EKF-Based Localization with LRF

Group 9 — Isabel Silva (73169), Renato Silva (73279), João Rosa (74149), Vítor Alveirinho (77003), José Mendes (82258), Maciej Przydatek (83022)

MEEC, Autonomous Systems - Instituto Superior Técnico, Universidade de Lisboa

Introduction

Localization is a fundamental problem in mobile robotics. In order for the robot to be autonomous, it needs to know its own pose in the environment. Localizing the robot using only data provided by odometry is inaccurate, since the measurement noise is constantly accumulating. Taking this into consideration, the robot will need to improve the information about its pose, which can be achieved by comparing the current environment scan with an already built global map, subsequently resulting in a better pose estimation. This map is then subjected to a post-process that transforms all its boundaries (mainly walls) into line segments, through a method of Least Squares. The environment is scanned with the aid of a Laser Rangefinder (LRF).

This localization problem can be split in two phases: the Prediction Step and the Update Step. The Prediction Step of the EKF is performed by simulating the kinematic model of the robot. The standard deviation of each of the wheels' angular speed of the robot is estimated as being proportional to the wheels' angular speed. In the Update Step of the EKF, the main objective is to correct the pose of the robot by subtracting the parameters of matching lines from local and global maps. For this we need to find the line segments from both the local and global maps, applying a matching procedure, where the most similar local and global line segments are paired if the difference is below a threshold.

Tools

The work was conducted using Adept MobileRobots PIONEER 3-AT, SICK LMS200 LRF and ROS. The Robot Operating System (ROS) is a set of software libraries and tools that help in building robot applications. Its modularity, extensibility and standardization allow system designers to cooperate flawlessly, while working separately. ROS consists of four main parts: **Roscore** - main framework and base program; **Nodes** - modules, which perform specific tasks; **Topics** - allow nodes to communicate between each other; **Parameter Server** - set of values, which can be read, set or deleted by every node. The following nodes were used: `rosaria` - allows to establish a connection between ROS and PIONEER 3-AT; `sicktoolbox_wrapper` - performs communication with SICK LMS200 Laser Rangefinder; `gmapping` and `map_server` - implementation of SLAM (Simultaneous Localization and Mapping); `rosaria_client` - allows to easily control the robot's velocity using the computer keyboard; `rviz` - visualization tool; `tf` - manages multiple coordinate frames transitions during program execution.

Figure 1: Pioneer 3-AT

Methods

An Extended Kalman Filter consists of two main parts: a prediction step and a correction step. Between them, there is a matching step inserted. Its purpose is to filter out such laser scan measurements that do not correspond well to the view calculated in the prediction step.

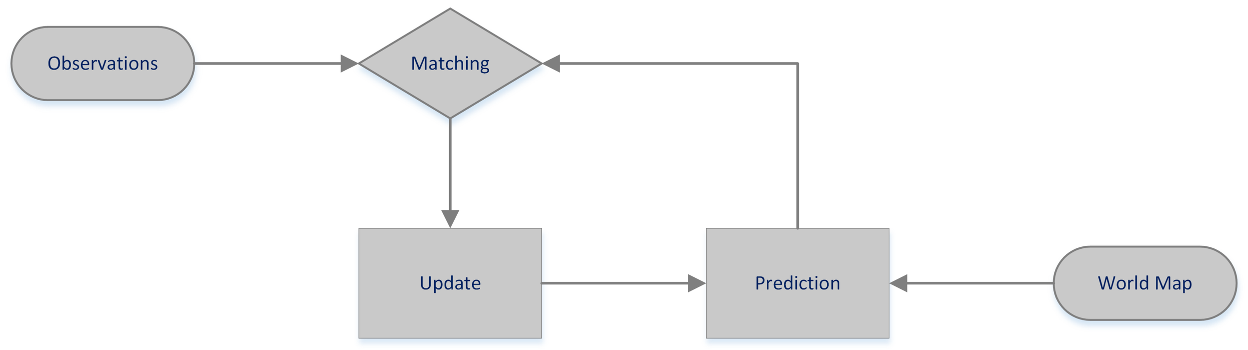

Figure 2: Simplified Extended Kalman Filter diagram

- **Prediction step** - the robot's pose is predicted by simulating the discrete kinematic model of the robot, taking into account the odometry errors;
- **Matching step** - measurements from the robot's new position are matched with measurements from the robot's predicted position;
- **Update step** - the robot's pose is corrected based on matches from the matching step.

Results

The first step in the implementation of an Extended Kalman Filter is to create a visual representation of the environment where the robot is in. A global map was created using a SICK LMS200 Laser Rangefinder and the gmapping SLAM package. The boundaries of this map (in black) are then transformed into line segments.

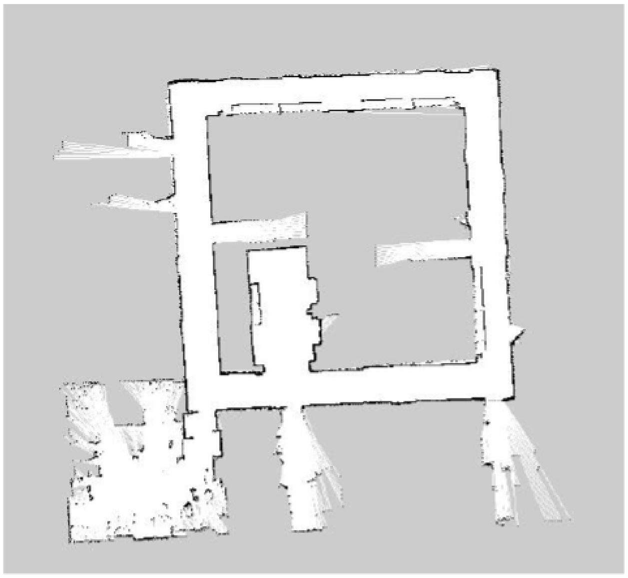

Figure 3: Global Map

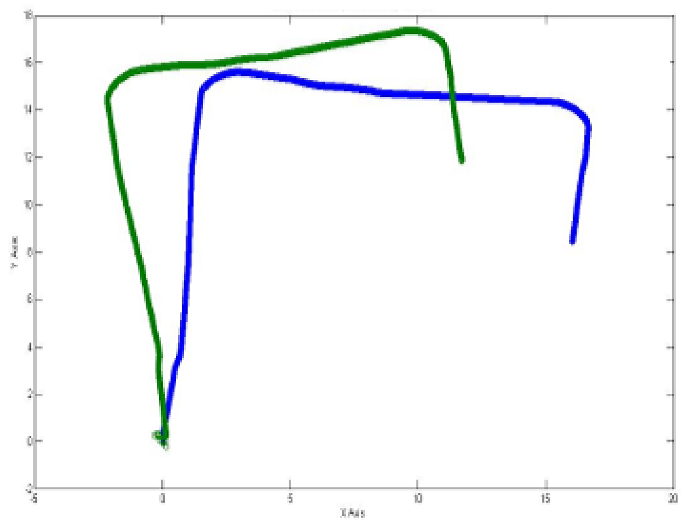

Figure 4: Early results

Initial tests, representing the trajectory made by the robot. The points in blue are the values obtained through odometry. The points in green are the values obtained with the EKF. There is a small amount of matched points, but each trajectory attempts to follow the path of the global map.

Further tests were made in order to obtain better results. It can be observed that EKF readings are almost accurate, while the odometry suffers from some miscalculations in one of the corridors of the floor. Even though, the number of matched points from both readings is significant.

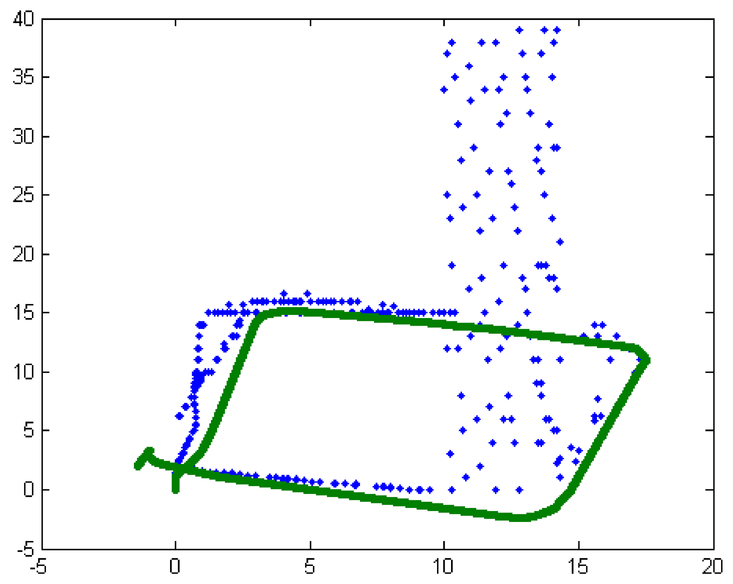

Figure 5: Results after code improvements

Conclusion

In this project, an EKF algorithm is implemented in order to achieve the most accurate localization of a mobile robot with four wheels. The robot navigates with the help of a global map of the fifth floor in the North Tower of Instituto Superior Técnico, obtained with scans from a Laser Rangefinder.

During the whole project, we faced several challenges. All of the algorithm was implemented using Python and the Robot Operating System (ROS), which were tools hard to grasp for us at the start, since none of the elements of the group knew them before the beginning of the academic term. Another problem arised during the creation of clusters of points from the global map. Using the Corner Harris detection method, the lines created were not perfect. So, in order to achieve the maximum similarity, we did the extraction of the first and last points of each line by observing the map.

References

[1] I. Teslic, L. Škrjanc and G. Klančar.
Ekf-based localization of a wheeled mobile robot in structured environments.
J Intell Robot Syst, 2010.

[2] M. Harris, C. Stephensčar.
A combined corner and edge detection.
1988.

[3] P. Lima.
Autonomous systems lecture notes.
2015.

TÉCNICO LISBOA