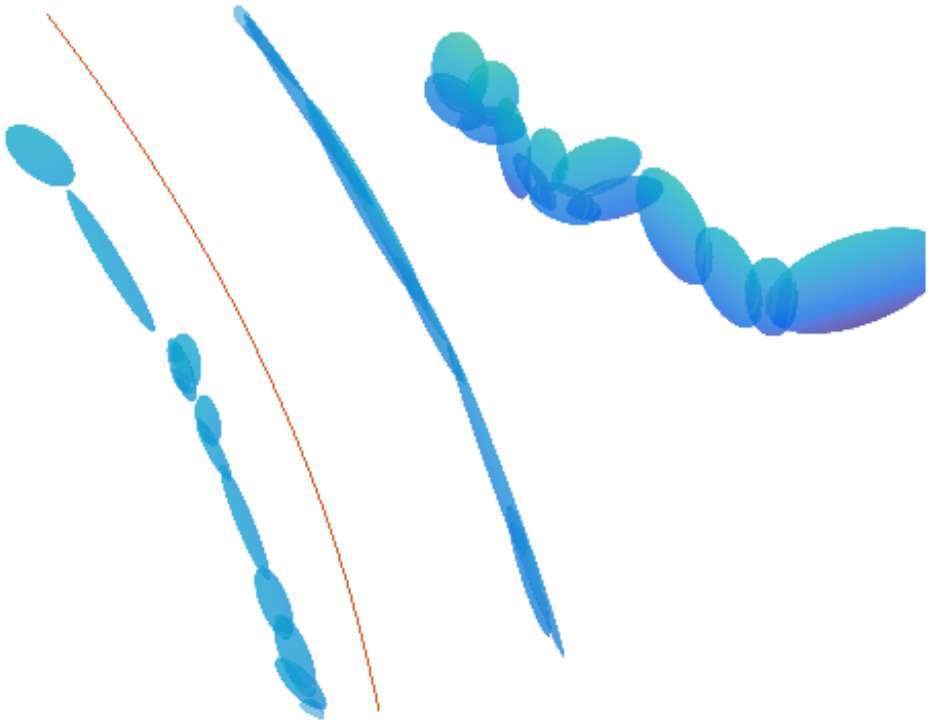
CHALMERS
UNIVERSITY OF TECHNOLOGY



Mapping and localization using automotive lidar

Poisson Multi-Bernoulli mapping and Marginalized particle filter localization

Master's thesis in systems control and mechatronics

Marcus Olsson
Pontus Kielén

Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Master's thesis EX034/2017

Mapping and localization using automotive lidar

Poisson Multi-Bernoulli mapping and Marginalized particle filter localization

Marcus Olsson

Pontus Kielén

Department of Signals and Systems
Division of Signal Processing and Biomedical Engineering
Signal Processing Group
Chalmers University of Technology
Gothenburg, Sweden 2017

Mapping and localization using automotive lidar
Poisson Multi-Bernoulli mapping and Marginalized particle filter localization
Marcus Olsson
Pontus Kielén

Supervisor: Johan Degerman, SafeRadar Research Sweden AB
Examiner: Karl Granström, Department of Signals and Systems

Master's Thesis EX034/2017
Department of Signals and Systems
Division of Signal processing and biomedical engineering
Signal processing group
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Section of the created map, with guard rails and 3 building corners together with the reference trajectory.

Typeset in LaTeX
Gothenburg, Sweden 2017

Mapping and localization using automotive lidar
Poisson Multi-Bernoulli mapping and Marginalized particle filter localization
Marcus Olsson
Pontus Kielén
Department of Signals and Systems
Chalmers University of Technology

Abstract

Vehicles that are capable of automating different aspects of the driving task are becoming increasingly potent. Lane keeping assist and autonomous cruise control systems are already capable of handling some parts of the driving task autonomously. One of the most challenging and vital parts of autonomous vehicle design is the problem of locating the vehicle with respect to the road and its surroundings. This is important in order to support lane-keeping, path planning, and prediction of other road users for threat assessment. The performance of these systems are often limited by the mapping and localization accuracy. In the following thesis, this problem is addressed and the proposed solutions are tested using real data recorded at a test track in Trollhättan Sweden. A test vehicle is equipped with a front facing lidar, inertial measurement unit (IMU) sensors, and a differential global positioning system (DGPS) used as reference.
A map consisting of landmarks is created using the lidar measurements, where the landmarks are objects represented by ellipsoids. A mapping algorithm makes use of Gibbs sampling to find the most likely data association, given the asumptions that the existence of landmarks is Bernoulli distributed and that measurements appear according to a Poisson point process. The map is then used together with new lidar and IMU measurements in a marginalized particle filter used for localization. The states estimated from the localization filter is position, heading, turn rate and speed. Training and test data is collected in a rural environment, where localization can be hard. When evaluating the localization accuracy, the position is compared to the DGPS reference system. The resulting accuracy during localization stays below 0.16 m lateral error and 0.6 m longitudinal error in this scenario. Furthermore, the system is also capable of finding the correct position when the initial position is known with a few meters deviation. It is shown that the proposed method for mapping is capable of creating accurate maps for localization even in sub-optimal rural environments with few distinct landmarks. The method is also capable of producing accurate maps with a relatively low number of landmarks, meaning that it is suitable for mapping of larger areas.

Keywords: Lidar, marginalized particle filter, Poisson multi-bernoulli mixture, Mapping, Localization.

Acknowledgements

We would like to thank Denso Sales Sweden's active safety team for the sensor data used in our work and our supervisor Johan Degerman for valuable input and also our examiner Karl Granström for some theoretical background and discussions.

Marcus Olsson and Pontus Kielén, Gothenburg, June 2017

Contents

1 Introduction **1**
1.1 Purpose, Objective and Contributions . . . 2
1.2 Thesis outline . . . 2

2 Problem description **5**
2.1 Scope . . . 5
2.2 System setup . . . 6
2.3 Simulation environment . . . 6
2.4 Test scenario . . . 6

3 Background theory **9**
3.1 Bayesian filtering . . . 9
3.1.1 The Kalman filter . . . 10
3.1.2 Particle filters . . . 11
3.1.3 Marginalized particle filters . . . 13
3.2 Mapping . . . 15
3.2.1 Poisson Multi-Bernoulli mapping with random finite sets . . . 15
3.2.1.1 Multi Bernoulli mixture . . . 16
3.2.1.2 Poisson point processes . . . 16
3.2.2 Gibbs sampling . . . 17

4 Models **19**
4.1 Vehicle model . . . 19
4.2 Measurement models . . . 20

5 Mapping methods **23**
5.1 Poisson Multi-Bernoulli mixture . . . 23
5.1.1 Mapping models . . . 23
5.1.1.1 Clutter . . . 24
5.1.1.2 Undetected landmarks . . . 24
5.1.1.3 Detected landmarks . . . 25
5.1.2 Partitions . . . 25
5.1.3 Landmark representation . . . 26
5.1.4 Computing the weight of a partition . . . 27
5.1.5 Sampling the partitions . . . 30
5.1.6 Batch mapping . . . 31
5.2 Proximity landmarking . . . 33

5.3 Implementational aspects 34

6 Localization method **37**
6.1 Marginalized particle filter 37
6.1.1 Prediction and measurement update 38
6.1.2 Expected number of measurements 39
6.2 Implementational aspects 40

7 Results **43**
7.1 Data sets 43
7.2 Map validation 44
7.2.1 Simulated data 44
7.2.2 Test track data 46
7.3 Localization performance 46

8 Discussion **51**

9 Conclusion **53**

Bibliography **55**

1

Introduction

The field of advanced driver-assistance systems (ADAS) and autonomous driving (AD) is an increasingly popular research area, in both the vehicle industry and academia. By adding the capability of making decisions and planning a safe route for the vehicle, the goal is to avoid problems caused by human factors, and also avoid situations where the human abilities are insufficient. An immediate benefit of these systems is the safety factor. According to the world health organization[1]. road traffic injuries are the number one cause of death among ages 15-29. A report published by the U.S. Department of Transportation shows that over 90% of all road crashes are partly, or fully, caused by human factors[2]. This means that even marginal improvements in this area, through ADAS and AD systems, can have a large impact on the number of saved lives, if implemented on a global scale.

Several researchers also believe that autonomous vehicles can have a positive effect on the environment through different kinds of rental and taxi services, see, e.g, [3],[4]. The researchers believe that personal vehicles can be replaced by small autonomous electric vehicles with optimized daily usage and optimized energy consumption. This would reduce the number of vehicles needed and thereby the emissions caused by the vehicle fleet.

The focus in this thesis is to investigate one of the crucial tasks of an autonomous vehicle, namely the positioning of the car. This task needs to be robust and should be able to work even when some sensors fail. Commonly used sensors for positioning include GPS, cameras, and Inertial measurement units (IMU). These are sensors that are prone to disturbances or complete failure. The GPS only works when there is a clear line of sight, and fails, e.g., when there are tall buildings or other obstacles in the way. In cities with tall buildings, the multi-path effect is also a big challenge for the GPS. The camera relies heavily on the lane markings, and is very sensitive to lighting conditions. Thus, redundancy is needed to safely navigate a vehicle.

This thesis have been carried out in cooperation with DENSO Sales Sweden AB, which provided sensor data. A lidar (light detection and ranging) sensor is used together with an accurate reference DGPS sensor that estimates the vehicle pose, to create a map of the environment. The localization is then performed using only the onboard IMU, together with a front facing lidar sensor and the created map. Both the mapping and localization problems are solved using probabilistic methods.

1.1 Purpose, Objective and Contributions

The purpose of this thesis is to implement a Bayesian probabilistic mapping method using random finite sets, and a Bayesian probabilistic localization method using a marginalized particle filter. The methods are tested using lidar data collected from a car in a country-side environment, and the results are evaluated to see if the proposed mapping and localization approach is desirable for this type of sensor and scenario.

The objectives of the thesis are the following:

- Create a map of the environment using a lidar sensor and a reference position sensor.
- Implement a localization system to position the vehicle using only IMU, lidar and the previously created map.
- Evaluate the mapping and localization methods used, to see if they are able to position a vehicle within the specified maximum deviation. The chosen specifications are based on previous localization work[5], and are to position the vehicle within 0.2 m laterally and 1 m longitudinally from the DGPS reference position.

The main contributions in this thesis are the following:

- An investigation regarding how a Poisson multi-Bernoulli mixture (PMBM) filter performs in mapping, evaluated on real lidar sensor data from a country-side road scenario.
- Performance enhanchements to an already existing PMBM mapping algorithm.
- An evaluation of the localization performance of a marginalized particle filter, together with a PMBM map, in a country-side road scenario.
- A discussion regarding the viability of the methods being used in autonomous vehicles in these type of scenarios.

1.2 Thesis outline

The thesis starts with a problem description that describes the system setup, the simulation environment and the collected data. Chapter 3 describes the theoretical background that the thesis is based on, which includes models used in mapping and filters used in localization. The different models used for describing the vehicle and the sensor dynamics are described in Chapter 4, and the methods implemented

for mapping and localization are described in Chapters 5 and 6. The results are presented in Chapter 7 and are discussed in Chapter 8. The thesis ends with a discussion of the results and a conclusion in Chapter 9.

2

Problem description

The problem of mapping in the context of AD and ADAS systems is to describe the environment in a way such that it is possible to use the created map for localization in this environment. There are two main approaches to representing a map, occupancy grid maps and landmark maps[6]. The method proposed in this thesis uses the landmark approach. When using landmarks to describe the environment, the goal is to estimate the state of each landmark, and also estimate the total number of landmarks. Given a set of measurements a challenge with landmark mapping is to solve the data association, i.e., associating each measurement to the landmark from which it originates.

Localization is the problem to describe the position and orientation of the vehicle using sensor measurements. In this thesis the measurements will come from an IMU sensor, wheel speed sensor, and a lidar sensor. Although these tasks can be solved simultaneously, i.e., simultaneous localization and mapping (SLAM), we have for simplicity separated the two problems. However, with the framework used in this thesis it is possible to connect the mapping and localization.

2.1 Scope

The scope of this thesis is to use a front facing lidar sensor to investigate a mapping algorithm and a localization algorithm. Both algorithms that are investigated use Bayesian inference and random sampling. The data used for both algorithms are collected on a test track with the lidar mounted to a test vehicle. The map, represented by extended objects in three dimensions, is then created offline to later be used in the localization algorithm.

The localization algorithm is designed under the assumption that the initial position is known with a few meters uncertainty and that a map exists beforehand. The position estimated by the algorithm is limited to the horizontal plane to improve computational efficiency. Information about the vertical height of the sensor or the road can be included in the map, but the results should not differ significantly in this scenario, because the environment is very flat.

2.2 System setup

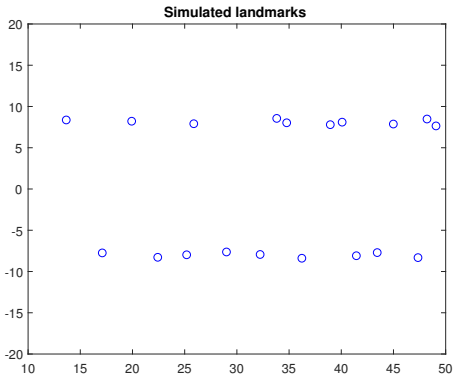
The system used in this thesis is provided by DENSO Sales Sweden AB (DSSE), and consist of a test vehicle equipped with a front facing lidar and IMU, and a reference sensor. The lidar gives range, bearing and elevation measurements. The IMU provides yaw-rate as well as wheel-speed. The reference position sensor, an Oxford RT 2000, contains a DGPS with 0.02 m standard deviation, and a precise IMU with below 0.1 degrees standard deviation in roll, pitch and yaw.

2.3 Simulation environment

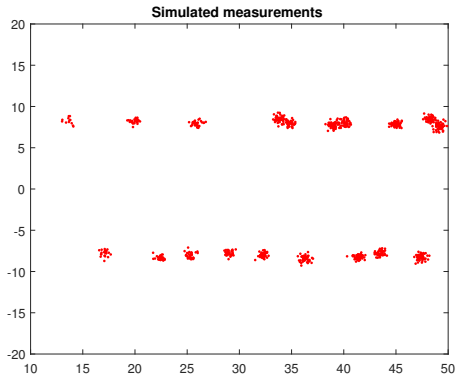
A simulation environment is used to verify that the mapping algorithm works as intended. The environment is created by using point sources placed next to a road; the points return measurements as the simulated vehicle passes them. The measurements have measurement noise in the x, y and z direction according to a normal distribution with standard deviation 0.1 m. The simulation is not designed to reproduce lidar measurements in particular, they are instead designed to match the proposed landmark model. The measurements should ideally be mapped to the landmark from which they are originating from, to get an as accurate representation of how the sensor perceives the landmark as possible. An illustration of the simulated environment can be seen in Figure 2.1.

2.4 Test scenario

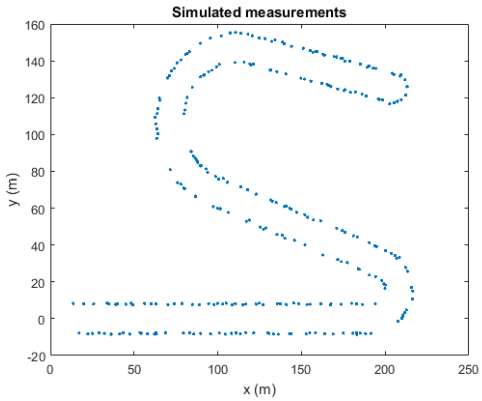
The data acquisition is carried out on the National Electric Vehicle Sweden (NEVS) test track in Trollhättan. The local environment at the test track contains a few houses, guardrails, some bushes and some trees. An areal photo of scenario can be seen in Figure 2.2a, and in Figure 2.2b the created three dimensional map is projected onto a two dimensional satellite photo of the scenario. The car was driven on the road closest to the four white houses. The two uppermost houses were not present during the data acquisition, and are therefore not represented in the map. Two data sets from the same environment are available, one data set is used for producing the map, and the other is used for localization in the created map.



(a) Landmarks from a small area of the simulated environment.

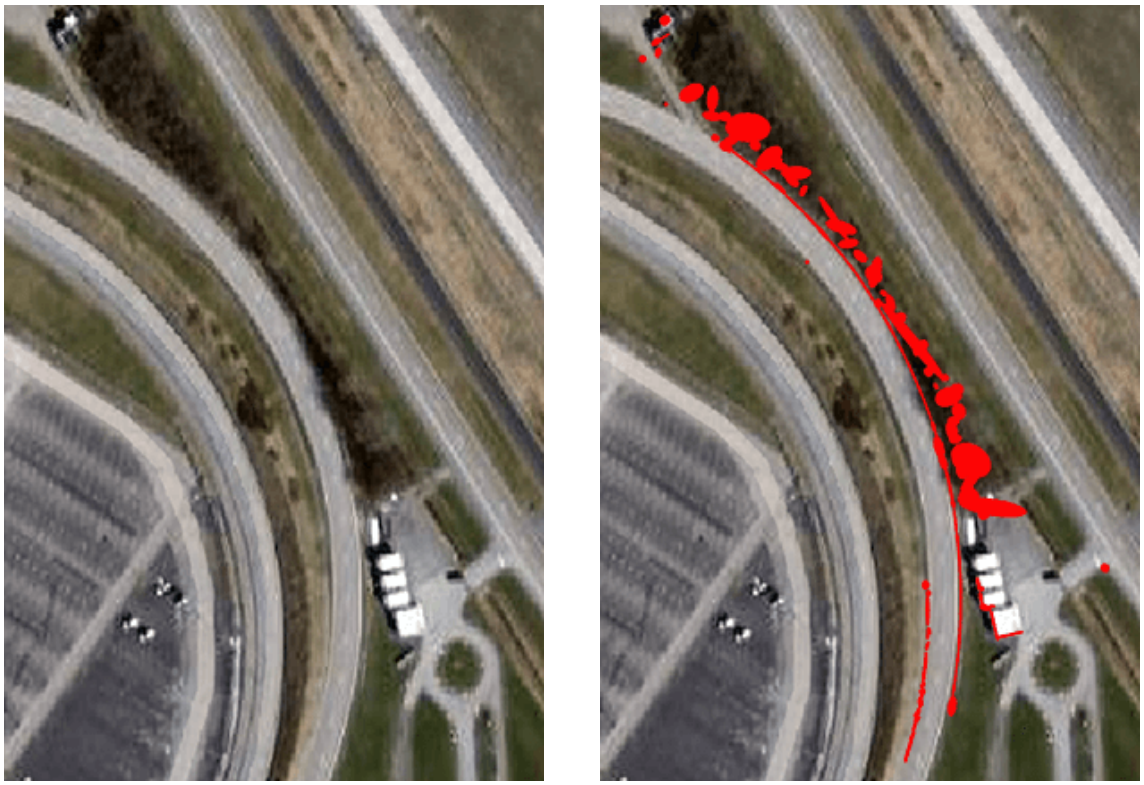


(b) Measurements from a small area of the simulated environment.



(c) Overview of the simulation environment

Figure 2.1: The simulated environment contains a set of landmarks and a set of noisy measurements originating from the landmarks. Note that both the landmarks and measurements are represented in three dimensions.



(a) An aerial photo of the testscenario in Trollhättan. From [7].

(b) The 3 dimensional map projected onto the scenario. Note that the real map is in three dimensions. From [7].

Figure 2.2: An overview of the test scenario. Note that structured objects like the walls of the houses and the guardrail are mapped as very distinct landmarks, while unstructured objects like trees and bushes are mapped as larger and less distinct landmarks.

3

Background theory

This chapter introduces different theoretical concepts used in the thesis, for the purpose of priming the presentation of the methods described in chapter 5 and 6. An overview is given of Bayesian filtering in general, and specifically of Kalman filters and particle filters which are used for localization. The statistical models used for modelling the map and its landmarks are also presented.

3.1 Bayesian filtering

In Bayesian filtering the aim is to recursively estimate some state of the surrounding world. A detailed description of the Bayesian filtering problem is given in, e.g., [8, p.17-20]; the following is a brief overview of the problem.

Each state vector $\mathbf{x}_k$ describes some properties of the world at time index k. The states are estimated given a series of measurements

$$\mathbf{z}_{1:K} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_K\}. \tag{3.1}$$

Both the states and the measurements are assumed to be stochastic, which is desirable when dealing with imperfect measurements and models. The posterior, $p(\mathbf{x}_k|\mathbf{z}_{1:k})$, is the probability density function of the current state $\mathbf{x}_k$ given all the measurements up until time k. Using Bayes' theorem, the posterior density can be derived as

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})}, \tag{3.2}$$

The likelihood $p(\mathbf{z}_k|\mathbf{x}_k)$ is given by a measurement model, and the prior is denoted $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$. Lastly, $p(\mathbf{z}_k|\mathbf{z}_{1:k-1})$ is a constant normalization factor. Not considering the normalization constant, the posterior density can be achieved from the process model and the measurement model proportionally as

$$\text{posterior} \propto \text{prior} \times \text{likelihood}. \tag{3.3}$$

To complete the iteration the predicted state, i.e., the new prior, is described by the Chapman-Kolmogorov equation,

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})\, d\mathbf{x}_{k-1} \tag{3.4}$$

where $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is the process model, and $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ is the posterior distribution from the last step.

3.1.1 The Kalman filter

The process model and the measurement model are often written as

$$\mathbf{x}_k = f_{k-1}(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}) \tag{3.5}$$
$$\mathbf{z}_k = h_k(\mathbf{x}_k, \mathbf{w}_k) \tag{3.6}$$

where $\mathbf{v}_{k-1}$ is the process noise and $\mathbf{w}_k$ is the measurement noise. If both the process and measurement model are linear with additive i.i.d. Gaussian noise then the Kalman filter (KF) performs optimal filtering. To indicate that the models are linear we write them as

$$\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{v}_{k-1} \tag{3.7}$$
$$\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{w}_k. \tag{3.8}$$

If the prior distribution $p(\mathbf{x}_0)$, the process model, as well as the likelihood, are assumed to have normally distributed noise, then the posterior distribution is also normally distributed for all times, and can be expressed as

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_k, \mathbf{P}_{k|k}), \tag{3.9}$$

where $\hat{\mathbf{x}}$ is the estimated state vector and $\mathbf{P}$ is the state estimate covariance matrix. Similarly to the general case, the Kalman filter can be divided into two steps: the prediction step and the update step. A detailed derivation of the Kalman filter is presented in e.g., [9]. The predicted mean and covariance are calculated by propagating the mean and covariance from the previous time step through the process model,

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}\hat{\mathbf{x}}_{k-1} \tag{3.10}$$
$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}, \tag{3.11}$$

where $\mathbf{Q}$ is the state noise covariance matrix and the notation $k|k-1$ is used to indicate that the variable at time k has been calculated using information up to time $k-1$. Next, the update step is calculated using equations

$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k, \tag{3.12}$$
$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}, \tag{3.13}$$
$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}), \tag{3.14}$$
$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^T, \tag{3.15}$$

where $\mathbf{S}$ is the innovation covariance, $\mathbf{R}$ is the measurement covariance matrix, $\mathbf{K}$ is the Kalman gain and $\hat{\mathbf{x}}_{k|k}$ is the updated state estimate.

3.1.2 Particle filters

The KF has assumed linear process models and measurement models. Other filtering methods such as the extended Kalman filter (EKF) can handle models that are close to linear, but are vulnerable to highly non-linear models. The particle filter is a non-parametric approach to the filtering problem, see, e.g., [8, p.24-25]. Instead of linearizing the models (3.5),(3.6), as is done in the EKF, the approach in particle filtering is to estimate the posterior distribution using N weighted samples, called particles, as

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N} \omega_k^{(i)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(i)}), \tag{3.16}$$

where $\omega_k^{(i)}$ is the weight and $\mathbf{x}_k^{(i)}$ is the state of particle i at time index k. When dealing with non-linear models, it is often hard, or impossible, to sample directly from the posterior distribution $p(\mathbf{x}_k|\mathbf{z}_{1:k})$. A sampling method called importance sampling (IS) is therefore used, where the idea is to sample from a proposal probability distribution $q(\mathbf{x}_k|\mathbf{z}_{1:k})$ instead. The samples are weighted according to

$$\omega_k^{(i)} \propto \frac{p(\mathbf{x}_k^{(i)}|\mathbf{z}_{1:k})}{q(\mathbf{x}_k^{(i)}|\mathbf{z}_{1:k})} \tag{3.17}$$

to fit the posterior distribution. If the proposal density is chosen so that it satisfies the Markov property,

$$q(\mathbf{x}_k^{(i)}|\mathbf{z}_{1:k}) = q(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_{1:k})q(\mathbf{x}_{k-1}^{(i)}|\mathbf{z}_{1:k-1}), \tag{3.18}$$

and

$$q(\mathbf{x}_k^{(i)}|\mathbf{z}_{1:k}) = q(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k), \tag{3.19}$$

then the weight calculation can be done sequentially as more measurements arrive. The resulting equation for calculating the weights is shown in [10, p.178] to be

$$\omega_k^{(i)} = \omega_{k-1}^{(i)} \frac{p(\mathbf{z}_k|\mathbf{x}_k^{(i)})p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)})}{q(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k)}. \tag{3.20}$$

This is known as sequential IS (SIS) and is the core of particle filters. A common choice for the proposal density is the transition density

$$q(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}, \mathbf{z}_k) = p(\mathbf{x}_k^{(i)}|\mathbf{x}_{k-1}^{(i)}), \tag{3.21}$$

which results in the simple and intuitive weighting equation

$$\omega_k^{(i)} = \omega_{k-1}^{(i)} p(\mathbf{z}_k|\mathbf{x}_k^{(i)}). \tag{3.22}$$

There are however many other methods for choosing the proposal density, e.g. the likelihood can be chosen, see [10].

A problem with particle filters is that they suffer from degeneracy, meaning that the number of particles with insignificant weights increase over time. The common solution to this problem is to resample the particles according to their weight. Particles with low weight are then likely to disappear, while particles with high weight are likely to be duplicated. After resampling, all weights are set equal as

$$\omega_k^{(i)} = \frac{1}{N}. \tag{3.23}$$

Another problem with particle filters is that the computational complexity increase rapidly with the dimension of the state vector $\mathbf{x}_k$. One approach to solve the problem is to exploit the linear properties of the system; this is the topic of the next section.

3.1.3 Marginalized particle filters

By using standard optimal filtering in the form of a KF for the linear states and a particle filter for the nonlinear states, the number of particles needed to efficiently estimate the posterior can be decreased, and in turn the computational complexity is also decreased[11]. In addition, an EKF can be used for the mild nonlinearities. The approach is to divide the state vector into one linear part and one nonlinear part as

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_k^l \\ \mathbf{x}_k^n \end{bmatrix}, \tag{3.24}$$

where $\mathbf{x}_k^l$ is the linear states and $\mathbf{x}_k^n$ is the nonlinear states. The state density for the nonlinear states is represented similar to the standard particle filter in (3.16), while the linear states are represented by a normal distribution as in (3.9). In [11, p.2280-2283], 3 process models of different complexity are presented. In this thesis, a constant turn motion model is chosen, see Chapter 4. Thus, the second marginalized particle filter model in [11], called the Triangular Model, is used since it fits that motion model well,

$$\mathbf{x}_{k+1}^n = \mathbf{f}_k^n(\mathbf{x}_k^n) + \mathbf{A}_k^n(\mathbf{x}_k^n)\mathbf{x}_k^l + \mathbf{v}_k^n, \tag{3.25a}$$
$$\mathbf{x}_{k+1}^l = \mathbf{A}_k^l(\mathbf{x}_k^n)\mathbf{x}_k^l + \mathbf{v}_k^l, \tag{3.25b}$$
$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k^n) + \mathbf{C}_k(\mathbf{x}_k^n)\mathbf{x}_k^l + \mathbf{w}_k, \tag{3.25c}$$

where $\mathbf{f}_k^n$ is the terms of the process model for the nonlinear states not containing the linear states, $\mathbf{A}_k^n$ is the terms of the process model for the nonlinear states containing the nonlinear states, $\mathbf{A}_k^l$ is the motion model for the linear states, $\mathbf{h}_k$ is the terms of the measurement model not containing the linear states, $\mathbf{C}_k$ is the terms of the measurement model containing the linear states and finally $\mathbf{v}_k^n$ and $\mathbf{v}_k^l$ is the additive i.i.d. Gaussian noise for the linear and nonlinear states respectively. The resulting measurement update equations for the linear states can be written as

$$\hat{\mathbf{x}}_{k|k}^l = \hat{\mathbf{x}}_{k|k-1}^l + \mathbf{K}_k(\mathbf{z}_k - \mathbf{h}_k - \mathbf{C}_k\hat{\mathbf{x}}_{k|k-1}^l), \tag{3.26a}$$
$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{C}_k\mathbf{P}_{k|k-1}, \tag{3.26b}$$
$$\mathbf{S}_k = \mathbf{C}_k\mathbf{P}_{k|k-1}\mathbf{C}_k^T + \mathbf{R}_k, \tag{3.26c}$$
$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{C}_k^T\mathbf{S}_k^{-1}, \tag{3.26d}$$

and the prediction update equations can be written as

$$\hat{\mathbf{x}}^l_{k+1|k} = \mathbf{A}^l_k \hat{\mathbf{x}}^l_{k|k} + \mathbf{L}_k(\mathbf{z}_k - \mathbf{A}^n_k \hat{\mathbf{x}}^l_{k|k}), \tag{3.27a}$$

$$\mathbf{P}_{k+1|k} = \mathbf{A}^l_k \mathbf{P}_{k|k}(\mathbf{A}^l_k)^T + \mathbf{Q}^l_k - \mathbf{L}_k \mathbf{N}_k \mathbf{L}^T_k, \tag{3.27b}$$

$$\mathbf{L}_k = \mathbf{A}^l_k \mathbf{P}_{\mathbf{k|k}}(\mathbf{A}^n_k)^T \mathbf{N}^{-1}_k, \tag{3.27c}$$

$$\mathbf{N}_k = \mathbf{A}^n_k \mathbf{P}_{k|k}(\mathbf{A}^n_k)^T + \mathbf{Q}^n_k. \tag{3.27d}$$

Here, $\mathbf{P}_{k|k}$ is the posterior error covariance matrix for the linear states, $\mathbf{Q}^l_k$ is the covariance of the process noise for the linear states $\mathbf{v}^l_k$ and $\mathbf{Q}^n_k$ is the covariance of the process noise for the nonlinear states $\mathbf{v}^n_k$. $\mathbf{R}_k$ is the measurement covariance.

The time and measurement updates for the nonlinear states are done according to the standard particle filter described in section 3.1.2. The algorithm for the marginalized particle filter is described as pseudo code in algorithm 3.1.

Algorithm 3.1: Marginalized particle filter, Algorithm 1 from [11]

1) ***for*** i=1:N
   Initialize the particles' non-linear states
   $\mathbf{x}^{n,(i)}_0 \sim p(\mathbf{x}_0)$
   and the linear state covariances
   $\mathbf{x}^{l,(i)}_0 = \bar{\mathbf{x}}^l_0$
   $\mathbf{P}^{(i)}_0 = \bar{\mathbf{P}}^{(i)}_0$
   end

2) ***for*** i=1:N
   Evaluate the importance weights
   $\omega^{(i)}_k = \omega^{(i)}_{k-1} p(\mathbf{y}_k | \mathbf{x}^{n,(i)}_{1:k}, \mathbf{z}_{1:k})$
   end
   and normalize
   $\bar{\omega}^{(i)}_k = \frac{\omega^{(i)}_k}{\sum_{j=1}^{N} \omega^{(j)}_k}$

3) Resample all particles with probability
   according to their weight
   for i=1:N

   $\Pr\left(\mathbf{x}^{n,(i)}_{k|k} = \mathbf{x}^{n,(j)}_{k|k-1}\right) = \bar{\omega}^{(j)}_k$
   end

4) Particle filter time update and Kalman filtering
   for i=1:N
   a) Kalman filter measurement update according to equations (3.26)

   b) Particle filter time update (prediction via the process model)
   $\mathbf{x}^{n,(i)}_{k+1|k} \sim p(\mathbf{x}^n_{k+1|k} | \mathbf{x}^{n,(i)}_{1:k}, \mathbf{z}_{1:k})$

   c) Kalman filter time update according to equations (3.27)
   end

5) Set k=k+1 and reiterate from step 2.

3.2 Mapping

In this thesis, the scope of the mapping problem is to estimate landmarks in a static environment using a known vehicle pose and lidar measurements. The number of landmarks is unknown, and also the size and location of these objects are unknown and need to be estimated. This is done using two different mapping methods. A random finite set (RFS) method modeling clusters of measurements originating from the same landmark as RFSs, and a simple clustering method based only on the proximity of measurements. The proximity clustering is used as a comparison to evaluate the performance of the more advanced RFS based method.

3.2.1 Poisson Multi-Bernoulli mapping with random finite sets

A Bayesian inference method based on random finite sets, specifically Poisson Multi-Bernoulli distributions, will be used for map estimation. The theoretical background used for this method is presented here.

A RFS X is a set which has a random cardinality specified by a discrete distribution, where all elements in the set (x_i), are random variables with probability distributions from the same family,

$$X = \{x_1, x_2, ..., x_n\}. \tag{3.28}$$

The RFS X has the density

$$p(X) = p(n)n! \sum_{N} f_n(x_1, x_2, ..., x_n) \tag{3.29}$$

where N represents all permutations of the set X, f_n is a joint distribution for the set, and $p(n)$ is a distribution which describes the cardinality of the set. This method of statistical models is useful when observing different patterns that are showing fluctuating behaviour, see e.g., [12].

In this thesis, RFSs are used to model the mapping problem. A landmark can be for example a wall or part of a guardrail, see Figure 2.2. The carnality of the map RFS, i.e, the number of landmarks in the map, is random since we have no prior information about the number of landmarks in the scenario. The random state (extent and location) of each landmark is then described by the measurements associated to each landmark. These states are described by the parameters of some probability distributions, in this thesis the location of a landmark is described by a multivariate normal distribution and the extent is described by an inverse Wishart distribution.

3.2.1.1 Multi Bernoulli mixture

A Bernoulli RFS[13] is a set which is empty with probability $1-r$, or with probability r contains a single element with state x and state distribution $f(x)$. In this thesis x represent a landmark. A Bernoulli RFS X has set density

$$f_B(X) = \left\{ \begin{array}{ll} 1-r & X = \emptyset \\ rf(x) & X = \{x\} \\ 0 & |X| > 1 \end{array} \right\}. \tag{3.30}$$

where $|X|$ is the cardinality, i.e, size of the set and can be either zero or one. If several objects are modelled, a multi Bernoulli (MB) RFS can be used. The MB is the union of multiple independent Bernoulli RFSs, and is used to describe all objects that are of interest. In this thesis a MB RFS is used to represent the whole map. The MB RFS $\{X_1, X_2, ..., X_I\}$ is described by the parameters $\{r_i, f_i(x)\}_{i \in I}$. The MB set density is

$$f_{MB}(X) = \sum_{X = X_1 \uplus \cdots \uplus X_I} \prod_{i \in I} f_{Bi}(X_i) \tag{3.31}$$

where $\uplus$ is a disjoint union. To model data association hypotheses, a multi Bernoulli mixture (MBM) can be used. A MBM is described by the parameters

$$\{W_j, \{r_{i,j}, f_{i,j}(x)\}_{i \in I_j}\}_{j \in J}, \tag{3.32}$$

where W_j is the weight of each MB, $r_{i,j}, f_{i,j}$ are the parameters of each Bernoulli in this MB, J is the index set of MBs in the mixture, I_j is the index set of Bernoullis in the j:th MB. The set density is

$$f_{MBM}(X) = \sum_{j \in J} W_j f_{MBj}(X) \tag{3.33}$$

3.2.1.2 Poisson point processes

In this thesis the landmarks are assumed to be extended objects, meaning that each object may generate more than one measurement at each time instant. This makes it possible to estimate both the size and the extent of an object, see [14]. The measurements are modelled as a Poisson point process (PPP) where the landmarks together with their expected number of measurements represent the PPP intensity, which is used during the localization. Also clutter measurements and landmarks that has not yet been measured are modeled as PPPs.

A PPP is a RFS Y with two parameters: the rate μ, and the spatial distribution $f(y)$, where $y \in Y$. The PPP has an intensity $\mu f(y)$, which in this thesis represents the intensity of measurements, or landmarks, appearing in the space. A PPP RFS Y has set density

$$f_P(Y) = e^{-\mu} \prod_{y \in Y} \mu f(y) \tag{3.34}$$

where the number of elements $y \in Y$ is a Poisson distributed random variable. Integrating the intensity $\mu f(y)$ over the area A gives the expected cardinality of Y in A,

$$E[|Y|_A] = \Lambda_A = \int_A \mu f(y) dy. \tag{3.35}$$

The number of elements in A is Poission distributed with the parameter Λ_A,

$$P(|Y|_A = n) = e^{-\Lambda_A} \frac{\Lambda_A^n}{n!} \tag{3.36}$$

In this thesis the rate μ represents the expected number of measurements from a landmark, or the expected number of landmarks in the map. The distribution $f(y)$ corresponds to either the landmark itself modeled as an extended object, or how the landmarks are expected to be distributed in the map.

3.2.2 Gibbs sampling

Gibbs sampling is a Markov chain Monte Carlo method for sampling from a probability distribution, see, e.g.,[15], and is a method that is often used when the posterior distribution is hard to calculate. Gibbs sampling can then be used to approximate different properties of the posterior distribution, such as the expected value. The method draws samples randomly from the marginal distribution of each variable in the distribution. The idea is that given enough iterations, the samples will spread out in the variable space, according to the true posterior distribution.

In some cases the parameter space can be large and impractical to sample from, and an option is then to use a collapsed Gibbs sampler. The collapsed Gibbs sampler marginalizes over one or more variables, to make sampling easier and more efficient.

In this thesis a collapsed Gibbs sampler is used to sample data associations from the space of all possible associations where each sample corresponds to a MB in the MBM. The posterior distribution of all detected landmarks that represents the map, can be marginalized over the landmark states to only describe the probability

of an individual data association. This makes the probability function used during sampling easier to describe and evaluate.

4

Models

In this chapter, the vehicle model and the measurement model used in the localization filter is presented. The vehicle model is used for state prediction and the measurement model is used to connect the measurements to the state vector.

4.1 Vehicle model

In Bayesian filtering a prediction of the next state given measurements up to the current state is needed. For the localization problem handled in this thesis, the state of the car can be predicted using a vehicle model. Vehicle models range in complexity from simple random walk models to more complex models using specific vehicle paramters.

For this thesis, a model with constant yaw-rate and constant velocity, often called constant turn (CT), has been chosen. This model is intuitive and easy to implement, for details see, e.g. [16]. Due to the fast sampling frequency of 5 Hz or faster for all sensors and a relatively low speed of maximum 30 km/h, there is no need for a highly complex model. The state vector for a CT model in two dimensions is

$$\mathbf{x}_k = \begin{bmatrix} x & y & \psi & v & \dot{\psi} \end{bmatrix}^T, \tag{4.1}$$

where x and y are the spatial position in the horizontal plane, ψ is the yaw angle, v is the speed and $\dot{\psi}$ is the yaw rate. The state transition can be expressed on state-space form as

$$\begin{bmatrix} x_k \\ y_k \\ \psi_k \\ v_k \\ \dot{\psi}_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + T v_{k-1} \cos(\psi_{k-1}) \\ y_{k-1} + T v_{k-1} \sin(\psi_{k-1}) \\ \psi_{k-1} + T \dot{\psi}_{k-1} \\ v_{k-1} \\ \dot{\psi}_{k-1} \end{bmatrix}, \tag{4.2}$$

where T is the sampling time. Notice that the vehicle position in the z direction is

not estimated even though the map is created in three dimensions. This is done to reduce the number of states estimated by the particle filter, and in turn reduce the computational complexity. This is a reasonable simplification in this scenario since it is very flat. In a more hilly environment it would most likely affect the accuracy of the localization. Pitch and roll angles are omitted for the same reason. Given that we have a good location, the map could provide altitude, pitch, and roll.

4.2 Measurement models

The purpose of the measurement models is to connect the measured variables to the latent variables used in the state vector (4.1). In the lidar sensor, the measurements are given as a range r and two angles referred to as azimuth α and elevation ϵ.

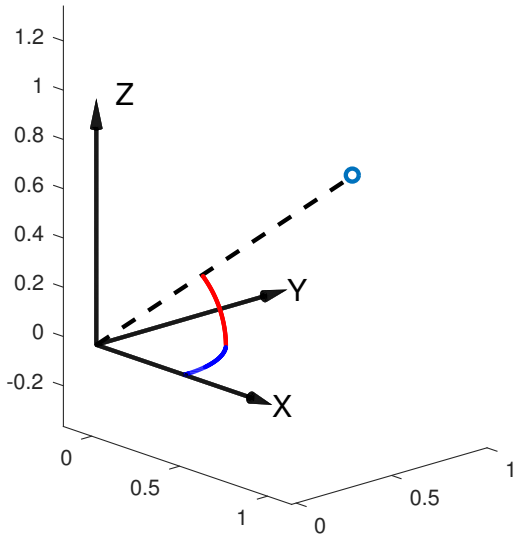


Figure 4.1: The illustration shows the sensor reference frame, the sensor is facing the x direction and detects a measurement at the blue circle. The elevation is the angle from the xy plane to the measurement in red and the azimuth is the angle from the xz plane to the measurement in blue. The range is the distance from the measurement to origo.

The measurements z can be transformed to the spatial sensor frame s as

$$\begin{bmatrix} x_z^s \\ y_z^s \\ z_z^s \end{bmatrix} = \begin{bmatrix} r\cos(\alpha)\cos(\epsilon) \\ r\sin(\alpha)\cos(\epsilon) \\ r\sin(\epsilon) \end{bmatrix} \tag{4.3}$$

The measurement can then be rotated and translated from the sensor frame to the global frame g using the vehicle pose and rotational matrices as

$$\begin{bmatrix} x_z^g \\ y_z^g \\ z_z^g \end{bmatrix} = R_z(\psi)R_y(\theta)R_x(\phi) \begin{bmatrix} x_z^s \\ y_z^s \\ z_z^s \end{bmatrix} + \begin{bmatrix} x_v^g \\ y_v^g \\ 0 \end{bmatrix} \tag{4.4}$$

where

$$R_z(\psi) = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.5}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}, \tag{4.6}$$

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}, \tag{4.7}$$

and x_v^g and y_v^g is the vehicle position in the global frame, note that subscript z in R_z represents a rotation around the z axis. The position in z is zero due to the assumption of a flat scenario, see Section 4.1. The angles ψ, θ and ϕ represents the yaw, pitch and roll euler angles. The choise of vehicle model also means that only the ψ angle is estimated by the filter. However, there exists pre-filtered measurements of ϕ and θ from the IMU, which in this thesis is used directly in the rotational matricies.

The measurement equation for the sensor (4.4) can then be summarized as

$$\mathbf{z}_z^g = h(\mathbf{x}_k) = \begin{bmatrix} x_z^g \\ y_z^g \\ z_z^g \end{bmatrix} \tag{4.8}$$

where $\mathbf{x}_k$ is the state vector from equation (4.1) and $\mathbf{z}_z^g$ is the measurements in the global frame. Note that there is no added measurement noise. We motivate this by the fact that the resolution and the accuracy of the lidar sensor is very high, in relation to the uncertainty introduced by the vehicle motion model.

The measurement model used for the IMU is a linear model with additive normal distributed noise $\mathbf{r}_k$,

$$\mathbf{z}_{IMU} = h_{IMU}(\mathbf{x}_k) = \begin{bmatrix} v_k \\ \dot{\psi}_k \end{bmatrix} + \mathbf{r}_k. \tag{4.9}$$

5

Mapping methods

The mapping problem in this thesis is to describe the environment around the vehicle for the purpose of localization. Landmarks are used to describe specific objects that are of interest for localization. Two methods are implemented, one probabilistic method that models the uncertainties in the environment and measurements from it, and focuses on a minimal representation of the environment. The other method does not model uncertainties in the landmark extent, but is motivated by its simplicity and low computational complexity during map creation. It will also serve as a performance reference when evaluating the performance of the probabilistic method. The derivation and implementation of these two methods are described in this section. Both mapping methods assume accurate knowledge about the vehicle pose at all times during mapping.

5.1 Poisson Multi-Bernoulli mixture

The map is created by grouping measurements that are likely to come from the same object into a cluster. The landmarks are estimated using a Poisson multi-Bernoulli mixture (PMBM) model, with a Normal Gamma inverse Wishart landmark model. The PMBM is used since it describes both the uncertainty in the number of measurements from a landmark, as well as the landmark location and extent, as random variables, and can thus estimate both simultaneously. The method is largely based on [17] with some changes to fit the problem, and to decrease the computational complexity. The following sections describes how the map is estimated from measurements.

5.1.1 Mapping models

To be able to create a map of the environment, models are used to describe the clutter, the previously undetected landmarks, and the detected landmarks. These models are based on the theory described in Section 3.2, previous work done in mapping [17],[18], and Poisson point process (PPP) models described in [19]. To simplify notation, the set of all measurements from the lidar sensor is described as

$$Z = \mathbf{z}_{1:K} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_K\}. \tag{5.1}$$

where measurements from each time instance $\mathbf{z}_k$ consists of several lidar detections z. This way, we have removed the time index from measurements completely. Please note that z is the assumed noise free measurements in the global cartesian domain. Furthermore, we denote the set of landmarks by

$$\Theta = \{\theta_1, \theta_2, ...\}, \tag{5.2}$$

where a landmark is defined by the triple θ_i, with the three entities μ_i, the location of a landmark, Σ_i, the extent of a landmark and ω_i, the expected number of measurements from a landmark,

$$\theta_i = (\mu_i, \Sigma_i, \omega_i). \tag{5.3}$$

The landmarks are further explained in 5.1.3.

5.1.1.1 Clutter

The clutter is modelled as a Poisson Point process (PPP) with two parameters, the rate λ_c and the spatial distribution $c(z)$. The PPP intensity is $\lambda_c c(z)$ and describes clutter intensity. The spatial distribution of the clutter is assumed to be uniformly distributed in the sensor's field of view (FoV) $c(z) = 1/V$, i.e., the clutter is equally likely to be measured anywhere inside the FoV. Here V is the volume of the FoV. The PPP for the clutter measurements $Z^c \subseteq Z$ is then described as

$$f_P(Z^c) = e^{-\lambda_c} \prod_{z \in Z^c} \lambda_c c(z). \tag{5.4}$$

5.1.1.2 Undetected landmarks

The yet undetected landmarks are also modelled as a PPP with two parameters, the rate λ_u and the spatial distribution $\phi(z|\theta)$. The spatial distribution is modelled by $\phi(z|\theta) = \mathcal{N}(z; \mu, \Sigma)$, where $\theta(\mu, \Sigma)$ is a landmark with the parameters μ for mean position and Σ is modelled by a random matrix which describes the extent of the landmark and z is the measurement from the sensor. The PPP for undetected landmarks Θ^u is then described as

$$f_P(\Theta^u) = e^{-\lambda_u} \prod_{\theta \in \Theta^u} \lambda_u \phi(z|\theta). \tag{5.5}$$

5.1.1.3 Detected landmarks

The previously detected landmarks are described by a multi-Bernoulli mixture (MBM), where each multi-Bernoulli represents a possible data association. The posterior distribution is

$$f_{MBM}(\Theta^d) = \sum_{j \in J} W_j \sum_{X = X_1 \uplus \cdots \uplus X_I} \prod_{\theta \in \Theta^d} f_B(\theta), \tag{5.6}$$

where Θ^d is the set of all detected landmarks and N represents all ways of partitioning the Bernoulli components. The weight W_j of each multi-Bernoulli represents the probability that the corresponding data association for all measurements is the right one given the modelling and a priori assumptions, and J is the total number of hypotheses in the mixture.

5.1.2 Partitions

By partitioning the measurements, the weight of each partition can be calculated to get a likelihood of that data association. The likelihood can then be used in an optimization algorithm to get the most likely data association. In this case, a modified Gibbs sampler is used. A partition is a hypothesis of the data association, meaning that the measurements are partitioned into cells associated with the landmarks. The partitions will be indexed by $j \in J$, and the subsets of landmarks by $i \in I$. The measurements that belong to a specific landmark is denoted $Z^{j,i}$, which is a subset of all the measurements Z. From each landmark there will be a series of measurements originating from different time samples, but since we do batch processing using all measurements from the whole time sequence we denote the measurement set with only the partition index Z^j.

The weight of each partition is proportional to the product of the likelihood of each Bernoulli in that partition, or the likelihood of each landmark in the partition. The partitions are used when creating the map, optimally the chosen partition should represent the most likely data association given the mapping models.

Sampling will be used as a Bayesian inference method to update the posterior density of the map. The method used to estimate the posterior of the map is to sample from the different partitions. The posterior distribution (5.6) can be rewritten to the joint distribution, for the purpose of being able to sample from it. When describing the joint posterior, we describe the map using one partition as

$$p(\Theta^d, j) = W_j \sum_{N} \prod_{\theta_i \in \Theta^d} f_B(\theta_i). \tag{5.7}$$

The detected landmarks can then be marginalized out to get the probability of

partition j, which is the weight of the partition

$$P(j) = W_j. \tag{5.8}$$

This weight represents the probability of each data association, and sampling from these associations can be done by calculating the weight of the corresponding partitions. This weight is proportional to the combined likelihood of each landmark

$$W_j \propto \prod_{i \in I} L_{j,i}. \tag{5.9}$$

To calculate the weights, the likelihood integral is calculated as

$$L_{j,i} = \lambda_0^u \int f_0^u(\theta) l_{j,i}(\theta) d\theta, \tag{5.10}$$

where $f_0^u(\theta)$ and λ_0^u is the prior spatial density and rate of an undetected landmark, and the likelihood $l_{j,i}$ comes from the PPP of undetected landmarks

$$l_{j,i} = e^{-\gamma(\theta)} \prod_{z \in Z_{j,i}} \gamma(\theta) \phi(z|\theta), \tag{5.11}$$

with the parameters described in 5.1.1.2. This likelihood will be used to calculate the weight of a partition, as a measure of how good the data association is.

5.1.3 Landmark representation

The map is built up by many landmarks θ_i, each represented by its parameters μ_i, Σ_i, ω which is the mean, extent matrix and expected number of measurements. The landmark state density is assumed to be a normal inverse Wishart gamma distribution,

$$f(\theta_i) = \mathcal{N}(\mu; \mu_i, (c_i)^{-1}\Sigma) \mathcal{IW}(\Sigma; S_i, v_i) \mathcal{GAM}(\omega; \alpha_i, \beta_i), \tag{5.12}$$

where the normal distribution describes the mean of the landmark spatial position (μ_i), the inverse Wishart describes the extent of the landmark (Σ_i), and the gamma distribution describes the expected number of measurements from the landmark (ω_i). The expected values of the distributions represents the parameters that are later used for each landmark.

The resulting map is represented by a PPP intensity function as an unnormalized Gaussian mixture over all the landmarks.

$$\mathcal{I}(z) = \sum_{i}^{I} \omega_i \mathcal{N}(z; \mu_i; \Sigma_i), \tag{5.13}$$

where i is the landmark index and I is the total number of landmarks and z is the noise free measurements in the cartesian domain. The intensity is proportional to the probability of a landmark generating a measurement in a specific location.

5.1.4 Computing the weight of a partition

The map posterior distribution will be estimated by sampling from the partition distribution, using the weight of the partitions (5.8), described in 5.1.2. The derivation of the weight function is described in this section. According to Theorem 1 in [17], the weight is proportional to the likelihood of all landmarks in the partition.

One landmark has the proportional likelihood

$$L_{j,i} \propto \int f_0^u(\theta) l_{j,i}(\theta) d\theta. \tag{5.14}$$

which will be used to calculate the weight of this specific landmark. The following assumptions are used to solve this integral: the probability of detection is constant in the FoV and the parameters of the map is a priori distributed in a conjugate prior form as

$$f_0^u(\theta) = \mathcal{U}(\mu)\mathcal{IW}(\Sigma; S_0, v_0)\mathcal{GAM}(\omega; \alpha_0; \beta_0), \tag{5.15}$$

where $\mathcal{U}$ denotes a uniform distribution over the map, $\mathcal{IW}$ is the inverse Wishart distribution and $\mathcal{GAM}$ is the gamma distribution. This means that the position μ of a landmark is a priori uniformly distributed, the extent Σ is a priori inverse Wishart distributed and the number of measurements ω is gamma distributed. These prior distributions have been used in previous work regarding PMBM tracking and mapping [13, 17]. The likelihood is

$$l_{j,i} = p_D(\theta) e^{-\gamma(\theta)} \prod_{z \in Z^{j,i}} \gamma(\theta) \phi(z|\theta), \tag{5.16}$$

where $\phi(z|\theta) = \mathcal{N}(z; \mu, \Sigma)$, and $\gamma(\theta) = \omega \text{FoV}(\mu, x_k)$, with FoV as the field of view function that returns 1 if μ is inside the FoV from location state x_k and 0 if it is not. The integral to solve (5.14) is then

$$\int f_0^u(\theta) l_{j,i}(\theta) d\theta = \int \mathcal{U}(\mu)\mathcal{IW}(\Sigma; S_0, v_0)\times$$
$$\mathcal{GAM}(\omega; \alpha_0; \beta_0) p_D^k e^{-\omega \mathrm{FoV}(\mu, x_k)} \times \prod_{z \in Z^{j,i}} (\omega \mathrm{FoV}(\mu, x_k)\mathcal{N}(z; \mu, \Sigma)) d\mu d\Sigma d\omega. \quad (5.17)$$

Assuming that the landmark that is updated has measurements ($Z^{j,i} \neq \emptyset$) and that the entire landmark is inside the FoV ($U(\mu) = 1/V_A, \mathrm{FoV}(\mu, x_k) = 1$), the integral can be simplified to

$$\int f_0^u(\theta) l_{j,i}(\theta) d\theta = p_D^k / V_A \int \omega^{|Z^{j,i}|} e^{-\omega} \mathcal{GAM}(\omega; \alpha_0, \beta_0) d\omega \times$$
$$\int \mathcal{IW}(\Sigma; S_0, v_0) \prod_{z \in Z^{j,i}} \mathcal{N}(z; \mu, \Sigma) d\mu d\Sigma. \quad (5.18)$$

This equation can be separated into two parts. The first part is the gamma distribution describing the expected number of measurements, and the second part consists of a normal and inverse Wishart distribution representing the position and extent of a landmark. The second part is proportional to a normal inverse Wishart distribution.

The first part is the expected number of measurements, this can be calculated by using the Gamma pdf

$$\mathcal{GAM}(\omega; \alpha_0, \beta_0) = \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} \omega^{\alpha_0 - 1} e^{-\beta_0 \omega}. \quad (5.19)$$

This gives the following solution

$$\begin{aligned}
&\int \omega^c e^{-\omega} \mathcal{GAM}(\omega; \alpha_0, \beta_0) d\omega = \\
&= \int \omega^c e^{-\omega} \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} \omega^{\alpha_0 - 1} e^{-\beta_0 \omega} d\omega \\
&= \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} \frac{\Gamma(\alpha_0 + c)}{(\beta_0 + 1)^{\alpha_0 + c}} \int \frac{(\beta_0 + 1)^{\alpha_0 + c}}{\Gamma(\alpha_0 + c)} \omega^{\alpha_0 + c - 1} e^{-\omega(\beta_0 + 1)} d\omega \\
&= \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} \frac{\Gamma(\alpha_0 + c)}{(\beta_0 + 1)^{\alpha_0 + c}} \int \mathcal{GAM}(\omega; \alpha_0 + c, \beta_0 + 1) d\omega \\
&= \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} \frac{\Gamma(\alpha_0 + c)}{(\beta_0 + 1)^{\alpha_0 + c}}, \quad (5.20)
\end{aligned}$$

where $c = |Z^{j,i}|$, i.e. the cardinallity of the partition set. This is the gamma distribution update by the measurements in $Z^{j,i}$, and can be simplified by introducing

$$\alpha^{j,i} = \alpha_0 + |Z^{j,i}| \tag{5.21a}$$
$$\beta^{j,i} = \beta_0 + 1, \tag{5.21b}$$

which gives

$$\int \omega^{|Z^{j,i}|} e^{-\omega} \mathcal{GAM}(\omega; \alpha_0, \beta_0) d\omega = \frac{\beta_0^{\alpha_0} \Gamma(\alpha^{j,i})}{(\beta^{j,i})^{\alpha^{j,i}} \Gamma(\alpha_0)}. \tag{5.22}$$

The second part of the integral (5.18) is the location and extent of the landmark. This is assumed to be proportional to a normal inverse Wishart

$$\mathcal{NIW}(\mu, \Sigma; \mu^{j,i}, c^{j,i}, S^{j,i}, v^{j,i}) = \mathcal{N}(\mu; \mu^{j,i}, (c^{j,i})^{-1}\Sigma)\mathcal{IW}(\Sigma; S^{j,i}, v^{j,i}) \tag{5.23}$$

and is calculated in the same way derived in [17, 20] and the parameters are updated by the measurements in $Z^{j,i}$ as

$$v^{j,i} = v_0 + |Z^{j,i}| - 1, \tag{5.24a}$$
$$S^{j,i} = S_0 + \sum_{z \in Z^{j,i}} (\mathbf{z} - \overline{\mathbf{z}})(\mathbf{z} - \overline{\mathbf{z}})', \tag{5.24b}$$
$$\mu^{j,i} = \overline{\mathbf{z}}, \tag{5.24c}$$
$$c^{j,i} = |Z^{j,i}|. \tag{5.24d}$$

The parameters in (5.21) and (5.24) defines the landmark, and are updated every time the data association is changed. The complete integral (5.18) is

$$\int \mathcal{IW}(\Sigma; S_0, v_0) \prod_{z \in Z^{j,i}} \mathcal{N}(z; \mu, \Sigma)) d\mu d\Sigma = \frac{|S_0|^{0.5v_0} \Gamma_2(0.5v^{j,i})}{|S^{j,i}|^{0.5v^{j,i}} \Gamma_2(0.5v_0) \pi^{v^{j,i}} c^{j,i0.5}}. \tag{5.25}$$

Please note that $|S_0|$ is the determinant of the matrix, not the cardinality which has the same notation. Together with the expected number of measurements, this gives the likelihood of an updated landmark,

$$L_{j,i} \propto \frac{\beta_0^{\alpha_0} \Gamma(\alpha^{j,i})}{(\beta^{j,i})^{\alpha^{j,i}} \Gamma(\alpha_0)} \frac{|S_0|^{0.5v_0} \Gamma_2(0.5v^{j,i})}{|S^{j,i}|^{0.5v^{j,i}} \Gamma_2(0.5v_0) \pi^{v^{j,i}} (c^{j,i})^{0.5}}. \tag{5.26}$$

The weight of a multi Bernoulli is proportional to the likelihood of each landmark

$$W_j \propto \prod_i L_{j,i}, \tag{5.27}$$

which can be simplified by removing the constants, since it is a proportionality,

$$W_j \propto \prod_i \frac{\Gamma(\alpha^{j,i})}{(\beta^{j,i})^{\alpha^{j,i}}} \frac{\Gamma_2(0.5v^{j,i})}{|S^{j,i}|^{0.5c^{j,i}} \pi^{v^{j,i}} (c^{j,i})^{0.5}}. \tag{5.28}$$

This is the resulting weight of a partition when it is updated by new measurements.

5.1.5 Sampling the partitions

To find the best possible map, a Bayesian inference method is used. The method is based on Gibbs sampling[17], but with some differences and is suitable when the parameter space is large and other methods are difficult to apply[21]. The method starts with an initial partitioning of the measurements: this could be any valid partition. Then a new set of possible partitions are considered by moving one or more measurements to another cell or to a new cluster, and the resulting partition weight of each one is calculated. These weights are normalized, used as probabilities, and sampled from, to get a new partition. The new partition is then used and a new set of possible partitions are considered. This results in a chain of partitions separated by an action. The possible actions are described below with N number of clusters. The first step is to choose a cluster and a measurement from that cluster randomly.

- Move the point to another cluster - One option is to move the point from the chosen cluster and place it in another cluster. This gives $N - 1$ options, since all clusters can receive the point except the cluster in which it is already placed.

- Create new cluster - The point could belong to a single point cluster, this represents that the point is assumed to be clutter. This gives 1 option.

- Merge two clusters - Merging two clusters is an option to increase the filter's speed of convergence. The measurements from the chosen cluster and another cluster are combined into one cluster. This gives $N-1$ options, since it cannot merge with itself.

- Split the cluster in two - The opposite action to merging two clusters is to split them in two. This gives 1 option. To do this, a clustering algorithm can be used. In this thesis *kmeans++*[22] is chosen, but any method to cluster points can be used. This gives a deterministic behaviour in the splitting of clusters, which means that the algorithm is no longer a valid Gibbs sampler, since there is no guarantee to get back from any merge to the same partition as

before. The method is therefore not a true Markov chain Mote Carlo (MCMC) sampling method.

5.1.6 Batch mapping

To create the map, a batch solution is implemented. This solution uses all recorded measurements to infer the posterior map. The mapping is done by calculating the final weight (likelihood) of the partition for all the possible $2N$ actions. The weights are then normalized and sampled from as probabilities. The chosen action is applied to the partition, which yields the next partition in the chain. This should increase the total weight of the partition until a maximum is reached, and the solution converges i.e. when the maximum partition weight have not increased for some set number of iterations. This could be a local maximum, but since stochastic methods are used, a global maximum should be reached given enough samples.

When the solution has converged, the partition with the highest weight and its corresponding MB is chosen as the resulting map. This is opposed to the standard Gibbs sampling where a set of samples would be used to represent the posterior. In the MBM case this would mean than a set of MB samples would be used to find the MB that best represents the MBM, examples of such a method is described in e.g. [23]. A motivation for using the maximum weighted MB is that we are not interested in estimating the whole MBM, but to obtain a MB with as large likelihood (5.8) as possible given the data. An example of the partition weights is shown in Figure 5.1.

The pseudo code for the mapping is shown in Algorithm 5.1, which uses the inputs reference location x and the set of all measurements z. A flowchart of the algorithm is presented in Figure 5.2 to complement the pseudo code.

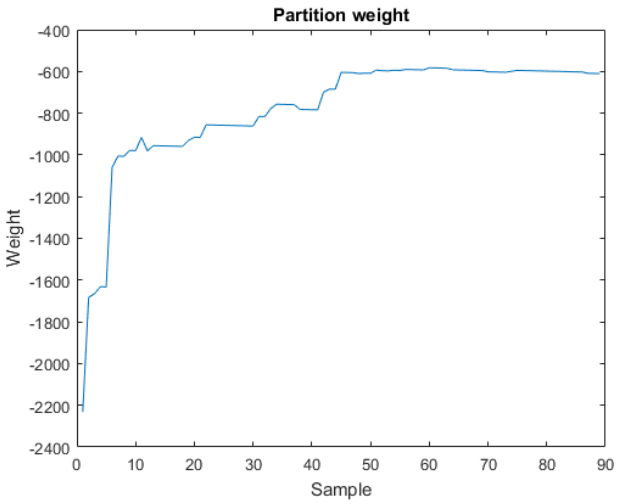


Figure 5.1: Example of the partition weight over time during sampling

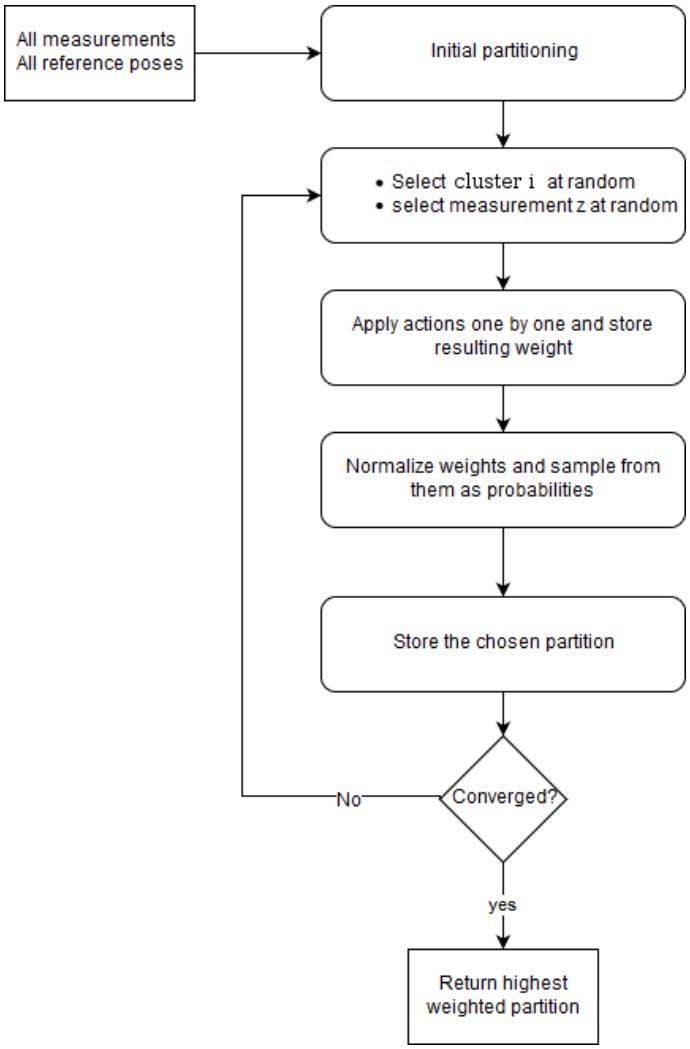


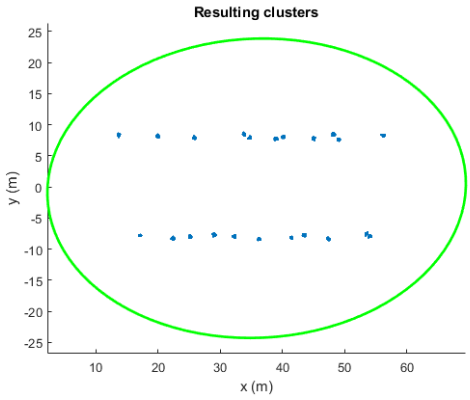
Figure 5.2: Flowchart of the mapping algorithm.

Algorithm 5.1: Mapping Pseudo Code

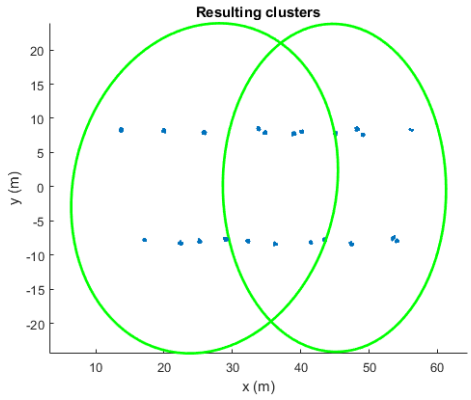
```
input: measurements z, pose x
output: partition P
begin
    Initialize all measurements in some partitioning.
    while(not converged)
        pick a random measurement from a random cluster
        evaluate the resulting weight of each action according to 5.28
        sample one of the possible actions according to their weights
    end
    return highest weighted partition
end
```

An example of the algorithm, during a few iterations is shown in Figure 5.3. Initially all measurements are chosen to be in the same cluster, see Figure 5.3a. After one iteration of the algorithm, the initial partition is split into two, see Figure 5.3b. The next iteration is also a split into two new clusters resulting in three clusters total, see Figure 5.3c. After 88 iterations the algorithm has converged i.e. the maximum partition weight have not increased for some number of iterations. The resulting

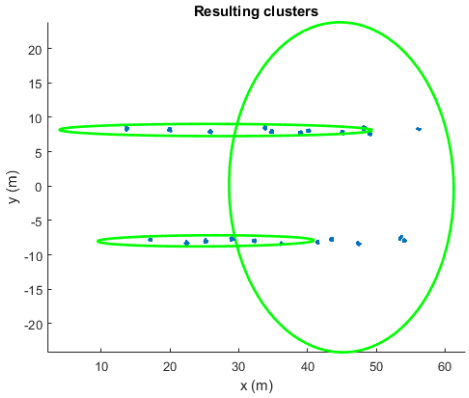
partition is shown in Figure 5.3d. This series of images illustrates the principle behind the algorithm. The weight of the current partition during the sampling is shown in Figure 5.1.



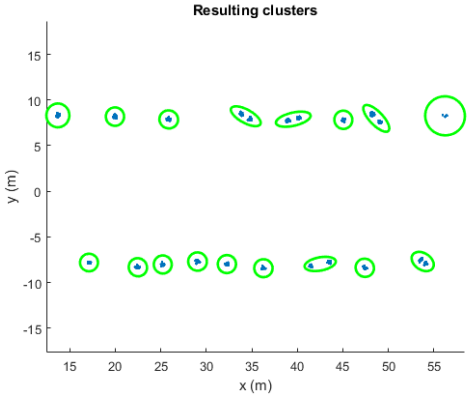
(a) Initial clustering of measurements.



(b) Resulting clusters after one iteration.



(c) Clusters after two iterations.



(d) The resulting clusters after 88 iterations when the algorithm has converged.

Figure 5.3: Example of the clustering in different iterations of the algorithm.

5.2 Proximity landmarking

An alternative method for data association is to group measurements based on the distance between them. The basic assumptions here are that measurements originating from the same landmark are located within a specified distance from each other. With this assumption, a new measurement can either be a part of an existing landmark or originating from a new previously unmeasured landmark.

The algorithm starts with calculating the euclidean distance to all existing landmarks when receiving a measurement. This distance is then compared to a parameter d that decides if it is a part of the landmark or from a new one.

If it is part of an existing landmark, the position of the landmark is updated according to

$$\mu = \frac{1}{N} \sum p_1 + p_2 + ... + p_N \tag{5.29}$$

or a new landmark is initialized at the measurements location. The extent is described by the sample covariance of the points

$$Cov(X) = \frac{1}{N} \sum (p_i^2 - \mu^2) \tag{5.30}$$

A flowchart of the algorithm is shown in Figure 5.4. The only parameter of this algorithm is d, which is tuned to create suitably large landmarks. The algorithm iterates over all the measurements and groups them based on the parameter d.

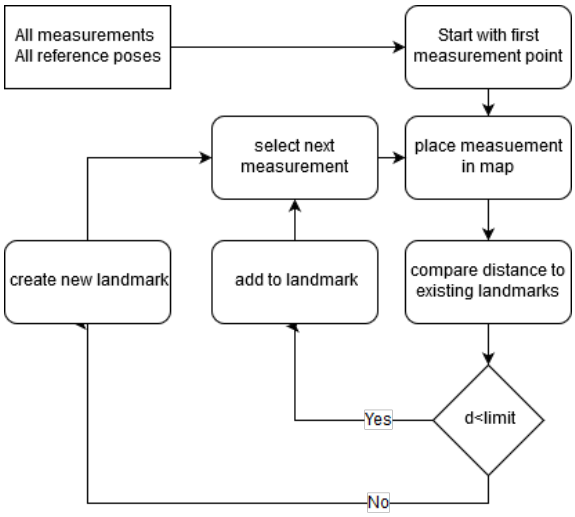


Figure 5.4: Flowchart of proximity mapping algorithm

5.3 Implementational aspects

The implemented methods are used only with pre-filtered data, such that only measurements that are aligned vertically within a small radius remain i.e. if two or more measurements from the same azimuth angle are closer than a set distance from each other when projected onto the horizontal plane, the measurements are kept. The idea is that only measurements originating from objects with a vertical surfaces remain such as houses, posts and trees, while measurements from the road and nearby hills are removed. This makes the algorithm run faster due to less data, while only filtering out measurements from horizontally flat objects that are redundant for localization.

The weight between two consecutive partitions in the chain differs only by the changed landmarks. Therefore only the weight of the affected landmarks are re-calculated. Hence, the new weight of a partition after an action is

$$LW_{new} = LW_{old} + LW_{source} + LW_{target}, \tag{5.31}$$

where

$$LW_{source} = LW_{source}^{after} - LW_{source}^{before} \tag{5.32}$$
$$LW_{target} = LW_{target}^{after} - LW_{target}^{before}, \tag{5.33}$$

where LW is the logarithmic weight of a partition ($log(W)$), the weights before and after an action is applied to the specific landmarks *source* and *target*, note that the weights are additive due to being logarithms of the true weights. All weights are calculated as logarithms of the true weights to avoid computational errors due to very small or large numbers.

When the PMBM algorithm is initiated, the initial clustering of the measurements is made by using the proximity clustering method with a large parameter d. This is to increase the speed of convergence, since measurements in close proximity do have a higher chance of belonging to the same landmark. Even if the association is wrong, the sampling will correct this. An alternative is to start with each measurement in a separate landmark, this gives a lot of landmarks and slows down the calculations significantly.

Since *kmeans++* is used when splitting landmarks, the sampling algorithm in Section 5.1.5 is not a true MCMC technique. Any valid clustering technique could be used when splitting, but this method was used for implementational simplicity. The end result should not be radically different from a standard Gibbs sampling algorithm, since the method mostly uses the split/merge in the beginning of the clustering. When the method is close to convergence mostly one point is moved, as is the case in a true Gibbs sampler. If a true Gibbs sampler is desired, the split/merge could be disabled when the sampling seems to have converged, i.e., when the maximum partition weight have not increased for a set number of iterations. This would give a true MCMC sampling technique.

6

Localization method

Localization is the problem to estimate the position and orientation of the vehicle using the motion and measurement models, together with a set of measurements. In this thesis, the localization is done using a marginalized particle filter, where some linear states are estimated using a Kalman filter, and some non-linear states are estimated using particles. The particles are weighted using a previously created map, together with lidar measurements of the environment. A calculation of the expected number of measurements from a landmark is also done from the estimated position and orientation.

6.1 Marginalized particle filter

The purpose of using a marginalized particle filter is to reduce the number of states estimated by the particle filter, and in turn reducing the number of particles needed to accurately estimate the states [11], see Section 3.1.3. In the presented case, the measurement model connecting the map and lidar measurements to the vehicle location and orientation is highly nonlinear. The vehicle state vector

$$\mathbf{x}_k = \begin{bmatrix} x & y & \psi & v & \dot{\psi} \end{bmatrix}^T, \tag{6.1}$$

is therefore divided into linear and nonlinear states. The orientation and location is treated as nonlinear states,

$$\mathbf{x}_k^n = \begin{bmatrix} x_k & y_k & \psi_k \end{bmatrix}^T. \tag{6.2}$$

The speed and angular velocities are measured directly by the internal sensors and expressed linearly in the CT motion model (4.2), and can therefore be estimated by the Kalman filter part of the marginalized particle filter,

$$\mathbf{x}_k^l = \begin{bmatrix} v_k & \dot{\psi}_k \end{bmatrix}^T. \tag{6.3}$$

Given the chosen motion model presented in (4.2), the state-space equations for the particle filter are given by

$$f^n(\mathbf{x}_k^n) = \begin{bmatrix} x_k & y_k & \psi_k \end{bmatrix}^T, \tag{6.4}$$

$$A^n(\mathbf{x_k^n}) = \begin{bmatrix} T\cos(\psi_{k-1}) & 0 \\ T\sin(\psi_{k-1}) & 0) \\ 0 & T \end{bmatrix}, \tag{6.5}$$

$$A^l = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \tag{6.6}$$

these equations are used for prediction in the Kalman filter of the marginalized particle filter, in (3.25a) and (3.25b).

6.1.1 Prediction and measurement update

The prediction step is done according to (3.27), using the motion models above. The measurement update is done in two steps. First the linear states are estimated using the measurement update (3.26). The nonlinear states are then updated by weighting the particles using the lidar measurements and the map. The weight for each particle is calculated by evaluating the map, represented by a poisson point process intensity (5.13), according to

$$w_k^p = \prod_{m=1}^{M} \sum_{i=1}^{I} \hat{\omega}_i \mathcal{N}(z_{m,p}; \hat{\mu}_i, \hat{\Sigma}_i), \tag{6.7}$$

where M is the number of measurements at time k, I is the total number of landmarks i, and $z_{m,p}$ is the spatial position of lidar measurement m given the state vector $\mathbf{x}^p$ of particle p. For details about the measurement transformation, see section 4.2. Furthermore, $\hat{\omega}$ $\hat{\mu}_i$ and $\hat{\Sigma}_i$ are the estimated landmark states. The derivation of the expected number of measurements $\hat{\omega}$ is presented in the next section, 6.1.2. It should also be noted that the roll angle ϕ and the yaw angle ψ are used in addition to the state vector when evaluating the spatial position of the measurement. The reason why the roll angle ϕ and the pitch angle θ are not represented in the state vector is because they are already pre filtered and very accurate.

The problem with data association is avoided for the localization when using the particle filter since the particle measurement update is performed given the whole static map instead of associating each measurement with a single landmark, as would be the case in a standard Kalman Filter measurement update.

6.1.2 Expected number of measurements

The expected number of measurements from a landmark is estimated by the updated gamma paramters when creating the map, however if the landmark is not seen from the same angle or even the same distance, the initial expected number of measurements might be incorrect. To get past this problem, a calculation of the expected number of measurements is made, based on the size of a landmark and the current pose of the sensor.

The expected number of measurements $\hat{\omega}$ can be obtained by setting up equations for the intersection between the laser ray from the lidar sensor to the landmark with the range r as unknown variable. If the equations have a real solution it means that the ray in theory should hit the landmark [24], not considering the possibility of other landmarks blocking the sensor field of view. The equation describing the laser ray with unknown variable r is

$$\begin{bmatrix} x_i^s \\ y_i^s \\ z_i^s \end{bmatrix} = \begin{bmatrix} r\cos(\alpha)\cos(\epsilon) \\ r\sin(\alpha)\cos(\epsilon) \\ r\sin(\epsilon) \end{bmatrix}, \tag{6.8}$$

where α is the azimuth angle and ϵ is the elevation angle of the laser ray. The intersection point can be described in the landmark frame by transforming the laser ray via the global frame

$$\begin{bmatrix} x_i^l \\ y_i^l \\ z_i^l \end{bmatrix} = (R_g^l)^{-1} \left(R_s^g \begin{bmatrix} x_i^s \\ y_i^s \\ z_i^s \end{bmatrix} + \begin{bmatrix} x_v^g \\ y_v^g \\ z_v^g \end{bmatrix} - \begin{bmatrix} x_l^g \\ y_l^g \\ z_l^g \end{bmatrix} \right). \tag{6.9}$$

Here the variable superscripts represent the frame in which the variable is represented; s for sensor, g for global, and l for landmark. The subscript indicates which point is represented by the variable, i for intersection between laser ray and landmark, l for the landmark location, and v for vehicle location. The two rotation matrices R_s^g and R_g^l rotates the intersection point from the sensor frame to the global frame and from the global frame to the landmark frame, respectively. The rotation matrix R_s^g is a standard rotation using the orientation of the car. The landmark orientation needed for R_g^l, on the other hand, is not known explicitly, and is calculated from the covariance matrix $\mathbf{\Sigma}_l$ as follows,

$$\begin{aligned} \psi_l &= \tan^{-1}(-y_1, x_1), \\ \theta_l &= \sin^{-1}(z_1), \\ \phi_l &= \tan^{-1}(-z_2, z_3), \end{aligned} \tag{6.10}$$

where

$$\mathbf{v}_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}, \quad \mathbf{v}_2 = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}, \quad \mathbf{v}_3 = \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix}, \tag{6.11}$$

are the eigenvectors of $\mathbf{\Sigma}_l$. Using equation (6.8) and (6.9), three equations are obtained describing the point of intersection, with only the radius r as unknown variable. The surface of the landmark is described by the ellipsoid equation

$$\frac{x_i^l}{s_1^2} + \frac{y_i^l}{s_2^2} + \frac{z_i^l}{s_3^2} = 1, \tag{6.12}$$

where s_1, s_2 and s_3 are given by the eigenvalues for the landmark covariance matrix Σ_l. Combining equation (6.8), (6.9), and (6.12) the resulting equation is a second order polynomial,

$$Ar^2 + Br + C = 0. \tag{6.13}$$

The derivation of the coefficients A, B and C was done in MATLAB using symbols. Given the vehicle state, the laser ray angles and the landmark state, equation (6.13) can be solved for r. If r is real the laser ray intersects the ellipsoid and the expected number of measurements counter $\hat{\omega}$ is increased by one.

6.2 Implementational aspects

In this section some of the more important implementations aspects and difficulties are described, to help further research of similar algorithms. An overview of the algorithm is presented in Figure 6.1. The localization algorithm can be initialized either with the nonlinear state vector (6.2) known from the reference sensor, with the states uniformly distributed or normally distributed. After the initialization, the algorithm starts the time iterations. First the algorithm takes in lidar measurements and particles are weighted accordingly as can be seen in the top center and top left box in Figure 6.1. The particle weighting is by far the heaviest part of the algorithm in terms of computation time. In each time step an order of 1000 measurements arrive from the lidar, each measurement should be evaluated for the whole map of an order 100 landmarks represented by 3d multivariate normal distributions. This means that the algorithm have to perform an order of 1000 rotations and translations and evaluate an order of 100 000 3d multivariate normal distributions. To decrease the computation time, some simplifications are made.

First we include the possibility to cluster measurements according to their mutual proximity, see Section 5.2, and only use the mean of the clusters as measurements,

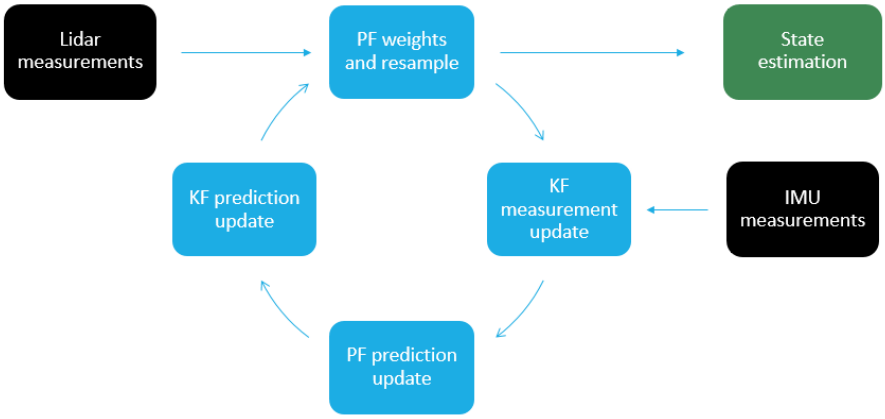


Figure 6.1: Overview of the marginalized particle filter algorithm

this is only used to improve computation time during testing and not used for the results presented in Chapter 7. Moreover, a windowing function is implemented for the purpose of limiting the weighting calculation to only include landmarks within a set distance from the individual measurements. The assumption is that landmarks far from the measurement will have a negligible contribution to the final weight. At this stage, an estimation of all states is made according to the weighted mean.

Next, a Kalman filter measurement update is made for the nonlinear states given the measurements from the IMU and wheel speed sensors. The particles are then propagated through the vehicle model, the linear states are updated through the Kalman filter prediction update, and the next iteration is initialized.

During the whole algorithm, the unnormalized weights are stored as log-likelihoods to avoid computational problems with small or large numbers. For summation when normalizing, the logarithmic summation identity is used,

$$\log(\sum W_j) = \log(W_{max}) + \log\left(1 + \sum_{j \neq max} \exp\left(\log W_j - \log W_{max}\right)\right), \tag{6.14}$$

where $W_{max} = \max(\{W_1, W_2, ...\})$. To improve computational speed, the expected number of measurements is only calculated for the estimated state, hence, all particles have the same $\hat{\omega}$ for each landmark.

7

Results

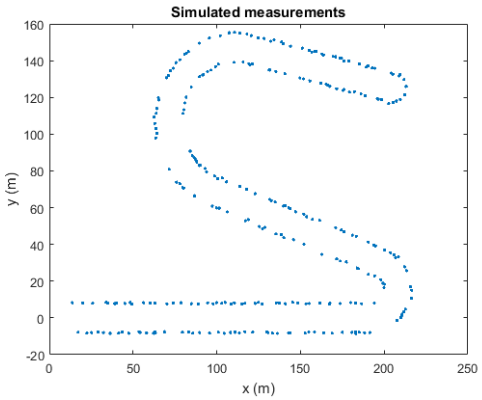
In this chapter, the results are presented. The main focus of the thesis is to develop a mapping algorithm and a localization algorithm that together perform as accurate localization as possible. First, both mapping algorithms are validated using simulated data, where the true state of all landmarks are known. The algorithm is also validated with the test track using an aerial photo as reference, and in terms of how well the localization performs. The localization is evaluated on the recorded test track data using several different maps; this evaluates the combination of the localization and mapping together.

7.1 Data sets

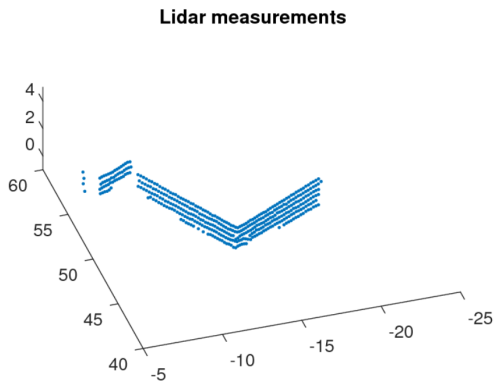
There are three data sets used: one simulated environment, and two data sets recorded on a test track. The recorded data sets are two different runs in the same scenario, where one is used for mapping and one for evaluation of the localization.

Examples of the lidar data and the simulated data is shown in Figure 7.1, which shows the measurements in the global frame. This should mimic the poles of a guard rail along a road with two distinct turns. The measurements are normally distributed around a point source, where both the location of the point and the variance of the measurements are known.

The data sets that were collected at the test track have been filtered by removing measurements that does not have 3 or more measurements aligned vertically within a small radius. This is done to remove measurements from the road or inclined nearby hills. Mapping such data into landmarks would not be optimal for localization, since such landmarks would be very sensitive to small changes in the vehicle roll or pitch. Another reason to pre-filter the data is to reduce the number of measurements and in turn reduce the computational effort needed.



(a) Measurements from the simulated data.



(b) An example of the collected lidar measurements from one frame.

Figure 7.1: Exampels of the used data sets. The simulated data is from a whole run, while the lidar data is only a portion of the data collected in one time frame.

7.2 Map validation

The mapping functions are evaluated on simulated data as well as on recorded data. The performance for the simulated data is evaluated by comparing the estimated number of landmarks with the known truth and with the integrated squared error measure. For the recorded data, the mapping is evaluated by the localization performance.

7.2.1 Simulated data

The two mapping methods are compared on the simulated data using a known number of landmarks. The evaluation is done both by comparing the number of landmarks estimated, and visually how well the map represents the data. The PMBM map is created using prior parameters $S_0 = 2I$, $v_0 = 5$, $\beta_0 = 0.2$, $\alpha_0 = 0.1$. The parameters are chosen by trial and error, until the estimated landmarks approximately represent the true landmarks. The resulting landmarks are shown in Figure 7.2 where the extent of the estimated landmarks are shown in red. The proximity map is created using the parameter $d = 1$ m which gives a visually similar map as the PMBM map. The number of landmarks are closer to the actual number, as can be seen in table 7.1.

Map	Landmarks	ISE
True	276	
PMBM	218	3.2398e+05
Proximity	260	1.5393e+05

Table 7.1: Mapping result on simulated data

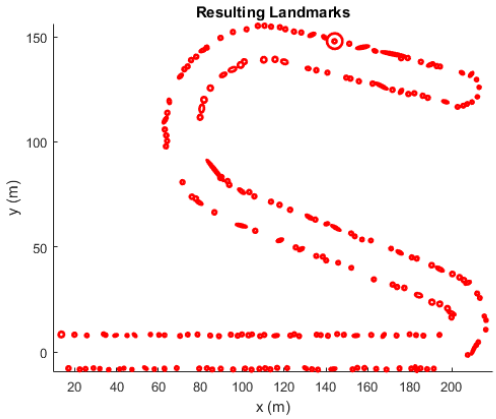


Figure 7.2: Resulting map after convergence, using the PMBM mapping algorithm

Both maps are compared in a section of the map together with the measurements, this is shown in Figure 7.3 and 7.4. This shows how the different methods cluster their measurements.

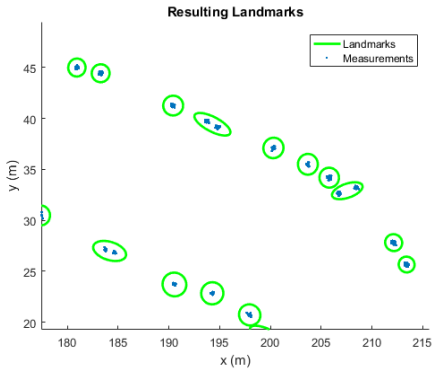


Figure 7.3: PMBM map and measurements in section of map

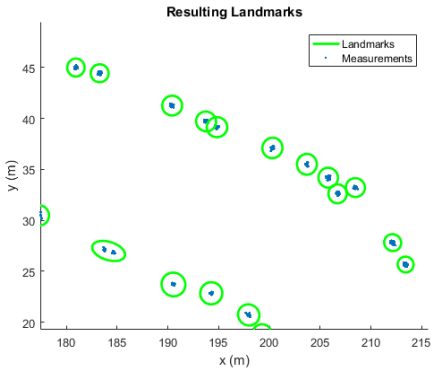


Figure 7.4: Proximity map and measurements in section of map

The two methods are also compared using the integrated squared error (ISE) [25], calculated as

$$\begin{aligned} ISE(\Theta, \hat{\Theta}) &= I_{TT} - 2I_{TE} + I_{EE} \\ I_{TT} &= \sum_{i=1}^{N_T} \sum_{j=1}^{N_T} \omega_i \omega_j N(\mu_i; \mu_j, \Sigma_i + \Sigma_j) \\ I_{TE} &= \sum_{i=1}^{N_T} \sum_{j=1}^{N_E} \omega_i \hat{\omega}_j N(\mu_i; \hat{\mu}_j, \Sigma_i + \hat{\Sigma}_j) \\ I_{EE} &= \sum_{i=1}^{N_E} \sum_{j=1}^{N_E} \hat{\omega}_i \hat{\omega}_j N(\hat{\mu}_i; \hat{\mu}_j, \hat{\Sigma}_i + \hat{\Sigma}_j) \end{aligned} \tag{7.1}$$

where N_T is the number of true landmarks, N_E is the estimated number of landmarks, ω is the number of measurements, μ is the mean of a landmark and Σ is the extent of a landmark. The ISE is presented together with the number of estimated landmarks for each method in Table 7.1.

7.2.2 Test track data

The test track data is recorded in two runs on the same road. One of the runs are used for mapping and the other one is used for evaluation. In Figure 7.5 and Figure 7.6, the mapping method is evaluated using aerial photos of the scenario. Distinct landmarks, such as the houses to the south in the figures, and the guardrails, are clearly represented in the map. Notice also that landmarks that are less distinct, such as trees and bushes, have a more rounded shape.

Figure 7.5: The testscenario in Trollhättan. Image from [7].

Figure 7.6: The figure shows the 3 dimensional map projected onto the scenario. Image from [7].

7.3 Localization performance

The localization is evaluated together with the map on the test track data. The localization is tested with an unknown starting location within 10×10 m^2 around the true starting location, to verify that, e.g., a GPS position can be used as an initial condition. The map and localization performance is evaluated by observing that the drift of the IMU is rejected and that the localization stays below the given performance from Section 1.1.

The test track scenario is described in Section 2.4, where the road has been driven twice. One run is used for mapping, and one is used during localization. The map is created using the DGPS sensor while the localization is performed without GPS, only relying on the on-board sensors in the vehicle as well as the lidar. All evaluations are made using the prior parameter settings as

$$Q^n = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & \pi/180 \end{bmatrix}, R = \begin{bmatrix} 8.08e\text{-}4 & 0 \\ 0 & 1.5e\text{-}5 \end{bmatrix}, Q^l = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{7.2}$$

The localization filter is implemented as described in Chapter 6 and the performance with only the IMU is evaluated to compare and verify the drift of this sensor system, the errors is shown in Figure 7.7.

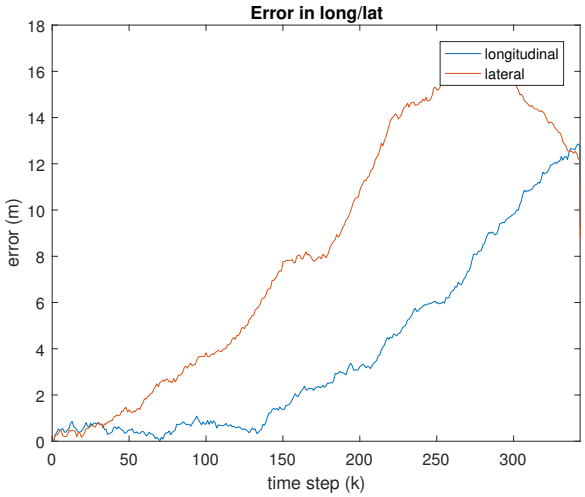


Figure 7.7: Longitudinal and lateral error using only the IMU

The map created using the PMBM method is evaluated in Figure 7.8, and shows the lateral and longitudinal error of the vehicle during localization. The map used is the one which gives the best performance during localization. The prior paramteters for mapping are shown in Table 7.2 and Table 7.3. However, only the localization result from the PMBM map with Setting 3 and Proximity map with $d = 0.5$ is shown since they performed best localization: both lower maximum longitudinal error, and lower average error. The performance of all maps are evaluated on the same data sets with the same localization filter. This gives the result in Figure 7.10.

Setting	S_0	v_0	α_0	β_0
1	100	5	0.1	0.5
2	50	5	0.1	0.5
3	10	50	0.1	0.5
4	10	5	1	0.5
5	10	5	0.1	0.5
6	1	5	0.1	0.5

Table 7.2: Map parameters during test for the PMBM method.

d [m]
/data set 3
2
1
0.5

Table 7.3: Map parameters during test for the proximity method.

The problem of uncertain initial position is also evaluated using the PMBM map. Using an unknown initial position of an area of 100m^2 around the starting position

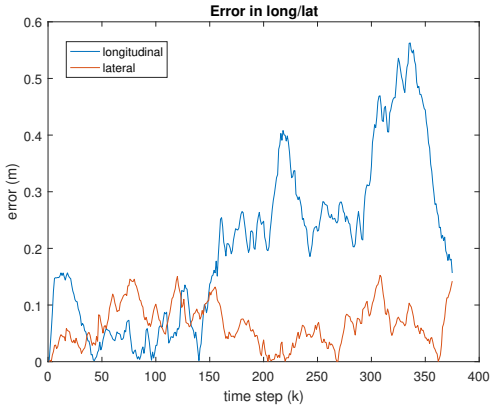


Figure 7.8: Longitudinal and lateral error using the map in Figure 7.9 and 500 particles.

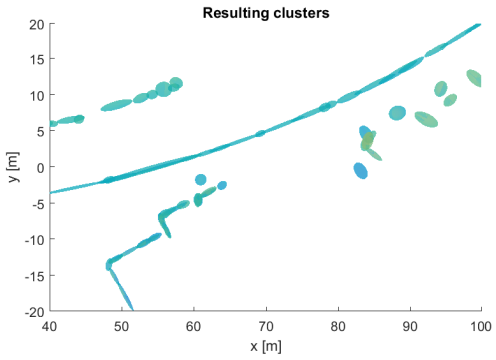


Figure 7.9: A small section of the map constructed using PMBM with parameters according to setting 3 in table 7.2. The total number of clusters is 190.

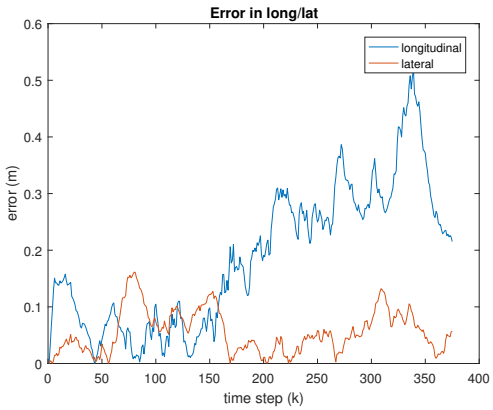


Figure 7.10: Longitudinal and lateral error using the map in Figure 7.11 and 500 particles.

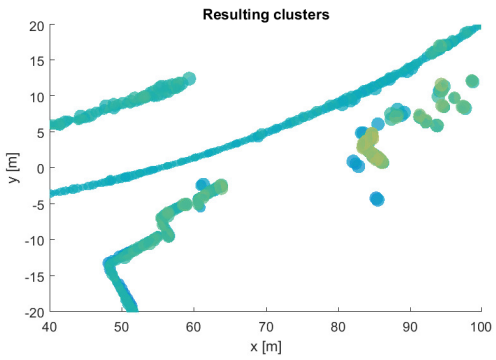


Figure 7.11: A small section map constructed using proximity clustering with distance parameter 0.5 m. The total number of clusters is 1452.

Map	Long/Lat	Mean	Std	Maximum
PMBM	Long	0.209	0.1468	0.5627
Proximity	Long	0.1882	0.1254	0.52
PMBM	Lat	0.06549	0.03639	0.1525
Proximity	Lat	0.05584	0.03782	0.1614

Table 7.4: Errors during localization in meters.

yields the result in Figure 7.12. Where the estimated position is unknown initially but does move closer to the result from the localization with known initial position.

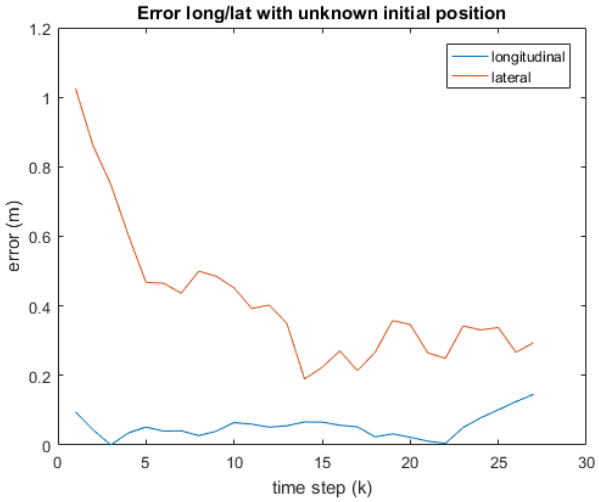


Figure 7.12: Lateral and longitudinal error when unknown initial position is used.

8

Discussion

The mapping results from the simulated data show that the proximity map have an amount of landmarks closer to the actual amount compared to the PMBM map. This indicates that the proximity map is estimating the true landmarks better. The map produced by the proximity method also have a lower ISE than the PMBM method, which also indicates that it is a better representation of the true landmarks. The Figures 7.3 and 7.4 show that both methods are similar, but the PMBM method clusters two landmarks into one in more occasions than the proximity method; this is the reason why the ISE differs between the methods.

This result is expected since the true landmarks are fairly distinct and all of them have the same extent. This makes the environment suitable to the proximity method, where the distance parameter can be set to match the extent of the landmarks. However the advantage of using the PMBM method is that landmarks of different extents can be estimated. The results indicates that both methods work as expected. It is worth noting that both methods could be tuned to match this scenario better, or possibly even exactly.

The test scenario that is used for data collection is challenging for mapping and localization. An easier environment would consist of many distinct and vertical landmarks that are spread out in the environment, such as posts, houses, fences and so on. In the given test scenario there exists only few distinct landmarks, including three small buildings in the beginning of the run, two posts and a guardrail.

When performing localization using the test track data, both mapping methods completely remove the drift in position that is induced by the IMU localization. This indicates that the global positioning for both of these methods do work. The lateral position is better estimated than the longitudinal position for both methods, as can be seen in Figures 7.8 and 7.10. This is most likely due to the guardrail. Since the guardrail is pointing in the direction of the road, measurements originating from the guardrail have a very wide range spread in the longitudinal direction, and a very narrow spread in the lateral direction. This means that there exists a very clear reference in lateral position during the whole run. However the longitudinal position only have a clear reference in the beginning of the run in the form of buildings and a post. After the buildings, the longitudinal reference consists of trees and bushes, which is challenging, because the measurements become less distinct, due to leaves

and branches. This might be the reason why we see a less accurate localization in the longitudinal direction.

Both methods gives similar performance during the evaluation, where they both succeed in keeping the localization under 0.2 m laterally and 1 m longitudinally. When comparing the complexity of the methods the proximity method need a higher number of landmarks to achieve the same accuracy as the PMBM method, possibly due to the PMBM method's ability to cluster landmarks of different shape and size. The number of landmarks increases with the area that is mapped and keeping the complexity of the map as low as possible is desirable when mapping larger areas, since memory often is limited. This is where the more advanced method using PMBM might be preferable. However, the proximity method can be tuned to also cluster fewer and larger landmarks while still performing decent localization, making it hard to draw any conclusions which method is preferable for mapping of larger areas.

On the other hand, the proximity method is less computationally complex, and is therefore much less time consuming. Even though computational performance was not the focus in this thesis, it is worth noting that the method of sampling partitions is rather demanding. However, alternative faster methods could be considered to handle data association in order to approximate the posterior map. However, when considering the mapping problem separate from the localization, as opposed to the SLAM problem, the mapping is done offline. Computational efficiency is therefore less important.

The system is capable of finding the true position when the initial position is unknown, as can be seen in Figure 7.12. This is desired if the system should work as a complement to, e.g., a GPS system, or other applications where the initial position is known with some standard deviation.

An objection to the proposed method might concern the choice of model for the extended object. It is questionable if the assumption that measurements from a landmark are normally distributed around the landmark mean is the best model. In good conditions, the lidar tends to collect dense and evenly spread out measurements with low noise. This means that the shape of landmarks is more visible compared to, e.g., a radar sensor, where measurements often originates from a single point with high reflectivity. This reasoning might suggest that some sort of shape from which measurements originate uniformly would be a better choice for the landmark extent in the presented case, and that the proposed model would fit better for a radar sensor. This is an interesting topic for future research.

9

Conclusion

This study shows that the lidar sensor used has high accuracy and resolution, this is desirable for a mapping and localization problem. It is also shown to be suitable for environments similar to the presented scenario.

The combination of a PMBM map with a marginalized particle filter does position the vehicle under the proposed limit of 0.2 m laterally and 1 m longitudinally in this scenario. It is also noted that the properties of particle filters, specifically the weighting function, fit the proposed map well in that no explicit data association is needed.

Using the more complex PMBM mapping method, a map with relatively few landmarks is achieved compared to the simpler proximity landmarking method with similar localization accuracy. This property is vital when mapping larger areas.

Even though the localization is accurate, the choice of landmark model might not fit the lidar sensor. A radar sensor might fit the model better, this is a subject for further investigation.

Bibliography

[1] W. H. Organization, *Global status report on road safety 2015*. World Health Organization, 2015.

[2] S. Singh, "Critical reasons for crashes investigated in the national motor vehicle crash causation survey," Tech. Rep., 2015.

[3] J. B. Greenblatt and S. Saxena, "Autonomous taxis could greatly reduce greenhouse-gas emissions of us light-duty vehicles," *nature climate change*, vol. 5, no. 9, pp. 860–863, 2015.

[4] D. J. Fagnant and K. M. Kockelman, "The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios," *Transportation Research Part C: Emerging Technologies*, vol. 40, pp. 1–13, 2014.

[5] M. Lundgren, E. Stenborg, L. Svensson, and L. Hammarstrand, "Vehicle self-localization using off-the-shelf sensors and a detailed map," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2014, pp. 522–528.

[6] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[7] Imagery ©2017 DigitalGlobe, Lantmäteriet/Metria, Map data ©2017 Google. [Online]. Available: https://www.google.se/maps/place/Hjulkvarn-Stallbacka,+Trollh%C3%A4ttan/@58.3135156,12.3438969,238m/data=!3m1!1e3!4m5!3m4!1s0x46453cbdc3f77599:0x8fe4cee3ee9f3c1!8m2!3d58.3092419!4d12.3266249

[8] M. Lundgren, *Bayesian filtering for automotive applications*. Chalmers University of Technology, 2015.

[9] G. Burgers, P. Jan van Leeuwen, and G. Evensen, "Analysis scheme in the ensemble kalman filter," *Monthly weather review*, vol. 126, no. 6, pp. 1719–1724, 1998.

[10] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on signal processing*, vol. 50, no. 2, pp. 174–188, 2002.

[11] T. Schon, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2279–2289, 2005.

[12] B. T. Vo, *Random finite sets in multi-object filtering*. University of Western Australia, 2008.

[13] K. Granstrom, M. Fatemi, and L. Svensson, "Poisson multi-Bernoulli conjugate prior for multiple extended object estimation," *ArXiv e-prints*, May 2016.

[14] K. Gilholm, S. Godsill, S. Maskell, and D. Salmond, "Poisson models for extended target and group tracking," in *Optics & Photonics 2005*. International Society for Optics and Photonics, 2005, pp. 59 130R–59 130R.

[15] R. M. Neal, "Markov chain sampling methods for dirichlet process mixture models," *Journal of computational and graphical statistics*, vol. 9, no. 2, pp. 249–265, 2000.

[16] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. part i. dynamic models," *IEEE Transactions on aerospace and electronic systems*, vol. 39, no. 4, pp. 1333–1364, 2003.

[17] M. Fatemi, K. Granström, L. Svensson, F. J. Ruiz, and L. Hammarstrand, "Poisson multi-bernoulli mapping using gibbs sampling," *IEEE Transactions on Signal Processing*, vol. 65, no. 11, pp. 2814–2827, 2017.

[18] M. Lundgren, L. Svensson, and L. Hammarstrand, "Variational bayesian expectation maximization for radar map estimation." *IEEE Trans. Signal Processing*, vol. 64, no. 6, pp. 1391–1404, 2016.

[19] K. Gilholm, S. Godsill, S. Maskell, and D. Salmond, "Poisson models for extended target and group tracking," in *Optics & Photonics 2005*. International Society for Optics and Photonics, 2005, pp. 59 130R–59 130R.

[20] K. Granstrom and U. Orguner, "A phd filter for tracking multiple extended targets using random matrices," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 5657–5671, 2012.

[21] K. Granström, S. Reuter, M. Fatemi, and L. Svensson, "Pedestrian tracking using velodyne data - stochastic optimization for extended object tracking," in *Proceedings of IEEE Intelligent Vehicles Symposium*, Redondo Beach, CA, USA, Jun. 2017, pp. 39–46.

[22] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[23] J. L. Williams, "An efficient, variational approximation of the best fitting multi-bernoulli filter," *IEEE Transactions on Signal Processing*, vol. 63, no. 1, pp. 258–273, 2015.

[24] K. Granström, C. Lundquist, and U. Orguner, "Tracking rectangular and elliptical extended targets using laser measurements," in *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*. IEEE, 2011, pp. 1–8.

[25] D. F. Crouse, P. Willett, K. Pattipati, and L. Svensson, "A look at gaussian mixture reduction algorithms," in *14th International Conference on Information Fusion*, July 2011, pp. 1–8.