Arbeit zum
Praktikum Mess- und Regelungstechnik
Sommersemester 2022

Simon Klüpfel, Lukas Zeller
Robotik und Telematik
Universität Würzburg
Am Hubland, D-97074 Würzburg
lukas.zeller@stud.uni-wuerzburg.de
simon.kluepfel@stud.uni-wuerzburg.de

Würzburg, 24.08.2022

1 Einleitung zum Volksbot-Roboter

In dieser Arbeit befassen wir uns mit dem Robot Operating System, kurz *ROS*, und der Verwendung dessen auf einem einfachen Roboter, dem *Volksbot*.
Der verwendete Roboter ist der *RT3-2* mit zwei passiven Rädern und der *RT-3* mit einem passiven Rad. Entwickelt wurden diese vom *Fraunhofer Institut IAIS* aus Sankt Augustin.
Die technischen Daten sind wie folgt:

Abmessungen	580x520x315mm (L x B x H)
Gewicht	17kg
Raddurchmesser	260x85mm (aktive Räder)
	200mm (passive Räder)
Maximale Geschwindigkeit	2,2 $\frac{m}{s}$
Maximale Zuladung	25kg

Auszug aus https://www.volksbot.de/rt3-de.php

Auf dem Roboter ist ein Laserscanner sowie Hardware zur Verbindung mit dem verwendeten Laptop installiert. Außerdem lässt sich der Roboter über einen Joystick, ähnlich eines Gamecontrollers, manuell steuern. Wir verwenden sowohl vorgegebene ROS-Nodes, die uns von Institut für Robotik und Telematik zur Verfügung gestellt wurden, als auch angepasste Nodes die wir selbst erstellt beziehungsweise geändert haben.

Abbildung 1: Volksbot RT3 und Logitech G710-Gamecontroller

2 Kartierung mit GMapping

Wir starten zuerst die Node zur Kommunikation mit dem Roboter:

```
roslaunch volksbot messtechnikpraktikum.launch
```

Um den Roboter zu steuern mit dem Controller benötigen wir die zugehörige Node:

```
roslaunch volksbot localjoystick.launch
```

Nun erstellen wir eine *rosbag*, die alle Topics des Roboters aufzeichnet:

```
rosbag record -a
```

Um nun eine Karte erstellen zu können, müssen wir die aufgenommene *bag* abspielen. Wichtig hier ist es den Parameter

```
rosparam set use_sim_time true
```

zu setzen um die Zeit aus dem *clock*-Topic zu nutzen anstelle der globalen Systemzeit. Nun kann man aus den Laserscanner-Daten, also hauptsächlich dem *LMS*-Topic, die Map erstellen. Hierzu starten wir das GMapping-Tool via

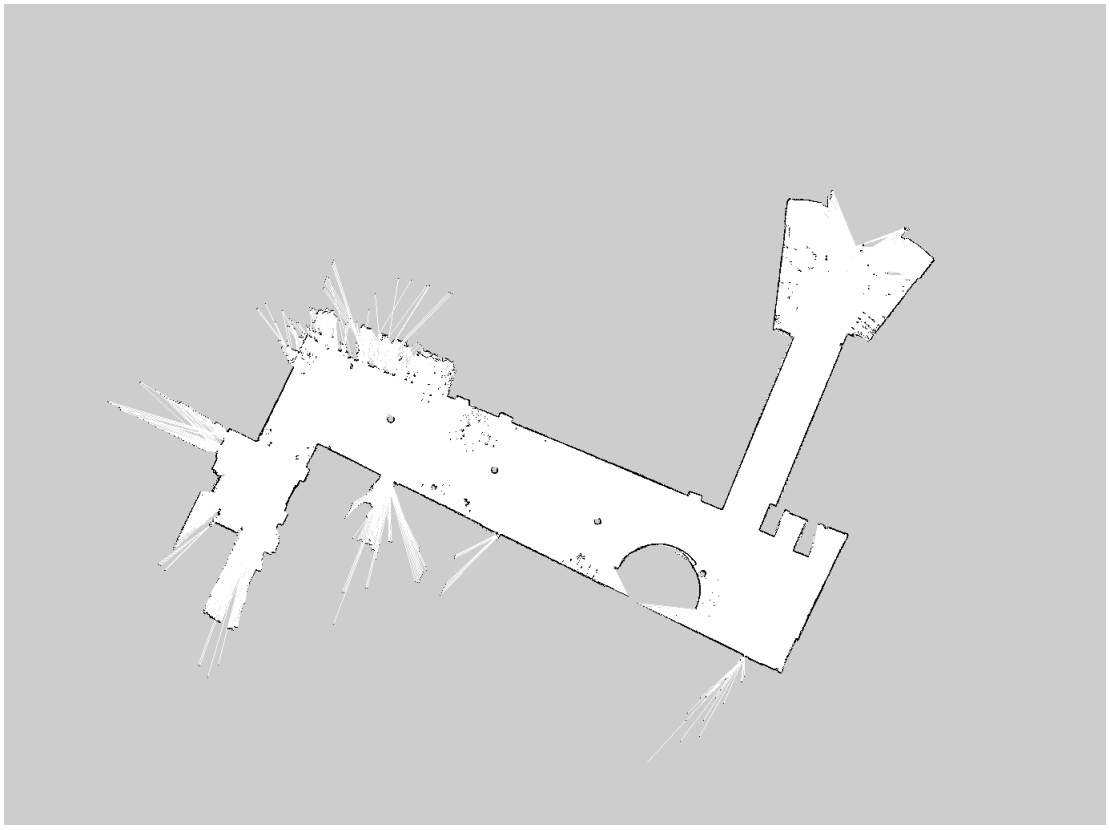
```
roslaunch volksbot messtechnikgmapping.launch
```

Die Bag kann dann abgespielt werden mit

```
rosbag play filename.bag --clock
```

Der *clock*-Befehl am Ende ist wichtig um die Aufnahmen der *rosbag* mit einem Zeitstempel zu versehen damit der Roboter sie nicht verwirft. Startet man nun RViz kann man über das *map*-Topic den Aufbau der Map betrachten. Um diese zu speichern startet man eine *ROS-node* die das *map*-Topic in eine Datei speichert.

Abbildung 2: Aufgezeichnete Karte des Informatik-Untergeschosses



3 AMCL-Lokalisierung

Um aus der Odometrie einen Pfad zu plotten erstellen wir ein eigenes ROS-Node. Das Node-Package heißt `data_saver`, die Node an sich `listener`. Es ist Subscriber des Odometrie-Topics und speichert die empfangenen Daten in einer Datei im Format `Xpos_Ypos`
Es hat zwei Modi, wie es diese Daten speichern kann:

`rel`: Koordinaten im Pfad sind relativ zum Roboter
`abs`: Koordinaten sind absolut aus dem AMCL-Frame

Man starte sie mit dem gewählten Modus:

```
rosrun data_saver listener_mode := "abs || rel"
```

Die ROS-Node speichert dann AMCL- und Odometrie-Daten in separaten Dateien.

Um AMCL zu starten nutzen wir die vorgegebene Launchfile:

```
roslaunch volksbot messtechnikamcl.launch
```

Bei Betrachten des launchfile-Codes fällt auf dass zusammen mit dem AMCL-Node ein `map_server`-Node gestartet wird, welches eine vorgegebene Map published. Wir wollen aber unsere eigene Map verwenden, weswegen wir diese Zeile auskommentieren und stattdessen eine eigene `map_server`-Node starten, welches die Map aus *GMapping* published:

```
rosrun map_server map_server map.yaml
```

Außerdem muss im Configuration-File von AMCL noch eingefügt werden, dass AMCL auf ein Map-Topic subscriben soll, und wie dieses Topic heißen soll.
In RViz muss man nun noch die Position des Roboters festlegen. Dazu setzt man die Map und das `amcl_pose`-Topic sichtbar, in unserem Fall mittels einer .config-File, die man über

```
rviz -d config.rviz
```

ausführt. In RViz drückt man nun den `2D-Pose-Estimate`-Button, dann an die Stelle der Karte wo der Roboter am Anfang steht, hält dabei gedrückt, und bewegt die Maus in die Richtung in die die Roboterlängsachse zeigt. Zum Bestätigen lässt man den Mausbutton los.
Um die `rosbag`-Datei nun abspielen zu können, führe man folgenden Befehl im Terminal aus:

```
rosbag play file.bag --clock
```

Der Roboter bewegt sich nun in RViz virtuell auf der Karte und lokalisiert sich dabei mit AMCL.

4 Pfadverfolgung

Pfade vergleichen, was gut, was schlecht, eventuell Code-Kommentare

5 Auswertung

todo: Testen Sie den Regler mit dem aufgenommenen Pfad aus dem Informatikgebäude. Vergleichen Sie das Verhalten, wenn einmal die Odometrie und einmal die von AMCL bestimmten Positionen als Messwerte für die Regelung verwendet werden.

also: plots, kommentare, was ist besser, was schlechter?