Anleitung

Hinweise

$ - Terminalbefehl

kursiv - Datei(pfad)

Für eine ausführliche Erklärung des Codes bitte das Dokument „RobotOnALine_Doku.pdf“ lesen

Tutorialvideo: https://youtu.be/heueVuk0v90

Vorbereitung

1. Ubuntu 18.04.4 LTS oder neuer muss installiert sein (https://www.linuxtechi.com/ubuntu-18-04-lts-desktop-installation-guide-screenshots/)
2. ROS-melodic oder neuer muss installiert sein (http://wiki.ros.org/melodic/Installation/Ubuntu)
3. Es ist von Vorteil sich bereits mit ROS, ROS Befehlen und seiner Ordnerstruktur auszukennen!
4. Erstelle einen Catkin-Workspace (http://wiki.ros.org/catkin/Tutorials/create_a_workspace)
    1. In Terminal folgende Befehle eingeben:
        $ mkdir -p ~/catkin_ws/src
        $ cd ~/catkin_ws/
        $ catkin_make
    2. Source-Befehle in der .bashrc speichern, damit sie nicht bei jedem Öffnen eines neuen Konsolen-Fensters erneut eingegeben werden müssen:
        1. neues Terminal öffnen

            $gedit ~/.bashrc
        2. folgende Zeilen am Ende der Datei einfügen und speichern:

            source /opt/ros/melodic/setup.bash
            source ~/catkin_ws/devel/setup.bash

Git Repositories clonen

5. Repo in den Ordner *PfadZumWorkspace/catkin_ws/src* clonen
    1. über Terminal dort hin navigieren

        $git clone https://github.com/Lizzylizard/3PiSimulation.git
    2. Turtlebot3-Git-Ordner in den Ordner *PfadZumWorkspace/catkin_ws/src* clonen
        1. diesem Tutorial folgen:

1. https://automaticaddison.com/how-to-launch-the-turtlebot3-simulation-with-ros/

2. oder diese Git-Repos clonen:

    1. $git clone https://github.com/ROBOTIS-GIT/turtlebot3_msgs.git
    2. $git clone https://github.com/ROBOTIS-GIT/turtlebot3.git
    3. $git clone https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git

Repositories ausführbar machen

6. Plugin für eigenen Message-Typ builden

    1. in ~/.bashrc (*$gedit ~/.bashrc*) folgende Zeilen einfügen:

export GAZEBO_PLUGIN_PATH=${GAZEBO_PLUGIN_PATH}:~/catkin_ws/src/3PiSimulation/my_msgs/build

    2. in den Ordner *PfadZumWorkspace/catkin_ws/src/3PiSimulation/my_msgs* navigieren
    3. einen Ordner mit dem Namen „build“ erstellen (*$mkdir build*)
    4. in diesen Ordner navigieren (*$cd build* oder Doppelklick auf den Ordner, dann Rechtsklick und „In Terminal öffnen“ wählen)
    5. das Plugin mit folgenden Befehlen builden:

        $cmake ../
        $make

7. Plugin für eigene Steuerung builden

    1. in ~/.bashrc (*$gedit ~/.bashrc*) folgende Zeilen einfügen:

export GAZEBO_PLUGIN_PATH=${GAZEBO_PLUGIN_PATH}:~/catkin_ws/src/ROS_Packages/plugins/vel_joint_motors/build/

    2. in den Ordner *PfadZumWorkspace/catkin_ws/src/3PiSimulation/plugins/vel_joint_motors* navigieren
    3. die Datei CMakeLists.txt mit einem Editor öffnen

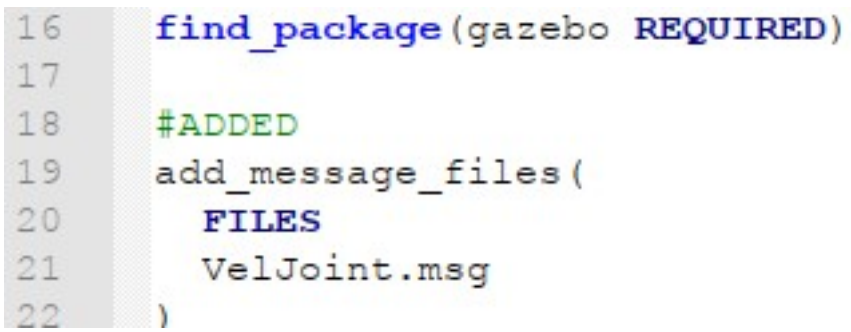

Abbildung 1: CMakeLists.txt

    4. in add_message_files (zwischen Zeile 19 und 20, siehe Abbildung 1) folgende Zeile einfügen:

        DIRECTORY /home/DEINNAME/catkin_ws/src/3PiSimulation/my_msgs/msg

1. statt DEINNAME den Benutzernamen des Linuxsystems einfügen

5. einen Ordner mit dem Namen „build“ erstellen (*$mkdir build*)
6. in diesen Ordner navigieren (*$cd build* oder Doppelklick auf den Ordner, dann Rechtsklick und „In Terminal öffnen“ wählen)
7. das Plugin mit folgenden Befehlen builden:

   $cmake ../
   $make

8. den Catkin-Workspace erneut builden
   1. in den Ordner *PfadZumWorkspace/catkin_ws* navigieren und folgenden Befehl ausführen:

      $catkin_make

Simulation starten

9. die Simulation starten
   1. Roscore starten
      1. neues Terminal Fenster
      2. *$roscore*
   2. 3Pi-Node starten
      1. neues Terminal Fenster
      2. *$roslaunch three_pi_description three_pi_race_city.launch*
   3. Driving-node starten
      1. neues Terminal Fenster
      2. *$rosrun drive_three_pi drive_on_line.py*
      3. alternativ kann dieser Befehl auch zum launch-File des 3Pi Roboter hinzugefügt werden, damit der Roboter sofort losfährt
         1. in *PfadZumWorkspace/catkin_ws/3PiSimulation/descriptions/three_pi_description/launch* navigieren
         2. three_pi_race_city.launch mit beliebigem Editor öffnen
         3. folgende Zeile einkommentieren:

<!--<node pkg="drive_three_pi" type="drive_on_line.py" name="drive_on_line" />-->

         4. dann muss Punkt 9.3. in Zukunft nicht mehr extra ausgeführt werden

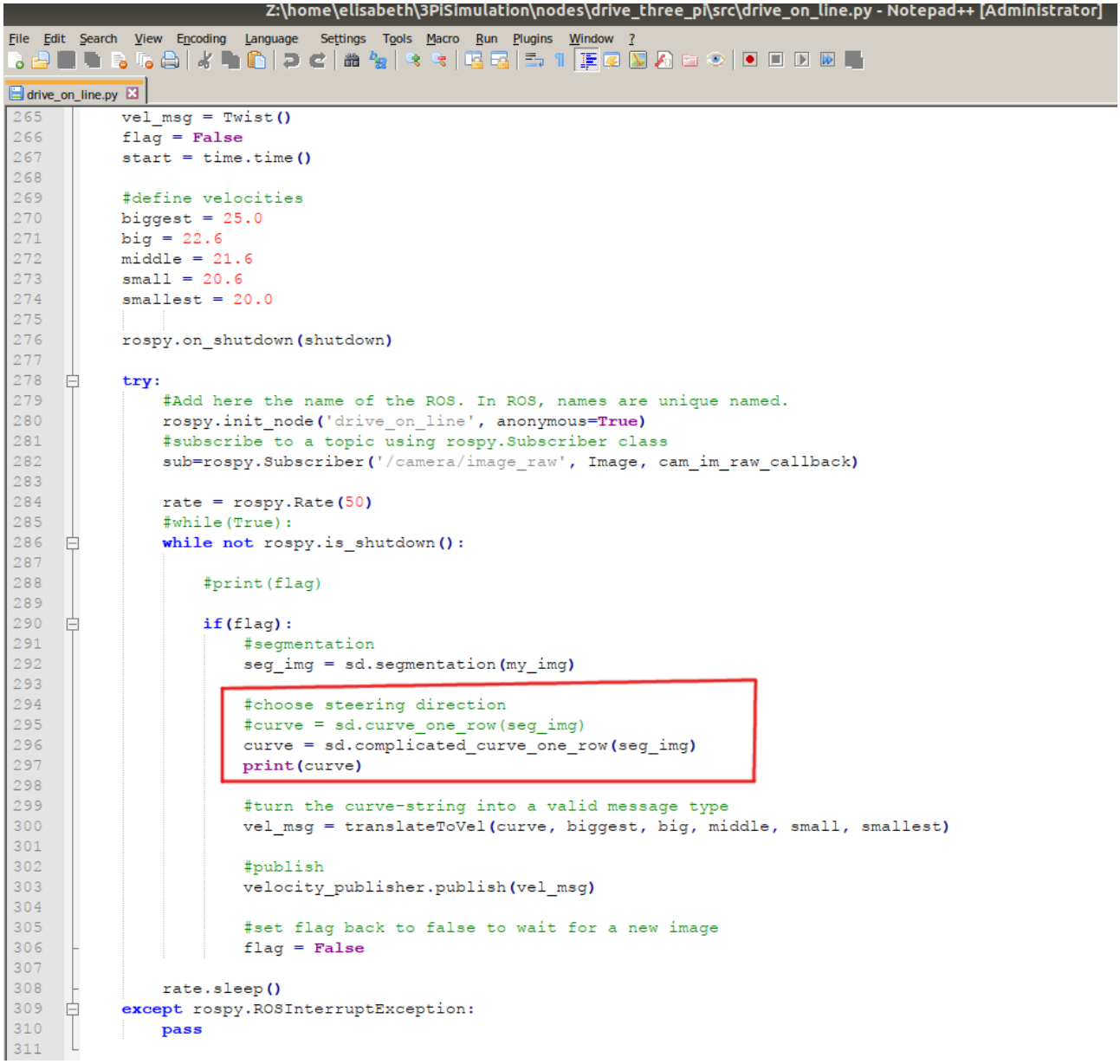

Abbildung 2: Verschiedene Algorithmen

Da verschiedene Algorithmen existieren, kann durch Ein- bzw. Auskommentieren der rot markierten Zeilen der jeweils andere Algorithmus verwendet werden. Danach einfach den Node neu starten.