MRT M2 practical work

Sensor fusion, application to the ROS middleware

Introduction

The goal of this exercise it to discover the ROS sensor fusion stack and more precisely the robot_localization package. ROS is a very used middleware in the field of robotics that provides a lot of open source ressources for researchers and industrial applications. ROS is mainly programmed in C++ and Python and has binding with a lot of other languages and software like Matlab.

Deliverable

You will provide at the end of this 4 hours pratical work a short document (max 3 pages) that details your results: what you have done, what do you understand about the ROS sensor fusion stack, do you succeed to implement a sensor fusion algorithm? Don't hesitate to add some figures and graphs to explain your results.

Exercises

1. Installation of ROS packages

You have at your disposal a computer with a pre-installed ROS melodic system under Ubuntu 18.04. Use the terminal to install the needed package as described in the references section. For example, you can use command like this:

```
sudo apt install ros-[ROS_VERSION]-robot-localization
```

by replacing [ROS_VERSION] by your own ROS installation.

2. Playing with ROS bags

During MRT lectures, you got courses about ROS middleware, read the documentation about ROS bags (the storage format used by ROS for the experimental data). Try to replay the data provided for this pratical class [5] and use the ROS tools to analyze the different types of data (IMU, position, attitude, odom, …).

3. Implement a Kalman Filter with ROS

Thanks to the documentation in [1] and [2], implement a robot localization algorithm. You don't need to code by yourself the algorithm. The robot_localization package provides to you pre-coded algorithm and you can use freely. Analyze the effect of the filter on the accuracy and robustness of the localization.

4. More advanced filtering method

Relatively to the hand-on in [3], adapt the robot_localization system that you have implemented in exercise 3 in order to add a laserscanner layer with the AMCL algorithm [4]. AMCL (Adaptive

Monte Carlo Localization) takes into input a laser scan and a know map to localize the robot. This method in based on a particular filtering theory.

Test the AMCL method on the provided data and map, in your report emphasize the gain of using such sensor fusion techniques to improve your localization.

References

- [1] ROScon 2015 talk : https://vimeo.com/142624091
- [2] robot_localization ROS package : http://docs.ros.org/melodic/api/robot_localization/html/index.html
- [3] A step-by-step tutorial for ROS sensor fusion implementation : https://github.com/methylDragon/ros-sensor-fusion-tutorial
- [4] AMCL package http://wiki.ros.org/amcl
- [5] Data for the practical work:
  - Navigation data of the Robtonik Summit XL robot on the Polytech Lille parking under normal conditions : https://nextcloud.univ-lille.fr/index.php/s/4BzipNdrPPbC2oN
  - Navigation data of the Robtonik Summit XL robot on the Polytech Lille parking with GPS outages : https://nextcloud.univ-lille.fr/index.php/s/QtzdkSo7FJ5zjiY