Navigating using a ROS Node - Simple Action Client

So far, we've had a lot of fun racing the TurtleBot around the factory, but when we think of (real world) ROS applications, having to use the RVIZ UI to set new navigation goals every time is not desirable.

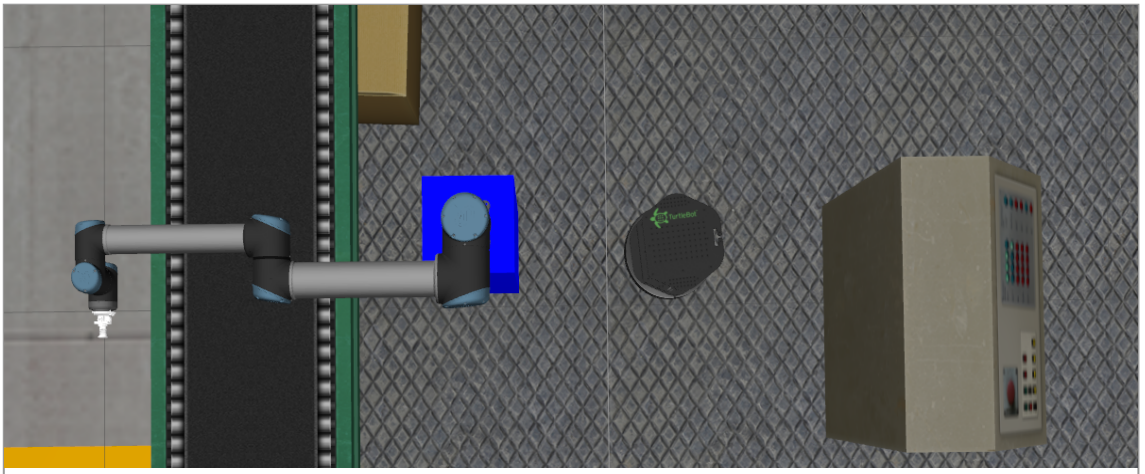
In this assignment, we will learn how to command navigation goals to the TurtleBot using a ROS node that implements a *simple action client* interface that can send goals the `move_base` action server. In fact, the `move_base` action server that has so far been servicing all our navigation goals, only they were sent via RVIZ.

In the last part of this assignment, we will also work with a new challenge, when previously unknown obstacles start showing up along a path that has already been planned. Will the TurtleBot be able to avoid unknown obstacles as well? Jump in and start playing with the scripts! You will find out soon!!

Week 3 - Assignment 3 - part 1 (of 2) --- 3 Points

The goal of this part of the assignment is to update two files: `week3_assignment3_part1.py` and `week3_assignment3_part1.launch`.

The goal is to have the TurtleBot navigate to its first target location for the factory operation, as shown in the figure below:



You will work towards this goal using the following steps:

Make sure you close all previous CCSs, if you have any open from previous assignments.

Step 1: Start the Factory simulation with:

```
$ roslaunch hrwros_week3
hrwros_turtlebot_navigation.launch
```

Step 2: Start the amcl_localization.launch file, it should now have the correct arguments for the map and the initial pose estimate of the TurtleBot.

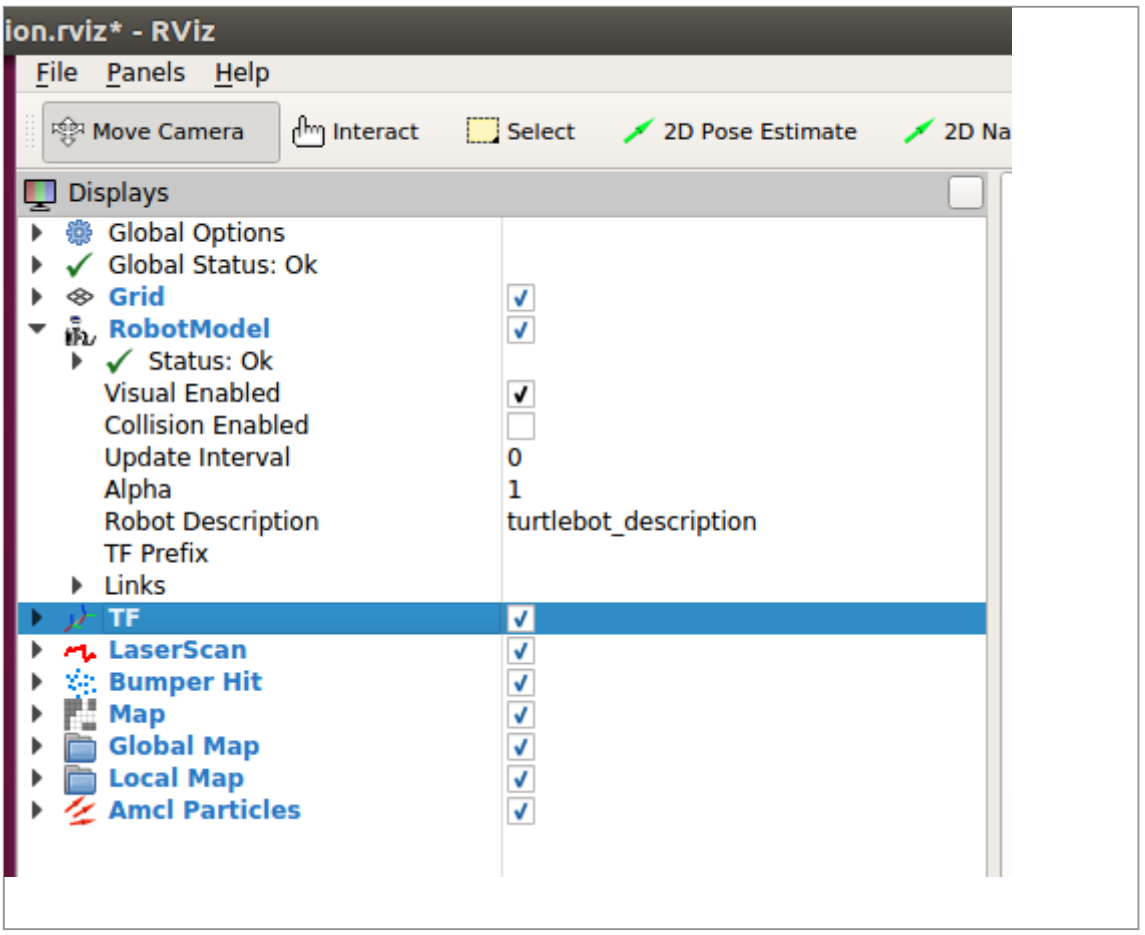
```
$ roslaunch hrwros_week3_assignment
amcl_navigation.launch
```

Step 3: Next, in another CCS, start the RViz visualization with:

```
$ roslaunch hrwros_week3_assignment
view_navigation.launch
```

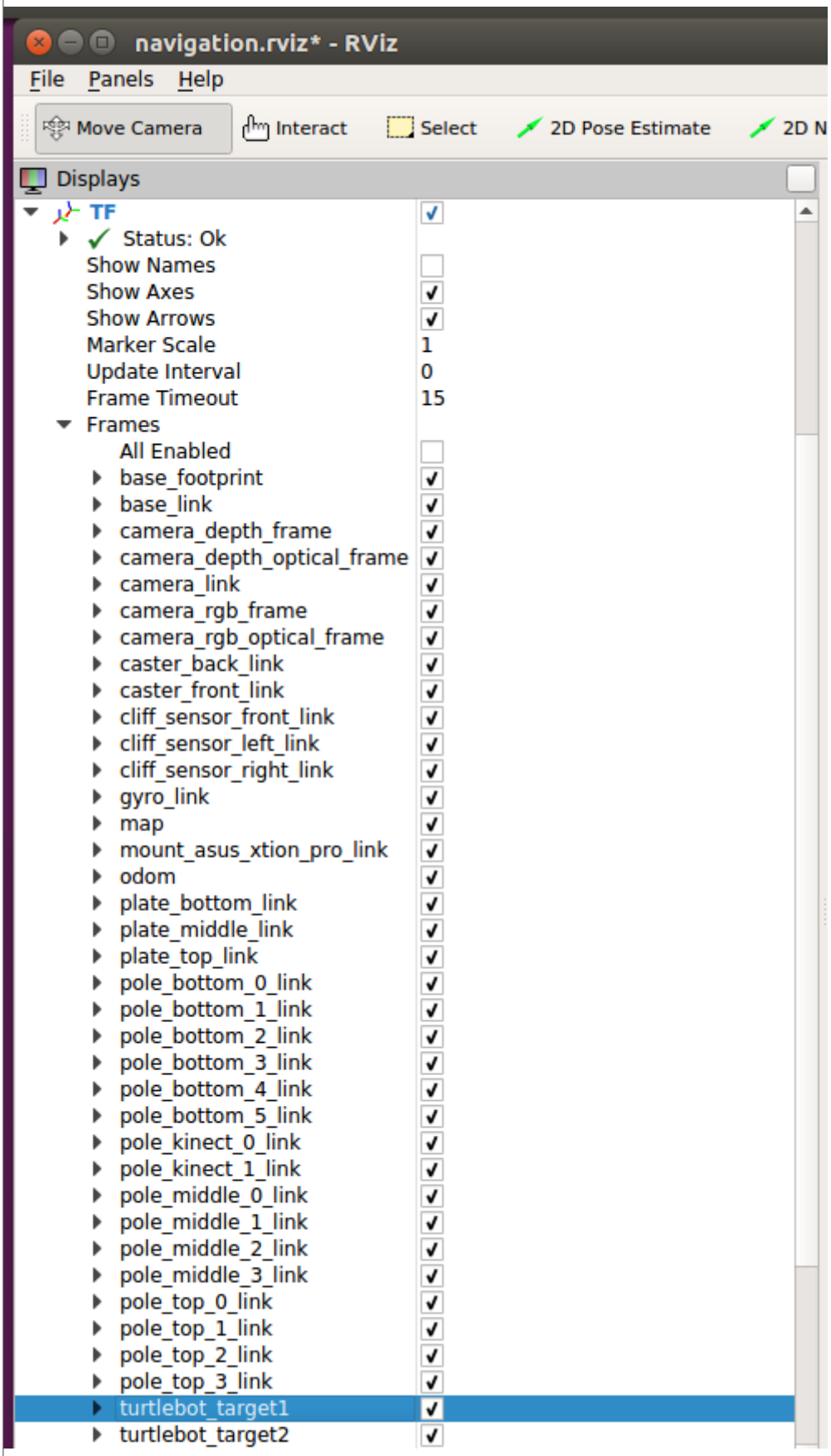
If you follow the steps from Assignment 1, you should visualize the TurtleBot in the green cloud of arrows indicating its potential initial position.

Step 4: In RViz, enable the TF `display` by clicking on the checkbox next to it like it is shown in the screenshot below:



Step 5: If you expand the TF display by clicking on the black triangle to the left of the letters TF, you will see several elements with a check mark against them under the category Frames.

If you scroll down a bit, you will be able to see a frame called `turtlebot_target1` as shown in the figure below:



The **tasks for completing** this assignment are:

Task 1: Update the relevant information to complete the missing parts of the script and add the first goal in the `week3_assignment3_part1.py`. (Hint: you only have to change four lines of code in this file)

> You can get the exact location from the `turtlebot_target1` frame display by clicking on the small triangle to its left.

Task 2: Update `week3_assignment3_part1.launch` so that we can actually start the ROS node that will move the TurtleBot to the first target location as shown at the beginning of this assignment.

To check if everything is correct, just launch the file you just edited with:

```
$ roslaunch hrwros_week3_assignment week3_assignment3_part1.launch
```

If everything is good, when you launch this, the TurtleBot will autonomously navigate to the first target location, **wait until the Turtlebot arrives to its goal.** It may take some time, and the TurtleBot may struggle to reach it, if it fails, just re-launch it. Once the TurleBot has arrived at its goal, you should get the following messages: "Goal sent to move_base action server.", "Goal position x: <> y: <>", "Hooray! Successfully reached the desired goal".

Week 3 - Assignment 3 - part 2 (of 2) --- 3 Points

In this part, the goal is to navigate the TurtleBot to its second target location, following similar steps as you followed in the first part, but with an added difficulty: To visualize and become aware of the "**unknown obstacle avoidance**" feature of the ROS autonomous navigation package which we have learned this week.

For this part, you will need to update two files: `week3_assignment3_part2.py` and `week3_assignment3_part2.launch`.

You will work towards this goal using the following steps:

Step 1: Following the same steps as in the Part 1, update the `week3_assignment3_part2.py` script.

- Complete the missing parts of the script and add the second target location for the TurtleBot.
  You will find the information under the `turtlebot_target2` frame, also in the TF Display in RViz.

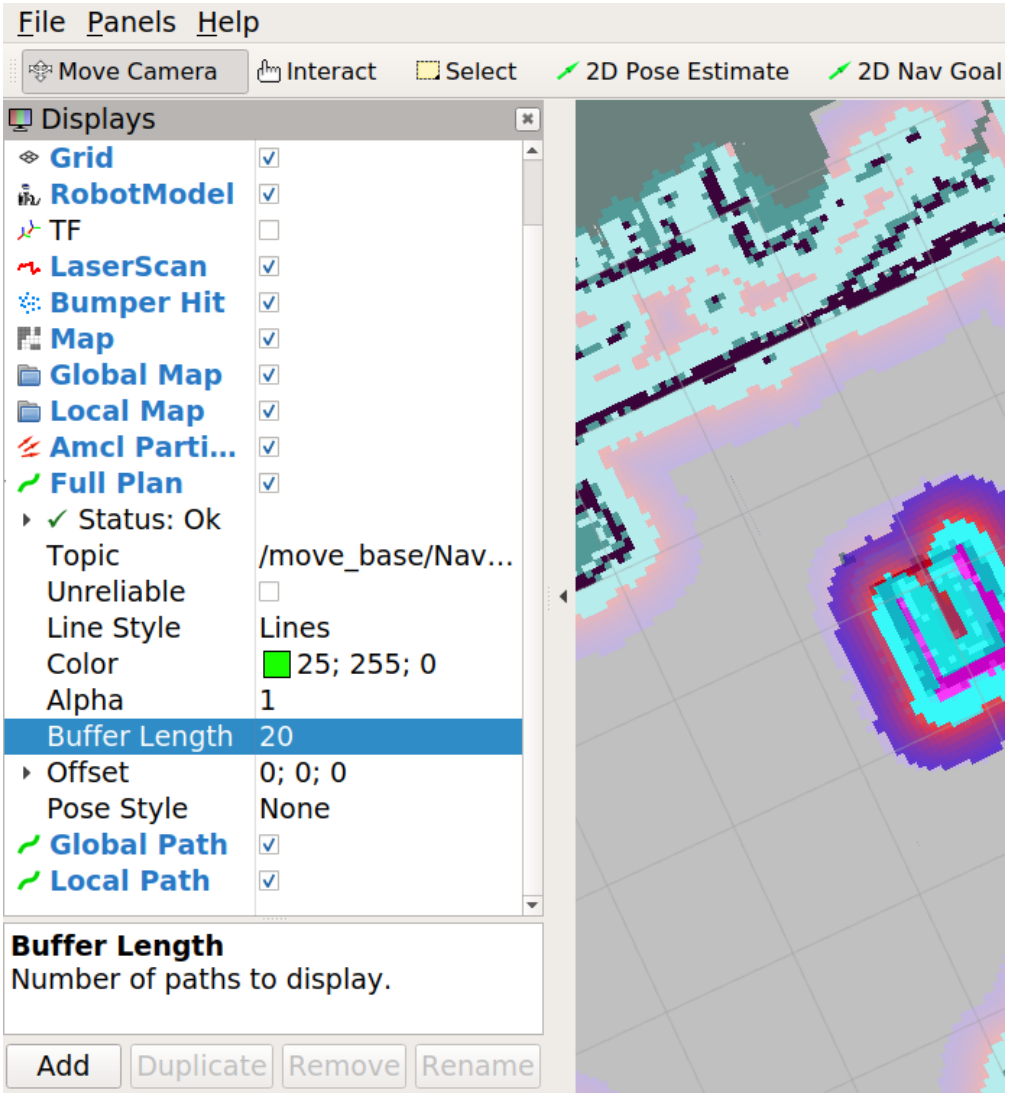
Step 2: Update the `week3_assignment3_part2.launch` with the corresponding type of node, but **don't launch it yet**.

Step 3: Make sure you have all the **Three Path Displays** enabled:

- One for the overall global plan topic `/move_base/NavfnROS/plan.`
- One for the global plan of the Dynamic Window Approach (DWA) planner which is used for obstacle avoidance (topic: `/move_base/DWAPlannerROS/global_plan`)
- One for the local plan of the DWA planner (topic: `/move_base/DWAPlannerROS/local_plan`).

Step 4: To better visualize the obstacle avoidance (the modified global path around the obstacle), you need to retain the global path.

- Change the "Buffer length" entry of the **Full Path** display to 20, as shown in the figure below:



Step 5: Assuming you did not close or stop any ROS nodes from the previous part, you can start the `week3_assignment3_part2.launch` file in a new CCS with:

```
$ roslaunch hrwros_week3 week3_assignment3_part2.launch.
```

If you did close or terminate the previous part, then make sure re-run the steps in Part 1, so the TurtleBot is at the first target location.

Step 6: A couple of seconds after the robot starts moving, you will notice that two new obstacles pop on Gazebo.

- These obstacles were not known to the global planner at the time of planning.
- So the initial path might actually collide with or go very close to the obstacles, however, as the TurtleBot approaches (one or both of) these obstacles, the global path gets modified.
- You should see the changes on the `Full Path` as it avoids the obstacles, depending on which way the global planner plans the overall path.
- Remember, the solution that the global planner finds can approach the target2 position either side of the obstacles, also the TurtleBot may have a hard time avoiding them, and take a few turns, but it's OK.

Remember to save the RVIZ configuration with `File -> Save config` **or** `Ctrl+S`

You have completed all the assignments of week 3! Please go to the submission unit and upload all the contents of the `hrwros_week3_assignments` package folder, as **a single .zip** file.