TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Elektro- und Informationstechnik Institut für Regelungs- und Steuerungstheorie

Projektarbeit

Hauptseminar Automatisierungs-Mess und Regelungstechnik

Modul: Control

Konstantin Wrede

Matrikelnummer: 4676755

Betreuer

Dr.-Ing. Carsten Knoll

Eingereicht am: 19. Januar 2020

Inhaltsverzeichnis

1. Analyse der Aufgabenstellung

1.1. Aufgabenstellung und Ziel

Das Ziel des Hauptseminars Automatisierungs- Mess und Regelungstechnik besteht darin, dass ein Fahrzeug (mobiler Roboter) dem Straßenverlauf eines vorgegebenen Parcours folgt, passende Parklücken erkennt, vermessen und bewerten kann sowie Parkvorgänge eigenständig durchführt. Der Bediener soll dabei über einen Tablet-Computer als Eingabegerät verschiedene Betriebsmodi des Fahrzeuges steuern und vom Roboter gesammelte Informationen abfragen können. Dabei ergibt sich für das Control-Modul die Aufgabe, Regelungen für den Roboterantrieb, Linienverfolgung und Bahnfolge zu implementieren.

1.2. Zusammenfassung einzelner Teilaufgaben

Zunächst ist ein gegebener Algorithmus zur Erkennung und Verfolgung einer schwarzen Linie auf dem Spielfeld zu verbessern. Dies soll mit einem PID-Regler unter Auswertung von Lichtsensoren, deren Messwerte vom Perception-Modul aufbereitet werden, geschehen. Aus kinematischen Überlegungen und einem experimentell ermittelten Zusammenhang zwischen der pulsweitenmodulierten Leistung (PWM) und Raddrehzahl kann eine Steuerung der translatorischen und rotatorischen Robotergeschwindigkeit vorgenommen werden. Daran anschließend ist eine PID-Regelung der Radgeschwindigkeiten vorzunehmen, um Anhängigkeiten von der Akkuladung und Ungleichheiten der Motore zu minimieren. Zur Querregelung der Geradeausfahrt ist ein gegebenes kinematisches Modell zu linearisieren und auszuwerten, so dass darauf basierend ein geeigneter Reglertyp ausgewählt wird. Die Drehzahlregelung kann dabei unterlagert eingesetzt werden. Zuletzt soll eine Möglichkeit zur Berechnung und Abfahrt von Polynomen auf dem Spielfeld implementiert werden, mit der glatte Ein- und Ausparkvorgänge realisiert werden.

1.3. Schnittstellen des Control-Moduls

Das Control-Modul benötigt die Sensordaten für die Winkeländerung der Räder und der Lichtsensoren vom Perception-Modul. Posendaten in Form von x- und y-Koordinate auf dem Spielfeld (Position) sowie Winkel werden vom Navigation-Modul bereitgestellt. Dabei werden im Konstruktor der Control-Klasse Instanzen des Perception- und Navigation-Moduls übergeben. Die Anweisungen bezüglich des Fahrmodus oder der Zielpose sind vom Guidance-Modul vorzugeben. So kann mittels setter-Methoden die translatorische sowie rotatorische Robotergeschwindigkeit, die aktuelle Pose oder auch Zielpose vorgegeben werden. Mit der `setDriveFor`-Methode kann eine Kombination alle dessen als Vorbereitung einer geregelten Geradeausfahrt auf eine Zielpose geschehen. Durch die `setParkingData`-Methode wird mit der Vorgabe von Start- und Zielpose die Abfahrt eines Polynoms zum

Ein- oder Ausparken vorgegeben. Durch die `setCtrlMode`-Methode kann als Modus die Linienverfolgung (in verschiedenen Geschwindigkeiten), die gergelte Posenanfahrt, das Abfahren von Polynomen oder Stehenbleiben festgelegt sowie mit der `getCtrlMode`-Methode abgefragt werden. Mit der `exec_CTRL_ALGO`-Methode wird der vorher festgelegte Control-Modus ausgeführt.

2. Digitaler PID-Regler

2.1. Mathematische Vorbetrachtungen

Zunächst wird von einem zeitkontinuierlichen PID-Regler der folgenden Form ausgegangen:

$$K(s) = K_P \left(1 + \frac{K_I}{s} + K_D s\right)$$

Dabei entspricht K_P dem Proportionalbeiwert, K_I dem Integrierbeiwert und K_D dem Differenzierbeiwert.
Durch Bilineare Transformationen [1] ergibt sich demnach im Z-Bereich mit Abtastzeit T_A :

$$K(z) = K_P \frac{\left(1 + \frac{K_I T_A}{2} + \frac{K_D}{T_A}\right) + \left(-1 + \frac{K_I T_A}{2} - 2\frac{K_D}{T_A}\right) z^{-1} + \left(\frac{K_D}{T_A}\right) z^{-2}}{1 - z^{-1}}$$

Für eine Rücktransformation in den diskreten Zeitbereich vergleicht man die Koeffizienten dieser Darstellung mit der allgemeinen Übertragungsfunktion:

$$G(z) = \frac{c_0 + c_1 z^{-1} + c_2 z^{-2}}{d_0 + d_1 z^{-1} + d_2 z^{-2}}$$

Demnach ergeben sich die Koeffizienten zu:

$$c_0 = K_P \left(1 + \frac{K_I T_A}{2} + \frac{K_D}{T_A}\right)$$
$$c_1 = K_P \left(-1 + \frac{K_I T_A}{2} - 2\frac{K_D}{T_A}\right)$$
$$c_2 = K_P \left(\frac{K_D}{T_A}\right)$$
$$\text{und } d_0 = 1,\ d_1 = -1,\ d_2 = 0.$$

Aus der Übertagungsfunktion des Reglers $K(z) = \frac{U(z)}{E(z)}$ und mittels inverser Z-Transformation ergibt sich die zeitdiskrete Differenzengleichung:

$$u[k] = c_0 e[k] + c_1 e[k-1] + c_2 e[k-2] + u[k-1]$$

Die aktuelle Stellgröße $u[k]$ kann in dieser Form zu jedem Methodenaufruf aus dem vorangegangenen Wert $u[k-1]$ und Samples der Regelabweichung e bestimmt werden.

2.2. Implementierung im Programm

Die Konstanten c_i sowie Variablen der Regelabweichungen e und Stellgrößen u werden als Klassenvariablen gespeichert. Im Konstruktor werden dem PID-Regler-Objekt die initiale Führungsgröße, Abtastzeit sowie Reglerbeiwerte übergeben und in die Klassenvariablen gespeichert. Die Konstanten c_i werden auch hier einmalig berechnet. Beim Aufruf der Methode `runControl` werden unter einem übergebenen Messwert der rückgeführten Regelgröße die verschiedenen Variablen für e und u berechnet und die Stellgröße zurückgegeben. Durch die Methode `updateDesiredValue` kann auch nach der Instanzierung noch die Führungsgröße geändert werden.

2.3. Anti-Windup Maßnahmen

Durch die Beschränkung der Stellgröße einer realen Regelstrecke kann ein so genannter Wind-up-Effekt auftreten. Dabei arbeitet die Integration des Reglers weiter, ohne dass die Stellgröße zunimmt. Wird die Regelabweichung kleiner, entsteht eine ungewünschte Verzögerung der Stellgröße, die das System destabilisieren kann. [2] Um diesem integralen Windup entgegen zu wirken (Anti-Integral-Windup-Reset, kurz AIWR), wird ein zusätzlicher Übergabeparameter für einen AIWR-Schwellwert und ein neuer Satz an Konstanten c_{iI} definiert, die beim Überschreiten des Schwellwerts anstelle der vorherigen c_i für die Berechnungen verwendet werden. Die Konstanten c_{iI} entsprechen dabei den c_i für $K_I = 0$.
Bei hohem D-Anteil im Regler kann es zum Schwingen auch bei kleinen Störungen um eine konstante Regelgröße kommen. In Analogie zum integralen Windup wird ein weiterer Satz an Konstanten c_{iD} definiert, um einen 'differentiellen Windup' zu unterbinden (Anti-Differential-Windup-Reset, kurz ADWR). Dabei berechnen sich diese Konstanten genau so wie die c_i für $K_D = 0$. Sie werden zur Berechnung der Stellgröße verwendert, sollte sich das Vorzeichen der Regelabweichungen $e[k]$ und $e[k-2]$ unterscheiden. Dies würde nämlich auf ein Schwingen um die Führungsgröße 0 (Auslegung auf Linienverfolgung) hindeuten. Auch wenn dies eine unkonventionelle Erweiterung eines Regelalgorithmus darstellt, sind damit erhebliche Verbesserungen der Linienverfolgung des Roboters bei der Regelung mit hohem D-Anteil auf Geradenstrecken zu beobachten, da hier ein Aufschwingen unterbunden wird. Die Ecken können dabei unverändert durchfahren werden, da sich das Vorzeichen der Regelabweichung nicht ändern sollte.

3. Linienverfolgung

3.1. Verbesserung des Beispielprogramms

Im gegebenen Beispielprogramm werden die Lichtsensoren nur genutzt um den vorausliegenden Untergrund als schwarz, grau oder weiß zu charakterisieren. Dementsprechend werden beide Motoren mit zwei möglichen PWM-Werten versorgt. Da allerdings kein Zustand definiert ist, in dem beide Sensoren grau detektieren und beide Motoren gleich schnell drehen, kommt es zum ständigen Überschwingen des Roboters beim Geradeausfahren. Eine Fahrt um rechtwinklige Ecken ist damit auch nur schwer möglich. Zuerst wurde das Programm also um einen weiteren Zustand erweitert, in dem beide Lichtsensoren den Grauwert detektieren, so dass eine stückweise Geradeausfahrt ermöglicht wird. Auch die PWM Werte für `highPower` und `lowPower` wurden entsprechend empirischer Beobachtungen angepasst. Die Grenzen eines solchen stark diskreten Reglers mit nur drei möglichen Lichtsensorwerten als Eingang und zwei PWM-Werten als Stellgrößen sind allerdings offensichtlich. Für ein besseres Führungsverhalten auf der schwarzen Linie und 'Störverhalten' beim durchfahren der Ecken ist also ein Regler zu implementieren, der weniger stark diskretisierte Ein- und Ausgaben aufweist.

3.2. Linienverfolgung und Kurvenfahrt mit PID-Regler

Als Regelgrößen ergeben sich die Ausgaben des linken und rechten Lichtsensors im Wertebereich ganzer Zahlen zwischen 0 und 100. Diese werden zurückgeführt und eine Differenz zwischen beiden gebildet, so dass auf die Führungsgröße von 0 geregelt wird. Diese entspricht unter einer optimalen Kalibrierung der Sensoren einer exakt mittigen Position des Roboters auf der schwarzen Linie. Die Stellgröße entspricht einem PWM-Wert, der auf einen Basis-Wert beim rechten Motor aufaddiert wird beziehungsweise vom linken Motor abgezogen wird. Im späteren Verlauf der Entwicklung wird durch den Einsatz einer unterlagerten Drehzahlregelung zunächst eine rotatorische Differenzgeschwindigkeit ausgegeben, die durch kinematische Berechnungen in die Raddrehzahlen eingeht. Der eingesetzte PID-Regler wurde dabei nach Abschnitt *7.3 Handeinstellung eines PID-Reglers* der Einführungs-Anleitung zum Praktikum Regelungstechnik eingestellt. Es war zu beobachten, dass der I-Anteil des Reglers nicht bemerkbar zur Verbesserung des Führungsverhaltens beiträgt, da etwa eine stationäre Abweichung vom Sollwert auch kalibrierungsabhängig ist.
Beim Einstellen des daraus folgenden PD-Reglers ergibt sich ein Problem. Der Differenzierbeiwert muss so gewählt werden, dass ein gutes Führungsverhalten mit geringem Ausschwingen von der schwarzen Linie (Überschwingen) auf den Geradenabschnitten gewährleistet wird und gleichzeit die als Störung interpretierbaren Ecken durchfahren werden können. Deshalb ist es sinnvoll zwei verschiedene Fahrmodi zur Linienverfolgung zu imple-

mentieren. Im Control-Modus `FAST` wird mit einer hohen Sollgeschwindigkeit von 15 $\mathrm{cm\,s^{-1}}$ und geringem D-Anteil mit Differenzierbeiwert $K_D = 0.002\,\mathrm{s}$ und $K_P = 0.02$ auf langen Geraden gefahren, wobei im Control-Modus `SLOW` mit einer geringen Sollgeschwindigkeit von 7 $\mathrm{cm\,s^{-1}}$ und hohem D-Anteil mit Differenzierbeiwert $K_D = 0.06\,\mathrm{s}$ und $K_P = 0.01$ sowie aktiviertem 'differentiellem Anti-Windup' (Vgl. Erläuterungen dazu in Kapitel 2) Ecken durchfahren werden können.
Prinzipiell ergibt sich auch die Möglichkeit einer gesteuerten Durchfahrt der Ecken, also einer geregelten Linienverfolgung bis zur Ecke und Drehung um 90 Grad beim Erreichen dieser. Dabei ist allerdings eine hohe Genauigkeit der Navigation bezüglich der Positionierung notwendig sowie eine genaue Steuerung der Drehung. Für die geregelte Eckendurchfahrt genügt es, wenn die Navigation grobe Bereiche vorgibt, in denen der Roboter im `SLOW`-Modus fährt. Beziehungsweise wird es invers dazu zur weiteren Erhöhung der Zuverlässigkeit in einigen Bereichen des Spielfelds erlaubt, in den Modus `FAST` zu wechseln. Die geregelte Eckendurchfahrt zeigt auch bei Tests eine höhere Robustheit, vorallem nach dem sich Positionsfehler über mehrerer Runden aufsummieren können beziehungsweise Positionsfehlern nach erfolgten Ausparkvorgängen. Selbst nach 10 oder mehr Runden ergeben sich keine Probleme bei der Linienverfolgung.

4. Regelung der Robotergeschwindigkeit

4.1. Geschwindigkeitssteuerung

4.1.1. Kinematisches Modell der Radgeschwindigkeiten

Aus dem Gleichungssystem zum kinematischen Robotermodell im Anhang der Anleitung des Control-Moduls ergibt sich für die linke und rechte Radgeschwindigkeit folgendes:

$$v_l = v_m - \frac{\omega d}{2}$$
$$v_r = v_m + \frac{\omega d}{2}$$

Die Winkelgeschwindigkeiten der Räder ergibt sich unter dem Raddurchmesser d_R dabei aus:

$$\omega_i = \frac{2v_i}{d_R} = \frac{v_i}{d_R \pi} \text{ im Folgenden in [RPM]}$$

Um einen Zusammenhang zwischen an den Motoren anliegenden PWM-Werten und der sich dabei einstellenden Drehzahl herzustellen, wurden über den in *leJOS* vorhandenen Datenlogger für 10 PWM-Werte jeweils die Raddrehzahlen ausgewertet und gespeichert. Durch Mittelung und anschließende lineare Regression ergeben sich für die beiden Räder die folgenden Geradengleichungen:

$$\text{PWM}_l = \left(0.72762 \cdot \frac{\omega_l}{\text{RPM}} + 8.61696\right) \text{ in [\%]}$$
$$\text{PWM}_r = \left(0.77850 \cdot \frac{\omega_r}{\text{RPM}} + 8.40402\right) \text{ in [\%]}$$

Im Programm können so in der `drive`-Methode unter Vorgabe der translatorischen und rotatorischen Robotergeschwindigkeit die PWM-Werte beider Motoren berechnet werden.

4.1.2. Genauigkeit der gesteuerten Radgeschwindigkeit

Die ermittelte kinematische Steuerung des Roboters kann nun mit verschiedenen Werten für translatorische und rotatorische Geschwindigkeit getestet werden. Dabei kann die Genauigkeit der Steuerung beispielsweise durch reine Rotation des Roboters und den Versatz vom Startpunkt nach einigen Umdrehungen bestimmt werden. Anschaulicher ist allerdings die Abweichung des Roboters von einer Geraden nach einer festen Fahrtstrecke. So ergibt sich nach 10 Versuchen bei der gesteuerten Geradeausfahrt von einem Meter eine mittlere Abweichung von der Geraden von 16 cm. Diese hohe Ungenauigkeit kann durch eine zu unpräzise PWM-RPM-Regression, Schlupf der Räder oder weiteren unbekannten Störeinflussen auf der Regelstrecke hervorgerufen werden. Auch ist eine stationäre Abweichung des Roboters von der Sollgeschwindigkeit vorhanden, was auf Ungenauigkeiten des Modells und der gefundenen Parameter zurückgeführt werden kann. Es gilt deshalb durch

Messung und Rückführung der Raddrehzahlen eine anschließende Regelung zu realisieren und so die Genauigkeit zu erhöhen.

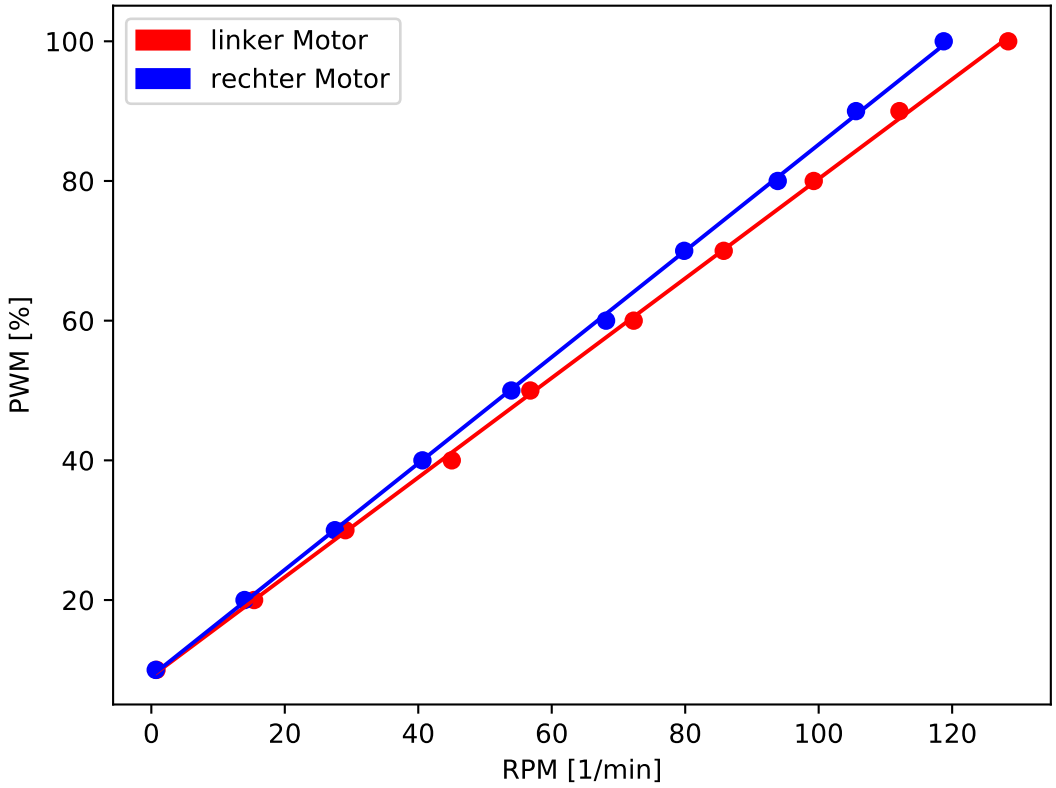

Abbildung 4.1: Mittelwerte der Drehzahlsamples mit Regressionsgerade

4.2. Geschwindigkeitsregelung

4.2.1. Implementierung der Geschwindigkeitsregelung

Die jeweils gemessene Raddrehzahl kann an einen dafür initialisierten PID-Regler übergeben werden. Dabei werden die drei Reglerbeiwerte wie beim Regler der Linienverfolgung empirisch bestimmt, allerdings werden hierbei alle Anteile des PID-Reglers verwendet. Das Testen der reinen Regelung ergibt eine relativ lange Anstiegszeit auf die konstante Führungsgröße. Durch das unterschiedliche dynamische Verhalten der Motoren kommt es auch zu Abweichungen von der Gerade, da unterschiedlich stark beschleunigt wird. Um dem entgegen zu wirken, werden die kinematischen Überlegungen der Geschwindigkeitssteuerung als Vorsteurung eingesetzt. Somit erfolgt eine deutlich schnellere Arbeitspunkteinstellung. Die Vorsteuerung ermöglicht ein gutes Führungsverhalten. Der PID-Regler kann demnach so dimensioniert werden, dass sich ein gutes Störverhalten ergibt.

4.2.2. Genauigkeit der geregelten Radgeschwindigkeit

Erneut wird die Abweichung von der Geraden auf einer Strecke von einem Meter als Gütekriterium herangezogen. Bei der Fahrt mit Regelung und Vorsteuerung ergibt sich nach Mittelung von 10 Versuchen ein Abstand von 7 cm. Dies folgt unter anderem daraus, dass die Geschwindigkeitsregelung keinen Schlupf kompensieren kann, aber auch eine systematische Ungenauigkeit der Raddrehzahlmessung vorliegt. Trotzdem beim Testen gleich viele Inkremente der Radencoder geloggt werden, dreht sich das linke Rad des Roboters etwas schneller, so dass stets ein Drift nach rechts stattfindet. Die Zusätzliche Regelung um den durch Steuerung eingestellten Arbeitspunkt führt also zu einer deutlichen Erhöhung der Genauigkeit bei geraden Strecken. Dem gegenüber weist eine reine Regelung durch die lange Anstiegszeit und den einzugehenden Kompromiss zwischen Führungs- und Störverhalten sowie eine reine Steuerung wegen stationärer Abweichungen und Modellungenauigkeiten offensichtliche Nachteile auf.

5. Regelung der Geradeausfahrt

5.1. Theoretische Betrachtungen am linearisierten Modell

Zunächst ist das im Anhang B der Anleitung gegebene kinematische Robotermodell zu linearisieren. Dabei wird als Arbeitspunkt der Ursprung $x_{10} = x_{20} = x_{30} = 0$ gewählt. Für die Geradeausfahrt ist es sinnvoll, als Winkelgeschwindigkeit $u_{20} = \omega_0 = 0$ und als translatorische Geschwindigkeit die Sollgeschwindigkeit $u_{10} = v_0$ zu wählen. Als linearisiertes Modell der Geradeausfahrt ergibt sich somit:

$$f(\underline{x}, (v, \omega)) = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{pmatrix} = \begin{pmatrix} v(t) \sin x_3 \\ v(t) \cos x_3 \\ \omega(t) \end{pmatrix} \simeq \tilde{f}(\underline{x}, (v, \omega)) = \begin{pmatrix} v_0 \cdot x_3(t) \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ v(t) - v_0 \\ \omega(t) \end{pmatrix}$$

Dabei kann die erste Zeile der Linearisierung zur Beschreibung der Querabweichung e bei Geradeausfahrt in x_2-Richtung genutzt werden:

$$\begin{aligned} e &= x_1 - x_{10} \\ \dot{e} &= \dot{x}_1 \simeq v_0 \cdot x_3(t) \\ \ddot{e} &= \ddot{x}_1 \simeq v_0 \cdot \dot{x}_3 = \underline{\underline{v_0 \cdot \omega}} \end{aligned}$$

Anhand des Modells eines Standardregelkreises kann dabei die Winkelgeschwindigkeit ω der Stellgröße und die Querabweichung e der Regelgröße zugeordnet werden. Als Übertragungsfunktion der Regelstrecke $P(s)$ ergibt sich somit im Bildbereich ein I2-Glied:

$$P(s) = \frac{E(s)}{\Omega(s)} = \frac{v_0}{s^2}$$

Um einen geeigneten Regler für diese Strecke auszuwählen, eignet sich das Wurzelortskurvenverfahren. Bei diesem wird unter Variation des Proportionalbeiwerts die Lage der sich dafür einstellenden Polstellen des geschlossenen Regelkreises betrachtet. Die Polstellen müssen für ein stabiles System einen Realteil $\operatorname{Re} s_p < 0$ aufweisen. Es sind die Wurzelortskurven für die Typen eines P-, PI-, PD- sowie PID-Reglers zu untersuchen.

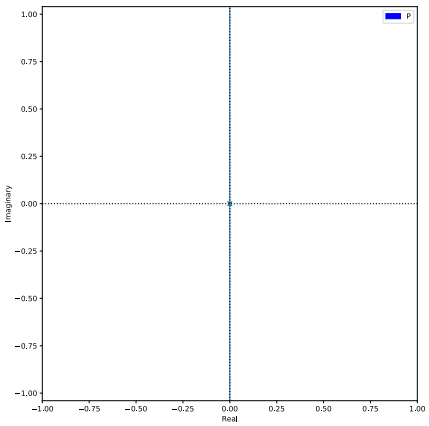

Abbildung 5.1: Wurzelortskurve der Strecke P mit P-Regler

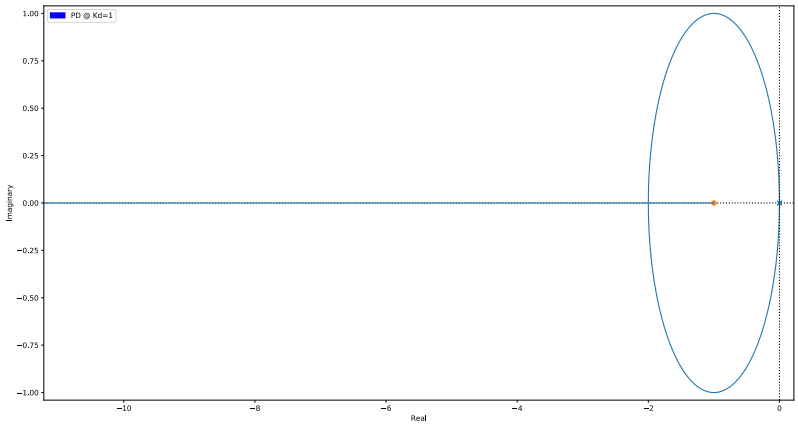

Abbildung 5.2: Wurzelortskurve der Strecke P mit PD-Regler ($K_D = 1$)

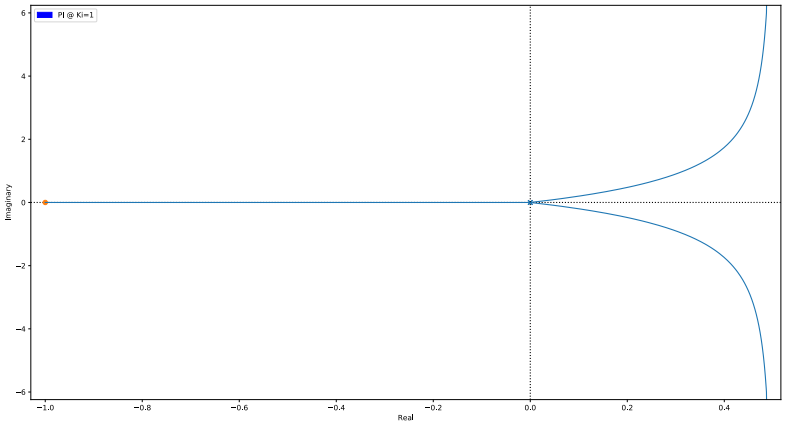

Abbildung 5.3: Wurzelortskurve der Strecke P mit PI-Regler ($K_I = 1$)

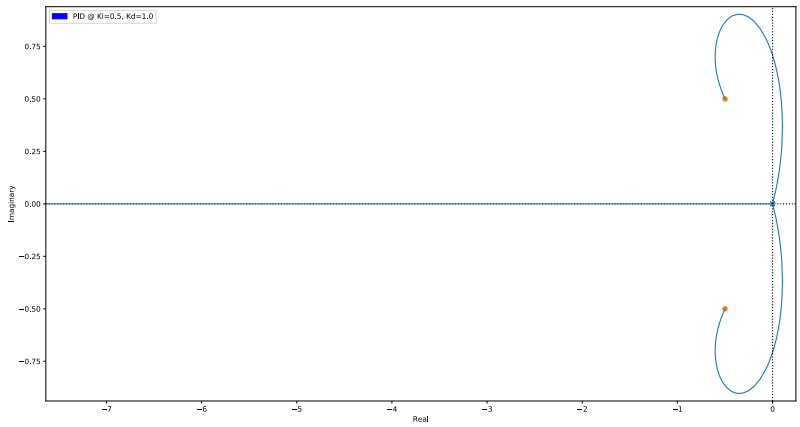

Abbildung 5.4: Wurzelortskurve der Strecke P mit PID-Regler ($K_I = 0.5$, $K_D = 1$)

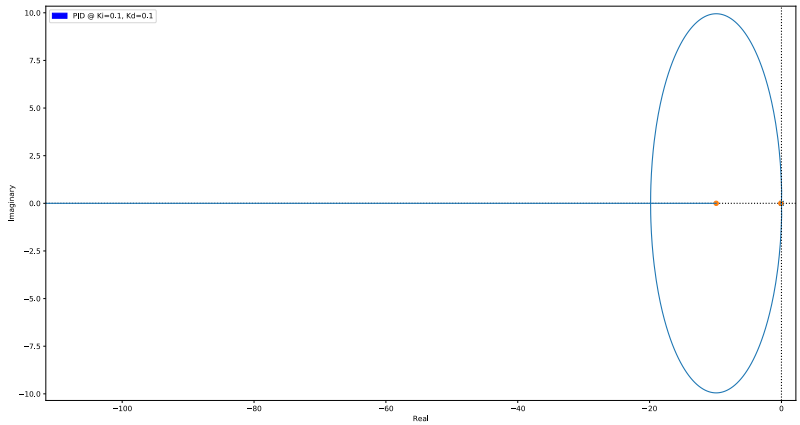

Abbildung 5.5: Wurzelortskurve der Strecke P mit PID-Regler ($K_I = 0.1$, $K_D = 0.1$)

In den Abbildungen 5.1 bis 5.5 sind diese Wurzelortskurven dargestellt. Bei einem P-Regler liegen die beiden Polstellen des geschlossenen Kreises stets auf der imaginären Achse, das System ist instabil. Bei einem PD-Regler liegen die Pole unabhängig der Wahl des Differenzierbeiwerts und der Verstärkung in der linken Halbebene, der geschlossene Kreis ist somit immer stabil. Für einen PI-Regler liegt die Verzweigungsstelle unabhängig von K_I im Koordinatenursprung. Demnach besitzen 2 Polstellen stets einen Realteil $\mathrm{Re}\, s_p \geq 0$ und der geschlossene Regelkreis ist für jede Verstärkung instabil. Abbildung Abbildung 5.5 zeigt, dass sich bei einer PID-Regelung mit günstiger Wahl von K_I und K_D ein immer stabiler Kreis bildet, wobei für ungünstige Beiwerte ein in Abbildung 5.4 ein geschlossener Regelkreis entsteht, der nicht für alle Verstärkungen stabil ist. Aus dieser Betrachtung ergibt sich, dass es sinnvoll ist einen PD-Regler in Verbindung mit der Strecke $P(s)$ zu wählen, da damit ein geringerer Einstellaufwand als beim PID-Regler sowie ein stets stabiler geschlossener Regelkreis verbunden ist.

5.2. Implementierung der Querregelung

Durch die `setDriveFor`-Methode werden zunächst alle notwendigen Daten für die bevorstehende Fahrt festgelegt. Es werden die Posendaten vor Fahrtbeginn gespeichert, die zur ständigen Überprüfung notwendig sind, ob die Zielpose erreicht wurde. Durch den Control-Modus `SETPOSE` und die damit verbundene Methode `exec_SETPOSE_ALGO` wird die gergelte Geradeausfahrt ausgeführt. Als Bedingung zur Fahrt soll beispielsweise für die x-Koordinate Folgendes gelten:

$$\operatorname{sgn}(x_0) \cdot (x - x_d) > 0.005 \text{ m}$$

Nur unter der Bedingung, dass der Abstand zwischen aktueller Koordinate x und Zielkoordinate x_d mehr als 5 mm beträgt, wird der Roboter fahren. Außerdem soll die Zielpose nicht überfahren werden, was durch den Einfluss des anfänglichen Vorzeichens von x_0 realisiert wird. Bei der Fahrt wird zunächst auf den notwendigen Winkel φ zwischen Start- und Zielpose gesteuert gedreht:

$$\varphi = \operatorname{atan2}(y_d - y, x_d - x)^1$$

Anschließend wird mit dem empirisch parametrierten PD-Regler auf die Führungsgröße φ (aus Näherung der der Querabweichung $e \propto \sin\varphi \approx \varphi$ für kleine Winkel) geregelt. Dabei wird als Stellgröße eine Winkelgeschwindigkeit ausgegeben, die zusammen mit der festgelegten translatorischen Geschwindigkeit an die unterlagerte Drehzahlregelung der `drive`-Methode übergeben wird. Damit ergibt sich eine Kaskadenregelung. Als Regelgröße wird statt der in der modellbasierten Vorbetrachtungen verwendeten Querabweichung e der Winkel der aktuellen Roboterpose genutzt. Es ist so bei Geraden, die nicht parallel zu den Koordinatenachsen liegen, mathematisch einfacher die Regelabweichung zu berechnen. Darauffolgende Tests zeigen eine Kompatibilität der theoretischen Ergebnisse mit dieser Implementierung.

5.2.1. Genauigkeit der Querregelung

Unter gleichen Bedingungen wie bei gesteuerter und geregelter Geradeausfahrt ergeben 10 Versuche einer Strecke von einem Meter eine mittlere Querabweichung von 2 cm. Damit kann eine weitere deutliche Verbesserung der Genauigkeit im Vergleich zur Fahrt unter Geschwindigkeitsregelung erreicht werden. Durch die Positionsregelung kann möglichen Störungen und kurzzeitigen Drifts anschließend entgegen gewirkt werden. Außerdem werden Ungenauigkeiten der Drehzahlmessung durch die generelle Positionsmessung auch mit geringerem Quantisierungsfehler verbessert.

Geradeausfahrt von 100 cm	v/ω-Steuerung	v/ω-Regelung	Querregelung
Querabweichung e	16 cm	7 cm	2 cm

Tabelle 5.1: Gegenüberstellung der Positionsgenauigkeit verschiedener Regelverfahren

[1] $\operatorname{atan2}(q) \in (-\pi, \pi]$ entspricht dabei dem Winkel im zweidimensionalen Polarkoordinatensystem im Gegensatz zu $\arctan(q) \in (-\frac{\pi}{2}, \frac{\pi}{2}]$

6. Geregeltes Ein- und Ausparken

Im Folgenden wird die Bahnplanung und Bahnfolgeregelung für Ein- und Ausparkvorgänge des Roboters beschrieben. Dabei liegen Vorteile einer expliziten Bahnplanung gegenüber einer Drehen-Fahren-Drehen-Sequenz in weniger Beschleunigungsvorgängen und damit einem schnelleren Abschluss des Parkvorgangs. Es kommt zu keiner direkten Fortpflanzung von Winkelfehlern auf anschließenden Geradenfahrten. In dieser Implementierung werden allerdings die Stellgrößen v und ω nicht explizit aus dem kinematischen Robotermodell berechnet. Durch eine Adaption der geregelten Geradeausfahrt bei konstanter translatorischer Geschwindigkeit wird stückweise ein erzeugtes Bahnpolynom mit kurzen Geradenstücken abgefahren, wobei nicht abgebremst wird für explizite Drehungen.

6.1. Theoretische Vorbetrachtungen der Bahnplannung

Als geometrische Randbedingungen für den abzufahrenden glatten Pfad ergeben sich für die Startposition (x_s, y_s) und Zielposition (x_z, y_z) eine horizontale Ausrichtung des Roboters in diesen Punkten durch $y'(x_s) = y'(x_z) = 0$. Die Trajektorie kann durch diese vier Randbedingungen mit einem Polynom dritten Grades dargestellt werden:

$$y(x) = ax^3 + bx^2 + cx + d$$

Es wird ein lokales Koordinatensystem durch die Translation und gegebenenfalls Drehung des absoluten Koordinatensystems in und um den Bahnmittelpunkt eingeführt. Dabei ergeben sich die Koeffizienten des dadurch ungeraden Bahnpolynoms unter Einsetzten der geometrischen Randbedingungen und Lösung des daraus folgenden linearen Gleichungssystems zu:

$$\begin{aligned} a &= \frac{y_z}{-2x_z^2} \\ c &= -3ax_z^2 \\ b &= d = 0 \end{aligned}$$

Durch die Rotation der absoluten Koordinaten am rechten Kopfende des Spielfelds und Berechnung der Funktion erst danch wird das Problem eines unendlich hohen Anstiegs des Bahnpolynoms im Start- und Zielpunkt vermieden.

6.2. Implementierung der Bahnfolgeregelung

Die Bahnplanung erflogt durch die `setParkingFor`- und `getTrajectory`-Methode im Control-Modul selbst und nicht in der Guidance. Der `setParkingFor`-Methode werden die Start- und Zielpose übergeben. Durch Addition und anschließende Mittlung der Komponenten beider Positionen wird der Mittelpunkt der zu berechnenden Bahn ermittelt. Der Zielpunkt wird um den Bahnmittelpunkt verschoben sowie um den Posenwinkel des Roboters rotiert. So können die Koeffizienten des Bahnpolynoms durch die entstandene Symmetrie günstig und unabhängig von der absoluten Lage der Parklücke berechnet werden. Daraufhin kann der Roboter in den `PARK_CTRL`-Modus versetzt werden, wodurch die `exec_PARKCTRL_ALGO`-Methode ausgeführt wird. In dieser wird analog zur geregelten Geradeausfahrt ein PD-Regler definiert und parametriert. Die Führungsgröße ist dabei ebenfalls der berechnete Zielwinkel des aktuellen Geradenstücks, von dem zur Berechnung der Regelabweichung der Winkel der aktuellen Roboterpose abgezogen wird. Der Regler gibt als Stellgröße die rotatorische Robotergeschwindigkeit aus. Auch hier werden die absoluten Koordinaten zur Berechnung der Zielpose des nächsten Geradensegments in den Bahnmittelpunkt verschoben und gegebenenfalls um den Winkel der Roboterpose rotiert, so dass das zuvor ermittelte Bahnpolynom genutzt werden kann. Stückweise wird das Bahnpolynom iterativ abgefahren. Für die aktuelle Zielposition wird die x-Koordinate als Parameter jeweils um 1 cm inkrementiert. Die y-Koordinate wird an dieser Stelle durch Einsetzen des x-Wertes in das Polynom berechnet. Über die Geradeausfahrt mit Querregelung wird nach Rücktransformation in das absolte Koordinatensystem auf den errechneten Zielpunkt geregelt.

6.3. Genauigkeit der Bahnfolgeregelung

Bei der Abfahrt von 10 Polynomen zum Ein- und Ausparken mit einer Weite von $x = 40$cm entlang der vorherigen Fahrtrichtung und $y = 25$ cm senkrecht dazu ergibt sich ein durchschnittlicher Positionsfehler von $e_x = 0,5$ cm und $e_y = 1$cm . Ein Problem stellt dabei die Synchronisation der gefahrenen Distanz und der inkrementierten Zwischenzielpose dar. Der Abstand von aktueller Pose zur Zwischenzielpose nimmt über den Parkvorgang stetig zu. Es ist durch Variation des Inkrements des x-Iterators nicht gelungen den abschließenden Winkelversatz des Roboters von der Sollpose zu korrigieren. Deshalb muss bei erreichen der Zielposition (x_z, y_z) stets auf den Winkel der Zielpose gesteuert gedreht werden.

7. Verbesserungsmöglichkeiten und Ausblick

Für die `PID`-Klasse und die vorangegangene mathematische Vorbetrachtung wäre es sinnvoller, den Proportionalbeiwert nur an den aktuellen Regelfehler zu multiplizieren. Beim parametrieren der Regler hat eine Änderung von K_P nämlich stets auch eine Auswirkung auf die Produkte der anderen Regelbeiwerte.
Bezüglich der geregelten Geradeausfahrt wäre es effizienter und weniger aufwändig, wenn nur der Fall des Fahrens parallel zu einer Koordinatenachse betrachtet wird. Bei den zu lösenden Aufgaben ist nie eine Fahrt um einen beliebigen Winkel benötigt worden. Dennoch hat sich aus der Winkelregelung die Möglichkeit einer schnellen Adaption zur Parkregelung ergeben
Eine Bahnfolgeregelung des berechneten Polynoms nach expliziter Berechnung der Stellgrößen v und ω aus dem Robotermodell könnte einen schnelleren Einparkvorgang ermöglichen. Allerdings wird aufgrund der hohen Positionsgenauigkeit und Zuverlässigkeit das bewährte Verfahren der geregelten Geradeausfahrt für diese Implementierung genutzt. Durch den geringen Aufwand der Implementierung konnte hierdurch zu einem relativ frühen Zeitpunkt des Projekts ein Parkvorgang erprobt werden und den anderen Modulen die Möglichkeit gegeben werden ihre Lösungen zu testen. Des Weiteren könnte die abzufahrende Trajektorie auch durch je ein Polynom für x- und y-Koordinate beschrieben werden, welche über einen gemeinsamen Parameter verknüpft sind, um beliebige Bahnen im absoluten Koordinatensystem abzufahren. Auf eine Rotation und Translation in lokale Koordinaten könnte dadurch verzichtet werden.
Es wäre sinnvoll einen dedizierter Mikrocontroller für die Regelung einzusetzen. Durch das Betriebssysten *leJOS* haben sich während des Testens oft nichtdeterministische Probleme gerade bei der Linienfolge ergeben. Das `Navigation`-Modul hat teilweise nur schwer reproduzierbar Threadzeiten überschritten und damit das `Control`-Modul gestört, welches auf die zeitlich äquidistante Abarbeitung des Regelkreises ausgelegt ist.

Quellenverzeichnis

[1] International Conference on Control, Engineering Information Technology (CEIT'14); Vgl. http://ipco-co.com/PET_Journal/Papers%20CEIT%2714/248.pdf S.113; Zugriff: 29.12.2019

[2] Wikipedia – Die freie Enzyklopädie; Vgl. https://de.wikipedia.org/wiki/Regler, Abschnitt: Zusammenfassung der Eigenschaften des I-Reglers; Zugriff: 09.01.2020