darüber hinaus sagte er, dass zahlreiche Besucher fragen, ob oder wann solche Rollstühle bei Menschen mit Behinderungen eingesetzt würden. Diese Besucher zeigten sich auch an der Weiterentwicklung interessiert und daran, wie lange diese voraussichtlich dauern wird.

Besonders hob der Kurator hervor, dass, nach anfänglichen Problemen, der HNF-Rolland nun zu den zuverlässigsten Exponaten der Ausstellung gehöre. 'Es ist eines der wenigen Exponate, die kaum Probleme bereiten.' berichtete er.

Nach Angaben des Kurators besuchen durchschnittlich 800 Besucher täglich die Ausstellung, am 'Tag der offenen Tür' waren es 4.000. Insgesamt hatten bis Ende März 2007 zirka 65.000 Besucher die Ausstellung besichtigt. Bis zum Ende der Ausstellung im Heinz Nixdorf MuseumsForum wird eine Gesamtzahl von 100.000 Besuchern erwartet. Alle Führungen für die letzten zwei Monate sind schon im Voraus ausgebucht.

Es ist geplant die Ausstellung noch in anderen Museen zu zeigen. Die nächste Station wird voraussichtlich Bochum sein.

## 5.2 Ausblick

Die nächste Aufgabe, der sich das Projekt 'Bremer autonomer Rollstuhl' stellen will, wird die Weiterentwicklung des Rollands III zur Serienreife sein. Dabei

soll das System besonders im Hinblick auf Kostenersparnis optimiert werden. Dabei soll weiterhin der Meyra Champ als Plattform zum Einsatz kommen.

Bei den Sensoren gibt es erhebliches Einsparpotential, es ist geplant, nur noch einen Laserscanner einzusetzen, den URG-04LX von Hokuyo [Hokuyo, 2005]. Dieser kann allein in einem 240° Winkel scannen. Seine Grundfläche beträgt $5cm \times 5cm$, seine Höhe $7cm$. Er soll unter dem Batteriekasten angebracht werden, so dass alle Seiten abgedeckt werden. Die Zahl der toten Winkel würde sich dadurch verändern, da nun auch die Antriebsräder Hindernisse bei den Scans darstellen. Dafür wäre, anders als beim Rolland III, der Bereich zwischen den Lenk- und den Antriebsrädern abgedeckt. Da der Laserscanner URG-04LX nur vier Meter weit entfernte Hindernisse sicher erkennt, und nicht, wie die rotoScan ROD-4, 40 Meter weit scannt, ist der Laserscanner URG-04LX deutlich billiger. Für die Absicherung des Rollstuhls ist ein Erkennungsbereich von vier Metern jedoch ausreichend.

Durch Wegfall der Odometriesensoren können weitere Kosten gespart werden. Die Erkennung der eigenen Bewegung soll anhand der erkannten Hindernisse und deren Verschiebung in den Scans geschehen.

Die Software soll soweit optimiert werden, dass es möglich ist, sie auf einem Microcontroller laufen zu lassen. Dadurch würde auch der Computer wegfallen.

Es wird weiteren Experimenten vorbehalten sein, zu klären wieweit diese Vorhaben zu realisieren sind. Dennoch würden sich auf diese Weise die Kosten, die über die Anschaffung des Meyra Champ hinausgehen, extrem verringern.

## 5.3 Einsetzbarkeit

Der HNF-Rolland wurde konzipiert um als Museumsexponat zu dienen. Dies ist kein typischer Einsatzbereich eines Rollstuhls. Demonstriert werden soll die Verwendbarkeit bei Menschen mit Behinderungen.

Angewendet werden können Rollstühle mit Kollisionsvermeidung von Menschen, deren Behinderungen klare und schnelle Fahrbefehle nicht zulassen und die zur ihrer Mobilität einen Rollstuhl benötigen, den sie nicht aus eigener Kraft fahren können.

Hauptsächlich sind hier Krankheiten, die ein chronisches Zittern der Hände zur Folge haben, zu nennen. Dies sind zum Beispiel Multiple Sklerose, die Parkinson-Krankheit und spastische Lähmungen.

Patienten, die unter diesen Krankheiten leiden, könnten mit dem Rolland selbständig durch Gebäude navigieren. Sie bräuchten zur Mobilität in Gebäuden keine Begleitung und vermieden das Risiko von Verletzungen oder Sachschäden.

Der Rolland III kann darüber hinaus mit einer Sprachsteuerung und einer Steuerung über die Neigung des Kopfes, sogar Menschen bei der Mobilität hel-

fen, die ihre Arme und Hände überhaupt nicht bewegen können. Damit könnten Menschen selbstständige Mobilität erhalten, die an starken Querschnittsyndromen leiden und sich ohne fremde Hilfe fast gar nicht bewegen können.

## 5.4 Zusammenfassung

Der HNF-Rolland stellt einen Nachbau des Rollands III dar, der konzipiert wurde, um als interaktives Museumsexponat eingesetzt zu werden.

Dazu wurde die Hardware weitest möglich auf Benutzerfreundlichkeit und Sicherheit optimiert. Es sind fast alle Bestandteile im Batteriekasten des Meyra Champ untergebracht. Die Laserscanner sind als einzige Hardware von außen sichtbar, die Odometriesensoren sind nicht sichtbar hinter den Antriebsrädern angebracht. Alle Kabel sind mit Kabelbindern eng am Rahmen verlegt, weitestgehend auf der Innenseite. Dadurch wird verhindert, dass sich Kabel an Gegenständen oder Personen verfangen.

Für die Inbetriebnahme gibt es nur einen Hauptschalter, somit ist es nur möglich, das Gesamtsystems zu starten. Sollte es nach dem Einschalten Fehler mit der Hardware geben, werden diese am Beleuchtungssystem angezeigt.

Das Betriebssystem Gentoo-Linux wurde so konfiguriert, dass nur notwendige Prozesse laufen. Auf diese Weise hat die Software GTRolland immer genug Prozesszeit, um sicher zu laufen. Die Software GTRolland selbst wurde entsprechend dem Betriebssystem angepasst.

Das Fahrverhalten wurde so optimiert, dass selbst unerfahrene Benutzer keine Probleme haben, den Rollstuhl zu bedienen. Der HNF-Rolland bremst sanft vor Hindernissen und bleibt immer in nahezu gleichem Abstand zu den Hindernissen stehen. Ein Festfahren des Rollstuhls kommt so nicht mehr zustande.

Leichte Unregelmäßigkeiten in der Bremsbewegung lassen sich aufgrund der weichen Fahrwerte, die beim HNF-Rolland auf Seiten des Meyra Champ eingestellt sind, nicht vermeiden. Tests des Fahrverhaltens des HNF-Rollands auf dem Rolland III zeigen, dass diese sich durch direktere Fahreinstellungen beseitigen lassen. Zur Optimierung könnte man zukünftig die weichen Fahreinstellungen nicht von Seiten des Meyra Champs sondern von Seiten der GTRolland Software realisieren.

Phänomene im Fahrverhalten des Rolland III, die nicht intuitiv und damit nicht benutzerfreundlich sind, wurden beseitigt. So zum Beispiel die Tatsache, dass man beim Rolland III nach einer Vollbremsung noch weiter an das Hindernis heranfahren kann oder dass man den Joystick nach einer Vollbremsung in die Nullstellung bringen muss, um weiter fahren zu können.

Der HNF-Rolland erfüllt alle Anforderungen, die im Vorfeld an ihn gestellt wurden. Auch Probleme, die sich erst beim Einsatz in der Ausstellung ergaben,

wurden gelöst. Der HNF-Rolland kann so problemlos über lange Zeit in Museen gezeigt werden. Darüberhinaus besteht die Möglichkeit, ihn zum Mobilitätsgewinn bei Menschen mit Behinderungen einzusetzen.

## 5.5 Danksagungen

Bedanken möchte ich mich bei meinen Gutachtern Prof. Dr. Bernd Krieg-Brückner und Dr. Udo Frese für die Bereitstellung des Themas meiner Diplomarbeit, sowie Dr. Thomas Röfer, der ebenfalls großen Anteil an der Entstehung dieses Themas hatte.

Für die Hilfestellung bei allen Problemen, die sich im Hard- sowie im Softwarebereich ergaben, danke ich Dr. Udo Frese, Dr. Thomas Röfer und Christian Mandel, die immer ein offenes Ohr für mich hatten und ohne deren Erfahrungen die Umsetzung für mich nicht möglich gewesen wäre.

Christoph Budelmann danke ich für den Entwurf und die Umsetzung der elektrischen Schaltungen für das Watchdog- und das Odometriemodul.

Ebenfalls danke ich den Mitarbeitern der Werkstatt der AG Robotik an der Universität Bremen, die mir professionelle Anleitung zum Bau aller Sensorhalterungen gaben.

Dank gebührt auch den Mitarbeitern des Heinz Nixdorf MuseumsForums, die es ermöglichten, den HNF-Rolland in der Ausstellung 'Computer.Medizin' zu zeigen und die mir immer die volle Unterstützung bei der Evaluation und den letzten Anpassungen in der Ausstellung gaben. Besonders möchte an dieser Stelle Herrn Mikolajzek, Kurator im Heinz Nixdorf MuseumsForum, danken, der vor Ort mein erster Ansprechpartner war und dem es hauptsächlich zu verdanken ist, dass der HNF-Rolland ausgestellt wird.

Außerdem danke ich allen Bremer Mitarbeitern des DFKI, die mich in vielen Bereichen unterstützten.

# A Anhang

## A.1 Schaltplan des Steuerungs- und Watchdogmoduls

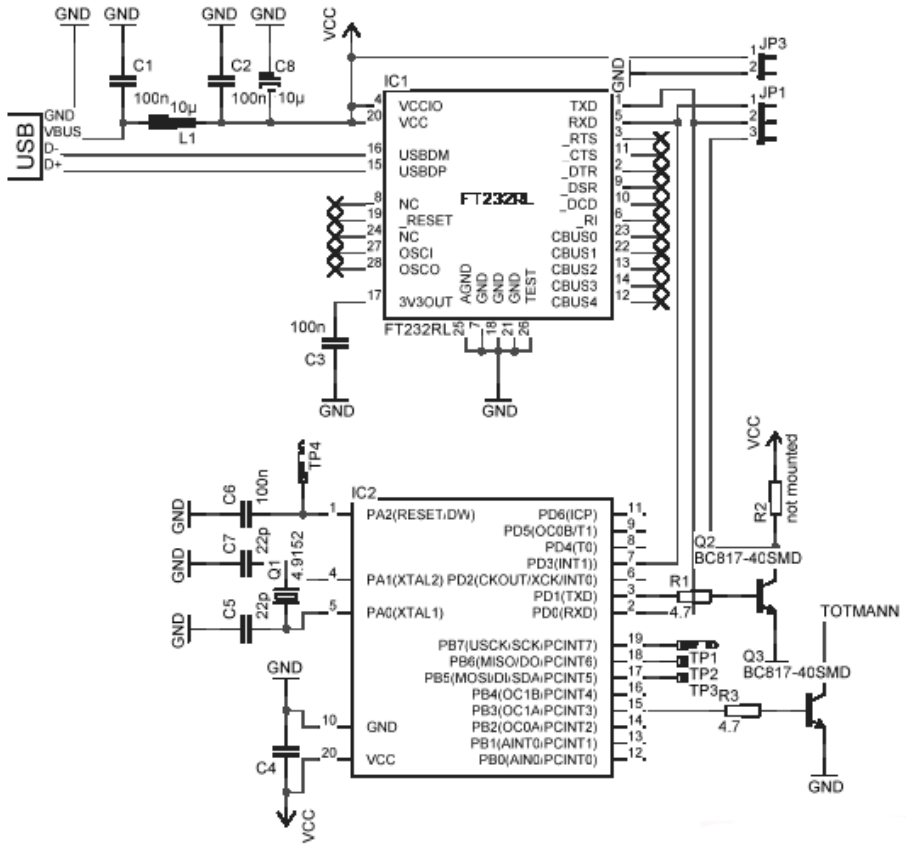



Abbildung 20: Schaltplan des Steuerungs- und Watchdogmoduls
von Christoph Budelmann

## A.2 Zeichnungen der Laserscannerhalterungen

Alle Angaben in mm

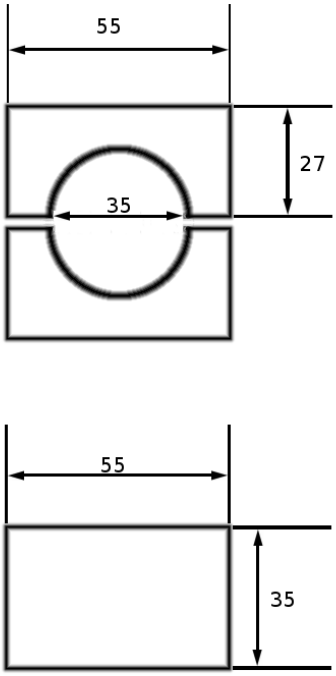



Abbildung 21: Manschetten zum Befestigen der Laserscannerhalterungen

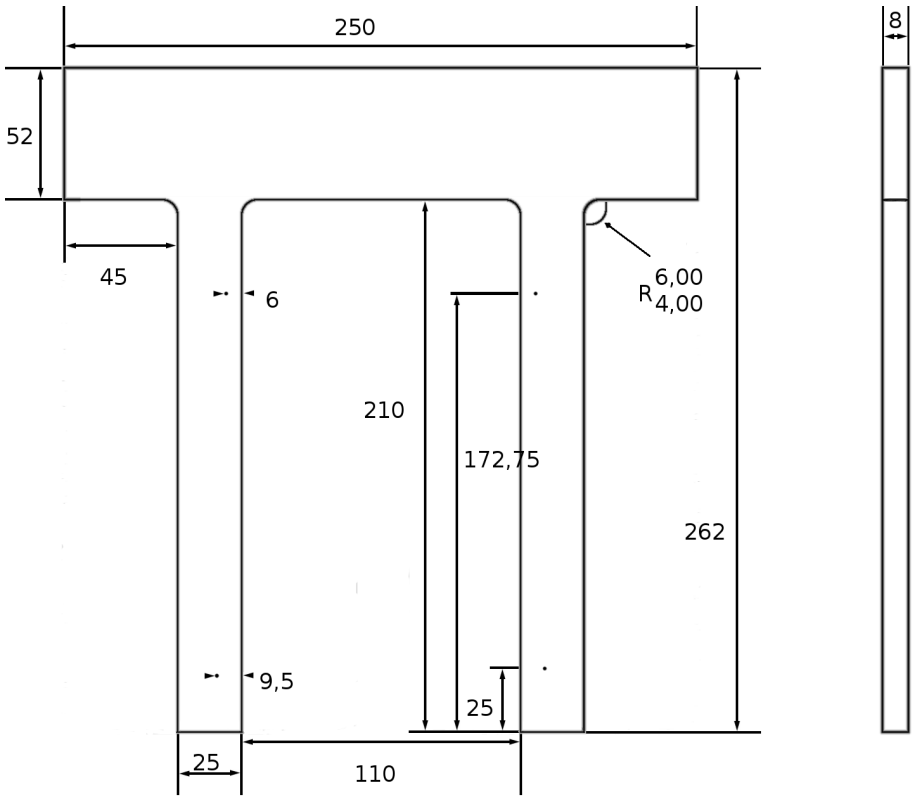



Abbildung 22: Laserscannerhalterung vorne

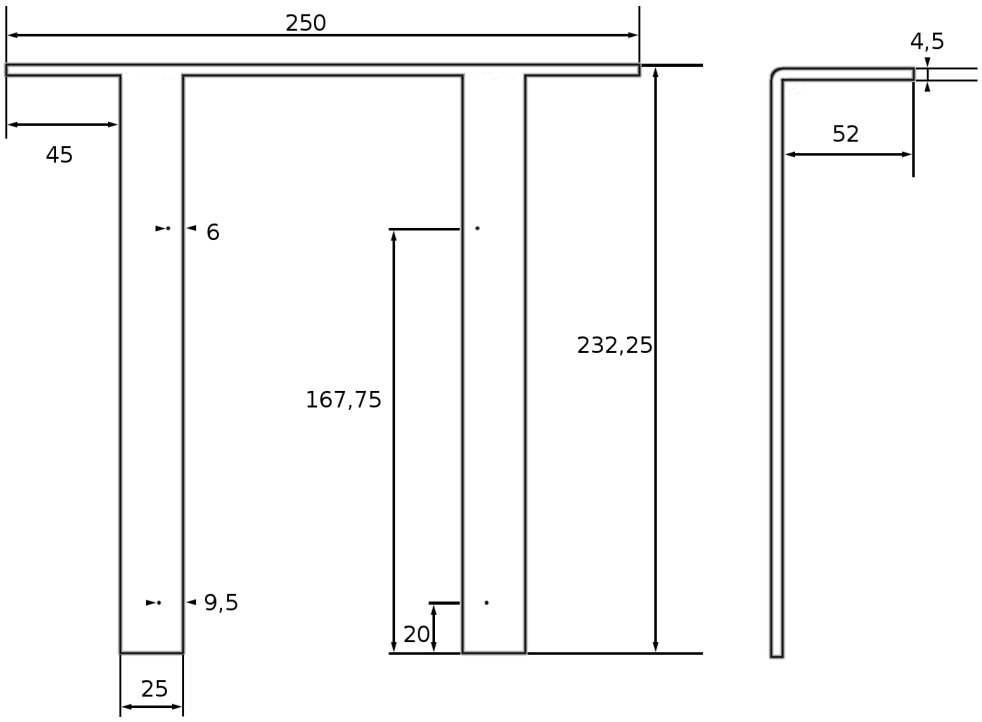
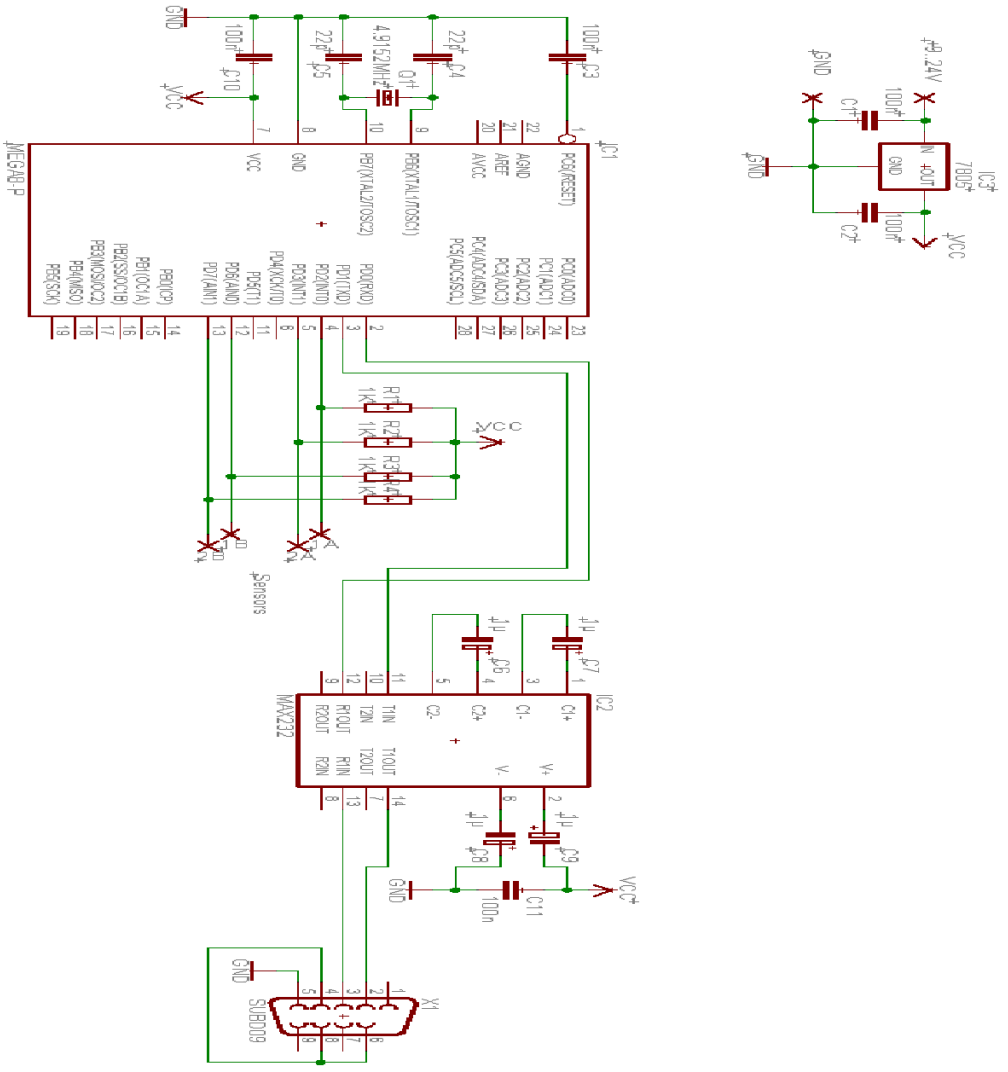



Abbildung 23: Laserscannerhalterung hinten

## A.3 Schaltplan des Odometriemoduls von Christoph Budelmann



## A.4 Bedienungsanleitung des HNF-Rollands

Bedienungsanleitung für den intelligenten Rollstuhl

# Rolland

Bitte beachten Sie zusätzlich alle Hinweise aus der Meyra Champ Bedienungsanleitung!

**Inbetriebnahme**

Bevor Sie den Rolland einschalten, stellen Sie sicher, dass

1. der Bremsentriegelungshebel auf Motorbetrieb (Fahren) geschaltet ist,
2. der Stecker des Ladegerätes nicht in der Batterie-Ladebuchse des Bedienmoduls steckt. (Einschalten des Rolland während des Ladevorgangs kann zur Schädigung der Sensoren oder des Computers führen),
3. alle Hinweise aus der Meyra Champ Bedienungsanleitung beachtet wurden.

Nun können Sie den Hauptschalter betätigen.

**Verhalten nach dem Anschalten**

1. Direkt nach dem Betätigen des Hauptschalters leuchten die mittleren Kontrollleuchten der Laserscanner auf, und der Lüfter des Computers beginnt sich zu drehen. Während die Laserscanner ihre Startkontrolle durchführen ändern sie ihre Kontrollleuchtenanzeige. Nach erfolgreicher Startkontrolle leuchten die Kontrollleuchten 2 und 3 (von links) an beiden Laserscannern.
2. Nach etwas über einer Minute schaltet sich das Bedienmodul ein.
3. Nach weiteren ca. 10 Sekunden ertönt kurz die Hupe. Dies ist das Zeichen, dass der Rolland nun voll funktionsfähig ist.

(Sollte nach der beschriebenen Zeit die Hupe nicht ertönen, lesen Sie bitte den Abschnitt „Fehlermeldungen“.)

**Ausschalten**

Schalten Sie einfach den Hauptschalter aus. Die Kontrollleuchter der Laserscanner, der Lüfter des Computers und das Bedienmodul gehen sofort aus.

**Fehlermeldungen**

Der Rolland ist so konfiguriert, dass es nicht möglich ist zu fahren, wenn ein Fehler vorliegt.
Die Fehler und Kontrollmeldungen des Rolland sind:

1. Ein Ertönen der Hupe, ca. eineinhalb Minuten nach dem Betätigen des Hauptschalters deutet darauf hin, dass keine Fehler vorliegen. Dieses Signal ertönt auch, wenn die Datenübertragung zwischen Computer und Bedienmodul nach einer Störung wiederhergestellt wird.
2. Einschalten der Beleuchtung des Rolland, weist darauf hin, dass der Computer keine Daten von den Bewegungssensoren empfängt.
   In diesem Fall sollten Sie die Verkabelung des Bewegungssensormoduls überprüfen.
3. Blinken des linken Blinkers, deutet auf ein Problem mit dem vorderen Laserscanner hin. Sollte der rechte Blinker blinken, so gibt es ein Problem mit dem hinteren Laserscanner. In diesem Fall sollten Sie, bei abgeschaltetem Hauptschalter, zunächst die Scheibe des betroffenen Laserscanners reinigen. Liegt das Problem weiterhin vor, überprüfen Sie bitte die Verkabelung des jeweiligen Laserscanners.
4. Sollte sich das Bedienmodul zwei Minuten nach dem Betätigen des Hauptschalters nicht

einschalten, konnte der Computer nicht booten, oder es gibt ein Problem mit dem Champ-Steuerungsmodul.
In diesem Fall schalten Sie den Hauptschalter aus, und schalten ihn ca. zwei Minuten später wieder ein.
Sollte das Problem weiter vorliegen, überprüfen Sie die Verkabelung des Champ-Steuerungsmoduls und den Halt des USB-Sticks im USB-Anschluss.

**Kollisionsvermeidung**

Der Rolland verhindert das Kollidieren mit Hindernissen. Dafür werden Fahrbefehle, die zu einer Kollision führen würden, nicht ausgeführt. Sollte die gewünschte Fahrtrichtung mit einer geringeren Geschwindigkeit möglich sein, wird der Fahrbefehl mit verringerter Geschwindigkeit ausgeführt. Vor Hindernissen wird zunächst stark gebremst, dann ist ein langsames Heranfahren möglich. Dafür können Sie dauerhafte Fahrbefehle auf das Hindernis zu ausführen (Joystick in Richtung des Hindernis drücken).

Hinweise:

Die beiden Laserscanner haben einen Messbereich von 190° nach vorne bzw. hinten. Sie nehmen nur Hindernisse innerhalb ihres Messbereichs in einer Höhe von ca. 5-10cm wahr. Hindernisse, die höher oder tiefer liegen, werden nicht erkannt. Zum Beispiel werden bei Tischen nur die Beine, nicht aber die Tischplatte wahrgenommen. Prinzipbedingt können die Laserscanner nur Hindernisse erkennen, die innerhalb ihres Messbereichs liegen und nicht von Teilen des Rollstuhls selbst verdeckt sind, d.h. insbesondere den vorderen Rädern. Die vorderen Lenkräder verhindern, je nach Stand, den „Blick“ des vorderen Laserscanners zu Teilen der Seitenbereiche. Hindernisse in diesen Bereichen können daher nicht wahrgenommen werden. Allerdings merkt sich der Rolland Hindernisse, die er einmal gesehen hat, für eine kurze Zeit, so dass er auch vor Hindernissen hält, die innerhalb seines Blindbereiches liegen. Diese werde aber nach einigen Sekunden „vergessen“, damit der Fahrer nicht in Situationen geraten kann, in denen er in keine Richtung mehr fahren kann. Daher kann es zu Kollisionen kommen, wenn man nach einen Abbremsen noch für längere Zeit auf ein Hindernis zuhält, das die Laserscanner nicht wahrnehmen können. Um Schäden zu vermeiden, fährt der Rolland nur sehr langsam in diese unbekannten Bereiche. Außerdem sind die Fußstützen speziell gepolstert.

Beim Anfahren können die vorderen Lenkräder, je nach Stand, die Fahrtrichtung leicht verändern. Dadurch kann die gewünschte Fahrtrichtung leicht verändert werden. Infolge dessen kann es zu kleinen Kollisionen kommen. Leider gibt es keine Möglichkeit, die Postion der Lenkräder und die daraus folgende Richtungsänderung zu berechnen. Diese Kollisionen können nur beim Anfahren, daher mit geringen Geschwindigkeiten und im Nahbereich, auftreten.

Bewegte Hindernisse werden erkannt, nicht aber ihre Bewegungen. Wenn Menschen oder Fahrzeuge den Weg kreuzen, bremst der Rolland erst, wenn sich das Hindernis im Fahrtweg des Rollands befindet, nicht vorausschauend.

Achten Sie darauf, dass während der Fahrt keine Gegenstände vor den Laserscannern sind. Dies könnten die Absätze von Schuhen sein, die nicht vollständig auf den Fußstützen stehen, Hosenbeine, die über die Fußstützen hinaus nach unten hängen, oder Mäntel, die über die Rückenlehne gelegt werden. Diese Gegenstände werden als Hindernisse erkannt, und können jeden Fahrbefehl blockieren.

**Allgemeine Hinweise zur Sicherheit**

Laden Sie die Batterien niemals, wenn der Hauptschalter eingeschaltet ist.

Stellen Sie den Bremsentriegelungshebel niemals auf Schieben, wenn der Hauptschalter eingeschaltet ist.

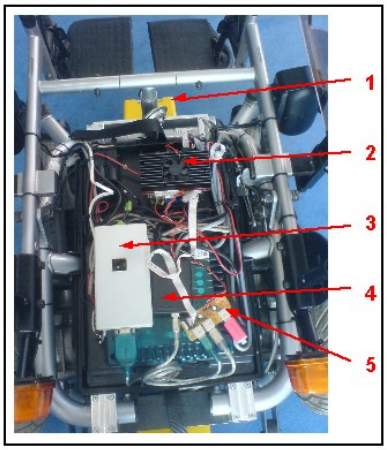
Belasten Sie niemals die Fußstützen mit dem vollen Körpergewicht. Beachten Sie dies besonders beim Ein- und Aussteigen.

Bitte beachten Sie zusätzlich alle Hinweise aus der Meyra Champ Bedienungsanleitung!

**Übersicht**


1. Joystick
2. Bedienmodul
   (Batterie-Lade-Buchse auf der Unterseite)
3. Fußstützen
4. Laserscanner
5. Hauptschalter (an der Unterseite)
6. Lenkräder
7. Antriebsräder

Der Bremsentriegelungshebel befindet sich auf der linken Rollstuhlseite vor dem Antriebsrad. Ganz nach vorne steht er auf Motorbetrieb (Fahren), ganz nach hinten steht er auf Schieben (Lösen der Bremse).




1. Laserscanner (vorne)
2. Computer
3. Bewegungssensormodul
4. Champ-Steuerungsmodul
5. USB-Anschlüsse
   (Sollten Sie die USB-Anschlüsse abziehen müssen, achten Sie auf die richtige Reihenfolge: von links nach rechts Bewegungssensormodul, Champ-Steuerungsmodul, USB-Stick mit Betriebssystem)

## A.5 Konfiguration des Betriebssystems

Das Betriebssystem wird für den HNF-Rolland auf einem 2-GB-USB-Flash-Stick von SanDisk installiert. Als erstes werden zwei Partitionen erstellt, eine für den Bootmanager und eine Systempartition. Die Bootpartition ist 32 MB groß. Der übrige Speicherplatz des USB-Sticks enthält die Systempartition. Es wird bewusst keine SWAP-Partition erstellt. Diese gehört normalerweise zum Standard bei Linuxsystemen und stellt Festplattenspeicher für die Auslagerungen aus dem Arbeitsspeicher bereit. Durch das Fehlen der SWAP-Partition wird das Betriebssystem gezwungen, keine Daten aus dem Arbeitsspeicher auszulagern, was, wie bereits beschrieben, zu Problemen führen kann.

Beide Partitionen werden mit dem Dateisystem ReiserFS formatiert. Dies ist ein Journaling-Dateisystem, was den Vorteil hat, dass bei jedem schreibenden Dateizugriff ein Journal geschrieben wird, welches die Zugriffe dokumentiert. Nach einem Systemabsturz müssen folglich nur die Dateien auf Konsistenz geprüft werden, die zur Zeit des Absturzes geöffnet waren. Da das System des HNF-Rolland nach dem Betrieb nicht heruntergefahren wird sondern mit dem Hauptschalter auch der Computer ausgeschaltet wird, ist diese Funktion ein wichtiger Sicherheitsfaktor. Zusätzlich wird die Systempartition nach dem Booten als read-only neu eingebunden, so dass sich die Zeit eines möglichen Datenverlustes auf den Bootvorgang beschränkt.

Auf der Systempartition wird nun das Gentoo-Basissystem entpackt, und der Paketmanager Portage konfiguriert. Außerdem wird die Zeit und die Zeitzone eingestellt.

Der Kernel kann nun konfiguriert werden. Für den HNF-Rolland wird nicht der von Gentoo gelieferte Kernel, sondern, wie bereits erwähnt, der RTLinux-Kernel verwendet. Eine genaue Beschreibung der Kernelkonfiguration findet sich im Abschnitt 3.1.5.

Nun werden Systemeinstellungen vorgenommen, wie die Konfiguration des Netzwerks und die fstab-Tabelle, die angibt, welche Partitionen beim Systemstart eingebunden werden sollen. Dafür werden die Dateien '/etc/conf.d/net' und '/etc/conf.d/hostname' für die Netzwerkkonfiguration, beziehungsweise '/etc/fstab' für die Partitionstabelle editiert. Außerdem werden Geringfügigkeiten, wie die Keymaps und das root-Passwort, eingestellt.

Auf alle optionalen Systemtools wird bei der Gentoo-Installation für den HNF-Rolland verzichtet. So enthält das System keinen System-Logger zum Protokollieren von Systemaktivitäten und keinen Cron-Deamon zum geplanten Ausführen von Tasks. Es wird jedoch ein SSH-Deamon installiert, um den HNF-Rolland leichter über ein Netzwerk zu warten.

Als Bootmanager wird Lilo installiert, weil es sich als leichter herausstellte,

mit Lilo von einem USB-Stick zu booten als mit Grub. In der Lilo-Konfiguration gibt es nur eine Bootoption, die nach einem Timeout von 0 Sekunden geladen wird.

Über das Tool 'rc-update', mit dem man Dienste in den Bootprozess einbinden und entfernen kann, werden nur die Dienste zur Initialisierung der Netzwerkkarte, der SSH-Deamon, und die selbst hinzugefügten Befehle in der Option 'local' gestartet.

Die 'local'-Option des rc-update-Tools wird in der Datei '/etc/conf.d/local.start' definiert. Diese wird beim Systemstart ausgeführt, wenn die 'local' Option über 'rc-update' in den Bootprozess eingeführt wurde. Entsprechend gibt es eine '/etc/conf.d/local.stop'-Datei, die beim Herunterfahren ausgeführt würde, diese wird jedoch auf dem Computer des HNF-Rolland nicht verwendet.

In der Datei '/etc/conf.d/local.start' wird mit dem Befehl 'mount -n -o remount,ro /' die Systempartition als 'read-only' neu eingebunden, um die Konsistenz nach dem Ausschalten, ohne Herunterfahren, zu gewährleisten. Des Weiteren wird in diesem Skript der Befehl 'gtrolland' ausgeführt. Dieser Befehl startet die GTRolland-Software und wird über ein Shellskript umgesetzt, das im Verzeichnis '/bin' liegt. Zuletzt wird mit dem Befehl 'echo "Rolland06 is ready“ 'Rolland06 is ready' auf die Konsole geschrieben, um einen erfolgreichen Bootvorgang zu bestätigen.

## A.6 Konfiguration des RTLinux-Kernels

Der RTLinux-Kernel wird nur mit dem Minimum benötigter Module konfiguriert, um ihn möglichst schlank zu halten. Es werden im Folgenden nicht alle Module einzeln aufgezählt, sondern beschrieben, welche Unterstützungen unabdingbar sind und welche Standardmodule eines normalen Linux-Kernels nicht installiert werden.

Es wird keine Unterstützung der Soundkarte, des CD-, beziehungsweise des DVD-Laufwerks und keine Plug and Play Unterstützung installiert. Im Gegensatz zu normalen Linux-Systemen wird nur das ReiserFS-Dateisystem und keine weiteren Linux-, CD-, oder Microsoft-Dateisysteme unterstützt.

Des Weiteren werden unterstützt: SCSI-Disk und -Emulation Unterstützung, USB-Dateisystem und -Massenspeicher, alles um auf den USB-Stick zugreifen zu können; Unterstützung von seriellen Verbindungen und USB-Seriell-Konvertern; von den Konvertern werden mehr unterstützt als benötigt werden, um Probleme bei einem eventuellen Austausch zu verhindern; Netzwerkunterstützung mit den entsprechenden Treibern für die Netzwerkkarte und Power-Management-Unterstützung, auch wenn Letzteres nicht unbedingt erforderlich

wäre. Als Prozessor ist, wie im System vorhanden, der Pentium 4 Celeron angegeben.

Alle Module werden fest in den Kernel und nicht als nachladbare Module kompiliert.

## A.7 HNFSafetyLayer-Quellcode

```
/**
* @file HNFSafetyLayer.cpp
* This file implements a safety layer based on the SAM-module of Rolland II.
* @author <a href="mailto:ufrese@tzi.de">Udo Frese</a>
* @author <a href="mailto:jogo@tzi.de">Jost Gollub</a>
*/


#include "HNFSafetyLayer.h"
#include "Platform/Win32/GTAssert.h"
#include "Tools/Debugging/GenericDebugData.h"

#include <iostream>
using namespace std;

/** BrakingModel parameter see \c BrakingModel
 */

/** HNF Wheelchair parameter */

#define AXLE_WIDTH 641
#define ROTATION_DISTURBANCE  0.01
#define TRANSLATION_DISTURBANCE 0.005
#define PROPORTIONAL_DECELERATION 0.5
#define COULOMB_DECELERATION 600
#define LATENCY 0.1
#define MAX_SPEED 2000
#define MAX_BACKWARD_SPEED 1000
#define MAX_ROTATION_SPEED 3
#define WHEEL_SPEED_DISCRETIZATION_ERROR 20

/** Resulting error in wheel speed (between actual wheel speeds and next discretized table. */

#define CELL_SIZE 25

/** Maximum distance of any robot point to the center of rotation.
  * Equals ChampShape::getMaxRadius()
  */
#define RADIUS 950

/**
  Below which safety factor should the speed be reduced
*/
#define SOFT_LAMBDA 3

/**
  Below which safety factor should an emergencey stop be initiated
*/
#define HARD_LAMBDA 1.25

/** We do not check for braking distances above LAMBDA_CUTOFF time maximum speed. So if
  * SOFT_LAMBDA*currentSpeed > LAMBDA_CUTOFF*maxSpeed speed is not reduced. This might sound
  * contraintuitive but allows to drive fast without looking very far ahead at the price of
  * harder braking.
  */
#define LAMBDA_CUTOFF 1.75

/**
  We compute the virtual sensor up to \c LAMBDA_VS times the maximal speed configured
 */
#define LAMBDA_VS 1.75

/**
  Below this speed (each wheel individually) we are allowed to move through unknown region.
  */
#define UNKNOWN_REGION_MAX_SPEED 150
```

```
HNFSafetyLayer::HNFSafetyLayer
(const SafetyLayerInterfaces& interfaces)
    : SafetyLayer(interfaces),
    brakingModel (AXLE_WIDTH,
                  ROTATION_DISTURBANCE, TRANSLATION_DISTURBANCE,
                  PROPORTIONAL_DECELERATION, COULOMB_DECELERATION, LATENCY,
                  MAX_SPEED*LAMBDA_VS, MAX_BACKWARD_SPEED*LAMBDA_VS, MAX_ROTATION_SPEED*LAMBDA_VS,
                  WHEEL_SPEED_DISCRETIZATION_ERROR),
    plotter(0), dist2(0)

{
  ASSERT (HARD_LAMBDA<=LAMBDA_VS && LAMBDA_VS<=LAMBDA_CUTOFF && LAMBDA_CUTOFF<=SOFT_LAMBDA);

  isBraking = isSafetyBraking = false;
  hasStopped = true;
  hasStopped = false;
  hasBeenMovingFast = false;
  scts.createOrLoad (Pose2D(0,0,0), robotShape, brakingModel, CELL_SIZE);
  targetSpd = 0;
  vDeltaI = 0;
  lastCmdScale = 1;
  showVL = showVR = 0;

  worstPose = OdometryPose();
  worstMaxSpd = 100000;
  commandIsNonzeroSince = INT_MAX;
  odometryBroken = false;
  showWhichCollisionTable = SHOW_CURRENT;
  wheelchairInactiveUntil = 0;
}

//! Returns a threshold for evidence grid cells to be considered empty
//! Depends on the speed so as faster you go the higher the possibility must be
//! that the cell must
//! is empty.
int HNFSafetyLayer::gridThreshold (double spd)
{
  /* We had the phenomen that the unobserved region below the wheelchair
     becomes to uncertain for the wheelchair to move. */
  /*  int myThreshold;
    double spdThresholdFactor = 40000;
    if (spd<spdThresholdFactor/120) myThreshold = 255;
    else myThreshold = (int) (128 + ceil(spdThresholdFactor/spd));
    return myThreshold;*/
  return 129;
}

void HNFSafetyLayer::modifySpeedForWorstCase(double vL, double vR)
{
  double realSpeedNorm1=poseApplied.speedNorm1();
  double scaleSpeed=100;
  if(!joystickZero && realSpeedNorm1<scaleSpeed)
  {
    double joystickSpeedNorm1=fabs(vL)+fabs(vR);

    double odometryPart=realSpeedNorm1/scaleSpeed;
    double joystickPart=1-odometryPart;

    double leftWheelJoystick=0;
    double rightWheelJoystick =0;
    if(joystickSpeedNorm1!=0)
    {
      double scaleJoystick=scaleSpeed/joystickSpeedNorm1;
      leftWheelJoystick  = vL*scaleJoystick;
      rightWheelJoystick = vR*scaleJoystick;
    }

    double leftWheelOdometry=0;
    double rightWheelOdometry =0;
    if(realSpeedNorm1!=0)
    {
      double scaleOdometry=scaleSpeed/realSpeedNorm1;
      leftWheelOdometry  = poseApplied.speedLeftWheel * scaleOdometry;
      rightWheelOdometry = poseApplied.speedRightWheel * scaleOdometry;
    }

    poseApplied.speedLeftWheel = leftWheelJoystick*joystickPart + leftWheelOdometry*odometryPart;
```

```
    poseApplied.speedRightWheel = rightWheelJoystick*joystickPart + rightWheelOdometry*odometryPart;
  }


}

void HNFSafetyLayer::slowDriveControll(double rotationSpeed,double realSpeed)
{
  int absSteering=abs(driveRequest.steering);
  double maxSteering = 20 + ((400-rotationSpeed)/400)*10;
  if(absSteering>maxSteering)
  {
    if(driveRequest.steering<0)
      driveRequest.steering=(int)-maxSteering;
    else
      driveRequest.steering=(int)maxSteering;
  }

  int absSpeed=abs(driveRequest.speed);
  double maxSpeed=30+((400-realSpeed)/400)*10;
  if(absSpeed>maxSpeed)
  {
    if(driveRequest.speed<0)
      driveRequest.speed=(int)-maxSpeed;
    else
      driveRequest.speed=(int)maxSpeed;
  }

}



void HNFSafetyLayer::executeSafety()
{

  bool oldIsSafetyBraking = isSafetyBraking;
  poseApplied = toPose2DSpeed(odometryPose);

  const int toZeroLimit = 10;
  if (driveRequest.steering < -toZeroLimit)
    driveRequest.steering = -toZeroLimit + driveRequest.steering * (63 - toZeroLimit) / 63;
  else if (driveRequest.steering > toZeroLimit)
    driveRequest.steering = toZeroLimit + driveRequest.steering * (63 - toZeroLimit) / 63;
  else
    driveRequest.steering = 0;

  double spdScale = 1000/64.0;
  double vL;
  double vR;
  if(driveRequest.speed<0)
  {
    vL = spdScale * ((driveRequest.speed/2) - driveRequest.steering);
    vR = spdScale * ((driveRequest.speed/2) + driveRequest.steering);
  }
  else
  {
    vL = spdScale * (driveRequest.speed - driveRequest.steering);
    vR = spdScale * (driveRequest.speed + driveRequest.steering);
  }

  double speedNorm1=poseApplied.speedNorm1();
  double realSpeed =fabs((poseApplied.speedLeftWheel+poseApplied.speedRightWheel)/2);
  bool fast=(realSpeed>700);
  double rotationSpeed=fabs(poseApplied.speedRightWheel-poseApplied.speedLeftWheel);
  // modify poseApplied according to vL vR with wheel-wise worst case policy
  modifySpeedForWorstCase(vL,vR);
  double worstCaseSpeed=poseApplied.speedNorm1();



  bool canMoveSlowly;
  double maxSafeSpd;  // .speedNorm1()

  int myThreshold;
  if(worstCaseSpeed<0)
    myThreshold=gridThreshold (poseApplied.speedNorm1());
  else
    myThreshold=gridThreshold (poseApplied.speedNorm1());
  Vector2<double> p;
```

```
  maxSafeSpd = scts.maxSafeSpeedNorm (canMoveSlowly, poseApplied, evidenceGrid, myThreshold, p);

  double softBrake=SOFT_LAMBDA; //this is for potential changing of brakeing-limits
  double hardBrake=HARD_LAMBDA;

  double timeStep  = 0.03;
  double softSpd = maxSafeSpd/softBrake;   // spd>softSpd   ==> brake
  double hardSpd = maxSafeSpd/hardBrake;

  if ((worstCaseSpeed)>=hardSpd) isSafetyBraking = true;
  else isSafetyBraking = false;

  if ((worstCaseSpeed)>softSpd&&!isSafetyBraking)
  {
    double requestedSpeed=fabs((vL+vR)/2);
    double scale1=(softSpd)/worstCaseSpeed;
    double scale2 = 10;
    if(requestedSpeed!=0)
      scale2 = softSpd/requestedSpeed;
    double scale = min(scale1,scale2);
    scale*=((hardSpd-worstCaseSpeed)/(hardSpd-softSpd));
    driveRequest.speed=(int) (driveRequest.speed*scale);
    driveRequest.steering=(int) (driveRequest.steering*scale);
    if(fast)
    {
      if(driveRequest.speed>0)
        driveRequest.speed-=5;
      else
        driveRequest.speed+=5;
    }
  }

  if (isSafetyBraking) driveRequest.speed = driveRequest.steering = 0;

  //changing the maximal speed and rotational speed
  slowDriveControll(rotationSpeed,realSpeed);

  if (isSafetyBraking && !oldIsSafetyBraking)
  {
    double cellSize = evidenceGrid.cellSize;
    LINE (safetyInfo, p.x-3*cellSize, p.y, p.x+3*cellSize, p.y, 1, Drawings::ps_solid, Drawings::red);
    LINE (safetyInfo, p.x, p.y-3*cellSize, p.x, p.y+3*cellSize, 1, Drawings::ps_solid, Drawings::red);
    DEBUG_DRAWING_FINISHED (safetyInfo);
  }

  static double t=0;
  t += 10;
  COMPLEX_DRAWING(driveControllerPlot,
                  plotter.plot (t, driveRequest.speed * 10, Drawings::pink);
                  plotter.plot (t, driveRequest.steering * 10, Drawings::orange);
                  plotter.plot (t, odometryPose.speed.translation.x, Drawings::red);
                  plotter.plot (t, toDegrees(odometryPose.speed.rotation), Drawings::blue);
                  plotter.plot (t, isSafetyBraking?100:0, Drawings::yellow);
                  plotter.plot (t, (hasStopped && !joystickZero)?90:10, Drawings::green);
                  plotter.send(Drawings::driveControllerPlot);
                 );
}


bool HNFSafetyLayer::isHardwareStateOk ()
{
  return hardwareState.isFrontScannerActive &&
         hardwareState.isBackScannerActive &&
         hardwareState.isOdometryActive &&
         hardwareState.isWheelchairActive &&
         !hardwareState.isWheelchairOdometryError;
}


void HNFSafetyLayer::execute()
{
  justStopped = !hasStopped;
  if (fabs(odometryPose.speed.translation.x)<20 && fabs(odometryPose.speed.rotation)<0.02) hasStopped = true;
  else if (fabs(odometryPose.speed.translation.x)>40 || fabs(odometryPose.speed.rotation)>0.04)
    hasStopped = false;
  if (!hasStopped) justStopped = false;

  joystickZero = abs(driveRequest.speed)+abs(driveRequest.steering)<6;
  if (isSafetyBraking || joystickZero)
```

```
  {
    // braking
    if (!isBraking)
    {
      startOfDeceleration = odometryPose;
    }
    isBraking = true;
    if (hasStopped) isSafetyBraking = false;
  }
  else
  {
    isBraking = false;
  }

  if (!joystickZero||!hasStopped) executeSafety ();

  if (abs(driveRequest.speed) + abs(driveRequest.steering)<160)
    commandIsNonzeroSince = INT_MAX;
  else if (commandIsNonzeroSince == INT_MAX)
    commandIsNonzeroSince = driveRequest.timeStamp;

  if (!isHardwareStateOk ()) // Sensors are wrong stop now
  {
    driveRequest.speed = driveRequest.steering = 0;
    wheelchairInactiveUntil = SystemCall::getCurrentSystemTime();
  }
  else
    signalsRequest.horn = SystemCall::getTimeSince(wheelchairInactiveUntil) < 100;


  if(hardwareState.isWheelchairActive) // without wheelchair, we cannot set signals
  {
    if(hardwareState.isFrontScannerActive && hardwareState.isBackScannerActive)
      signalsRequest.winker = SignalsRequest::off;
    else if(!hardwareState.isFrontScannerActive && !hardwareState.isBackScannerActive)
      signalsRequest.winker = SignalsRequest::warning;
    else if(!hardwareState.isFrontScannerActive)
      signalsRequest.winker = SignalsRequest::left;
    else
      signalsRequest.winker = SignalsRequest::right;
    signalsRequest.light = !hardwareState.isOdometryActive || hardwareState.isWheelchairOdometryError;
  }

  visualizeCollisionTable1 ();
}

bool HNFSafetyLayer::handleMessage(InMessage& message)
{
  if(message.getMessageID() == idGenericDebugData)
  {
    GenericDebugData rq;
    message.bin >> rq;
    if(rq.id == GenericDebugData::samVirtualSensor)
    {
      showVL = rq.data[0];
      showVR = rq.data[1];
      showWhichCollisionTable = SHOW_VLVR;
    }
    return true;
  }
  return false;
}


Pose2DSpeed HNFSafetyLayer::toPose2DSpeed (const OdometryPose& p)
{
  return brakingModel.posePlusSpeedToPose2DSpeed (p, p.speed);
}

void HNFSafetyLayer::visualizeCollisionTable1 ()
{
  ASSERT(evidenceGrid.width == evidenceGrid.height);
  const double gridHalf = evidenceGrid.width / 2;
  EvidenceGrid grid (evidenceGrid.width, evidenceGrid.widthInCells, evidenceGrid.heightInCells);
  grid.timeStamp = evidenceGrid.timeStamp;
  grid.clear (0);
  grid.center = evidenceGrid.center;
  Pose2DSpeed start;
  bool usePrecomputed = true;
```

```
  if (showWhichCollisionTable!=SHOW_NOTHING)
  {
    start = poseApplied;
    if (showWhichCollisionTable==SHOW_VLVR)
    {
      start.speedLeftWheel  = showVL;
      start.speedRightWheel = showVR;
    }
    else usePrecomputed = true;
    if (usePrecomputed) scts.decode (grid, start);
    else
    {
      // compute SafetyCollisionTable online
      SafetyCollisionTable2 ct;
      Pose2DSpeed rStart (start);
      if (rStart.speedNorm1()==0)
      {
        rStart.speedLeftWheel = rStart.speedRightWheel = 1;
      }
      brakingModel.maxSpeedVector (rStart.speedLeftWheel, rStart.speedRightWheel);
      ct.create (rStart, robotShape, brakingModel, evidenceGrid.cellSize);
      ct.decode (grid, start);
    }

    grid.colorMask = 0x00ff00;
    INFO (send_collisionTable, idCollisionTable, bin, grid);
    if (!usePrecomputed) showWhichCollisionTable = SHOW_NOTHING;
  }
}
```

## Literatur

[Bell u. a. 1994] BELL, D.A. ; BORENSTEIN, J. ; LEVINE, S.P. ; KOREN, Y. ; JAROS, J.: An assistive navigation system for wheelchairs based upon mobilerobot obstacle avoidance. In: *Proceedings of the IEEE International Conference on Robotics and Automation* 3 (1994), S. 2018–2022

[Bühlmeier u. a. 1996] BÜHLMEIER, A. ; MANTEUFFEL, G. ; ROSSMANN, M. ; GOSER, K.: Robot Learning in Analog Neural Hardware. In: *Artificial Neural Networks – ICANN 96, Proceedings, Lecture Notes in Computer Science* 1112 (1996), S. 311–316

[Bühlmeier und Herwig 1995] BÜHLMEIER, Andreas ; HERWIG, Christoph: Analog neural networks and behavior based. In: *ZKW Bericht* 7 (1995)

[Borenstein und Koren 1991] BORENSTEIN, J. ; KOREN, Y.: The vector field histogram - fast obstacle avoidance for mobile robots. In: *IEEE Transactions on Robotics and Automation* 7 (1991), Nr. 3, S. 278–288

[Digital-Logic 2000] DIGITAL-LOGIC: *Technical User's Manual for: Microspace, PC/104 plus, smartModule855, MSM855*. Luterbach, Switzerland : Digital-Logic AG, 2000

[Fox u. a. 1997] FOX, Dieter ; BURGARD, Wolfram ; THRUN, Sebastian: The Dynamic Window Approach to Collision Avoidance. In: *IEEE Robotics and Automation Magazine* 4 (1997), März, S. 23–33

[Hokuyo 2005] HOKUYO: *Scanning Laser Range Finder for Robotics, URG-04LX*. Osaka, Japan : Hokuyo Automatic Co.LTD., 2005

[Khatib und Chatila 1995] KHATIB, M. ; CHATILA, R.: An extended potential field approach for mobile robot sensor-based motions. In: *In Proceedings of the Internation Conference on Intelligent Autonomous Systems (IAS'4)* (1995), S. 27–30

[Khatib 1986] KHATIB, Oussama: Real-time obstacle avoidance for robot manipulator and mobile robots. In: *The International Journal of Robotics Research* 5 (1986), Nr. 1, S. 90–98

[Krieg-Brückner u. a. 2005] KRIEG-BRÜCKNER, B. ; FRESE, U. ; LÜTTICH, K. ; MANDEL, C. ; MOSSAKOWSKI, T. ; ROSS, R.: Specification of an Ontology for Route Graphs. In: FREKSA, C. (Hrsg.) ; KNAUFF, M.

(Hrsg.) ; KRIEG-BRÜCKNER, B. (Hrsg.) ; NEBEL, B. (Hrsg.) ; BARKOWSKY, T. (Hrsg.): *Spatial Cognition IV* Bd. 3343. Springer-Verlag; D-69121 Heidelberg, Germany; http://www.springer.de, 2005, S. 390–412

[Krieg-Brückner u. a. 1998] KRIEG-BRÜCKNER, B. ; RÖFER, T. ; CARMESIN, H.-O. ; MÜLLER, R.: A Taxonomy of Spatial Knowledge for Navigation and its Application to the Bremen Autonomous Wheelchair. In: FREKSA, C. (Hrsg.) ; HABEL, C. (Hrsg.) ; WENDER, K. F. (Hrsg.): *Spatial Cognition*, Springer; http://www.springer.de/, 1998 (Lecture Notes in Artificial Intelligence 1404), S. 373–397

[Lankenau und Röfer 2000] LANKENAU, A. ; RÖFER, T.: Smart Wheelchairs - State of the Art in an Emerging Market. In: *Künstliche Intelligenz. Schwerpunkt Autonome Mobile Systeme* (2000), Nr. 4, S. 37–39

[Lankenau und Röfer 2001] LANKENAU, A. ; RÖFER, T.: A Safe and Versatile Mobility Assistant. In: *Reinventing the Wheelchair. IEEE Robotics and Automation Magazine* 8 (2001), Nr. 7, S. 29–37

[Lankenau und Röfer 2002] LANKENAU, A. ; RÖFER, T.: Mobile Robot Self-Localization in Large-Scale Environments. In: *Proceedings of the IEEE International Conference on Robotics and Automation 2002 (ICRA-2002)*, IEEE, 2002, S. 1359–1364

[Lankenau u. a. 2003] LANKENAU, A. ; RÖFER, T. ; KRIEG-BRÜCKNER, B.: Self-Localization in Large-Scale Environments for the Bremen Autonomous Wheelchair. In: FREKSA, C. (Hrsg.) ; BRAUER, W. (Hrsg.) ; HABEL, C. (Hrsg.) ; WENDER, K. F. (Hrsg.): *Spatial Cognition III*, Springer; http://www.springer.de/, 2003 (Lecture Notes in Artificial Intelligence 2685), S. 34–61

[Lenord+Bauer 2002] LENORD+BAUER: *GEL 248 MiniCODER, Betriebsanleitung*. Oberhausen, Germany : Lenord+Bauer & Co. GmbH, Januar 2002

[Leuze_electronic 2001] LEUZE_ELECTRONIC: *Flächendeckender Distanzsensor rotoScan ROD-4, Technische Produktbeschreibung*. Owen/Teck, Germany : Leuze electronic GmbH + Co. KG, 2001

[Levine u. a. 1999] LEVINE, S.P. ; BELL, D.A. ; JAROS, L.A. ; KOREN, Y. ; SIMPSON, R.C. ; BORENSTEIN, J.: The NavChair Assistive Wheelchair Navigation System. In: *Proceedings of the IEEE International Conference on Robotics and Automation* 7 (1999), S. 443–451

[Mandel u. a. 2006] MANDEL, C. ; FRESE, U. ; RÖFER, T.: Robot Navigation based on the Mapping of Coarse Qualitative Route Descriptions to Route Graphs. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2006)*, 2006, S. 205–210

[Mandel u. a. 2007] MANDEL, C. ; FRESE, U. ; RÖFER, T.: Design Improvements for Proportional Control of Autonomous Wheelchairs Via 3DOF Orientation Tracker. In: *Proceedings of the 9th International Work-Conference on Artificial Neural Networks (IWANN'2007)*, Springer; Berlin; http://www.springer.de, 2007 (Lecture Notes in Computer Science)

[Mandel u. a. 2005] MANDEL, C. ; HUEBNER, K. ; VIERHUFF, T.: Towards an Autonomous Wheelchair: Cognitive Aspects in Service Robotics. In: *Proceedings of Towards Autonomous Robotic Systems (TAROS 2005)*, URL `http://www.taros.org.uk/`, 2005, S. 165–172

[Meyra 2005] MEYRA: *CHAMP Modell 1.594, CHAMP Lift Modell 1.594-27, CHAMPI Modell 1.594-603, Bedienungsanleitung*. Vlotho, Germany : Wilhelm Meyra GmbH & Co. KG, September 2005

[Röfer 2003] RÖFER, Thomas: An Architecture for a National RoboCup Team. In: *Lecture Notes in Artificial Intelligence* (2003), S. 417–425

[Röfer 1997] RÖFER, T.: Routemark-Based Navigation of a Wheelchair. In: *Proceedings of the 3rd ECPD International Conference on Advanced Robotics, Intelligent Automation and Active Systems*, 1997, S. 333–338

[Röfer 2002] RÖFER, T.: Using Histogram Correlation to Create Consistent Laser Scan Maps. In: *Proceedings of the IEEE International Conference on Robotics Systems (IROS-2002)*, EPFL; Lausanne, Switzerland, 2002, S. 625–630

[Röfer und Lankenau 1999] RÖFER, T. ; LANKENAU, A.: Ein Fahrassistent für ältere und behinderte Menschen. In: SCHMIDT, G. (Hrsg.) ; HANEBECK, U. (Hrsg.) ; FREYBERGER, F. (Hrsg.): *Autonome Mobile Systeme 1999*, Springer; http://www.springer.de/, 1999 (Informatik aktuell), S. 334–343

[Röfer und Lankenau 2000] RÖFER, T. ; LANKENAU, A.: Architecture and Applications of the Bremen Autonomous Wheelchair. In: WANG, P. (Hrsg.): *Information Sciences* Bd. 1-4. Elsevier Science BV; http://www.elsevier.nl/, 2000, S. 1–20

[Röfer und Lankenau 2002] RÖFER, T. ; LANKENAU, A.: Route-Based Robot Navigation. In: *Künstliche Intelligenz - Themenheft Spatial Cognition* (2002), S. 29–31

[Röfer und Müller 1998] RÖFER, T. ; MÜLLER, R.: Navigation and Routemark Detection of the Bremen Autonomous Wheelchair. In: LÜTH, T. (Hrsg.) ; DILLMANN, R. (Hrsg.) ; DARIO, P. (Hrsg.) ; WÖRN, H. (Hrsg.): *Distributed Autonomous Robotics Systems*, Springer; http://www.springer.de/, 1998, S. 183–192

[Ross u. a. 2004] ROSS, R. ; SHI, H. ; VIERHUFF, T. ; KRIEG-BRÜCKNER, B. ; BATEMAN, J.: Towards Dialogue Based Shared Control of Navigating Robots. In: FREKSA, C. (Hrsg.) ; KNAUFF, M. (Hrsg.) ; KRIEG-BRÜCKNER, B. (Hrsg.) ; NEBEL, B. (Hrsg.) ; BARKOWSKY, T. (Hrsg.): *Spatial Cognition 2004* Bd. 3343, Springer-Verlag; D-69121 Heidelberg, Germany; http://www.springer.de, 2004

[Schlegel 1998] SCHLEGEL, C.: Fast Local Obstacle Avoidance under Kinematic and Dynamic Constraints for a Mobile Robot. In: *In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* 1 (1998), Oktober, S. 594–599

[Siegwart u. a. 2003] SIEGWART, R. ; ARRAS, K.O. ; BOUABDALLAH, S. ; BURNIER, D. ; FROIDEVAUX, G. ; GREPPIN, X. ; JENSEN, B. ; LOROTTE, A. ; MAYOR, L. ; MEISSER, M. ; PHILIPPSEN, R. ; PIGUET, R. ; RAMEL, G. ; TERRIEN, G. ; TOMATIS, N.: Expo.02: A large-scale installation of personal robots. In: *Robotics and Autonomous Systems* 42 (2003), März, S. 203–222

[Siemens 2003] SIEMENS: *SIGUARD Laserscanner für Personenschutz und Meßaufgaben, Technische Produktbeschreibung*. Siemens AG, 2003

[Simmons 1996] SIMMONS, Reid: The curvature-velocity method for local obstacle avoidance. In: *In Proceedings of the IEEE International Conference on Robotics and Automation* 4 (1996), S. 3375–3382

[Stachniss und Burgard 2002] STACHNISS, Cyrill ; BURGARD, Wolfram: An Integrated Approach to Goal-directed Obstacle Avoidance under Dynamic Constraints for Dynamic Environments. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* 1 (2002), S. 508–513

[Yasuda u. a. 2006] YASUDA, Toshihiko ; NAKAMURA, Kazushi ; KAWAHARA, Akihiro ; TANAKA, Katsuyuki: Neural network with variable type

connection weights for autonomous obstacle avoidance on a prototype of six-wheel type intelligent wheelchair. In: *International Journal of Innovative Computing, Information and Control* 2 (2006), Nr. 5, S. 1165–1177

## Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt, dass die vorliegende Diplomarbeit selbständig von mir verfasst wurde. Dabei habe ich keine anderen Quellen und Hilfsmittel benutzt als die angegebenen. Alle wörtlichen oder sinngemäß den Schriften anderer entnommenen Stellen sind unter Angabe der Quellen kenntlich gemacht.

Die Arbeit war noch nicht Gegenstand eines anderen Prüfungsverfahrens.

Bremen, im Mai 2007