UNIVERSITÄT ZU LÜBECK
INSTITUTE FOR ROBOTICS
AND COGNITIVE SYSTEMS

Komplett abdeckende Pfadplanung für kostengünstige Roboter

Complete coverage path planning for low cost robots

Bachelorarbeit

Im Rahmen des Studiengangs
Robotik und Autonome Systeme
der Universität zu Lübeck

Vorgelegt von
Robin Denz

Ausgestellt und bewertet von
Prof. Dr. Elmar Rückert

Mit Unterstützung von
M.Sc. Nils Rottmann

Lübeck, den 11. November 2019

Ich, Robin Denz (Robotik und Autonome Systeme an der Universität zu Lübeck, Matrikelnummer 665487), versichere an Eides statt, die vorliegende Bachelorarbeit

Komplett abdeckende Pfadplanung für kostengünstige Roboter

selbständig und nur unter Benutzung der angegebenen Quellen und Hilfsmittel angefertigt zu haben. Die Arbeit wurde in dieser oder ähnlicher Form noch keiner Prüfungskommission vorgelegt.

Robin Denz 11. November 2019

Zusammenfassung

Die Nachfrage in der Bevölkerung nach Haushaltsrobotern steigt immer weiter an. Dazu gehören besonders solche mobilen Saug- und Rasenmähroboter. Diese sind dabei meist sehr teuer und trotzdem auch noch sehr uneffizient. Gerade für Rasenmähroboter ist es zum richtigen Erfüllen ihrer Aufgabe essentiell den kompletten Arbeitsraum besucht zu haben. Den aktuellen Stand der Technik stellen dabei allerdings immernoch Randomwalk-Algorithmen dar, die sehr unzuverlässig und uneffizient arbeiten. Die vorliegende Bachelorarbeit stellt daher eine Methode zur intelligenten Pfadplanung für mobile "low cost"Roboter anhand eines Rasenmähroboters dar. Dieser ist lediglich mit binärer Sensorik ausgestattet um seine Position in seinem durch hohe Unsicherheiten behafteten Arbeitsraum wahrzunehmen. Durch eine intelligente Darstellung des bereits besuchten Arbeitsraumes sowie der durch neuronale Netze inspirierten Pfadplanung, schafft es der Rasenmähroboter eine ausschlaggebende Verbesserung der Effizienz gegenüber dem Randomwalk zu erlangen.

Abstract

The demand among the population for household robots continues to rise. These include in particular mobile cleaning and lawn mowing robots. These are usually very expensive and still very inefficient. Especially for lawn mowing robots, it is essential to have visited the entire working space in order to perform their task correctly. However, the current state of the art is still random walk algorithms, which are very unreliable and inefficient. The present bachelor thesis therefore presents a method for intelligent path planning for mobile "low cost"robots using a lawn mower robot. The robot is only equipped with binary sensors to detect its position in its working space, which is fraught with high uncertainties. By an intelligent representation of the already visited working space as well as the path planning inspired by neural networks, the lawn mower robot manages to achieve a decisive improvement in efficiency compared to the random walk.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Durch den immensen Fortschritt in vielen nutzerorientierten Technologiebereichen wie zum Beispiel den Smartphones, steigt in der Bevölkerung auch immer mehr die Erwartung an autonome Haushaltsroboter [1]. Diese sollen die Menschen von regelmäßigen, langwierigen und dementsprechend meist unbeliebten Aufgaben entlasten. Dazu zählen besonders Putzaufgaben oder auch das Rasenmähen [2]. Für beide dieser Anwendungsfälle wird ein sehr ähnlicher Roboter verwendet. Im folgenden wird sich darum auf die Anwendung für einen Rasenmähroboter beschränkt.

Aktuell im Handel erhältliche Rasenmähroboter wie der in Abb. 1.1 fahren fast ausschließlich nach dem Zufallsprinzip, also ohne Strategie. Sie besitzen dabei kein Wissen über ihren Arbeitsraum und erfahren nur über ihre Sensorik wenn sie auf ein Hinderniss oder die Arbeitsraumbegrenzung stoßen. Sie gehen dabei davon aus, dass sie sich immer auf dem Rasen befinden. Sie fahren solange geradeaus, bis sie auf ein Hinderniss oder die Begrenzung ihres Arbeitsraumes stoßen und sich dann für einen zufälligen Zeitraum auf der Stelle drehen bevor sie wieder geradeaus fahren. In manchen Fällen setzt der Rasenmähroboter noch ein kurzes Stück zurück bevor er den Drehvorgang einleitet. Generell wird dieses Verhalten auch als Randomwalk bezeichnet.

Dieses Verhalten bietet zwar eine stochastische Vollständigkeit für die volle Abdeckung des Arbeitsraumes, dies allerdings immer in sehr unterschiedlicher, und hoher Zeit. Zudem werden viele Flächen sehr oft abgefahren, obwohl diese schon abgemäht sind, was natürlich die Effektivität beeinträchtigt.

Nachteile die sich durch dieses Verhalten auf den Nutzer auswirken sind neben einem unregelmäßigen Mähbild (siehe Abb. 1.2) die lange Arbeitszeit des Roboters und die damit verbundene Lärmbelastung, sowie der höhere Energieverbrauch und Verschleiß des Geräts. Einhergehend mit dem meist sehr hoch angesetzten Preis der Produkte und der nicht intuitiven Einrichtung des Systems, sind viele potenzielle Käufer meist noch abgeschreckt, sich

Abbildung 1.1: Rasenmähroboter von Worx (Quelle: www.worx.com)

einen Rasenmähroboter zuzulegen. Ein wichtiger Teil der heutigen Forschung bezieht sich dementsprechend auf die so genannte *low cost* Robotik. Diese versucht, wie der Name schon andeutet, Systeme mit günstiger Sensorik und einer ausreichenden und zuverlässigen Genauuigkeit zu entwickeln.

Ein großer Bestandteil dieser Forschung liegt darin, einen mit dieser Sensorik harmonierenden Pfadplanungsalgorithmus zu entwickeln. Die Anforderungen an diesen bestehen darin, den kompletten Arbeitsraum abzufahren ohne dabei die selben Stellen wiederholt zu besuchen. Solche Algorithmen wurden zum Beispiel in [3], [4], [5] und [6] mit verschiedenen Herangehensweisen und für unterschiedliche Einsatzgebiete vorgestellt. In [3] wird dies durch einen Gradientenabfall über den gesamten Arbeitsraum zu der Zielposition realisiert. [4] stellt einen Algorithmus für große Arbeitsräume anhand von Landwirtschaftsmaschinen vor. [5] hingegen unterteilt den Arbeitsraum in kleinere Teilräume und steuert den Roboter anhand einer Kostenfunktion durch diese. In [6] wird ein solcher Ansatz durch eine globale Pfadplanung erweitert. Die größte Schwierigkeit bei der Umsetzung dieser Algorithmen besteht allerdings in der Lokalisierung des autonomen Roboters. Die Realisierungen dafür beginnen bei verschiedenen rechenintensiven, für Systeme mit ausgeprägter Sensorik entwickelten SLAM-Algorithmen (Simultanous localization and mapping), die im Grundsatz in [7] vorgestellt und erläutert werden. Eine Abwandlung davon stellt zum Beispiel [8] als faktorisierte SLAM Variante dar. Im Gegensatz dazu benötigen Methoden wie der Partikelfilter [9], der in diesem Paper beispielhaft vorgestellt wird, oder Kalmanfilter [10], der durch Sensorfusion arbeitet, keine stark ausgeprägte Sensorik.

Abbildung 1.2: unregelmäßiges Mähbild durch Randomwalk (Quelle: www.robomaeher.de)

1.1 Aufbau der Arbeit

Die folgende Bachelorarbeit wird in drei Hauptteile gegliedert. In Kapitel 2 wird detailliert auf die den Rasenmähroboter betreffenden Grundlagen eingegangen. Dazu gehören das kinematische und das Odometrie Modell des Systems und desssen genereller Aufbau. Desweiteren wird die verwendete Lokalisierungsmethode anhand des Partikelfilters erläutert.

Kapitel 3 befasst sich mit zwei verschiedenen, auf dem Partikelfilter basierenden, Ansätzen eine Abdeckungskarte für die vom Roboter bereits besuchten Bereiche seines Arbeitsraumes darzustellen. Diese Herangehensweisen werden schließlich miteinander veglichen und es wird sich für das weitere Vorgehen auf eine festgelegt.

In Kapitel 4 wird ein auf der Abdeckungskarte aufbauender Pfadplanungsalgorithmus vorgestellt und erläutert. Dieser wird auf seine Effektivität sowie seine Laufzeit bewertet.

Letztendlich werden die aus dieser Arbeit hervorgehenden Ergebnisse in Kapitel 5 zusammengefasst und ein Ausblick auf weitere, korrespondierende Arbeiten gegeben.

Kapitel 2

Grundlagen

Dieses Kapitel behandelt die notwendigen Grundlagen eines typischen Rasenmähroboters wie er auch am Institut für Robotik und kognitive Systeme der Universität zu Lübeck entwickelt wird. Im ersten Abschnitt wird die Hardware und dessen Dynamik beschrieben. Dafür wird auch auf die Aktorik und Sensorik des Rasenmähers eingegangen und schließlich das damit einhergehende kinematische Modell und das Odometriemodell hergeleitet. Darauffolgend wird der Partikelfilter als Möglichkeit zur Lokalisierung des Rasenmähers vorgestellt.

2.1 Hardware und Dynamik

Der dieser Arbeit zugrunde liegende Rasenmähroboter verfügt über zwei gleiche Räder, die unabhängig voneinander angesteuert werden können. Die Räder liegen auf einer Achse und zur Stabilisierung ist ein zusätzliches, nicht angetriebenes Stützrad im vorderen Bereich angebracht. Das Ursprungskoordinatensystem des Roboters, welches auch den Arbeitspunkt darstellt, liegt zentral auf dieser Achse.

Die Sensorik des Rasenmähers umfasst jeweils einen Schrittzähler pro Reifen welcher die zurückgelegte Strecke eines jeden Reifens misst und für das Odometriemodell zur Verfügung stellt. Dies sind für gewöhnlich Hall-Sensoren, die die Magnetfelder von in den Reifen verbauten Magneten messen, und daran das Mitzählen der Umdrehung der Reifen ermöglichen. Für eine genauere Beschreibung wird auf [11] verwiesen.

Desweiteren verfügt der Rasemäherroboter über zwei Grassensoren, die messen, ob sich der Roboter in seinem Arbeitsraum befindet. Dabei handelt es sich um Chlorophyllsensoren, die anhand eines Spektrometers die zurückgeworfene Wellenlänge eines Lichtimpulses messen und daraus ein binä-

res Messergebnis erzeugen, ob sich der Rasenmäher über dem Rasen oder nicht befindet. Die beiden Sensoren sind an der Unterseite des Roboters angebracht. Ein Prototyp dieser Sensoren wird aktuell im Institut für Robotik und kognitive Systeme der Universität zu Lübeck entwickelt [12]. Der Vorteil dieser Sensoren gegenüber den aktuell in der Industrie verkauften autonomen Rasenmähersystemen liegt einerseits in der Einfachheit, andererseits auch in einem Kostenvorteil. In den heutigen Rasenmähersystemen werden hauptsächlich teure Spulen verwendet, die eine Induktionsspannung erzeugen und so aktiviert werden, sollte sie sich außerhalb des Arbeitsraumes befinden. Erzeugt wird diese Induktionsspannung meist durch einen auf dem Rasen liegenden oder unterm Rasen eingegrabenen Kabel, auch Induktionsschleife genannt, welches dauerhaft unter einer leichten Spannung steht und somit ein den Arbeitsraum begrenzendes Magnetfeld aufbaut. Die Installation dieser Systeme ist meist sehr aufwendig und der Rasenmäher ist Out-of-the-Box nicht einsatzfähig.

Durch die vorgestellten Eigenschaften des in dieser Arbeit verwendeten Rasenmähroboters ist dieser eindeutig in die Kategorie der *low cost* Roboter einzuordnen.

2.1.1 Kinematisches Modell

Aufbauend auf [13] entspricht das kinematische Modell des Rasenmähers dem eines mobilen Roboters mit zwei angetriebenen Rädern. Somit handelt es sich um einen klassischen Differentialantrieb. Die Pose $\boldsymbol{x}$ zum Zeitpunkt t ist definiert als

$$\boldsymbol{x}_t = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} \tag{2.1}$$

mit der Position des Roboters bestehend aus der x- und y-Koordinate auf der 2-dimensionalen Arbeitsebene und der Orientierung θ. Dementsprechend ergibt sich ein Zustandsübergang von

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \Delta\boldsymbol{x_t} \text{ mit } \Delta\boldsymbol{x}_t = \begin{pmatrix} \Delta x_t \\ \Delta y_t \\ \Delta\theta_t \end{pmatrix}. \tag{2.2}$$

Die Zustandsänderung $\Delta\boldsymbol{x}$ errechnet sich dabei aus dem Abstand L von den Rädern des Roboters zu seiner Drehachse, welche hier im Mittelpunkt zwischen den Rädern liegt, und dem Eingang in das System. Der Eingang ist definiert als

$$\boldsymbol{u}_t = \begin{pmatrix} v_t \\ \omega_t \end{pmatrix} \tag{2.3}$$

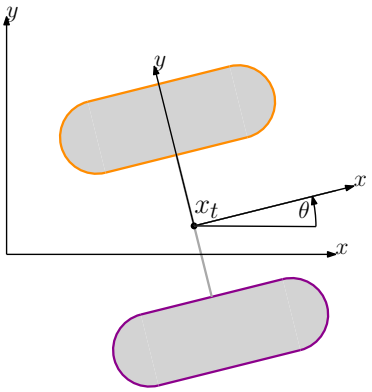


Abbildung 2.1: Darstellung des globalen Koordinatensystems der Welt und des lokalen Koordinatensystems des Roboters

mit der Translationsgeschwindigkeit v und der Winkelgeschwindigkeit ω. Diese entspricht der Geschwindigkeit mit der sich der Roboter um seine senkrecht zur Arbeitsebene liegende Drehachse dreht. Da sich bei dem Differentialantrieb des Roboters die beiden Räder mit unabhängigen und unterschiedlichen Geschwindigkeiten drehen, um eine Änderung des Zustandes $\boldsymbol{x}_t$ hervorzurufen, werden diese auch getrennt voneinander angesteuert. Dazu wird hier weiter differenziert in v_r für die Geschwindigkeit des rechten Rads und v_l für die Geschwindigkeit des linken. Es ergibt sich

$$v_l = \omega L - v \qquad v_r = \omega L + v. \tag{2.4}$$

Aus diesen Gleichungen leiten sich schließlich die Geschwindigkeit v und die Winkelgeschwindigkeit ω des Roboters ab

$$\begin{aligned} \omega L &= v_l + v \\ \omega L &= v_r - v \\ \Rightarrow v &= \frac{v_r - v_l}{2}. \qquad (2.5) \\ \Rightarrow \omega &= \frac{v_r + v_l}{2L}. \qquad (2.6) \end{aligned}$$

Mit Hilfe der grundlegenden geometrischen Eigenschaften des Systems ergibt sich letztendlich die Zustandsänderung $\Delta\boldsymbol{x}$ mit

$$\Delta\boldsymbol{x}_t = \begin{pmatrix} \frac{v}{\omega}(\sin(\theta + \Delta t\omega) - \sin(\theta)) \\ \frac{v}{\omega}(\cos(\theta) - \cos(\theta + \Delta t\omega)) \\ \Delta t\omega \end{pmatrix}. \tag{2.7}$$

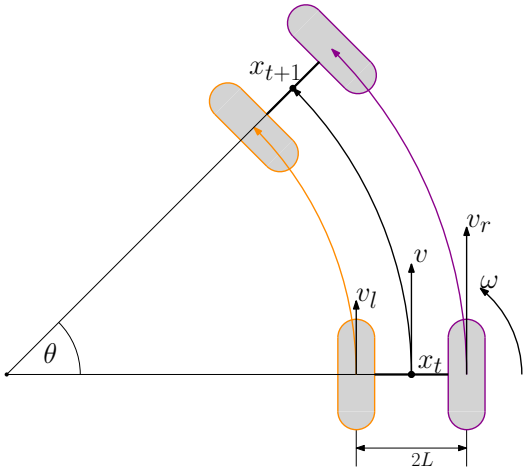


Abbildung 2.2: Kinematisches Modell des Rasenmähers mit Radgeschwindigkeiten v_l und v_r, resultierender Geschwindigkeit v, Winkelgeschwindigkeit ω um Drehachse, Radabstand $2L$ und Zustandsübergang von $\boldsymbol{x}_t$ zu $\boldsymbol{x}_{t+1}$ um Orientierung θ

Dabei ist zu beachten, dass davon ausgegangen wird, dass sowohl die Geschwindigkeit v, als auch die Winkelgeschwindigkeit ω konstant über Δt sind.

Dieses kinematische Modell des Rasenmähers wird benutzt, um nach dem Eingang eines Kontrollsignals $\boldsymbol{u}$, welches sich aus den einzelnen Geschwindigkeitsbefehlen für die beiden Reifen ergibt, eine neue Pose $\boldsymbol{x}$ für den Roboter zu errechnen. Dies ist allerdings noch auf eine perfekte Welt ohne äußere Störeinflüsse oder Messrauschen im System limitiert und somit unrealistisch.

Um Unsicherheiten in das System zu integrieren, wird wiederum auf das in [13] vorgestellte Samplingverfahren zurückgegriffen. Dieses erzeugt eine Zufallsvariable X mit der Wahrscheinlichkeitsdichte der Gaußnormalverteilung um den Erwartungswert $\mu = 0$ und der Varianz σ, welche beschrieben ist als

$$X(\sigma) \sim \mathcal{N}(\mu = 0, \sigma) \quad \text{mit} \quad \mathcal{N}(\mu, \sigma)\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{\mu^2}{\sigma}}. \tag{2.8}$$

Es wird auf Grundlage der [14] davon ausgegangen, dass die Fehler im dynamischen System des Rasenmähroboters Gaußverteilt sind. Im System des Rasenmähers wird nun der Eingang $\boldsymbol{u}_t$ und die Orientierung θ durch diese Zufallsvariable X mit Rauschen behaftet und im folgenden als $\hat{v}$, $\hat{\omega}$ und $\hat{\theta}$

definiert mit

$$\hat{v} = X(\sigma_v) + v$$
$$\hat{\omega} = X(\sigma_\omega) + \omega$$
$$\hat{\theta} = X(\sigma_\theta) + \theta.$$

Die Varianzen σ_v, σ_ω und σ_θ ergeben sich dafür aus den sechs Bewegungs Fehlerprametern $\alpha_1, ..., \alpha_6$

$$\sigma_v = \alpha_1 * |v| + \alpha_2 * |\omega|$$
$$\sigma_\omega = \alpha_3 * |v| + \alpha_4 * |\omega|$$
$$\sigma_\theta = \alpha_5 * |v| + \alpha_6 * |\omega|.$$

Somit ergibt sich letztendlich ein Zustandsübergang von

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{pmatrix} = \begin{pmatrix} x_t + \frac{\hat{v}}{\hat{\omega}}(\sin(\theta_t + \Delta t\hat{\omega}) - \sin(\theta_t)) \\ y_t + \frac{\hat{v}}{\hat{\omega}}(\cos(\theta_t) - \cos(\theta_T + \Delta t\hat{\omega})) \\ \theta_t + \Delta t\hat{\omega} + \Delta t\hat{\theta} \end{pmatrix}. \tag{2.9}$$

Nun bleibt nur noch der Fall einer Singularität für $\hat{\omega} = 0$, bei der sich der Roboter nicht um die eigene Achse dreht, zu behandeln. Da sich $\hat{\omega}$ in der Praxis nur an null annähert, allerdings nie genau null sein wird, wird diese Singularität für den Fall $\hat{\omega} < 10^{-6}$ definiert. Hierfür wird ein Spezialfall eingeführt, in dem wir $\hat{\omega}$ aus der Gleichung entfernen

$$\begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{pmatrix} = \begin{pmatrix} x_t + \hat{v}\cos(\theta_t + \Delta t\hat{\theta})\Delta t \\ y_t + \hat{v}\sin(\theta_t + \Delta t\hat{\theta})\Delta t \\ \theta_t \end{pmatrix}. \tag{2.10}$$

Somit ist das komplette kinematische Modell für den per klassischem Differentialantrieb angetriebenen Rasenmähroboter aufgestellt und es können aus den Motorbefehlen der beiden Räder des Roboters Vorhersagen über dessen Pose getroffen werden.

2.1.2 Odometrie Modell

Mit Hilfe des vorgestellten kinematsichen Modells wird ein nach [13] nicht parametrisiertes Odometriemodell $\boldsymbol{\delta}_t$ für den Rasenmähroboter erstellt. Die Grundstruktur ist in Abb. 2.3 dargestellt. $\boldsymbol{\delta}_t$ sei dafür definiert als

$$\boldsymbol{\delta}_t = \begin{pmatrix} \delta_{\text{rot1}} \\ \delta_{\text{trans}} \\ \delta_{\text{rot2}} \end{pmatrix}. \tag{2.11}$$

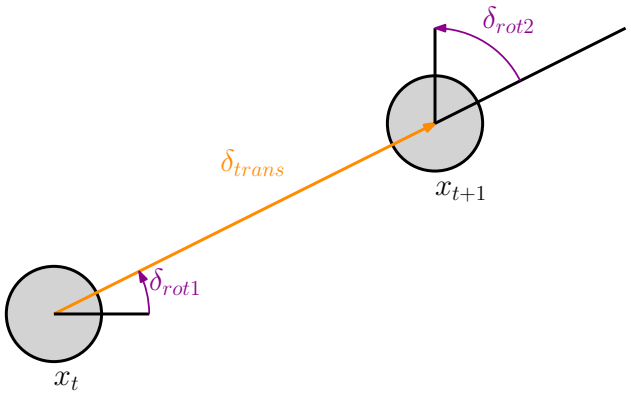


Abbildung 2.3: Darstellung der Odometrieparameter δ_{rot1}, δ_{trans} und δ_{rot2} bei einer Zustandsänderung des Roboters

Wobei δ_{rot1} das Anvisieren des Roboters zu seinem Zielpunkt darstellt, δ_{trans} die gefahrenen Strecke und δ_{rot2} schließlich die Rotation zu der gewollten Ausrichtung repräsentiert.

Hierfür werden die, durch Sensorik gemessenen Radgeschwindigkeiten v_l und v_r benutzt, um die Bewegungsinformationen der einzelnen Räder $\delta_l = v_l \Delta t$ und $\delta_r = v_r \Delta t$ zu modellieren. Daraus ergeben sich dann analog zu Gl. (2.5) und Gl. (2.6) der zurückgelegte Weg Δs und die Orientierungsänderung $\Delta\theta$.

Die durch die Bewegungsinformationen errechnete, neue relative Pose des Roboters ergibt sich aus Gl. (2.10) zu

$$\Delta \boldsymbol{x}_t = \begin{pmatrix} \Delta x_t \\ \Delta y_t \\ \Delta\theta \end{pmatrix} = \begin{pmatrix} \cos(\theta_t)\Delta s \\ \sin(\theta_t)\Delta s \\ \Delta\theta \end{pmatrix} \tag{2.12}$$

und wird schließlich benutzt um $\boldsymbol{\delta}_t$ zu berechnen

$$\begin{pmatrix} \delta_{\text{rot1}} \\ \delta_{\text{trans}} \\ \delta_{\text{rot2}} \end{pmatrix} = \begin{pmatrix} \text{atan2}(\Delta y_t, \Delta x_t) - \theta_t \\ \sqrt{\Delta {x_t}^2 + \Delta {y_t}^2} \\ \Delta\theta - \delta_{\text{rot1}} \end{pmatrix}. \tag{2.13}$$

Hierfür wird die mathematische Funktion atan2 verwendet, welche eine Erweiterung der inversen Winkelfunktion Arkustangens darstellt um als Umkehrfunktion des Tangens auf einen Wertebereich von 360° abzubilden. Da genauso wie im kinematischen Modell auch bei der Aufnahme der Bewegungsinformationen durch die Sensorik diese mit einem Messrauschen und Messfehlern behaftet sind, wird wie in Gl. (2.8) eine normalverteilte Zufalls-

variable $X(\sigma)$ zu den Odometrieparametern $\boldsymbol{\delta}_t$ hinzugenommen

$$\begin{aligned}\hat{\delta}_{\text{rot1}} &= \delta_{\text{rot1}} - X(\sigma_{\text{rot1}}) \\ \hat{\delta}_{\text{trans}} &= \delta_{\text{trans}} - X(\sigma_{\text{trans}}) \\ \hat{\delta}_{\text{rot2}} &= \delta_{\text{rot2}} - X(\sigma_{\text{rot2}}).\end{aligned}$$

Die Varianzen σ_{rot1}, σ_{trans} und σ_{rot2} ergeben sich dafür aus den vier Roboterspezifischen Fehlerprametern $\alpha_1, ..., \alpha_4$

$$\begin{aligned}\sigma_{\text{rot1}} &= \alpha_1 * |\delta_{\text{rot1}}| + \alpha_2 * \delta_{\text{trans}} \\ \sigma_{\text{trans}} &= \alpha_3 * |\delta_{\text{trans}}| + \alpha_4 * (\delta_{\text{rot1}} + \delta_{\text{rot2}}) \\ \sigma_{\text{rot2}} &= \alpha_1 * |\delta_{\text{rot2}}| + \alpha_2 * \delta_{\text{trans}}.\end{aligned}$$

Durch Rückrechnung, auch inverse Kinematik genannt, erhält man letztendlich die neue, geschätzte und fehlerbehaftete Pose $\boldsymbol{x}_{t+1}$ des Rasenmähroboters

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \begin{pmatrix} \cos(\theta_t + \hat{\delta}_{\text{rot1}})\hat{\delta}_{\text{trans}} \\ \sin(\theta_t + \hat{\delta}_{\text{rot1}})\hat{\delta}_{\text{trans}} \\ \hat{\delta}_{\text{rot1}} + \hat{\delta}_{\text{rot2}} \end{pmatrix}. \tag{2.14}$$

2.2 Partikelfilter

Der Partikelfilter bildet eine nicht-parametrisierte Darstellung des Bayes Filters [13],[9]. Beide verlassen sich dabei nicht auf eine festgelegte Funktion für die Einschätzung der Pose des Roboters sondern approximieren diese anhand der Beobachtungen im Arbeitsraum. Diese Posteriors über alle Zustände sind definiert als

$$bel(x_{0:t}) = p(x_{0:t}|u_{1:t}, z_{1:t}). \tag{2.15}$$

Der Partikelfilter repräsentiert die Posteriors anhand einer gesetzten Anzahl an Stichproben und kann so Aussagen über die Wahrscheinlichkeitsverteilung der Pose des Roboters treffen.

Dabei stellt jede Stichprobe, im Folgenden auch Partikel genannt, eine mögliche fehlerbehaftete Pose des Roboters zum Zeitpunkt t dar. Dementsprechend repräsentiert jeder Partikel einen Zustand, sowie eine zusätzliche Gewichtung w_t, die anhand der Messdaten der Grassensoren die Wahrscheinlichkeit der Korrektheit eines Partikels im Verhältniss zu allen Partikeln beschreibt,

$$\boldsymbol{X}_t := \boldsymbol{x}_t^{[1]}, \boldsymbol{x}_t^{[2]}, ..., \boldsymbol{x}_t^{[N]}, \text{ mit } \boldsymbol{x}_t^{[n]} = \begin{pmatrix} x_t^{[n]} \\ y_t^{[n]} \\ \theta_t^{[n]} \\ w_t^{[n]} \end{pmatrix} \text{ und } 1 \leq n \leq N. \tag{2.16}$$

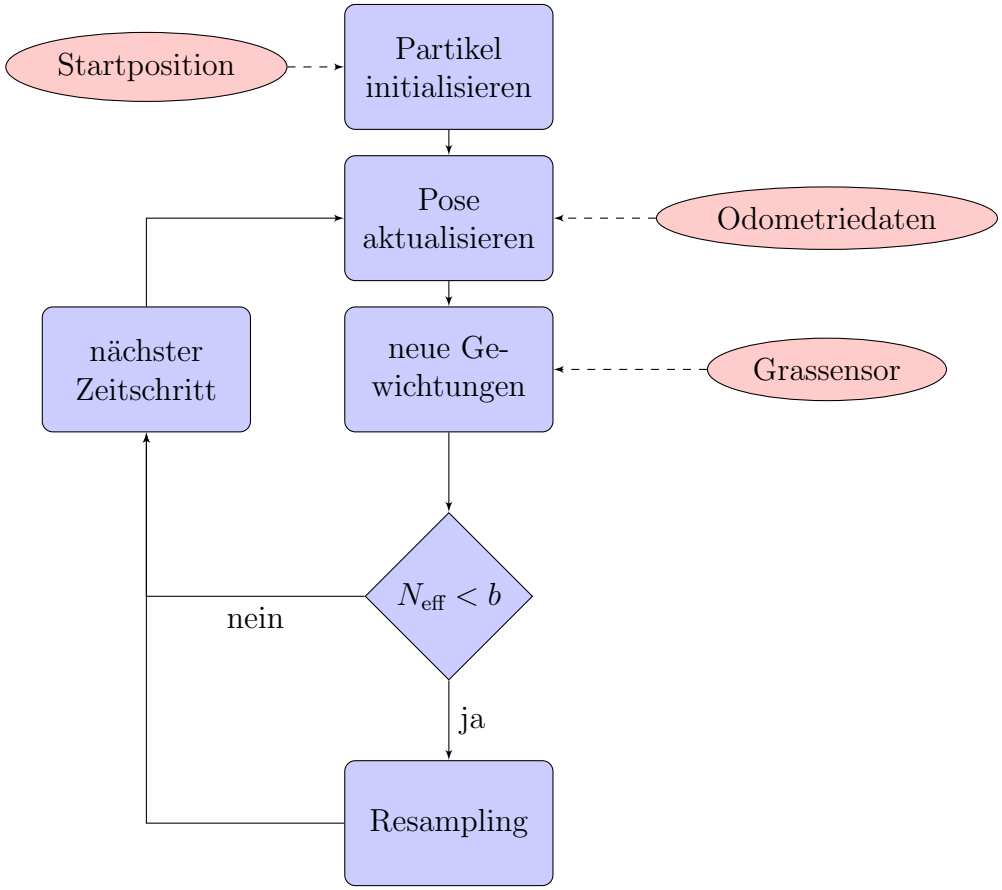


Abbildung 2.4: Flussdiagramm für den Ablauf des Partikelfilters

N definiert dabei die Anzahl an Partikeln, die für die Hypothese über den realen Zustand $\boldsymbol{x_t}$ herangezogen werden.

Der Ablauf eines typischen Partikelfitlers ist im Flussdiagramm in Abb. 2.4 dargestellt und wird in den nächsten Abschnitten genauer erläutert.

2.2.1 Aktualisierung der Partikel

Das Aktualisieren der Partikel von Zeitschritt t zu $t+1$ verläuft rekursiv und jeder Partikel $\boldsymbol{x}_{t+1}^{[n]}$ ist nur von seinem Vorgänger $\boldsymbol{x}_t^{[n]}$ abhängig. Dafür wird das vorgestellte Odometriemodell genutzt, um anhand der gemessenen Odometriedaten δ_t des Roboters eine per Gaußnormalverteilung verrauschte neue Pose $(x_{t+1}^{[n]}, y_{t+1}^{[n]}, \theta_{t+1}^{[n]})^T$ für jeden Partikel $\boldsymbol{x}_t^{[n]}$ zu erstellen (siehe Gl. (2.14)). Dabei ist wichtig zu erwähnen, dass bei dem Aktualisieren jedes einzelnen Partikels neue Zufallsvariablen $X(\sigma_{\text{rot1}})$, $X(\sigma_{\text{trans}})$ und $X(\sigma_{\text{rot2}})$ erzeugt wer-

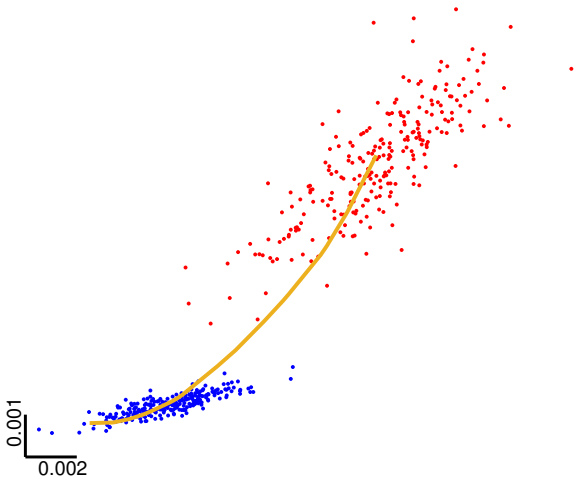


Abbildung 2.5: Beispielhafte Verteilung von 250 Partikeln nach 0.5 Sekunden (blau) und 1.5 Sekunden (rot) um den tatsächlichen Pfad

den, um die Odometrieparameter zu verrauschen

$$\boldsymbol{x}_{t+1}^{[n]} = \begin{pmatrix} x_{t+1}^{[n]} \\ y_{t+1}^{[n]} \\ \theta_{t+1}^{[n]} \\ w_{t+1}^{[n]} \end{pmatrix} = \boldsymbol{x}_t^{[n]} + \begin{pmatrix} \cos(\theta_t^{[n]} + \hat{\delta}_{rot1}^{[n]})\hat{\delta}_{trans}^{[n]} \\ \sin(\theta_t^{[n]} + \hat{\delta}_{rot1}^{[n]})\hat{\delta}_{trans}^{[n]} \\ \hat{\delta}_{rot1}^{[n]} + \hat{\delta}_{rot2}^{[n]} \\ w_{t+1}^{[n]} \end{pmatrix}. \tag{2.17}$$

2.2.2 Gewichtung der Partikel

Für das Aktualisieren der einzelnen Gewichtungen $w_{t+1}^{[n]}$ der Partikel wird nun auf die weitere Sensorik des Rasenmähroboters zurückgegriffen. Dafür werden die beiden Grassensoren S_r und S_l, die rechts und links an dem Roboter angebracht sind, genutzt. Diese unterscheiden binär ob sie sich auf Gras oder nicht befinden. Dabei ist anzunehmen, dass der Arbeitsraumes des Roboters aus Gras besteht und sich außerhalb davon kein Gras befindet

$$w_S = 1 \text{ wenn S über Gras}$$
$$w_S = 0 \text{ sonst.}$$

Neben den tatsächlichen Messungen des Grassensors werden für die Pose $(x_{t+1}, y_{t+1}, \theta_{t+1})^\top$ jedes Partikels die potenziellen Messergebnisse beider Sen-

soren angenommen. Dafür wird anhand der Umgebungskarte das Messergebnis der Sensoren simuliert. Diese Vorhersage kann auch mit prozentualen Messfehlern versehen werden, indem bei einem festgelegten Anteil der simulierten Messungen das Messergebnis geflipt wird.

Die Gewichtung $w_{t+1}^{[n]}$ jedes Partikels wird dann durch den Vergleich der simulierten Messung für seine Pose mit der tatsächlichen Messung der Sensoren erstellt. Dabei seien die tatsächliche Messung als w_{S_r} beziehungsweise w_{S_l} und die simulierte Messung als s_{S_r} beziehungsweise s_{S_l} definiert

$$w_{t+1}^{[n]} = \frac{1 - |w_{S_r} - s_{S_r}^{[n]}| + 1 - |w_{S_l} - s_{S_l}^{[n]}|}{2} + a. \tag{2.18}$$

Die Variable a stellt dabei einen Ausgleich für Messfehler dar, um Partikel, deren Gewichtung durch potenzielle Messfehler falsch sind, nicht direkt auszuschließen.

Letztendlich werden die Gewichtungen noch auf die gesamte Anzahl N der verwendeten Partikel im Partikelfilter normalisiert

$$w_{t+1}^{[n]} = \frac{w_{t+1}^{[n]}}{\sum_{n=1}^{N} w_{t+1}^{[n]}}. \tag{2.19}$$

2.2.3 Resampling der Partikel

Da jeder Partikel $\boldsymbol{x}_{t+1}^{[n]}$ aus $\boldsymbol{X}_{t+1}$ immer abhängig von seinem Vorgänger $\boldsymbol{x}_t^{[n]}$ ist und bei jeder Aktualisierung, also jedem Zeitschritt, in seiner Position mit Fehlern behaftet wird, kann es vorkommen, dass sich diese Fehler aufaddieren. Durch dieses Phänomen entfernen sich die Partikel untereinander immer weiter und beschreiben somit unterschiedliche, potenzielle Positionen des Roboters (ersichtlich in Abb. 2.5). Je nachdem wie groß die Varianzen im kinematischen Modell und im odometrischen Modell sind, fallen diese Entwicklungen stärker beziehungsweise schneller aus.

Um nun besonders die sehr stark ausreißenden Partikel herauszufiltern, wird der Anteil an effektiven Partikeln N_{eff} anhand der Gewichtungen der Partikel errechnet

$$N_{\text{eff}} = \frac{\frac{1}{\sum_{n=1}^{N} (w_t^{[n]})^2}}{N}. \tag{2.20}$$

Unterschreitet N_{eff} einen vorher definierten Grenzwert $0 < b < 1$ wird das Resampling durchgeführt. Dies ist der Fall, wenn die Verteilung der Gewichtungen aller Partikel sehr groß ist. Wenn der Roboter zum Beispiel an den Rand seines Arbeitsraumes gelangt und sich somit eine Unverhältnismäßigkeit

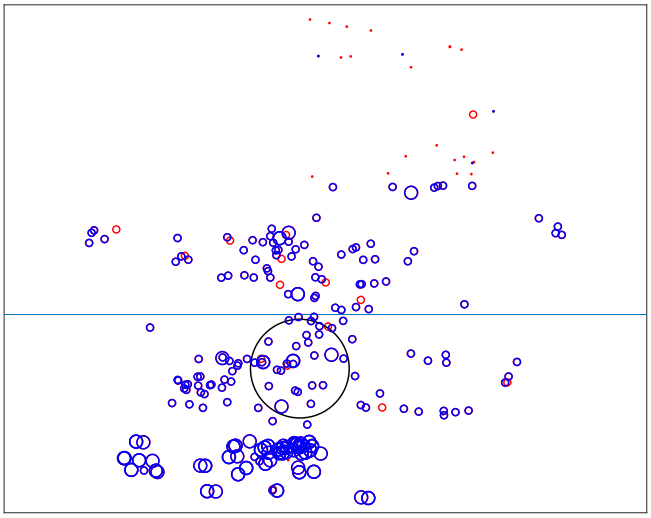


Abbildung 2.6: Resampling der Partikel nachdem der Roboter (schwarz) die Rasengrenze (horizontale Linie) überschritten hat. Rote Partikel mit geringer Gewichtung werden entfernt und blaue behalten. Die Durchmesser der Partikel sind dabei proportional zu der Höhe ihrer Gewichtung.

zwischen den tatsächlichen Messwerten der Sensoren S_r und S_l und den simulierten Messwerten für jeden Partikel einstellt. Die Gewichtung jedes einzelnen Partikels bestimmt dabei die Wahrscheinlichkeit, ob dieser weiter in den Partikeln $\boldsymbol{X}_{t+1}$ enthalten bleibt oder ersetzt wird. Dadurch tendieren Partikel mit geringen Gewichtungen, die also durch die Messungen der Sensoren ausgeschlossen werden können, dazu aus $\boldsymbol{X}_{t+1}$ entfernt zu werden.

Damit N immer gleich bleibt, werden die entfernten Partikel schließlich durch Kopien von anderen, hoch gewichteten Partikeln ersetzt.

Schlussendlich kann noch eine Annahme über die Position $\boldsymbol{x}_{\text{est}}$ des Roboters getroffen werden indem über alle Partikel aus $\boldsymbol{X}_{t+1}$ der Durchschnitt genommen wird

$$\boldsymbol{x}_{\text{est}} = \begin{pmatrix} \frac{\sum_{n=1}^{N} x^{[n]}}{N} \\ \frac{\sum_{n=1}^{N} y^{[n]}}{N} \\ \frac{\sum_{n=1}^{N} \theta^{[n]}}{N} \end{pmatrix}. \tag{2.21}$$

2.3 Simulationsumgebung

Die im Oberen vorgestellte Mechanik und Dynamik des Rasenmähroboters ist in einer, auf der beigelegten CD enthaltenen, Simulation in *Matlab* implementiert. Dort werden auch die im Folgenden vorgestellten Ideen und Algorithmen implementiert und können mit verschiedenen Fehlerparametern getestet werden. Darunter verstehen sich die Parameter $\boldsymbol{\alpha}_k = (\alpha_1, ..., \alpha_6)^\top$ der Kinematik und $\boldsymbol{\alpha}_o = (\alpha_1, ..., \alpha_4)^\top$ der Odometrie. Aus [14] gehen dafür für den Rasenmähroboter in der realen Welt die Folgenden hervor

$$\boldsymbol{\alpha}_k = \begin{pmatrix} \sqrt{0.0012} \\ \sqrt{0.001} \\ \sqrt{0.0057} \\ \sqrt{0.0032} \\ \sqrt{0.0035} \\ \sqrt{0.0046} \end{pmatrix} \qquad \boldsymbol{\alpha}_o = \begin{pmatrix} 0.0849 \\ 0.0412 \\ 0.0316 \\ 0.0173 \end{pmatrix}. \tag{2.22}$$

Weiterhin ist zu erwähnen, dass aufgrund der hohen Rechenzeit beim Aktualisieren der Partikel eine Anzahl von $N = 250$ Partikeln genutzt wird.

Kapitel 3

Abdeckung

In diesem Kapitel wird diskutiert, wie die durch den Rasenmähroboter abgedeckte und dementsprechend gemähte Rasenfläche möglichst zuverlässig dargestellt werden kann. Durch die im kinematischen Modell und der Odometrie hinterlegten Fehler und Unsicherheiten wirft diese Darstellung einige Schwierigkeiten auf, die im folgenden behandelt werden. Dafür wird zuerst der Grundgedanke einer Abdeckungskarte für einen Rasenmähroboter erklärt und welche Vorraussetzungen vorhanden sein sollten, um diese umzusetzen. Darauf aufbauend werden zwei unterschiedliche Verfahren vorgestellt, die geeignet sind, eine solche Abdeckungskarte zu erstellen und in das System zu integrieren. Schlussendlich werden die beiden Verfahren verglichen und eines als Grundlage für die komplett abdeckende Pfadplanung ausgewählt.

3.1 Kartendarstellung

Die Abdeckungskarte resembliert den Arbeitsraum eines Roboters und markiert darin die bereits bearbeitete Fläche. Im Falle des in dieser Arbeit behandelten Rasenmähroboter ist dies eine 2-Dimensionale Karte der zu mähenden Rasenfläche. Dafür wird vorrausgesetzt, dass die Ausmaße des Arbeitsraumes bekannt sind. Zusätzlich wird davon ausgegangen, dass der Arbeitsraum nur aus Ecken v (engl. vertices) und geraden Kanten e (engl. edges) besteht, also als Polygon dargestellt wird. Das Polygon ist dabei durch ein Tupel $\boldsymbol{P}$ definiert

$$\boldsymbol{P} := \boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_n, \qquad \text{mit} \quad \boldsymbol{p}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}. \tag{3.1}$$

Um dies möglichst effizient zu speichern wird auf eine vektorielle Form zur Darstellung der Ecken des Arbeitsraumes zurückgegriffen. Dabei wird es sich zu nutzen gemacht, dass von jeder Ecke eine Kante zu genau einer weiteren

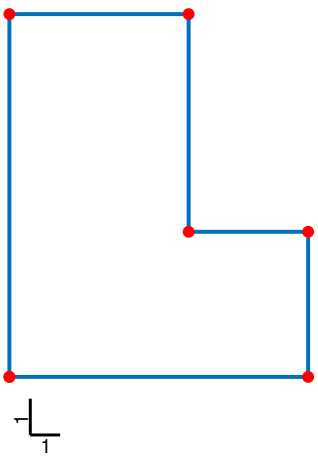


Abbildung 3.1: Darstellung des Arbeitsraumes mit rot markierten Eckpunkten

Ecke führt.

$$\boldsymbol{x}_v = \begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_N \\ x_1 \end{pmatrix} \qquad \boldsymbol{y}_v = \begin{pmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_N \\ y_1 \end{pmatrix} \tag{3.2}$$

für N Ecken des Arbeitsraums und zugehörigen x- und y-Koordinaten x_n und y_n für eine Ecke. Die den eigentlichen Arbeitsraum begrenzenden Kanten ergeben sich durch die Geradengleichung von $\boldsymbol{p}_n$ zu $\boldsymbol{p}_{n+1}$

$$\vec{\boldsymbol{e}}_n = \begin{pmatrix} x_n \\ y_n \end{pmatrix} + \begin{pmatrix} x_{n+1} - x_n \\ y_{n+1} - y_n \end{pmatrix}, \tag{3.3}$$

wobei noch zu beachten ist, dass die $N+1$-ten Einträge identisch mit den ersten sind, um das Polygon zu schließen. Beispielhaft wird dieses Konzept

in Abb. 3.1 für folgende Ecken illustriert

$$\boldsymbol{x}_v = \begin{pmatrix} 0 \\ 10 \\ 10 \\ 6 \\ 6 \\ 0 \\ 0 \end{pmatrix} \qquad \boldsymbol{y}_v = \begin{pmatrix} 0 \\ 0 \\ 4 \\ 4 \\ 10 \\ 10 \\ 0 \end{pmatrix}. \tag{3.4}$$

3.1.1 Matrixdarstellung

Zusätzlich zu der Karte muss noch die Auflösung r der kartesischen Darstellungsmatrix festgelegt werden. Dabei bietet es sich an den effektiven Radius des Arbeitswerkzeugs des Rasenmähers zu beachten. Die Klinge des in dieser Arbeit betrachteten Rasenmähroboters hat einen Durchmesser von circa 20 Zentimeter. Mit einer Auflösung von

$$r = 10 \text{ Zellen pro Meter} \tag{3.5}$$

ergibt sich eine Diagonale von jeder Zelle von

$$d = \sqrt{10^2 cm + 10^2 cm} \approx 14 cm. \tag{3.6}$$

Um zu garantieren, dass der Roboter eine besuchte Zelle auch auf jeden Fall abgemäht hat, darf die Diagonale jeder Zelle maximal 10 Zentimeter betragen. Da der Roboter aber idealerweise durch jede Zelle durch beziehungsweise mindestens an jeder Zelle vorbeifährt, ist mit einer Auflösung von $r = 10$ Zellen pro Meter dafür gesorgt, dass die Zellen auch gemäht werden, wenn der Roboter sie besucht.

Da die Matrixdarstellung der Karte nur eine rechteckige Darstellung erlaubt, werden noch Limitationen x_{lim} und analog y_{lim} mit einem Spielraum von einem Meter definiert als

$$x^+_{\text{lim}} = max(\boldsymbol{x}_v) + 1 \qquad x^-_{\text{lim}} = min(\boldsymbol{x}_v) - 1. \tag{3.7}$$

Die Dimensionen dieser Matrix ergeben sich schließlich zu

$$N = (x^+_{\text{lim}} - x^-_{\text{lim}})r \qquad M = (y^+_{\text{lim}} - y^-_{\text{lim}})r. \tag{3.8}$$

Und damit hat die Abdeckungsmatrix die Form

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & \dots & a_{1M} \\ \vdots & \ddots & \vdots \\ a_{N1} & \dots & a_{NM} \end{bmatrix}. \tag{3.9}$$

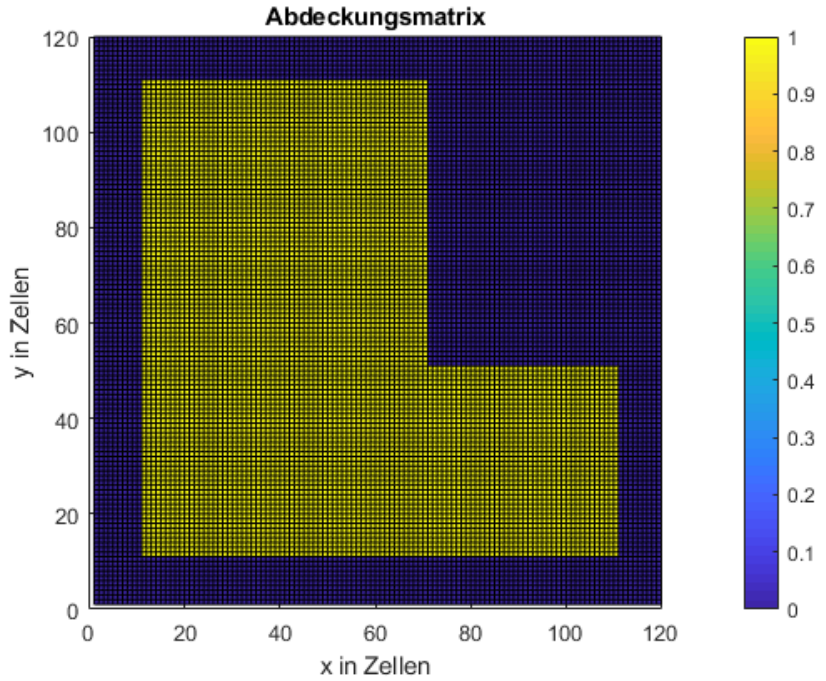


Abbildung 3.2: Darstellung der Abdeckungsmatrix nach vollständigem Mähvorgang, wobei alle gemähten Zellen mit einer Eins und alle ungemähten mit einer Null belegt sind

Auch die Abdeckungskarte wird als eine solche Matrix dargestellt. In ihren Zellen befindet sich die als Wahrscheinlichkeit angegebene Zuversicht, ob sich der Rasenmähroboter dort aufgehalten hat. Zu Beginn sind logischerweise alle Zellen der Matrix mit Nullen gefüllt und nach einer kompletten Abdeckung der Rasenfläche in einem Mähvorgang sind alle Zellen innerhalb des Arbeitsraumes mit einer Eins belegt. Dies wird ersichtlich in Abb. 3.2. Um nun zu einer Position $(x, y)^\top$ die zugehörige Zelle $(\boldsymbol{A})_{ij}$ in der Matrix zuzuordnen werden die Koordinaten der Pose aufgerundet

$$i = \text{round}(x - x_{\text{lim}}^-)r \qquad j = \text{round}(y - y_{\text{lim}}^-)r. \tag{3.10}$$

In einer perfekten Welt ohne Messfehler und ohne Störeinflüsse auf die Dynamik des Rasenmähers ist auch für den Rasenmäher durch seine Odometrie selber immer eindeutig, wo er sich in seinem Arbeitsraum befindet. Damit kann jede besuchte Zelle $(\boldsymbol{A})_{ij}$ genau mit einer Eins belegt werden wenn der Roboter diese betritt. Wie in den Grundlagen bereits erklärt, ist dies bei einem realen Roboter nicht der Fall. Im folgenden werden zwei, auf dem Partikelfilter beruhende, Lösungen dafür vorgestellt.

3.2 Mittelwert basierender Ansatz

Um die im System enthaltenen Unsicherheiten der Kinematik und Odometrie nun in die Abdeckungskarte $\boldsymbol{A}$ mit einzubeziehen, werden immer, wenn sich die geschätzte Posititon (siehe Gl. (2.14)) des Rasenmähroboters in eine neue Zelle der Darstellungsmatrix $\boldsymbol{A}$ bewegt, die Partikel aus dem Partikelfilter in diese übertragen. Die stochastische Verteilung der Partikel wird somit in die Abdeckungskarte übernommen. Dafür wird der Ansatz aus [15] verwendet und die Partikel des aktuellen Zeitpunktes t (siehe Gl. (2.16)) in eine eigene Matrix $\boldsymbol{P}$ überführt. Dies erfolgt über die Anzahl K_{ij} aller Partikel aus $\boldsymbol{X}_t$, die sich mit ihren gerundeten Koordinaten i und j in der gleichen Zelle befinden (illustriert in Abb. 3.3)

$$(\boldsymbol{P})_{ij} = \frac{K_{ij}}{N}. \tag{3.11}$$

Die Aktualisierung der Abdeckungskarte A ergibt sich daraus zu

$$(\boldsymbol{A})_{ij} = (\boldsymbol{A})_{ij} + (\boldsymbol{P})_{ij} - ((\boldsymbol{P})_{ij} * (\boldsymbol{A})_{ij}. \tag{3.12}$$

Dies liegt einer Binomialverteilung zugrunde und wird in [15] näher erklärt.

3.3 Partikel basierender Ansatz

Der zweite, in dieser Arbeit behandelte, Ansatz für eine Abdeckungskarte $\boldsymbol{A}$ beinhaltet, diese direkt in den Partikelfilter zu integrieren. Dafür wird der Partikelfilter erweitert, indem für jeden Partikel aus $\boldsymbol{X}_t$ eine eigene, absolute Abdeckungskarte mitgeführt wird. Diese sind definiert als

$$\boldsymbol{P} = \boldsymbol{P}^{[1]}, \boldsymbol{P}^{[2]}, ..., \boldsymbol{P}^{[N]}. \tag{3.13}$$

Jede dieser Matrizen represäntiert den Pfad eines einzelnen Partikels. Die Zellen $\boldsymbol{P}_{ij}^{[n]}$ der einzelnen Matrix $\boldsymbol{P}^{[n]}$ werden dabei auf eine Eins gesetzt, wenn der zugehörige Partikel $\boldsymbol{x}_t^{[n]}$ mit seinen gerundeten Koordinaten i und j zu einem Zeitpunkt t diese Zelle belegt.

Schließlich wird die Abdeckungskarte $\boldsymbol{A}$ erstellt durch

$$\boldsymbol{A} = \frac{\sum_{n=1}^{N} \boldsymbol{P}^{[n]}}{N}. \tag{3.14}$$

Der grundlegende Unterschied zu dem bereits vorgestellten Ansatz ist, dass durch das Resampling der Partikel im Partikelfilter, auch diese fehlerhaften Annahmen aus der Abdeckungskarte genommen werden. Genau wie ein Partikel $\boldsymbol{x}_t^{[n]}$ aus $\boldsymbol{X}_t$ entfernt und durch eine Kopie eines anderen Partikels $\boldsymbol{x}_t^{[m]}$ ersetzt wird, wird auch die ihn repräsentierende Matrix $\boldsymbol{P}^{[n]}$ aus $\boldsymbol{P}$ entfernt und analog durch eine Kopie der Matrix $\boldsymbol{P}^{[m]}$ ersetzt.

3.4 Vergleich

Um die beiden vorgestellten Herangehensweisen für eine Abdeckungskarte auf ihre Qualität hin zu bewerten, wird nach einem, nicht vollständigen, Mähvorgang die mittlere quadratische Abweichung zwischen dem tatsächlich gefahrenen Pfad des Roboters und der Abdeckungskarte verglichen. Je kleiner dieser Wert ist, desto qualitativ represäntativer ist die Abdeckungskarte. Bei einer mittleren quadratischen Abweichung von null ist diese also perfekt. Dafür fährt der Rasenmähroboter nach dem in der Praxis verbreiteten Zufallsprinzip, so lange geradeaus, bis er auf eine Arbeitsraumbegrenzung trifft und sich dann für einen zufälligen Zeitraum auf der Stelle dreht bis er wieder geradeaus weiterfährt. Dies wird in Abb. 3.3 ersichtlich. Die mittlere quadratische Abweichung, die im folgenden der Einfachheit halber als MSE($\boldsymbol{A}$,$\boldsymbol{G}$) von der Abdeckungskarte $\boldsymbol{A}$ und dem tatsächlichen Pfad $\boldsymbol{G}$ bezeichnet wird, errechnet sich durch

$$\mathrm{MSE}(\boldsymbol{A}, \boldsymbol{G}) = \frac{1}{N * M} \sum_{i=1}^{N} \sum_{j=1}^{M} ((\boldsymbol{A})_{ij} - (\boldsymbol{G})_{ij})^2. \tag{3.15}$$

Für den Vergleich wurden auf zwei unterschiedlichen Karten zehn unabhängige Mähvorgänge bis zu einer absoluten Abdeckung von 90% mit den realen Fehlerparametern $\boldsymbol{\alpha}_k$ und $\boldsymbol{\alpha}_o$ simuliert. Wie in Abb. 3.4 zu sehen sind die beiden Karten auf ihre Größe sowie ihre Komplexität, welche durch die Anzahl der Verwinklungen einhergeht, verschieden. Dabei stellt die zweite Karte die größere und gleichzeitig komplexere da.

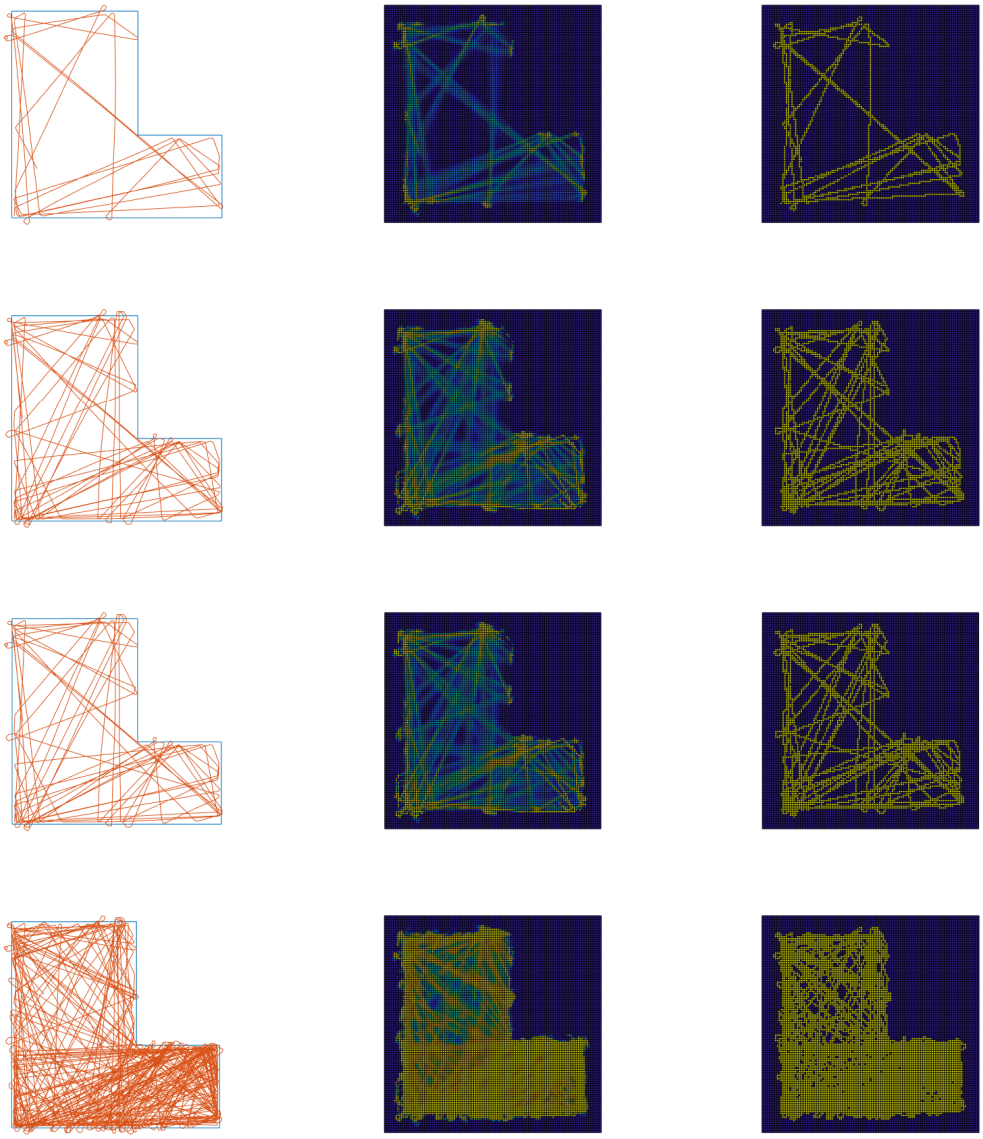


Abbildung 3.3: (von links nach rechts) Fahrtverlauf mit zufälliger Fahrstrategie mit zugehöriger Abdeckungskarte und tatsächlicher Abdeckung bei (von oben nach unten) 25, 50, 75 und 90 Prozent Abdeckung

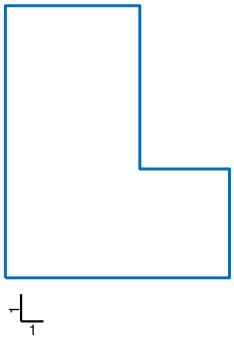


Abbildung 3.4: erste, einfache Karte

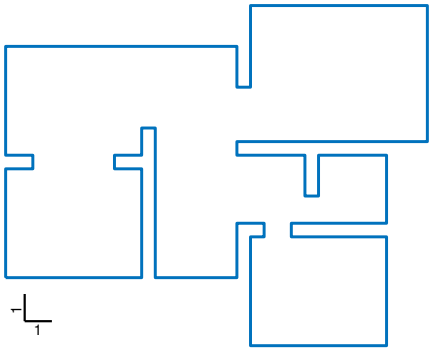


Abbildung 3.5: zweite, komplexere Karte

Die folgende Abb. 3.6 zeigt nun das Ergebnis dieses Vergleiches indem die mittlere quadratischen Abweichung, gemittelt über die $J = 10$ Mähvorgänge für jede Karte, als Verlauf über die prozentuale Abdeckung der Karte abgebildet wird. Zusätzlich dazu wird noch die, durch die zehn Vorgänge entstandene, Standardabweichung mit abgebildet. Wie gut zu erkennen ist, liegt das Maximum des Fehlers bei der ersten Karte höher als bei der zweiten Karte. Dies ist darin zu begründen, dass bei der ersten Karte eine deutlich größere freie Fläche vorhanden ist und dort durch den Partikelfilter mehr, beziehungsweise größere Fehler entstehen. Bei der zweiten Karte trifft der Roboter öfter auf eine Arbeitsraumbegrenzung und relokalisiert den Partikelfilter dementsprechend. Mit der gleichen Begründung ist bei der ersten Karte die Standardabweichung des mittleren quadratischen Fehlers mit steigender Abdeckung deutlich höher.

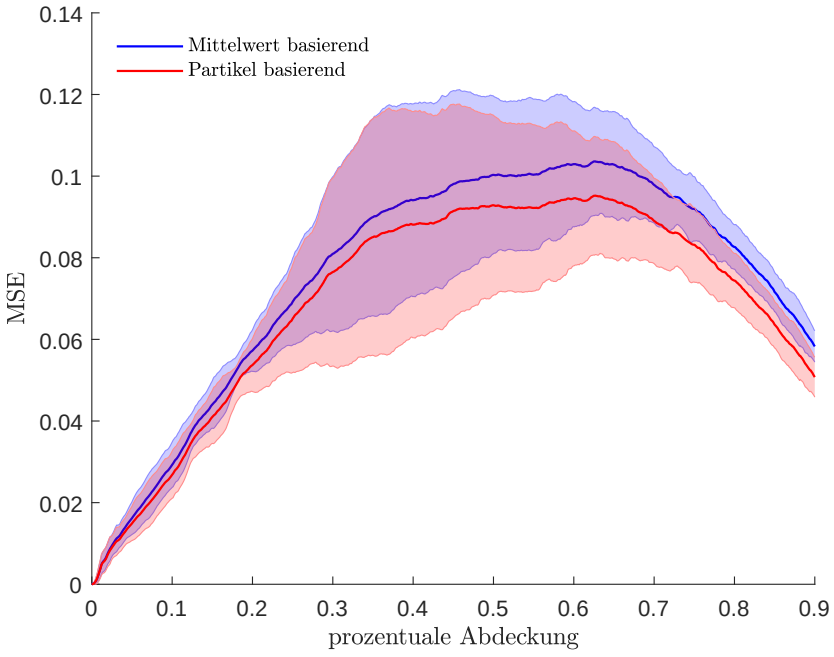
Auffällig ist, dass sich bei beiden Karten der Verlauf der mittleren quadratischen Abweichung sehr ähnlich verhält. Bis zu einer Abdeckung von circa 60 Prozent steigt der Fehler erst noch sehr stark und dann langsam schwächer an und ab 60 Prozent Abdeckung fällt der Fehler wieder ab. Dies liegt an der mit steigender Abdeckung steigender Wahrscheinlichkeit, dass der Roboter über Flächen des Arbeitsraumes fährt, die er vorher bereits besucht hatte. Jedes Mal, wenn der Roboter über eine bereits besuchte Fläche fährt und auch neue Partikel des Partikelfilters dort positioniert werden, werden diese auch in die Abdeckungskarte mit einbezogen. Dadurch wird die Sicherheit an diesen Stellen gewesen zu sein in der Abdeckungskarte höher, und der Fehler geringer. Der Zeitpunkt ab dem der Roboter öfter über bereits besuchte

Flächen fährt und somit Fehler verringert als in neuen Flächen neue Fehler zu erzeugen liegt also bei beiden Karten bei einer tatsächlichen Abdeckung von 60 Prozent. Dies macht besonders diesen Teil bis zum Höhepunkt für den Vergleich interessant. Da allerdings auch höhere Abdeckung wichtig ist, ob der Roboter auch tatsächlich alle Zellen des Arbeitsraumes besucht hat, wird auch dieser Teil nicht vernachlässigt.

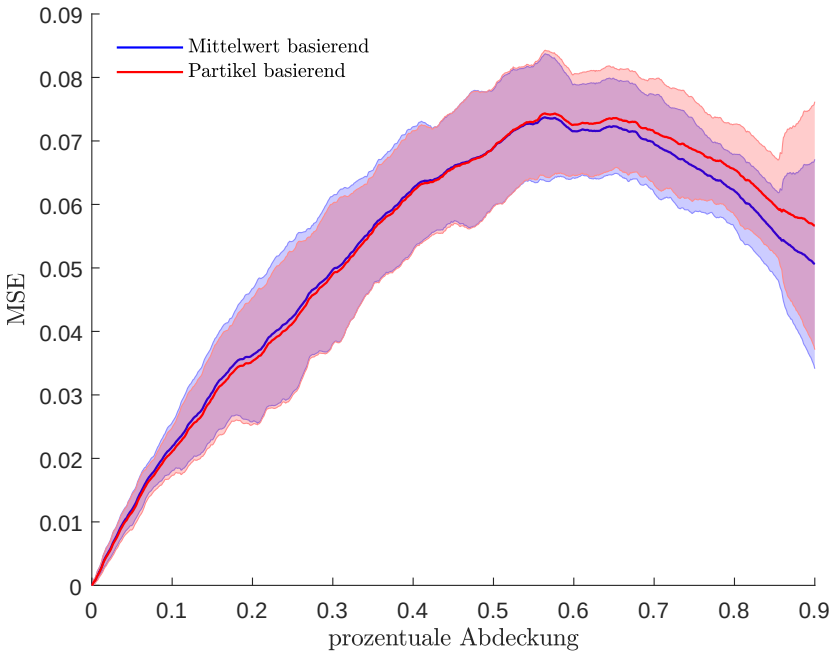
Vergleicht man nun die beiden unterschiedlichen Ansätze für die Abdeckungskarte und ihrene Verlauf des mittleren quadratischen Fehlers, so ist logischerweise kein großer Unterschied zu erkennen, da beide auf die gleichen Partikel als Vorraussetzung zurückgreifen. Bei der ersten Karte schneidet der Partikel basierende Ansatz zwar besser ab, bei der zweiten Karte allerdings gerade bei höherer Abdeckung schlechter. Dass der Fehler bei der komplexeren Karte für den Partikel basierenden Ansatz größer ist, kommt durch das vermehrte Resampling der Partikel des Partikelfilters, wobei ein Teil der Abdeckung aus der Abdeckungskarte gelöscht wird und der Roboter diese Stellen nicht nochmal besucht.

Da aus dem Vergleich der beiden Ansätze für zwei unterschiedliche Karten kein eindeutig besserer Ansatz hervorsticht, wird im folgenden mit dem Mittelwert basierenden Ansatz weiter verfahren. Dies liegt an der besseren Speicherfreundlichkeit des Mittelwert basierenden Ansatzes im Vergleich zum Partikel basierenden. Während der Mittelwert basierende Ansatz nur eine Matrix A der Größe $N \times M$ als Speicher benötigt, ist bei dem Partikel basierenden Ansatz noch zusätzlich für jeden Partikel eine so große Matrix notwendig. Dementsprechend mehr Rechenoperationen sind beim Partikel basierenden Ansatz nötig, wenn beim Resampling der Partikel auch die zugehörigen Matrizen der kopierten Partikel kopiert werden müssen.

Hierbei kann schon ein Ausblick auf eine Kombination der beiden Ansätze gegeben werden, der in künftigen Arbeiten behandelt werden kann. Dabei könnten zum Beispiel die verschiedenen Karten eines jeden Partikels beim Resampling auf die Weise des Mittelwert basierenden Ansatzes verrechnet werden.



(a) erste Karte



(b) zweite Karte

Abbildung 3.6: Verläufe des MSE mit Standardabweichung

Kapitel 4

Pfadplanung

Im folgenden Kapitel wird ein Ansatz aus [16] vorgestellt, um der Problematik des Randomwalks der meisten Rasenmähroboter entgegenzuwirken und die Effektivität von Rasenmährobotern zu steigern. Dafür wird der verwendete Algorithmus vorgestellt und wie die vorher vorgestellte Abdeckungskarte darin mit einbezogen wird.

4.1 Intelligente, komplett abdeckende Pfadplanung

Die im Folgenden beschriebene und erklärte Herangehensweise für eine komplett abdeckende Pfadplanung für mobile Roboter stammt aus [16]. Es handelt sich dabei um ein Verfahren, dass eine lokale Pfadplanung erstellt und keine globale, also vorab definierte. Dies ist sehr wichtig für das dynamische System des Roboters, da der Roboter durch die in der Kinematik und Odometrie enthaltenen Fehler und Unsicherheiten sich nicht sicher sein kann, ob er sich noch auf dem vorab definierten Pfad befindet. So würden die Steuersignale den Roboter nicht genau auf dem Pfad führen und es würden viele ungemähte Flächen entstehen.

Im Gegensatz dazu betrachtet die lokale, also sich immer wieder aktualisierende Pfadplanung, nur die direkte Umgebung des Rasenmähers und kann somit auf Abweichungen in der angenommenen Position des Roboters reagieren und steuert diesen trotzdem zu ausgelassenen Stellen im Arbeitsraum.

4.1.1 Algorithmus

Die Grundidee des Algorithmus besteht aus einer, durch neuronale Netze inspirierten Architektur. Hierfür wird der Arbeitsraum als eine, wie in Ab-

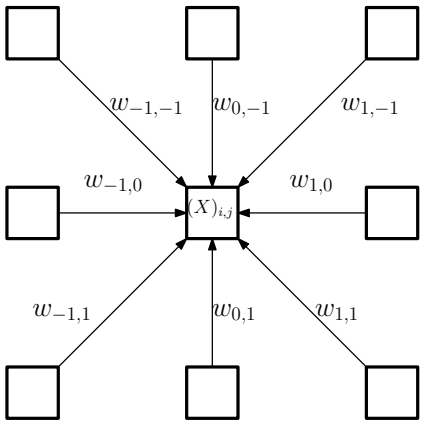


Abbildung 4.1: Gewichtungen auf eine Zelle der neuronalen Aktivität

schnitt 3.1.1 vorgestellte, kartesische Matrix angenommen. Diese zentrale Matrix stellt die neuronale Aktivität dar und wird im Folgenden als $\boldsymbol{X}$ bezeichnet. Diese wird in jedem Zeitschritt Δt anhand der aktuellen äußerlichen Begebenheiten aktualisiert. Jede Zelle der Matrix $\boldsymbol{X}$ entspricht dabei einem Neuron, welches sich abhängig von seinem Aufnahmefeld ändert. Das Aufnahmefeld versteht sich als die Verbindungen von anderen Neuronen in ein Neuron im nächsten Zeitschritt. Dafür seien die acht direkten Nachbarschaftszellen des betrachteten Neurons definiert. Zusätzlich dazu seien noch der äußere Einfluss auch als kartesische Matrix $\boldsymbol{E}$ definiert. Sie beinhaltet die Abgrenzungen des Arbeitsraumes sowie die bereits abgemähten Flächen davon. Da ergibt sich auch die Schwierigkeit in einer fehlerbehafteten Welt, da der Roboter nicht sicher weiß, wo er schon gemäht hat. Darauf wird in Abschnitt 4.2.2 weiter eingegangen. Der äußere Einfluss bildet sich grundsätzlich als

$$(\boldsymbol{E})_{ij} = \begin{cases} E & \text{, wenn nicht gemäht} \\ -E & \text{, wenn im Hinderniss} \\ 0 & \text{, sonst.} \end{cases} \tag{4.1}$$

Die Variable E ist dabei eine große, positive Konstante. Weitere represäntative Konstanten sind A als die passive Verfallsrate, B als die obere Grenze der neuronalen Aktivität $\boldsymbol{A}$ und D als die untere Grenze. Dabei ist zu beachten, dass $E \gg B$ [16]. Dementsprechend ist das Aktualisieren der neuronalen Aktivität definiert als

$$\frac{(\boldsymbol{X})_{ij}}{\Delta t} = -A * (\boldsymbol{X})_{ij} + \big(B - (\boldsymbol{X})_{ij}\big) * (\boldsymbol{W})_{ij} - \big(D + (\boldsymbol{X})_{ij}\big) * (\boldsymbol{E})^{-}_{ij} \tag{4.2}$$

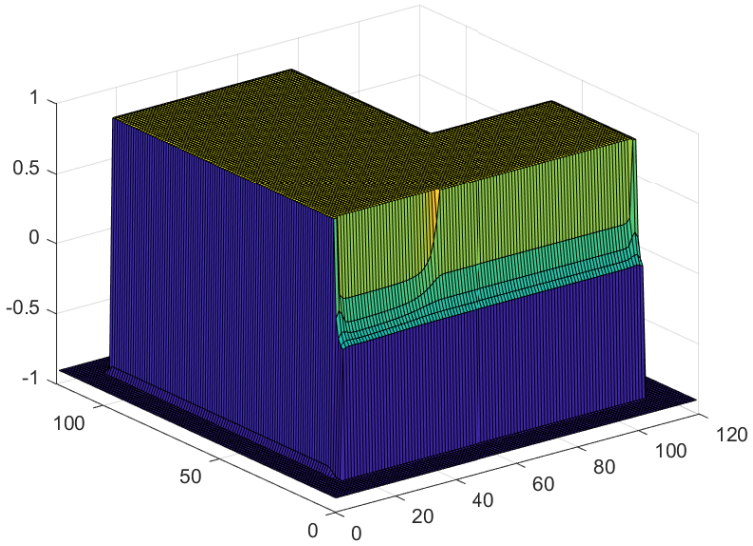


Abbildung 4.2: Abfall der neuronalen Aktivität durch die Aktualisierungsfunktion mit beispielhaften Werten für $B = 1$ und $D = 1$

, mit den Gewichtungseinflüssen

$$(\boldsymbol{W})_{ij} = \left((\boldsymbol{E})^{+}_{ij} + \sum_{k=-1}^{1} \sum_{l=-1}^{1} w_{kl} * (\boldsymbol{X})^{+}_{(i+k)(j+l)} \right) \tag{4.3}$$

und $\boldsymbol{E}^{-}$ als den negativen äußeren Einfluss, $\boldsymbol{E}^{+}$ als den positiven äußeren Einfluss und $\boldsymbol{X}^{+}$ als die positive neuronale Aktivität. Damit sei nur noch festzulegen, wie sich die Gewichtung w zusammenstellt. Da bietet sich der euklidische Abstand zwischen den Zellen, die betrachtet werden, an

$$w_{kl} = \sqrt{k^2 + l^2}. \tag{4.4}$$

Diese Gewichtung wird auch in Abb. 4.1 ersichtlich. Grundlegend beträgt die neuronale Aktivität durch die Aktualisierungsfunktion somit in nicht besuchten Bereichen den Wert B, in Hindernissen den Wert D und nähert sich in besuchten Bereichen langsam der Null an. Dieses Verhalten wird in Abb. 4.2 gezeigt. Um dem Rasenmähroboter nun eine Zelle $\boldsymbol{x}_g$ als Ziel zu übermitteln, werden die lokalen acht Nachbarn der Zelle $(\boldsymbol{X})_{ij}$, in der sich die vermutete Position des Roboters befindet, betrachtet. Davon wird dann unter Zunahme eines Lenkungsparameters C (engl. control) die Zelle mit dem größten Wert als Ziel ausgesucht

$$\boldsymbol{x}_g = \max\big((\boldsymbol{X})_{kl} - C * \Delta\theta, \quad k, l = -1, 0, 1\big). \tag{4.5}$$

Diese Zielzelle wird letztendlich durch den, den Rasenmäher steuernden, PD-Regler angesteuert.

4.1.2 Einbeziehung der Abdeckungskarte

Wie bereits erwähnt weiß der Rasenmähroboter in der Realität nie genau, wo er sich in seinem Arbeitsraum befindet und welche Flächen er bereits abgemäht hat. Er kann lediglich eine Annahme $\boldsymbol{x}_{\text{est}}$ über diese Position anhand des Partikelfilters setzen (siehe Gl. (2.21)) treffen. Aufgrund dessen ergeben sich Probleme bei dem vorgestellten Algorithmus bei der Darstellung des äußeren Einflusses $\boldsymbol{E}$. Wie in Kapitel 3 bereits erklärt werden die Unsicherheiten des Systems in eine Abdeckungskarte übernommen, um die Wahrscheinlichkeit darzustellen, welche Flächen bereits abgemäht wurden. Die Abdeckungskarte kann nun herangezogen werden, um $\boldsymbol{E}$ zu beschreiben. Dafür muss definiert sein, ab welcher Zuversichtlichkeit, also ab wann eine Zelle $(\boldsymbol{A})_{ij}$ aus der Abdeckungskarte $\boldsymbol{A}$, diese Zelle als sicher abgemäht angesehen werden kann. Dies kann mittels eines einfachen Grenzwert $0 < b < 1$ realisiert werden und die Erstellung des äußeren Einfluss $\boldsymbol{E}$ ändert sich damit zu

$$(\boldsymbol{E})_{ij} = \begin{cases} E & \text{, wenn } (\boldsymbol{A})_{ij} < b \\ -E & \text{, wenn im Hinderniss} \\ 0 & \text{, wenn } (\boldsymbol{A})_{ij} \geq b. \end{cases} \tag{4.6}$$

Der generelle Algorithmus wird im folgenden per Pseudocode illustriert

Algorithm 1 Algorithmus zur komplett abdeckenden Pfadplanung

- **Parameter**
  A, B, D, E, C, b, r
- **Eingang**
  $\boldsymbol{x}_{\text{est}}$, $\boldsymbol{x}_v$, $\boldsymbol{y}_v$, $\boldsymbol{A}$
- **Ausgang**
  $\boldsymbol{x}_g$

$\boldsymbol{O} = \text{zeros}(N, M)$
$\boldsymbol{O}(\text{notInPolygon}(\boldsymbol{x}_v, \boldsymbol{y}_v)) \leftarrow 1$
$\boldsymbol{G} = \text{zeros}(N, M)$
for $j \leftarrow 1$ to M **do**
    $(\boldsymbol{G})_{:j} \leftarrow (\boldsymbol{G})_{:j} * (G * (M - j - y^{+}_{\text{lim}} - y^{-}_{\text{lim}})$
end for
$\boldsymbol{X} = \text{zeros}(N, M)$
if neuer Zeitschritt t **then**
    $\boldsymbol{A}(\boldsymbol{A} \geq b) \leftarrow 1$
    $\boldsymbol{A}(\boldsymbol{A} < b) \leftarrow 0$
    $\boldsymbol{E} = E * (\text{ones}(N, M) - \boldsymbol{A}) - 2E * \boldsymbol{O}$
    $\boldsymbol{X}^{+} = \boldsymbol{X}(\boldsymbol{X} \geq 0)$
    $\boldsymbol{W} = \text{zeros}(N, M)$
    for $k, l \leftarrow -1$ bis 1 **do**
        $(\boldsymbol{W})_{i,j} \leftarrow (\boldsymbol{W})_{i,j} + \text{norm}(l, k) * (\boldsymbol{X})^{+}_{i+k,j+l}$
    end for
    $\boldsymbol{E}^{+} = \boldsymbol{E}(\boldsymbol{E} \geq 0)$
    $\boldsymbol{E}^{-} = -\boldsymbol{E}(\boldsymbol{E} < 0)$
    $\Delta\boldsymbol{X} \leftarrow -A{*}\boldsymbol{X} + (B{*}\text{ones}(N, M) - \boldsymbol{X}).{*}(\boldsymbol{E}^{+} + \boldsymbol{W}) - (D{*}\text{ones}(N, M) + \boldsymbol{X}).{*}\boldsymbol{E}^{-}$
    $\boldsymbol{X} \leftarrow \boldsymbol{X} + \Delta\boldsymbol{X} * \Delta t$
    $\text{dec} \leftarrow -\text{inf}$
    for $k, l \leftarrow -1$ bis 1 **do**
        $\text{dec}_{\text{tmp}} \leftarrow (\boldsymbol{X})_{\boldsymbol{x}_{\text{est}}(x)+k, \boldsymbol{x}_{\text{est}}(y)+l} - C * \Delta\boldsymbol{x}_{\text{est}}(\theta)$
        if $\text{dec}_{\text{tmp}} > \text{dec}$ **then**
            $\boldsymbol{x}_g(x) \leftarrow \frac{\boldsymbol{x}_{\text{est}}(x)+k-0.5}{r} + x^{-}_{\text{lim}}$
            $\boldsymbol{x}_g(y) \leftarrow \frac{\boldsymbol{x}_{\text{est}}(y)+l-0.5}{r} + y^{-}_{\text{lim}}$
            $\text{dec} \leftarrow \text{dec}_{\text{tmp}}$
        end if
    end for
end if

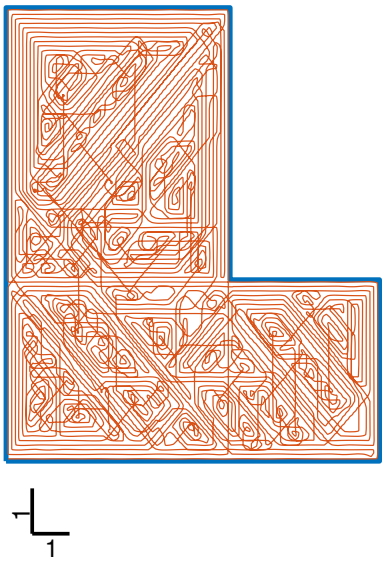


Abbildung 4.3: Fahrtverlauf des Grundalgorithmus bis 99 Prozent Abdeckung

4.1.3 Erweiterungen

Befindet man sich in einer fehlerlosen Umgebung, das heißt besonders, dass die Odometrie nicht von Fehlern behaftet wird, und der Roboter immer genau weiß, wo er sich befindet, so sorgt der Grundalgorithmus für folgendes Fahrmuster. Wie in Abb. 4.3 zu erkennen ist, wird der Roboter durch den kompletten Arbeitsraum gelenkt. Dies geschieht ohne direkte Taktik, aber auch ohne die gleichen Stellen mehrfach abzufahren. Durch die vielen Kurven, die der Roboter dabei zu fahren hat, dauert dieser Mähvorgang letztendlich trotzdem 206 Minuten.

Effektiver wäre es, wenn der Roboter beim Mähvorgang den Arbeitsraum mit möglichst wenig Richtungswechseln abfährt. Dies könnte zum Beispiel erreicht werden durch eine zeilenweise Fahrstrategie erreicht werden. Um dieses Verhalten zu realisieren, wurde der Algorithmus um einen stationären Gradienten erweitert. Dieser sei definiert über den Parameter G, der den Abfall des Gradienten beschreibt. Er wird vor der Entscheidung über die Zielzelle $\boldsymbol{x}_g$ mit einer neuen temporären neuronalen Aktivität $\boldsymbol{X}_{tmp}$ elementeweise

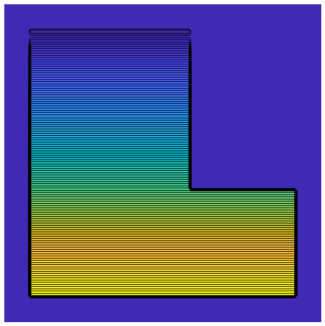


Abbildung 4.4: Gradientverlauf von hoch (gelb) zu niedrig (blau)

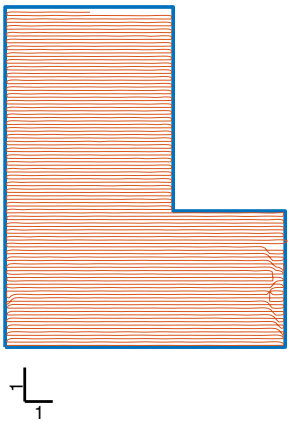


Abbildung 4.5: Fahrtverlauf mit Gradienterweiterung bis 99 Prozent Abdeckung

multipliziert

$$\boldsymbol{X}_{tmp} = \boldsymbol{X} .* \boldsymbol{G} \tag{4.7}$$

Beispielhaft ist dieser in Abb. 4.4 dargestellt. Der beschriebene Gradientenverlauf spiegelt sich auch im Fahrtverlauf des Roboters in Abb. 4.5 wieder. Durch diese taktische Beeinflussung der neuronalen Aktivität verbessert sich auch die Zeit für eine Abdeckung von 99 Prozent in einer fehlerlosen Umgebung auf 37 Minuten. Ein weiterer Vorteil dieser Fahrstrategie kommt bei Einbeziehung der Odometriefehler zur Geltung. Durch das regelmäßige Aufsuchen der Arbeitsraumbeschränkungen kriegt der Partikelfilter öfter die Möglichkeit, sich zu relokalisieren und behält eine durchweg höhere Genauigkeit in der Positionsschätzung. Da sich diese Fehler durch die vielen freien Flächen allerdings trotzdem immer aufaddieren und so zu fehlerhaften Positionsschätzungen des Partikelfilters führen, wurde als weitere Erweiterung ein Grenzwert für die Varianz der einzelnen Partikel eingeführt. Die Varianz bezieht sich dabei auf die Unterschiede der Position und Orientierung unter den einzelnen Partikeln. Sobald dieser Grenzwert überschritten wird, gibt der Rasenmähroboter seine Pfadplanungsstrategie auf und verfällt in ein primitives Wallfolowing-Verhalten. Dabei sucht er die nächstgelegene Arbeitsraumbegrenzung auf und folgt dieser solange, bis er wieder auf die Grenze des bereits abgemähten Arbeitsbereiches trifft. Durch dieses Verhalten relo-

kalisiert sich der Partikelfilter durch das Resampling der Partikel wieder und die Varianz der Partikel geht wieder gegen null.

Als eine weitere Erweiterung, um den Roboter möglichst Zeilenweise und ohne viele Kurven durch den Arbeitsraum zu lenken, wurde der Algorithmus mit einer Variabilität des Steuerungsparameters C erweitert. Durch diesen soll im Idealfall immer wenn der Roboter gerade eine Zeile des Arbeitsraumes abfährt, diese Zeile bis zur Arbeitsraumbegrenzung zu Ende gefahren werden. Dies erweist sich selbst dann als effektiver, wenn laut den äußeren Einflüssen $\boldsymbol{E}$ eine Zelle der aktuellen Zeile bereits gemäht wurde, da der Roboter durch das Ausweichen dieser Zelle Zeit verliert und sich Ungenauigkeiten anhäufen. Die Variabilität von C wurde so gelöst, dass wenn die geschätzte Orientierung des Roboters in der Zeile liegt, also horizontal in eine der beiden Richtungen zeigt, C als C_+ sehr hoch gewählt wird und sonst als C_- sehr niedrig.

4.2 Parameterwahl

Wie in Abschnitt 4.1 bereits erläutert, beinhaltet der Pfadplanungsalgorithmus mehrere ausschlaggebende Parameter die großen Einfluss auf das Verhalten des Roboters ausüben. Dabei ist gerade das Zusammenspiel der Parameter wichtig zu beachten. Dafür wird zunächst die passive Verfallsrate der neuronalen Aktivität mit $A = 10$ definiert. Somit kommt es zu einem langsamen Verfall der neuronalen Aktivität in den abgemähten Bereichen. Diese wird durch den Parameter B mit $B = 1$ als obere Grenze beschränkt. B wird dabei so gewählt, damit der Gradient $\boldsymbol{G}$ möglichst unbeeinflusst auf die neuronale Aktivität multipliziert werden kann. Aufbauend darauf wird der Parameter E, der die äußeren Einflüsse $\boldsymbol{E}$ bestimmt, nach [16] als $E \gg B$ definiert. Dies führt zu der Wahl von $E = 100$. Bei der Wahl der unteren Grenze D der neuronalen Aktivität ist es wichtig, diese auch recht hoch zu wählen, um zu verhindern, dass der Rasenmähroboter eine Zelle außerhalb des Arbeitsraumes als Ziel ansteuert. Dabei erwies sich die Wahl von $D = 1000$ als vorteilhaft.

Der Abfall G des Gradienten $\boldsymbol{G}$, der in Abschnitt 4.3 als Erweiterung eingeführt wurde ist so zu wählen, dass der Roboter von sich aus den Arbeitsraum komplett in einer Richtung Zeilenweise abfährt. Dafür wird er mit $G = \pi$ bestimmt, da dies genau der Differenz der Richtung des Rasenmähers beim Abfahren einer Zeile entspricht. So werden immer noch nicht befahrene Zellen mit höherem Gradientenwert bevorzugt. Damit einhergehend müssen schließlich noch die Lenkungsparameter C_+ und C_- gesetzt werden. Dabei erwies sich die Wahl von $C_+ = 100$ und $C_- = 1.5$ als passend, um das gewünschte Verhalten hervorzurufen.

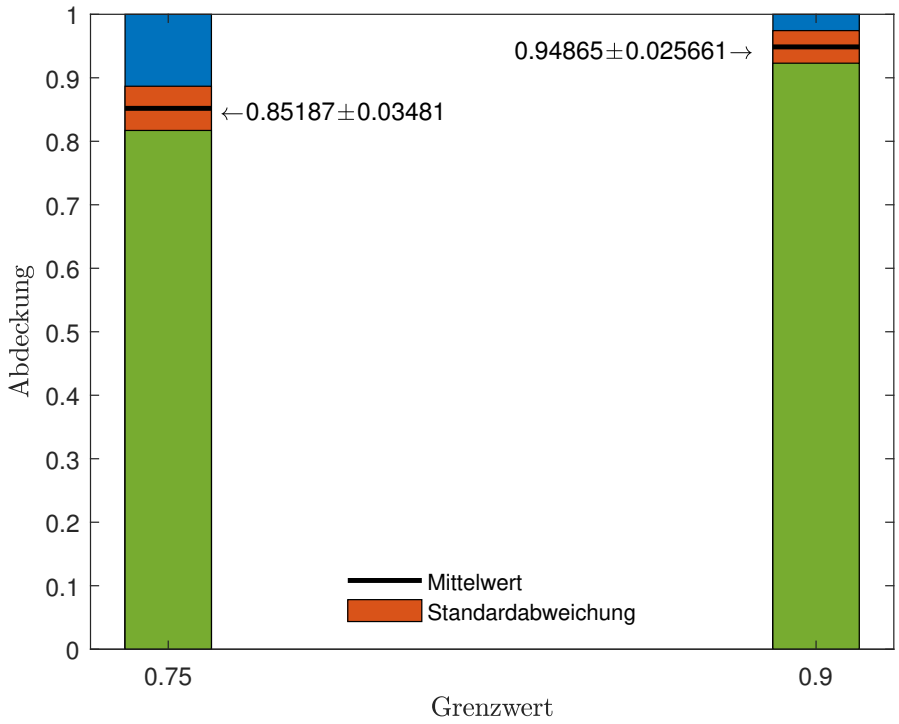


Abbildung 4.6: Durchschnittliche, tatsächliche Abdeckung der Mähvorgänge mit zugehöriger Standardabweichung für die verschiedenen Grenzwerte für die Zuversicht eine Zelle besucht zu haben. Die immer erhaltene Abdeckung ist dabei in grün dargestellt, die Standardabweichung des in schwarz markierten Mittelwertes in orange und die fehlende Abdeckung in blau.

4.3 Evaluation

Die tatsächliche Effektivität des Pfadplanungsalgorithmus wird unter den in Kapitel zwei, Abschnitt 3 vorgestellten Fehlerparametern in der Simulationsumgebung getestet. Dafür werden die gemittelten Ergebnisse aus 15 unabhängigen Mähvorgängen nach dem Randomwalk Prinzip bis zu einer tatsächlichen Abdeckung von 99 Prozent als Maßstab gesetzt. Als Vergleich werden 15 unabhängige Mähvorgänge nach dem vorgestellten Algorithmus zur Hand genommen. Ausschlaggebend für die Bewertung ist dabei die vom Roboter zurückgelegte Strecke, die der Pfadplaner in Anspruch nimmt um eine Abdeckung von 99 Prozent zu erreichen, da dies auch gleichzeitig die Energieeffizienz des Roboters wiederspiegelt. Auf die Fahrtzeit des Roboters wird dabei bewusst nicht geachtet, da diese sehr stark von der Einstellung

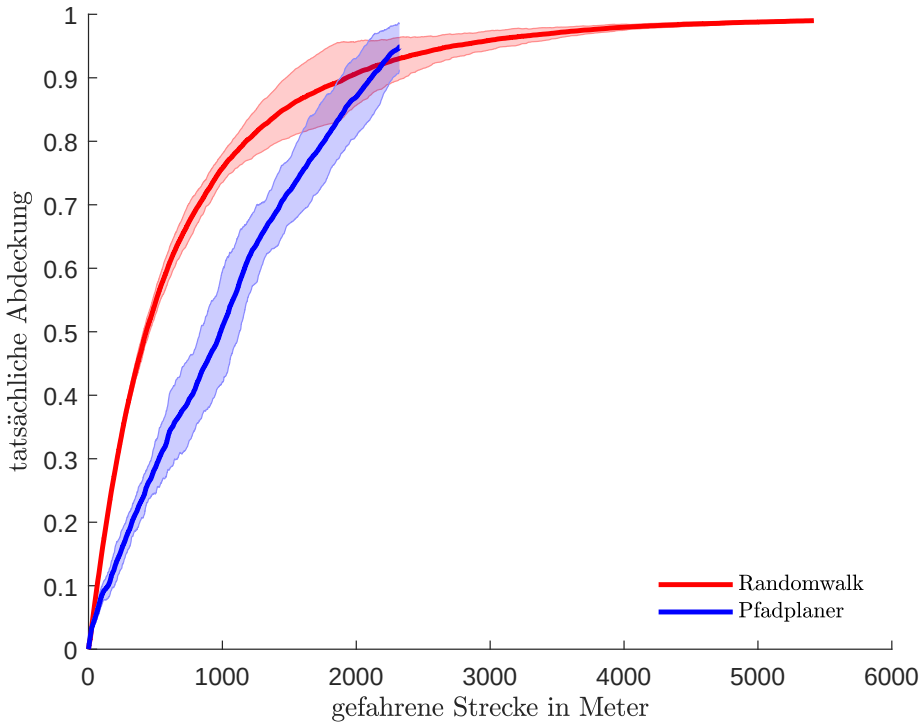


Abbildung 4.7: Vergleich des Randomwalk und dem Pfadplaner gemessen an der tatsächlich abgedeckten Fläche über die dabei zurückgelegte Strecke

des zur Lenkung verwendeten PD-Reglers abhängt und so die Evaluation verfälschen kann. Bei der Bewertung ist darauf zu achten, dass der Pfadplaner nur seine geschätzte Abdeckung, die in der Abdeckungskarte hinterlegt ist, als Ziel nehmen kann. Dafür muss im Voraus festgelegt werden, ab welchem Grenzwert für die Zuversichtlichkeit eine Zelle besucht zu haben, diese auch als abgemäht gilt und in die äußeren Einflüsse mit eingeht. Dies wurde mit jeweils 15 unabhängigen Mähvorgängen mit Grenzwerten 0.75 und 0.9 getestet. Wie in Abb. 4.6 illustriert ist, sorgt der Grenzwert von 0.9 zu einer um zehn Prozent höheren mittleren tatsächlichen Abdeckung und zusätzlich auch zu einer geringeren Standardabweichung. Diesbezüglich wird im Weiteren mit einem Grenzwert von 0.9 für die Zuversicht in einer Zelle gewesen zu sein weitergearbeitet.

In Abb. 4.7 ist nun das Resultat der Simulationen dargestellt. Es ist deutlich zu erkennen, dass der Rasenmähroboter durch den Pfadplanungsalgorithmus deutlich effizienter die Rasenfläche abfährt und dabei trotz der Odometriefehler im Durchschnitt eine hohe Abdeckung von fast 95 Prozent

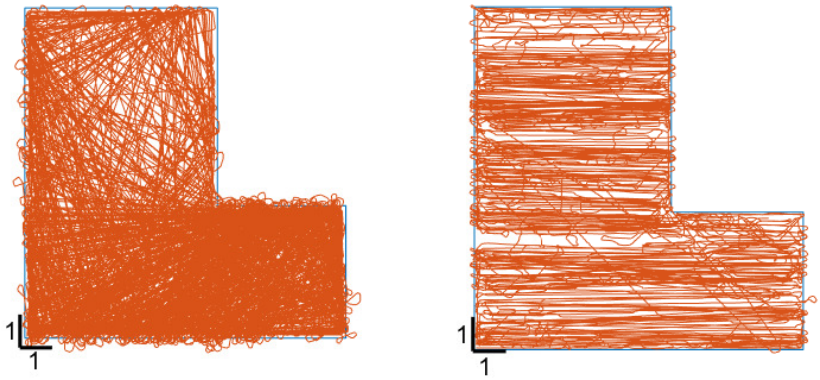


Abbildung 4.8: Beispielhafter Pfadverlauf mit einer Abdeckung von 99 Prozent durch den Randomwalk (links) und von 98 Prozent durch den Pfadplanungsalgorithmus (rechts)

erreicht. Das alles mit einer durchschnittlich zurückgelegten Strecke von nur 2363 Metern. Dementgegen hat der Randomwalk diese Abdeckung erst nach 2718 gefahrenen Metern erreicht. Dies entspricht einer höheren Effizienz von 15 Prozent. Zusätzlich dazu ist dem Roboter, wenn er nach dem Pfadplaner fährt, auch bewusst, dass er seinen Mähvorgang beendet hat und er muss nicht nach einer manuellen und visuellen Einschätzung des Mähergebnisses von Hand abgeschaltet werden.

Es ist deutlich der schwerwiegende Nachteil des Randomwalks an der zwar schnell ansteigenden, aber dann stark abflachenden Kurve der Abdeckung des Arbeitsraums zu erkennen. Dies sorgt dafür, dass der eher geradlinig ansteigende Verlauf der Abdeckung des Pfadplanungsalgorithmus den Randomwalk einholt und übersteigt.

Die in Abb. 4.8 dargestellten beispielhaften Fahrtwege für einen Randomwalk bis 99 Prozent Abdeckung und einem intelligent geplanten Weg bis zu einer Abdeckung von 98 Prozent zeigen sehr gut wie der Roboter durch den

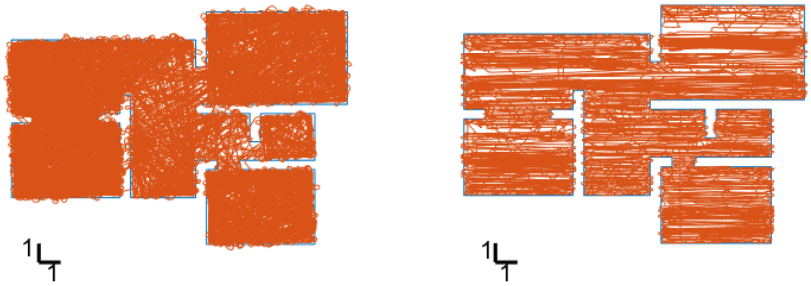


Abbildung 4.9: Pfadverlauf mit einer Abdeckung von 99 Prozent durch den Randomwalk (links) und von 97 Prozent durch den Pfadplanungsalgorithmus (rechts)

Pfadplaner grösstenteils geradlinige Zeilen des Arbeitsraums abfährt. Dabei arbeitet er sich gemäß dem Gradienten von unten nach oben vor. Dass er manche Zeilen mehrfach abfährt, ist dem Grenzwert für die Zuversicht eine Zelle der Abdeckungskarte als besucht zu erachten geschuldet. Es ist noch anzumerken, dass die verwendete Karte, wie bereits in Kapitel 3.4 erwähnt, einen recht einfachen Arbeitsraum darstellt. Gerade für den Randomwalk ist dies sehr vorteilhaft, da es keine Abschnitte gibt in denen er längere Zeit verweilt, da er einen kleinen Ausgang nicht findet. Im Gegensatz dazu ist eine solche Karte sogar von Nachteil für den Pfadplanungsalgorithmus, da dieser durch die Abhängigkeit vom Partikelfilter schneller Probleme mit seiner Genauigkeit bekommt. Deshalb wurden auch für die aus Kapitel 3.4 bereits bekannte, kompliziertere Karte, eine Anzahl von fünf Probedurchläufen mit dem Pfadplanungsalgorithmus auf eine geschätzte Abdeckung von 99 Prozent durchgeführt und mit fünf Randomwalks verglichen. Dies ist beispielhaft in Abb. 4.9 abgebildet. Auch auf dieser Karte ist gut zu erkennen, wie der Ra-

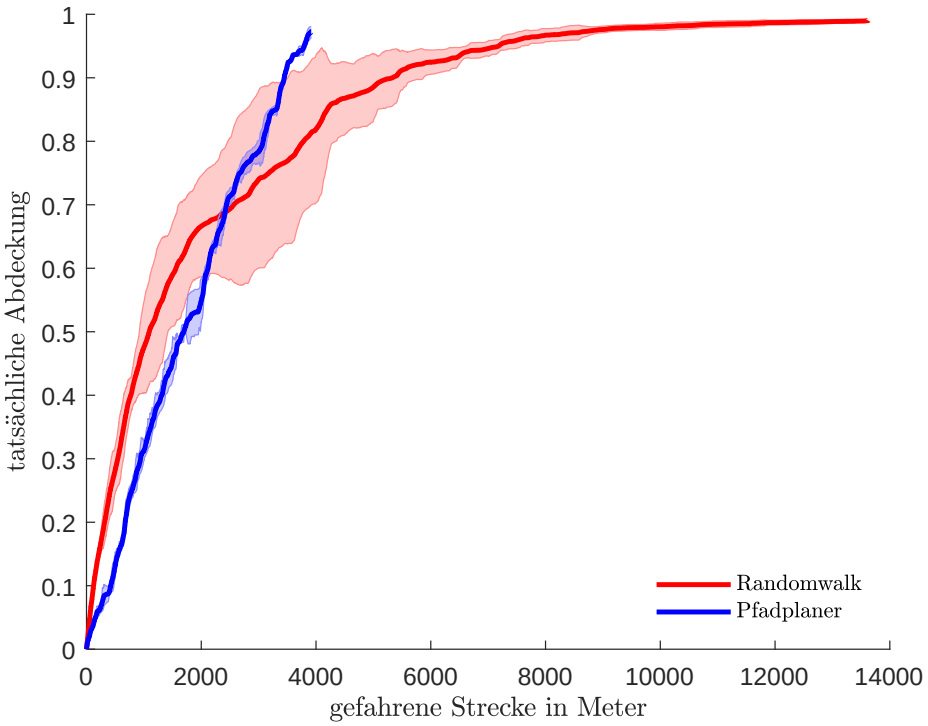


Abbildung 4.10: Tatsächliche, gemittelte Abdeckung des Pfadplaners und des Randomwalks mit der Standardabweichung über der zurückgelegten Strecke für die kompliziertere Karte

senmähroboter durch den Pfadplaner den Arbeitsraum Zeilenweise abfährt und dabei auch jeden Teil des Arbeitsraumes besucht.

Die Verläufe der tatsächlichen Abdeckung, der jeweiligen fünf Mähvorgänge, sind in gemittelter Form auch in Abb. 4.10 illustriert. Dort ist auch sehr gut das erläuterte Problem des Randomwalks auf solch komplizierteren Karten zu erkennen. Dies spiegelt sich in der großen Standardabweichung in dem Bereich von einer Abdeckung von 40 bis 90 Prozent wieder und liegt daran, dass der Roboter in einem Kartenabschnitt verweilt und mehrfach über die gleichen bereits gemähten Stellen fährt. Im Durchschnitt über die fünf durchgeführten Durchläufe schaffte der Roboter eine tatsächliche Abdeckung von knapp über 97 Prozent bei einer zurückgelegten Strecke von 3915 Metern. Im Vergleich dazu hat der Randomwalk für die komplette Abdeckung von 99 Prozent insgesamt 13623 Meter abgefahren und dabei auch erst nach 8689 Metern eine Abdeckung von 97 Prozent erreicht. Dies entspricht einer um 122 Prozent erhöhten Effizienz.

Kapitel 5

Fazit

In der vorliegenden Bachelorarbeit wurde gezeigt, dass bei kostengünstigen Rasenmährobotern trotz hoher Unsicherheiten der Lokalisierung mittels einer intelligenten Pfadplanung eine höhere Effizienz im Vergleich zu nach Randomwalk fahrenden Robotern erreicht werden kann.

Nach einer kurzen Einführung in die Problematik aktueller Rasenmähroboter, folgt eine detaillierte Beschreibung des dieser Arbeit zugrunde liegenden *low cost* Roboters. Dem inbegriffen ist das kinematische und das Odometriemodell und wie die durch den Arbeitsraum und die Sensorik begründeten Fehler in das System einfließen. Außerdem wurde die Funktionsweise des zur Lokalisierung verwendeten Partikefilters erläutert.

Aufbauend auf dem Partikelfilter wurde anschließend die Idee der Darstellung des bereits druch den Roboter besuchten Arbeitsraumes als Abdeckungskarte vorgestellt. Dies wurde an zwei unterschiedlichen Ansätzen getestet, welche auf ihre Genauigkeit hin bewertet wurden. Schließlich wurde für das weitere Vorgehen einer der beiden Ansätze ausgewählt.

Um die Problematik der intelligenten Pfadplanung zu bewältigen, wurde in Kapitel 4 ein Algorithmus vorgestellt, der den Rasenmäherroboter zielgerichtet durch den Arbeitsraum leitet. In diesen wurde die vorher beschriebene Abdeckungskarte miteinbezogen und noch weitere Erweiterungen zur Verbesserung des Fahrverhaltens aufgezeigt. Letztendlich wurde gezeigt, dass der Rasenmähroboter durch den Pfadplanungsalgorithmus auf den getesteten Karten um 15 bis 122 Prozent effizienter arbeitet als durch Randomwalk.

5.1 Diskussion

Den Erwartungen entsprechend wurde in dieser Bachelorarbeit erfolgreich gezeigt, dass der vorgestellte und verwendete Pfadplanungsalgorithmus für eine

Verbesserung der Effizienz gegenüber einem Randomwalk sorgt. Dabei liegt die Verbesserung von 15 Prozent nicht unter diesen. Im Vergleich von dem Kostenfaktor der verwendeten *low cost* Sensorik zu der ausgeprägten Sensorik, die in den meisten teuren autonomen Rasenmähermodellen verbaut ist, ist es nach diesem Ergebnis geradezu erstaunlich, dass diese trotzdem durch einen Randomwalk gesteuert werden. Dazukommend ist außerdem noch die Tatsache, dass die für den Vergleich verwendete Karte zugunsten des Randomwalks aufgebaut ist. Umso bestätigender ist das Ergebnis auf der komplizierteren Karte. Eine um 122 Prozent höhere Effizienz übersteigt dabei jegliche Erwartung für den aktuellen Stand des Algortihmus und bestätigt dadurch stark den Grundgedanken dieser Arbeit.

Die grundsätzliche Implementierung der in dieser Arbeit vorgestellten Funktionalitäten stellte keine große Problematik dar. Durch diverse bereits implementierte Funktionen der Simulationsumgebung von *Matlab* konnten diese der Simulationszeit zugute kommend programmiert werden. Trotzdem stellte die Laufzeit der Simulationen oft Probleme dar. Durch den Partikelfilter wurde diese mit einer geringen Anzahl von 250 Partikeln geradeso auf Echtzeit gehalten. Somit dauerte ein einzelner Durchgang meist bis zu fünf Stunden. An dieser Stelle wäre es noch interessant zu sehen, wie der Algorithmus bei einer deutlich höheren Anzahl Partikeln, zum Beispiel 2000, abschneidet. Dies müsste zu einer höheren Genauigkeit in der Positionsschätzung durch den Partikelfilter führen und damit für eine bessere Darstellung der Abdeckung sorgen. Dies sollte dann schließlich in einem noch besseren Ergebnis des Pfadplaners resultieren. Das größte Problem im Zusammenhang damit stellten allerdings die vielen Parameter, die in den Pfadplaner einfließen dar. Um die Auswirkung der Änderung eines dieser Parameters zu erfahren, bedurfte es einer sehr langen Wartezeit. Somit nahm es eine lange Zeit in Anspruch eine geeignete Konfiguration der wichtigsten Parameter herauszufinden, welche in Kapitel 4 auch erläutert wurde.

Gerade die sehr hoch ausfallenden, unter realen Bedingungen gemessenen Fehlerparameter [12] des kinematischen Modells und der Odometriemessungen stellten den Pfadplaner vor eine große Herausforderung. Erst durch die Implementierung der in Kapitel 4 erklärten Erweiterungen konnte eine richtige Funktionalität des Algorithmus hergestellt werden. Auch diese sind noch nicht perfekt ausgereift und bedürfen noch weiterer Bearbeitung. Dies sollte sich allerdings besonders auf das Perfektionieren der Parametereinstellungen begrenzen.

Durch die erwähnte zeitliche Beanspruchung wurde es bisher auch versäumt, die Funktionalitäten auf einem realen Modell des Rasenmähroboters und unter realen Bedingungen außerhalb der Simulationsumgebung zu testen.

5.2 Ausblick

Wie in Kapitel 3 bereits angedeutet, kann für die Darstellung der Abdeckungskarte noch eine Kombination aus den beiden vorgestellten Ansätzen entwickelt werden und getestet werden, ob diese die Funktionalität des Pfadplaners weiter verbessert.

Desweiteren sollte in weiteren Arbeiten versucht werden, die Parameter des Pfadplanungsalgorithmus besser einzustellen. Diese könnten zum Beispiel mittles Bayesschem Lernen auf einem mit ausreichend Rechenkapazität ausgestatteten Cluster-Rechner erlernt werden.

Letztendlich gilt noch zu testen, wie sich der Algorithmus auf einem realen Rasenmähroboter und unter realen Bedingungen schlägt. Dabei sei zu beachten, den Algorithmus in seiner Rechenzeit den durch einen Mikrocontroller spärlichen Ressourcen anzupassen, um die Echtzeitberechnung der Partikel und des Planers nicht zu überschreiten.

Ausschlaggebend für die Funktionalität bleibt jedoch die hohe Ungenauigkeit der Odometrie. Sollte diese durch weitere, kostengünstige Sensorik noch verbessert werden, so würde sich auch die generelle Effektivität erhöhen. Dafür bietet sich zum Beispiel eine *IMU* (Inertial Measurement Unit) an, die per Sensorfusion in das System miteinbezogen werden kann.

Literaturverzeichnis

[1] A. Sawall, *Zahl der verkauften Haushaltsroboter steigt stark an.* golem.de, 2017.

[2] B. Grüling, *Hausarbeit? Wird künftig von Robotern erledigt.* welt.de, 2016.

[3] R. u. B. J. u. Y. S. Zelinsky, A. und Jarvis, *Planning Paths of Complete Coverage of an Unstructured Environment by a Mobile Robot*, 1993.

[4] P. u. F. H. u. C. L. Taïx, Michael und SouÃ¨res, *Path Planning for Complete Coverage with Agricultural Machines.* SpringerLink, 2006.

[5] S. J. u. C. M. J. u. M. H. u. P. J. H. u. B. S. W. Kang, Jung Won und Kim, *Path Planning for Complete and Efficient Coverage Operation of Mobile Robots.* IEEE, 2007.

[6] W. u. Y. D. Chibin, Zhang und Xingsong, *Complete Coverage Path Planning Based on Ant Colony Algorithm.* IEEE, 2008.

[7] T. Durrant-Whyte, H. und Bailey, *Simultaneous localization and mapping: part I.* IEEE, 2006.

[8] S. u. K. D. u. W. B. Montemerlo, Michael und Thrun, *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem.* AAAI, 2002.

[9] A. u. d. F. N. u. W. E. van der Merwe, Rudolph und Doucet, *The Unscented Particle Filter.* nips.cc, 2001.

[10] S. u. V. G. Jetto, L. und Longhi, *Development and experimental validation of an adaptive extended Kalman filter for the localization of mobile robots.* IEEE, 1999.

[11] G. u. V. R. Bilotti, Alberto und Monreal, *Monolithic magnetic Hall sensor using dynamic quadrature offset cancellation.* IEEE, 1997.

[12] K. Samoilovs-Zuravlovs, *Entwicklung eines Chlorophyllsensors zur zuverlässigen Detektion von Rasenflächen fÃ¼r autonome Mähroboter*, 2017.

[13] W. u. F. D. Thrun, Sebastian und Burgard, *Probabilistic robotics*. MIT press, 2005.

[14] R. Klinckenberg, *Lokalisierung und Regelung für Trajektorienverfolgung mit autonomen Rasenmähern*, 2018.

[15] M. u. B. W. Hess, Jürgen und Beinhofer, *A Probabilistic Approach to High-Confidence Cleaning Guarantees for Low-Cost Cleaning Robots*. IEEE, 2014.

[16] C. Yang, Simon X. und Luo, *A neural network approach to complete coverage path planning*. IEEE, 2004.