AA 274A: Principles of Robot Autonomy I
Section 4: Visualizing Information from Robots
(In-person section)

Our goals for this section:

1. Potentially become familiar with catkin package installation
2. Become familiar with tools for visualizing information from your robots.
3. Write a marker using rviz.

1 Package Installation

Before we dive into information visualization, we may have to install some packages to endow our robots with the necessary capabilities we're looking for. Chiefly, some of your turtlebots do not have the `velodyne` catkin package installed. This poses a problem since, without it, we cannot use the scans coming out of the velodyne.

First, we need to check to see if the package is already on our robot. We can do this by `ssh`'ing into our robot (remember to first connect to the correct network):

```
# Run this command on the laptop
ssh aa274@<lowercase_robot_name>.local
```

with the password `aa274` to remotely log into the onboard robot computer

Once inside the robot, you will need to check the contents of the `catkin_ws/src` directory

```
# Run this command on the robot
cd catkin_ws/src
ls
```

If you see a folder named `velodyne` then you already have the package.

Already have the package: Great! Let's update it to make sure it's up to date.

```
# Run this command on the robot
cd velodyne
git pull
```

Don't have the package: Great! Let's get it now. To add a new package to the our catkin workspace, simply clone the package from where it resides to the `catkin_ws/src` directory. Fortunately for us, the `velodyne` package is easily-accessible through GitHub. To obtain it, execute the following from within the `catkin_ws/src` directory (you can check this by executing `pwd`):

```
# Run this command on the robot
git clone https://github.com/ros-drivers/velodyne.git
```

Update `asl_turtlebot`: Before we build our workspace, make sure the `asl_turtlebot` repository is up-to-date.

```
cd asl_turtlebot && git pull
```

Build the package: Once the package has finished cloning/pulling, we need to rebuild our catkin workspace. Recall, we can do this as follows

```
# Run this command on the robot
cd ~/catkin_ws # We want to be within the catkin_ws directory
catkin build
```

This will take some time, so leave it alone until it's fully finished. Once it completes successfully, rejoice! You now have the necessary packages to move on.

2 Information Visualization with `rviz`

`rviz` is a 3D visualization tool for ROS, visualizing information such as maps and point clouds in a much more intuitive way compared to `rostopic echo`.

2.1 Full Turtlebot bring up

Next, in a terminal window, ssh into the TurtleBot. Now that we have all of the necessary packages, we can go ahead and run:

```
# Run this command on the robot
roslaunch asl_turtlebot asl_turtlebot_core.launch
```

Problem 1: Once this is all running, which rostopics are available? You can run `rostopic list` from a new terminal window on the laptop, but remember to run `rostb3` in every terminal window you open. Paste the list of topics in your submission.

2.2 Starting up `rviz`

Open another terminal (remember to run `rostb3`) and execute

```
# Run this on the laptop
rviz
```

Now let's visualize some data!!

2.2.1 Velodyne's 3D Point Cloud

In `rviz`, to add a topic to the display, you need to click on the "Add" button on the bottom left. Once you've clicked this, a popup will come up. In the popup, click the "By topic" tab and scroll down until you see "PointCloud2." Double-click this and watch the pretty colors appear.

Note: Use the throttled channel if you are seeing large network latency.

2.2.2 `gmapping`'s Environment Map

Next, add "Map" under "/map." This is what the mapping package we use provides us from LIDAR data.

2.2.3 `gmapping`'s Odometry

Next, add "Odometry" under "/odom." This is what the mapping package we use provides us as the robot's odometry from LIDAR data.

2.2.4 Coordinate Axes

Next, add "Axes" from under the "By display type" tab (not "By topic"). This shows us where the world origin point is and the orientation of the axes.

2.2.5 `tf`'s Transform Tree

Next, add "TF" from under the "By display type" tab. This shows us the axes origins and orientations of any other frames we've defined. You can see we've defined quite a few for this robot (you may have to zoom in to read them).

Problem 2: Take a screenshot of your `rviz` display after all of the above are running. Feel free to wave to the camera to show it works in real time.

2.3 Saving `rviz`'s Current Configuration

That was quite a lot of topics to open, imagine doing that every time you wanted to debug your ROS stack! Thankfully, there's a way we can save this current configuration to a file and open it with `rviz`, using `File -> Save Config As`.

Problem 3: Paste the contents of your created `.rviz` configuration file into your submission.

Close and reopen `rviz`, load the config file you just created so you can see that it reloads all of your previously-opened topics.

3 `rviz` Markers

Markers are a very helpful tool for visualizing intermediate rewards, computed trajectories, etc. all in `rviz`! The way markers work is by taking advantage of `rviz`'s built-in handling of ROS topics and messages. To create a Marker, we'll publish a Marker message to a topic and then view the topic in `rviz`.

Code to create a Marker object can be found in the `code/` directory of today's section. Running it creates a green oblong sphere at world coordinates $(1, 1, 1)$, each in meters.

Problem 4: Change the Marker to look like a normal red sphere and place it $1m$ in front of the robot (think about which axis this corresponds to), include a screenshot of your marker and its placement in your submission.

Hint: Thinking about changing the `frame_id` of the marker to make it relative to a moving frame (`base_footprint`) attached to the robot, and the position coordinates for this marker is just (1, 0, 0).

4 Cleanup

When you have completed all of the hardware tasks, run the following commands in your ssh-ed terminal window:

```
# run this command on the robot
```

```
sudo shutdown -h now
```

with the same password you used to log in: `aa274`. This should log out all windows that were ssh-ed into the robot. Wait 4-5 seconds, then flip the power switch on the powerboard to "OFF".

Please stop all running processes on the laptop.