TECHNISCHE UNIVERSITÄT DRESDEN

IfA LS-AT

Fakultät Elektrotechnik und Informationstechnik Institut für Automatisierungstechnik

Dokumentation

zum Thema

Hauptseminar Automatisierungs-, Mess- und Regelungstechnik

vorgelegt von Max Kirchner
im Studiengang Elektrotechnik, Jg. 2017
geboren am 03.05.1998 in Räckelwitz

Betreuer: Dr.-Ing. Carsten Knoll
Dipl.-Ing. Elias Scharf
Dipl.-Ing. Julian Rahm
Dr.-Ing. Andrey Morozov
Dipl.-Ing. Mustafa Saraoglu
Verantwortlicher Hochschullehrer: Prof. Dr. techn. Klaus Janschek
Tag der Einreichung: 12.01.2020

Inhaltsverzeichnis

1 Analyse der Aufgabenstellung

1.1 Allgemeine Funktionsbeschreibung und Ziele

Das Ziel des Moduls Guidance (deutsch Führung) ist die anderen Module miteinander zu verbinden und sie zu steuern. Damit dies möglich ist, wurde mit den Modulverantwortlichen alle Funktionalitäten besprochen.

Die Hauptfunktion des Roboters ist das Abfahren eines Parcours. Dabei wird nach passenden Parklücken gesucht. Dieser Zustand heißt Scout-Modus. In eine der gefundenen Lücken soll anschließend eingeparkt werden. Mit Hilfe einer Android-App werden Steuersignale übermittelt. Wenn zum Beispiel das Ausparksignal übertragen wird, folgt der Roboter einem Polynom und wechselt anschließend in den Scout-Modus.

1.2 Geplantes Vorgehen

Um den Missionsplaner und Pfadgenerator zeit effektiv zu implementieren, wurde als erster Schritt eine theoretische Vorbetrachtung getätigt. Dabei wurde ein Zustandsdiagramm mit entsprechenden Aktionstabellen und die Berechnungsvorschrift für das Pfadpolynom entworfen. Anschließend wurde der entstandene Zustandsautomat implementiert.

1.3 Schnittstellen und Zusammenarbeit zu/mit anderen Modulen/Modulverantwortlichen

Das Projekt wurde in der objektorientierten Programmiersprache Java umgesetzt. Dies ermöglicht eine klare Modultrennung.

Mit Hilfe von Setter-Methoden ist es möglich festzulegen, welche Aktionen in den entsprechenden Modulen gemacht werden soll. Im Gegenzug kann mit den Getter-Methoden aktuelle Eigenschaften des Roboters abgefragt werden. Mit

diesen Informationen werden die jeweiligen Zustandsübergänge überprüft.

Unsere Gruppe hat sich regelmäßig getroffen. Wir haben uns gegenseitig Etappenziele gesetzt.

2 Entwurf Missionsplaner

2.1 Geforderte Funktionen

Die geforderten Funktionen können in fünf Kategorien eingeteilt werden.

i Scoutmodus mit Parklückensuche

ii Einparken

iii Parken

iv Ausparken

v Inaktiv sein

Diese Kategorien definieren die Hauptzustände.

Was der Roboter in jener Kategorie machen soll, ist in den Unterzuständen definiert.

Scoutmodus bedeutet, dass das Fahrzeug einem Parcours abfährt. Dabei wird einer Linie gefolgt. Wenn eine Ecke detektiert wird, muss die Geschwindigkeit reduziert werden. Das Ziel dabei ist, dass die Regelung genauer arbeiten kann und die Kurve abgefahren wird. Während dem Abfahren sucht der Roboter nach Parklücken.

In dem Zustand Ëinparken"wird als erstes die Startpose angefahren. Danach wird ein Polynom abgefahren. Wenn die Zielpose in der Lücke erreicht ist, wechselt der Automat automatisch in den Zustand Park. In diesem wird gewartet, bis über die App das Ausparksignal übermittelt wird.

Beim Ausparken folgt der Apparat erneut einem Polynom zurück auf die Linie. Dabei muss sichergestellt werden, dass nach vorne genügen Freiraum vorhanden ist, sonst gibt es einen Unfall.

2.2 Hauptzustandsmaschine

-Orientierung an den geforderten Funktionen
-Aufteilung so, dass die Zustandsübergänge möglichst einfach sind

Zustand	Eingangsaktion	Nominalaktion	Ausgangsaktion
Driving	Wahl des richtigen Controlmodus	siehe Unterzustands-maschine	Parklückensuche ausschalten
	Parklückensuche einschalten		
Inactive	Auswahl Controlm-odus INACTIVE	siehe Unterzustands-maschine	
Exit		Abschalten des Sys-tems	
Park	Auswahl Controlm-odus INACTIVE		
Park This	Überprüfung ob die Anfahrt schon erfolgte	siehe Unterzustands-maschine	
	Parkplatzinformationen abfragen		
Park Out	Auswahl des Richti-gen Zustands	siehe Unterzustands-maschine	

2.3 Unterzustandsmaschine

2.3.1 Scout

Zustand	Eingangsaktion	Nominalaktion	Ausgangsaktion
Slow	Controlmodusauswahl langsam		
Fast	Controlmodusauswahl schnell		

2.3.2 Pause

Es gibt keine Unterzustandsmaschine.

2.3.3 Exit

Es gibt keine Unterzustandsmaschine.

2.3.4 Park

Es gibt keine Unterzustandsmaschine.

2.3.5 Park This

Zustand	Eingangsaktion	Nominalaktion	Ausgangsaktion
To Slot	Anfahrort festlegen	DRIVING bis Pose erreicht ist	
Reached Slot	Start- und Endpose vom Polynom festlegen		
	Parkcontrolmudus festlegen	Überprüfung ob Polynom abgefahren wurde	
In Slot			
Correct			

2.3.6 Park Out

Zustand	Eingangsaktion	Nominalaktion	Ausgangsaktion
Backwarts	Überprüfung ob genügend Platz vorhanden ist	Überprüfung ob das Zurückfahren erfolgt ist	
ParkOut	Start- und Endpose vom Polynom festlegen		
	Parkcontrolmodus festlegen	Überprüfung ob Polynom abgefahren wurde	

3 Entwurf Pfadgenerator

3.1 Geforderte Funktionen

Damit der Roboter einparken kann, soll er einem Polynom folgen. Dabei ist zu beachten, dass für jede Parklücke individuell ein Polynom entwickelt werden muss.

In dem Modul Guidance wird die Start- und Endpose festgelegt. Anschließend muss der entsprechende Controlmodus gestartet werden. Die Berechnung des Polynoms wird danach im Control-Modul aufgerufen.

3.2 Mathematische Beschreibung

Eine einfache und gute Einparkkurve entspricht einer Funktion dritten Grades:

$$f = y(x) = a * x^3 + c * x^2 + b * x + d \tag{3.1}$$

Die Funktion soll durch den Koordinatenursprung gehen und punktsymmetrisch sein. Das Problem vereinfacht sich auf folgendes Polynom:

$$f = y(x) = a * x^3 + b * x \tag{3.2}$$

Es werden zwei Bedingungen benötigt, damit die Koeffizienten berechnet werden können.

$$y'_{ende} = f'(x_{ende}) = 0 \tag{3.3}$$

$$y_{ende} = a * x^3_{ende} + b * x_{ende} \tag{3.4}$$

Es folgt:

$$a = -\frac{y_{ende}}{2 * x_{ende}^3} \tag{3.5}$$

und

$$b = -3 * a * x_{ende}^3 \tag{3.6}$$

Damit die Brechung stets auf das selbe Problem zurückzuführen ist, gibt es einen globales und lokales Koordinatensystem. Wie in der Abbildung 3.1 zu sehen, ist der Mittelpunkt zwischen Start- und Zielpose im globalen Koordinatensystem identisch mit dem Ursprung des lokalem Systems. Dieses ist je nach Richtungswinkel des Roboters entsprechend gedreht.

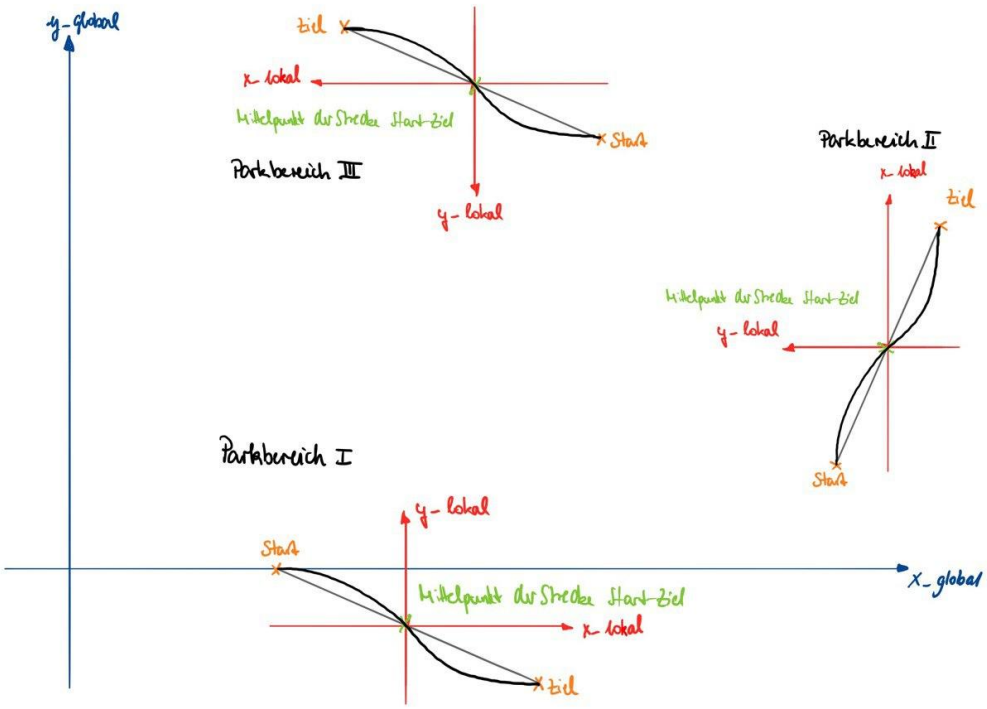

Abbildung 3.1: Skizze für die Parklückenberechung

4 Implementierung

4.1 Missionsplaner

Ausgangspunkt für die Implementierung ist das gegebene Beispielprogramm. In diesem wurde die Hauptzustandsmaschine mit drei Grundzuständen (DRIVE, INACTIVE und EXIT) vorgegeben. Mit Hilfe von Switch-Case-Abfragen wird nur der Code des entsprechenden Zustandes ausgeführt. Nach der Abfrage wird überprüft, ob in einem neuen Zustand gewechselt werden muss.

Die Grundversion verfügt noch über keine Unterzustandsmaschinen. Es fehlen auch die wesentlichen Zustände für das Ein- und Ausparken. Der getätigte Entwurf des Missionsplaners wurde mit weiter Switch-Case-Abfragen implementiert.

4.2 Pfadgenerator

Zusammen mit dem Controlmodul wurde entschieden, dass im Guidancemodul die Start- und Zielpose, die Geschwindigkeit übergeben wird und zum Schluss der richtige Controlmodus aktiviert wird. Alle Berechnungen werden im Controlmodul getätigt.

Damit das Polynom richtig berechnet wird und anschließend dieses abgefahren werden kann, muss zwischen dem lokalem und globalem Koordinatensystem transformiert werden.

```
// transform coordinates
if(this.destination.getHeading() == 0 ) {
        // do nothing
}else if(Math.abs(this.destination.getHeading() - Math.PI/2) <
    0.0001) {
        double n = x_local;
        x_local = y_local;
        y_local = -n;
}else if(Math.abs(this.destination.getHeading() - Math.PI) <
    0.0001) {
        x_local = -x_local;
        y_local = -y_local;
}
```

und

```
// back transformation
if(this.destination.getHeading() == 0 ) {
        // do nothing
}else if(Math.abs(this.destination.getHeading() - Math.PI/2) <
    0.0001) {
        double n = x_local;
        x_local = -y_local;
        y_local = n;
}else if(Math.abs(this.destination.getHeading() - Math.PI) <
    0.0001) {
        x_local = -x_local;
        y_local = -y_local;
}
```

5 Anmerkungen und Verbesserungsmöglichkeiten

- Programmierung mit Fachsprachen IEC 1131 z. B. Indistrielle Steuerungen (SPS, PLS, PC) -Erweiterung der Module, sodass es auf anderen Strecken funktioniert

Selbstständigkeitserklärung

Hiermit versichere ich, Max Kirchner, geboren am 03.05.1998 in Räckelwitz, dass ich die vorliegende Dokumentation zum Thema

Hauptseminar Automatisierungs-, Mess- und Regelungstechnik

ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Bei der Auswahl und Auswertung des Materials sowie bei der Herstellung des Manuskripts habe ich Unterstützungsleistungen von folgenden Personen erhalten:

Dr.-Ing. Carsten Knoll, Dipl.-Ing. Elias Scharf, Dipl.-Ing. Julian Rahm, Dr.-Ing. Andrey Morozov, Dipl.-Ing. Mustafa Saraoglu

Weitere Personen waren an der geistigen Herstellung der vorliegenden Dokumentation nicht beteiligt. Mir ist bekannt, dass die Nichteinhaltung dieser Erklärung zum nachträglichen Entzug des Diplomabschlusses (Masterabschlusses) führen kann.

Dresden, den 12.01.2020 .....

Unterschrift