Project Intelligent Robotics
Assignment #2

Task 2.1 Simulation Update: This assignment sheet is meant for use with the real physical robots. As you do not have access to them, most of the tasks can not be implemented as described. Instead, please read through the rest of the assignment sheet first without working on them and discuss changes with the advisors.

Next, install support for a TurtleBot2 simulation in your ROS workspace. To do so, first download the turtlebot.rosinstall file from `https://tams.informatik.uni-hamburg.de/lectures/2020ss/projekt/ir/doc/turtlebot.rosinstall` and save it locally in your `$HOME/ros/src/` folder as `.rosinstall` (note the leading period to mark the file as hidden). Afterwards install and use the `wstool` utility to download and manage the repositories listed in the file as part of your ROS workspace:

```
sudo apt install python-wstool
cd $HOME/ros/src
wstool update
```

Lastly, rebuild your workspace (`catkin build` in your `$HOME/ros` folder) and you should have a working turtlebot simulation available.

In order to start and interact with it, the following commands can be used:

```
roslaunch turtlebot_gazebo turtlebot_world.launch
roslaunch turtlebot_teleop keyboard_teleop.launch
roslaunch turtlebot_rviz_launchers view_robot.launch
```

NOTE: In this assignment, you are introduced to the robotic platform “turtlebot” which you will program in the rest of this course. You will get to know its sensors and their data visualization. In a second step you will write a program to move the Turtlebot and use the sensor data to stop before it collides with obstacles in the environment.

Real robots require to be handled with care.
Be prepared to rescue the robot from hitting obstacles.

Task 2.2 Starting the Turtle and ROS: You can login to the robot via SSH with the username `project2018` on the hosts `donny`, `leo`, `mikey`, and `raph` from any TAMS pool computer. You will find a pre-built ROS workspace `~/ros` on each robot you should use to develop your own ROS packages.

2.2.1: Boot the Turtlebot & its laptop. Login to the system.

2.2.2: Start ROS and the nodes for a basic setup to use the mobile base and the camera. Good tutorials are available at the Turtlebot wiki page:

```
http://wiki.ros.org/Robots/TurtleBot
```

To account for custom changes on TAMS’ turtlebots (i.e. an additional laser scanner), you have to start TAMS-specific launch files which are modified for our setup. To bringup the turtlebot, don’t run any launch file from the package `turtlebot_bringup`. Instead run:

```
roslaunch tams_turtlebot_bringup tams_turtlebot.launch
```

2.2.3: Start the visualization tool RViz on your local computer. Try to get a live camera image and drive around manually with the `turtlebot_teleop` node. Visualize the point cloud provided by the Kinect sensor. Use the 3D visualization tutorial to read the data provided by the Kinect sensor:

```
http://wiki.ros.org/turtlebot/Tutorials/indigo/3D%20Visualisation
```

Make sure to correctly export the `ROS_MASTER_URI` on your desktop computer before you start the RViz visualization tool.

Task 2.3 Kinect Listener: Write a listener node to read the Kinect data

2.3.1: Find out the identifier and the message type of the Kinect topic.

2.3.2: Look up the ROS tutorial for a simple subscriber. There is one for C/C++ and one for Python. You can choose the language you like. Make sure you are looking up the right version of ROS. You used this tutorial last time for the publisher.

```
C/C++: http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber(c++)
Python: http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber(python)
```

Task 2.4 Simple Movement: Write a node to move the Turtlebot.

2.4.1: Extend your sensor node to move the Turtle bot.

2.4.2: Test your node on the robot. Make sure there is enough space around the robot.

Never leave the robot unattended!

2.4.3: Extend your node so that the turtle stops, when you hit a key.

Task 2.5 Detect obstacles: Write a program to move the TurtleBot and stop in front of an obstacle.

2.5.1: Use your code from the previous two tasks and extract the distance to an object directly to the front of the Kinect.

2.5.2: Now let the robot move forward until it detects an obstacle closer than 1 m. Use a soft obstacle to test your code.

Task 2.6 Optional: Wall following: Write a program to move the TurtleBot along a wall to explore the environment.

Task 2.7 Optional: Mapping: Use gmapping on the Turtlebot with RViz to create a map of the TAMS floor.

Note: You could use the `gmapping.launch` *file in the* `tams_turtlebot_navigation` *package.*