P r o j e k t d o k u m e n t a t i o n

Projekttitel:	IndiaNIBO Jones
Hochschule:	Technische Hochschule Wildau
Studiengang:	Dualer praxisintegrierender Bachelor Studiengang Telematik
Seminargruppe:	T16
Semester:	5. Semester
Modul:	Eingebettete Systeme und Robotik
Dozent:	Prof. Dr. rer. nat. Janett Mohnke
Bearbeitungszeitraum:	31.10.2018 bis 07.12.2018
Projektmitglieder:	Tanke, Hendrik (50032925) Petschelt, Benedikt (50033047) Mitzlaff, Philip (50043376)
Datum der Einreichung:	07.12.2018

1. Allgemeine Projektbeschreibung

1.1 Motivation

Die Durchführung des Projektes soll dem Projektteam zeigen, wie mit eingebetteten Systemen umgegangen werden muss. Besonderer Fokus liegt hierbei auf die Vorgehensweise mit den zur Verfügung stehenden Ressourcen, die, im Gegensatz zu den bisherigen Entwicklungsumgebungen, äußerst schonend benutzt werden müssen. Des Weiteren wird das Projekt in der Programmiersprache C geschrieben. Da C, im Unterschied zu der bisher hauptsächlich genutzten Sprache Java, sehr hardwarenah geschrieben werden kann und dazu nicht objektorientiert ist, wird das Projektteam vor weitere neue Herausforderungen gestellt.

1.2 Projektziel

Ziel des Projekts ist es innerhalb von 6 Wochen, beginnend ab dem 31.10.2018, einen NIBO2 zu programmieren, sodass dieser die ihm aufgetragenen Aufgaben absolvieren kann. Die zwei zu bewältigenden Aufgaben werden nachfolgend Pflicht- und Kürteil genannt und werden in dem Dokument getrennt voneinander betrachtet. Beide Aufgabenbereiche sind Bestandteil des Projekts “IndiaNIBO Jones”.

1.2.1 Projektziel Pflichtteil

Ziel des Pflichtteils ist es eine Anwendung zu schreiben, die es dem NIBO-Roboter ermöglicht sich frei im Raum zu bewegen und Hindernissen selbstständig auszuweichen. Dabei sollen mindestens die Motoren, Distanzsensoren, LEDs und das Display des NIBO genutzt werden.

1.2.2 Projektziel Kürteil

Ziel des Kürteils ist die Entwicklung eines Programms, das dem NIBO ermöglicht sich selbständig entlang der Raumbegrenzung einer aufgebauten, geschlossenen Struktur zu bewegen. Gleichzeitig soll der NIBO Daten über die zurückgelegte Strecke per XBee mit einem Computer auszutauschen. Anhand der Daten soll ein proportionales Abbild des Grundrisses der Struktur auf dem NIBO-Display und auf dem Computer erzeugt werden.

1.3 Projektrahmen

1.3.1 Projektrahmen Pflichtteil

Nach Betätigung des Einschalters beginnt der NIBO sich selbstständig im Raum zu bewegen, indem er, vorausgesetzt er sieht kein Hindernis, geradeaus fährt. Sehen die Distanzsensoren auf der rechten Seite ein Hindernis, so weicht er nach links aus und ist auf der linken Seite des NIBOs ein Hindernis, so weicht er nach rechts aus. Befindet sich auf beiden Seiten ein Hindernis, so hält er sich selbstständig mittig der beiden Hindernisse, bis

entweder keine Hindernisse mehr vorhanden sind und er wieder frei im Raum fahren kann oder zusätzlich noch ein frontales Hindernis erscheint. Daraufhin erkennt der NIBO eine Sackgasse, stoppt und fährt anschließend so lange rückwärts, bis auf mindestens einer der Seiten kein Hindernis mehr zu erkennen ist und wendet dann in die jeweils freie Richtung ab. Das Rückwärtsfahren wird mithilfe der zwei hinteren auf orange geschalteten LEDs signalisiert. Eine seitliche LED zeigt jeweils die Drehrichtung an. Fährt der NIBO frontal auf ein Hindernis zu, bleibt er stehen und prüft die Seiten. Darauffolgend weicht er in die Richtung, deren Sensorwerte kleiner sind aus. Zusätzlich werden die Abstände zu den Hindernissen visuell auf dem Display dargestellt. Eine Batteriezustandsanzeige auf dem Display gibt Aufschluss über die einwandfreie Funktion der Anwendung, denn ab einer zu niedrigen Spannung liefern die Distanzsensoren unplausible Werte zurück und der NIBO kann Hindernissen womöglich nicht mehr rechtzeitig ausweichen.

1.3.2 Projektrahmen Kürteil

Der Startpunkt des NIBOs wird durch eine schwarze Linie auf dem Boden markiert. Ab diesem Punkt beginnt der NIBO die Fahrdaten zu ermitteln. Der Grundriss gilt als vollständig abgefahren, wenn der NIBO diese schwarze Markierung ein zweites Mal überfährt. Der NIBO muss zum Start einer Fahrt am Rand der Struktur platziert werden. Daraufhin wird er selbständig messen, an welcher Seite er sich orientieren muss. Während der Fahrt wird der NIBO versuchen einen Mindestabstand von 5 cm zur Wand einzuhalten. Dadurch sollen mögliche Kollisionen vermieden werden. Pro vollständigen Umriss darf nur eine schwarze Markierung existieren. Der Grundriss darf eine maximale Fläche von 3m² nicht überschreiten, da die Leistung der Akkus nicht ausreichend ist, um größere Strecken zu absolvieren. Dabei darf der Grundriss über keine Sackgassen oder andere komplexere Hindernisse verfügen, da sonst die visuelle Darstellung nicht mehr zu gewährleisten wäre. Die Ecken des Grundrisses müssen immer einen Winkel von 90° aufweisen. Andere Winkel können nicht vom NIBO selbständig gemessen werden. Zudem muss ein minimaler Abstand zwischen den Ecken von mindestens 30 cm bestehen. Dieser Mindestabstand soll gewährleisten, dass der NIBO eine 180° Drehung ausführen kann. Der Grundriss muss eine durchgehende Wand ohne Lücken sein. Während der Fahrt speichert sich der NIBO lokal Daten zu den gefahrenen Strecken ab. Die Strecken beginnen und enden mit einem Richtungswechsel. Wenn ein Richtungswechsel durchgeführt wird, dann werden die Daten der Odometrie ausgelesen, gespeichert und zurückgesetzt. Dabei werden die Daten der Odometrie als virtuelle Koordinaten interpretiert, wobei 5 Ticks einer Längeneinheit entsprechen. Durch die Abstrahierung soll intern sofort die gefahrene Streckenlänge berechnet werden, wodurch sowohl weniger Speicherplatz benötigt als auch die Visualisierung vereinfacht wird. Mithilfe des XBee-Moduls überträgt er die Daten regelmäßig an einen verbundenen Computer. Die Daten werden von dem Computerprogramm so verarbeitet, dass der Umriss nach jedem Streckenabschnitt aktualisiert dargestellt werden kann. Nach Beendigung der Fahrt werden die Daten für den NIBO aufbereitet und anschließend an diesen übertragen, sodass der Umriss auch auf dem Display ausgegeben werden kann.
Das im Laufe dieses Projekts entwickelte Programm zur Darstellung des Grundrisses auf dem Computer trägt den Namen “NIBOBrody”. Das Programm für den NIBO trägt den Namen “IndiaNIBOJones”.

Der NIBO ist in der Lage, den Grundriss im und entgegen dem Uhrzeigersinn abzufahren. Je nach Umlaufrichtung dreht der Nibo bei einem frontalen Hindernis nach links oder rechts ab. Diese Problematik lässt sich auch auf weitere Aspekte des Projekts übertragen. Zur Vereinfachung wird in diesem Dokument von “äußeren” und “inneren” bzw. nach außen und nach innen gesprochen. In Kombination mit der Umlaufrichtung ergibt sich dann sinngemäß ein links oder rechts. Die Begriffe beziehen sich dabei immer auf den Mittelpunkt des Grundrisses. Dementsprechend sind die äußeren Sensoren der Wand zugewandt, an der sich der NIBO orientiert und die inneren Sensoren zeigen Richtung Mittelpunkt.

2. Anforderungsanalyse

2.1 Liste der Anforderungen

2.1.1 Anforderungsliste Pflichtteil

siehe Anhang 7.1 Anforderungsliste Pflichtteil

2.1.2 Anforderungsliste Kürteil

siehe Anhang 7.2 Anforderungsliste Kürteil

2.2 Liste der verwendeten Komponenten des NIBOs

2.2.1 Verwendet Komponenten im Pflichtteil

Distanzsensoren (DS)
Die Distanzsensoren des NIBO-Roboters werden innerhalb der Anwendung stetig abgefragt. Dabei haben sich für die jeweiligen Sensoren verschiedene Threshold-Werte (Grenzwerte) herausgestellt. Diese sind sowohl abhängig von dem Umgebungslicht, der Farbe als auch der Oberfläche des zu erkennenden Hindernisses. Ausgehend von der normalen Raumbeleuchtung der Telematik - Labore und den zur Verfügung gestellten matt-weißen Hindernissen liegen die normalisierten Grenzwerte für den frontalen Sensor bei 100, für die seitlichen beiden Sensoren bei 50 und für die jeweils dazwischenliegenden, diagonalen Sensoren bei 90. Zusätzlich zum rechtzeitigen Erkennen von Hindernissen werden die Distanzwerte auch für die visuelle Darstellung auf dem Display verwendet. Dort ist ein Umriss des NIBO-Roboters zu sehen. Angelehnt an eine Park-Distance-Control eines Autos zeigt jeweils ein Strich ein Hindernis und somit den Abstand zu diesem an. Dieser Strich wandert jeweils in Richtung des NIBO-Roboter-Umrisses, wenn das Hindernis näher kommt bzw. der NIBO sich dem Hindernis nähert. Insgesamt repräsentieren fünf solcher Striche die einzelnen Sensoren.

Odometrie (OD)

Die Odometrie und damit verbunden die Ansteuerung der Motoren, wird für das freie Fahren im Raum benötigt. Abhängig von den verschiedenen Hinderniskonstellationen müssen die Motoren stoppen können, mit gleicher sowie auch mit unterschiedlicher Drehzahl betrieben werden können. Angegeben wird die Drehzahl der Motoren in Ticks pro Sekunde (Ticks/s). Ein Tick wird gezählt, wenn ein Loch des Plastikzahnrades, über welches das Rad angetrieben wird, an der Infrarotdiode vorbeikommt und das Signal vom Infrarotempfänger somit gemessen werden kann. 29 Ticks/s entsprechen ungefähr 20 cm/s. Soll der NIBO einem Hindernis ausweichen, also vor diesem abbiegen, so wird je nachdem in welche Richtung ausgewichen werden soll einer der beiden Motoren genau entgegengesetzt angesteuert. Beim Rückwärtsfahren drehen dann beide Motoren in die andere Richtung. Umgesetzt wird dies durch die Angabe einer negativen Anzahl von Ticks/s.

Status-LEDs (SL)
Der NIBO besitzt pro Distanzsensor eine LED und zusätzlich zwei weitere LEDs am Heck. Diese lassen sich in den Farben grün, orange und rot ansteuern. Weiterhin gibt es zwei sogenannte Frontscheinwerfer, die weiß leuchten und dimmbar sind. In diesem Projekt wird lediglich auf die Funktionalitäten der LEDs zurückgegriffen, die Abbiege- bzw. Wendevorgänge und Rückwärtsfahrten durch oranges Aufleuchten visualisieren. Zugunsten der Akkulaufzeit wird auf die Frontscheinwerfer verzichtet.

Display-Modul (DM)
Das Display-Modul des NIBO-Roboters besteht aus einem 128x64 Pixel großen monochromen Display ohne Graustufen. Auf diesem lassen sich alle Pixel einzeln über die Koordinaten x und y ansprechen, löschen und überschreiben. Somit ist es möglich anhand der Distanzwerte und gegebenenfalls benötigter Skalierung die erkannten Hindernisse darzustellen.

2.2.2 Verwendete Komponenten im Kürteil

Distanzsensoren (DS)
Die Distanzsensoren werden für mehrere Funktionen benötigt. Zunächst soll der NIBO anhand der äußeren Distanzsensoren in der Lage sein, einer Wand zu folgen. Dazu misst der äußere Distanzsensor den Abstand zur Führungswand und gibt mit den zurückgelieferten Werten darüber Aufschluss, ob der NIBO näher an die Wand heran oder weiter von der Wand wegfahren muss.
Weiterhin soll der NIBO anhand der Distanzwerte selbstständig entscheiden, ob und in welche Richtung der NIBO abdrehen soll. Sollte eine Wand frontal vor dem NIBO erkannt werden, dann dreht dieser nach innen ab. Sollte der äußere Distanzsensor keine Führungswand mehr erkennen, dann dreht der NIBO nach außen ab.
Die durchgeführten Tests haben gezeigt, dass die Sensorwerte stark schwanken und Einzelwerte nicht repräsentativ für den Abstand zur Führungswand sind. Bei der Erkennung von frontalen Hindernissen oder dem allgemeinen Kontakt zur Führungswand, sind die Sensorwerte ausreichend.

Bodensensoren (BS)

Mit Hilfe der vorderen Bodensensoren soll eine schwarze Startlinie erkannt werden. Beim erstmaligen Erkennen der Linie wird das Senden von Daten aktiviert. Beim nächsten Erkennen einer schwarzen Linie, wird dieser Prozess beendet. Mit Hilfe dieser Funktion kann der Beginn und das Ende des Umrisses klar definiert werden.
Die Tests haben gezeigt, dass schwache Farbunterschiede schwer unterschieden werden können. Starke Kontraste können dagegen sehr gut erkannt werden. Auch kleine oder kurze Markierungen können schnell übersehen werden.
Aus diesem Grund wurde als Markierung eine schwarze Linie mit mehr als einen halben Zentimeter Dicke gewählt.

Odometrie (OD)
Die Odometrie wird benötigt, um den NIBO an der Wand entlang zu bewegen. Dabei wird ausschließlich von einer Vorwärtsbewegung ausgegangen. Teil dieser Vorwärtsbewegung sind 90° Wenden nach links oder rechts und kleinere Manöver zur Korrektur der entstehenden Ungenauigkeiten durch potenzielle Unterschiede der Räder.
Des Weiteren wird auf die Tickzähler des Co-Prozessors zugegriffen, um Daten über zurückgelegte Strecke zu erhalten.
Die Tests haben gezeigt, dass der NIBO auf glatten Untergründen nicht in der Lage ist die programmierte Bewegung zu 100 % auszuführen. Besonders das Fahren der 90° Wenden war nie exakt. Des Weiteren fiel auf, dass der Nibo bei zu hohen Geschwindigkeiten nicht mehr in der Lage ist, Hindernisse schnell genug zu erkennen.

XBee-Modul (XM)
Die verwendeten XBee-Module ermöglichen den Datenaustausch zwischen einem Computer und den NIBOs. Dazu gehören sowohl das Senden von Daten, als auch das Empfangen. Das am Computer angeschlossene XBee-Modul ist primär dafür zuständig die Daten zu den abgefahrenen Streckenabschnitten zu empfangen. Gleichermaßen werden dem NIBO, der den Umriss abgefahren ist, die abschließende Skizze übergeben, sodass dieser ebenfalls den Grundriss auf dem Display ausgeben kann.
Des Weiteren sendet der Computer ein Startsignal an einen weiteren NIBO, sobald der NIBO in dem Umriss erstmals auf eine Bodenmarkierung stößt.
Die Tests haben gezeigt, dass die XBee-Kommunikation nur ein Byte auf einmal übertragen kann. Des Weiteren viel in den Tests auf, dass nicht jede Übertragung auch wirklich durchkam.

Display-Modul (DM)
Auf dem Display des NIBOs soll anhand empfangener Daten zum Ende des Protokolls der Raumerfassung der Grundriss des Raums angezeigt werden. Des Weiteren kann auf diesem Display ein Konterfei von Indiana Jones angezeigt werden, was einen Hinweis darauf gibt, dass der NIBO begonnen hat, den Grundriss des Raums zu erfassen.

Status-LED (SL)
Der NIBO kann mit Hilfe der Status-LEDs Richtungswechsel und Bremsvorgänge anzeigen. Richtungswechsel werden durch oranges Leuchten auf der entsprechenden Seite, in die der NIBO abdreht, angezeigt. Die Bremsvorgänge werden mit zwei roten Status-LEDs auf der

Rückseite des NIBOs angezeigt. Bremsvorgänge treten immer kurz vor den 90° Wenden auf.

Tonausgabe-Modul (TM)
Mit Hilfe der Tonausgabe kann ein weiterer NIBO eine 8 Bit-Interpretation des Titelsongs von Indiana Jones abspielen. Ausgelöst wird diese Aktion durch den Computer, sobald der aktive NIBO erstmals über eine Bodenmarkierung fährt. Wenn die Markierung ein zweites Mal überfahren wird, dann beendet sich der Prozess.

3. Konzeptentwicklung

3.1 Pflichtteil

3.1.1 Zustandstabelle

Die Zustandsmaschine des Pflichtteils basiert auf der Erkennung von Hindernissen, mit Hilfe der Distanzsensoren. Insgesamt kann der NIBO sechs unterschiedliche Zustände annehmen.

Anmerkung:
Den Distanzsensoren wurden folgende Namen zugeordnet:

- Der seitliche rechte Sensor entspricht der ID 0
- Der vordere rechte Sensor entspricht der ID 1
- Der frontale Sensor entspricht der ID 2
- Der vordere linke Sensor entspricht der ID 3
- Der seitliche linke Sensor entspricht der ID 4

Aktueller Zustand	**Ereignis**	**Aktion**	**Folgezustand**
NO_OBSTACLE	Kein Hindernis erkannt.	Der NIBO fährt geradeaus.	NO_OBSTACLE
	Der frontale Sensor erkennt ein Hindernis.	Der NIBO bleibt stehen.	OBSTACLE_FRONT
	Der vordere linke Sensor erkennt ein Hindernis.	Der NIBO wechselt die Richtung.	OBSTACLE_LEFT
	Der vordere rechte Sensor erkennt ein Hindernis.	Der NIBO wechselt die Richtung.	OBSTACLE_RIGHT
	Die seitlich angebrachten	Der NIBO fährt geradeaus.	OBSTACLE_SIDES

	Sensoren erkennen ein Hindernis.		
OBSTACLE_FRONT	Der frontale Sensor überschreitet den Grenzwert.	Der NIBO dreht sich so lange, bis kein Hindernis mehr erkannt wird.	NO_OBSTACLE
	Der frontale und die seitlich angebrachten Sensoren überschreiten die Grenzwerte.	Der NIBO fährt rückwärts.	OBSTACLE_DEADEND
OBSTACLE_LEFT	Der vordere linke Sensor überschreitet den Grenzwert.	Der NIBO fährt eine dynamische Rechtskurve.	OBSTACLE_LEFT
	Der frontale und der vordere linke Sensor erkennen keine Hindernisse mehr.	Der NIBO bleibt stehen.	NO_OBSTACLE
OBSTACLE_RIGHT	Der vordere rechte Sensor überschreitet den Grenzwert.	Der NIBO fährt eine dynamische Linkskurve.	OBSTACLE_RIGHT
	Der frontale und der vordere rechte Sensor erkennen keine Hindernisse mehr.	Der NIBO bleibt stehen	NO_OBSTACLE
OBSTACLE_SIDES	Die seitlich angebrachten Sensoren überschreiten den Grenzwert.	Der NIBO fährt geradeaus.	OBSTACLE_SIDES
	Der vordere linke Sensor erkennt ein Hindernis.	Der NIBO wechselt die Richtung.	OBSTACLE_LEFT
	Der vordere	Der NIBO wechselt die	OBSTACLE_RIGHT

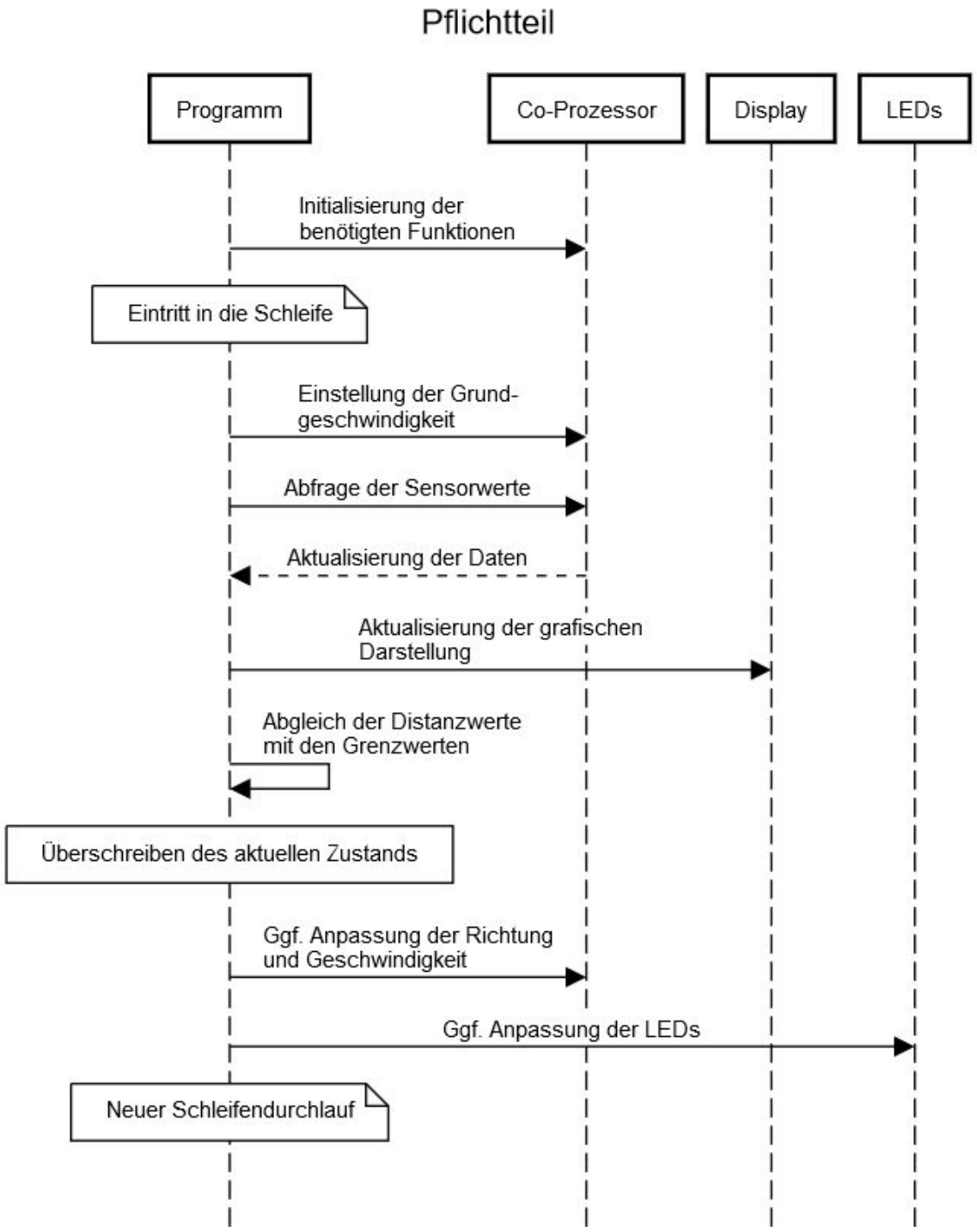
	rechte Sensor erkennt ein Hindernis.	Richtung.	
	Kein Hindernis erkannt.	Der NIBO fährt geradeaus.	NO_OBSTACLE
	Der frontale und die seitlich angebrachten Sensoren überschreiten die Grenzwerte.	Der NIBO fährt rückwärts.	OBSTACLE_DEADEND
OBSTACLE_DEADEND	Der frontale und die seitlich angebrachten Sensoren überschreiten die Grenzwerte.	Der NIBO fährt rückwärts.	OBSTACLE_DEADEND
	Die seitlichen Sensoren überschreiten den Grenzwert.	Der NIBO fährt rückwärts.	OBSTACLE_DEADEND
	Mindestens einer der seitlichen Sensoren liegt unterhalb des Grenzwertes.	Der NIBO fährt für eine weitere Sekunde rückwärts, und dreht in die freie Richtung ab.	NO_OBSTACLE

3.1.2 Ressourcensparende Datenstrukturen

Komplexe Datenstrukturen sind im Pflichtteil nicht zum Einsatz gekommen. Die einzige Ausnahme stellt der genutzte Array der Distanzwerte an, der beim aktualisieren des Co - Prozessor neu belegt wird. Da allerdings immer auf spezifische Elemente dieses Arrays zugegriffen wird, kann die dafür benötigte Zeit und Rechenleistung vernachlässigt werden.

3.1.3 Sequenzdiagramm

Anmerkung: Zur besseren Übersicht wurden die Funktionalitäten der Odometrie und die der Distanzsensoren unter dem Feld “Co-Prozessor” zusammengefasst.



3.2 Kürteil

3.2.1 Zustandstabelle

Die Zustandsmaschine des Kürteils basiert auf dem Manöver, welches der NIBO gerade ausführt. Abhängig von dem Manöver und dem eintretenden Ereignis wechselt der NIBO gegebenenfalls in einen neuen Zustand und führt das spezifische Manöver aus.

Anmerkungen:

Der Messstatus ist ein Flag, das darüber Aufschluss gibt, ob der NIBO gerade Daten über den Grundriss sammelt.
Im Folgenden wird von frontalen und seitlichen Distanzsensoren gesprochen. Der frontale Distanzsensor ist der Sensor mit der ID 2 auf der Platine des NIBOs. Bei den seitlichen Distanzsensoren kann es sich je nach Umlaufrichtung um den Sensor mit der ID 0 oder 4 handeln.

Aktueller Zustand	**Ereignis**	**Aktion**	**Folgezustand**
WAITING	- Zustand-wechsel	- Anzeigen des Tooltips	WAITING
	- Taster S3 gedrückt, einmalig	- Löschen des Tooltips	INITIALIZATION
	- Taster S3 gedrückt, sonst immer	- Löschen des Tooltips	RUNNING_FORWARD
INITIALIZATION	- Zustand-wechsel	Ermittlung der Laufrichtung	RUNNING_FORWARD
RUNNING_FORWARD	- weder Wand frontal voraus noch fehlt der Kontakt zur Führungswand	NIBO fährt geradeaus.	TRACK_CORRECTION
	überfahren einer schwarzen Linie & Messstatus ist gestoppt	- Umschalten von Messstatus von gestoppt auf gestartet. - Zurücksetzen der Tickzähler des Coprozessors	SENDING_DATA
	überfahren einer schwarzen Linie & Messstatus ist gestartet	- Umschalten von Messstatus von gestartet auf gestoppt. - Auslesen der Tickzähler des Coprozessors für beide Räder, ermitteln des Durchschnittswert und abspeichern	SENDING_DATA
	Der frontale Distanzsensor	NIBO bremst.	TURN_INSIDE

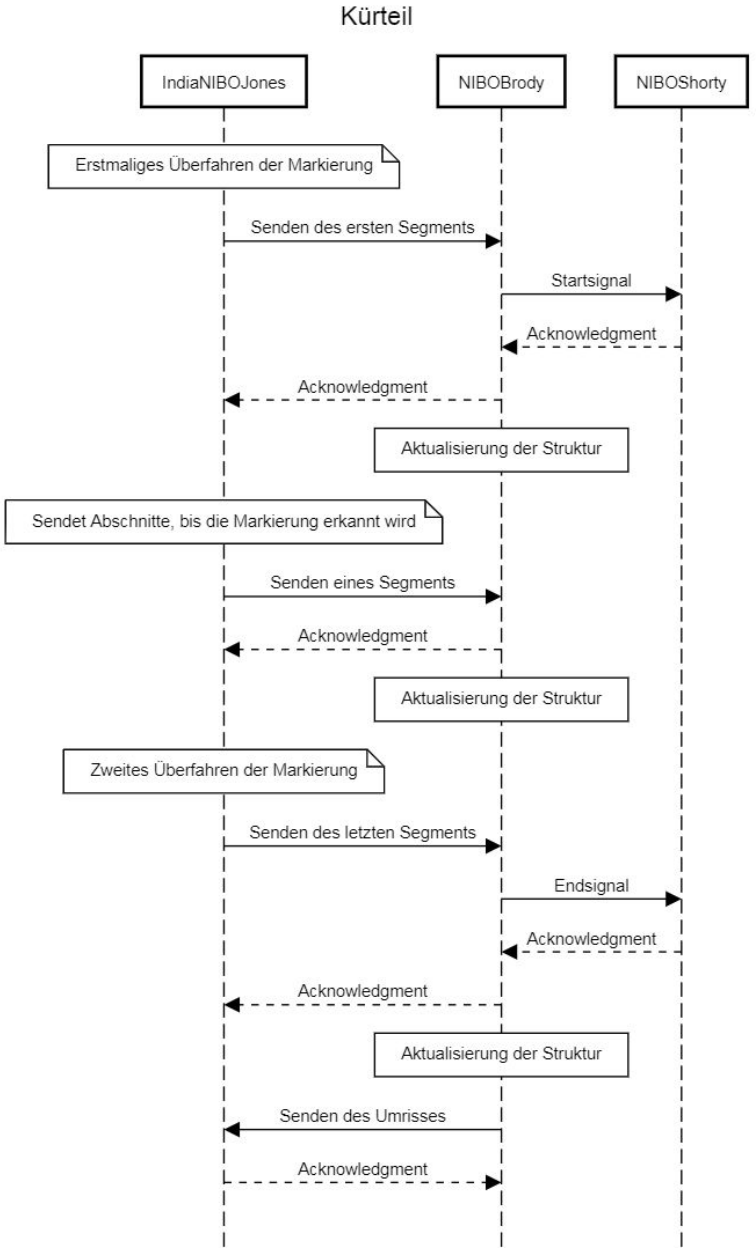
	erkennt ein Hindernis		
	Der seitliche Distanzsensor erkennt fehlende Führungs-wand	NIBO fährt weiter zwei Sekunden gerade aus und stoppt dann.	TURN_OUTSIDE
	Taster S3 gedrückt	NIBO stoppt.	WAITING
TRACK_CORRECTION	Der Wert des seitlichen Distanzsensors unterschreitet den Maximalabstand zur Führungswand	- Bewegungs-richtungs-korrektur zur Führungswand hin	RUNNING_FORWARD
	Der Wert des seitlichen Distanzsensors überschreitet den Minimalabstand zur Führungswand	- Bewegungs-richtungs-korrektur von der Führungswand weg	RUNNING_FORWARD
TURN_INSIDE	- Messstatus ist gestartet	- liest die Tickzähler des Coprozessors für beide Räder aus, ermittelt den Durchschnittswert und speichert ab - NIBO biegt nach Innen ab (im 90° Winkel)	SENDING_DATA
	- Messstatus ist gestoppt	- NIBO biegt nach Innen ab (im 90° Winkel)	RUNNING_FORWARD
TURN_OUTSIDE	- Messstatus ist gestartet	- liest die Tickzähler des Coprozessors für beide Räder aus, ermittelt den Durchschnittswert	SENDING_DATA

		und speichert ab - NIBO biegt nach Außen ab (im 90° Winkel)	
	- Messstatus ist gestoppt	- NIBO biegt nach Außen ab (im 90° Winkel)	RUNNING_FORWARD
SENDING_DATA	Senden ermittelter Werte an NIBOBrody erfolgreich	- zurücksetzen der Tickzähler des Coprozessors für beide Räder.	RUNNING_FORWARD
	Senden ist fehlgeschlagen	-	SENDING_DATA
	- Messtatus ist gestoppt - Senden ermittelter Werte an NIBOBrody erfolgreich	- zurücksetzen der Tickzähler des Coprozessors für beide Räder.	FINISHING_RUN
FINISHING_RUN	- Empfangen der Daten von Brody erfolgreich	- Anzeigen des Grundrisses anhand der empfangen Daten vom NIBOBrody	WAITING

3.2.2 Ressourcensparende Datenstrukturen

Der IndiaNIBOJones kann ebenfalls auf das Anlegen von Datenstrukturen verzichten. Die benötigten Angaben zu der Odometrie, den Distanzwerte und den Bodenwerten, können vollständig über den Co-Prozessor abgefragt werden. Dementsprechend verfällt die Notwendigkeit diese separat abzuspeichern.
Der NIBOBrody hingegen, verwendet sowohl eine Structure, ein Array, als auch eine doppelt verkettete List. Die Structure repräsentiert ein Segment des Umrisses. Sie beinhaltet dabei eine Angabe zu der Richtung, der Länge, dem vorherigen Element und dem nachfolgendem Element. Das Array beinhaltet den eigentlichen Umriss. Er wird mit jedem neuen Segment neu allokiert und anschließend befüllt. Da die Ausgabe über die Konsole erfolgt und die abschließende Größe der Struktur nicht absehbar ist, eignete sich ein Array am besten. Die Liste hingegen ermöglicht theoretisch unbegrenzt viele Segmente anzuhängen. Im Gegensatz zu dem Array muss also nur für ein neues Segment der Speicherplatz reserviert werden. Da der NIBOBrody auf einem Computer ausgeführt wird, sind die Ressourcen- und Leistungsbegrenzungen allerdings eher eine nebensächliche Angelegenheit.

3.2.3 Sequenzdiagramm



4. Umsetzung

4.1 Pflichtteil

Da die durchzuführenden Aktionen des NIBOs vollständig davon abhängen, in welchem Zustand er sich aktuell befindet, musste zu Beginn die Zustandsmaschine implementiert werden. Dafür wurden die annehmbaren Zustände in dem Enumerator *STATES* abgespeichert. Im gleichen Zuge wurde eine globale Variable *currentState* definiert, die immer den derzeitigen Zustand beinhaltet.
Die Kernkomponente des Programms besteht allerdings aus einer endlosen *While-Schleife*. Damit alle benötigten Komponenten des NIBOs funktionsfähig sind, mussten diese vor den Eintritt in die Endlosschleife initialisiert und gegebenenfalls gestartet werden. Anschließend

wurden in jedem Schleifendurchlauf die aktuellen Sensordaten des Co-Prozessors abgefragt. Die Distanzwerte wurden nachfolgend über *If-Else-Zweige* mit den zuvor getesteten und festgelegten *Grenzwerten* verglichen. Sobald sich durch die Abfrage eine Überschreitung herausgestellt hat, wurde der currentState aktualisiert.
Nachfolgend musste, abhängig vom derzeitigen Zustand, die durchzuführende Aktion des NIBOs gewählt werden. Dafür wurde auf eine *Switch-Case-Anweisung* zurückgegriffen.

4.1.1 Betrachtung umgesetzter Anforderungen

Folgende Anforderungen mit MUSS-Kriterium wurden umgesetzt:

- P-01, P-02, P-03, P-04, P-05, P-06, P-07, P-08, P-09, P-10, P-11, P-12, P-13, P-14, P-15

Folgende Anforderungen mit KANN-Kriterium wurden umgesetzt:

- keine

Folge Anforderungen mit MUSS-Kriterium wurden nicht umgesetzt:

- keine

Folgende Anforderungen mit KANN-Kriterium wurden nicht umgesetzt:

- keine

4.1.2 Gesamttest Pflichtteil

Der Abschlusstest des Pflichtteils des Projekts IndiaNIBOJones wurde im Raum A107 in der Halle 14 der Technischen Hochschule durchgeführt. Dazu wurden mit mehreren mattweißen Brettern bestimmte Szenarios nachgebildet, um alle Funktionen bzw. Reaktionen des Nibo-Roboters zu testen. Getestet wurde dabei, ob der Nibo sich innerhalb einer geschlossenen Struktur bewegen kann, ohne an ein Hindernis zu stoßen. Des Weiteren wurde mit zwei parallel zueinander stehenden Brettern ein Tunnel nachgebildet, um zu prüfen, ob der Nibo innerhalb des Tunnels selbstständig seinen Kurs korrigiert und mittig innerhalb des Tunnels fährt. Danach wurde der Tunnel mithilfe eines dritten Bretts senkrecht zu den beiden anderen zu einer Sackgasse umfunktioniert. Der Nibo soll diese erkennen, stoppen und solange rückwärts fahren, bis auf mindestens einer der beiden Seiten kein Hindernis mehr zu sehen ist. Außerdem wurde jeweils ein Brett rechts und links vor dem Nibo aufgestellt, um zu überprüfen, ob dieser dynamisch nach links bzw. nach rechts ausweicht.
Der Nibo hat die soeben aufgezählten Tests bestanden und somit auch den Gesamttest erfolgreich abgeschlossen.

4.2 Kürteil

4.2.1 Betrachtung umgesetzter Anforderungen

Folgende Anforderungen mit MUSS-Kriterium wurden umgesetzt:

- für IndiaNIBOJones:
  IJ-01, IJ-02, IJ-03, IJ-04, IJ-05, IJ-06, IJ-07, IJ-08, IJ-09, IJ-10, IJ-11, IJ-12, IJ-13, IJ-14, IJ-15, IJ-16

- für NIBOBrody:
  NB-01, NB-02, NB-03, NB-04, NB-05, NB-06

Folgende Anforderungen mit KANN-Kriterium wurden umgesetzt:
- für IndiaNIBOJones:
  IJ-17, IJ-18, IJ-19
- für NIBOShorty:
  NS-01, NS-02, NS-03, NS-04

Folge Anforderungen mit MUSS-Kriterium wurden nicht umgesetzt:
- keine

Folgende Anforderungen mit KANN-Kriterium wurden nicht umgesetzt:
- keine

4.2.2 Besondere Entscheidungen

Die Grundlage für das Programm IndiaNIBOJones ist eine Zustandsmaschine.
Die Zustandsmaschine besteht aus einer unendlichen While-Schleife und einer Switch-Case-Anweisung, die die einzelnen Zustände ansteuert. Die Gesamtheit aller Funktionen, die zu einem Zustand gehört, wurden Protokolle genannt. Im Quelltext äußert sich das als eine Methode mit Postfix "_protocol", die in der Switch-Case-Anweisung aufgerufen wird.

Des Weiteren fiel während der Entwicklung auf, dass die Gesamtheit aller Methoden und Hilfsmethoden zur Darstellung der Protokolle einen Großteil des Quellcodes einnahmen. Es wurde entschieden, diese Methoden in eine eigene C-Datei (protocols.c) samt Header-Datei zu verschieben, um die Haupt-Datei mit der Main-Methode im Umfang zu entlasten.
Im Laufe der Implementation wurden eine handvoll von Methoden entworfen, um bei Tests eine Ausgabe zu generieren. Da die meisten Methoden nicht für den Normalbetrieb des NIBOs benötigt werden, wurden alle Ausgabemethoden in eine weitere Datei gfxOutput.c mit dazugehöriger Header-Datei ausgelagert.

4.2.3 Übertragungsprotokoll

Ein weiteres Problem stellte die äußerst begrenzte Übertragungskapazität der XBee - Module dar. Da insgesamt nur 8 Bits als Nutzdaten zur Verfügung stehen, mussten die benötigten Daten diesen Voraussetzungen angepasst werden. Für die Darstellung der Segmente auf dem NIBOBrody wurden insgesamt drei Parameter benötigt:

a) Ein Flag das angibt, ob der NIBO eine Markierung überfahren hat. Dieses Flag wird benötigt, damit das Programm schlussendlich alle genutzten Ressourcen wieder freilegen kann. Zur Übertragung der Information wird nur ein einzelnes Bit gebraucht, wobei die 1 eine erkannte Markierung repräsentiert und bei einer 0 nichts erkannt wurde.

b) Ein weiteres Flag muss angeben, in welche Richtung der NIBO abgebogen ist, nachdem er ein neues Segment betreten hat. Die Information ist ein essenzieller Bestandteil für die Darstellung des Umrisses. Diese Problematik kann ebenfalls mit einem Bit gelöst werden. Eine übertragene 1 gibt an, dass der NIBO nach links abgebogen ist und eine 0 nach rechts.

Die eigentliche Verarbeitung der konkreten Richtung - oben, rechts, unten, links - wird im NIBOBrody vorgenommen.

c) Fünf Bits werden für die eigentliche Strecke benötigt. Dafür wird die Odometrie des NIBOs abgefragt und durch 27 und dann noch durch 10 geteilt. Dadurch werden die abgefahrenen Ticks in Dezimeter umgewandelt. Die Umwandlung in Dezimeter ermöglicht eine maximale Segmentlänge von 3,1 Meter.

d) Da der Kürteil insgesamt zwei NIBOs beinhaltet, muss ein weiteres Flag übertragen werden, aus dem hervorgeht, welcher NIBO angesprochen werden soll. Der IndiaNIBOJones entspricht dem aktiven NIBO und wird durch eine übertragene 1 angesprochen. Im Gegensatz dazu wird der NIBOShorty durch eine 0 angesprochen.

Beispiel:

NIBO	Markierung	Richtung	Segmentlänge
1	0	1	0 0 1 0 1

Hinausgehend aus dieser Lösung können alle benötigten Informationen in einem Byte übertragen werden. Spezielle Kommunikationsabläufe, wie die Acknowledgments oder den Austausch von der Anzahl der Zeilen und Spalten benötigen nur das Flag des NIBOs und die entsprechende Information, welche ohne Probleme in 7 Bits dargestellt werden kann.

4.2.4 Gesamttest Kürteil

Der Abschlusstest des Kürtteils des Projekts IndiaNIBOJones wurde im Raum A107 in der Halle 14 der Technischen Hochschule durchgeführt. Dazu wurden mehrere einfache geschlossene Strukturen mit matt-weißen Brettern aufgebaut. An einer langen Seite dieser Strukturen wurde jeweils mit schwarzem Isolierband auf dem Boden eine Linie als Markierung angebracht. Als Strukturen wurden ein Rechteck, eine breite L-Form als auch ein Rechteck mit 2 nach innen verschobenen Ecken gewählt.
Für den Test wurde ein NIBO mit der Anwendung IndiaNIBOJones bespielt (aktiver NIBO). Ein weiterer NIBO bekam die Anwendung NiboShorty aufgespielt (passiver NIBO). Des Weiteren wurde auf einem MacBook ein VMWare Image installiert, auf dem der NIBOBrody ausgeführt wurde.
Der Gesamttest gilt als bestanden, wenn alle drei Teiltests mit den einzelnen Strukturen bestanden wurden. Ein Teiltest gilt als bestanden, wenn zum Einen der Grundriss der Struktur mit Hilfe des NIBOBrody auf dem Bildschirm des MacBooks als auch auf dem Display des aktiven NIBOs ohne Lücken oder Fehler angezeigt wird. Zum Anderen muss der passive NIBO genau im gleichen Takt als synchron mit den aktiven NIBO beginnen, die Musik zu spielen und das Bild anzuzeigen.
Dazu muss der aktive NIBO ein Mal vollständig die Struktur abfahren, die Längen der Strukturkanten messen und an den NIBOBrody übermitteln. Dieser muss die empfangen Daten verarbeiten und anzeigen. Die Daten müssen zu dem für den aktiven NIBO aufbereitet und an diesen gesendet werden. Anhand dieser Daten kann der aktive NIBO dann den Grundriss der Struktur wiedergeben. Mit Beginn des Messvorgangs muss der

aktive NIBO zu dem den passiven NIBO synchronisieren, so dass dieser bei Messungsstart die Titelmusik abspielt.

Das Projekt hatte alle 3 Teiltests gemäß den Vorgaben bestanden und somit auch den Gesamttest bestanden.

5. Fazit/Zusammenfassung

Fazit Pflichtteil

Das freie Fahren im Raum zeigt in einer wesentlich abgespeckten Variante die Herausforderungen und Gefahren des autonomen Fahrens. Allerdings sind durch die Genauigkeit der Sensoren und die nicht vorhandene Parallelität von Prozessen auf dem eingebetteten System bestimmte Grenzen relativ früh erreicht. Beispielsweise haben die Räder auf manchen Oberflächen nicht genug Grip und drehen durch. Deshalb und aufgrund der Beschaffenheit der Räder fährt der Nibo-Roboter nie zu 100% geradeaus. Außerdem geben die Distanzsensoren bei unterschiedlichen Lichtverhältnissen und Materialoberflächen von Hindernissen leicht bis stark schwankende Werte bei gleichem Abstand aus. Weiterhin wird die Distanzmessung stark von der Akkukapazität beeinflusst, sodass möglichst genaue Werte nur bei geladenem Akku zu erwarten sind. Bei höheren Geschwindigkeiten kann die Hinderniserkennung und das Stoppen bzw. Ausweichen vor diesen nicht mehr ordnungsgemäß durchgeführt werden, da sowohl die Distanzsensoren als auch die Motoren vom Coprozessor gesteuert werden und die Verarbeitungszeit der Distanzmessung und Motoransteuerung zu lange dauern würde. Der Nibo würde gegen das Hindernis stoßen.
Für den Einstieg in die Funktionalitäten des Nibo-Roboters und weiterführend für einen selbst ausgedachten Kürteil eignet sich die Pflichtaufgabe sehr gut, denn grundlegende Funktionen der wichtigsten Komponenten, wie Motoren, Distanzsensoren, LEDs und das Display müssen hierfür eingebunden werden.

Fazit Kürteil

Im Prinzip ließ sich die Aufgabe umsetzen. Trotzdem verhält sich der NIBO zum Beispiel nicht so, wie im Konzept angedacht. Das lässt sich auf die Ungenauigkeiten in der Bewegung des NIBOs zurückführen, wenn dieser zum Beispiel beim Wenden mit dem einen Rad wegrutscht. Diese Ungenauigkeit sorgt auch dafür, dass der Nibo in Schlangenlinien der Wand der aufgebauten Struktur folgt. Es erfüllt trotzdem den Zweck der Messung, aber sieht nicht besonders schön aus. Auch das XBee-Modul und die 8Bit-kommunikation sind nicht sehr komfortable und benötigt umfangreichere Kommunikationsprotokolle.

Hinweise für zukünftige Veranstaltungen

Das Akkusystem des NIBO-Roboters könnte überdacht bzw. neu entwickelt werden. So kann man statt 1,5V AA Akkus z.B. zusammenhängende LiPo-Akkus verwenden, die eventuell sogar durch einen Einschub schnell austauschbar und flexibel zu laden wären. Außerdem könnte man versuchen ein gleichzeitiges Laden und Arbeiten mit dem NIBO zu implementieren. Somit wäre das Debuggen und generell die Anwendungsentwicklung für den NIBO, abgesehen von Tests der Odometrie, effizienter.

Als nicht zuverlässig hat sich das Flashen des Nibos ergeben. Dies könnte sowohl auf das Flachbandkabel als auch auf die Anbindung des Programmieradapters an den PC zurückzuführen sein.

6. Quellen

https://elearning.th-wildau.de/pluginfile.php/226200/mod_folder/content/0/NiboTutorial.pdf

7. Anhang

7.1 Anforderungsliste Pflichtteil

ID	Beschreibung	Komponenten-ID	Kriterium
P-01	Der NIBO muss die Motorik ansteuern können.	OD	Muss
P-02	Der NIBO muss vorwärtsfahren können.	OD	Muss
P-03	Der NIBO muss nach links fahren können.	OD	Muss
P-04	Der NIBO muss nach rechts fahren können.	OD	Muss
P-05	Der NIBO muss rückwärts fahren können.	OD	Muss
P-06	Der NIBO muss mit dem Co-Prozessor kommunizieren können.	-	Muss
P-07	Der NIBO muss auf die Distanzsensoren zugreifen können.	DS	Muss
P-08	Der NIBO muss frontale Hindernisse erkennen können.	DS	Muss
P-09	Der NIBO muss seitliche Hindernisse erkennen können.	DS	Muss
P-10	Der NIBO muss Sackgassen erkennen können.	DS	Muss
P-11	Der NIBO muss erkannten Hindernissen ausweichen können.	DS	Muss
P-	Der NIBO muss auf das grafische Display zugreifen können.	DM	Muss

12			
P-13	Der NIBO muss erkannte Hindernisse auf dem Display ausgeben können.	DM	Muss
P-14	Der NIBO muss die LEDs ansteuern können.	SL	Muss
P-15	Der NIBO muss mithilfe der LEDs seine Aktionen visualisieren können.	SL	Muss

7.2 Anforderungsliste Kürteil

Anforderungen IndiaNIBOJones

Anmerkung: Um die Verständlichkeit der Anforderungen zu verbessern, werden im nachfolgenden Abschnitt die Anforderungen an das Programm IndiaNIBOJones auf den NIBO projiziert.

ID	**Beschreibung**	**Komponenten-ID**	**Kriterium**
IJ-01	Der NIBO muss vorwärtsfahren können.	OD	Muss
IJ-02	Der NIBO muss im 90° Winkel nach links fahren können.	OD	Muss
IJ-03	Der NIBO muss im 90° Winkel nach rechts fahren können.	OD	Muss
IJ-04	Der NIBO muss in der Lage sein eine schwarze Markierung auf dem Boden beim Darüberfahren erkennen zu können.	BS	Muss
IJ-05	Anhand der Sensordaten muss die Umrandung erkannt werden können.	DS	Muss
IJ-06	Der NIBO muss beim Programmstart selbstständig messen können, auf welcher Seite sich die Wand befindet und sich dementsprechend kalibrieren.	DS	Muss
IJ-07	Der NIBO muss unter der Verwendung der äußeren Distanzsensoren der Umrandung folgen können.	DS	Muss
IJ-08	Sobald der vordere Distanzsensor eine Wand erkennt, muss der NIBO selbstständig nach innen im 90° Winkel abbiegen können. (Innen meint hin zum Zentrum der Struktur)	DS	Muss

IJ-09	Sobald der äußere Distanzsensor keine Umrandung mehr erkennt, muss der NIBO nach außen im 90° Winkel abbiegen können. (Außen meint weg vom Zentrum der Struktur)	DS	Muss
IJ-10	Sobald die seitlichen Distanzsensoren eine zu hohe oder zu niedrige Distanz messen, muss der NIBO selbstständig den Kurs leicht korrigieren können.	DS	Muss
IJ-11	Der NIBO muss mithilfe der Odometrie die Länge der gefahrenen Strecken berechnen können.	OD	Muss
IJ-12	Der NIBO muss Daten zu seinen Richtungswechseln sammeln können.	OD	Muss
IJ-13	Der NIBO muss eine Verbindung mit dem NIBOBrody aufbauen können.	XM	Muss
IJ-14	Der NIBO muss regelmäßig, nach einem Richtungswechsel, die Daten an den NIBOBrody schicken können.	XM	Muss
IJ-15	Der NIBO muss Daten über XBee empfangen können.	XM	Muss
IJ-16	Der NIBO muss die empfangenen Daten auf dem Display ausgeben können.	DM	Muss
IJ-17	Der NIBO kann eine Richtungsänderung mithilfe der entsprechenden Status LED links oder rechts anzeigen.	SL	Kann
IJ-18	Der NIBO kann das Bremsen optisch über rote Rücklichter wiedergeben.	TM	Kann
IJ-19	Der NIBO kann Feedback zu seinen Bewegungen oder zur Beendigung der Aufgabe zurückliefern.	DM	Kann

Anforderungen NIBOShorty

NIBOShorty ist eine weitere Komponente des Systems, die einen zusätzlichen Nibo erfordert. Alle folgenden Anforderungen beziehen sich auf diesen Nibo.

ID	**Beschreibung**	**Komponenten-ID**	**Kriterium**
NS-01	Der NIBO kann Daten über XBee empfangen.	XM	Kann
NS-02	Der NIBO kann Daten über XBee senden.	XM	Kann

NS-03	Der NIBO kann die Indiana Jones Titelmusik abspielen.	TM	Kann
NS-04	Der NIBO kann eine Indiana Jones Figur und eine Peitsche auf dem Display anzeigen.	DM	Kann

Anforderungen NIBOBrody

ID	**Beschreibung**	**Komponenten-ID**	**Kriterium**
NB-01	NIBOBrody muss in der Lage sein, das XBee-Modul über USB ansteuern zu können.	XM	Muss
NB-02	NIBOBrody muss Daten über das XBee-Modul empfangen können.	XM	Muss
NB-03	NIBOBrody muss aus den Daten ein Modell der Struktur entwickeln können.	-	Muss
NB-04	NIBOBrody muss das Modell nach jedem Streckenabschnitt aktualisieren und darstellen können.	-	Muss
NB-05	NIBOBrody muss das Modell für den NIBO aufbereiten können.	-	Muss
NB-06	NIBOBrody muss die Daten zurück an den NIBO senden können.	XM	Muss