Abschlussarbeit Projektseminar Echtzeitsysteme

Team SegFault
Studienarbeit eingereicht von
J. Greven, M. Kramer, M. Kreischer, A. Poljakow, B. Till
am 18. April 2017

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachgebiet Echtzeitsysteme

Elektrotechnik und Informationstechnik (FB18)

Zweitmitglied Informatik (FB20)

Prof. Dr. rer. nat. A. Schürr
Merckstraße 25
64283 Darmstadt

www.es.tu-darmstadt.de

Gutachter: Prof. Dr. rer. nat. A. Schürr
Betreuer: Géza Kulcsár

Erklärung zur Studienarbeit

Hiermit versichere ich, die vorliegende Studienarbeit selbstständig und ohne Hilfe Dritter angefertigt zu haben. Gedanken und Zitate, die ich aus fremden Quellen direkt oder indirekt übernommen habe, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen und wurde bisher nicht veröffentlicht.

Ich erkläre mich damit einverstanden, dass die Arbeit auch durch das Fachgebiet Echtzeitsysteme der Öffentlichkeit zugänglich gemacht werden kann.

Darmstadt, den 17. April 2017

(J. Greven, M. Kramer, M. Kreischer, A. Poljakow, B. Till)

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Aufgabenstellung

Im Projektseminar wurden verschiedene Aufgaben als Wahlmöglichkeiten zur Verfügung gestellt. Hiervon wurden von den Teams drei Gebiete ausgewählt, die dann implementiert werden mussten.
In dieser Ausarbeitung werden die folgenden Themen vorgestellt und näher beschrieben:

- Rundkurs
- Rundkurs mit Hindernissen
- Einparken

1.1 Rundkurs

Die einzige Pflichtaufgabe war es, einen vorher bekannten Rundkurs in möglichst kurzer Zeit zu bewältigen. Hierfür ist eine Karte des Stockwerkes gegeben, die genutzt werden konnte. Außerdem konnte hier die Fahrtrichtung beliebig gewählt werden.

1.2 Rundkurs mit Hindernissen

Bei dieser Aufgabe wurden die gleichen Voraussetzungen wie bei dem Rundkurs ohne Hindernisse angenommen, jedoch wurden Hindernisse in den Weg gestellt. Diese waren zuvor unbekannt, waren also nicht vorher in der Karte erfasst und mussten mittels Sensoren während der Fahrt ermittelt werden.

1.3 Einparken

Die Aufgabe bestand darin, seitlich und parallel an der Wand einzuparken. Hier gab es einiges an Interpretationsfreiraum, weshalb in dieser Ausarbeitung zwei Punkte genauer betrachtet werden:

- Abstand zur Wand d
- Winkel zur Wand φ

Es wird also zunächst einmal geprüft, wie groß der Abstand zur Wand ist, um möglichst nah an der Wand zu stehen und wie groß der Abweichwinkel φ ist (siehe Abbildung 1.1), sodass das Auto am Ende parallel zur Wand parkt.

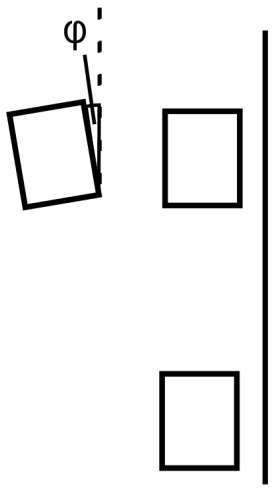

Abbildung 1.1: Abweichwinkel bei dem Einparken

2 Rundkurs

In diesem Abschnitt sollen die Schritte für den Rundkurs kurz näher erläutert werden. Hierfür wird, mittels der Ultraschallsensoren, der Abstand zur Wand über einen PID-Regler geregelt. Sobald die Sensoren einen deutlichen Sprung bei dem Wandabstand wahrnehmen, wird dies als Kurve interpretiert und es wird eine Kurvenfahrt begonnen.

2.1 Umsetzung von Einschlagswinkel auf steering level

Da die Regelung mit Winkeln in Grad rechnet, das Auto jedoch am Ende nur die Werte in steering level bekommt, muss zunächste die Umwandlung bestimmt werden. Dafür wurde durch eine Messreihe folgende Werte bestimmt:

Tabelle 2.1: Übersetzungswinkel von Grad in steering level

Winkel [°]: steering level:	Messreihe 1	Messreihe 2	Messreihe 3	Messreihe 4	∅
-50	-25.65	-23.43	-23.67	-24.69	-24.37
-40	-23.99	-21.99	-22.287	-22.96	-22.81
-30	-20.19	-15.86	-19.19	-18.61	-18.48
-20	-14.15	-15.07	-13.30	-14.18	-14.17
-10	-7.11	-6.32	-5.99	-7.11	-6.64
0	0	0	0	0	0
10	6.86	6.68	6.04	5.85	6.37
20	14.71	12.69	17.63	14.84	14.58
30	18.93	18.18	23.19	18.35	19.18
40	20.34	21.80	24.94	29.16	21.51
50	21.96	23.07	25.92	22.72	22.6

Diese Werte werden dann in einem Graphen Abb. 2.1 aufgetragen und es wird ein Polynom mithilfe dieser Punkte approximiert, sodass die Werte möglichst nahe an der Wirklichkeit liegen.
Durch dies Funktionen kann nun immer die Umrechnung durchgeführt werden als eine Art Look-Up-Table.

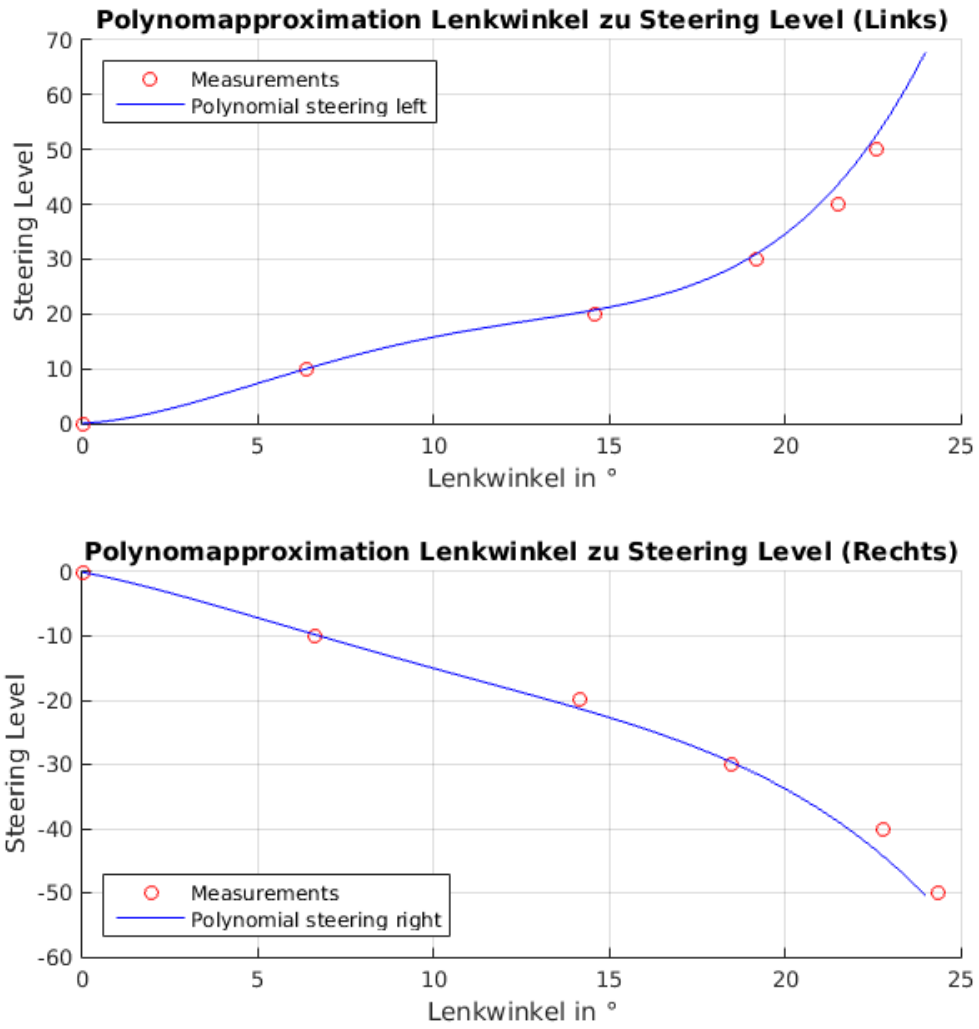

Abbildung 2.1: Approximation der Messwerte

2.2 Regelung des Abstand zur Wand

Zur Regelung des Querversatzes des Fahrzeugs zur Wand ist zuerst ein Modell des Fahrzeugs notwendig, um dann mittels des Modells einen ersten Reglerentwurf machen zu können. Aufgrund seiner Einfachheit und der relativ geringen Geschwindigkeiten mit denen das Fahrzeug fahren wird (bei einem theoretischen Maximum von 2 $\frac{m}{s}$, was in etwa 7 $\frac{km}{h}$ entspricht) bietet es sich an, von einer einspurigen Ackermann-Kinematik auszugehen. Die einspurige Ackermann-Kinematik nähert die Spur beider Radpaare auf jeweils die Spur eines Rades im Mittelpunkt der jeweiligen Achsen an. Aufgrund dieses kinematischen Modells lässt sich nun eine Regelung entwerfen. Zwecks des Reglerentwurfs werden nun die Modellgleichungen in Matlab in Zustandsraumdarstellung

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}, \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{x} \end{aligned} \tag{1}$$

implementiert, um dann mithilfe des *lqr*-Befehls eine linear-quadratische Regelung zu entwerfen. Das Modell sieht in der in Gleichung 1 gezeigten Form aus wie folgt:

$$\begin{aligned} \begin{bmatrix} \dot{y} \\ \dot{\psi} \end{bmatrix} &= \begin{bmatrix} 0 & v \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y \\ \psi \end{bmatrix} + \begin{bmatrix} v\frac{l_H}{l} \\ \frac{v}{l} \end{bmatrix} \varphi \\ y &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} y \\ \psi \end{bmatrix}. \end{aligned} \tag{2}$$

Die in Gleichung 2 auftretende Stellgrösse φ entspricht einer Linearisierung des eigentlichen Lenkwinkels ϕ. Ferner sind die in Gleichung 2 auftretenden Grössen: v die als konstant angenommene Geschwindigkeit des Fahrzeugs, l_H der Abstand zwischen Fahrzeugschwerpunkt und Hinterradachse und l die Gesamtlänge des Fahrzeugs. Der Regler wird nun entworfen, indem das quadratische Gütemaß

$$J = \int_0^\infty \left(\boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + \boldsymbol{u}^T \boldsymbol{R} \boldsymbol{u} \right) dt \tag{3}$$

minimiert wird. Die in Gleichung 3 auftretenden, frei wählbaren Matrizen $\boldsymbol{Q}$ und $\boldsymbol{R}$, gewichten jeweils die Zustände des Systems (hier: der Abstand des Fahrzeuges zur Wand y und der Kurswinkel ψ; $\boldsymbol{x} = [y,\ \psi]^T$) und die aufgebrachten Stellgrössen $\boldsymbol{u}$, was hier jedoch lediglich dem Lenkwinkel ϕ entspricht. Zuletzt muss noch ein entsprechender Vorfilter F ausgelegt werden, um mit der Regelung stationäre Genauigkeit zu erreichen, da der LQR-Regler in seiner Struktur einem PD-Regler entspricht und daher keine stationäre Genauigkeit gewährleistet ist. Gemäß [Lun08, S. 281ff.], lässt sich der Vorfilter $\boldsymbol{F}$ zu

$$\boldsymbol{F} = \left(\boldsymbol{C}(\boldsymbol{B}\boldsymbol{R} - \boldsymbol{A})^{-1}\boldsymbol{B} \right)^{-1} \tag{4}$$

wählen. Die Matrix $\boldsymbol{R}$ in Gleichung 4 ist jedoch die Reglermatrix, die sich aus dem Entwurf des linear-quadratischen Reglers ergibt. Die Zustandsrückführung ergibt sich

damit zu $\boldsymbol{u} = -\boldsymbol{Rx}$. Mit dem dadurch erhaltenen Regler und dem kinematischen Modell in Simulink lässt sich nun die Güte der Regelung abschätzen.

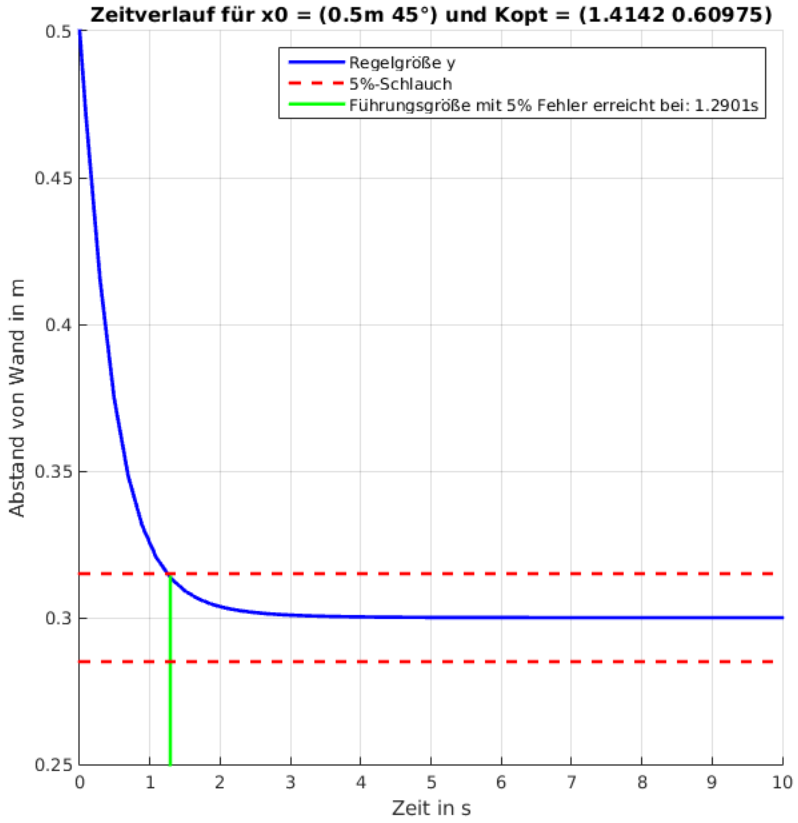

Abbildung 2.2: Zeitverlauf der Regelgrösse y

Wie anhand von Abbildung 2.2 zu erkennen ist, startet das System mit einem Abstand zur Wand von $y_0 = 0.5\ m$ und einem Kurswinkel von $\psi = 45°$. Das System wird ausgehend von diesen Anfangswerten sehr schnell auf den vorgegebenen Wandabstand und einen Sollkurswinkel von $\psi^* = 0°$ eingeregelt, wobei zu beachten ist, dass die Dynamik der Regelung hier stark überschätzt ist, da die Systemdynamik nicht mit in die Reglerauslegung einbezogen wurde und ausserdem Reibungseffekte vernachlässigt werden. Am echten Fahrzeug ist ausserdem noch ein I-Anteil der Regelung beigefügt, wobei dessen Verstärkungsfaktor K_I heuristisch bestimmt worden ist [Lun08, S. 281ff.].

2.3 Kurvenerkennung

Während der Fahrt wird immer wieder der Abstand zur Wand abgefragt. Sobald dieser einen bestimmten Wert überschreitet, wird solange mit Volleinschlag gelenkt bis ein Winkel von 90° erreicht wird. Ist dies geschehen, wird wieder in die Regelung umgeschaltet und die Fahrt wird fortgesetzt. Wie in der Abbildung 2.3 zu sehen, nimmt der linke Sensor an dieser Stelle wahr, dass sich dort keine Wand mehr befindet und es wird

im nächsten Schritt mit der Linkskurve begonnen.

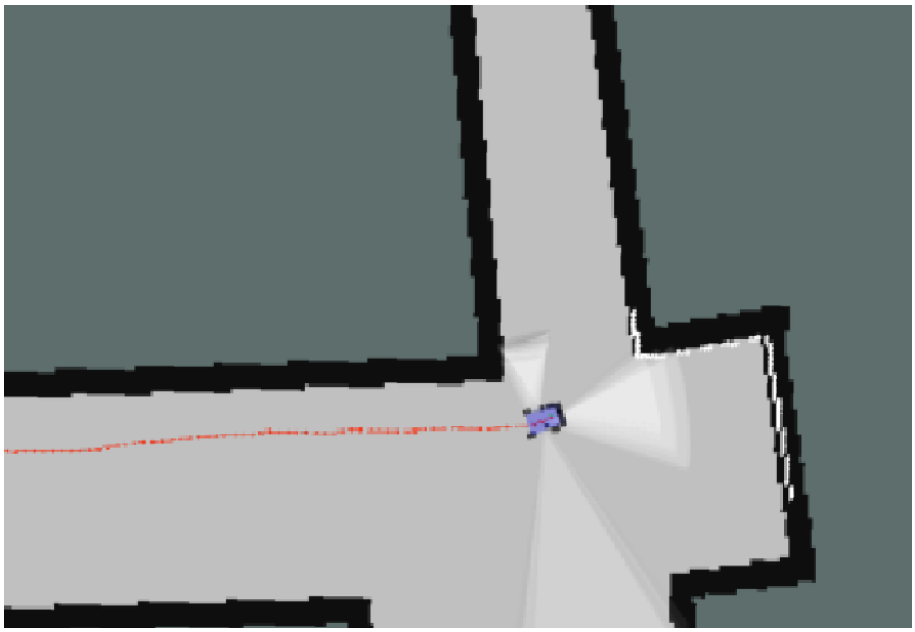

Abbildung 2.3: Kurvenerkennung: Sprung im Wandabstand

2.4 Simulation

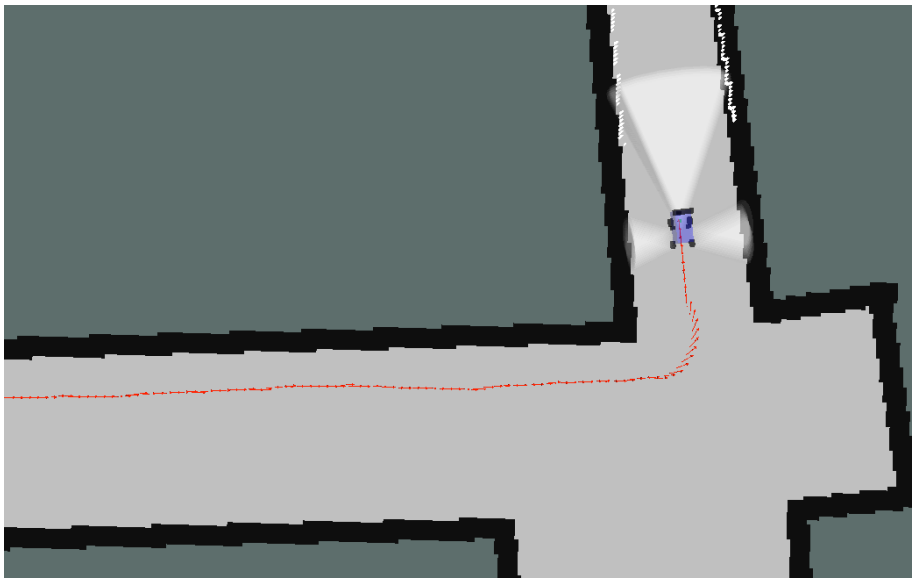

Abbildung 2.4: Kurvenfahrt

In der Simulation ist zunächst der Fahrtweg zu erkennen, der sich an der linken Wand orientiert. Wie schon in Abb. 2.3 zu erkennen, wird dann die Kurve erkannt und es wird 90° gefahren, bevor wieder der PID-Regler übernimmt, sodass die Strecke weiter parallel zur Wand verläuft.

2.5 Problematik

Da die Ultraschallsensoren erst relativ langsam neue Werte bekommen, sind sie für enge Gänge eher ungeeignet. Dieser Fall trat bei der Teststrecke jedoch auf, weshalb aus diesen Gründen hier der gleiche Ansatz wie bei dem Rundkurs mit Hindernissen (siehe Kapitel 3) genutzt wurde. Da der einfache Rundkurs nur eine Vereinfachung zum nächsten Kapitel ist, macht es dies ohne Probleme möglich.

Die in Abschnitt 2.5 beschriebene Problematik wird umgangen, indem der Kinect-Sensor des Fahrzeugs im Folgenden genutzt wird, um ein Tiefenbild der Umgebung zu erhalten. Aus diesem Tiefenbild wird dann, wie in Abbildung 3.1 erkennbar, mittels des ROS-Paketes *depthimage_to_laserscan* ein Laserscan gemacht.

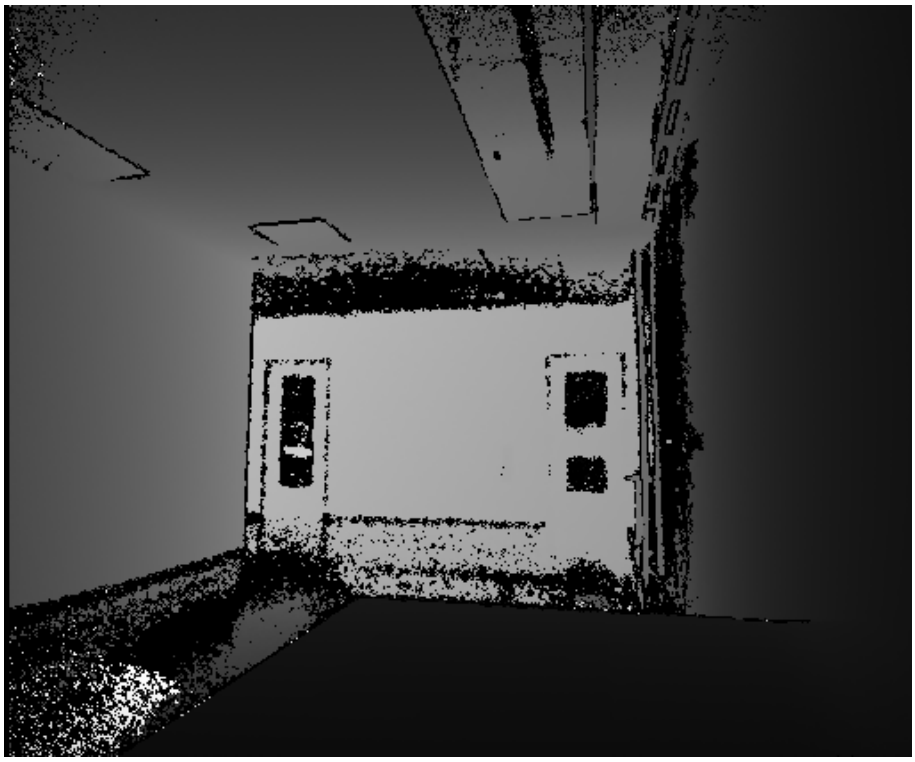

Abbildung 3.1: Durch Kinect-Sensor erhaltener Laserscan

Im Folgenden wird zwecks Ortung des Fahrzeugs im Raum und Trajektorienplanung der *navigation-stack* von ROS benutzt. Dieser untergliedert sich in einen globalen und einen lokalen Planner. Der globale Planner erhält eine Karte der Umgebung, in der jedoch noch keine lokalen Hindernisse eingetragen sind. Außerdem ortet sich der Roboter in der Karte mittels Adaptive-Monte-Carlo Lokalisierung (durch das ROS-Paket *AMCL* gewährleistet). Nun kann aufgrund der Karte, der Startposition des Roboters und der angegebenen Zielpose eine Trajektorie errechnet werden, mittels derer der Roboter zum gewünschten Ziel gelangt. Um diese Trajektorie zu bestimmen wird der *A**-Algorithmus benutzt, der den Dijkstra Algorithmus, um die Verwendung einer Heuristik erweitert [HNR68].

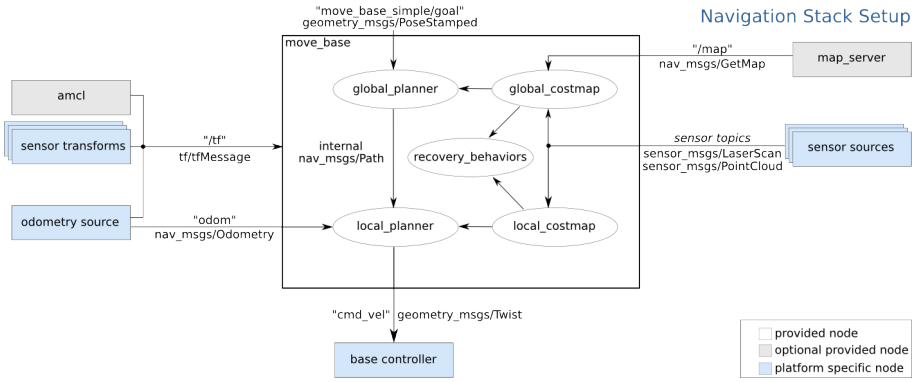

Abbildung 3.2: Blockschaltbild des ROS navigation-Stack [Nav16]

Demnach sind, wie in Abbildung 3.2 erkennbar alle Inputs für den Navigation-Stack gegeben: der Laserscan, die Zielpose des Fahrzeugs, die Startpose über *AMCL* und die Odometrie durch das bereits vorgegebene *Odometry*-Topic des Fahrzeugs. Der Output, das *cmd_vel*-Topic wird von einer einfachen ROS-Node abonniert, die die Geschwindigkeiten und Lenkwinkel von den durch den navigation Stack vorgegebenen SI-Einheiten in die jeweiligen *steering_level* und *motor_commands* umsetzt, die die Mikrocontroller Node des Fahrzeugs erwarten.

3.1 TEB-Local Planner

Als lokaler Planner wurde der *time-elastic band local planner* benutzt. Das Navigationsziel soll kollisionsfrei, in minimaler Zeit erreicht werden, ohne dass dabei kinematische oder dynamische Einschränkungen des Roboters verletzt werden. Zu diesem Zweck werden mehrere mögliche Trajektorien nebeneinander ausgewertet, um zu verhindern, dass der Planner in ein lokales Minimum konvergiert. Es wird die vom globalen Planner errechnete Trajektorie genommen und gemäß den während der Fahrt auftauchenden Hindernissen deformiert. Die On-Line gesichteten Hindernisse werden dann in eine lokale Kostenkarte aufgenommen (vgl. Abb. 3.3).
Dabei bedeuten die roten Markierungen für den Roboter unzulässiges Gebiet, er darf diese Stellen dort also nicht einmal berühren. Bei den blauen Felder darf der Roboter zwar durchfahren, aber sie nicht mit seinem Mittelpunkt überqueren. Mit diesen Voraussetzungen wird dann versucht den bestmöglichsten Weg zu finden.

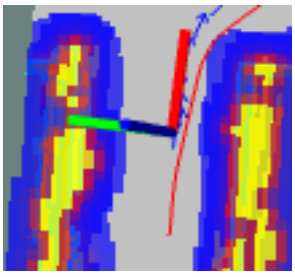

Abbildung 3.3: Lokale Kostenkarte

[RHB15]

4 Einparken

Die Aufgabe bestand darin, eine gute Einparkbewegung zu finden, sodass am Ende das Auto möglichst nah und parallel zu der Wand einparkt, ohne eines der parkenden Autos oder die Wand zu berühren.

4.1 Lückenerkennung

Zunächst muss eine geeignete Parklücke gefunden werden. Geeignet bedeutet in diesem Fall, dass die Lücke breit genug ist, sodass das Auto hineinpasst, und eine bestimmte Mindesttiefe besitzt. Dabei wird die Mindesttiefe der Parklücke so groß wie die Breite des Autos gesetzt (vgl. Abb. 4.1).

Das Auto misst beim Anfahren zunächst den Abstand zur Wand über die Ultraschallsensoren und überprüft, ob die Parklücke groß genug ist. Kommt ein parkendes Auto, hier als Kiste symbolisiert, wird die Lücke als zu klein angenommen und es wird gewartet bis an dem Hindernis vorbeigefahren wurde, bevor die Messung von Neuem beginnt.

Ist eine passende Möglichkeit gefunden, bleibt das Auto stehen und geht in den nächstens Schritt, das Bestimmen des Korrekturwinkels, über.

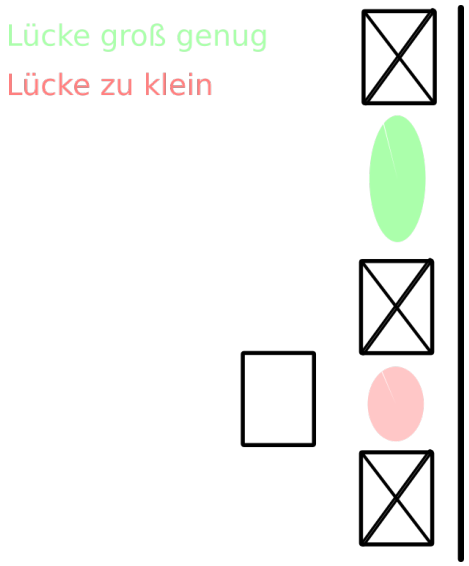

Abbildung 4.1: Lücke wird detektiert

4.2 Rückwärtsfahrt

4.2.1 Korrekturwinkel

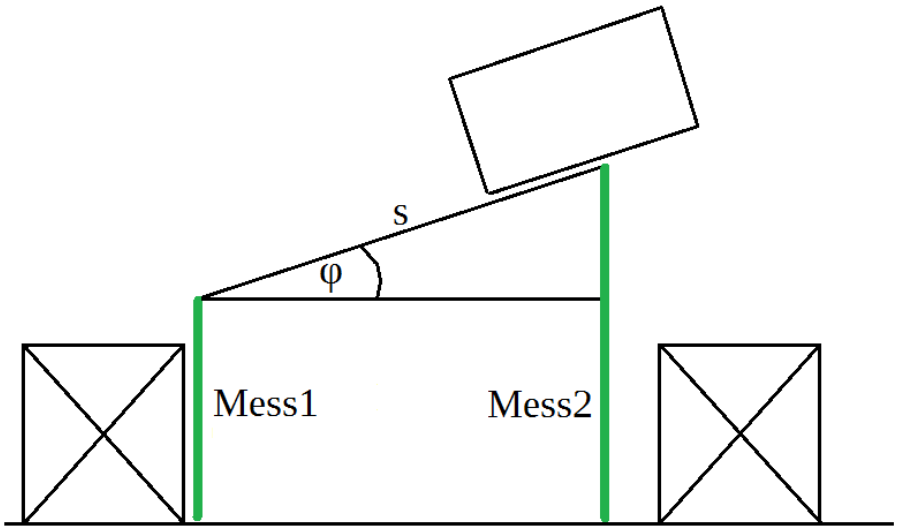

Abbildung 4.2: Draufsichzt einer Einparksituation

Da sich in der Realität die Lenkung des Autos verzogen hat und es oftmals nicht mehr geradeaus fuhr, hatte unsere Gruppe die Idee, das Ausgleichen dieses Winkels als Feature einzufügen. Hierbei wird der Abstand zur Wand kurz vor dem Rückwärtsfahren (Mess2) mit dem Abstand zu Beginn seiner Fahrt verglichen (Mess1), wie in Abb. 4.2 deutlich wird. Hieraus wird über

$$\varphi = sin^{-1}((Mess2 - Mess1)/S) \tag{5}$$

der Abweichwinkel berechnet. Dieser Winkel φ wird bei der Rückwärtsfahrbewegung berücksichtigt, sodass das Auto zum Schluss parallel zur Wand parkt.

4.2.2 Einparkbewegung

Das seitliche Einparken findet in zwei Phasen, einer Links- und einer Rechtskurve, statt. Beide Kurvenkreise unterscheiden sich nur durch ihre Radien. Der Bogenwinkel Θ der gefahren wird, ist bei den jeweiligen Kurvenbewegungen der Gleiche.

$$\Theta = cos^{-1}\Big(1 - \frac{a}{r + r0}\Big) \tag{6}$$

ist einzig und allein abhängig vom Wandabstand der Messung Mess2 (siehe Abb. 4.2). In Phase 1 steht das Fahrzeug still und beginnt mit der Rückwärtskurve. Die einzelnen

Phasen werden auch noch einmal in Abb. 4.3 deutlich. Das Auto wird jedoch niemals, wie im Bild, gerade an der Wand steht, sondern immer leicht verschoben um einen Winkel φ sein. Der Winkel φ aus der vorherigen Berechnung wird von Θ abgezogen, jedoch nur in der ersten Phase, in Phase zwei bleibt Θ unverändert. Dadurch wird simuliert, dass das Auto ein Stück rückwärts gefahren ist beziehungsweise wird die Kurvenfahrt in der 1. Phase als kürzer angenommen, da das Auto nicht perfekt parallel zur Wand steht. Es gilt aber zu beachten, dass die vorgenommene Vereinfachung nur für kleine Winkel von φ gilt. Für die benötigten und betracheten Fälle aber vollkommen ausreichend und genau genug.

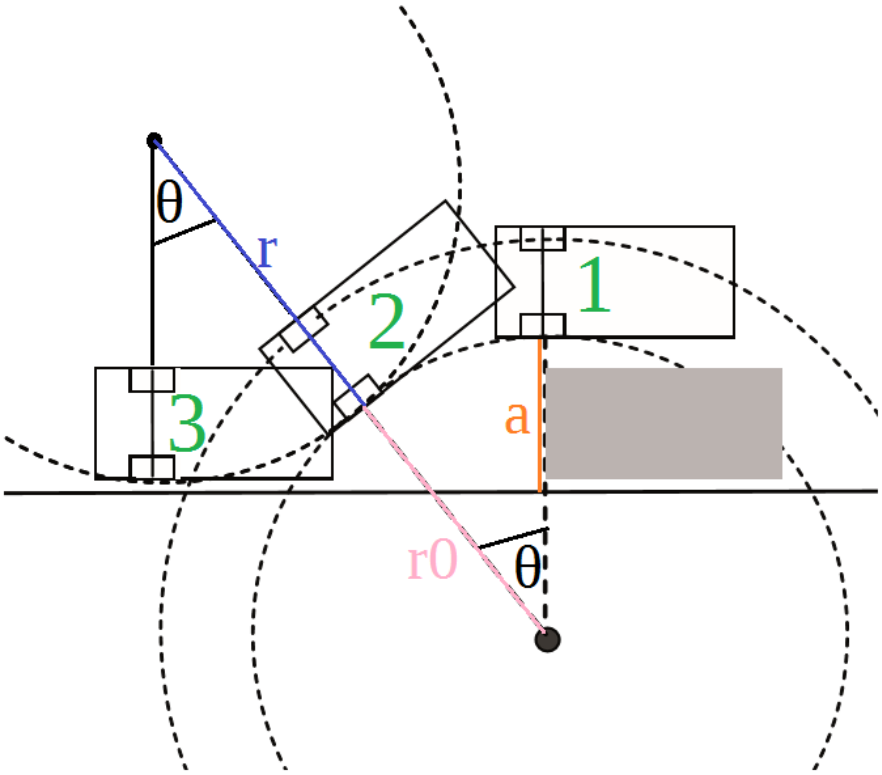

Abbildung 4.3: Einparkbewegung

Wenn ca. 80% der Rückwärtskurve in der ersten Phase erreicht sind, reduziert sich die Geschwindigkeit, bis die zweite Phase eintritt. Der Lenkeinschlag wird maximal entgegengesetzt gestellt und das Tempo wird erhöht. Gegen Ende der zweiten Phase wird das Tempo langsam auf Null reduziert. Phase 3, die Endposition ist erreicht: das Auto parkt.

Zur abschließenden Sicherstellung, dass das Auto nicht doch gegen ein Hindernis oder eine Wand prallt, wird Phase 2 unterbrochen, wenn ein Mindestseitenabstand unterschritten wird. Es geht also direkt in Phase 3 über.

Literatur

[HNR68] Hart, P. E. ; Nilsson, N. J. ; Raphael, B.: A Formal Basis for the Heuristic Determination of Minimum Cost Paths. In: *IEEE Transactions on Systems Science and Cybernetics* 4 (1968), July, Nr. 2, S. 100–107. – ISSN 0536–1567

[Lun08] Lunze, J.: *Regelungstechnik 2*. Springer, 2008 (Springer-Lehrbuch Bd. 2). – ISBN 9783540784623

[Nav16] *Setup and Configuration of the Navigation Stack on a Robot*. http://wiki.ros.org/navigation/Tutorials/RobotSetup. Version: 2016

[RHB15] Rösmann, C. ; Hoffmann, F. ; Bertram, T.: Planning of multiple robot trajectories in distinctive topologies. In: *2015 European Conference on Mobile Robots (ECMR)*, 2015, S. 1–6

A Anhang

A.1 Sonstige Quellen

http://www.inf.fu-berlin.de/inst/ag-ki/rojas_home/documents/tutorials/Weltformel.pdf