SLAM - Implementation and Benchmarking

Mangal Deep .B.M

7206937

Submitted in partial satisfaction of the requirements for the degree of

M.Eng Embedded Systems for Mechatronics

School of Informatics

Dortmund University of Applied Sciences and Arts

15-3-2021

A blank page

This is an optional page environment you could use for things like:

- Your own custom preamble chapters (use `\chapterTitle` for titles!)
- *"This work is dedicated to Dad and Mom"*
- A copyright notice
- Additional notes
- Quotes
- List of publications
- An actual blank page
- Nomenclature / glossaries, etc.

It is not recommended to use this in your undergraduate thesis for submission. It has been left in here to make you aware of its existence. This is largely because it does not follow the guidelines set out for undergraduate theses, but you could include this environment for your own personal printed copy. Consult with your supervisor to see if you can use it.

In addition, you can pass an optional parameter to this environment with the value `c` to centre this text vertically on the page (use the `center` environment to align horizontally).

Abstract

This is where your abstract will go. Usually this is written last, after writing the entirety of your thesis.

Acknowledgements

Firstly, I want to thank Dad and Mom.[1] Here is another note.

[1]Here is a footnote

Table of Contents

Chapter 1

Introduction

Simultaneous Localization and Mapping, commonly abbrevated as *SLAM* is one of the fundamental task of any mobile robots which helps in building the Map of the environment without the knowledge of actual location and orientaion of the robot. It is one of the key initial step for the development of navigation systems for any mobile robot. It is also refered as the chiken or the egg problem, as it requires precise locaion of the robot in the environment, to get an accurate map of the environment,but in order to determine the precise locaion, the map of the environment is required. This complexity has made it harder to find a precise solution, accurate localization and mapping.

The complex task of localizing and mapping the surrounding has triggered a huge intrest widely in the research filed and it continues to improve as the technology evolves in both software and hardware. Over the past decade numerous stratergies and methods were developed to solve the SLAM problem.

A typical SLAM system uses a combination of odometry data, GPS ,IMU for localization and Camera, Lidar or Radar for perception. The information obtained from these sensors are fused to obtain a full estimate on the state of the system.Though technologies evolved in accquiring precise measurement uncertainities and errors are still prevelant due to various factors. Hence designing a solution considering the uncertainities in the measurement is a key to achieve a good estimate of the location

and map of the surrounding. Therefore a probabilistic approach to the solution is the key to globally consistent Map.

The SLAM problem can be classified on various basis.SLAM problem can be broadly classified into *Online SLAM* and *Full SLAM*. In the *Online SLAM* problem,only the current state of the robot is estimated along with Map. In the *Full SLAM* problem, the entire state of the path traversed by the robot is estimated.Based on the setup of environment to the mapped the SLAM system can be braodly classified into *Landmark-based SLAM* and *Grid-based SLAM*. In *Landmark-based SLAM*, the robot knows the locaion of landmarks *a priori*. However in the real world scenario, the robot should establish a correspondence between the expected landmark maeasurement and the actual landmark observed. The data association plays a key role in assigning the measurement to its corresponding landmark. With the known correspondence and known pose of the robot relative to a landmark, the problem reduces to Mapping with known poses. The *Landmark-based SLAM* is widely applied in many robotics application for instace, underwater exploration, mapping the underground mines. These SLAM system commonly use Ultrasonic sensors, camera for perception and GPS, odometry are used to find the motion of the robot.

In absence of landmarks, *Grid-based SLAM* provides a full SLAM solution with the help of 2-D or 3-D laser scanners. The observations made by the LiDar are converted to a obstacle map also referred to as occupancy map. These observations are also used to correct the pose estimate of the robot. This SLAM approach provides a more useful way to map the surroundings as landmarks cannot be installed and observed in many challenging environment. More details on this approach will be provided in the later chapters.

Numerous research has been done in the SLAM community. Some of the implementations are made available to the public in ROS. The most common are GMapping and Cartographer

This paper presents insights on Particle Filter based and Graph based solutions to SLAM problem. These approaches are implemented, tested and results are com-

pared thoroughly to provide detailed report on throughput of the algorithms. The algorithm is desgined and the testing is performed on the data set obtained from CARLA. The paper is organized as follows, The detailed literature review is provided on exisiting SLAM algorithms and applications, variants of SLAM in chapter 2. Probabilistic formulation of the SLAM problem is derived in chapter 3. Particle Filter based solution to SLAM is discussed and different implementations such as Rao-Blackweillized Particle Filter and GMapping algorithm are dicussed in detail and analyzed in chapter 4. Graph based solution to SLAM is dicussed and its various flavors are discussed in detail in chapter 5. Comparison and benchmarking of the above methods are performed based on test results and computation is provided in chapter 6. Concluding remark on all the observations made so far in the paper is discussed in chapter 7.

Chapter 2

Literature review

Another cool thing about LaTeX is its referencing system. This template is set up to use harvard-style referencing. You can do this by using `\citep{citekey}`. It will print out something like this: [**?**]. Or alternatively, you can use `\cite{citekey}` to cite things like this: [**?**]. This template uses BibLaTeX for referencing, with a Biber backend. This is primarily due to the extensive features BibLaTeX provides, along with the option of glossaries. If you want to customise the referencing style, you can either modify the template slightly to use different options, or use `\usepackage` again to reimport it. There's probably some commands to change its options after its been imported too.

2.1 Ludography

This thesis template also contains an optional ludography. This is primarily for Games Development students, who wish to cite games in their thesis. To use this, just put references into your bib file as usual with the game's details. Then, make sure `keywords` is set to `{game}`. This is what is used to determine which references are games, and which are actual papers. For a more elaborate example, see `bib/ludography.bib`.

Also, make sure that the `title` key is actually the author of the game, and the `author` is the title of the game. The reason this is swapped around is because BibLaTeX likes to print references out with the author first. Then, just add `\printLudography`

with an optional title argument to print out all citations like `\printLudography` or `\printLudography[Games]`.

You can also use the `ludography` environment if you wish to print out some text before the list of games is printed. An example of this can be seen in `main.tex`. To cite games, you can `\cite` it like any other reference. However, if you want it to display the title instead of the standard referencing style, you can use `\citeGame` instead.

Here is an example of a cited game with a normal reference style: [**?**]. Ugh, pretty ugly. Instead, here the two are cited in the next sentence as games with `\citeGame`. Both **?** and **?** were games made by Atari. Much better!

Chapter 3

Formulation

3.1 Introduction

3.2 Probabilistic approach to state estimation

3.3 State Prediction

3.3.1 Odometry Motion Models

3.3.2 Kinematic Motion Models

3.4 State Correction

3.4.1 Observation Models

3.4.2 Scan Matching

3.5 Summary

Chapter 4

Particle Filter

4.1 Introduction

4.2 Particle Filter

4.2.1 Sample

4.2.2 Import weighting

4.2.3 Resample

4.3 Rao-Blackwellized Particle Filter

4.4 gMapping-Improved RBPF

4.5 Results and Analysis

4.6 Summary

Chapter 5

Factor Graphs

Chapter 6

Comaprison and Benchmarking

Chapter 7

Conclusions

Chapter 8

Reflective Analysis

Appendix A
Some random python code

This template includes the `minted` package, which allows you to import code and syntax highlight it. For example, the text below is imported directly from the `code/test.py` file using the `\inputminted` command:

And here is a snippet of Python with the `minted` environment:

Minted supports many, many languages – so you're not just limited to Python. For example, here's some random C++ code.

Appendix B
Some other stuff

Here is just an example of some other stuff.