AMCL und Odometrie - ein Vergleich

Jonathan Erhard, Konstantin Winkel
Robotik und Telematik
Universität Würzburg
Am Hubland, D-97074 Würzburg
jonathan.erhard@stud-mail.uni-wuerzburg.de konstantin.winkel@stud-mail.uni-wuerzburg.de

Abstract

Aufbauend auf dem von Giovanni und Corradini entwickelten Steuerungsgesetz für lineare Pfade wird ein Controller implementiert mithilfe dessen ein Vergleich der Lokalisierungsarten AMCl und Odometrie durchgeführt wird. Die Arbeit zeigt, dass die Pfadverfolgung mit Odometrie bei kurzen Strecken nicht weit vom ursprünglichen Pfad abweicht, sich die Fehler bei längeren Pfaden allerdings summieren, während AMCL keine Schwierigkeiten mit Akkumulation von Fehlern aufweist.

1 Einleitung

Weltweit ist Deutschland für seine technisch herrausragend ausgestatteten und luxuriösen Automobile bekannt [7]. Der nächste große technische Schritt, dem sich die Automobilindustrie momentan widmet, ist es, Autos autonom sicher fahren zu lassen. Hierzu werden zwar schon einige Prototypen getestet. Diese Technik ist jedoch noch lange nicht ausgereift und es gibt noch viele Probleme mit den bestehenden Software-Applikationen. Deshalb werden auch in der Zukunft Softwaredesigner benötigt werden, die sich mit den Grundlagen der Software von autonom fahrenden Autos auskennen und das Kartieren, Erstellen und Verfolgen von Pfaden verstehen sowie objektive Aussagen über die Qualität von verschiedenen Möglichkeiten zur Lokalisierung machen können. Natürlich sind sie Applikationen der Automobilindustrie viel weiter entwickelt als die im Verlauf dieser Arbeit beschriebenen. Daher geht es hier anhand von einigen Beispielen nur um grundlegende Methoden, wie solche Software funktionieren kann. In dieser Arbeit sollen daher ein Kartierungsalgorithmus und zwei Algorithmen zur Lokalisierung vorgestellt werden und gezeigt werden, wie man diese nutzen kann um die Güte der Lokalisierungsarten zu bestimmen.

2 Problembeschreibung

Ein sinnvoller Einsatz von mobilen Robotern ist nur möglich, wenn Positionen genau bestimmt werden können. Dies ist ein grundlegendes Problem der Robotik, das gelöst werden muss. Zwei Lokalisierungsarten, die dafür geeignet erscheinen, sind Adaptive Monte Carlo Localization und Odometrie. Im Rahmen dieses Praktikums soll ein fahrender Roboter vorgegebene Pfade mit beiden Lokalisierungsarten verfolgen. Die jeweiligen Abweichungen sollen quantifiziert und so ein objektiver Vergleich durchgeführt werden. Nach Auswertung der Ergebnisse soll die besser geeignete Lokalisierungsart bestimmt werden.

Diese Arbeit ist Bestandteil des Praktikums zur Mess- und Regelungstechnik

3 Stand der Technik

3.1 Hardware

Für das Praktikum ist die Arbeit mit einem Robotersystem vom Typ Volksbot vorgesehen. Die Systeme sind entweder mit dem VMC oder EPOS2 Motor Controller ausgestattet. Jedes der Robotersystem einen SICK LMS100 Laserscanner. Der Scanner hat einen Öffnungswinkel von 270° und nimmt innerhalb dieses Winkels einen horizontalen Schnitt der Umgebung um die Front des Volksbots auf [3]. Um die Manövrierung des Roboters so einfach wie möglich zu gestalten, verfügt dieser über einen Differentialantrieb. Ein solcher Antrieb kann die Geschwindigkeiten der beiden Räder unabhängig voneinander einstellen. Zur Benutzung des Roboters ist ein Laptop mit ROS nötig. Die grundlegenden Dateien sowie eine detaillierte Beschreibung, um den Roboter zu starten und mit einem Gamepad zu steuern, wurden uns zur Verfügung gestellt [14]. Die Kommunikation mit dem Roboter wird über eine USB-Schnittstelle realisiert, während die Messdaten des SICK LMS100 Laserscanners über ein Ethernetkabel zur Verfügung gestellt werden.

3.2 ROS - Robot Operating System

ROS ist eine seit über 10 Jahren verfügbare Open Source Software Library zum einfachen Designen und Implementieren von Roboterapplikationen [9] [13]. Neben den Libraries zur Entwicklung stellt ROS ebenfalls Tools zum Erstellen und Testen dieser Applikationen zur Verfügung, zum Beispiel RViz und ROS-Bag. RViz ist eine 3D-Visualisierungsumgebung, welche dem Entwickler einen Einblick in das gibt, was der Roboter mit Hilfe seiner Sensoren "sieht" [12]. ROS-Bag stellt ein Command-Line-Tool zur Verfügung, mit dem Nachrichten aus ROS-Topics gespeichert, abgespielt und analysiert werden können [10]. Viele ROS-Pakete stellen Launchfiles zur Verfügung, mit denen sofort mehrere Nodes gestartet und verschiedene Parameter gesetzt werden können. Damit lassen sich komplizierte Programme sehr einfach starten [11].

3.3 Odometrie

Odometrie ist eine sehr einfache Möglichkeit, die Pose eines Roboters relativ zu einem Startpunkt zu bestimmen. Dazu wird fortlaufend die Orientierung und Geschwindigkeit des Roboters sowie die verstrichene Zeit gemessen. Dies wird als Koppelnavigation bezeichnet [6]. Werden dabei die ersten beiden Parameter (Orientierung und Geschwindigkeit) durch Winkelmessungen an den Rädern bestimmt, so spricht man von Odometrie. Odometrie kann statt zur relativen Lokalisierung auch zur absoluten Lokalisierung verwendet werden, wenn die absoluten Koordinaten des Startpunktes bekannt sind.
Da der Roboter eine Differentiallenkung besitzt, lässt sich so ein Modell für die Odometrie herleiten. Sei Θ die Orientierung des Roboters in Z-Richtung im mathematisch negativem Sinn, b der Abstand zwischen den beiden betrachteten Rädern sowie v_r und v_l die Geschwindigkeiten des rechten und linken Rads, welche als konstant angenommen werden. Der Drehwinkel, welcher auch als Änderung des Winkels Θ bezeichnet werden kann, kann folgendermaßen beschreiben werden [4]:

$$\frac{\mathrm{d}\Theta}{\mathrm{d}t} = \frac{(v_l - v_r)}{b}. \tag{1}$$

Die Integration über die Änderung der Orientierung $\frac{\mathrm{d}\Theta}{\mathrm{d}t}$ zusammen mit dem Startwert der Orientierung Θ_0 ergibt die Orientierung zum Zeitpunkt t.

$$\Theta_t = \Theta_0 + \frac{(v_l - v_r)t}{b} \tag{2}$$

Hiermit lassen sich nun, wie in [16] beschrieben, Gleichungen für die zeitliche Ableitung der Position in X- und Z-Richtung bestimmen:

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= (\frac{v_l + v_r}{2}) \cdot \sin\Theta_t \\ \frac{\mathrm{d}z}{\mathrm{d}t} &= (\frac{v_l + v_r}{2}) \cdot \cos\Theta_t. \end{aligned} \tag{3}$$

Aus diesen erhält man durch Integration über die Zeit die Position des Roboters in X- und Z-Richtung zum Zeitpunkt t:

$$\begin{aligned} x_t &= (\frac{v_l + v_r}{2}) \cdot \int \sin\Theta_t \mathrm{d}t \\ z_t &= (\frac{v_l + v_r}{2}) \cdot \int \cos\Theta_t \mathrm{d}t. \end{aligned} \tag{4}$$

Odometrie liefert besonders über kurze und gerade Strecken sehr gute Ergebnisse. Über längere und gekrümmte Strecken jedoch akkumulieren sich durch ungenaue Winkelmessungen und die Integration der Positionsänderung Fehler, welche über alle Grenzen wachsen können. Diese Arbeit wird zeigen, dass aufgrund dieser Fehler Odometrie nicht für längere Strecken geeignet ist. Dennoch ist sie ein wichtiges Hilfsmittel zur Lokalisierung, da mit der Fusion der Odometriemesswerte mit unabhängigen Messwerten anderer Sensoren (zum Beispiel eines Laserscanners) die Fehler erheblich verkleinert werden können. Dies soll ebenfalls im Zuge dieser Arbeit demonstriert werden.

3.4 GMapping

GMapping ist ein Rao-Blackwellized Partikelfilter, mit dem man das Problem der simultanen Lokalisierung und Kartographierung (SLAM) lösen kann. Als Eingaben für diesen Filter werden zum einen Odometrie, zum anderen die Daten eines Laser-Range-Scanners benutzt. GMapping ist für Langstreckenscanner wie SICK LMS- oder PLS-Scanner optimiert [18].
Für dieses Praktikum sollte mit Hilfe der bereitgestellten Launch-Files zu GMapping eine Karte des Untergeschosses des Informatik-Gebäudes angefertigt werden. Dafür wurde eine ROS-Bag erstellt, während der Roboter per Gamepad durch das Untergeschoss gesteuert wurde. In dieser ROS-Bag wurden alle Daten, die vom Roboter zur Verfügung gestellt wurden, insbesondere die Odometriedaten und die Daten des Laserscanners, gespeichert. Mit der ROS-Bag und den bereitgestellten Launch-Files konnte eine Karte angefertigt werden. Diese Karte konnte danach mit RViz angezeigt und analysiert und im Anschluss mit dem Map-Saver-Tool des Map-Server-Pakets von ROS abgespeichert werden [5].
Den Vorgang des Kartierens sieht man im beigefügten Video zu dieser Arbeit[1]. Im Vorfeld wurden einige Bereiche an den Wänden mit Ordnern abgedeckt. Dies sind Felder, in denen Metall in die Wand eingelassen ist (zum Beispiel als Halterung für Feuerlöscher). Hier konnte der Laserscanner die Daten nicht richtig auswerten und verwechselte so eine feste Wand mit einem begehbaren Abschnitt. Ein ähnliches Verhalten war bei Glaswänden zu beobachten (siehe Ziffern 1-4 in Abbildung 1). Abbildung 2 zeigt die Odometriedaten des Roboters während der Kartierungsfahrt.

[1] https://drive.google.com/file/d/189bfWs88sH6pSbycHe954gEAW2w69VUO/view?usp=share$_l ink$

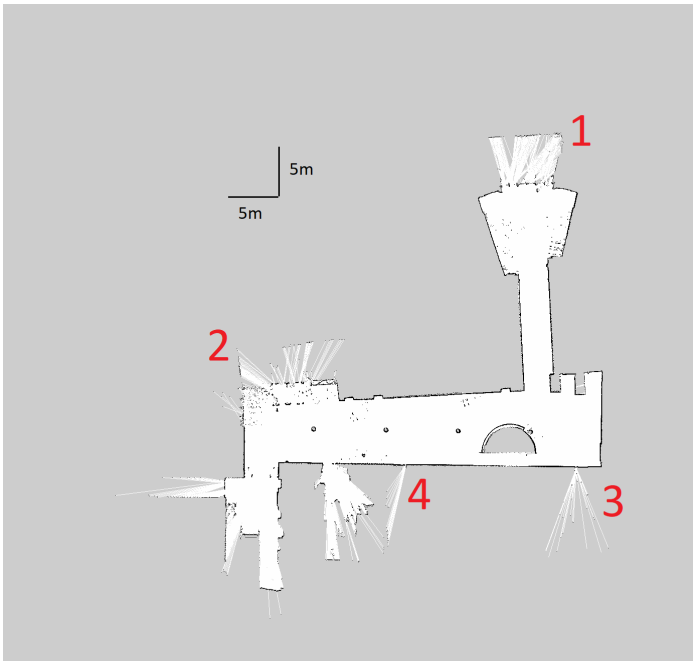

Figure 1: Karte des Untergeschosses des Informatikgebäudes. Weiß: Flächen, die vom Scanner erfasst wurden und als erreichbar zählen. Schwarz: Bereiche, die der Scanner als feste Wände identifiziert hat. Grau: Gebiete, über die dem Scanner keine Informationen vorliegen. Ziffern 1-4: Wände aus Glas, die der Scanner nicht als Wände erkennt.

3.5 AMCL

Adaptive Monte Carlo Localization (AMCL) ist ein Partikelfilter, der durch Approximationen die Position des Roboters in einer vorgegebenen Karte bestimmt. Dafür wird zuerst eine Startposition benötigt, die manuell gesetzt werden muss. Um diese Startposition wird nun eine Punktwolke generiert, die sogenannten Partikel. Diese Partikel stellen die Wahrscheinlichkeiten darstellen, zu denen sich der Roboter an der entsprechenden Position befindet. Bewegt sich nun der Roboter, werden die aufgenommenen Daten des Laserscanners mit der vorgegebenen Karte abgeglichen und so die Wahrscheinlichkeiten der Partikel angepasst. So konvergieren mit der Zeit alle Partikel auf einen Punkt, der die Position darstellt, an dem sich der Roboter befindet [1] [2]. Abbildung 3 zeigt die AMCL-Daten des Roboters während der Kartierungsfahrt.

3.6 Implementierung des Controller

Um diese Lokalisierungsarten sinnvoll nutzen zu können, wird ein Controller benötigt, mit dem man Pfade verfolgen kann. Dafür wurde der bereits in der Vorlesung ,,Automatisierungs- und Regelungstechnik" vorgestellte Giovanni-Controller gewählt. Der Großteil des Controllers wurde im Rahmen des Praktikums bereitgestellt. Die mathematischen Grundlagen sind in [15] nachzulesen. Das von Giovanni Indiveri und Maria Letizia Corradini veröffentliche Paper stellt unter anderem das Steuerungsgesetz

$$\omega = -h \cdot u \cdot y \cdot \frac{\sin\theta}{\theta} - \gamma \cdot \theta \tag{5}$$

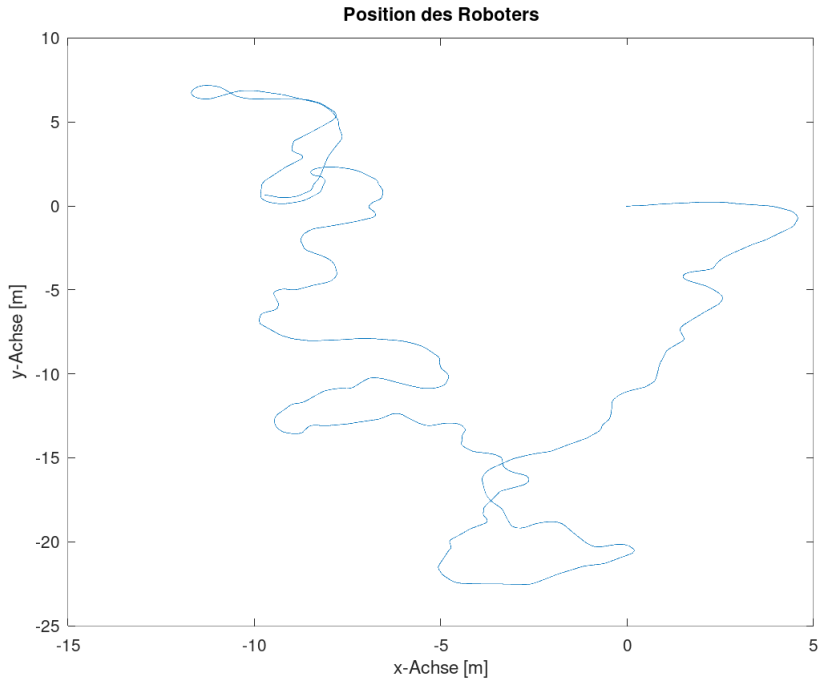

Figure 2: Odometriedaten während der Kartierungsfahrt, geplottet wird die Position mit X- und Y-Koordinate in Meter. Der Startpunkt liegt bei (0,0).

vor, das asymptotische Konvergenz auf einem linearen Pfad garantiert. y ist dabei die Höhe des Roboters über der Referenzgeraden, u ist die Geschwindigkeit, die über die gesamte Fahrt konstant bleibt, θ ist der Winkel zwischen der Referenzgeraden und der Orientierung des Roboters. h und γ sind Konstantinen, die frei wählbare sind. Die berechnete Größe ω ist die Winkelgeschwindigkeit des Roboters [15]. Gleichung 5 sagt aus, wie sich der Roboter bewegen muss, um eine Gerade anzufahren, wenn die Pose des Roboters relativ zur Geraden bekannt ist. Um einen komplizierteren Pfad anzufahren, wird der Pfad zunächst in Punkte aufgeteilt. Daraufhin wird mit den ersten beiden Punkten eine Sekante konstruiert, die zunächst als Referenzgerade dient. Sobald der Roboter den zweiten Punkt passiert hat, wird das nächste Punktepaar zur Definition der nächsten Sekante verwendet. Diese Vorgehensweise wird angewandt, bis der Roboter den letzten Punkt in der Liste erreicht hat [8].

4 Konzept des Experiments

Um den Vergleich zwischen AMCL und Odometrie durchführen zu können, wurde ein eigener Pfad mit einer ROS-Bag aufgenommen. Mit Hilfe der ROS-Bag und der ROS-Node zur Erzeugung von neuen Pfaden wurden dann zwei Pfaddateien erzeugt, wobei die Basisdaten einmal aus dem Odometrie- und einmal aus dem ACML-Topic entnommen wurden. Danach wurden diese Pfade abgefahren - die Odometriedatei mit Odometrie und die AMCL-Datei mit AMCL als Input für den Controller. Dabei blieben die Lokalisierungsarten streng von einander getrennt. Um einen Anhaltspunkt für die Setzung der Markierungen zu haben, wurde in die Mitte der Front des Roboters ein Schraubenzieher als Hilfswerkzeug geklebt (sieht Abbildung 4). In regelmäßigen Abständen hielt der Roboter an, sodass Markierungen auf den

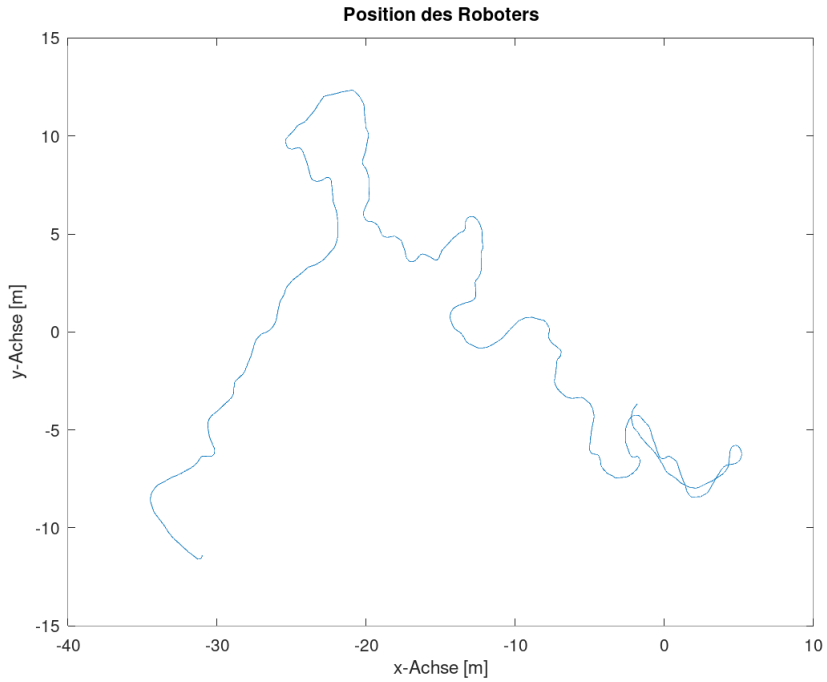

Figure 3: AMCL-Daten während der Kartierungsfahrt, geplottet wird die Position mit X- und Y-Koordinate in Meter. Der Startpunkt ist bei ca. (-31,-12).

Boden genau unter dem Schraubenzieher gesetzt werden konnten. Mit den Markierungen der Odometrie- und AMCL-Pfade und den Markierungen des ursprünglichen Pfads konnte die Abweichung vom ursprünglichen Pfad bestimmen werden, wie in Abbildung 5 dargestellt wird. Hierbei sind der schwarze und grüne Punkt Punkte des Orginalpfades und der blaue Punkt ein Punkt, bei dem der Roboter angehalten hat. Die rote Strecke bildet das Lot zwischen dem Anhaltepunkt und der Verbindungsstrecke der Punkte des Orginalpfades. Durch Messen der roten Strecke lässt sich die Abweichung quantifizieren. Um danach eine quantitative Aussage darüber zu erhalten, welche Lokalisierungsmethode genauer ist, wird über alle Messpunkte gemittelt und die dadurch erhaltenen durchschnittlichen Abweichungen miteinander verglichen. Dies wurde einmal mit einem kürzeren und einmal mit einem längeren Pfad durchgeführt.

5 Experimente und Ergebnisse

5.1 Kurzer Pfad

Der kürzere Pfad bildet eine geschlossene Kurve. Er beinhaltet mehrere Kurven, die Winkelgeschwindigkeit veränderte sich aber nur in wenigen Bereichen stark (Bildarchiv [17]). Der kürzere Pfad wurde mit Odometrie etwas präziser als mit AMCL absolviert. Die durchschnittliche Abweichung bei Odometrie betrug 6,29 cm, bei AMCL 8,07 cm. Der Roboter mit AMCL als Lokalisierung kam genau am Startpunkt wieder an, während der Roboter mit Odometrie am letzten Punkt eine Abweichung von ca. 16 cm aufwies. Daraus folgt, dass Odometrie für kürzere Strecken die bessere Wahl ist. Das bestätigt die Aussagen in [16]. Dennoch ist AMCL für kürzere Pfade ebenfalls eine geeignete Lokalisierungsart, da der Wert

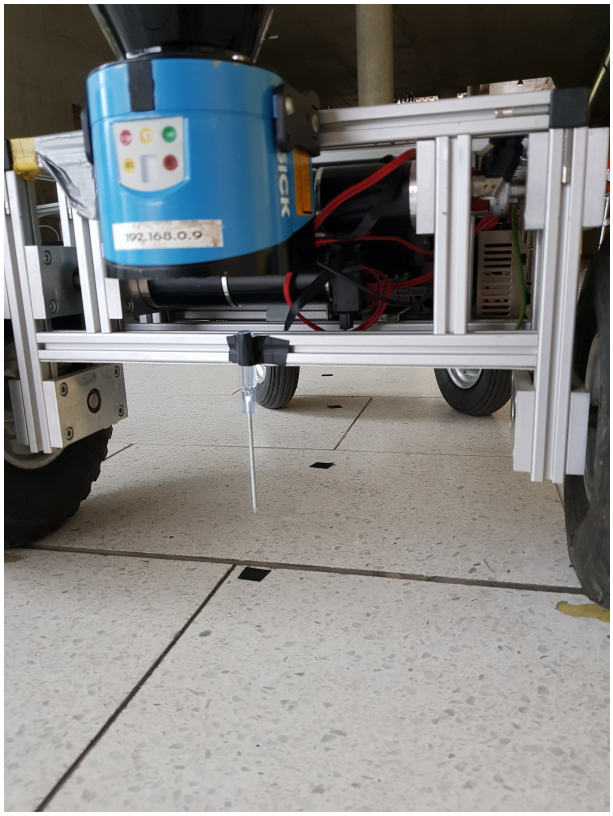

Figure 4: Modifikation des Roboters: Ein in der Mitte der Front befestigter Schraubendreher zeigt nahe am Boden den Mittelpunkt zwischen den Rädern und hilft beim Setzen der Markierungen

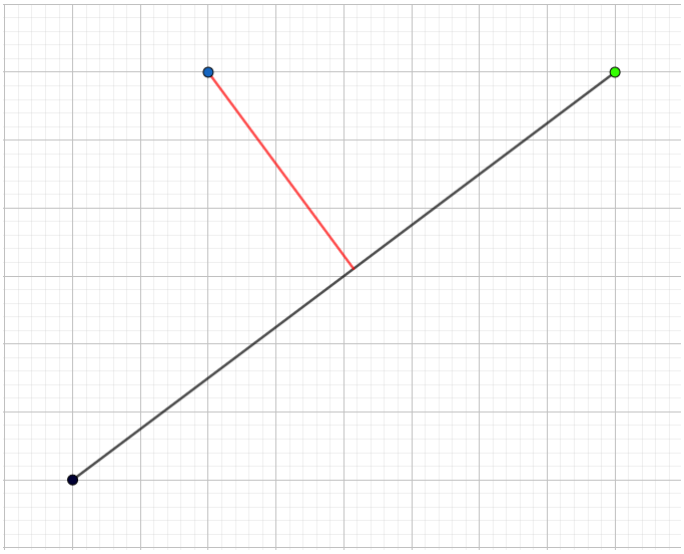

Figure 5: Beispiel zur Bestimmung der Abweichung: Der schwarze und der grüne Punkt sind Teil des Orginalpfads, der blaue Punkt ist ein Messpunkt. Durch Messung der roten Strecke kann die Abweichung quantifiziert werden.

des durchschnittlichen Fehlers in der gleichen Größenordnung liegt als bei Odometrie. Der höhere Fehlerwert könnte daher kommen, das AMCL nicht genug Zeit während der gesamten Fahrt hatte, um seine Position korrekt abzuschätzen. Ein möglicher Grund dafür könnte in der niederfrequenten Veröffentlichung der aktuellen Position bei AMCL liegen. Weitere Fehler könnten durch den Ort des Experiments kommen, da es in der Nähe eines großen Fensters durchgeführt wurde, was zu Schwierigkeiten bei der Verwendung des Laserscanners führen kann (vergleiche Abbildung 1). Für die genauen Messwerte vergleiche Abschnitt 7.

5.2 Langer Pfad

Der längere Pfad erstreckt sich über einen Großteil des Untergeschosses des Informatikgebäudes. Da der Pfad um alle drei dort vorhandenen Pfeiler ging, enthielt er viele Stellen, an denen sich die Winkelgeschwindigkeit stark veränderte. Der AMCL-Controller konnte den gesamten Pfad nachfahren und wies dabei nur geringfügige Abweichungen auf. Der durchschnittliche Fehler lag hier bei 4,91 cm, die gleiche Größenordnung wie beim kurzen Pfad. Odometrie hingegen wies schon recht früh größere Abweichungen vom Orginalpfad auf. Bereits am 4. Punkt betrug die Abweichung schon 10 cm. Diese Abweichungen wurden im Laufe des Pfades immer größer, bis hin zu einer maximalen Abweichung von 124,5 cm am 23. Messpunkt. Am 27. Messpunkt musste dann die Fahrt mit dem Controller abgebrochen werden, da der Roboter sonst in einen der Pfeiler gefahren wäre. Bei Odometrie lag die durschnittliche Abweichung über die vorhandenen Messpunkte verteilt bei 44,20 cm - das entspricht etwa dem neunfachen von AMCL.
Hieran lässt sich feststellen, dass die Integrationsfehler, die sich bei Odometrie mit der Zeit anhäufen, bei langen Pfaden so groß werden, dass Odometrie nicht mehr zur Lokalisierung nutzbar ist. Für die genauen Messwerte vergleiche Abschnitt 7.

6 Zusammenfassung und Ausblick

Im Zuge dieser Arbeit wurden die beiden Lokalisierungsarten Odometrie und AMCL verglichen. Wie erwartet hat sich gezeigt, dass Odometrie bei kurzen Fahrten eine genaue Positionsbestimmung ermöglicht, sich bei längeren Pfaden allerdings zu viele Fehler akkumulieren, um eine präzise Ortung zu erreichen. Im Gegensatz dazu ermöglicht AMCL auch bei einer langen Fahrt eine präzise Lokalisierung. Die Genauigkeit der Lokalisierungsmethoden ist allerdings nur eine Facette der verschiedenen Ortsbestimmungsvarianten. Da die Verarbeitung der AMCL- und Odometriedaten bei den beschriebenen Experimenten mit leistungsstarken Laptops durchgeführt wurden, sind beispielsweise Fehler durch den unterschiedlichen Rechenaufwand nicht ins Gewicht gefallen. Bei Anwendungen mit niedriger Rechenleistung können diese Fehler aber nicht mehr vernachlässigt werden und können in weiteren Arbeiten genauer erforscht werden.

7 Anhang

Table 1: Gemessene Abweichungen vom Orginalpfad in Centimetern.

Kurzer Pfad	**Odometrie**	**AMCL**	**Langer Pfad**	**Odometrie**	**AMCL**
1	7	6	1	3,5	4,5
2	6	0,7	2	2,5	4,5
3	2	8	3	4	7,4
4	5,5	4,3	4	10	0,5
5	1,2	5,4	5	10,3	7,7
6	3,1	11,1	6	21,5	1,4
7	7,4	16	7	25	0
8	2,7	20,7	8	32	2,5
9	1,7	17,8	9	29	6,7
10	3,4	9,6	10	19,5	3,2
11	4	3,5	11	40,5	11,5
12	5,5	8,8	12	23	14
13	4,4	9,5	13	33,5	4
14	12,1	6,1	14	44,5	0,5
15	9.3	15,8	15	70,5	2
16	15,2	4	16	80,5	4,3
17	6,4	5,5	17	68,7	1,2
18	16,3	0,5	18	89	2,5
19		0	19	32,5	3,2
20			20	21	2,8
21			21	52	1,5
22			22	98,5	3
23			23	124,5	1,5
24			24	116,5	3,5
25			25	60,5	1
26			26	5	3
27			27	75,5	8,5
28			28		9,5
29			29		10,5
30			30		9,5
31			31		9,5
32			32		10,5
33			33		8,4
34			34		7
35			35		4
36			36		1,5
AVG	6,29	8,07	AVG	44,20	4,91

References

[1] Adaptive Monte Carlo Localization. https://roboticsknowledgebase.com/wiki/state-estimation/adaptive-monte-carlo-localization/. Accessed: 2022-09-11.

[2] AMCL - Package Summary. http://wiki.ros.org/amcl. Accessed: 2022-09-14.

[3] LMS100-10000 — Mess- und Detektionslösungen — SICK. https://www.sick.com/mess-und-detektionsloesungen/2d-lidar-sensoren/lms1xx/lms100-10000/p/p109841. Accessed: 2022-10-24.

[4] Lokalisierung und Odometrie eines Roboters mit differentiellem Antrieb. https://spacehal.github.io/docs/robotik/odometrie. Accessed: 2022-10-24.

[5] map_server - Package Summary. http://wiki.ros.org/map_server. Accessed: 2022-10-24.

[6] Navigation - RN-Wissen.de. https://rn-wissen.de/wiki/index.php/Navigation#Koppelnavigation. Accessed: 2022-10-19.

[7] Premiumfahrzeuge - Marktanteile der Herstellerländer — Satista.

[8] ROS-Code zum Praktikum. https://www.ros.org/. Erstellt im Rahmen des Praktikums.

[9] ROS: Home. https://www.ros.org/. Accessed: 2022-10-16.

[10] rosbag - Package Summary. http://wiki.ros.org/rosbag. Accessed: 2022-10-16.

[11] roslaunch - Package Summary. http://wiki.ros.org/roslaunch. Accessed: 2022-10-24.

[12] rviz - Package Summary. http://wiki.ros.org/rviz. Accessed: 2022-10-16.

[13] Why ROS? https://www.ros.org/blog/why-ros/. Accessed: 2022-10-16.

[14] Fabian Arzberger, Andreas Nüchter, Jasper Zevering. Irma3D - Getting Started.

[15] Giovanni Indiveri and Maria Letizia Corradini. Switching linear path following for bounded curvature car-like vehicles. *IFAC Proceedings Volumes*, 37(8):185–190, 2004. IFAC/EURON Symposium on Intelligent Autonomous Vehicles, Lisbon, Portugal, 5-7 July 2004.

[16] Joachim Herzberg, Kai Lingemann, Andreas Nüchter. *Mobile Roboter*. Springer Verlag, 2012.

[17] Jonathan Erhard, Konstantin Winkel. Bilderarchiv zu dieser Arbeit. https://drive.google.com/file/d/1aOoQrr1fBRJzhYzIwYFZBe13gN00omeb/view?usp=sharing. Erstellt im Rahmen des Praktikums.

[18] Wolfram Burgard, Giorgio Grisetti, Cyrill Stachniss. Gmapping. https://openslam-org.github.io/gmapping.html. Accessed: 2022-09-11.