ARS4 Project A24

Localization using poles and signs detected by a lidar

1 Objectives

Accurate positioning in urban environments remains a complex problem, despite the ever-increasing performance of GNSS systems.
A 360° vehicle-mounted lidar can be used to detect landmarks accurately geo-referenced on a map (cf. figure 1).
The aim of this project is to study a multi-sensor system that fuses lidar detections on geo-localized landmarks (signs and poles) with a GNSS receiver and kinematic sensors.

Figure 1: Photo taken on board an experimental vehicle of the Heudiasyc laboratory in Compiègne. One can clearly see poles and road signs that can be detected by a lidar.

2 Dataset Description

The data correspond to a specific section of a real dataset acquired in Compiègne in 2022 (cf. figure 2). The dataset is divided into two parts: simulated data and real data.

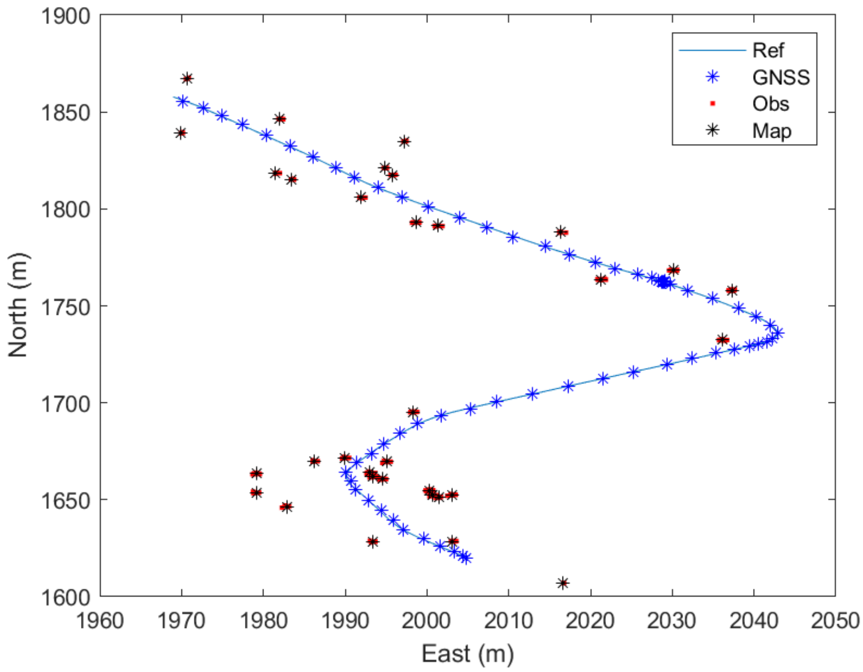

Figure 2: View of vehicle trajectory with map landmarks.

- **Simulated Data**: lidar observations are generated based on a high-definition vector map of Compiègne from 2021, which contains the 2D positions of all poles in the environment. Gaussian noise is added to simulate realistic detections. GNSS observations are derived from reference poses by adding noise as well. This dataset will be useful to develop Kalman filters in favorable conditions.
- **Real Data**: This part contains actual lidar detections obtained using two different methods: one based on intensity thresholding and another using machine learning approaches. This dataset will be useful to evaluate the performance of the developed filters in real conditions.

All data are synchronized to lidar timestamps (approximately 10 Hz), meaning that for any given timestamp, kinematic data, GNSS data (when available), and lidar data can be integrated into the filter. However, some data may be missing for specific timestamps.
For example:

- The GNSS data frequency is 1 Hz for both simulated and real datasets, meaning that GNSS data is unavailable for approximately 9 out of 10 timestamps.
- lidar observations may be absent due to missing map data in the field of view of the sensor or insufficient detections in the real data.

2.1 Simulation Data

The simulation data is structured as follows:

- **t**: Vector containing all observation timestamps (in seconds).
- **v**: Vector containing the longitudinal speeds (in m/s) for each corresponding timestamp in **t**.

- **omega**: Table containing angular velocities (in rad/s) for each corresponding timestamp in **t**.
- **gnss**: Structure containing simulated GNSS poses $(x(m), y(m), \text{heading(rad)}$ at 1 Hz (in a ENU working frame). If no data is available for a timestamp, the values are set to NaN. GNSS poses are derived from reference poses with added Gaussian noise:
  - Standard deviation: 0.2 m for x and y,
  - Standard deviation: 0.01 rad for the heading.
- **obs**: Structure containing simulated lidar observations at 10 Hz (aligned with all timestamps in **t**). For each timestamp:
  - x, y: Coordinates of the simulated lidar detections in the **mobile** frame.
  - x_map, y_map: Coordinates of the corresponding map elements in the ENU working frame.

  The simulated detections include Gaussian noise with a standard deviation of 0.1 m for both x and y coordinates. Please note that the data association between the lidar points and the map has been done in the structure.
- **ref**: Structure containing reference poses $(x, y, \text{heading})$ for each timestamp in **t** in the ENU working frame.

2.2 Real data

As previously mentioned, two lidar-based detectors are employed in this work:

1. **Intensity-Based Detector**: This detector identifies the most reflective surfaces in the environment, which typically correspond to traffic signs. It achieves this by retaining only the most intensive points from the lidar point cloud and clustering them to determine the centers of potential signs. However, false positives may occur, for example, due to highly reflective surfaces like license plates or other reflective structures. Besides, traffic signs can be missed
2. **Random Forest-Based Detector**: This detector classifies clusters of points as poles or non-poles using a machine learning approach. The process begins by removing ground points from the lidar point cloud through a ground segmentation algorithm. The remaining points are then grouped into clusters, and only those identified as poles are retained. The 2D coordinates of the centers of these pole candidates are extracted as detections. Detection performance is limited by the clustering approach and the training performance of the random forest leading to potentially misdetections and false positives.

To train the Random Forest classifier, automatic annotations were used. These annotations were derived from a high-definition map of Compiègne, which provides precise positioning information for poles during the mapping phase, and lidar point cloud segmentations computed on our datasets in the laboratory. It is important to note that this segmentation process is not real-time applicable.
Since the detections are not obtained by simulating them from the map data, the corresponding feature for each detection is unknown, and a data association stage must be added to identify the links between detections and mapped features.
The real data is structured as follows:

- **t**: Vector containing all observation timestamps (in seconds).

- **v**: Vector containing the longitudinal speeds (in m/s) for each corresponding timestamp in **t**.
- **omega**: Table containing angular velocities (in rad/s) for each corresponding timestamp in **t**.
- **gnss**: Structure containing GNSS poses $(x(m), y(m), \text{heading(rad)}$ at 1 Hz provided by a Septentrio Mosaic X5 gnss receiver in the ENU working frame. If no data is available for a timestamp, the values are set to NaN. In the structure, estimation of x, y and heading variances are also provided.
- **poles_obs**: Structure containing lidar detections using the Random-Forest Based detectors at 10 Hz (aligned with all timestamps in **t**). For each timestamp:
  - x, y: Coordinates of the lidar detections in the **mobile** frame.
- **signs_obs**: Structure containing lidar detections using the Intensity-Based detectors at 10 Hz (aligned with all timestamps in **t**). For each timestamp:
  - x, y: Coordinates of the lidar detections in the **mobile** frame.
- **map**: Matrix containing the coordinates of the poles and signs in the map $(x, y$ in the ENU working frame.

2.3 Programs provided

In the simulation kit, you'll find several programs to load data or to help you develop your own programs.

3 Work to do

Propose a stochastic state space representation that allows all available sensors to be fused together (state vector, input, output, evolution model, observation model and noises)[1]. Next, develop an extended Kalman filter, which you will first test in simulation. Tune it for 95% consistency.
To apply the filter to real data, you need to implement a data association algorithm to associate lidar measurements with map landmarks. Explain the method(s) you've implemented. Report filter results on real data and analyze them.
In a second stage, implement a UKF on simulated and real data.
Compare the two filters in this problem.

4 Final report and deliverable

Write a short report and provide your programs in a zip file. Upload them on the Moodle of the course.
Deadline:

- Wednesday, January 22 at midnight at the latest

Prepare an oral presentation with few slides for the 23rd of January.
On the day of the defense, please provide two paper versions of your report.

[1]We recommend that you take a look at the ARS4 A23 final exam available on the course Moodle.