Automatisches Mapping des dynamischen Umfelds in cyber-physischen Systemen

Taihao Li[1*], Marian Göllner[1], Sven Jacobitz[1], Xiaobo Liu-Henke[1]

[1]Institut für Mechatronik, Ostfalia Hochschule für angewandte Wissenschaften, Salzdahlumer Str. 46/48, 38302 Wolfenbüttel; *ta.li@ostfalia.de*

Kurzfassung. Im Rahmen des Niedersächsischen Zukunftslabors für Mobilität hat die Fachgruppe um Prof. Liu-Henke ein Cyber-Physisches-Labortestfeld zur Echtzeit-Erprobung und -Optimierung intelligenter, vernetzter, autonomer Fahrfunktionen entwickelt. Eine wesentliche Voraussetzung für die autonome Fahrt in cyber-physischen Systemen ist das automatische Mapping. Dieses muss die Umgebung hinreichend genau abbilden und dazu dynamisch an geänderte Umfeldbedingungen angepasst werden. In diesem Beitrag wird unter Nutzung des Robot Operating Systems (ROS) eine Funktion zur kontinuierlichen, automatischen Kartierung des Umfelds im CPS-Testfeld ausgelegt. Durch Auswertung von Zustandssensoren zur Umfelderfassung und anschließener Sensordatenfusion wird die Grundlage zur Erstellung einer dynamischen Umfeldkarte gelegt.

Einleitung

Die intelligente Mobilität wird in der Hightech-Strategie des Bundesministeriums für Bildung und Forschung [1] als eine der wichtigsten Zukunftstechnologien bewertet. Das automatisierte und vernetzte Fahren spielt dabei eine wesentliche Rolle um die Anforderungen der Mobilität der Zukunft zu erfüllen und einen sauberen, leisen, zuverlässigen und verbrauchsgünstigen Personen- und Güterverkehr zu ermöglichen. Die Fachgruppe für Regelungstechnik und Fahrzeugmechatronik unter der Leitung von Frau Prof. Liu-Henke erforscht als Mitglied des niedersächsischen Zukunftslabors [2] Lösungen durch Nutzung digitaler Technologien und baut zu diesem Zweck ein cyber-physisches-Labortestfeld für die Entwicklung intelligenter und vernetzter Mobilitätsfunktionen auf [3]. Ziel dieses CPS-Labortestfelds ist es, ein zu testendes autonomen System wiederholt in Echtzeit unter Laborbedingungen zu verifizieren. Derzeit unterstützt das Testfeld die Entwicklung und Erprobung innovativer Funktion in zwei Anwendungsfällen: dem autonomen Fahren im intelligenten Straßenverkehr und der fahrerlosen Intralogistik in einer Industrie-4.0 Umgebung. Dazu steht eine autonome Mobilplattform (AGV) zur Verfügung, die sich durch modulare und rekonfigurierbare Bauweise auf den jeweiligen Anwendungsfall anpassen lässt.

Damit die autonome Mobilplattform (AGV) innerhalb der dynamischen Umgebung des Labortestfeldes navigieren kann, ist es nötig eine lokale dynamische Karte mithilfe fahrzeugeigener Sensorik aufzubauen, diese auf ein globales Koordinatensystem zu referenzieren und sich selbst in dieser Karte zu lokalisieren. Die Positionsbestimmung ist nach Cox [4] dabei eines der grundlegendsten Probleme, welches hier detailliert beschrieben werden soll.

Dazu ist dieser Artikel wie folgt gegliedert: In Abschnitt 1 wird zunächst die Methodik vorgestellt, welche bei der Entwicklung der Kartenerstellungsfunktion zur Anwendung kam. Im Kapitel 2 wird ein Überblick über den aktuellen Stand der Technik auf diesem Gebiet gegeben. Abschnitt 3 erläutert die Konzeption der Funktionen. Entwurf und Ausarbeitung dieser Funktion erfolgt in Abschnitt 4 basierend auf der vorherigen Konzeptionen. Abschnitt 5 beschreibt die Bewertung der Kartenerstellung. Zum Schluss wird die Arbeit zusammengefasst und ein Ausblick gegeben.

1 Methodik

Die Entwicklung des Kartenerstellungsalgorithmus ist ein iterativer Prozess, welcher auf der Methodik der mechatronischen Komposition beruht. Die Entwicklung solcher Funktionen erfolgt auf der Grundlage von Anforderungen und Modellen. Der erste Schritt besteht daher darin, die Anforderungen zu definieren. Danach folgt die Abbildung der realen Funktion mit Hilfe mathematischer Gleichungen, auch Modellbildung genannt. Anschließend wird die Funktion konzeptioniert und realisiert. Abschließend wird die entwickelte Funktion validiert werden, um festzustellen, ob sie die Anforderungen erfüllt.

Um den iterativen Prozess darzustellen, wird in die-

ser Arbeit die Funktionsentwicklung nach dem mechatronischen Kreislauf [5] durchgeführt. Abb. 1 stellt den mechatronischen Entwicklungskreislauf der Entwicklung der Kartenerstellungsfunktion dar.

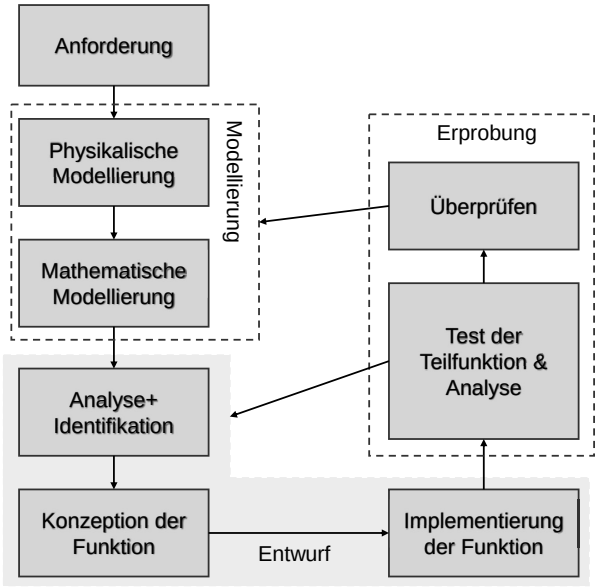

Abbildung 1: Mechatronischer Entwicklungskreislauf

Der Entwurfsprozess beginnt mit der Modellbildung basierend auf dem realen System, welches gemäß der Anforderungen reduziert bzw. vereinfacht wird, sodass sich zunächst ein physikalisches Modell ergibt. Dieses wird mithilfe physikalischer Gesetzmäßigkeiten in ein mathematisches Modell überführt, welches wiederum bspw. in Form von Signalflussplänen im Rechner abgebildet und mithilfe von CAE-Werkzeugen und entsprechender Numerik simuliert werden kann.

Der Modellbildungsprozess umfasst zudem Messungen am realen System, um zum einen die Parameter des mathematischen Modells zu identifizieren und zum anderen die Simulation zu validieren. Eine anschließende Analyse der Simulationsergebnisse lässt Schlussfolgerungen über die grundlegenden statischen und dynamischen Eigenschaften des realen Systems zu, auf dessen Grundlage die Konzeption der Funktionen erfolgt. Es werden sowohl die Funktionsarchitektur als auch Ansätze zu dessen Auslegung und Optimierung festgelegt. Komplexe Funktionen werden zudem nach dem verallgemeinerten Kaskadenprinzip in hierarchische Teilfunktionen zerlegt, um die Funktionskomplexität zu reduzieren und so den Auslegeprozess handhabbar zu machen. Dazu werden bereits in einem frühen Entwicklungsstadium Hard- und Softwareanforderungen berücksichtigt und Schnittstellen für die funktionsübergreifende Kommunikation definiert.

Der Erprobungsprozess wird parallel zur Entwicklung durchgeführt. Wenn eine Teilfunktion entwickelt worden ist, wird diese getestet und analysiert.

2 Stand des Wissens

2.1 Robot Operating System (ROS)

Um die Funktionen der Mobilplattform mit verschiedener Hardware schnell zu entwickeln, verwendet diese Arbeit das Robot Operating System (ROS) als Entwicklungsplattform. Abb.2 zeigt ein Beispiel für die Datenübertragung zwischen verschiedenen Geräten in einer ROS-Umgebung. Die Kommunikation erfolgt über das Bereitstellen (publish) von Nachrichten durch Knoten (Nodes) welche wiederum von anderen Knoten abonniert (subscribe) werden können. Aus Abb.2 ist ebenfalls ersichtlich, dass in einem Gerät mehr als ein Knoten gesetzt werden kann. weiterhin ist ein ROS-Master für die Kommunikation erforderlich, welcher die Verwaltung und Registrierung von Knoten und deren bereitgestellte Nachrichten verantwortet. Die Nachrichten werden zudem in drei Kategorien unterschieden, Thema-, Aktion- und Service-Nachrichten, um deren Priorisierung und Zyklidizität zu deklarieren.

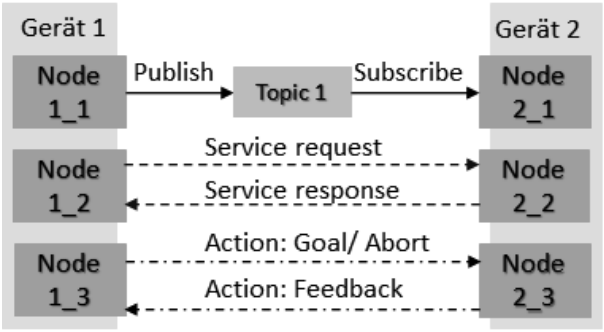

Abbildung 2: Architektur der Kommunikation im ROS

2.2 Darstellung der Karte

Die Erstellung von Umgebungsmodellen ist für die Entwicklung autonomer / hochautomatisierter mobiler Robotersysteme unerlässlich. Das Erstellen des Umgebungsmodells erfolgt mittels Algorithmen zur simultanen Lokalisierung und Kartierung (eng. Simultaneous

Localization and Mapping (SLAM)) welche im nächsten Abschnitt näher vorgestellt werden. Zur Darstellung und Speicherung dieses Umgebungsmodell eignen sich Topologische und geometrische Karten.

Topologische Karte: Bei der topologischen Karte wird die Umgebung in Form eines topologischen Graphen dargestellt. Die Topologie vermeidet die direkte Messung von Umgebungsdaten, erfordert keine weiteren Kartendetails und konzentriert sich mehr auf die Beziehung zwischen Kartenelementen. erstmalige verwendet für die freie Navigation findet sich im Jahr 1988 bei Kuiper und Byun [6].

Geometrische Karte: Eine geometrische Karte ist ein Kartenmodell, das die Umgebung genau darstellen kann. In [7] wurde z.B. eine kontinuierliche Karte, welche Merkmale in der Umgebung durch Punkte und Linien darstellt, zur freien Navigation verwendet. Eine weitere Art der geometrischen Karte ist die Gitterkarte. In dieser wird die kontinuierliche Umgebung in mehrere diskreten Gitter zerlegt wie dies z.B. in [8]gezeigt wurde.

2.3 Etablierte Simultaneous Localization and Mapping (SLAM) Verfahren

Die derzeit am häufigsten genutzten SLAM-Verfahren wurden in verschiedenen Studien [9, 10, 11, 12] vorgestellt und bewertet. Dabei wurden im wesentlichen die vier SLAM-Algorithmen GMapping, Cartographer, Karto-SLAM und HectorSLAM miteinander verglichen. Aus den Bewertungen ist zu schlussfolgern, dass Cartographer in kleinen und großen Szenarien und in jedem Bewegungszustand des Roboters die genaueste und realitätsgetreueste Karte erstellt und zudem auch sehr robust gegenüber Störungen ist. Allerdings haben die verwendeten Sensoren und die Größe der Umgebung Auswirkungen auf die Kartierungsergebnisse dieses Algorithmus. Deswegen kann in dieser Arbeit der Cartographer SLAM-Algorithmus nur durch einsatzspezifische Optimierung der Parameter während der Verwendung angepasst werden, um die Umgebung genau kartieren zu können.

3 Konzept der Funktion

Das Ziel dieser Arbeit ist es, eine autonome mobile Plattform in die Lage zu versetzen, eine zuverlässige und genaue Karte in einer unbekannten, unstrukturierten Umgebung zu erstellen.

Ein Problem des derzeitigen Ansatzes besteht darin, dass beim händischen Mapping nur feste und konstante Objekte in der Umgebung sowie die Länge und Breite des Raums abgebildet werden können. Eine genauere Beschreibung des Raumes ist mit der händischen Kartierung nicht möglich. Das Problem führt daher zu der grundlegenden Anforderung, mobile Plattformen in die Lage zu versetzen, eine detaillierte und genaue Karte in einer unstrukturierten und flexiblen Umgebung zu erstellen. Um dieses Ziel zu erreichen, wird eine funktionale Architektur, wie in der Abbildung 3 dargestellt, vorgeschlagen.

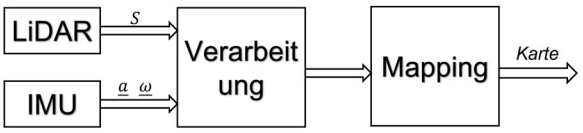

Abbildung 3: Konzeption der Funktion

Für die Kartenerstellung ist es notwendig, entsprechende Sensoren zu nutzen, die die Umgebung zuverlässig abtasten und Rückschlüsse auf Relativbewegungen des Fahrzeugs zulassen. Hier werden also nicht nur Umgebungsdaten, sondern auch die Bewegungsdaten des eigenen Fahrzeugs benötigt. Die Umgebungsdaten werden dabei von LiDAR-Sensoren der Plattform in Form eines Scans erfasst. Ein Scan S ist eine Sammlung, die aus einer endlichen Anzahl K von Laserpunkten s_i besteht und durch die Formel 1 ausgedrückt wird.

$$S = \left\{ \underline{s_i} \mid i \leq K, i \in \mathbb{N} \right\} \tag{1}$$

Bewegungsdaten werden von einer IMU des Fahrzeuges gesammelt. Sie misst die Beschleunigung $\underline{a}$ in den Achsen x, y, z und die Winkelgeschwindigkeit $\underline{\omega}$ um die Achsen x, y und z. Die gesammelten Messdaten müssen zu einem Zeitpunkt einmal transformiert werden, um sie in einem einheitlichen Koordinatensystem miteinander fusionieren zu können. Nur so ist es möglich nicht nur eine Karte aufzubauen, sondern gleichzeitig auch die Position der Plattform innerhalb dieser zu ermitteln (SLAM).

4 Entwurf der Kartenerstellung

Die Kartenerstellung basiert wie bereits erwähnt auf dem Cartographer Algorithmus [13]. Abb. 4 zeigt den Ablauf der Kartenerstellung. Diese Funktion enthält die folgenden Schritte: Filterung, Transformation,

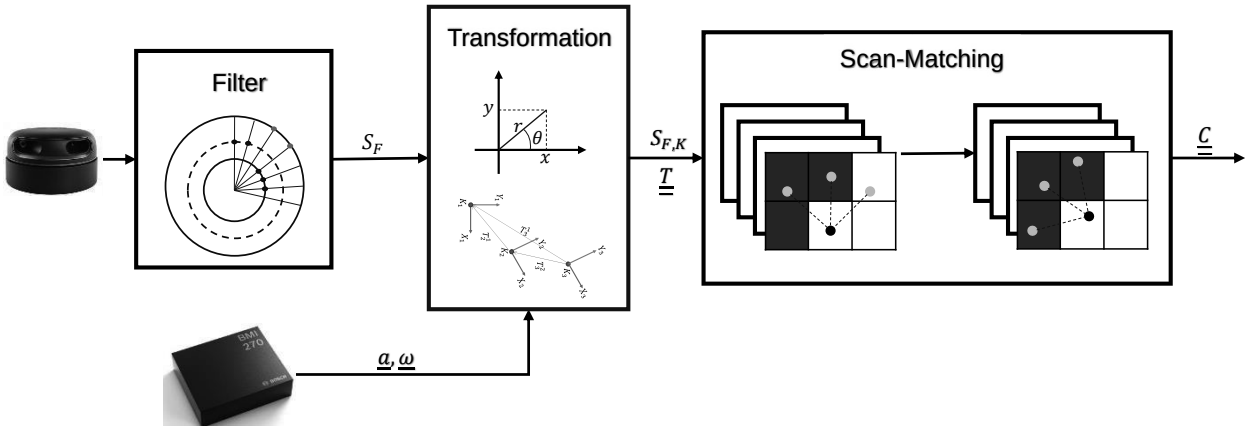

Abbildung 4: Entwurf der Kartierung

Scan-Matching und Kartenaktualisierung. In den folgenden Abschnitten werden diese einzelnen Bestandteile detailliert erläutert. Weiterhin werden die verwendeten Sensoren und die Datenstruktur der Karte vorgestellt.

4.1 Verwendete Sensorik

In diesem Abschnitt werden zunächst die verwendeten Sensoren der Plattform vorgestellt. Der in dieser Arbeit verwendete LiDAR-Sensor ist ein kostengünstiger RPLiDAR A2, der mittels rotierendem Einzellaserkopf die Scandaten S als 2-dimensionale Koordinatenpunktemenge liefert. Rohdaten werden in Polarkoordinaten dargestellt. Jeder Laserpunkt $\underline{s}_i$ ist definiert über einen Entfernungswert d_i und einen Winkel θ_i, durch die Formel $\underline{s}_i = (d_i, \theta_i)$.

Das in dieser Arbeit verwendete IMU ist ein BMI055 Chip innerhalb einer Intel Realsense D435i, ein 6-Achsen IMU, welches einen Beschleunigungsmesser und ein Gyroskop mit jeweils drei zueinander orthogonalen Achsen umfasst.

4.2 Filterung der Sensordaten

Die Qualität der Kartenerstellung ist abhängig von der Qualität der Sensordaten. Deswegen ist es notwendig, dass die Sensordaten vor der Kartenerstellung gefiltert werden. Beispielsweise führt die Montage LiDAR-Scanners auf dem Roboterchassis dazu, dass dieser den Roboterkörper selbst mitscannt. Ebenfalls möglich ist, dass ein großer Messwert stark rauscht, was ebenfalls eine Filterung notwendig macht.

Zur Filterung der LiDAR-Daten wird eine Pass-Through-Filterung durchgeführt, um weiter entfernte oder nahe Messwert zu filtern. In dieser Filterung werden zwei Schwellwerte eingestellt, nämlich „d_{min}“ und „d_{max}“. Wenn der gemessene Wert Formel 2 erfüllt, kann dieser Laserpunkt zur Kartenerstellung verwendet werden. Der gefilterte Laserpunkt wird als $\underline{s}_{F,K_k}$ gezeichnet. Ebenso enthält jeder gefilterte Laserpunkt $\underline{s}_{F,K_k}$ auch entsprechende Entfernungsinformationen d_k und Winkelinformationen θ_k. Mithilfe dieser Filterung können alle Laserpunkte, die sich negativ auf der Kartenerstellung auswirken würden, zuvor gelöscht werden.

$$\{d_i \in \mathbb{R} \mid d_{\min} < d_i < d_{\max}\} \tag{2}$$

4.3 Definition des Koordinatensystem

Die Definition des Koordinatensystem ist der Schüssel zu einer genauen Beschreibung der Position des Fahrzeugs. Abb.5 zeigt zunächst alle benötigten Koordinatensysteme. In dieser Arbeit werden rechtshändige Koordinatensysteme genutzt; wenn die jeweilige X-Achse nach vorne zeigt, zeigt also die jeweilige Y-Achse nach rechts.

- AGV-Koordinatensystem BCS_{AGV}: Dieses Koordinatensystem ist ein Köperkoordinatensystem. Es befindet sich in der geometrischen Mitte des AGVs. Die positive Richtung der X-Achse des AGV-Koordinatensystems wird entlang der Vorwärtsrichtung des Fahrzeugs definiert.
- LiDAR-Koordinatensystem BCS_{LiDAR}: Die Position des Koordinatensystem entspricht der realen

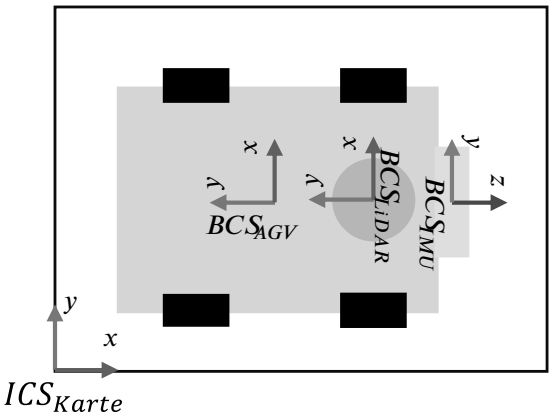

Abbildung 5: Definition der Koordinatensysteme

Position des Sensors im AGV. Hier zeigt die positive X-Achse zur rechten Seite des AGVs.

- IMU-Koordinatensystem BCS_{IMU}: Das Köperkoordinatensystem des IMU - hier zeigt die Z-Achse Richtung Front des Fahrzeugs.
- Kartenkoordinatensystem ICS_{Karte}: Dieses Koordinatensystem wird als globales Koordinatensystem definiert, welches während der Kartenerstellung fest ist. Dieses Koordinatensystem wird im ersten Rechenschritt an der Position des AGV-Koordinatensystems initialisiert.

Die Koordinatensysteme der beiden Sensoren BCS_{LiDAR} und BCS_{IMU} werden entsprechend der realen Position im Fahrzeug definiert (siehe auch Abb.5). Zur Erstellung der Karte müssen zunächst alle erfassten Sensordaten in kartesischen Koordinaten vorliegen. Für die LiDAR- Scandaten ist dazu die in Formel 3 beschriebene Umwandlung der Polarkoordinaten in das kartesische Koordinatensystem nötig.

$$\begin{aligned} x_k &= d_k \cos\theta_k \\ y_k &= d_k \sin\theta_k \end{aligned} \tag{3}$$

Nach der Umwandlung werden die Scandaten wie in Formel 4 dargestellt, wobei $\underline{s}_{FK_k}$ ein umgewandelter Laserpunkt ist.

$$S_{F,K} = \left\{ \underline{s}_{F,K_k} \mid j \leq M, i \in \mathbb{N} \right\} \tag{4}$$

Anschließend werden die umgerechnete Daten aus den jeweiligen lokalen Koordinatensystemen $BCS_{Sensor} = \{BCS_{LiDAR}, BCS_{IMU}\}$ in das globale Koordinatensystem ICS_{Karte} transformiert. Im weiteren Verlauf wird jeder Laserpunkt $\underline{s}_{F,K_k}$ für die Kartierung mehrere Koordinatentransformationen durchlaufen, deren Ablauf durch die Formel 5 und Formel 6 definiert ist.

$$\underline{s}_{F,K_k}^{BCS_{AGV}} = \underline{\underline{T}}_{BCS_{LiDAR}}^{BCS_{AGV}} \cdot \underline{s}_{F,K_k} \tag{5}$$

$$\underline{s}_{F,K_k}^{ICS_{Karte}} = \underline{\underline{T}}_{BCS_{AGV}}^{ICS_{Karte}} \cdot \underline{s}_{F,K_k}^{BCS_{AGV}} \tag{6}$$

Wobei $\underline{\underline{T}}_{BCS_{\text{LiDAR}}}^{BCS_{\text{Fahrzeug}}}$ und $\underline{\underline{T}}_{BCS_{\text{Fahrzeug}}}^{ICS_{\text{Karte}}}$ zwei Transformationsmatrizen sind. Eine allgemeine Form der Transformationsmatrix besteht aus zwei Teilen, der Drehmatrix $\underline{\underline{R}}$ und der Translationsmatrix $\underline{t}$, wie durch die Formel 7 dargestellt.

$$\underline{\underline{T}} = \begin{pmatrix} \underline{\underline{R}} & \underline{t} \\ 0 & 1 \end{pmatrix} \tag{7}$$

Die Transformationsmatrix $\underline{\underline{T}}_{BCS_{Fahrzeug}}^{ICS_{Karte}}$ hängt von der Pose des Fahrzeug $\underline{\zeta}$ ab. Die Pose enthält die Position des Fahrzeugs $\underline{t_f}$ und die Ausrichtung θ_f, die durch IMU (Formel 8) geschätzt wird. Die Pose θ_f wird durch die Formel 9 in eine Transformationsmatrix $\underline{\underline{T}}_{BCS_{Fahrzeug}}^{ICS_{Karte}}$ umgewandelt.

$$\underline{\zeta} = f(\underline{a}, \underline{\omega}, \underline{\underline{T}}_{BCS_{IMU}}^{BCS_{AGV}}) \tag{8}$$

$$\underline{\zeta} = \begin{bmatrix} \underline{\underline{R}}(\theta_f) & \underline{t_f} \\ 0 & 1 \end{bmatrix} \tag{9}$$

4.4 Datenstruktur des verwendeten Kartentypus

In dieser Arbeit wird die Gitterkarte als Untertypus der geometrischen Karten zur Darstellung und Speicherung des Umgebungsmodells verwendet. Im Gitterkartentypus wird wiederum die Belegungsgitterkarte ausgewählt. Wie oben erwähnt, wird in dieser Karte der Zustand jedes Gitters durch Wahrscheinlichkeitswert bestimmt. Die Gitterkarte und die Elemente der Gitterkarte werden als eine Matrix $\underline{\underline{C}}$ mit den Matrizenelementen c_{ij} definiert (Formel 10):

$$\underline{\underline{\mathbf{C}}} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \cdots & \cdots & & \cdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix} \tag{10}$$

Jedes Gitter hat drei Zustände, nämlich frei, belegt und unbekannt. Diese drei Zustände können jeweils

durch die Nummern 0, 1 und -1 dargestellt werden. Wegen des Rauschens der Messung kann ein Laserpunkt nicht immer mit einem bestimmten Gitterpunkt zusammenfallen. Daher wird der Zustand jedes Gitters durch eine Wahrscheinlichkeit bestimmt. Wenn der Laserpunkt einmal in einem Gitter liegt, wird der Wahrscheinlichkeitswert, dass das Gitter als belegt eingestuft werden kann, erhöht. Der Aktualisierungsprozess wird in dem nächsten Abschnitt beschrieben.

4.5 Scan-Matching

Das Ziel des Scan-Matching ist eine optimale Plattform-Pose $\underline{\zeta}_f$ zu finden, und so den Positionierfehler zwischen einem Scan-Laserpunkt und dem entsprechenden Kartegitter zu minimieren. In diesem Fall wird dieser Prozess als ein nicht-lineares kleinste-Quadrate Problem (Formel 11) bezeichnet.

$$\min e = \sum_{k=1}^{M} \left(\underline{\underline{C}}_k - \underline{\underline{T}}_f \cdot \underline{s}_k \right)^2 \tag{11}$$

Wobei $\underline{s_k}$ ein Laserpunkt des Scans im AGV-Koordinatensystem (BCS_{AGV}) , d.h. $\underline{s}_{F,K_k}^{BCS_{AGV}}$ ist. Der Index des Laserpunktes in einem Scan ist k. Die Anzahl der Laserpunkte in einem Scan wird hier mit M bezeichnet. Jeder Laserpunkt wird vom AGV-Koordinatensystem in das Karten-Koordinatensystem(ICS_{Karte}) transformiert. $\underline{\underline{C}}_k$ ist die Koordinate der korrespondierenden Karte. $\underline{\underline{T}}_f$ ist eine vereinfachte Notation für $\underline{\underline{T}}_{BCS_{Fahrzeug}}^{ICS_{Karte}}$. Der Zweck der Gleichung ist es, eine optimale Transformationsmatrix $\underline{\underline{T}}_f$ zu finden, welche die Abweichung zwischen Laserpunkt und dem korrespondierenden Gitter minimiert. Der Anfangswert der Matrix $\underline{\underline{T}}_f$ wird durch die IMU entsprechend der Formeln 8 und 9 geschätzt.

4.6 Kartenaktualisierung

Jedes Gitter der Belegungskarte hat einen initialen Wahrscheinlichkeitswert für den Belegungszustand $P(c_{ij} = 1)$. Der Zustand des gesamten Gitters kann mit der Formel 12 beschrieben werden. Wenn eine neue Beobachtung eingeht, kann die Aktualisierung der Karte gemäß der Bayes'schen Formel erfolgen, wobei z die Beobachtung selbst ist. Um die Berechnung zu vereinfachen, kann die Formel 13 Logarithmiert werden.

$$\text{odds}\,(c_{ij}) = \frac{P(c_{ij} = 1)}{P(c_{ij} = 0)} \tag{12}$$

$$\text{odds}\,(c_{ij} \mid z) = \frac{P(z \mid c_{ij} = 1)}{P(z \mid c_{ij} = 0)} \,\text{odds}\, s\,(c_{ij}) \tag{13}$$

Abb. 6 zeigt des Aktualisierungsprozess der Karte. Zum Zeitpunkt $t = 0$ wird jedes Gitter der Karte mit null Log-odds-Werten initialisiert. Diese Initialisierung ist identisch mit der Wahrscheinlichkeit, dass die Zellen frei sind. Zum Zeitpunkt $t = 1$ werden neue Messungen von dem LiDAR-Sensor empfangen, der in diesem Beispiel viele einzelne Strahlen aussendet. Auf dieser Abbildung ist deutlich zu sehen, dass die gelben Zellen als besetzt und die hellgrünen Zellen als freier, leerer Raum gemessen werden. Der Wert von $odds_{belegt}$ wird als $0,8$ und der Wert von $odds_{frei}$ als $-0,5$ angenommen.

Anschließend werden die Werte der beobachteten Gitter aktualisiert. Die Werte der unbeobachteten Gitter bleiben unverändert. Wenn der Log-odds-Wert einen Schwellenwert überschreitet, wird das Gitter als Belegt gekennzeichnet. Wenn der Log-odds-Wert unter einem Schwellenwert liegt, wird das Gitter als frei betrachtet.

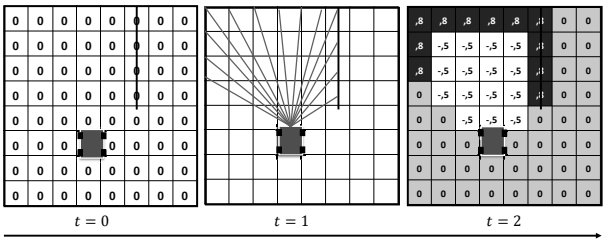

Abbildung 6: Aktualisierung der Karte

5 Realisierung der Kartenerstellung

Dieses Kapitel beschreibt die Realisierung der Kartenerstellung auf der Mobilplattform innerhalb der ROS-Umgebung.

5.1 Einrichtung und Konfiguration in ROS

Vor der Kartierung muss zunächst der räumliche Zustand des Fahrzeugs in ROS beschrieben werden, wobei ROS eine eigene Koordinatensystemdefinition

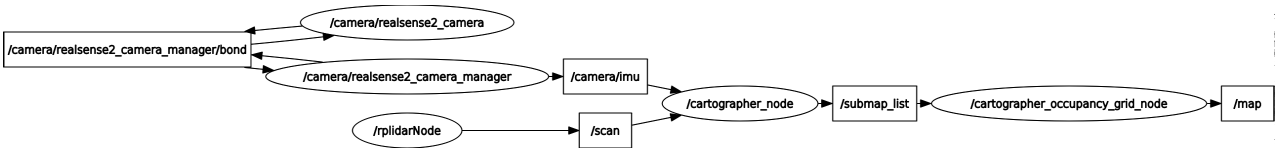

Abbildung 7: Knotendiagramm

nutzt. Daher müssen die in diesem Artikel in Entwurf definierten Koordinatensysteme gemäß den ROS-Koordinatensystemregeln REP105 und REP103 angepasst werden. Danach können die Beziehungen zwischen den verschiedenen Sensorkoordinatensystemen auf der Plattform mithilfe derer CAD-Modelle bestimmt und in einer URDF-Datei, einem Werkzeug zur Beschreibung des Objektmodells, eingetragen werden. Nach der Beschreibung der Beziehungen jedes Koordinatensystem kann das Modell der Plattform im ROS wie in Abbildung 8 dargestellt werden.

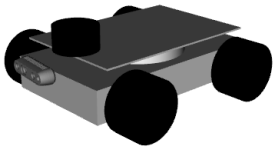

Abbildung 8: Fahrzeug-Modell im ROS

Um die Kartenerstellung in ROS umzusetzen, werden zunächst alle benötigten Knoten konfiguriert. Abb. 7 zeigt das entsprechende Knotendiagramm. Dadurch können die erforderlichen Informationen für die Kartenerstellung korrekt zwischen den Knoten übertragen werden.

5.2 Ergebnis

Abb. 9 zeigt die erstellte Karte. Die grauen Linien im Diagramm sind die Hilfslinien, die zum Aufteilen der Bereiche verwendet werden, um eine Fabrik zu simulieren. Um die Genauigkeit der erstellten Karten zu überprüfen, muss eine Karte durch externe Messungen zur Verifikation erstellen und mit der durch den Algorithmus erstellten Karte vergleichen werden. Die geometrische Daten des Raums werden durch einen Laserentfernungsmesser gemessen. Dann wird die Verifikationskarte mit der erstellte Karte ausgerichtet und aufeinander gelegt, wie die Abb. 10 gezeigt. In X-Richtung ist der Messfehler 3 cm und in Y-Richtung 2 cm groß. Die Fehler in beide Richtungen sind kleiner als die Auflösung der Karte. Deswegen kann diese erstellte Karte für weitere Arbeiten als Basis verwendet werden.

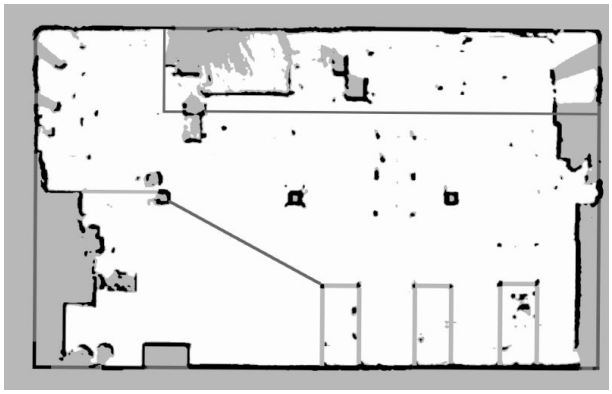

Abbildung 9: Die erstellte Karte

Abbildung 10: Vergleich der Karten

6 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde eine Funktion für eine autonome Mobilplattform zur Kartenerstellung realisiert. Der große Vorteil der Funktion ist, dass die Mobilplattform in einer unstrukturierten Umgebung automatisch und schnell eine zuverlässige Karte erstellen kann. Gleichzeitig kann die Plattform auch selbst die eigene

Position bestimmen. In dieser Arbeit werden die benötigen Koordinatensysteme und der Informationsfluss definiert, der für die Erstellung der Karte erforderlich ist. Deshalb ist dieser auch übergreifend für verschiedene Mobilplattformen und verschiedene Sensoren zu verwenden. Zum Schluss wird die erstellte Karte verifiziert und gezeigt, dass die durch den Algorithmus erstellte Karte als eine Referenzkarte verwendet werden kann.

Allerdings enthielt diese Arbeit auch Teile, die verbessert werden können. Der Zustand des Fahrzeugs könnte durch mehr Sensoren besser geschätzt werden, umso die Eigenlokalisierung zu verbessern. Außerdem kann die Auflösung der Karte aufgrund der Leistungsfähigkeit des Computers nicht erhöht werden, was jedoch in Zukunft optimiert werden kann. Sobald die Karte erstellt ist, wird sie als statische Referenzkarte für nachfolgenden Arbeiten wie die Lokalisierung und Navigation verwendet.

Danksagung

Gefördert vom Niedersächsischen Ministerium für Wissenschaft und Kultur unter Fördernummer ZN3495 im Niedersächsischen Vorab der VolkswagenStiftung und betreut vom Zentrum für digitale Innovationen (ZDIN).

Literatur

[1] Bundesministerium für Bildung und Forschung (BMBF). *Die neue Hightech-Strategie - Innovationen für Deutschland*. Berlin. 2014.

[2] Zentrum für digitale Innovationen Niedersachsen (ZDIN). *Digital klar voraus. Branchenübergreifende Digitalisierung und Weiterentwicklung am ZDIN*. Oldenburg: OFFIS e. V. 2022.

[3] Liu-Henke X, Jacobitz S, Göllner M, Zhang J, Scherler S, Yarom OA. Cyber-physical Industry 4.0 laboratory test field to simulate self-optimizing intralogistics. In: *2020 19th International Conference on Mechatronics-Mechatronika (ME)*. IEEE. 2020; pp. 1–6.

[4] Cox IJ. Blanche-an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on robotics and automation*. 1991; 7(2):193–204.

[5] Liu-Henke X. *Mechatronische entwicklung der aktiven feder-/neigetechnik für das schienen-fahrzeug RailCab*. VDI Verlag. 2005. 149 p.

[6] Kuipers B, Byun YT. A Robust, Qualitative Method for Robot Spatial Learning. In: *AAAI*, vol. 88. 1988; pp. 774–779.

[7] Tomatis N, Nourbakhsh I, Siegwart R. Simultaneous localization and map building: A global topological model with local metric maps. In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, vol. 1. IEEE. 2001; pp. 421–426.

[8] Li Y, Ruichek Y. Occupancy grid mapping in urban environments from a moving on-board stereo-vision system. *Sensors*. 2014;14(6):10454–10478.

[9] Fan X, Wang Y, Zhang Z. An evaluation of Lidar-based 2D SLAM techniques with an exploration mode. In: *Journal of Physics: Conference Series*, vol. 1905. IOP Publishing. 2021; .

[10] Filipenko M, Afanasyev I. Comparison of various slam systems for mobile robot in an indoor environment. In: *2018 International Conference on Intelligent Systems (IS)*. IEEE. 2018; pp. 400–407.

[11] Santos JM, Portugal D, Rocha RP. An evaluation of 2D SLAM techniques available in robot operating system. In: *2013 IEEE international symposium on safety, security, and rescue robotics (SSRR)*. IEEE. 2013; pp. 1–6.

[12] Yagfarov R, Ivanou M, Afanasyev I. Map comparison of lidar-based 2d slam algorithms using precise ground truth. In: *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE. 2018; pp. 1979–1983.

[13] Hess W, Kohler D, Rapp H, Andor D. Real-time loop closure in 2D LIDAR SLAM. In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2016; pp. 1271–1278.