MB FAKULTÄT FÜR MASCHINENBAU

Otto-von-Guericke-Universität Magdeburg

Fakultät für Maschinenbau

Institut für Mobile Systeme

Bachelorarbeit

von

Erik Sommer

Erarbeitung und Implementierung eines globalen Orientierungsansatzes für eine autonome Roboterplattform in einer zweidimensionalen Umgebung

Betreuer:

Hon.-Prof. Dr. sc. techn. Ulrich Schmucker

10. April 2012

Vorwort

Diese Bachelorarbeit entstand im Rahmen der Projektgruppe robOTTO an der Otto-von-Guericke-Universität Magdeburg. Ich möchte allen danken, die zu ihrem Gelingen beigetragen haben.
Besonderer Dank gilt Hon.-Prof. Dr. sc. techn. Ulrich Schmucker für die Betreuung der Arbeit und seiner Begleitung durch Hinweise und Anregungen.
Weiterhin gilt mein Dank Alexander Ratai und Stefan Wilske für ihre Unterstützung und Ratschläge.
Prof. Dr.-Ing. Roland Kasper danke ich für die Erstellung des Zweitgutachtens.

Aufgabenstellung

Thema:
„Erarbeitung und Implementierung eines globalen Orientierungsansatzes für eine autonome Roboterplattform in einer zweidimensionalen Umgebung“

Eines der wichtigsten Forschungsfelder in der Robotik ist die Orientierung in bekannten, wie auch in unbekannten Umgebungen. Dieser Vorgang, auch Lokalisierung genannt, ist durch verschiedene Sensordaten realisierbar. Je nach Art der verwendeten Sensoren ist es einem Roboter möglich, sich in bekannten oder unbekannten Gebieten zu orientieren und Erkenntnisse über seine Umgebung zu gewinnen. Ein Spezialfall ist hierbei das Kidnapped-Robot-Problem. Hier ist dem Roboter zwar seine Umgebung bekannt, nicht aber seine momentane Position.

Auf Grundlage zweidimensionaler Messungen eines Laserscanners soll ein autonomer Roboter in die Lage versetzt werden, sich in einer bekannten Umgebung zu orientieren. Hierbei ist davon auszugehen, dass keinerlei Vermutungen oder Schätzungen über die Position vorliegen. Der theoretische Ansatz soll im Rahmen der RoboCup-Logistikliga implementiert werden.

Konkret soll die Bachelorarbeit folgende Punkte umfassen:

- Recherche: Lokalisierung mobiler Roboter
- Überblick über die verwendete Sensorik
- Analyse der Anforderungen und Limitationen, die durch Hardware, Umgebung und die Rahmenbedingungen der Liga bestehen
- Entwurf des Orientierungsansatzes
- Implementierung auf der Robotino®-Plattform
- Verifikation der Ergebnisse anhand von Testreihen

Kurzfassung

Eine der wichtigsten Aufgaben der Robotik besteht in der Lokalisierung eines mobilen Roboters. Hierbei kann es sich nur um die Korrektur von vorhanden Positionsdaten handeln. Im Gegensatz dazu ist beim Kidnapped-Robot-Problem keine Vermutung über die Pose des Roboters bekannt. Dabei muss der Roboter dann aus den Sensorinformationen seiner Umwelt und einem Abgleich mit einer Landkarte seine Pose bestimmen.

Die vorliegende Arbeit behandelt den Entwurf und die Implementierung eines globalen Lokalisierungsansatzes für eine mobile Roboterplattform. Dabei werden mobile Landmarken zur Detektierung durch einen 2D-Laserscanner entworfen. Weiterhin wird ein Algorithmus zur Filterung und Analyse der Scandaten vorgestellt.

Abstract

One of the most important tasks in robotics is to localize a mobile robot. This may solely be the act of correcting available position data. In contrast, the kidnapped-robot-problem involves no assumption about the position of the robot. Therefore the robot then needs to select the sensor information from its environment and compare it with a map to determine its position.

The bachelor thesis at hand deals with the design and implementation of a global localization approach for a mobile robot platform. For that purpose mobile landmarks for detection by a 2D laser scanner will be designed. Furthermore, an algorithm for filtering and analysis of the scandata is presented.

Erklärung der Selbstständigkeit

Hiermit versichere ich, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie die Zitate deutlich kenntlich gemacht zu haben.

Magdeburg, den 10. April 2012

Erik Sommer

Inhaltsverzeichnis

Abbildungsverzeichnis

Symbolverzeichnis

Symbol	Bedeutung	Einheit
c	Lichtgeschwindigkeit	$\mathrm{m/s}$
d_{Marke}	Durchmesser einer Landmarke	m
l_{Feld}	Länge des Spielfeldes	m
NR	Nummer eines Scanpunktes	
r	Betrag einer Polarkoordinate	m
s	Weg	m
t	Zeit	s
x	Längenkoordinate	m
x_1, y_1	initial vermutete Positionskoordinaten des Roboters	m
$x_{2,3,4}$, $y_{2,3,4}$	alternativ mögliche Positionskoordinaten des Roboters	m
x_R, y_R	Koordinaten des lokale Roboter-Koordinatensystems	m
x_{sch}, y_{sch}	gedrehte Koordinaten des Schnittpunktes	m
y	Breitenkoordinate	m
z	Höhenkoordinate	m
α	Winkel einer Polarkoordinaten	$^\circ$
β	Drehwinkel zwischen Roboter- und globalem Koordinatensystem	$^\circ$
λ	Wellenlänge	nm
ϕ	Posenwinkel	$^\circ$
σ	Standardabweichung	

1. Einleitung

1.1. Motivation und Vorarbeiten

Roboter sind heutzutage aus dem Alltag der Menschen nicht mehr wegzudenken. Es erweitern längst mobile Robotersysteme das Bild der starr fixierten Industrieroboter. Sie übernehmen dabei vielfältigste Aufgaben. Nicht zuletzt können sie den Menschen dabei unterstützen in Gebiete vorzudringen, die für ihn selbst zu gefährlich sind. So wurde bei dem Atomkraftwerksunfall von Fukushima zum ersten Mal erfolgreich eine Katastrophe von Robotern erkundet. [Vol, Plu11]

Für solche mobilen Roboter ist es natürlich von entscheidender Bedeutung, sich in ihrer Umgebung orientieren zu können. Dabei muss man die Komplexität der Umgebung beachten und den Bekanntheitsgrad der Umgebung für den Roboter.

Seit dem Jahr 2009 besteht an der Otto-von-Guericke-Universität Magdeburg das robOTTO-Projekt. In dieser Initiative haben sich Studenten aus verschiedenen Fachbereichen zusammengeschlossen, um gemeinsam an der neu gegründeten Festo-Logistics-League, im Rahmen des RoboCup, teilzunehmen. Ziel dieser Arbeit ist es, die Orientierung des Roboters in dieser Liga zu verbessern.

1.2. Gliederung der Arbeit

Die vorliegende Bachelorarbeit unterteilt sich in fünf Abschnitte. In Kapitel 2 werden die *Grundlagen* dieser Arbeit dargestellt. Es wird die verwendete Sensorik erläutert, des Weiteren wird ein Überblick über verschiedene Lokalisierungsmethoden gegeben. Kapitel 3 legt die *Voraussetzungen* dar, stellt den vorhandenen Roboter vor und gibt einen Einblick in die Logistics-League. Auch stellt sie bereits vorhandene Lokalisierungsmethoden vor, die vom robOTTO-Team aktuell genutzt werden. In *Entwicklung des Algorithmus*, dem 4 Kapitel, wird dann die Entwicklung des Lokalisierungsalgorithmus vorgestellt. Die bei der Evaluierung erzielten *Ergebnisse* werden in Kapitel 5 vorgestellt. Beschlossen wird die Arbeit durch Kapitel 6. Welches eine *Zusammenfassung* und einen *Ausblick* auf zukünftige Lokalisierungsverfahren gewährt.

2. Grundlagen

2.1. Sensorik

Der Mensch nimmt seine Umwelt über Sinnesorgane wahr. Die Aufgabe, sich im Raum zu lokalisieren, spielt sich bei uns hauptsächlich über die Augen ab. Dabei sind die Informationen, die wir über die Augen erhalten, qualitativ nicht hochwertig. So ist das Sichtfeld nicht nur eingeschränkt, sondern auch mit einem blinden Fleck versehen. Dass wir daraus trotzdem hochwertige Informationen über unsere Umwelt erhalten können, ist dem Gehirn zu verdanken, welches die Informationen verarbeitet. Ein solches ausgereiftes neuronales Netz hat der Roboter nicht zur Verfügung. Daher ist er nur in der Lage, auf sehr eingeschränkte Aufgabenstellungen Antworten zu haben und die bewältigt er auch nur durch eine Vielzahl zusätzlich Sensoren, die im Folgenden vorgestellt werden.

2.1.1. 2D-Laserscanner

Der Sensor, dem in dieser Arbeit das größte Interesse gilt, ist der 2D-Laserscanner. Dieser Sensor ermöglicht es, die Umgebung in einer Ebene abzuscannen und dadurch einen horizontalen Schnitt der umgebenden Objekte zu liefern. Dabei werden verschiedene Verfahren eingesetzt, um den Weg zu bestimmen, den ein Laserstrahl zurücklegt. Abbildung 2.1 zeigt einen Laserscan einer quadratischen Umgebung.

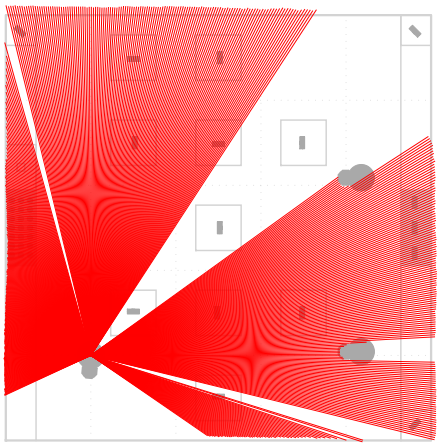


Abbildung 2.1.: Laserscan einer quadratischen Umgebung

Aufbau

Ein Laserscanner stellt immer den Mittelpunkt seines Messbereiches dar. Davon ausgehend muss er den gesamten Messbereich abtasten, um in jedem Winkel Tiefeninformationen zu erhalten. Daher muss das jeweilige Messverfahren mit einer Einrichtung gekoppelt werden, die den Laserstrahl in verschiedene Richtungen ablenken kann. Dies kann durch einen rotierenden Spiegel, der von einem Motor angetrieben wird, realisiert werden. Der Motor muss dabei natürlich sehr fein steuerbar sein, denn ohne eine exakte Winkelinformation zu den Tiefeninformationen lässt sich keine Aussage über die tatsächliche Koordinate treffen. Es bietet sich hier ein Servo- oder Schrittmotor an. Abbildung 2.2 zeigt den prinzipiellen Aufbau eines 2D-Laserscanners.

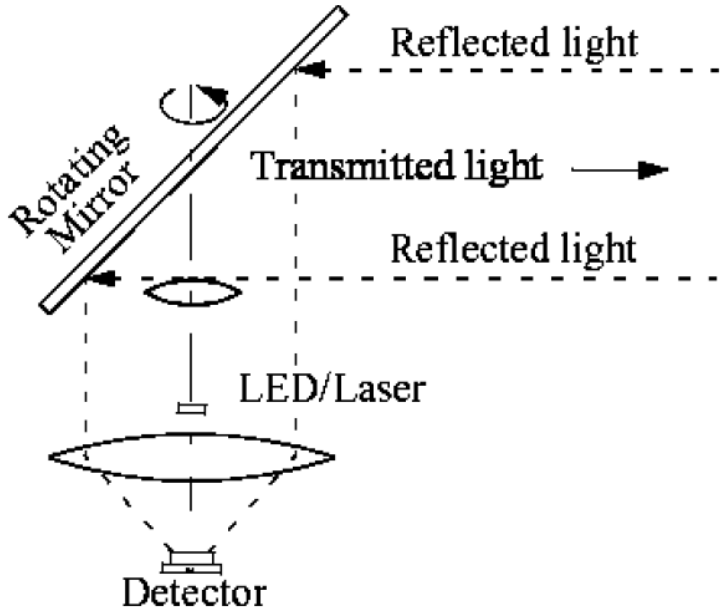


Abbildung 2.2.: Prinzipieller Aufbau eines Laserscanners [Sch09]

Messverfahren

Die Information, welche benötigt wird, ist die Strecke, die zwischen dem Laserscanner und dem reflektierenden Punkt liegt. Diese Information ist verschiedenartig messbar. Die Zeit ist messbar, die der Strahl benötigt um zurückzukehren. Es ist auch eine Messung der Phasenverschiebung möglich. Beide Verfahren sollen im Folgenden kurz erläutert werden.

Messung der Laufzeit

Bei diesem Verfahren wird der Laserstrahl nicht kontinuierlich emittiert, sondern gepulst, um einen definierten Beginn des Laserstrahls zu erzeugen. Bei Aussendung eines neuen Pulses wird dabei ein Startsignal an eine Zähleinrichtung gesendet. Wird durch einen Detektor der reflektierte Laserstrahl detektiert, so folgt dann ein Stoppsignal an den Zähler übertragen. Daraus ergibt sich die Zeitdifferenz, die mittels der Formel 2.1 in eine Wegstrecke umgerechnet

werden kann.

$$s = c \cdot t \tag{2.1}$$

Das Problem bei diesem Verfahren liegt in der enorm hohe Geschwindigkeit des Laserstrahls ($c \approx 3 \cdot 10^8 \frac{\mathrm{m}}{\mathrm{s}}$). Hieraus ergibt sich, dass zur Messung von $0,15\ \mathrm{m}$ eine Messung mit einer Genauigkeit von $1\ \mathrm{ns}$ möglich sein muss. Es eignet sich also nur für große Entfernungen. [Sch09]

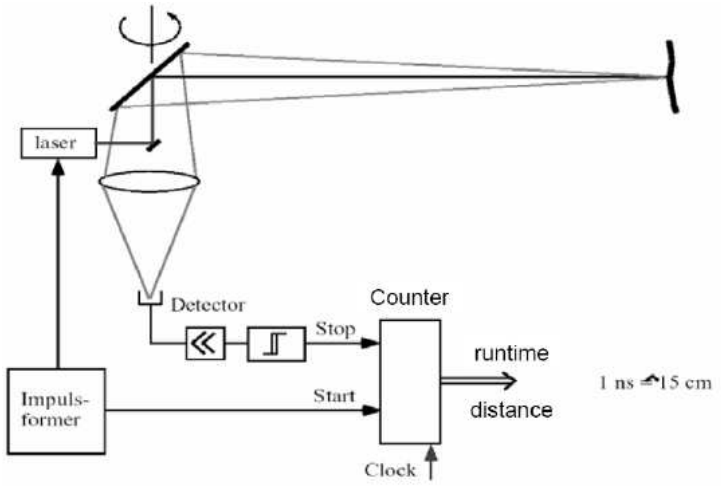


Abbildung 2.3.: Schema eines Laserscanners nach dem Laufzeitprinzip [Sch09]

Messung der Phasenverschiebung

Das zuvor beschriebene Verfahren eignet sich nicht zur Messung kurzer Entfernungen. Hierfür verwendet man Methoden zur Messung der Phasenverschiebung. Dabei wird der entsendete Laserstrahl durch einen halb durchlässigen Spiegel aufgespalten und der reflektierte Laserstrahl direkt zum Detektor abgelenkt. Der passierende Strahl erreicht wiederum das zu vermessende Objekt (möglicherweise abgelenkt durch weitere Spiegel), wird reflektiert und zum Detektor zurückgesendet. Dort überlagern sich die beiden Strahlen und es kommt zu einer Phasenverschiebung. Jetzt kann der Phasendetektor die Phasenverschiebung ermitteln und daraus kann ein Rückschluss gezogen werden, welchen Weg der nicht-reflektierte Laserstrahl zurückgelegt hat. Dieses Verfahren ist noch nicht eindeutig, da alle $\lambda/2$-Schritte die gleiche Phasendifferenz gemessen wird. Um dieses Problem zu lösen wird der emittierte Laserstrahl zusätzlich moduliert, damit eine Eindeutigkeit im geforderten Messbereich erzielt wird.
Abbildung 2.4 zeigt schematisch den Aufbau eines Systems, welches auf der Messung der Phasenverschiebung basiert.

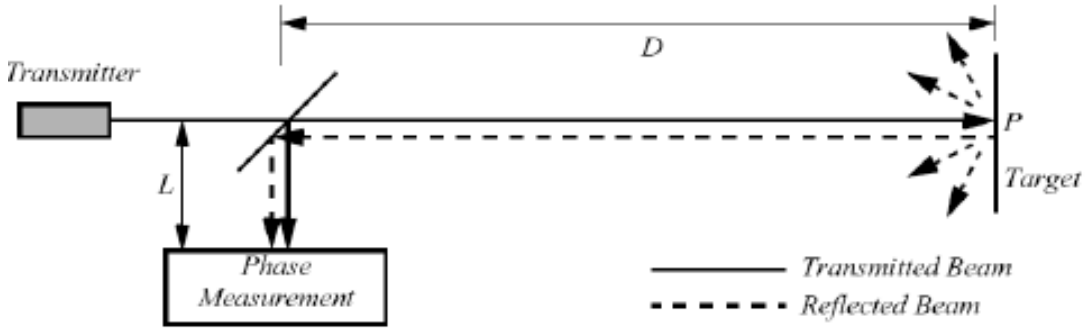


Abbildung 2.4.: Schema eines Laserscanners nach dem Prinzip der Phasenverschiebung [Sch09]

Mögliche Fehlerquellen

Bei einem Laserscanner handelt es sich um ein sehr komplexes Messsystem. Dies bietet dadurch leider auch eine Vielzahl von Fehlermöglichkeiten. Daher müssen alle Komponenten (Motor, Optik, Zeit- oder Phasensensorik und Laser) sehr genau gearbeitet sein, um brauchbare Ergebnisse zu erzielen. Dies führt dazu, dass ein solches System mit mindestens 1000€ sehr teuer ist. Zusätzlich zu den systematischen und zufälligen Fehlern der einzelnen Komponenten, kann es auch zu groben Fehlern kommen. Hier wären vor allem optische Effekte zu nennen, die durch Reflexion oder Brechung entstehen können. Dadurch kann es zu vereinzelten Messfehlern kommen, die entsprechend aus den Scan-Rohdaten heraus gefiltert werden müssen. Abbildung 2.5 zeigt z.B. die Totalreflexion einer Reihe von Messpunkten an einer glatten Wand, wodurch nicht genug Licht zurückgeworfen wird um eine Phasenverschiebung zu messen.

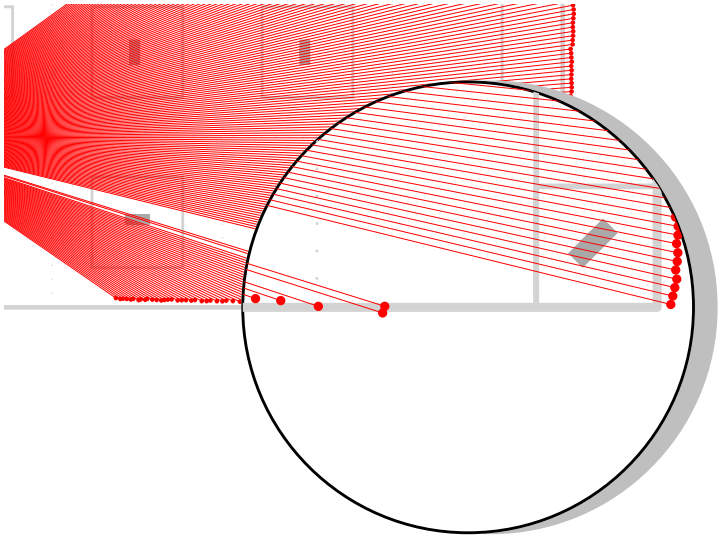


Abbildung 2.5.: Fehlende Messpunkte in einem Laserscan

Hokyo URG-04LX-UG01

Als Laserscanner in dieser Arbeit findet der URG-04LX-UG01 Verwendung. Es handelt sich um ein Basismodell. Er ist in der Lage einen Bereich von $240°$ in $0,1\ \mathrm{s}$ abzuscannen. Dabei erreicht er eine Winkelauflösung von $\approx 0,3°$ und kann Entfernungen von $20\ \mathrm{mm}$ bis $4000\ \mathrm{mm}$ laut Spezifikation detektieren. Weitere Daten finden sich in [HOK09a, HOK09b]

2.1.2. Inkrementalgeber

Einen sehr wichtigen Messwert für die Lokalisierung eines Roboters stellen die Umdrehungen seiner Räder dar. Grundsätzlich unterscheidet man dabei Verfahren, welche die relative Bewegung der Räder messen, also ihre Geschwindigkeit, und Methoden, die die absolute Position der Radachse (im Bereich $0°$-$360°$) erfassen. Zur Messung dieses Wertes stehen verschiedenste Messprinzipien zur Verfügung. Das populärste Verfahren, auch weil es preiswert ist, sind die Codescheiben. Hierbei werden an der Radachse Marken angebracht, die eindeutig auszuwerten sind. Es ist nicht immer nötig künstliche Marken anzubringen, es kann z.B. ein bestehendes Zahnrad genutzt werden, welches durch einen Induktivsensor ausgewertet werden kann. Um aber hohe Auflösungen zu erreichen, ist es vorteilhafter, sich künstlicher Codescheiben zu bedienen, deren Auswertung erfolgt optisch. Dazu wird paralleles Licht auf eine transparente Codescheibe geworfen. Bescheint sie einen durchsichtigen Bereich, kann der dahinter liegende Photosensor dies wahrnehmen und so kann ein binäres Signal erzeugen (hell/dunkel). Abbildung 2.6 zeigt den prinzipiellen Aufbau eines Inkrementalgebers.

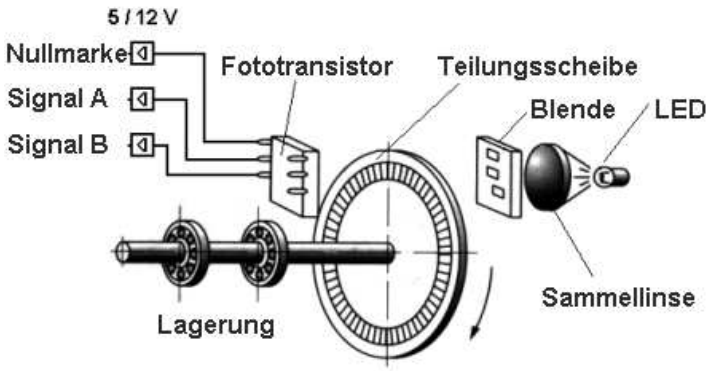
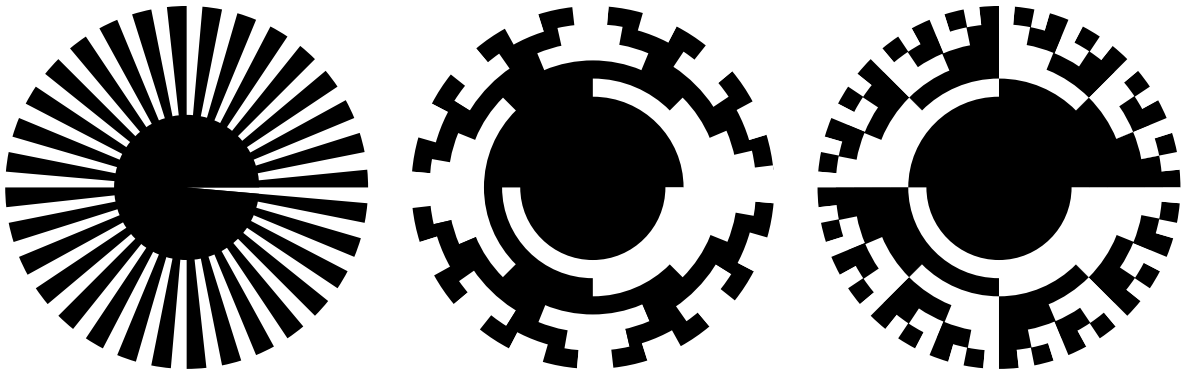


Abbildung 2.6.: Aufbau eines Inkrementalgeber [Käs07]

Bei einem einfachen Aufbau mit nur einem Sensor kann so die Relativbewegung der Radachse mithilfe einer Codescheibe ermittelt werden, wie in Bild 2.7a. Kann man mehrere Sensoren verwenden und so mehrere Spuren auf der Codescheibe auswerten, so kann man absolute Codescheiben nutzen. In Bild 2.7b ist eine Codescheibe mit dem sogenannten Gray-Code dargestellt. Der Aufbau einer solchen Scheibe mag zwar zunächst unpraktisch erscheinen, da die Codierung nicht binär hochzählt und somit ein weiterer Wandler eingesetzt werden muss, um aus dem Signal die Position zu ermitteln. Es ergibt sich aber ein Vorteil dadurch, dass immer nur eine Bahn ihr Binärsignal ändert. Bei Codescheiben wie in Bild 2.7c kann es vorkommen,

dass sich mehrere Signale an einer Position ändern. Wodurch im Grenzbereich Fehler entstehen können, da Binärsignale erkannt werden, die an dieser Position gar nicht vorhanden sind.



(a) für die Messung der relativen Bewegung

(b) Codescheibe mit Graycode

(c) Codescheibe mit Binärsignal

Abbildung 2.7.: Codescheiben für Inkrementalgeber

2.1.3. Gyrosensoren

Der Robotino® ist um einen Gyrosensor erweiterbar. (Abbildung 2.8) Dieser verbessert die Odometrie (Abschnitt 2.2.1) signifikant, da die Odometrie vor allem in der Ermittlung des Drehwinkels starken Fehlern unterworfen ist. Bei Anschluss des Gyrosensors wird also die Messung des Drehwinkels durch diesen ergänzt.

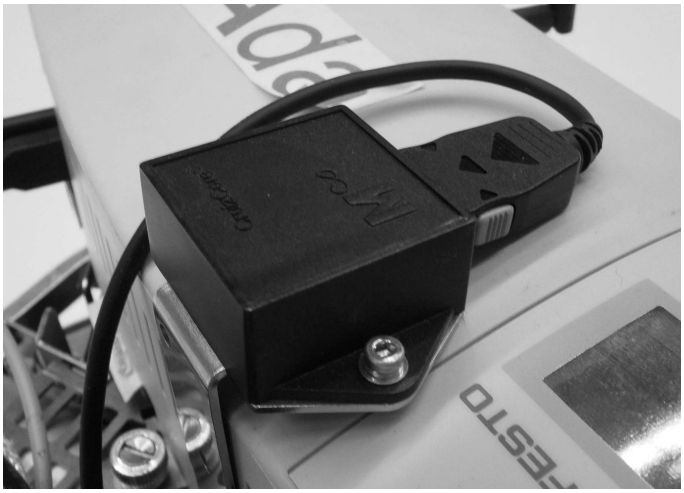


Abbildung 2.8.: Gyroskop am Robotino®

Klassische Gyrosensoren bestehen aus einem frei aufgehängten Kreisel. Mit Hilfe des den Drehimpulserhaltungssatzes kann dann eine Drehung relativ zur Anfangsorientierung gemessen werden. (Abbildung 2.9)

Eine sehr viel präzisere Messmethode stellt der Faserkreisel dar. Unter Nutzung des Sagnac-Effekts wird hier der Strahl eines Lasers aufgeteilt und in jeweils eine Richtung einer Glasfa-

serwindung eingespeist. Durch die Ermittlung der Phasendifferenz des geteilten Strahls kann eine Aussagen über die Winkelgeschwindigkeit getroffen werden.

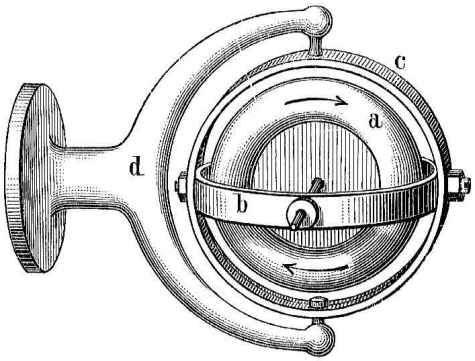
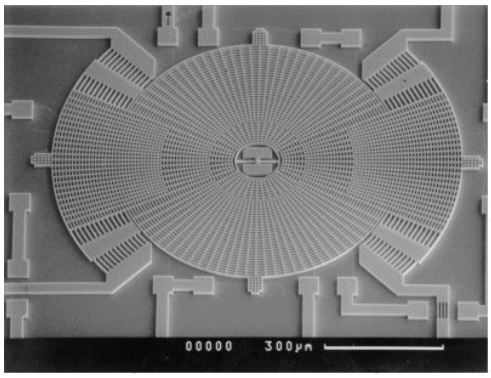
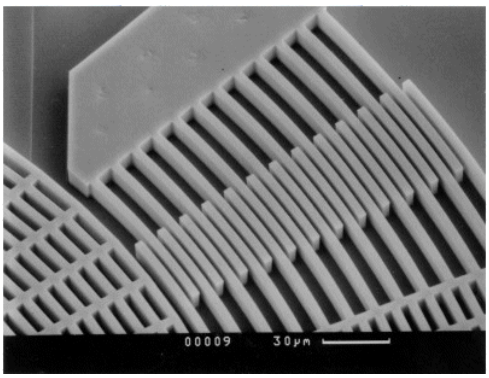


Abbildung 2.9.: Ansicht eines Kreiselsensor [Zen12]

Beide bisher vorgestellten Methoden bringen das Problem eines großen Bauraumes mit sich und sind teure Bauweisen. Daher werden für mobile Plattformen einfacher aufgebaute Gyroskope verwendet, deren Funktionsprinzip schlicht gehalten ist. Es wird ein elastisch verformbarer Körper frei gelagert, dessen rotatorische Verformung dann entweder piezoelektrisch, piezoresistiv oder kapazitiv messbar ist. Solche Körper können durch plastische Mikrotechniken in sehr kleinen Maßstäben gefertigt werden. Abbildung 2.10a zeigt einen solchen Sensor. Wenn sich eine Winkelbeschleunigung auf den zentralen Körper auswirkt, dreht er sich um sein Zentrum, wodurch sich die Stellung der Lamellen ändert (Detail: Ausschnitt 2.10b). Dies bewirkt eine Änderung der Kapazität der Lamellen, deren Wert für die Winkelbeschleunigung über eine Messbrücke abgeleitet werden.



(a) Gesamtansicht



(b) Detail

Abbildung 2.10.: Mikroskopische Ansicht eines Gyrosensors [GBH+88]

Alle diese Verfahren liefern nur Winkelbeschleunigungen. Ein Weg kann also nur durch zweifaches Integrieren erhalten werden, wodurch sich Messfehler aufsummieren. Ein Einsatz als alleiniges Verfahren ist also nicht optimal. Es wird deshalb eine Sensordatenfusion der Systeme Odometrie und des Gyroskops angestrebt. Das heißt in diesem Fall, dass die Messergebnisse verschiedener Sensoren zu einem Messwert verrechnet werden.

2.2. Lokalisierungsmethoden

Als Lokalisierung eines Roboters bezeichnet man die Ermittlung seiner Pose bezüglich seiner Umgebung. Eine Pose umfasst neben den Raumkoordinaten auch Informationen über die Neigungswinkel. Dabei ist zunächst davon auszugehen, dass jeder Körper, also auch ein Roboter, sechs Freiheitsgrade in einer dreidimensionalen Umgebung hat. Eine komplette Pose eines Roboters besteht also aus den drei Raumrichtungen x,y,z und drei möglichen Drehrichtungen. Nun sind ein Großteil der mobilen Roboter erdgebunden und bewegen sich zumeist auf ebenem Gelände. Daher sind die meisten nicht in der Lage, sich in die Höhe zu bewegen oder sich um ihre x- bzw. y-Achse zu drehen. Dies vereinfacht die Pose zu $\begin{pmatrix} x \\ y \\ \phi \end{pmatrix}$. Um sich zu orientieren, zieht der Roboter Sensordaten heran, die er aus seinem lokalen Koordinatensystem heraus wahrnimmt.
Bild 2.11 zeigt den Roboter in einem ihn umgebenden absoluten Koordinatensystem. x_R und y_R sind dabei die Koordinatenachsen seines lokalen Koordinatensystems.

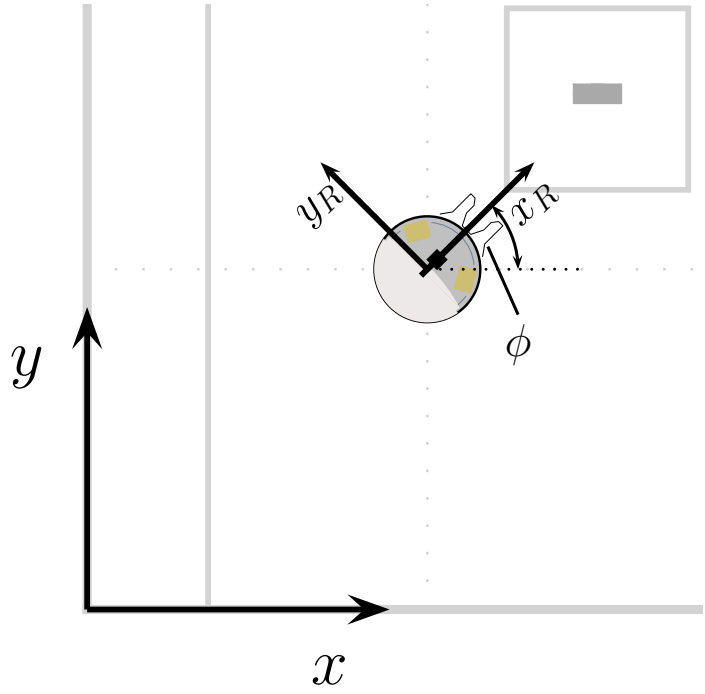


Abbildung 2.11.: Roboter im globalen Koordinatensystem

Die Verarbeitung dieser Daten hängt nun von der Aufgabe ab. Einerseits braucht der Roboter Information über seine Eigenbewegung (lokale Lokalisierung, Abschnitt 2.2.1). Andererseits ist es auch nötig festzustellen, wo er sich in einer gegebenen Karte befindet (globale Lokalisierung, Abschnitt 2.2.2). Verfahren dieser verschiedenen Lokalisierungsarten sollen im Folgenden erläutert werden.

2.2.1. Lokale Lokalisierung

Als Systeme zur relativen Positionsbestimmung eignen sich die Odometrie oder die Nutzung von Trägheitsdaten aus Gyrosensoren (Abschnitt 2.1.3).

Odometrie

Die Odometrie ist eines der einfachsten Verfahren, um den Ort eines mobilen Systems zu bestimmen. Es handelt sich um ein inkrementelles Verfahren. Zwischen jedem Messschritt wird die Positionsveränderung gemessen und zur vorher bekannten Pose addiert. Für eine eindimensionale Bewegung ergibt sich damit Gleichung 2.2.

$$s_{neu} = s_{alt} + \Delta s \tag{2.2}$$

Je geringer der Abstand zwischen jeder Messung gewählt wird, desto genauere Ergebnisse sind erzielbar.

Die Bestimmung von Δs ist nun stark von der Geometrie des Roboters abhängig. Im Falle eines nicht-schienengebunden Roboters erweitert sich Δs zu einer komplexeren Pose der Form $\begin{pmatrix} \Delta x \\ \Delta y \\ \Delta \phi \end{pmatrix}$. Der Roboter kann sich also in der Ebene bewegen und sich dabei noch drehen. Beim Robotino® handelt es sich um einen Roboter mit Omniwheel-Antrieb. Dies ist ein Rad, welches wie ein normales Rad um seine Achse rollen kann, aber im Unterschied dazu längs seiner Achse geschoben werden kann. Abbildung 2.12 zeigt ein Omniwheel.

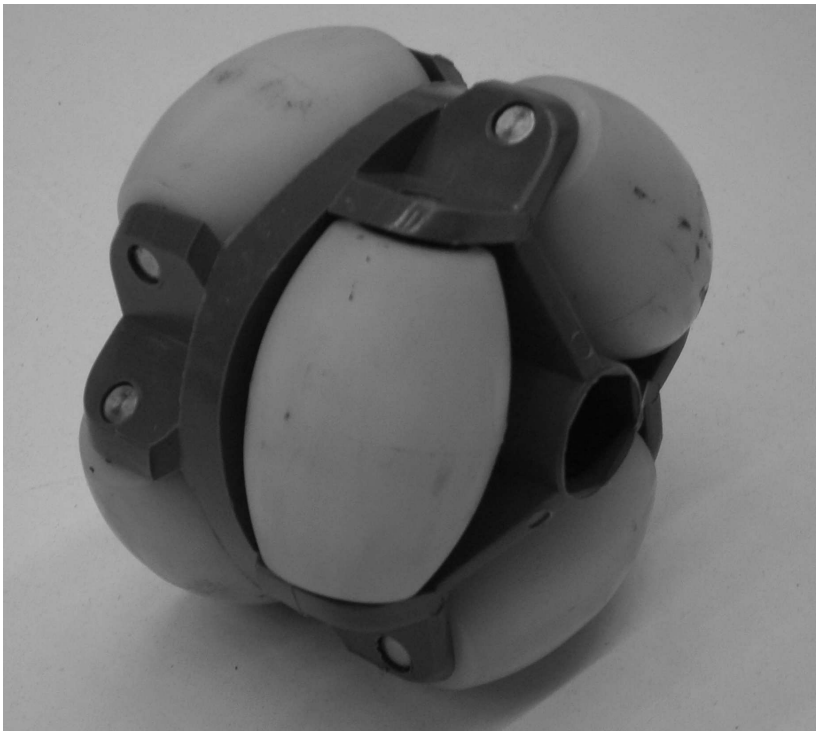


Abbildung 2.12.: Ansicht eines Omniwheel

Fehler der Odometrie

Omniwheels können eine Bewegung parallel zu ihrer Achse ausführen, welche aber sensorisch nicht erfasst werden kann. Dies stellt eine große Fehlerquelle für die Odometrie dar.

In [BEF96] findet sich weitere Fehlerquellen der Odometrie.

systematische Fehlerquellen:

- ungleiche Raddurchmesser
- durchschnittlicher Raddurchmesser weicht, vom im Algorithmus angegebenen, ab
- ungleiche Gewichtsverteilung
- ungenaue Startposition
- Versatz der Räder
- Auflösung des Drehgebers
- Abtastrate des Drehgebers

nichtsystematische Fehlerquellen:

- unebene Böden
- Objekte auf dem Boden
- Schlupf
  - glatter Untergrund
  - zu starke Beschleunigung
  - Schlittern durch abruptes Stoppen
  - Kräfte aus der Umgebung
  - interne Kräfte (z.B. durch Stützräder)
  - Rad liegt nichtpunktförmig auf

Diese umfangreiche Liste macht deutlich, dass viele Fehlerquellen die Odometrie stark beeinträchtigen können. Da es sich bei diesem Verfahren um ein integrierendes Verfahren handelt, summieren sich auch die Fehler auf. Daraus folgt, dass die Odometrie nur für kleine Entfernungen akzeptable Messwerte liefert. Für längere Beobachtungszeiträume und Entfernungen müssen entweder korrigierende Algorithmen angewandt werden oder andere Verfahren müssen anhand anderer Merkmale die odometrischen Daten korrigieren.

In [BEF96] finden sich Beispiele für korrigierende Algorithmen. Zum einen wäre hier ein Odometrieanhänger für den Roboter zu nennen, der durch seinen sehr steifen Aufbau, die Odometriedaten signifikant verbessert. Weiterhin kann man die systematischen Fehler wegtrainieren. Hierbei lässt man den Roboter ein exaktes Quadrat abfahren. Wenn er wieder an der Startposition angekommen ist, kann man aus den Abweichungen zur Sollposition Korrekturdaten für die Odometrie ableiten.

Alle Korrekturalgorithmen können aber nur bei systematischen Fehlern eine gewisse Abhilfe schaffen. Grobes Einwirken auf den Roboter, durch die Umwelt, kann hierbei nicht betrachtet werden. Deshalb ist es nötig, für lange Zeiträume die Odometriedaten durch andere Methoden zu korrigieren. Dabei soll es nicht um einen kompletten Ersatz für die Odometrie gehen, denn ein wichtiger Vorteil der Odometrie liegt in der hohen Abtastrate, mit der neue Positionsdaten geliefert werden. So ist es möglich, die Positionsdaten eines Roboters nahezu kontinuierlich nachzuführen, was durch andere Verfahren schwer zu leisten ist.

Trägheitsnavigationssysteme

Bei der Messung der Trägheit wird unter Zuhilfenahme von Gyrosensoren die Beschleunigung oder Winkelbeschleunigung des Roboters ermittelt. Durch zweimalige Integration kann daraus die Position des Roboters ermittelt werden. Dieses Verfahren ist ähnlich der Odometrie einem sich aufsummierenden Fehler unterworfen. Daher werden beide Verfahren zumeist nur unterstützend eingesetzt.

2.2.2. Globale Lokalisierung

Bei der globalen oder auch absoluten Lokalisierung besteht das Ziel darin, sich in einem globalen Koordinatensystem einzuordnen. Hierzu nutzt der Roboter seine Sensordaten, um einen Abgleich seiner momentanen Position mit gegebenen Informationen von seiner Umwelt vorzunehmen oder er entwickelt schrittweise seine eigene Karte der Umgebung. Dabei unterscheidet man zwischen kartenbasierten und markenbasierten Verfahren, die im Folgenden kurz erläutert werden.
[BEF96]

Kartenbasierte Verfahren

Bei diesem Verfahren nimmt der Roboter seine Umgebung möglichst vollständig wahr (soweit das aus seiner Perspektive möglich ist). Dafür kann er Laserscanner oder Kameras verwenden. Aus diesen Daten generiert er dann ein digitales Abbild der Umgebung. Nun versucht er diesen Ausschnitt seiner Umgebung mit der globalen Karte abzugleichen. Das Modell der Umgebung kann in verschiedenen Formaten vorliegen. Es können CAD-Daten, Punktwolken oder für eine ebene Umgebung auch nur ein Linienmodell sein.

Landmarken

Die Wahrnehmung einer gesamten Umgebung kann sehr rechenaufwendig sein. Um diesem Problem zu entgehen, wurden Verfahren entwickelt, bei denen nur exponierte Punkte der Umgebung gezielt wahrgenommen werden, sogenannte Landmarken. Man unterscheidet zwischen aktiven und passiven Landmarken.

Aktive Landmarken

Roboter erhalten von aktiven Landmarken Informationen Als bestes und alltägliches Beispiel ist hierbei GPS (Global Positioning System) zu nennen, dass weltweite Orientierung auf dem Globus sicherstellt. Die Basis des GPS- Systems sind Satelliten, die jeder ein sehr exaktes Zeitsignal auf einem $1,5\ \mathrm{GHz}$ Frequenzbereich sendet. Dieses Zeitsignal wird durch eine Atomuhr generiert. Nach Empfang dieses Signals kann der Empfänger dann sehr genau den Abstand zum Satelliten bestimmen. Durch Verbindungen zu vier Satelliten (drei für die Ermittlung des Schnittpunktes aus drei Großkugeln und ein weiterer zur Korrektur des Zeitsignals) kann der Empfänger dadurch weltweit seine Position bestimmen. Dieses Prinzip ist natürlich auch auf erdstationäre Aufbauten übertragbar und durch die geringere Entfernung der Landmarke sind höhere Genauigkeiten erreichbar. [Kop11]

Passive Landmarken

Passive Landmarken werden durch den Roboter erkannt. Dies können sowohl künstliche, als auch natürliche Landmarken sein. Die Sensorik misst dabei die Position bzw. den Abstand zum Roboter. Durch Triangulation kann die Position des Roboters bestimmt werden. Dazu ist es selbstverständlich notwendig, dass die Position der Landmarken bekannt ist.

Kidnapped-Robot-Problem

Das Kidnapped-Robot-Problem ist ein Spezialfall der Lokalisierung. Hierbei war dem Roboter in der Vergangenheit seine Position bekannt. Dann geht man davon aus, dass er durch Fremdeinwirkung auf eine beliebige andere Position der ihm bekannten Karte gesetzt wurde. Die Herausforderung besteht nun darin, seine exakte Position wiederzufinden. Eine analoge Aufgabenstellung ist dabei das erste Aktivieren eines Roboters in einer bekannten Umgebung.

Verfahren

Die meisten Verfahren zur Implementierung der marken- bzw. kartenbasierten Verfahren sind abgleichende Verfahren. Bei komplexen Problemstellungen, besonders bei großen oder sich selbst ähnelnden Umgebungen, können diese Verfahren durch probabilistische Ansätze ergänzt oder ersetzt werden. Diese Verfahren ermitteln keine exakten Positionen, sondern geben nur Wahrscheinlichkeitswerte für Positionen wieder. Eine detaillierte Darstellung solcher Verfahren findet sich z.B. bei [BEF96].

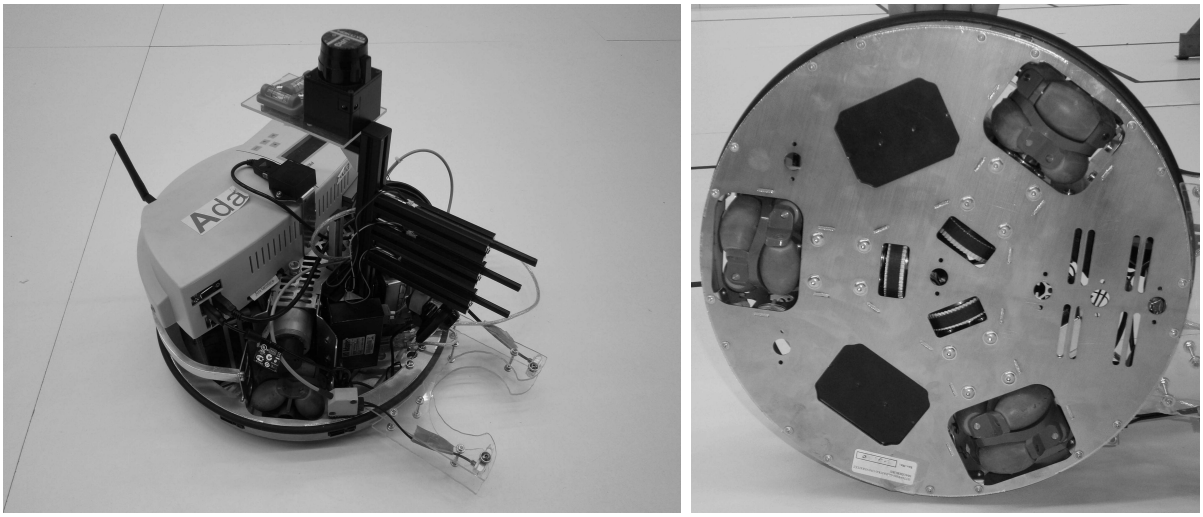
3. Voraussetzungen

3.1. Robotino®

Als Roboterplattform kommt das didaktische Robotersystem Robotino® zum Einsatz. Es wird von der Firma „Festo Didactic“ vertrieben und soll eine einfach zu handhabende Plattform für die Aus- und Weiterbildung sein. Um einen Überblick über den verwendeten Roboter zu bekommen wird er im Folgenden kurz vorgestellt. Weiterführende Informationen finden sich bei [rob12].

3.1.1. Hardware

Abbildung 3.1 zeigt zwei Ansichten des Robotino®. Das Gestell bildet ein rundes Edelstahlchassis. Zusammen mit dem in Bild 3.1b zu erkennenden omnidirektionalen Antrieb bildet sich so eine sehr gut navigierbare Plattform, da der Robotino® sich in jede beliebige Richtung bewegen oder drehen kann. Er kann Geschwindigkeiten von bis zu $10\,\mathrm{km/h}$ erreichen. Die Spannungversorgung erfolgt über zwei $12\,\mathrm{V}$ Blei-Gel-Akkumulatoren.



(a) Robotino® **(b)** Ansicht des Omniwheel-Antriebs

Abbildung 3.1.: Ansichten des Robotino®

Gesteuert wird der Robotino® über eine Kopplung von einer eigenen Steuerungskarte und einem PC104-System. Dabei übernimmt die Steuerungskarte die Auswertung der analogen und

digitalen Sensoren, sowie die Ansteuerung der Motoren. Das PC104-System wiederum verfügt über ein Ubuntu Linux als Betriebssystem und macht so die Anbindung von USB-Geräten möglich. Der eigentliche Programmcode wird hier ausgeführt. Leider verfügt der Robotino® derzeit nur über einen $500\ \mathrm{MHz}$ Prozessor, was rechenintensive Algorithmen stark einschränkt.

3.1.2. Sensorik

Der Robotino® verfügt über eine umfangreiche Grundausstattung von Sensoren:

- Inkrementalgeber, die die Motorumdrehungen messen
- 9 Infrarotsensoren, die eine Abstandsmessung rund um den Robotino® ermöglichen
- ein Kontaktsensor, der in einer Höhe von $20\ \mathrm{mm}$ rund um den Roboter angebracht ist
- eine Webcam, die eine Bildverarbeitung ermöglicht

Darüber hinaus können weitere Sensoren an den Robotino® angeschlossen werden. So bietet Festo beispielsweise auch ein eigenes Orientierungssystem an, welches durch die Erzeugung eines künstlichen Sternenhimmels den Roboter in einem $4\ \mathrm{m} \times 4\ \mathrm{m}$ Gelände lokalisieren kann. Neben weiteren, von Festo angebotenen, Sensoren können aber auch Fremdfabrikate genutzt, bzw. Eigenentwicklungen, werden. Die Anbindung solcher Sensoren erfolgt über eine universelle Schnittstelle, die analoge und digitale Anschlüsse bietet.

3.1.3. Programmierung

Festo hat für den Robotino® eine eigene Programmierumgebung entwickelt. Robotino®-View ist dabei LabView nicht unähnlich. Die einzelnen Komponenten werden durch Blöcke symbolisiert und mit Verbindungen verknüpft. Weiterhin können Skripte eingebunden werden. API's für die Sprachen C,C++,C#,.Net und Java stehen bereit. Auch sind Schnittstellen für LabView und Matlab vorgesehen.

3.2. Logistics League

Die Logistics League (LL) ist ein im Jahr 2010 eingeführter Wettbewerb (damals noch als Festo Logistics League) im Rahmen des RoboCup und löste eine davor existierende Hockey-Liga ab. Der Grundgedanke war, eine Liga zu schaffen, die das industrielle Umfeld widerspiegelt. Sponsor dieser Liga ist die Firma Festo. Daher finden auch deren Roboter Robotino® (Abschnitt 3.1), als Standardplattform Verwendung. Durch das Spielfeld soll eine Fabrik simuliert werden. Es werden aus Rohstoffen, über Zwischenstufen, Produkte hergestellt. Hierbei dienen Hockeypucks mit RFID-Chips als Modell der Güter. Die Roboter müssen hierbei den Materialfluss aufbauen und so den Produktionsablauf und die Auslieferung gewährleisten. Nur bei Defekten können die Teams minimal eingreifen. Eine nähere Erklärung des logistischen Ablaufs

findet sich in Abschnitt 3.2.2.[Pen12]

Seit der Gründung der Liga 2010 haben sich die Regeln nicht grundsätzlich geändert. Zunächst war das Konzept noch sehr am alten Hockey Wettbewerb orientiert, was sich durch direkte Konfrontation der beiden Teams auf dem Feld zeigte. Darunter litt aber der logistische Aspekt sehr. Im Jahr 2011 wurde dann das zweite Feld für das konkurrierende Team eingeführt und die Mannschaften konnten sich vollständig auf die Bewältigung der logistischen Produktionskette konzentrieren. Auch wurden nun für kleinere Teilaufgaben Punkte vergeben, was verhinderte, dass erneut viele Spiele 0:0 enden. Im Jahr 2012 wurden die Regeln minimal verändert und werden zukünftig um andere Elemente erweitert. So sollen Hindernisse eingeführt werden, die von den Robotern erkannt werden müssen. Auch ist eine Kommunikation zwischen Schiedsrichter und Robotern angedacht, um ihnen Anweisungen zu erteilen. Weiter in der Zukunft soll dann auch die Interaktionen mit den Maschinen verändert werden. Es sollen Aufgaben eingeführt werden, bei denen man mit dem Gegner zusammenarbeiten muss, wobei den Robotern eine komplexe Bestellung übermittelt wird. Es ist zu erwarten, das sich das Anforderungsspektrum für die Teams stetig erweitern wird. [Fes12] [Pen11]

3.2.1. Spielfeld

Die LL wird auf einem quadratischen Spielfeld der Größe $5,625\,\mathrm{m} \times 5,625\,\mathrm{m}$ ausgetragen. Auf einer Unterkonstruktion aus Aluminiumprofilen werden 5×5 Platten, welche aus hoch gepressten Laminatplatten bestehen, ausgelegt und von einer Bande, der Höhe $0,5\,\mathrm{m}$, umrandet. Die Grundfarbe des gesamten Feldes ist weiß. Dies dient der einfacheren Bildverarbeitung, da die Kamera nicht von unnötig vielen Störquellen beeinflusst wird. Das Feld ist in verschiedene Zonen eingeteilt. Grob werden Wareneingangsbereich, Maschinenbereich und Warenausgangsbereich unterschieden. Die Roboter starten neben dem Wareneingangsbereich. Im Wareneingang befinden sich zum Start jedes Spiels Hockeypucks, die für Rohstoffe stehen (näher ausgeführt in Abschnitt 3.2.1). Der Wareneingang ist blau gefärbt und kann von der Bildverarbeitung als Information genutzt werden, genau wie die grüne gegenüberliegende Seite. Dies ist der Warenausgang, der durch drei Maschinen (siehe Abschnitt 3.2.1) repräsentiert wird. Der Maschinenbereich wird von zehn Maschinen ausgefüllt, welche jeweils in der Mitte einer Platte angebracht sind. Eine Ansicht des Spielfeldes ist in Abbildung 3.2 zu sehen, eine vergrößerte Ansicht im Anhang B.

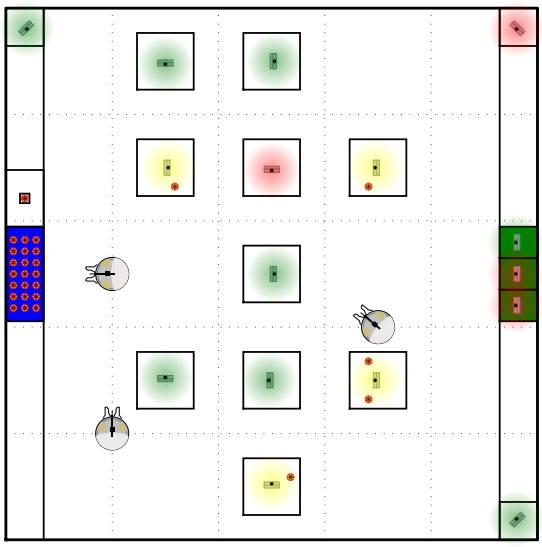


Abbildung 3.2.: Draufsicht des Spielfeldes

Pucks

Die Pucks repräsentieren Güter. Diese Güter können sein:

- Paletten
- Rohstoff
- 1. Zwischenprodukt
- 2. Zwischenprodukt
- Endprodukt

Auf die Bedeutung der einzelnen Puckarten wird in Abschnitt 3.2.2 eingegangen. Die Pucks werden von den Robotern mittels einer Schiebevorrichtung vor sich hergeschoben. Auf den Pucks befindet sich ein RFID-Chip, der durch die Maschinen beschrieben und gelesen werden kann. Die Pucks sind in einem einheitlichen Rotton gehalten, der durch die Kameras der Roboter gut erfasst werden kann. Abbildung 3.3 zeigt einen Spielpuck.

Abbildung 3.3.: Ansicht eines Spielpucks

Maschinen

Maschinen sind der Bestandteil des Feldes, mit denen der Roboter hauptsächlich interagiert. Sie werden repräsentiert durch einen Stahlwinkel, auf dem eine industrielle Ampel sitzt, wie sie auch in typischen Produktionsanlagen eingesetzt wird. Vor dem Stahlwinkel ist ein RFID-Schreib-/Lesegerät angebracht, dass den Puck auslesen kann, der von der Seite an die Ampel herangeführt wird. Es gibt vier Arten von Maschinen auf dem Feld:

- Produktionsmaschinen
- Auslieferungsmaschinen
- Lesemaschine
- Recyclingmaschinen

Produktionsmaschinen werden im Maschinenbereich jeweils mittig auf einer Platte angeordnet. Ihre Ausrichtung sind dabei zufällig vor jedem Wettkampf bestimmt. Auslieferungsmaschinen befinden sich im Warenausgangsbereich. In jeweils drei Ecken sind eine Lesemaschine und zwei Recyclingmaschinen untergebracht, die von den Robotern angefahren werden können. In Bild 3.4 ist das Maschinenmodell des robOTTO-Teams zu sehen, bei welchem der RFID-Scanner fehlt.

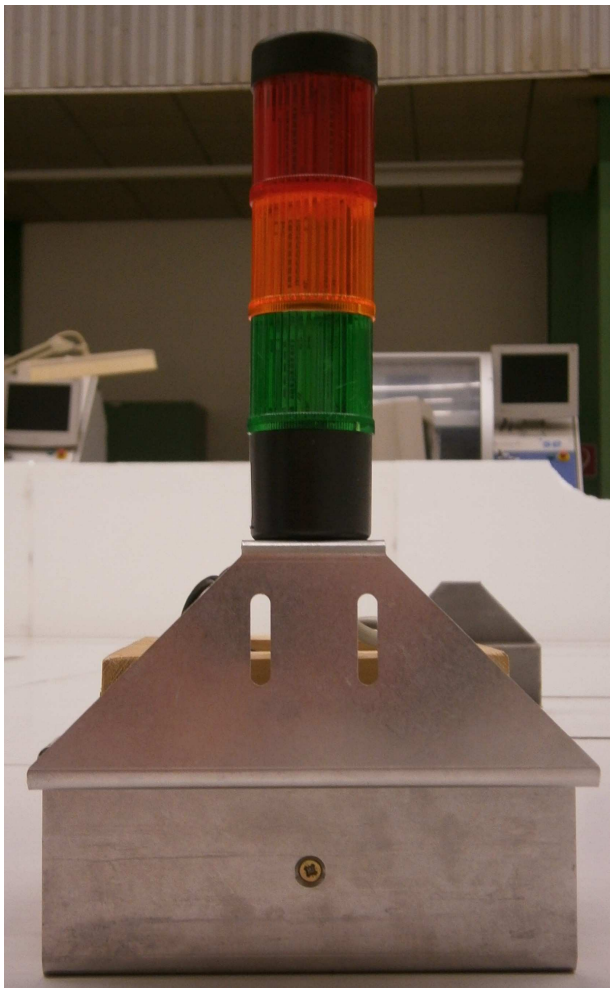


Abbildung 3.4.: Maschinenmodell des robOTTO-Teams

Abwandlungen im Testaufbau

Ein originalgetreuer Wettkampffeldnachbau war dem robOTTO-Team nicht möglich. Schon das Grundmaterial (Aluminiumprofile, Laminat) ist zu teuer, um es originalgetreu nachzubilden. Daher wurde das Feld im Rahmen der finanziellen Möglichkeiten nachgebaut. Der Untergrund des Feldes an der Otto-von-Guericke-Universität besteht aus wesentlich dickeren Spanplatten, die mit einer weißen Beschichtung überzogen sind. Das hat verschiedene Auswirkungen auf die Fahrqualität für die Roboter. Einerseits ist die Struktur der Oberfläche ungefähr identisch andererseits ist das nachgebaute Feld sehr viel steifer, da es auf einem festen Untergrund ruht. Daraus folgt, dass der Nachbau besser für odometrische Messungen ist und es auf dem Originalfeld zu weiteren Problem kommen kann, die im Nachbau des robOTTO Teams nicht erprobt werden können. Die Banden des Nachbaus bestehen aus Styropor, was die Reflektionscharakteristik erheblich verändert. Auch in diesem Aspekt ist der Nachbau eigentlich zu gut, da es durch die rauere Oberfläche zu weniger Reflektionen kommen kann, die für Messungen mit einem Laserscanner störend sind. Was in der nachgebauten Umgebung nicht bedacht werden konnte ist die komplette Maschinenanlage. So verfügt der Nachbau des Feldes zwar über Maschinenwinkel, denen sowohl die Ampel, zur Signalisierung eines Status als auch das RFID Lese-/Schreibgerät fehlt. Folglich müssen bei den Probeläufen diese Signale simuliert werden.

3.2.2. Logistischer Ablauf

Dieser Abschnitt soll einen kurzen Überblick über die logistische Aufgabenstellung geben, die hinter der LL steht. Eine exakte Beschreibung ist dem offiziellen Regelwerk [Fes12] zu entnehmen. Da die Liga einen Wettkampf zwischen zwei konkurrierenden Teams darstellt, wird der Erfolg an Punkten festgemacht, die für einzelne Teilaufgaben vergeben werden. Dabei gilt: Je weiter man im Produktionsprozess fortschreitet, desto mehr Punkte werden dafür vergeben. Der Produktionsprozess ist in Abbildung 3.5 dargestellt. Die einzelnen Blöcke haben dabei folgende Bewandtnis:

S0 Rohprodukt, wird an allen Produktionsmaschinen benötigt

S1 1. Zwischenprodukt

S2 2. Zwischenprodukt

P Produkt, wird an einem Warenausgang abgeliefert

LP leere Palette, kann an einer Recycling Einheit wieder befüllt werden

M1 Maschine Typ 1, produziert aus einem S0 ein S1

M2 Maschine Typ 2, produziert aus einem S0 und einem S1 ein S2 und eine LP

M3 Maschine Typ 3, produziert ein P und 2 LP aus einem S0, einem S1 und einem S2

MA Warenausgang, liefert ein fertiges P aus

MR Recyclingmaschine, macht alle angelieferten Pucks wieder zu einem S0

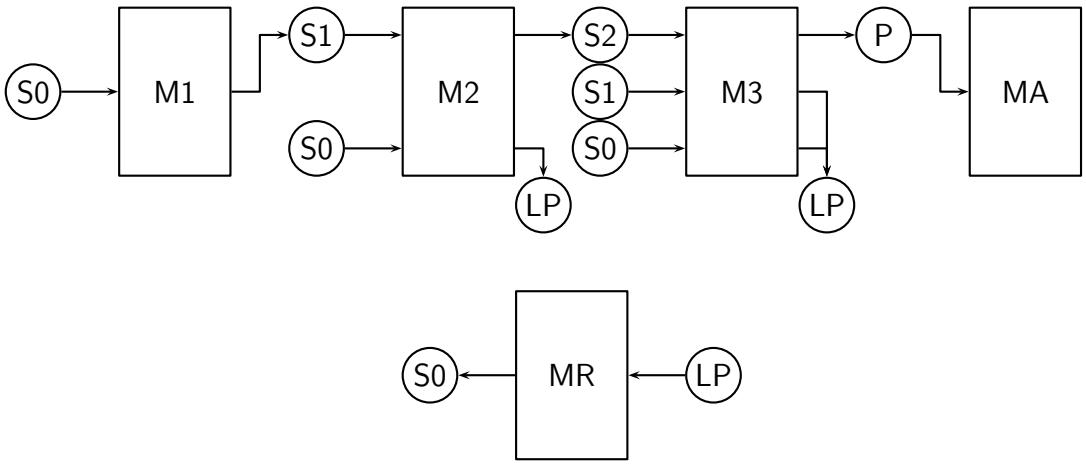


Abbildung 3.5.: Schema des logistischen Ablaufs

3.3. Vorhandene Methoden

Diese Arbeit baut auf vorhandene Algorithmen auf. Es existiert bereits ein lokaler Lokalisierungsalgorithmus, der auch im produktiven Softwarestand des robOTTO-Teams benutzt wird. Er soll kurz beschrieben werden. Weiterhin wird in der Entwicklung des Programms (Kapitel 4) ein Algorithmus zur Kantendetektion in Punktmengen nutzt, der hier kurz erläutert wird.

3.3.1. Bisheriger Lokalisierungsansatz des robOTTO-Teams

Schnell wurde klar, dass eine zuverlässige Navigation nicht nur durch Odometrie und Gyroskop gegeben sein würde. Die einzige Möglichkeit, diese Daten anhand der Umwelt zu korrigieren, bestand in einem Reset der Position beim Anfahren einer jeden Maschine (Abbildung 3.4). Wenn nun aber durch fehlerhafte Odometriedaten die gewünschte Position sehr weit verfehlt wurde, war auch dieser Korrekturalgorithmus wirkungslos und der Roboter navigationsunfähig. Daher wurde 2011 damit begonnen 2D-Laserscanner einzusetzen, um die Wände des Spielfeldes als Landmarken nutzen zu können. Ausführlich wird dieser Algorithmus in [Ler11] dargelegt, er soll hier nur grob umrissen werden. Die Notwendigkeit eines ergänzenden globalen Lokalisierungsansatzes ist vor allem in der Tatsache begründet, dass bisher stets eine Ausgangspose benötigt wird. Dies stellt ein Problem dar, welches nachfolgend erläutert wird.

Mithilfe der Kenntnis seiner ungefähren Position kann der Algorithmus bestimmen, welche Wand hinter dem Roboter liegt und dementsprechend nur diesen Ausschnitt des Scans nutzen. Danach wird mittels eines Histogramms der Winkel dieser Wand zum Roboter bestimmt und

der Scan um diesen Winkel korrigiert. Mittels weiterer Histogramme der x- und y- Koordinaten ist es möglich die gesamte Pose zu bestimmen.

Notwendigkeit der Ausgangspose

Die Kenntnis der ungefähren Position des Roboters ist zwingend erforderlich für den bisherigen Algorithmus, da das Spielfeld ist quadratisch ist. Der Algorithmus kann zwar leicht feststellen, welche Pose der Roboter zu einer Wand hat, er weiß aber nicht welche Wand, der vier möglichen Wände, dies ist. Daher stellt das bisherige Verfahren nur eine Korrekturmöglichkeit dar. Falls die reale Position um mehr als $\pm 1\ \mathrm{m}$ oder $\pm 20°$ von der zuletzt bekannten Pose abweicht, kann sich der Algorithmus nicht mehr eindeutig für eine Pose entscheiden. Dann stellt diese Situation ein Kidnapped-Robot-Problem dar und es wird bisher ein Fehler gemeldet, der zur Navigationsunfähigkeit des Roboters führt.

3.3.2. Linienextraktion

In [Gut99] wird ein Algorithmus zur Extraktion von Linien (Wänden) aus Scandaten vorgestellt. Dieser Ansatz soll an dieser Stelle kurz erläutert werden, da er später zur Detektierung der Wände dienen soll. Gedacht ist der Algorithmus für Scandaten, hinter denen sich größtenteils gerade Wände verbergen. Dazu kann der Scan beliebig groß sein. Das Vorgehen des Algorithmus wird an folgendem Beispiel erläutert. Abbildung 3.6 zeigt theoretische Scandaten eines einfachen Scans mit drei Wänden.

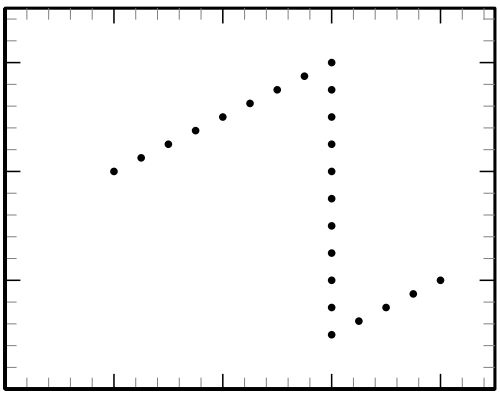


Abbildung 3.6.: Theoretische Rohdaten eines Laserscans mit 3 Wänden

Das Grundvorgehen besteht darin, dass der Algorithmus prüft, wie gut die Punkte zu einer Ausgleichsgeraden passen. Um den Grad der Abweichung zu bestimmen, benutzt man die Standartabweichung σ. Je kleiner dieser Wert ist, desto besser repräsentiert eine Punktwolke ihre Ausgleichsgerade. Die Berechnung der Standardabweichung (bzw. der Varianz σ^2) erfolgt

nach Formel 3.1:

$$\sigma^2 = \frac{1}{2n}\left(S_{x^2} + S_{y^2} - \sqrt{4S_{xy}^2 + (S_{y^2} - S_{x^2})^2}\right) \tag{3.1}$$

mit

$$S_{x^2} = \sum_{i=0}^{n-1} (x_i - \overline{x})^2 \tag{3.2}$$

$$S_{y^2} = \sum_{i=0}^{n-1} (y_i - \overline{y})^2 \tag{3.3}$$

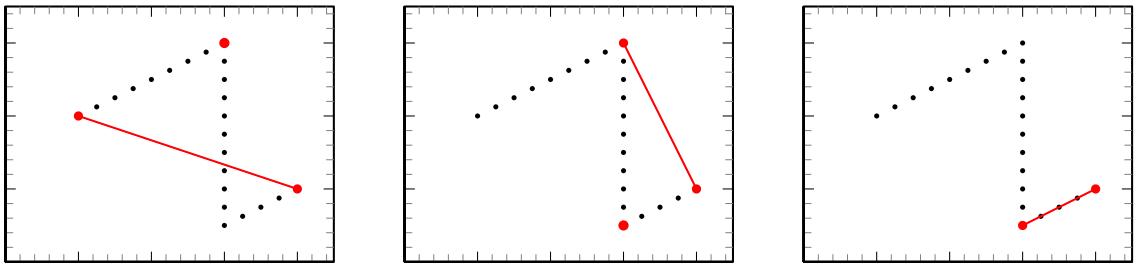
$$S_{xy} = \sum_{i=0}^{n-1} (x_i - \overline{x})\,(y_i - \overline{y}) \tag{3.4}$$

$$\overline{x} = \frac{1}{n}\sum_{i=0}^{n-1} x_i \tag{3.5}$$

$$\overline{y} = \frac{1}{n}\sum_{i=0}^{n-1} y_i \tag{3.6}$$

Ablauf des Algorithmus

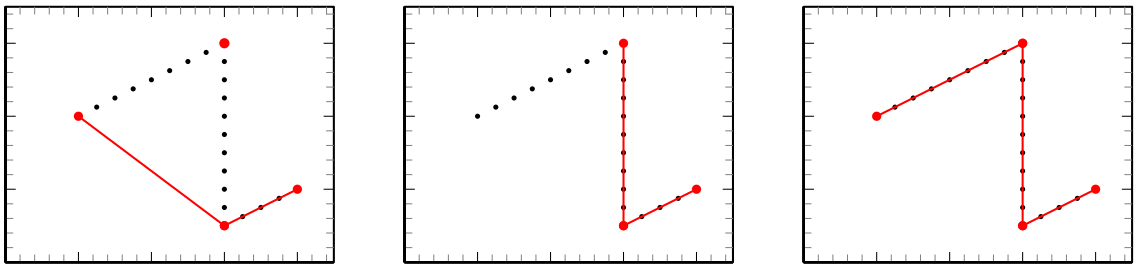
Zunächst wird angenommen, dass der Beginn und das Ende des Scans eine durchgehende Wand sind. Es wird also eine Gerade durch den ersten und letzten Punkt des Punktfeldes gelegt und die Standardabweichung berechnet. Ist sie wie hier zu hoch, wird der am weitesten davon entfernte Punkt berechnet (Abbildung 3.7a). Dieser Punkt wird nun als neue obere Grenze angenommen und die Berechnung der Standardabweichung erneut vorgenommen. Danach wird der Punkt bestimmt, der von der Ausgleichsgeraden am weitesten entfernt ist als neuer Endpunkt angenommen(Abbildung 3.7b). In Schritt 3 (Abbildung 3.7c) liegt die Standardabweichung schließlich unter der Schranke und das Intervall gilt als erkannte Linie.



(a) Schritt 1 **(b)** Schritt 2 **(c)** Schritt 3

Abbildung 3.7.: Schritt 1-3 der Linienextraktion

Nun wird der Endpunkt der erkannten Linie als Startpunkt der nächsten angenommen und der Endpunkt wieder auf das Ende des Feldes gesetzt (Abbildung 3.8a). Auf diese Weise wird das restliche Feld behandelt (Abbildungen 3.8b - 3.8c) und komplett in Linien aufgeteilt.



(a) Schritt 4 **(b)** Schritt 5 **(c)** Schritt 6

Abbildung 3.8.: Schritt 4-6 der Linienextraktion

Realistische Scandaten haben eine wesentlich höhere Streuung, weshalb es von entscheidender Bedeutung ist, wie man die Schranke für die Standardabweichung wählt. Ist sie zu hoch, werden möglicherweise Wände, die in einem stumpfen Winkel zusammen stehen, als eine Wand erkannt. Ist sie allerdings zu niedrig gewählt, wird eine Wand wegen hoher Streuung möglicherweise in viele kleine Wände bei der Extraktion zerteilt.

3.4. Anforderungen und Annahmen des Lokalisierungsalgorithmus

Da der globale Lokalisierungsalgorithmus Probleme beheben soll, die durch den lokalen Ansatz bisher nicht gelöst werden konnten, muss er mit diesem kompatibel sein. Aus diesem Grund müssen Änderungen an der Hardware so durchgeführt werden, dass sie den lokalen Ansatz nicht beeinträchtigen.

Die Genauigkeit, die durch den globalen Ansatz erreicht werden muss, hat die Anforderungen des lokalen Algorithmus zu erfüllen (Abschnitt 3.3.1). Der globale Lokalisierungsalgorithmus muss also die Pose des Roboter auf $\pm 1\ \mathrm{m}$ genau bestimmen können und der Winkelfehler muss kleiner als $\pm 20°$ sein. Anforderungen an die Rechenzeit des Algorithmus sind nicht hoch, da es sich um eine Backuproutine handelt. Es genügt daher, wenn die Rechenzeit unter $1\ \mathrm{s}$ liegt.

Der Algorithmus kann keinen völligen Ausfall mehrerer Roboter abfangen. Es muss davon ausgegangen werden, dass immer nur ein Roboter orientierungslos ist. Bei einem Ausfall von zwei oder gar drei Robotern können nicht mehr genug Informationen zur Verfügung stehen um einen Roboter zu lokalisieren.

3.5. Anbindung des Laserscanners

Um die Kompatibilität zum bereits vorhandenen lokalen Lokalisierungsansatz zu wahren, wurde die Anbindung des Laserscanners nicht verändert. Die hard- und softwareseitige Anbindung kann daher [Ler11] entnommen werden. Abbildung 3.9 zeigt die Befestigung des Laserscanners auf dem Roboter.

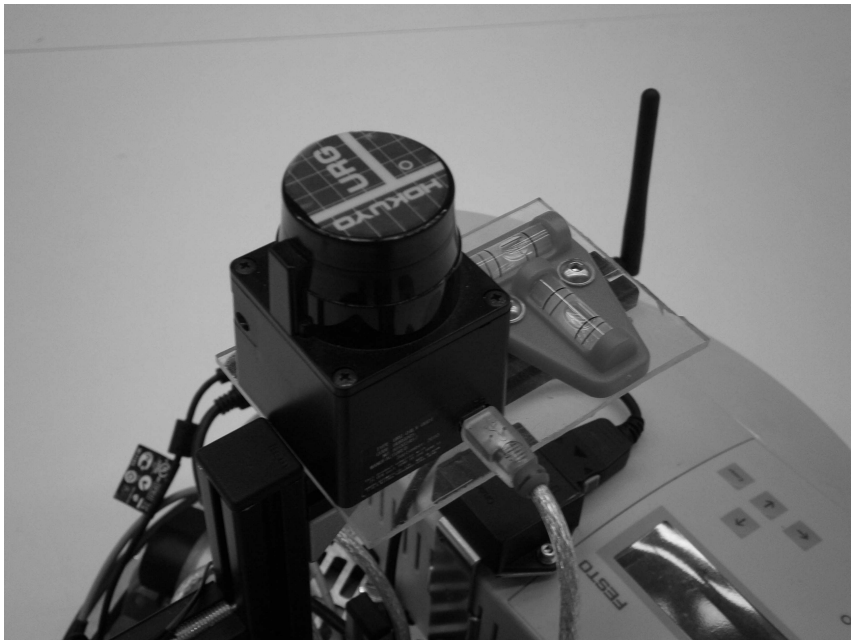


Abbildung 3.9.: Befestigung des Laserscanners auf dem Robotino®

4. Entwicklung des Algorithmus

Für die Behebung der Probleme des bisherigen lokalen Lokalisierungsalgorithmus ist es nötig, zusätzliche Orientierungspunkte zu schaffen, die das quadratische Spielfeld, um eindeutige Elemente erweitern. Für die Sensorik des Laserscanners kommen nur die Maschinen oder die Roboter selbst in Betracht.
Bei ersteren würden sich mehrere Probleme ergeben. Erstens sind es zu viele Maschinen, die dabei auch noch sehr gleichmäßig angeordnet sind. Weiterhin besteht der obere Teil zwar aus einem Zylinder und wäre damit als Landmarke gut geeignet, leider ist er aber auch aus einem durchsichtigen Werkstoff gefertigt und damit für den Laserscanner durchsichtig und somit nicht erkennbar. Der untere Teil der Ampeln hingegen ist zu groß und ein Absenken des Laserscanners auf dieses Niveau würde den ursprünglichen Algorithmus stark beeinträchtigen.

In Abschnitt 3.4 wurde die Annahme getroffen, dass nur einer von drei Robotern nicht lokalisiert ist. Durch eine lokale Lokalisierung der anderen zwei Roboter und die Übermittlung der Positionen an den zu lokalisierenden Roboter können ihm also die Positionsdaten zugänglich gemacht werden. Daraus ergibt sich zusammen mit dem Laserscan eine quasi-statische Situation. Daher wurde entschieden, die Roboter selbst in Landmarken zu verwandeln.

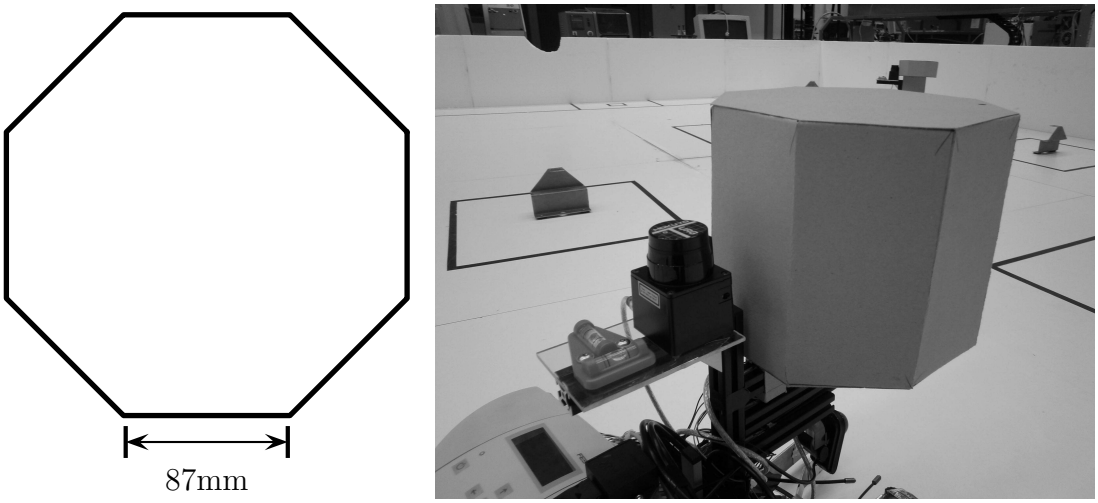
4.1. Entwurf der Landmarken

Nun liegt es Nahe, die Scanner als höchstes Bauteil auf dem Roboter zu positionieren. Dadurch wird der lokale Lokalisierungsansatz am geringsten beeinträchtigt und für die Laserscanner sind wirklich nur die anderen Roboter und die Wände sichtbar.
In der Vergangenheit wurde bereits festgestellt, dass die Beeinträchtigung der Scanner untereinander minimal ist, da sie mit einem Durchmesser von $40\ \mathrm{mm}$ kein Objekt im Scan darstellen. Die Planung und der Bau von größeren Landmarken wurde nötig, vier verschiedene Designs wurden getestet, welche im Folgenden kurz vorgestellt werden sollen.

4.1.1. Externe achteckige Marke

Für die ersten Testdaten, die in einer Reihe von statischen Tests erhoben wurden, wurde eine achteckige Marke gefertigt. Es wurde darauf geachtet, dass durch das achteckige Grunddesign keine spitzen Winkel entstehen, was ansonsten zu Totalreflexionen führen kann. Weiterhin stand die schnelle Fertigung im Vordergrund, weshalb zwei Marken aus Pappe hergestellt wurden.

Befestigt wurden sie an der schon vorhandenen Sensorik, so dass sie den Sensorbereich des Laserscanners nicht beeinträchtigen. Abbildung 4.1 zeigt die achteckige Marke.

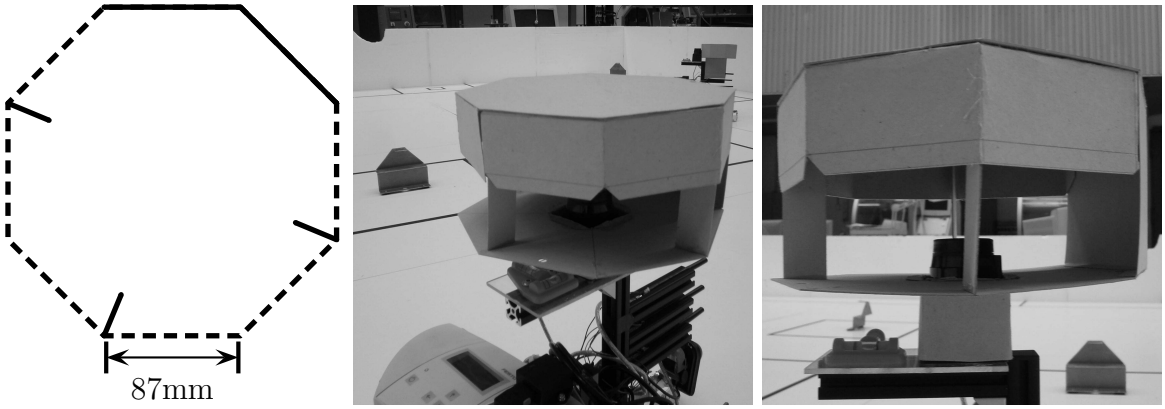


(a) Profil der Marke **(b)** Befestigung am Robotino®

Abbildung 4.1.: Externe achteckige Marke

4.1.2. 3-Lamellen-Marke

Nach einigen Fahrtests mit der externen achteckigen Marke wurde klar, dass sie durch schnelle Fahrbewegungen leicht in Schwingungen geraten. Es entstand die Idee, die Landmarke direkt auf dem Laserscanner zu befestigen. Hierbei besteht natürlich das Problem, dass die Landmarke einerseits für den innen befestigten Laserscanner durchsichtig sein muss, andererseits muss sie von außen für die anderen Laserscanner zu erkennen sein und das aus möglichst vielen Blickwinkeln. So wurde ein Design entwickelt, welches ein Fenster für den Scanner enthält. Leider waren die anderen Scanner auch so gut eingestellt, dass sie ebenfalls genau das Fenster trafen und die Marke ist unter schlechten Umständen einfach durchsichtig (Abbildung 4.2c). Positiv war die geringe Rückwirkung auf den umschlossenen Laserscanner, da seine Scandaten kaum beeinträchtigt waren. Das Design wurde beim Entwurf der 8-Lamellen-Marke verbessert. Abbildung 4.2 zeigt die 3-Lamellen-Marke.

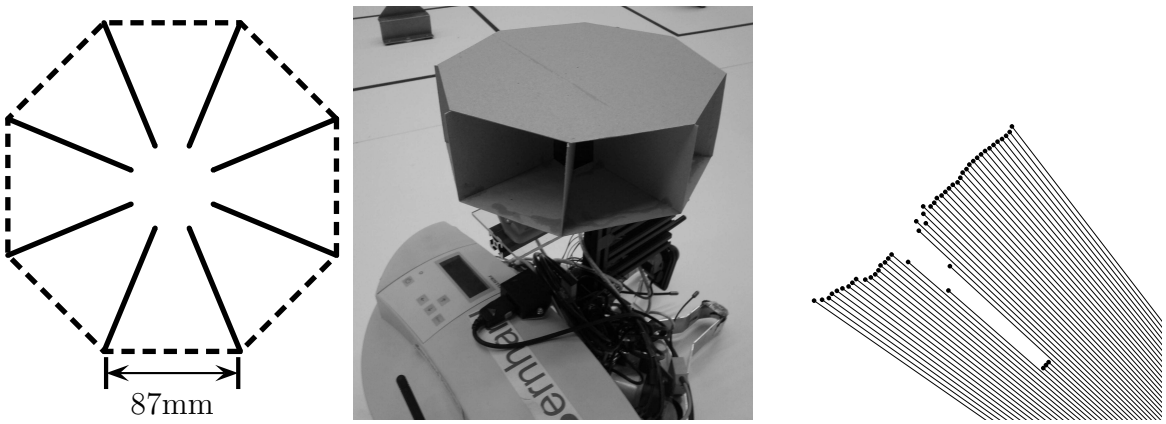


(a) Schnitt durch die Marke **(b)** Befestigung am Robotino® **(c)** Problemstellung

Abbildung 4.2.: 3-Lamellen-Marke

4.1.3. 8-Lamellen-Marke

Als dritter Versuch wurde eine Marke konstruiert, die komplett nur aus längeren Lamellen besteht. Davon wurde sich erhofft, dass aus jeder Sichtrichtung ein solider Körper erscheint, ohne den eingeschlossenen Laserscanner zu beeinträchtigen. Leider entsprach die Untersuchung nicht den Erwartungen. Das Scandatenbild für den umschlossenen Laserscanner wurde schlechter, was zur Folge hatte, dass an fünf Punkten des Scans mindestens 15 Scanpunkte hätten ausgeschlossen werden müssen. Das hätte die Datenmenge um 11% verringert. Abbildung 4.3c zeigt die Beeinträchtigung der Scandaten an den Lamellen. Weiterhin ergaben sich aus fast allen Blickrichtungen immer noch Lücken, in denen die Landmarke durchsichtig wurde. Abbildung 4.3 zeigt die 8-Lamellen-Marke.

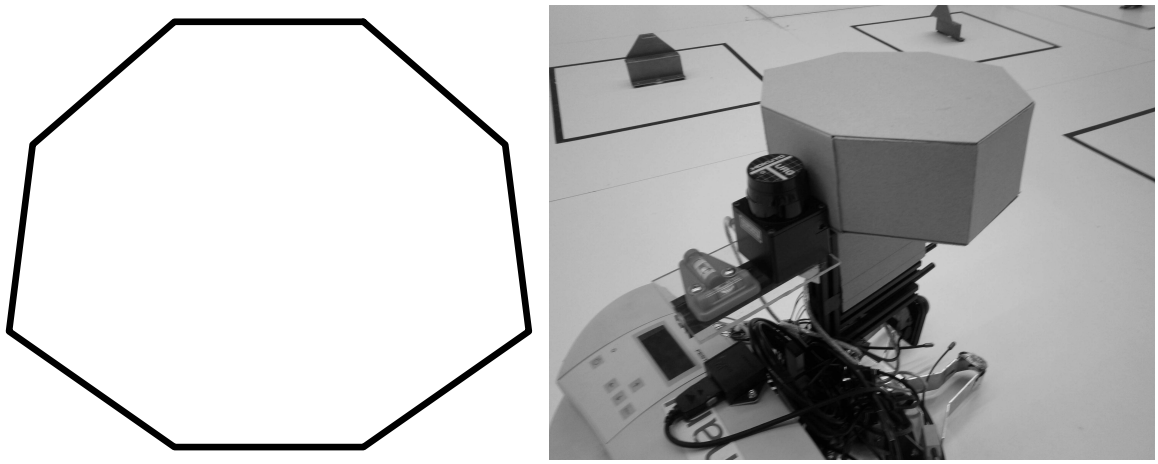


(a) Schnitt durch die Marke **(b)** Befestigung am Robotino® **(c)** Beeinträchtigung der Scandaten an den Lamellen

Abbildung 4.3.: 8-Lamellen-Marke

4.1.4. Modifizierte externe achteckige Marke

Aus den Erfahrungen der Landmarken, die in Abschnitt 4.1.1 - 4.1.3 vorgestellt wurden, entstand das endgültige Design. Da es die größte Zuverlässigkeit, bei gleichzeitig kleinster Beeinträchtigung hatte, wurde das Konzept der externen Marke wieder aufgenommen und die Grundform des Achtecks beibehalten. Aber um den Richtlinien der LL zu genügen und die Marke mehr der Kontur des Roboters anzupassen, wurde sie gestaucht und die Halterung versteift, so wie es Abbildung 4.4 zeigt.



(a) Profil der Marke **(b)** Befestigung am Robotino®

Abbildung 4.4.: Gestauchte externe achteckige Marke

4.2. Verarbeitung der Scandaten

Dieser Abschnitt soll die Prozeduren darstellen, die aus den Scan-Rohdaten eine Pose des Roboters bestimmen. Abbildung 4.5 zeigt eine Übersicht in Form eines Ablaufdiagramms, deren Teilalgorithmen nachfolgend beschrieben werden.

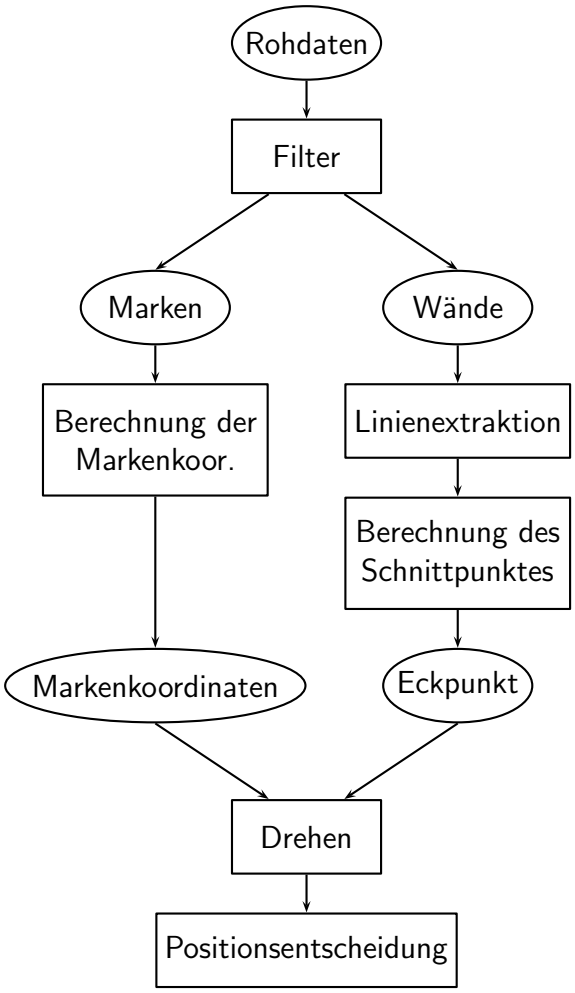


Abbildung 4.5.: Ablaufdiagramm des Algorithmus

4.2.1. Vorfiltern und Einlesen der Scandaten

Rohdaten des Scanners haben methodisch bedingt die Form von Polarkoordinaten. Für einen Großteil der weiteren Schritte müssen aber kartesische Koordinaten vorliegen. Um schon an dieser Stelle Rechenzeit zu sparen, werden ungültige Messwerte ausgefiltert, wobei alle Betragswerte $\leq 20\mathrm{mm}$ dabei Fehlercodes des Laserscanners darstellen. Weiterhin sind Messwerte $\leq 200\mathrm{mm}$ ebenfalls unplausibel, da dies der Radius des Robotinos® ist. Aus diesem Grund können alle Messwerte, die nicht $> 200\mathrm{mm}$ sind ausgefiltert werden. Sie werden gelöscht. Von den übrigen Messwerten wird dann aus der Nummer NR des Scanpunktes der Winkel α der Polarkoordinate bestimmt (Gleichung 4.1). Der Betrag r wird in Meter umgerechnet.

$$\alpha = \left((NR - 340) \cdot 0,3515625 \cdot \frac{\pi}{180}\right) + \pi \tag{4.1}$$

Nun können die kartesischen Koordinaten bestimmt werden. (Gleichungen 4.2 und 4.3)

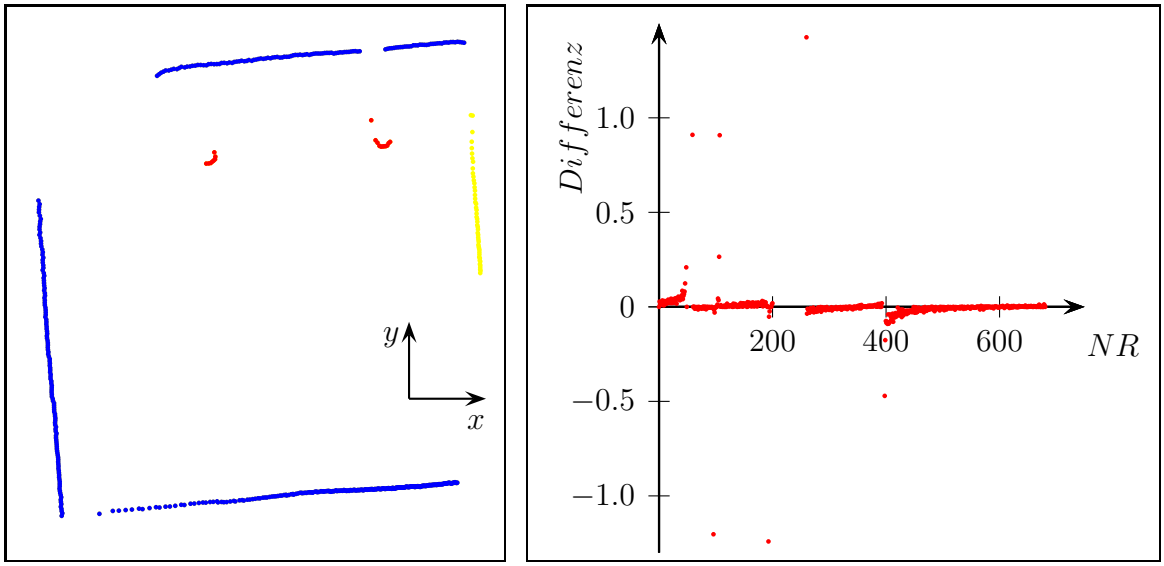
$$x = r \cdot \cos(\alpha) \tag{4.2}$$

$$y = r \cdot \sin(\alpha) \tag{4.3}$$

Um die folgenden Schritte vorzubereiten wird ebenfalls in dieser Schleife noch die Differenz zum vorangegangenen Betragswert abgelegt.

4.2.2. Haupt- und Nachfilter

Um in dem Scan Messwerte der Landmarken von Messwerten der Wände unterscheiden zu können, wird ein eindeutiges Kriterium gebraucht. Hierfür wurde die Differenz der Beträge der Messwerte gewählt, die im vorangegangenen Vorfilter bestimmt wurde. Die meisten Differenzen sind dabei sehr klein, da ein Objekt seine Distanz zum Sensor nicht in einem Winkelschritt signifikant ändert. Eine große Differenz tritt nur ein, wenn sich zwischen zwei Messwerten ein neues Objekt im Scan erscheint, wie z.B. eine Landmarke. Wenn also eine große negative Differenz vorliegt, dann ist es wahrscheinlich, dass der Scan von einer Wand auf eine Marke gesprungen ist. Ist wiederum eine große positive Differenz zu verzeichnen, dann ist es wahrscheinlich, dass der Scan von einer Marke zurück zu einer Wand gesprungen ist. Hier gilt es, eine Schranke für die Größe der Differenz festzulegen, um einen Sprung zu kennzeichnen. Wird die Schranke zu klein gewählt, werden auch Wandabschnitte als Marken erkannt. Ist sie hingegen zu groß, werden nahe der Wand stehende Marken nicht als Marke erkannt. Durch fehlerbehaftete Scandaten kann es bei diesem Separierungsvorgang zu Problemen kommen. Wie in Abschnitt 2.1.1 bereits erwähnt, kann es zu Totalreflexionen bei einem zu spitzen Winkel zwischen der Wand und dem Laserstrahl kommen. Dann kommt es zwischen dem zuletzt korrekt erkannten Wert und dem Nächsten, auf der benachbarten Wand zu einem großen Betragssprung. So werden Teile der Wand fälschlich als Marken erkannt. Um diese falschen Marken herauszufiltern, wurde ein einfacher Nachfilter nachgeschaltet. Dieser untersucht die erkannten Marken auf ihre Länge. Ist die Marke deutlich zu groß ($\geq 2 \cdot d_{Marke}$) bzw. zu klein ($\leq 0.5 \cdot d_{Marke}$), werden sie als Messfehler aussortiert. Abbildung 4.6 zeigt einen fertig gefilterten Scan mit dazugehörigem Plot der Differenz zum vorherigen Betrag über die Nummer des Messpunktes.



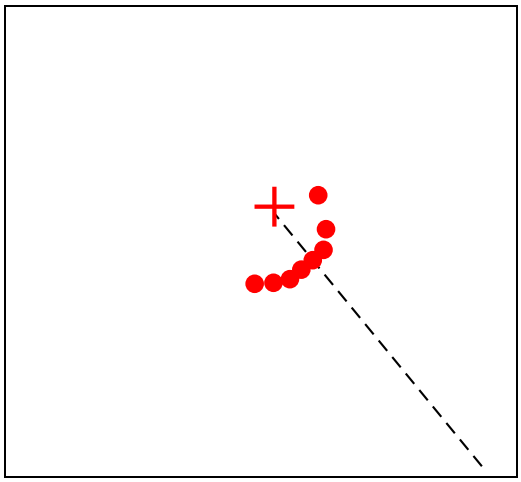
(a) gefilterter Scan **(b)** Differenz zum vorherigen Betrag

- Marken
- Wände
- falsch erkannte Marken

Abbildung 4.6.: Scan nach Haupt- und Nachfilter

4.2.3. Verarbeitung der Markendaten

Sind alle gültigen Marken ermittelt worden, wird aus jedem Punktfeld eine Markenpose ermittelt. Das Hauptproblem hierbei ist die Position der Marke auf dem Roboter, weshalb aus allen Punkten der Marke, der mit dem geringsten Abstand zum Scanner ausgewählt wird. Da die Marken ungefähr kreisförmig sind, ist davon auszugehen, dass dies der Punkt ist, der auf der Geraden liegt, die durch den Mittelpunkt des Scanners und den Mittelpunkt der Marke läuft. Ist dieser Punkt ermittelt worden, kann die Gerade zum Mittelpunkt, um den Radius, der Marke hin verlängert werden und ergibt so eine Position für den Mittelpunkt der Marke. Abbildung 4.7 illustriert dieses Vorgehen.



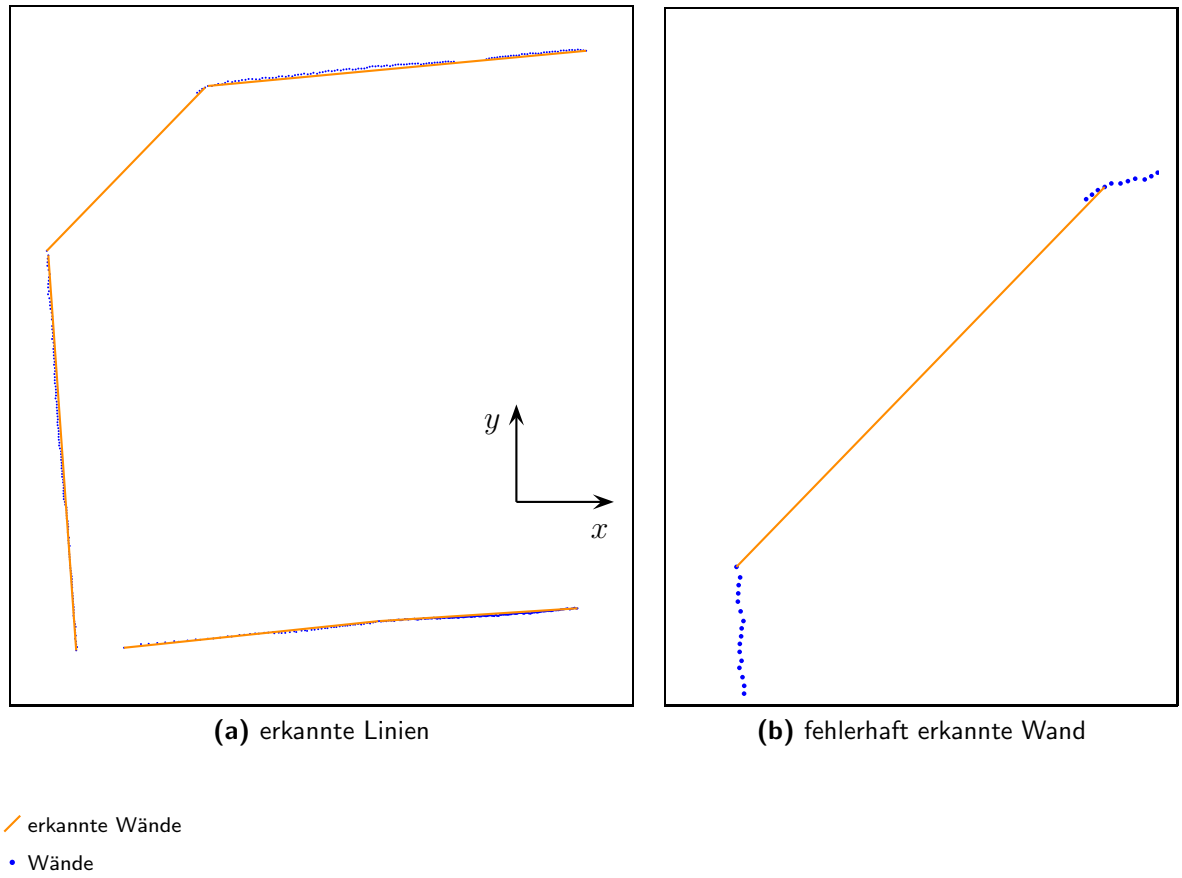
• Marken
+ gemessene Positionen der Landmarken

Abbildung 4.7.: Bestimmung der Markenposen

Dieses Vorgehen wird auf alle erkannten Marken angewendet und ergibt dann ein Feld aus Markenposen.

4.2.4. Linienextraktion

Um die möglichen Posen des Roboters ermitteln zu können ist es notwendig, einen Eckpunkt des Wandfeldes zu ermitteln. Zunächst ist es erforderlich, die Wände selbst zu erkennen, wobei das Verfahren der Linienextraktion zum Einsatz kommt, welches in Abschnitt 3.3.2 bereits erläutert wurde. Für diesen speziellen Anwendungsfall wurden einige Abwandlungen vorgenommen. So wird nach der Detektierung einer korrekten Linie nicht der Endpunkt dieser zum neuen Startpunkt, sondern der nächste Punkt des Wandfeldes. Dies hat die Bewandtnis, dass Lücken, die im Wandfeld bestehen, zu überspringen, falls diese sich in einer Ecke befinden. Um weiterhin zu vermeiden, dass eine Ecklücke übersprungen wird, wurde ein sehr kleiner Korrelationskoeffizient als Schranke gewählt. Dies begünstigt zwar die Zerteilung einer Wand in kleinere Abschnitte, verhindert aber die Ermittlung von Scheinwänden (Abbildung 4.8b). Da die spätere Betrachtung von kleineren Abschnitten hierbei das geringere Problem darstellt, wird einer kleineren Schranke für den Korrelationskoeffizienten der Vorrang gegeben. Abbildung 4.8a zeigt eine Linienextraktion, während das Detail in Abbildung 4.8b das Phänomen einer fehlerhaft erkannten Wand (durch Messwertstreuung an einer entfernten Ecke) veranschaulicht.



(a) erkannte Linien **(b)** fehlerhaft erkannte Wand

Abbildung 4.8.: Vorgang der Linienextraktion

4.2.5. Schnittpunktberechnung

Das Linienfeld ermöglicht die Berechnung der Eckpunkte, wobei nur der erste Schnittpunkt von zwei Linien berechnet wird, welcher die erste Ecke repräsentiert. Zur Berechnung eines Schnittpunktes von zwei Geraden ist es dabei notwendig, die kartesische Geradengleichung aufzustellen. Wie im vorherigen Abschnitt erwähnt, kann es bei der Bestimmung der Wände zu einer Teilung einer durchgehenden Wand in verschiedene Linien kommen. Sollte dies der Fall sein, ist es nicht nützlich, von diesen beiden Linien einen Schnittpunkt zu berechnen. Daher wird der Winkel der ersten Linie im Feld mit den folgenden verglichen. Damit eine folgende Linie zur Schnittpunktberechnung herangezogen werden kann, muss sich ihr Winkel um $90° \pm 10°$ von der ersten Linie unterscheiden. Wenn dies der Fall ist, kann davon ausgegangen werden, dass dies eine Linie auf der nächsten Wand ist und somit zur Berechnung eines Schnittpunktes geeignet ist. Ist diese Linie detektiert, werden beide Geradengleichungen aufgestellt und der Schnittpunkt aus ihnen berechnet. Abbildung 4.9 zeigt einen so bestimmten Schnittpunkt.

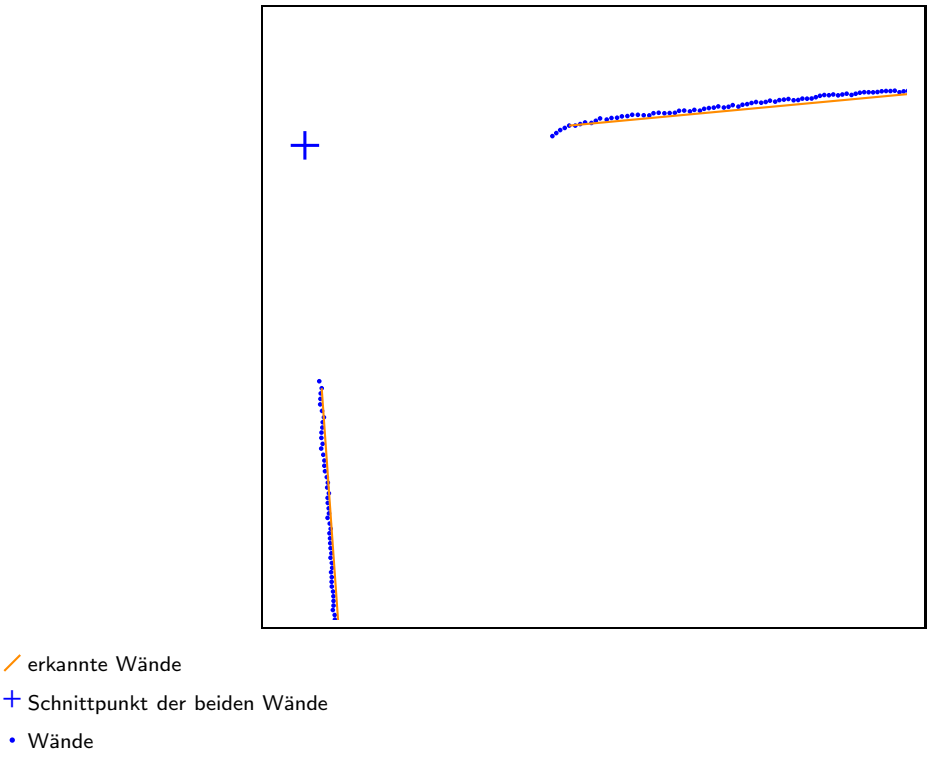


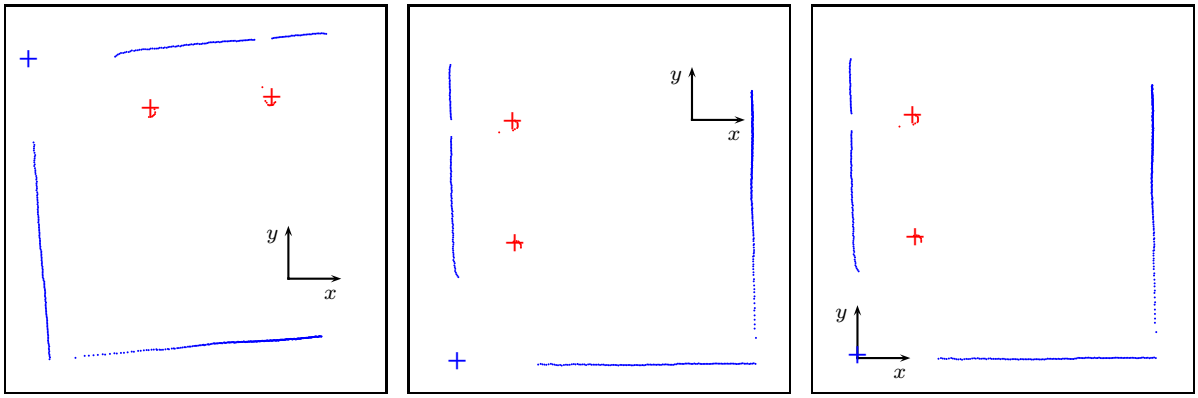
Abbildung 4.9.: Berechneter Schnittpunkt aus Linien zweier Wände

4.2.6. Drehen der Scandaten

Es liegen nun alle Informationen aus Sicht des Roboter-Koordinatensystems vor. Jetzt gilt es, diesen Scan so zu transformieren, dass er in das globale Koordinatensystem der Karte eingepasst werden kann. Zu diesem Zeitpunkt ist allerdings noch nicht klar, welche der vier Ecken der berechnete Schnittpunkt repräsentiert. Daher wird zunächst angenommen, dass es der Schnittpunkt im Koordinatenursprung des globalen Koordinatensystems ist. Daraus folgt, dass die zweite, zur Schnittpunktberechnung herangezogene Gerade, im globalen Koordinatensystem einen Winkel von $0°$ haben muss, weshalb alle lokalen Koordinaten entgegen des Winkels β dieser Geraden gedreht werden. Es wird der vermutete globale Koordinatenursprung in den dritten Sektor des Koordinatensystems verschoben. Eine weitere Korrektur um die gedrehten Koordinaten des Schnittpunktes, transformiert diesen zum globalen Koordinatenursprung und alle anderen Scandaten in ihre vermutete Position im globalen Koordinatensystem. Die Transformationsgleichnung lautet daher:

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{pmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{pmatrix} \begin{pmatrix} x_R \\ y_R \end{pmatrix} - \begin{pmatrix} x_{sch} \\ y_{sch} \end{pmatrix} \tag{4.4}$$

Der bisherige lokale Koordinatenursprung wird damit zur vermuteten Position des Roboters. Abbildung 4.10 verdeutlicht den Transformationsvorgang.



(a) lokale Scandaten **(b)** gedrehter Scan **(c)** fertig transformierter Scan

- Marken
+ gemessene Positionen der Landmarken
- Wände

Abbildung 4.10.: Transformation vom lokalen zum globalen Koordinatensystem

Aus der initial vermuteten Position $\binom{x_1}{y_1}$ müssen nun durch drei Koordinatentransformationen die weiteren möglichen Stellungen des Roboters abgeleitet werden.

$$\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} \cos 90° & -\sin 90° \\ \sin 90° & \cos 90° \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} l_{Feld} \\ 0 \end{pmatrix} \tag{4.5}$$

$$\begin{pmatrix} x_3 \\ y_3 \end{pmatrix} = \begin{pmatrix} \cos 180° & -\sin 180° \\ \sin 180° & \cos 180° \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} l_{Feld} \\ l_{Feld} \end{pmatrix} \tag{4.6}$$

$$\begin{pmatrix} x_4 \\ y_4 \end{pmatrix} = \begin{pmatrix} \cos 270° & -\sin 270° \\ \sin 270° & \cos 270° \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} 0 \\ l_{Feld} \end{pmatrix} \tag{4.7}$$

Abbildung 4.11 zeigt alle möglichen Positionen des Roboters.

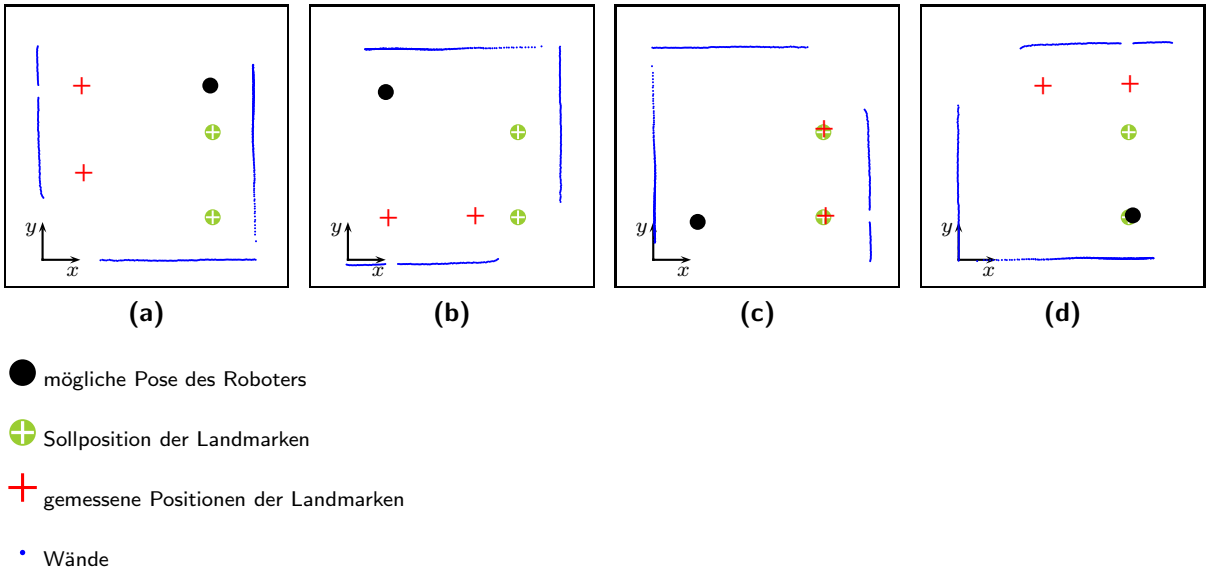


Abbildung 4.11.: Mögliche Posen des Roboters

4.2.7. Probabilistische Positionsentscheidung

Der finale Schritt der Lokalisierung erfolgt nun bei der Entscheidung für eine der vier Positionen. Hierbei werden zusätzlich zu den vom Roboter aufgenommenen Messdaten die bekannten Koordinaten der Landmarken betrachtet. Diese nehmen in allen Positionsvarianten dieselben Koordinaten an und haben damit eine variierende Stellung zu den vermessenen Landmarken. Es wird nun ein Kriterium benötigt, welches die Übereinstimmung von übermittelten und gemessenen Koordinaten quantifiziert. Hierfür wurde der minimale Abstand von den theoretischen zu den vermessenen Marken gewählt. In einer Schleife werden alle gemessenen Marken auf ihren geringsten Abstand zu den theoretischen Marken untersucht. Diese minimalen Abstände werden für jede mögliche Positionierung zu einem Identitätsfaktor aufsummiert, der die Übereinstimmung quantifiziert. Je geringer dieser ist, desto besser passen die gemessenen Marken zu den theoretischen. Nach diesem Faktor urteilend wird sich dann für eine der vier Möglichkeiten entschieden. Woraus eine vermutete Pose für den Roboter resultiert. Die x- und y-Koordinate können hierbei aus den unter Abschnitt 4.2.6 erzeugten Feldern übernommen werden. Die Winkelorientierung ergibt sich aus der Summe des initialen Drehwinkels und den Drehungen um $90°$ entsprechend der gewählten Möglichkeit.

$$\phi = \beta + (i - 1) \cdot 90° \tag{4.8}$$

Diese Pose kann als Schätzwert betrachtet werden und durch die Anwendung des weniger rechenintensiven lokalen Lokalisierungsansatzes (Abschnitt 3.3.1) verfeinert werden.

4.3. Implementierung

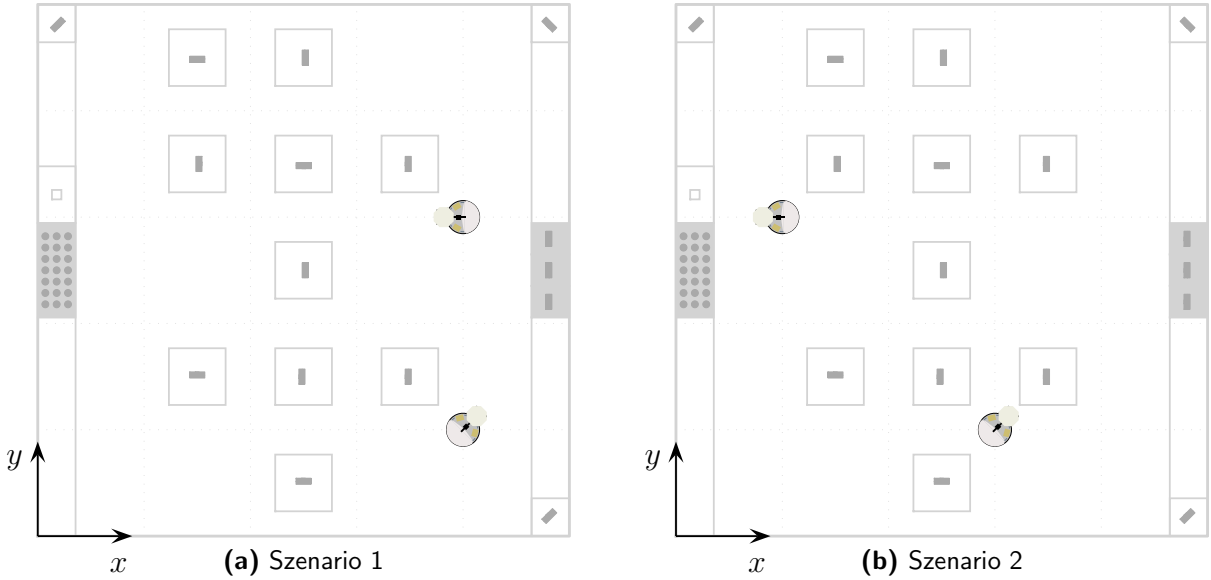
Umgesetzt wurde der Algorithmus in MATLAB, was geschah, um die Ergebnisse mittels grafischer Ausgaben schnell zu evaluieren.

Eine Implementierung in C++ wurde im Rahmen dieser Bachelorarbeit nicht umgesetzt, da zu diesem Zeitpunkt keine Umgebung für die direkte Ansteuerung des Robotino® zur Verfügung steht. Diese Implementierung wird zu einem späteren Zeitpunkt umgesetzt werden. Es sind dabei keine weiteren Probleme zu erwarten, da für den globalen Lokalisierungsalgorithmus keine spezifischen Funktionen verwendet wurden, die es nur in MATLAB gäbe.

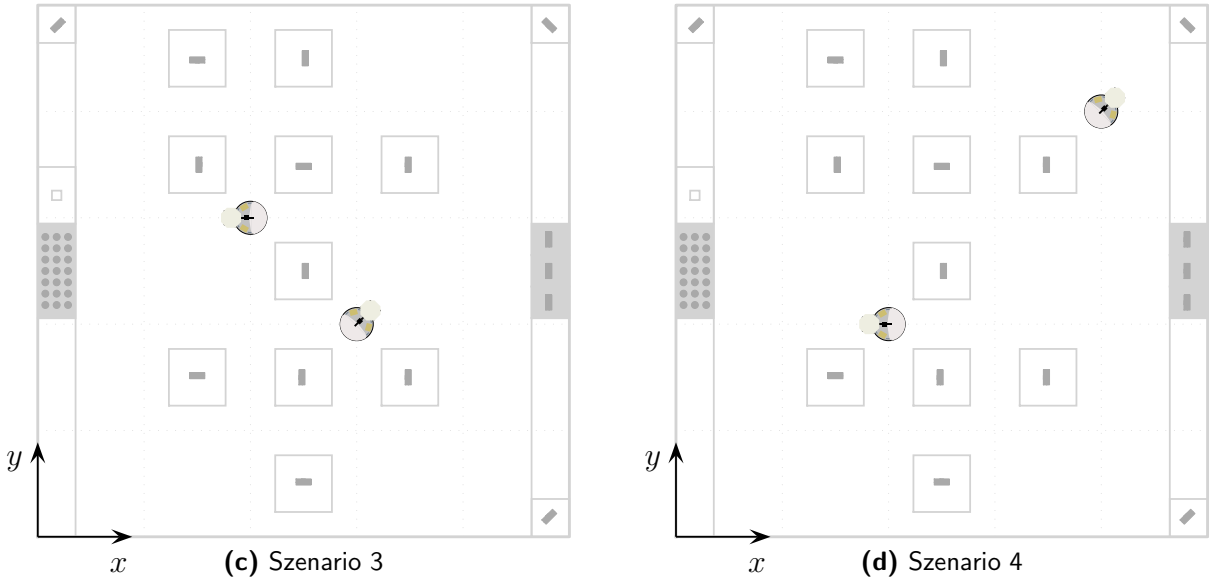
5. Ergebnisse

5.1. Validierung des Algorithmus

Für die Validierung des Algorithmus wurde während der gesamten Entwicklungszeit eine Auswahl zuvor aufgenommener Testdaten verwendet. Bei der Aufnahme der Testdaten lag das Augenmerk darauf, dass möglichst vielfältige Situationen nachgestellt werden. Es wurden vier verschiedene Testszenarien umgesetzt, in denen die jeweilige Position der Landmarken variiert. Um eine bessere Einlesbarkeit für den Algorithmus zu gewährleisten, wurden die Roboter dabei auf einem Gitter aufgestellt. Da das Feld rotationssymmetrisch zum Mittelpunkt ist, wurde der zu lokalisierende Roboter nur in einem Viertel des Feldes aufgestellt, da eine Ausweitung der Testdaten auf die anderen Viertel keinen Erkenntnisgewinn gebracht hätte. Daraus ergaben sich vier Punkte zur Aufstellung des sich lokalisierenden Roboters (A1-B2). Weiterhin wurde die Blickrichtung des Roboters in $45°$-Schritten variiert. Daraus ergeben sich maximal 32 Datensätze pro Szenario. Alle zur Validierung aufgenommenen Testdaten wurden in Anhang A beigefügt. In Abbildung 5.1 sind die vier verschiedenen Testszenarien dargestellt.



(a) Szenario 1 **(b)** Szenario 2



(c) Szenario 3 **(d)** Szenario 4

Abbildung 5.1.: Unterschiedliche Szenarien für die Testdaten

Bei einem finalen Durchlauf aller Testdaten konnten 75% dieser Testdaten korrekt erkannt werden. Die erkannte Pose lag dabei in den Grenzen, die in Abschnitt 3.4 aufgestellt wurden. Bei den restlichen Datensätzen wurden Posen falsch erkannt oder es ergaben sich Probleme, die im Folgenden dargestellt werden.

5.2. Aufgetretene Probleme

5.2.1. Nichtsichtbare Marken

Ein Großteil der nicht erkannten Posen war auf nicht sichtbare Landmarken zurückzuführen. Obwohl der Laserscanner einen Winkel von $240°$ abtastet, kann es doch passieren, dass sich beide Landmarken im toten Winkel des Scanners befinden. Um dieses Problem zu beheben, ist es notwendig, eine Rotation des Roboters einzuleiten, um den toten Winkel vermessen zu können. Man kann diese Testdaten nicht als Fehler ansehen, da ein Blick aus einem anderen Winkel häufig eine richtige Lokalisierung nach sich zog.

5.2.2. Spiegelsymmetrie

Ein geometrisches Problem ergab sich bei den Datensätzen der dritten Reihe (Abbildung 5.1c). Hier befinden sich beide Marken in Punktsymmetrie zum Mittelpunkt, weshalb je zwei mögliche Posen einen nahezu gleichen Identitätsfaktor haben. Es kam zu rund 50% falsch erkannten Posen, wobei es sich um ein Problem handelt, welches der Algorithmus nicht abfangen kann. Es kann höchstens der funktionsaufrufenden Instanz eine Rückmeldung gegeben werden, dass zwei Posen eine identische Wahrscheinlichkeit haben. Dann kann das Problem durch eine Wartezeit gelöst werden, bei der man eine Änderung der Positionen der Landmarken abwartet. Auch dies ist zwar ein unvorteilhafter Fehlerfall, aber ein nicht sehr wahrscheinlicher.

5.2.3. Weiterbewegende Landmarken

In einer weiteren Testreihe wurde die Robustheit des Algorithmus überprüft. Hierzu wurden dem Algorithmus vergangene Positionen der Landmarken mitgeteilt, während diese sich danach weiterbewegt haben. Es zeigte sich, dass solch eine Datenlage unvorhersehbares Verhalten auslöste, weil der Algorithmus nicht entscheiden konnte, welche theoretische Position welcher real vermessenen Landmarke zuzuordnen war. Hieraus ergibt sich die Notwendigkeit, von sehr aktuellen Informationen über die Position der Landmarken.
Abbildung 5.2 zeigt die simulierte Testsituation.

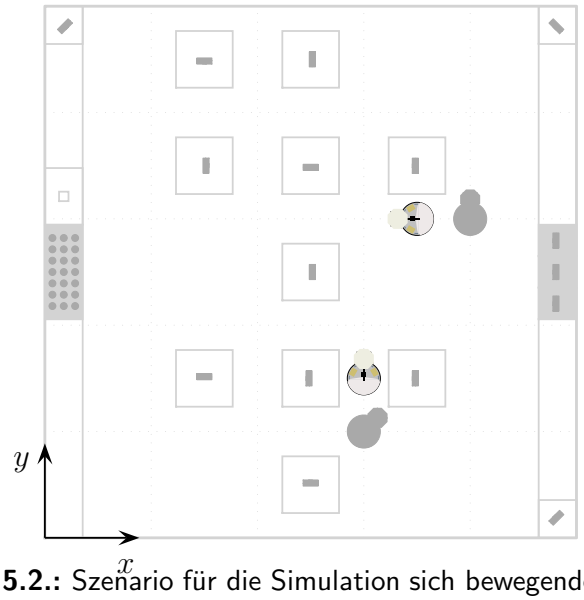


Abbildung 5.2.: Szenario für die Simulation sich bewegender Landmarken

5.3. Messung zur Bestimmung der Messungenauigkeit

Um die Genauigkeit des Algorithmus zu ermitteln, wurde ein Szenario von verschiedenen Punkten auf dem Spielfeld vermessen, die genau definiert sind. Dabei wurde ein maximaler Fehler von $\pm 0,269\mathrm{m}$ bei der Positionsbestimmung und ein maximaler Winkelfehler von $\pm 2,8^\circ$ erreicht. Das zeigt, dass der globale Lokalisierungsalgorithmus nicht die Genauigkeit des lokalen Lokalisierungsalgorithmus erreichen kann, aber hinreichend genau arbeitet, um die Eingangsdaten für diesen zu errechnen.

Abbildung 5.3 zeigt das verwendete Messszenario. Die Messdaten sind in Anhang C dargelegt.

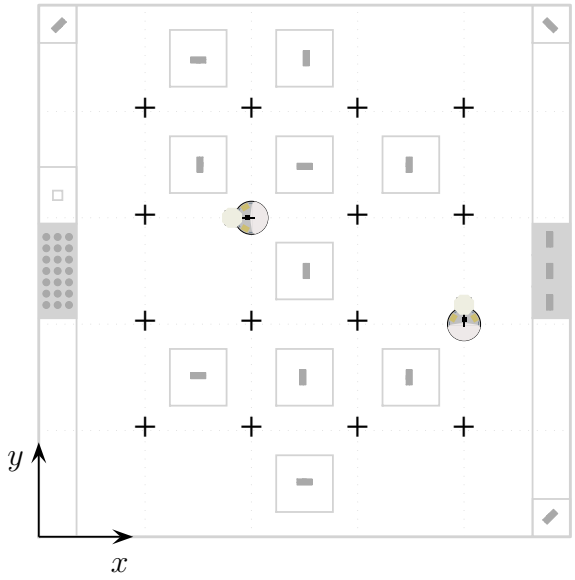


Abbildung 5.3.: Szenario für die Bestimmung der Messungenauigkeit

6. Zusammenfassung und Ausblick

6.1. Zusammenfassung

In dieser Arbeit wurde ein Algorithmus zur globalen Positionierung eines autonomen Roboters im Rahmen der LL des RoboCup umgesetzt. Hierbei wurden die Roboter mit Landmarken ausgerüstet, damit sie für die jeweils Anderen durch einen Laserscanner detektierbar sind. Das Ziel einen Algorithmus zu entwickeln, der aus diesen Sensordaten eine absolute Pose auf dem Spielfeld der LL in einer statischen Situation errechnet, wurde erreicht. Dabei wurden die Vorgaben erreicht, die durch den vorhanden lokalen Lokalisierungsalgorithmus gegeben sind. Daraus folgt wenn ein Roboter Zugang zu aktuellen Positionsdaten der übrigen Roboter hat, ist er in der Lage, seine Pose auf dem Spielfeld eindeutig zu bestimmen. Eine Aussage über die Auslastung der CPU auf der Hardwareplattform des Robotino® kann in dieser Arbeit, auf Grund der in Abschnitt 4.3 erwähnten Einschränkungen, nicht gegeben werden

6.2. Ausblick

Der in dieser Arbeit entwickelte Ansatz ist für eine klar strukturierte 2D-Umgebung leistungsfähig genug. Da aber die Gegebenheiten der LL sich weiter verändern werden, wird auch dieser Algorithmus einem weiteren Wandel unterworfen sein und sich mit dem Regelwerk weiterentwickeln.
Einen Ansatz aus [Del06] aufgreifend wurde die Idee evaluiert, den Laserscanner durch die Entwicklung einer Rotationsplattform auch in den Stand zu versetzen, ein 3D-Abbild seiner Umgebung anzufertigen. Hierfür wurde eine Testplattform konstruiert, um einen Testscan aufzunehmen. Abbildung 6.1 zeigt den errechneten 3D-Scan.

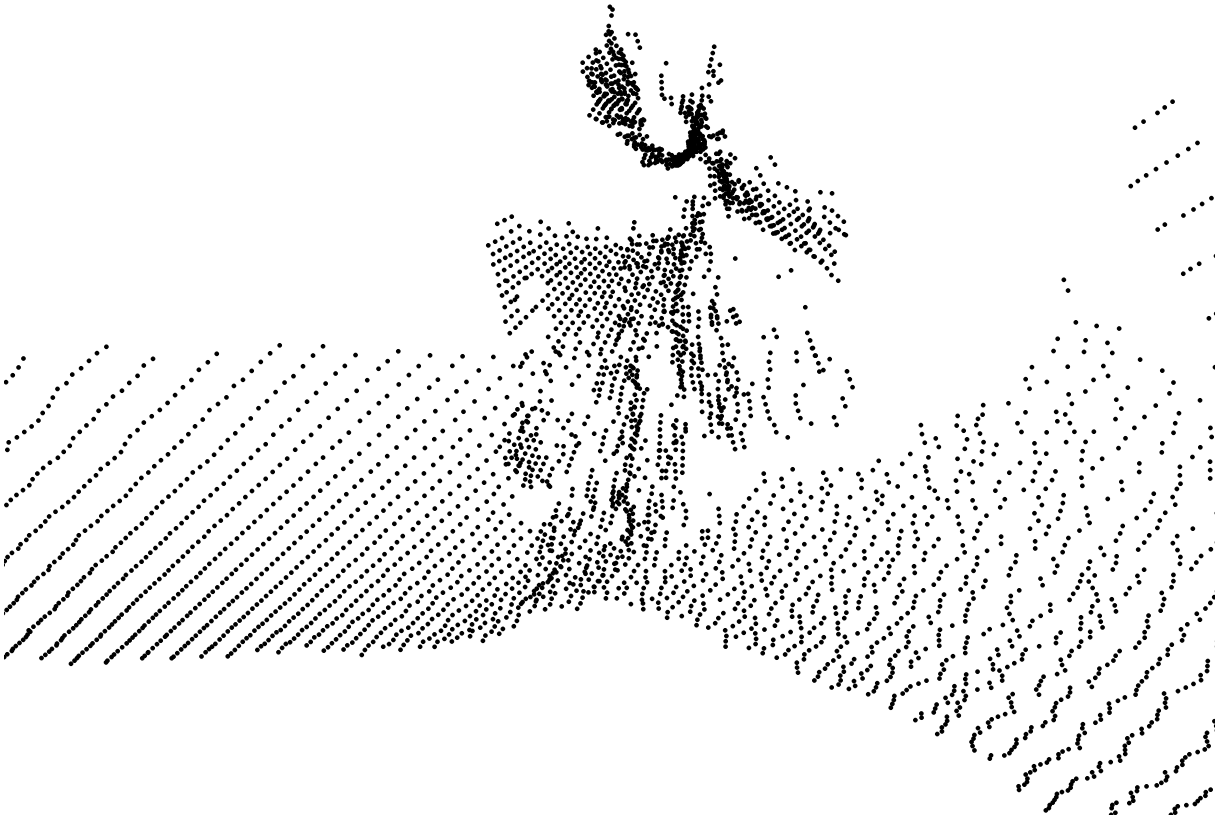


Abbildung 6.1.: 3D-Scan eines Robotino®

Literaturverzeichnis

[BEF96] Borenstein, Johann ; Everett, H.R. ; Feng, Liqiang: *Where am I? Sensors and Methods for Mobile Robot Positioning*. 1996

[Del06] Delis, Christian: *Entwicklung einer Rotationsplattform für den Hokuyo URG-04LX Laserscanner*, Universität Koblenz Landau, Studienarbeit, 2006

[Fes12] Festo Didactic (Hrsg.): *Logistics League 2012 Rulebook*. Festo Didactic, 2012. – Version: 1.801.2012

[GBH+88] Globus, Al ; Bailey, David ; Han, Jie ; Jaffe, Richard ; Levit, Creon ; Merkle, Ralph ; Srivastava, Deepak: NASA application of molecular nanotechnology. In: *The Journal of the British Interplanetary Society* 51 (1988), 145-152. http://www.resonancepub.com/nanotech.htm. – abgerufen am 04.04.2012

[Gut99] Gutmann, Jens-Steffen: *Robuste Navigation autonomer mobiler Systeme*. 1999

[HOK09a] HOKYO (Hrsg.): *Scanning Laser Range Finder URG-04LX-UG01 - Specifications*. HOKYO, 2009

[HOK09b] HOKYO (Hrsg.): *Simple-URG External Dimension*. HOKYO, 2009

[Kop11] *Kapitel* Zeitmessung. In: Kopp, Hartmut: *Taschenbuch der Messtechnik*. Hanser Verlag, 2011, S. 255

[Käs07] Käser, Erich: *Inkrementales Messsystem*. http://www.fachlexika.de/technik/mechatronik/pulscoder_schema.gif. Version: 2007. – abgerufen am 04.04.2012

[Ler11] Lerez, Christoph: *Lokalisierung mobiler Roboter mittels 2D- Laser Range Finder in strukturierter Umgebung*, Otto-von-Guericke-Universität Magdeburg, Studienarbeit, 2011

[Pen11] Pensky, Dr. D. ; Festo Didactic (Hrsg.): *Festo Logistics Competition - Roadmap*. Festo Didactic, 2011

[Pen12] Pensky, Dr. D. ; Festo Didactic (Hrsg.): *Logistics League RoboCup 2012*. Festo Didactic, 2012

[Plu11] Pluta, Werner: *Roboter räumen das Gelände des Atomkraftwerks*. http://www.golem.de/1104/82742.html. Version: 2011. – abgerufen am 27.03.2012

[rob12] Festo Didactic: *Robotino® ? Forschen und Lernen mit Robotern*. www.robotino.de. Version: 2012. – abgerufen am 04.04.2012

[Sch09] Schmucker, Ulrich: *Vorlesung: Grundlagen mobiler Roboter*. 2009

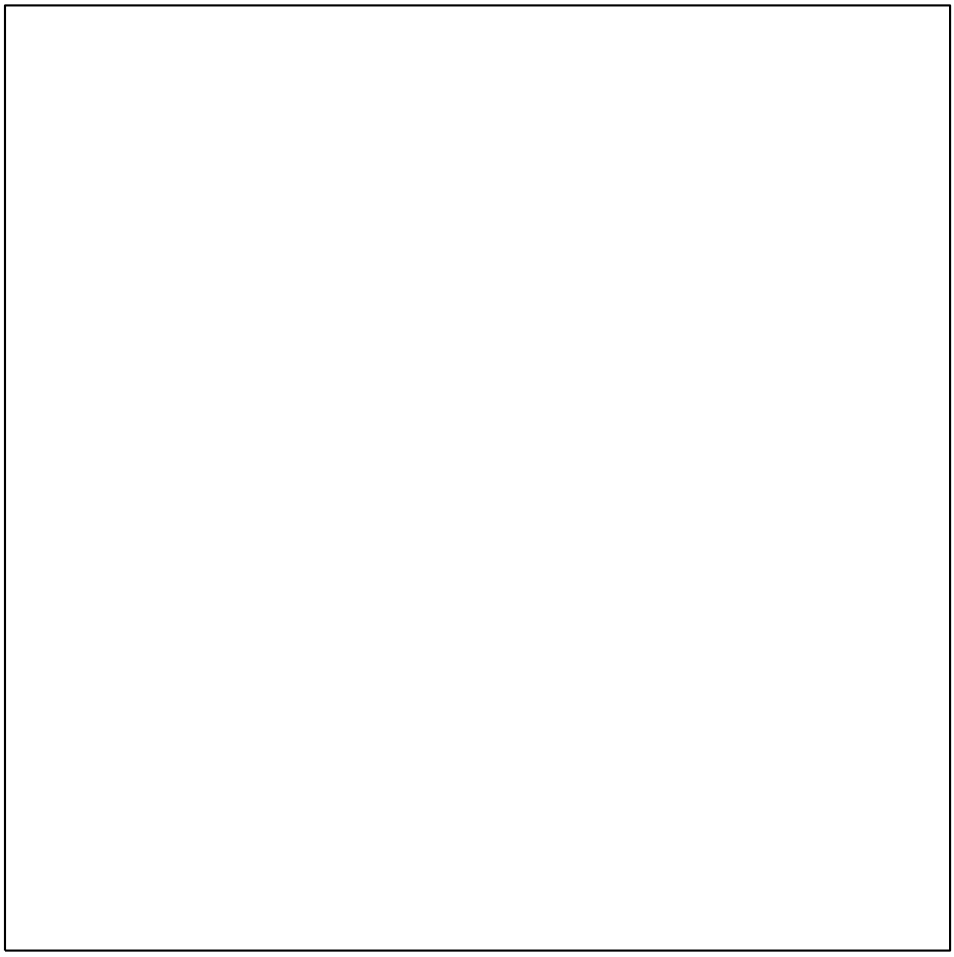
[Vol] Vollmer, Rosa: *Roboter erkunden erstmals Katastrophen-Reaktoren*. http://www.spiegel.de/wissenschaft/technik/0,1518,757732,00.html. – abgerufen am 27.03.2012

[Zen12] *Obrysches Gyroskop zur selbsttätigen Torpedosteuerung*. http://images.zeno.org/Meyers-1905/I/big/Wm19624b.jpg. Version: 2012. – abgerufen am 27.03.2012

A. Compact Disc

Inhalt:

- MATLAB-Dateien des globalen Lokalisierungsalgorithmus
- alle verwendeten Rohdaten
- PDF dieser Bachelorarbeit
- Rohdaten dieser Bachelorarbeit



B. Darstellung des Spielfeldes

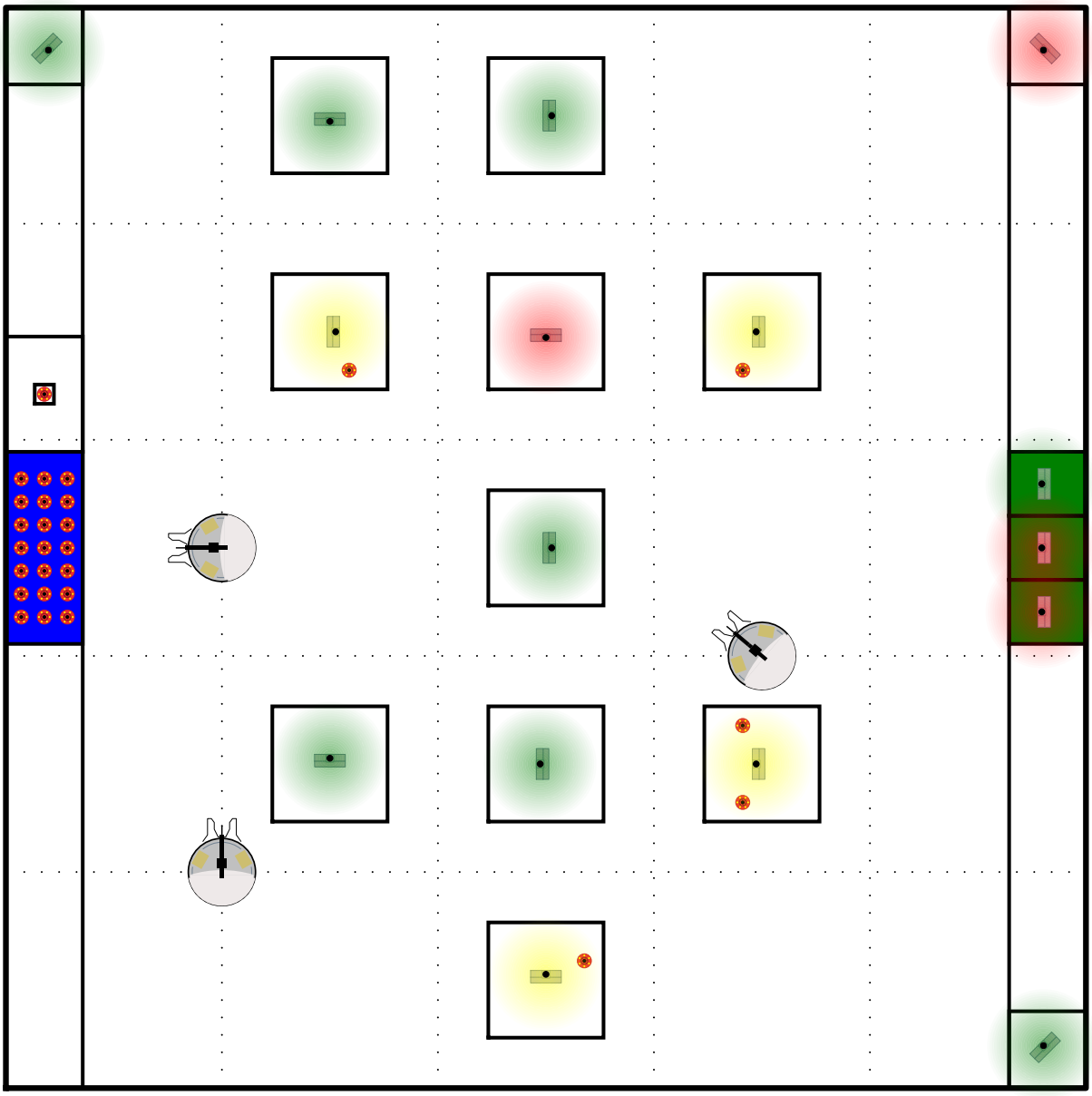


Abbildung B.1.: Spielfeld

C. Messabweichung der Posenbestimmung

vermessene Position	exakte Pose	gemessene Pose	Δx	Δy	$\Delta\phi$
A1	$\begin{pmatrix} 1{,}125\text{m} \\ 1{,}125\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 1{,}095\text{m} \\ 1{,}030\text{m} \\ -91{,}4° \end{pmatrix}$	$0,030$	$0,095$	$1,4$
A2	$\begin{pmatrix} 1{,}125\text{m} \\ 2{,}250\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 1{,}073\text{m} \\ 2{,}175\text{m} \\ -91{,}1° \end{pmatrix}$	$0,052$	$0,075$	$1,1$
A3	$\begin{pmatrix} 1{,}125\text{m} \\ 3{,}375\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 1{,}063\text{m} \\ 3{,}307\text{m} \\ -92{,}0° \end{pmatrix}$	$0,062$	$0,069$	$2,0$
A4	$\begin{pmatrix} 1{,}125\text{m} \\ 4{,}500\text{m} \\ 180{,}0° \end{pmatrix}$	$\begin{pmatrix} 1{,}029\text{m} \\ 4{,}402\text{m} \\ 178{,}7° \end{pmatrix}$	$0,096$	$0,098$	$1,3$
B1	$\begin{pmatrix} 2{,}250\text{m} \\ 1{,}125\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 2{,}188\text{m} \\ 1{,}114\text{m} \\ -92{,}2° \end{pmatrix}$	$0,062$	$0,011$	$2,2$
B2	$\begin{pmatrix} 2{,}250\text{m} \\ 2{,}250\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 2{,}166\text{m} \\ 2{,}155\text{m} \\ -91{,}3° \end{pmatrix}$	$0,084$	$0,095$	$1,3$
B3	$\begin{pmatrix} 2{,}250\text{m} \\ 3{,}375\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 2{,}146\text{m} \\ 3{,}330\text{m} \\ -92{,}8° \end{pmatrix}$	$0,104$	$0,045$	$\mathbf{2,8}$
C1	$\begin{pmatrix} 3{,}375\text{m} \\ 1{,}125\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 3{,}342\text{m} \\ 1{,}047\text{m} \\ -91{,}8° \end{pmatrix}$	$0,033$	$0,078$	$1,8$
C2	$\begin{pmatrix} 3{,}375\text{m} \\ 2{,}250\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 3{,}325\text{m} \\ 2{,}137\text{m} \\ -91{,}1° \end{pmatrix}$	$0,050$	$\mathbf{0,113}$	$1,1$
C3	$\begin{pmatrix} 3{,}375\text{m} \\ 3{,}375\text{m} \\ 180{,}0° \end{pmatrix}$	$\begin{pmatrix} 3{,}330\text{m} \\ 3{,}466\text{m} \\ 177{,}7° \end{pmatrix}$	$0,045$	$0,091$	$2,3$
C4	$\begin{pmatrix} 3{,}375\text{m} \\ 4{,}500\text{m} \\ 90{,}0° \end{pmatrix}$	$\begin{pmatrix} 3{,}424\text{m} \\ 4{,}507\text{m} \\ 88{,}3° \end{pmatrix}$	$0,049$	$0,007$	$1,7$
D1	$\begin{pmatrix} 4{,}500\text{m} \\ 1{,}125\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 4{,}231\text{m} \\ 1{,}035\text{m} \\ -91{,}9° \end{pmatrix}$	$\mathbf{0,269}$	$0,090$	$1,9$
D3	$\begin{pmatrix} 4{,}500\text{m} \\ 3{,}375\text{m} \\ -90{,}0° \end{pmatrix}$	$\begin{pmatrix} 4{,}492\text{m} \\ 3{,}361\text{m} \\ -1{,}3° \end{pmatrix}$	$0,008$	$0,014$	$1,3$
D4	$\begin{pmatrix} 4{,}500\text{m} \\ 4{,}500\text{m} \\ 0{,}0° \end{pmatrix}$	$\begin{pmatrix} 4{,}566\text{m} \\ 4{,}548\text{m} \\ -1{,}2° \end{pmatrix}$	$0,066$	$0,048$	$1,2$