Differential Drive Mobile Robot Simulation and Dead Reckoning

Loc Pham, and Mohamed Khaled

November 6, 2023

Abstract

This lab report presents the implementation of a Grid Localization method for the Differential Drive Mobile robot using as input the robot velocity obtained form its dead reckoning and using distance measurements with respect to an a priori known map of two dimensional Cartesian features.

1 Introduction

The objective of this lab was the understanding the Grid Localization method for the Differential Drive Mobile robot. For more organized and easy implementation, a GitHub repository had been used; which contained blueprint classes designed for robot simulation.

2 GL: Part 1 (Update)

In this lab report, a detailed methodology is illustrated to implement the *Update* step of the filter. To achieve this, the distances towards the three Cartesian landmarks are read from simulated robot and added a Gaussian noise before used as an observation of the *Update* step.

2.1 Reading the distance towards the landmarks

Program the method named *ReadRanges()* within the class *DifferentialDriveSimulatedRobot*. Particularly, this function simulates the sonar reading of the robot which measures range from the robot to the landmarks and the covariance of its noise. We computed these ranges from the location of the robot and the landmarks.
Program the method *GetMeasurements()* of the *GLDifferentialDrive* which will call the previous one to get the distance to the nearby features.

2.2 Check the HF class

We read the documentation of the HF class and check its implementation. This is a base class already programmed. We have a *main* function in the *HF.py*, we used it to test methods and properties of the *HF* class.

2.3 Check the GL class

1. **LocalizationLoop:** This is the main loop of the program. In this loop, we simulate the robot motion by *fs* method, get measurements from robot (range from the robot to the landmarks mentioned above) and use these measurements for the *Update* step of the filter. Besides, in this loop, we plot the updated grid histogram of the localization system.
2. **Localize:**
3. **MeasurementProbability:**

2.4 Test the Update step

3 Issues

During implementation, there are two issues we encountered: Encoder modeling and System modeling. These issues influence the accuracy of the localization system and the error of Dead-reckoning problem besides the noise of the sensors.

4 Conclusions

In conclusion, through the lab, we have an overview of a robot simulation system including simulation components and localization components. More specifically, with the simulation block, we need to model the robot and its sensors, while the localization block requires computing the robot's state from the sensor's measurement signals. With the localization system being used, there is a Dead-reckoning problem when noise is added to the encoder sensor's measurement signal. Besides, the Dead-reckoning problemproblem is also the result of encoder sensor modeling and robot system modeling.

References

[1] Pere Ridao. "MR1-23-24-Introduction". Version 0.1. October 02, 2023

[2] Pere Ridao. "Probabilistic Robot Localization and Mapping". Version 2.0. September, 2022

[3] Pere Ridao. "prpy: Probabilistic Robot Localization Python Library". Version 0.1. October 02, 2023