TurtleBot Simulator Exercise 10/3/2018

Type in your Seminar Username and password.

When the Unity Desktop comes up, press Ctrl + Alt + T and bring up a terminal window.

Reference: These exercises outline the information and commands for the TurtleBot Simulator presented in ROS Robotics By Example, Chapter 3 –Driving Around with TurtleBot. Pages 81 – 118 in this book will provide a description of the commands and additional information about TurtleBot's behavior. Troubleshooting information is also provided.

Tip: Remember to use tab completion to see the fields of the message! Remove any CR in commands.

To start Gazebo and the TurtleBot 2 simulator, first open a terminal window and type the roslaunch command:

$ roslaunch turtlebot_gazebo turtlebot_world.launch

Move the Gazebo environment around to look at TurtleBot from various perspectives.

Open a 2nd terminal window and find the position and orientation of TurtleBot's base in Gazebo:

$ rosservice call gazebo/get_model_state '{model_name: mobile_base}'

Now it is time to examine some of TurtleBot's ROS components: topics and services. Try the following commands to explore these ROS elements and examine the framework of TurtleBot:

$ rosservice list

$ rostopic list | grep mobile_base

$ rostopic type /mobile_base/commands/velocity

(page 86) Then drive TurtleBot with the following command:

$ rostopic pub -r 10 mobile_base/commands/velocity \geometry_msgs/Twist '{linear: {x: .2}}'

Use Ctrl+C to kill the process.

Next, launch teleop mode to command TurtleBot to move with the keyboard keys:

$ roslaunch turtlebot_teleop keyboard_teleop.launch

(The keyboard teleop window must be "active" for the keys to work.)

Use Ctrl+C to kill the process.

(skip to page 96) Examine the TurtleBot dashboard
$ roslaunch turtlebot_dashboard turtlebot_dashboard.launch

Play with icons at the top of dashboard to see what can be viewed and controlled. Most buttons do not work on the simulated TurtleBot. This is an rqt screen.

(skip to page 100) Control TurtleBot's movement with the following commands:
$ rostopic pub -r 10 /mobile_base/commands/velocity \geometry_msgs/Twist '{linear: {x: 0.2}}'
Use Ctrl+C to kill the process.

$ rostopic pub -r 20 /mobile_base/commands/velocity \geometry_msgs/Twist '{linear: {x: -0.2}}'
Use Ctrl+C to kill the process.

(Remove <CR>)
$ rostopic pub -r 10 /mobile_base/commands/velocity \geometry_msgs/Twist '{linear: {x: 0.2}, angular: {x: 0, y: 0, z: 1.0}}'

Open another terminal window to see the messages passed on this topic:
$ rostopic echo /mobile_base/commands/velocity

Use Ctrl+C to kill the processes in both windows.

(page 102) Python script

$ rosnode list

Copy the file ControlTurtleBot.py from the Seminar website.

$ chmod +x ControlTurtleBot.py

$ python ControlTurtleBot.py
Use Ctrl+C to kill the process.

(page 105) rqt graph
$ rqt_graph

Open another terminal window and start keyboard teleop:
$ roslaunch turtlebot_teleop keyboard_teleop.launch

Refresh rqt_graph to see the new node and topic.

In another terminal window, type the command:
$ rosnode info /turtlebot_teleop_keyboard

Use Ctrl+C to kill the processes in both windows.

(page 107) rqt message publisher and topic monitor
$ rqt

Open Topic Monitor and Message Publisher from the Plugins → Topics menu option.

(page 110) TurtleBot 2 Odometry
Use Ctrl_C to kill process for rqt.

$ rostopic type /odom

$ rosmsg show nav_msgs/Odometry

$ rostopic echo /odom
Use Ctrl+C to kill the process.

$ rostopic type /mobile_base/commands/reset_odometry

$ rostopic pub /mobile_base/commands/reset_odometry std_msgs/Empty

In another terminal window, type:
$ rostopic echo /odom/pose/pose

In another terminal window, type:
$ rostopic echo /mobile_base/sensors/imu_data

(page 114) More TurtleBot 2 Odometry
Kill all processes and close all windows to restart Gazebo and TurtleBot 2 Simulator.

In the first terminal window, type the command:
$ roslaunch turtlebot_gazebo turtlebot_world.launch

In another terminal window, run this command:
$ roslaunch turtlebot_rviz_launchers view_robot.launch

In rviz: Add the Odometry display to the rviz display. Under Global Options, select odom for Fixed Frame. Check the Odometry checkbox to make visible. Under Odometry, select the Topic /odom. Deselect the Covariance checkbox if it is checked.

In another terminal window, run one of the following commands: (Remove <CR>)
$ rostopic pub -r 10 /cmd_vel_mux/input/teleop \geometry_msgs/Twist '{linear: {x: 0.1, y: 0, z: 0}, angular: {x: 0, y: 0, z: -0.5}}'

(Remove <CR>)
$ rostopic pub -r 10 /mobile_base/commands/velocity \geometry_msgs/Twist '{linear: {x: 0.1, y: 0, z: 0}, angular: {x: 0, y: 0, z: -0.5}}'
Use Ctrl+C to kill the process.

$ python ControlTurtleBot.py
Use Ctrl+C to kill the process.
Kill all processes and close all windows to restart Gazebo and TurtleBot 2 Simulator.

TurtleBot 2 Navigation

WARNING: This exercise in Gazebo is a poor example of the autonomous navigation of a real TurtleBot. Better TurtleBot simulation controllers are needed.

Driving without steering TurtleBot 2

Reference: ROS Robotics By Example, Chapter 4 –Navigating the World with TurtleBot explains TurtleBot navigation (mapping a room and autonomous navigation) for a real TurtleBot 2. These exercises outline the information and commands for autonomous navigation using the TurtleBot Simulator. Pages 175 – 193 in this book will provide a description of the commands and additional information about TurtleBot's autonomous navigation.

Copy the files gazebo_map.pgm and gazebo_map.yaml from the Github website:
https://github.com/FairchildC/ROS-Seminar-files

(Pages 175) To start Gazebo and the TurtleBot 2 simulator, first open a terminal window and type the command:
$ roslaunch turtlebot_gazebo turtlebot_world.launch

Use the pwd command in the directory where the gazebo_map files are stored to find the path to that working directory. Open the gazebo_map.yaml file in the editor (gedit) and change the first line of the file to contain the path to your gazebo_map.pgm file. Then save and close the file.

In a second terminal window, start the amcl operation: (REMOVE <CR>)
$ roslaunch turtlebot_gazebo amcl_demo.launch
map_file:=/users/student/rosuser#/gazebo_map.yaml
(Change the # to your rosuser number.)

Look for the following text on your window:
odom received!

Open another terminal window and view navigation on rviz:
$ roslaunch turtlebot_rviz_launchers view_navigation.launch

TurtleBot should be told its current location on the map as it corresponds to the Gazebo environment. Click the **2DPose Estimate** button on the top toolbar of rviz and then click on the location on the map where TurtleBot should be located. Orient the large green arrow by dragging the cursor in the direction TurtleBot is facing.

To get TurtleBot to navigate autonomously across the map, click on the **2D Nav Goal** button on the top toolbar. Next, click on the map location where you want TurtleBot to go and drag the green arrow in the direction TurtleBot should be facing when he reaches that location.

Navigating to a designated location

Kill all processes and close all windows to restart Gazebo and TurtleBot 2 Simulator.

(page 183) In the first terminal window, type the command:
$ roslaunch turtlebot_gazebo turtlebot_world.launch

Open another terminal window, check the initial pose of TurtleBot:
$ rostopic echo /odom/pose/pose
Use Ctrl+C to kill the process.

In a third terminal window, launch gmapping_demo:
$ roslaunch turtlebot_navigation gmapping_demo.launch

Open another terminal window and move TurtleBot ahead about 1 meter by typing the command:
$ rostopic pub /move_base_simple/goal geometry_msgs/PoseStamped
"header:
seq: 0
stamp:
secs: 0
nsecs: 0
frame_id: 'map'
pose:
position:
x: 1.0
y: 0.0
z: 0.0
orientation:
x: 0.0
y: 0.0
z: 0.0
w: 1.0"
(Don't forget about tab completion!)

(Sometimes TurtleBot does not move or is slow to move. Be patient.)

In the second window, check TurtleBot's final pose with the command:
$ rostopic echo /odom/pose/pose

Navigating to waypoints with a Python script using a map

Kill all processes and close all windows to restart Gazebo and TurtleBot 2 Simulator.

Copy the file MoveTBtoGoalPoints2.py from the Seminar website.

(page 185) In the first terminal window, type the command:
$ roslaunch turtlebot_gazebo turtlebot_world.launch

Open another terminal window, launch amcl operation: (REMOVE <CR>)
$ roslaunch turtlebot_gazebo amcl_demo.launch
map_file:=/users/student/rosuser#/gazebo_map.yaml
(Change the # to your rosuser number.)

Next, open a third window and launch rviz and display the map:
$ roslaunch turtlebot_rviz_launchers view_navigation.launch

Open a fourth window to display TurtleBot's initial pose on the map. Type the command:
$ rostopic echo /initialpose

When you use the **2D Pose Estimate** button in rviz to indentify TurtleBot's initial position and orientation, information on TurtleBot's initial pose will appear in the **echo /initialpose** terminal window.

Find other points on the map by using the **Publilsh Point** button on the rviz toolbar, type the following command:
$ rostopic echo /clicked_point

Then use the **Publish Point** button and click on a point on the map.

The goal locations of (0, -2..0) and (-2.0, 2.0) have been selected on the map in Gazebo and used in the Python script MoveTBtoGoalPoints.py. Other locations can be selected as you wish. Execute this script with the command:
$ python MoveTBtoGoalPoints.py

C heck the robot's final position on the map by typing the command:
$ rostopic echo /amcl_pose

To view only the position and orientation, use the command:
$ rostopic echo /odom/pose/pose