7. Übungszettel Robotik WS15/16

Prof. Daniel Göhring, Zahra Boroujeni
Institut für Informatik, Freie Universität Berlin
Abgabe online bis Dienstag, 08.12.2015, 12 Uhr s.t.

Fassen Sie Ihre Ergebnisse (Bilder und Beschreibung der Ergebnisse) in einer PDF-Datei zusammen und benennen Sie diese "RO-07-<Nachnamen der Studenten>.pdf". Quellcode soll *nicht* im PDF erscheinen.

1. Aufgabe (10 Punkte): Control (Programmieraufgabe)

Installieren Sie das Ackermann-Vehicle-Package:

sudo apt-get install ros-indigo-ackermann-vehicle*
sudo apt-get install ros-indigo-fake-localization

Installieren Sie danach folgendes Repository:
https://github.com/ZahraBoroujeni/ackermann_vehicle

nach ……/catkin_ws/src

Danach ausführen von catkin_make unter catkin_ws/

Führen Sie
roslaunch ackermann_vehicle_gazebo ackermann_vehicle_.launch , ein Fahrzeug sollte im Simulator angezeigt werden

in topic /ackermann_vehicle/odom (nav_msgs/Odometry) kann man die Position des Fahrzeugs sehen (x,y,z sowie Quarternion)

a. Programmieren Sie einen Regler, der beim gegebenen Fahrzeug und bei einer Geschwindigkeit von 10 m/s (36 km/h) die y-Koordinate auf einen bestimmten Wert (z.B. 0) fixiert. Die Regelfrequenz soll 10 Hz betragen.

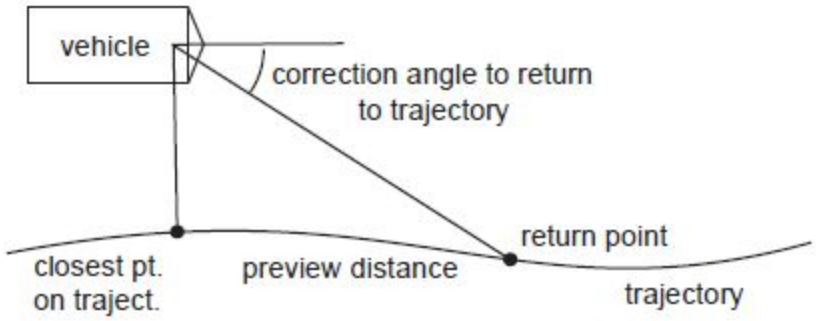
Verwenden Sie dazu einen P sowie einen PD Regler.

b. Ermitteln Sie zudem den Wert für P, bei dem das System (die y-Position des Fahrzeugs) anfängt, zu oszillieren.

Berechnen Sie daraus die Gewichte für einen PID-Regler nach der Ziegler-Nichols-Methode. Fahren Sie nun die Trajektorie noch einmal mit den ermittelten Werten für PID ab.

Plotten Sie für alle 3 Regler (P, PD, PID) aussagekräftige Trajektorien (x,y - Koordinate), z.B. über 100 Regelungsschritte.

Hinweis: Insbesondere bei Ackermann-Fahrzeugen ist es oft sinnvoll, nicht die eigenen Position (x-Koordinate)als Soll- und Istgröße zu verwenden, sondern den Winkel zur Wunschtrajektorie (bei einer bestimmten Vorausschau von z.B. 10 Metern) - siehe Abbildung.



Die Regelkommandos können über folgenden Codeschnipsel an das Fahrzeug kommuniziert werden:

```
ros::Publisher chatter_pub = n.advertise<ackermann_msgs::AckermannDriveStamped
>("ackermann_vehicle/ackermann_cmd",1000);
ackermann_msgs::AckermannDriveStamped msg;
    msg.header.stamp = ros::Time::now();
    msg.drive.steering_angle=0;
    msg.drive.steering_angle_velocity=0;
    msg.drive.speed=argc;
    msg.drive.acceleration=0;
    msg.drive.jerk=0;
    chatter_pub.publish(msg);
```