

# Exercise Session 2

## Theory

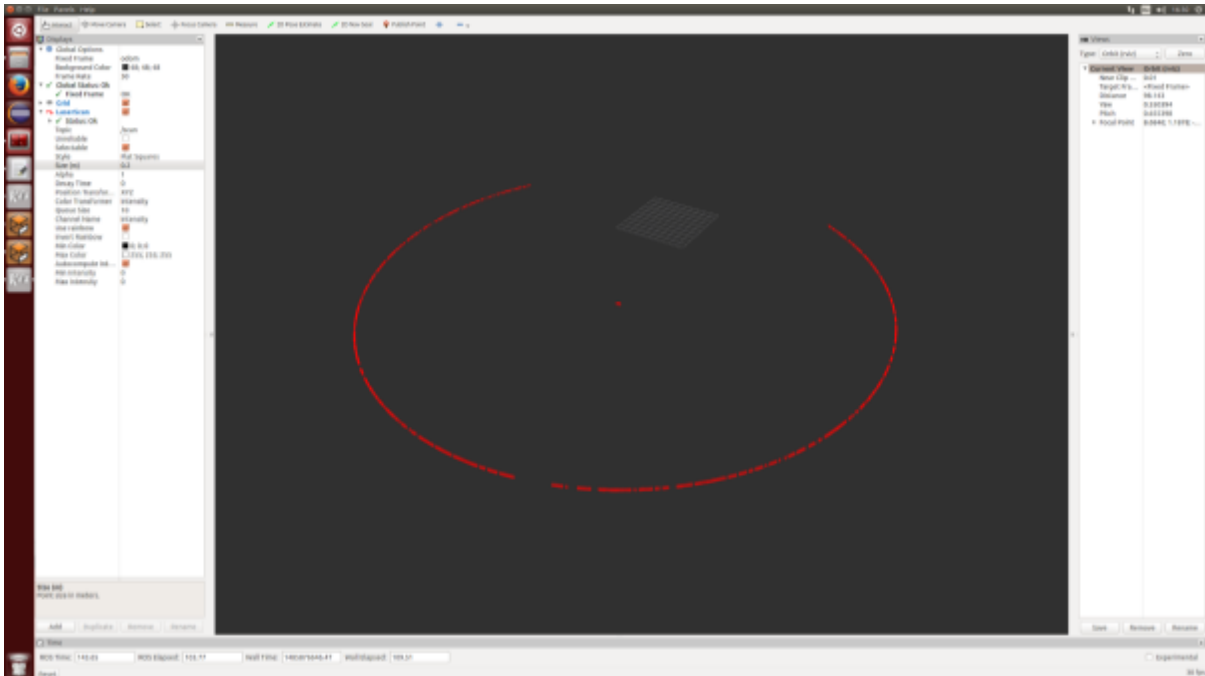
- ROS package structure
- Integration and programming with Eclipse
- ROS C++ client library (roscpp)
- ROS subscribers and publishers
- ROS parameter server
- RViz visualization

## Exercise

In this exercise, you will create your first ROS package. The package should in the end be able to subscribe to a laser scan message from the Husky robot and process the incoming data. This node will be the basis for the next exercises.

1. **OPTIONAL** (more difficult): Create the package `husky_highlevel_controller` from scratch (look at the ROS template for reference [https://github.com/ethz-asl/ros\\_best\\_practices](https://github.com/ethz-asl/ros_best_practices)). Use the command `catkin_create_pkg` to create a new package with the dependencies `roscpp` and `sensor_msgs`.
2. **OR** (easy): Download the Zip archive containing prepared files of the package `husky_highlevel_controller` from the course website.
3. Inspect the `CMakeLists.txt` and `package.xml` files.
4. Create a subscriber to the `/scan` topic.
5. Add a parameter file with topic name and queue size for the subscriber of the topic `/scan`.
6. Create a callback method for that subscriber which outputs the smallest distance measurement from the laser scanner to the terminal. Inspect the message type here [http://docs.ros.org/indigo/api/sensor\\_msgs/html/msg/LaserScan.html](http://docs.ros.org/indigo/api/sensor_msgs/html/msg/LaserScan.html).
7. Add your launch file from Exercise 1 and add the current node and add loading the parameter file to the launch file.
8. Pass the argument `laser_enabled` from your launch file to the `husky_empty_world.launch` file with value `true`.

9. Show the laser scan in RViz and add RViz to your launch file. Make sure to set the correct *Fixed Frame* (under *Global Options*) and adapt the size of the laser scan.



RViz visualization of a single laser scan. An obstacle can be seen in the middle. Note the changed "Fixed Frame" as well as "Size (m)".

## Evaluation

- Start the launch file. Drive the Husky. There should be changing output from the laser scanner in the terminal. [40%]
- Check if the node is implemented as the template suggests. [30%]
- Is a parameter file used? [15%]
- Is the laser scan visualized in RViz as shown in the image? [15%]