Reinforcement Learning with a simulated robot via Gazebo

Elisabeth Milde

June 17, 2020

Contents

1 Abstract **3**

2 Introduction to Gazebo and ROS **4**

2.1 Set up for Linux Ubuntu 18.04.4 Bionic Beaver System 4

2.1.1 Requirements 4

2.1.2 Creating a Catkin Workspace 4

2.1.3 Clone Git Repository 5

2.1.4 Build the necessary plugins 5

2.1.5 Start simulation 6

2.2 Quick overview of ROS 7

3 The code explained **8**

3.1 The main program 8

3.2 The image processing class 8

3.3 The robot class: Reward and Q-Matrix 8

4 Running the code **9**

5 Results **10**

6 Summary **11**

1 Abstract

In this paper we are going to learn how to set up your own simulated Robot using ROS and Gazebo, how to create a simple reinforcement learning algorithm using a matrix and how to run and test the code.

2 Introduction to Gazebo and ROS

2.1 Set up for Linux Ubuntu 18.04.4 Bionic Beaver System

2.1.1 Requirements

Please make sure that you have installed the required software.

1. Ubuntu 18.04.4 LTS Bionic Beaver or higher
   (a) You can find the installation guide here (https://www.linuxtechi.com/ubuntu-18-04-lts-desktop-installation-guide-screenshots/)
2. ROS melodic or higher
   (a) Installation guide for ROS melodic (http://wiki.ros.org/melodic/Installation/Ubuntu)

2.1.2 Creating a Catkin Workspace

Follow this (http://wiki.ros.org/catkin/Tutorials/create_a_workspace) tutorial or do the following:

1. Run the following commands in your command shell:

```
$ mkdir -p ~/catkin_ws/src
$ cd ~/catkin_ws/
$ catkin_make
```

   You can create your workspace anywhere on your harddrive by replacing "~"by the path to your local disc space, but notice that in the following it will be assumed that you used the displayed path for your workspace.

2. Source commands to your .bashrc
   (a) In your command shell run

```
$ gedit ~/.bashrc
```

   (b) Paste the following lines to the end of the newly opened file and save it

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

2.1.3 Clone Git Repository

1. Move into the folder *~/catkin_ws/src*

2. Run the following command in your command shell

   ```
   $ git clone https://github.com/Lizzylizard/
       ReinforcementMatrix.git
   ```

3. Run the following commands in your command shell or follow this (`https://automaticaddison.com/how-to-launch-the-turtlebot3-simulation-with-`

   ```
   $ git clone https://github.com/ROBOTIS–GIT/
       turtlebot3_msgs.git
   $ git clone https://github.com/ROBOTIS–GIT/
       turtlebot3.git
   $ git clone https://github.com/ROBOTIS–GIT/
       turtlebot3_simulations.git
   ```

2.1.4 Build the necessary plugins

1. Message Type *vel_msg*

   (a) Paste the following line to the end of your .bashrc (see Section 2):

   ```
   export GAZEBO_PLUGIN_PATH=${
       GAZEBO_PLUGIN_PATH}:~/catkin_ws/src/
       ReinforcementMatrix/my_msgs/build
   ```

   (b) Navigate into the folder *~/catkin_ws/src/ReinforcementMatrix/my_msgs*

   (c) Create a folder named *build*

   ```
   $ mkdir build
   ```

   (d) Move into this folder and run the following commands:

   ```
   $ cmake ../
   $ make
   ```

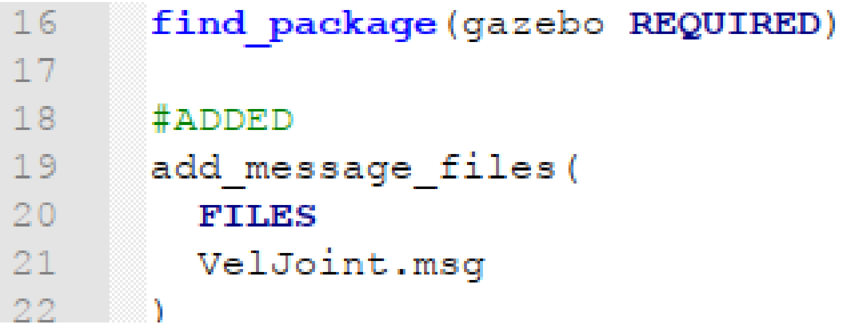
2. Controls plugin

   (a) Paste the following line to the end of your .bashrc (see Section 2):

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

(b) Move into the folder *~/catkin_ws/src/ReinforcementMatrix/plugins/vel_joint_motors*

(c) Open up the file *CMakeLists.txt* with a text editor



```
16  find_package(gazebo REQUIRED)
17
18  #ADDED
19  add_message_files(
20    FILES
21    VelJoint.msg
22  )
```

Figure 1: CMakeLists.txt

(d) Add the following line between line 19 and 20 (see Figure 1):

```
DIRECTORY /home/YOURNAME/catkin_ws/src/
    ReinforcementMatrix/my_msgs/msg
```

i. Replace *YOURNAME* by the username of your linux system

(e) Redo the steps 1b to 1d inside the folder *~/catkin_ws/src/ReinforcementMatrix/plugins/vel_joint_motors*

3. Rebuild the catkin workspace

(a) Navigate to *~/catkin_ws*

(b) Run

```
$ catkin_make
```

2.1.5 Start simulation

1. In a new shell window run:

```
$ roscore
```

2. In a second shell window run:

```
$ roslaunch three_pi_description
    three_pi_race_city.launch
```

3. In yet another shell window run:

   ```
   $ rosrun drive_three_pi reinf_matrix_4.py
   ```

   (a) As an alternative you can add this last command to the launch file of the robot having the effect of not having to run the command 3 in the future anymore

      i. Move to *~/catkin_ws/ReinforcementMatrix/descriptions/three_pi_description/launch*

      ii. Add the following line

      ```
      <node pkg="drive_three_pi" type="
          reinf_matrix_4.py" name="reinf_matrix_4
          " />
      ```

This will result in the robot starting to learn how to properly follow the line on the ground of the racetrack. It will do so in a finite number of steps which can be adjusted as we wish. We will discuss this later in Section **??**. It will also produce a lot of output. It might be helpful to redirect the output into an external text file to have a better look at what happens during the learning algorithm. This can be done by using the following command when starting the node:

```
$ rosrun drive_three_pi reinf_matrix_4.py >>
    Schreibtisch/Output.txt
```

2.2 Quick overview of ROS

3 The code explained

3.1 The main program

3.2 The image processing class

3.3 The robot class: Reward and Q-Matrix

4 Running the code

5 Results

6 Summary