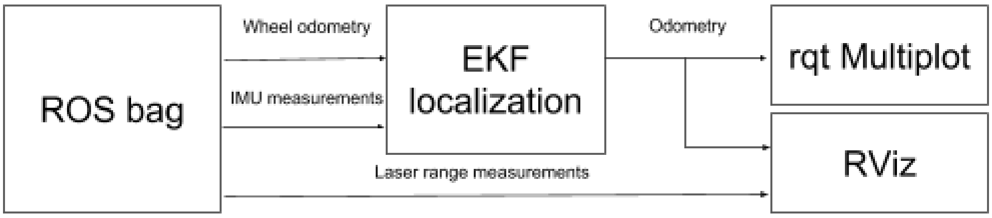
Exercise Exam 3

Theory

- ROS bag
- ROS time
- rqt_multiplot
- ROS launch
- RViz

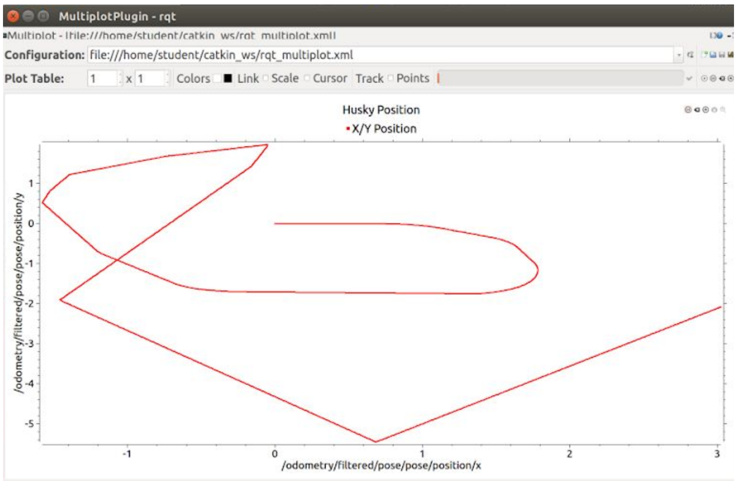
Exercise

The goal of this exercise is to work with data that was recorded on a real Husky robot. The recorded bag file contains sensor measurements from wheel odometry, inertial measurement unit (IMU) and laser scanner. Your task is to use this raw sensor data to localize the robot with a provided state estimation node that implements an extended Kalman filter (EKF). The same localization node is already running if you launch the Husky simulation. To see the results, plot the output of the localization node using rqt Multiplot and visualize the laser data in RViz.



1. Start the launch file from the last exercise. Check what the node `/ekf_localization` is doing. What is it subscribing to and what is it publishing? Visit http://docs.ros.org/kinetic/api/robot_localization/html/index.html for further information.

2. Launch your controller from the last exercise. Use *rqt_mulitplot* to plot the path of the simulated robot in the *x*/*y*-plane (Tip: use the topic `/odometry/filtered`).



MultiplotPlugin - rqt

Multiplot - [file:///home/student/catkin_ws/rqt_multiplot.xml]

Configuration: file:///home/student/catkin_ws/rqt_multiplot.xml

Plot Table: 1 x 1 Colors ■ Link □ Scale □ Cursor Track □ Points

Husky Position
X/Y Position
/odometry/filtered/pose/pose/position/y
1
0
-1
-2
-3
-4
-5
-1
0
1
2
3
/odometry/filtered/pose/pose/position/x

Traveled path of Husky.

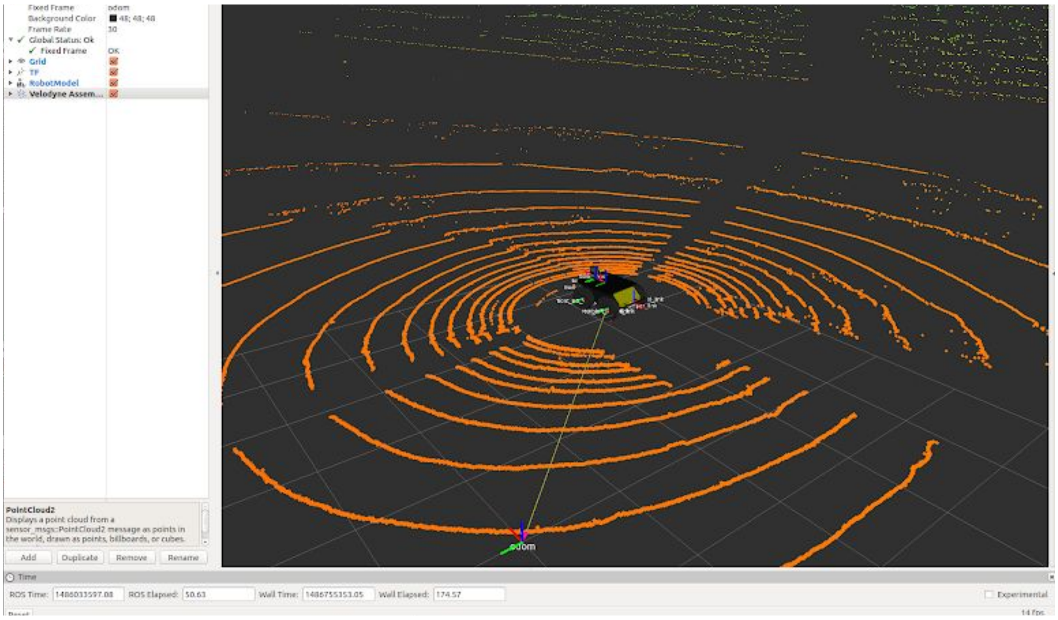
3. Download the provided rosbag `husky_navigation.bag` from the course website, investigate the content with the command `rosbag info`.

4. Write a launch file that starts an `ekf_localization_node` subscribing to the topics provided in the bag-file. Load the parameters from the same config file as it is done by the simulation (Tip: The config file `localization.yaml` can be found with `roscd husky_control/config`).

5. Use `rqt_mulitplot` to plot the path of the recorded robot in the *x*/*y*-plane.
    - Tip: Remember to set the parameter `/use_sim_time` to `true`: http://wiki.ros.org/Clock.
    - Tip: Play the bag-file with: `rosbag play` *`mydata.bag`* `--clock` which publishes also the time of the recorded data http://wiki.ros.org/rosbag/Commandline.

6. Visualize the motion of husky by using TF markers in RViz. Add a `robot_state_publisher` node to your launch file and load the husky robot description to the parameter server. Now you can visualize the husky model in RViz. (Tip: Use the `control.launch` file from the `husky_control` package as reference)

7. The rosbag also contains laser data from a Velodyne LiDAR. Visualize the point cloud in RViz. It should be moving with the robot.



Fixed Frame: odom
Background Color: ■ 48; 48; 48
Frame Rate: 30
✓ Global Status: Ok
✓ Fixed Frame: OK
Grid ☑
TF ☑
RobotModel ☑
Velodyne Assem... ☑

PointCloud2
Displays a point cloud from a sensor_msgs::PointCloud2 message as points in the world, drawn as points, billboards, or cubes.

ROS Time: 1486033597.08 ROS Elapsed: 50.63 Wall Time: 1486755353.05 Wall Elapsed: 174.57

Visualization of Husky with point cloud from Velodyne LiDAR.

Evaluation

❏ Plot the *x*/*y*-plane of the simulated Husky in `rqt_multiplot`.	[30%]
❏ Plot the *x*/*y*-plane of the recorded Husky in `rqt_multiplot`.	[40%]
❏ Visualize point cloud from Velodyne LiDAR in RViz.	[30%]