Lab05

Localization: ROS 2 robot_localization package

Objectives

- Understand the structure of the robot_localization package in ROS 2
- Understand the parameters used by the robot_localization nodes
- Set up a filter to track TurtleBot3 using odometer and IMU

Introduction

robot_localization is a ROS 2 package designed to provide state estimation for mobile robots and other robotic systems. It's primarily used for fusing data from various sensors (such as odometry, GPS, IMU - Inertial Measurement Unit, LiDAR, etc.) to estimate the robot's pose (position and orientation) within its environment.

The package implements sensor fusion algorithms, including Extended Kalman Filters (EKF) and Unscented Kalman Filters (UKF), to integrate information from multiple sensors. By combining data from different sources, it aims to enhance the accuracy and reliability of the robot's pose estimation, especially in scenarios where individual sensors might provide noisy or incomplete information.

robot_localization is helpful in scenarios where a robot needs to maintain an accurate understanding of its position and orientation in the presence of sensor uncertainties or when multiple sensors provide complementary information about its state. It's widely used in robotics applications for tasks like autonomous navigation, mapping, localization, and path planning.

Nodes

- *ekf_node*: implementation of a configurable EKF for complete pose estimation. It uses an omnidirectional motion model to project the state forward in time, and corrects that projected estimate using perceived sensor data.
- *ukf_node*: implementation of a configurable UKF for complete pose estimation. It uses a set of carefully selected sigma points to project the state through the same motion model that is used in the EKF, and then uses those projected sigma points to recover the state estimate and covariance. This eliminates the use of Jacobian matrices and makes the filter more stable. However, it is also more computationally taxing than ekf_localization_node.
- *navsat_transform_node*: it is a node to manage geographical coordinates. It produces an odometry message in coordinates that are consistent with your robot's world frame. This value can be directly fused into your state estimate. You will not need this node for this laboratory but you will learn more in a future lecture.

Parameters

The most useful and relevant parameters can be found in this commented example of configuration file.

For a complete reference about all the available parameters and node usage you can refer to the official documentation of the *robot_localization* package.

Exercise

In this exercise you have to tune the EKF/UKF implemented in *robot_localization* to track the position of your robot using only the odometer and an IMU.

The data are recorded and provided you inside two rosbags:

- **bag1**: this recorded path is regular and velocities are smooth
- **bag2**: this recorded path is more heterogeneous and velocities change abruptly in some cases

Setup

- Download the repository from GitHub **inside your src folder** with the command

```
git clone https://github.com/SESASR-Course/lab05.git
```

- Install the dependencies with rosdep

```
rosdep install --from-paths src --ignore-src -r -y
```

- Build the package with colcon

```
colcon build --symlink-install --packages-select lab05_pkg
```

Task 1

To play one bag (you choose which one inserting the correct path) with the command:

```
ros2 bag play <path_to_bag_folder>
```

While the bag is playing, in a second terminal read the covariance reported by the sensors from their topic messages:

- **IMU topic**: `/imu_broadcaster/imu`
- **Odometry topic**: `/diff_drive_controller/odom`

Report: which is the covariance of the two sensors? Report them with the correct measurement units.

Task 2

Run the EKF with the provided parameters (`lab05_pkg/config/ekf.yaml`) and evaluate its performances according to the following metrics with respect to `/ground_truth`. Compute these metrics both for the `/diff_drive_controller/odom`, and the `/odometry/filtered` topics.

```
ros2 launch lab05_pkg ekf.launch.py bag:=<path_to_bag>
```

Position (x,y):

- Root Mean Squared Error (RMSE)
- Maximum Absolute Error
- Total cumulative error, i.e. the sum of the error at each time step
- Error percentage of the final position relative to the total traveled distance

Orientation (yaw angle):

- RMSE
- Maximum Absolute Error
- Total cumulative error, i.e. the sum of the error at each time step

Publication frequency: actual vs. desired frequency of the `/odometry/filtered` topic.

Hint: you can modify and use the `recorder.py` node to gather all the data and save them. Then, you can load the `.npy` files and use NumPy to compute the required metrics.

Report: report in a suitable table the required metrics. Plot the timeseries of the position and orientation error. Plot the XY trajectory of the robot reported by `/ground_truth`, `/diff_drive_controller/odom` and `/odometry/filtered`.

Task 3

Fine tune the process noise covariance matrix, changing the values relative to linear acceleration, linear and angular velocity.

Hint: in first approximation it is not important the actual value of the covariance but its order of magnitude. It is suggested to try values ranging from 1e2 to 1e-4.

Report: insert in the report only your best result and justify the chosen parameters. The required metrics and results are the same exposed in Task 2.

Task 4

Create a copy of the launch file `lab05_pkg/launch/ekf.launch.py` and call it `ukf.launch.py`.

Modify the launch file you just copied to run the Unscented Kalman Filter instead of the EKF, using the same parameter file.

Run the UKF and evaluate its performance.

Report: compute the same metrics you used in Task 2 also for the UKF results. Compare the performance of the EKF and UKF using the same parameters.

Appendix

Root Mean Square Error

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(p_i - \hat{p}_i\right)^2}$$

Where p_i is the error of the i-th position from ground truth and $\hat{p}_i$ is the expected value (0 in this case).