A Bayesian Approach to Map-Aided Vehicle Positioning

Examensarbete utfört i Reglerteknik
vid Tekniska Högskolan i Linköping
av

Peter Hall

Reg nr: LiTH-ISY-EX-3102
A Bayesian Approach to Map-Aided Vehicle Positioning

Examensarbete utfört i Reglerteknik
vid Tekniska Högskolan i Linköping
av

Peter Hall

Reg nr: LiTH-ISY-EX-3102

Supervisor: Urban Forssell
Per-Johan Nordlund
Examiner: Fredrik Gunnarsson

Abstract

There is an increasing market for navigation systems and services associated with positioning in today’s automotive informatics and telematics. Contemporary positioning systems have some drawbacks, mainly related to expensive hardware requirements and that the reliability occasionally is rather low. Therefore it is interesting to develop positioning techniques that do not share these disadvantages.

In this thesis the approach is to use low cost sensor equipment and digitally stored map information to produce a high performance vehicle positioning module. To be able to do this, the problem has been formulated as a nonlinear estimation problem, to which Bayesian estimation has been applied. Two approximate solutions have been evaluated: A grid based method, referred to as the point-mass filter, and a sequential Monte Carlo method.

Simulations show that it is possible to obtain independent position estimates with an accuracy comparable to GPS (Global Positioning System), using this technique. A positioning system based on the suggested method can thus be used stand-alone or as a complement to existing systems.

Key Words: Positioning, Vehicle Navigation, Nonlinear Filtering, Bayesian Estimation, Monte Carlo Methods.
Acknowledgments

First of all I would like to thank my supervisors, Urban Forssell and Per-Johan Nordlund for their support and encouragement during my work on this thesis. I also would like to thank my examiner Fredrik Gunnarson and Professor Fredrik Gustafsson for valuable comments and Ulf Gustafsson for reviewing this thesis. Furthermore, I am grateful to Stefan Ahlqvist, Martin Enqvist and Erik Wernholt for interesting discussions and technical assistance. Finally I would like to thank Anders Stenman for helping me out with \LaTeX{} and the rest of the staff at NIRA Automotive AB in Linköping for making my time there enjoyable.

Linköping, Januari 2001
Peter Hall
Notation

Notational conventions for mathematical symbols and operators and some common abbreviations are presented in this section.

Symbols

\( p(x) \) Probability density function (PDF) for the stochastic variable \( x \).

\( p(x, y) \) Joint PDF for \( x \) and \( y \).

\( p(x|y) \) Conditional PDF for \( x \) given a certain value of \( y \).

\( \hat{x} \) The estimate of a stochastic variable \( x \).

\( \mathbb{R}^n \) The n-dimensional Euclidian space.

\( T_s \) Sample period, i.e. if data is sampled with \( f_s \) Hz, then \( T_s = \frac{1}{f_s} \).

Operators and Functions

\( \mathbb{E}(x) \) Expectation of a stochastic variable \( x \).

\( \|u\|_2 \) The 2-norm of a vector \( u \).

\( tr(A) \) The trace of a matrix \( A \).

Abbreviations

ABS Anti-Lock Braking System.
CAN Controller Area Network.
CEP Circular Error Probable.
DGPS Differential GPS.
EKF Extended Kalman Filter.
GLONASS Global Navigation Satellite System.
GPS Global Positioning System.
GUI Graphical User Interface.
INS Inertial Navigation System.
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAP</td>
<td>Maximum A Posteriori.</td>
</tr>
<tr>
<td>MC</td>
<td>Monte Carlo.</td>
</tr>
<tr>
<td>MMS</td>
<td>Minimum Mean Square.</td>
</tr>
<tr>
<td>PDF</td>
<td>Probability Density Function</td>
</tr>
<tr>
<td>PMF</td>
<td>Point-Mass Filter.</td>
</tr>
<tr>
<td>RMSE</td>
<td>Root Mean Square Error.</td>
</tr>
<tr>
<td>SIR</td>
<td>Sampling Importance Resampling.</td>
</tr>
<tr>
<td>SIS</td>
<td>Sequential Importance Sampling.</td>
</tr>
</tbody>
</table>
Contents

1 Introduction 1
1.1 Background .............................. 1
1.2 NIRA Automotive AB ...................... 2
1.3 Problem Specification ..................... 3
1.4 Objectives .............................. 3
1.5 Thesis Outline ............................ 4

2 Positioning Technologies 5
2.1 Coordinate Systems ....................... 5
2.2 Dead Reckoning ........................... 6
2.3 Relative Sensors .......................... 6
   2.3.1 Gyroscopes .......................... 6
   2.3.2 Accelerometers ...................... 7
   2.3.3 Wheel Speed Sensors ................. 7
2.4 Additional Sources of Error ............... 8
2.5 Absolute Sensors ........................ 8
   2.5.1 Satellite-based Navigation Systems .... 8
   2.5.2 Magnetic Compasses ................ 9
   2.5.3 Positioning in Cellular Radio Systems ... 10
2.6 Sensor Fusion ........................... 10
2.7 Map Matching ............................ 11
   2.7.1 Semi-Deterministic Algorithms ........ 11
   2.7.2 Probabilistic Algorithms ............. 11
2.8 Map Representation ....................... 12

3 Map-Aided Positioning 13
3.1 Problem Specification .................... 13
3.2 Limitations .............................. 14
3.3 Methods ................................. 14

4 Bayesian Estimation 15
4.1 Basics ................................. 15
4.2 Recursive Estimation .................... 15
4.3 Grid Based Methods ...................... 17
   4.3.1 The Point-Mass Filter ............... 17
   4.3.2 Grid Adaptation .................... 19
4.4 Monte Carlo Methods .................... 20
   4.4.1 Monte Carlo Integration ............. 20
List of Figures

4.4.2 Sequential Monte Carlo Methods .................. 21
4.4.3 Particle Filters ................................... 22

5 Vehicle Positioning ................................. 25
5.1 Implementation .................................... 25
  5.1.1 The Point-Mass Filter ......................... 25
  5.1.2 The Particle Filter ............................ 26
5.2 Performance Evaluation ............................. 26
  5.2.1 Measurement Data Acquisition .................. 26
  5.2.2 Particle Clustering ............................ 28
  5.2.3 Improvements of the Particle Filter Implementation ........... 30
  5.2.4 Evaluation Using Real Measurements .......... 31
  5.2.5 Monte Carlo Simulations ....................... 35
5.3 Discussion ......................................... 38

6 Heading Angle Estimation ............................ 39
6.1 Model Extension .................................... 39
6.2 Implementation .................................... 40
6.3 Performance Evaluation ............................. 40
  6.3.1 Monte Carlo Simulations ....................... 41
  6.3.2 Robustness .................................... 41
6.4 Discussion ......................................... 44

7 Conclusions .......................................... 47

8 Future Work .......................................... 47

References ............................................. 48

Appendix A: Bayes’ Rule ................................ 50

Appendix B: Bayesian Solution to the 3D Problem .... 51

List of Figures

2.1 Coordinate systems .................................. 5
4.1 The point-mass representation ....................... 19
4.2 Importance weighting and resampling procedure of the particle filter .................................. 23
5.1 Convergence problems with the particle filter ............ 27
List of Figures

5.2 The test drives used in the simulations. 28
5.3 An example of particle clustering 29
5.4 Simulation results of the PMF using data from sequence $S1$. 32
5.5 Simulation results of the particle filter using data from sequence $S1$. 32
5.6 The prior editing ratio for the simulation in Figure 5.5. 33
5.7 Simulation results of the particle filter when the track is lost. 34
5.8 RMSE for the particle filter based on 100 Monte Carlo simulations 36
5.9 Monte Carlo simulations of two different driving scenarios using the two-dimensional particle filter 37
6.1 Monte Carlo simulations of two different driving scenarios using the two-dimensional particle filter 42
6.2 Estimated heading angle when the gyro signal was used 43
6.3 Estimated heading angle when the wheel-speed signals were used to compute the yaw rate 44
1 Introduction

1.1 Background

Vehicle navigation systems and various services associated with vehicle positioning, e.g. traffic information, emergency services, anti-theft devices and “yellow-pages”\(^1\) services, is a growing market in the field of automotive information systems. Contemporary systems for these purposes almost exclusively rely on position information provided by external sources such as GPS (Global Positioning System). Although these satellite-based systems are very accurate\(^2\) they suffer from disturbances, especially in dense urban areas, woodlands and tunnels, and therefore occasionally have a significant performance deterioration. In order to maintain high performance under these circumstances, the systems often feature dead-reckoning capabilities. This means that relative position information, provided by e.g. wheel-speed sensors or inertial sensors, is utilized to predict future positions of the vehicle. Another additional function in many positioning systems is some kind of map-matching, i.e. information in a digital map is utilized to enhance the accuracy of the position fix. The general performance of GPS-based systems has improved somewhat due to the removal of the selected availability function, which was used to distort the civilian GPS-signal in order to lower its accuracy. Despite the effort put on improvements of satellite-based navigation systems, problems with accuracy in the scenarios such as the ones mentioned above remain.

Today’s navigation systems for cars are either delivered together with the vehicle or sold by retailers as self-contained systems. Since there is an increasing use of additional sensors in cars, e.g. wheel-speed sensors in the ABS (Anti-Lock Braking System) or inertial sensors in anti-skid systems, it is possible to make use of this information for positioning purposes. The accessibility of these signals is facilitated by the increasing use of a common data bus, such as the CAN (Controller Area Network), in the vehicle.

A drawback of built-in systems is that they in general require rather expensive hardware equipment, such as a GPS-receiver, a CD-ROM drive for digital map access and some kind of display unit. Furthermore, these systems often require manufacturer specific software and maps.

A current trend is to make the navigation unit portable. There exist several handheld units on the market, usually equipped with a GPS-receiver and

\(^1\)Answers to questions like e.g. “Where is the nearest gas station?”

\(^2\)In this thesis GPS will be used as a performance reference.
maps stored in an internal memory. In general these systems are not suited for vehicle navigation and are basically handheld GPS-receivers with map displaying capabilities.

In order to reduce the price of the complete navigation package, several different solutions are represented on the market. For example, the CD-ROM unit can be replaced by downloadable maps, route guidance calculations can be delegated to a call center via a cellular connection etc. In the case of handheld navigators, there exist systems based on personal digital assistants (PDA’s), which provide the fundamental computing and displaying hardware. These can be attached to a compact GPS-receiver and furnished with navigation software.

The backbone of a vehicle navigation system is a high precision positioning module, available at a low cost. In other words, to continuously provide the driver with reliable information about the position of the vehicle, without requiring expensive additional hardware. It is thus of great interest to develop alternative positioning techniques that does not have the same weaknesses as the satellite-based ones, and that provides state-of-the-art performance at a low cost. The latter is interesting for car manufacturers as well as for the car owners.

1.2 NIRA Automotive AB

The project on which this master thesis is based, has been performed at NIRA Automotive AB in Linköping. NIRA Automotive AB is a research and development company specialized in design and implementation of reliable, time-critical electronic control systems for automotive and vehicular applications as well as for harsh industrial environments.

The office in Linköping is active mainly in the areas of signal processing and control. A basic concept is to develop systems, which use information from existing or low-cost add-on sensors to compute high-precision virtual sensor signals. Examples of such projects are the development of a road friction indicator and a virtual tire pressure sensor. Other responsibilities are the development of NIRA’s antispin systems for cars and motorcycles.

More information about NIRA Automotive AB and other projects can be found on their website, www.nira.se.
1.3 Problem Specification

A fundamental component in a navigation system is a reliable and accurate positioning module. Positioning is the determination of the global (or absolute) coordinates of the vehicle, i.e. to specify the vehicle’s position in a known reference frame. One example of such a reference frame is a geographical map.

In order to present correct information about the vehicle location to the driver, the position provided from the positioning module is usually matched to a digital map. The basic idea of map matching is to compare the output from the positioning module, i.e. the trajectory of the vehicle, against the shape of nearby roads and choose the closest match.

The idea in this thesis is to fuse positioning and map matching into a map-aided positioning algorithm, which does not require external absolute position fixes, e.g. provided by GPS. This is achieved by integrating signals from relative sensors (e.g. wheel speed sensors, accelerometers, gyroscopes) with the information contained in a digital map. As can be imagined, this is a very complex and nonlinear filtering problem, to which Bayesian estimation methods are applied. This approach is inspired by the Ph.D. work of Bergman [1]. The ideas on map-aided positioning, as defined here, are patent pending.

The problem has been divided into two parts:

- To estimate the absolute position when the absolute heading is known.
- To extend the model and estimate the absolute heading angle as well.

1.4 Objectives

The purpose of this thesis is to investigate the possibilities of applying Bayesian estimation techniques to the problem described in the previous section.

To achieve this, the following issues have to be considered:

- A mathematical model suitable for the vehicle positioning problem has to be developed. This includes how to model the map information.
- A simulation environment, where the studied algorithms are to be implemented, has to be developed.
- The positioning filters should be evaluated using real measurement data, collected during authentic driving scenarios.
1.5 Thesis Outline

In Section 2 a review of position technologies as well as common sources of error and an introduction to map-matching techniques is presented as a background to the problems discussed in this report. Section 3 describes the first part of the actual problem, the estimation of the absolute position when the heading angle is known. Furthermore an introduction to Bayesian estimation methods is given in Section 4, while Section 5 treats the application of these methods to the problem in Section 3. In Section 6 the problem is extended to include estimation of the heading angle. Finally, the report is concluded in Section 7, while future work is discussed in Section 8.
2 Positioning Technologies

There are mainly three positioning technologies used in vehicle navigation: stand-alone (e.g., dead reckoning), satellite-based (e.g., GPS), and terrestrial radio based. This section will give a brief review of these areas as well as an introduction to the basic ideas of map matching. A more thorough introduction to vehicle positioning and navigation systems can be found in [12].

2.1 Coordinate Systems

Positioning is the procedure of determining the global coordinates of an object. In this thesis two coordinate systems will be used for this purpose. The global system consists of the two-dimensional Cartesian coordinates in the east and north direction, \((x^1, x^2)\). The orientation of the vehicle is defined by a body-fixed system (see Figure 2.1). The vehicle’s direction of travel is aligned with the longitudinal axis, \(X^{\text{long}}\). The angle from the north axis to this axis is referred to as the heading angle or yaw angle, \(\psi\). The second body-fixed axis, \(X^{\text{lat}}\), is referred to as the lateral axis. The yaw rate is defined as the time derivative of the yaw angle, \(\dot{\psi}\).

![Figure 2.1](image)

**Figure 2.1.** The coordinate systems. The global frame consists of the east and north components, \((x^1, x^2)\), relative to a map reference point. The body-fixed frame defines the longitudinal and lateral direction \((X^{\text{long}}, X^{\text{lat}})\).
2.2 Dead Reckoning

Dead reckoning is a primitive positioning technique that involves integration of relative movements with respect to time starting from a known reference point. The idea is that, if the initial position is known, it is possible to calculate the vehicle position at any instance by measuring the displacements in distance and direction.

A common dead-reckoning system is the Inertial Navigation System (INS) which uses inertial sensors like accelerometers and gyroscopes to measure the magnitude of the acceleration and the angular rates (e.g. yaw rate) of the vehicle.

2.3 Relative Sensors

A generic term for sensors that measure changes in position or heading, i.e. movements relative to an absolute position, is relative sensors [12]. Apart from the inertial sensors there exist several other relative automotive sensors, e.g. transmission pickups, which measure the angular position of the transmission shaft, or wheel speed sensors (such as the ones used in the Anti-Lock Braking System, ABS).

Important characteristics of dead reckoning using relative sensors are that if the initial position is unknown, the information can not be used for absolute positioning purposes. Another feature is that measurement errors are accumulated due to the integration operation and after some time the deviation from the actual trajectory can be considerably large.

2.3.1 Gyroscopes

A rate-sensing gyroscope measures the angular rate using either mechanical, optical, pneumatic or vibrational devices. The performance of the gyroscope is characterized by several factors. For example the measurements usually suffer from scale factor errors, drift (offset error) and measurement noise.

The output from a gyroscope can be modeled as

\[
\dot{\psi}_{\text{GYRO}}(t) = (1 + \alpha)\dot{\psi}(t) + \delta_{\text{GYRO}} + C \cdot t + e(t),
\]

where \(\alpha\) is a scale factor error, \(\delta_{\text{GYRO}}\) is an offset term, \(C\) is a linear drift term and \(e\) is measurement noise.
2.3 Relative Sensors

2.3.2 Accelerometers

An accelerometer can be used to measure the acceleration in a given direction, e.g. along the coordinate axis of a body-fixed frame. In similarity with the gyroscope, the following model can be used for the accelerometer signal

\[ a_{\text{ACC}}(t) = (1 + \alpha)a(t) + \delta_{\text{ACC}} + C \cdot t + e(t). \]  

(2.2)

2.3.3 Wheel Speed Sensors

A wheel speed sensor consists of a toothed ferrous wheel, mounted on the wheel axle, and some kind of sensing element (e.g. a variable reluctance or a Hall-effect sensor [12]). The sensor measures the rotational velocity of the wheel, \( \omega_w \). Assuming that the nominal radius of the wheel is \( r_n \) and that the actual radius has some offset, \( \delta_w \), the wheel speed of a free-rolling (undriven) wheel can be modeled as

\[ v_w(t) = \frac{(r_n + \delta_w)\omega_w(t)}{n} + e(t), \]  

(2.3)

where the measurement noise is modeled as an additive term, \( e \).

If the wheel is driven, or during braking, another error is introduced due to the fact that the absolute velocity of the wheel, \( v_w \), does no longer coincide with its circumferential velocity as assumed in (2.3). This error, which plays an important role in automotive engineering, is called wheel slip and is usually defined as the relative difference [7]

\[ s(t) = \frac{(r_n + \delta_w)\omega_w(t) - v_w(t)}{v_w(t)}. \]  

(2.4)

Wheel speed sensors can be used to estimate the longitudinal velocity of the vehicle. Usually this is achieved by averaging the angular rates of either the front or the rear wheel pairs and multiplying by a proper scale factor (e.g. the nominal wheel radius)

\[ v_{\text{long}}(t) = r_n \frac{\omega_{w, \text{right}}(t) + \omega_{w, \text{left}}(t)}{2}. \]  

(2.5)

A yaw rate estimate can be obtained by calculating the difference in angular rate between the right and left rear wheels and dividing by the rear axle length, \( l_{\text{rear}} \).
\[ \dot{\psi}(t) = r_n \frac{\omega_{w,\text{right}}(t) - \omega_{w,\text{left}}(t)}{l_{\text{rear}}} \]  \hspace{1cm} (2.6)

This technique to provide both longitudinal velocity and yaw rate information from a pair of wheel speed sensors is called differential odometry [12].

### 2.4 Additional Sources of Error

Apart from inaccuracy in the sensors there are some other sources of error that have to be considered. Some examples are the wheel radii offset and the wheel slip mentioned above. Another disturbance is the road inclination, which, if not compensated for, causes erroneous sensor signals and also conflicts with the implicit assumption that the vehicle is located on a flat map.

**Example 2.1.** A longitudinal accelerometer measures the magnitude of the acceleration in the direction of travel. If the road has a slight inclination, \( \alpha \), the accelerometer will also measure a component of the gravitational acceleration, \( g \). If the other errors are assumed to be zero this yields

\[ a_{\text{acc}}(t) = a(t) + g \sin \alpha \]  \hspace{1cm} (2.7)

Even if the inclination is moderate this will give a significant deterioration in a dead-reckoning system due to the accumulation of errors, described in Section 2.3.

### 2.5 Absolute Sensors

The opposite to a relative sensor is an absolute sensor, which measures the absolute position (e.g. GPS) or the absolute heading (e.g. magnetic compasses).

#### 2.5.1 Satellite-based Navigation Systems

There are two global satellite-based navigation systems available today, originally developed for military purposes: GPS, which is the U.S. alternative and most commonly used in vehicle navigation systems, and the corresponding Russian system GLONASS. Some more expensive navigation systems use
2.5 Absolute Sensors

a combination of these systems in order to achieve higher performance. The European Space Agency (esa) is also planning for a third, all European system, which is to be operational from 2005 under current plans.

The Global Positioning System [12] (GPS) is a satellite-based radio navigation system, that consists of 24 satellites arranged in six orbital planes, designed to provide worldwide coverage 24 hours per day. Each of the satellites carries a high precision atomic clock and broadcasts encoded messages at regular and known time instants. Each message includes an identity number and the location of the satellite. A receiver on the ground decodes the signal and uses the signal propagation time to calculate a pseudorange. To determine its position, the receiver needs to know the pseudoranges to the satellites as well as their locations. If a position estimate in three dimensions is desired this information should be available from at least four unique satellites. Three satellites are required to solve the three unknown Cartesian coordinates and the extra satellite handles the fact that the measured pseudorange is not the actual range, due to receiver clock bias.

If a reference receiver with known position is used to calculate a differential correction for each satellite, the accuracy of GPS can be significantly improved. This idea is adopted in the differential GPS (DGPS), which uses a constellation of ground-based reference stations to provide the mobile receivers with position fixes.

The position obtained from a GPS receiver includes several error terms of more or less importance for the general performance of the system. For example there are range errors due to ionospheric and tropospheric delay, but such disturbances can usually be estimated using atmospheric models. Other errors that are involved are receiver noise, multipath propagation error (i.e. when the signal is received with a time lag caused by reflections), and satellite orbit error. For vehicle navigation applications, GPS receiver performance under heavy foliage and in urban canyon areas is particularly important. Overhead foliage may cause signal attenuation and high buildings may block the satellite signals or cause multipath errors.

2.5.2 Magnetic Compasses

By absolute heading one usually refers to the orientation of the vehicle relative to the north. This quantity can be determined by measuring the earth’s magnetic field, e.g. by using a magnetic compass. However, the magnetic north deviates from the true north and this difference is known
as declination. To correct the deviation, which varies both with time and geographical location, a declination table can be used.

The most commonly used type of magnetic compass is the electronic fluxgate compass. Although electronic compasses have higher vibration durability and quicker response than conventional compasses, they still are quite sensitive to magnetic anomalies and noise. However, there exist several methods of improving the performance, e.g. low-pass filtering and correction tables [12].

If a compass is used in a vehicle, the magnetic field of the vehicle itself will interfere with the measurement. This error is often referred to as magnetic deviation, and its magnitude depends on the true heading of the vehicle. One way of compensating for magnetic deviation is to use a look-up table containing correction information for any heading. Another way is to estimate the error dynamically, which is fairly simple since the measured field vector is the sum of the earth’s field vector and the field vector generated by the vehicle.

### 2.5.3 Positioning in Cellular Radio Systems

An example of a terrestrial radio-based system used for positioning are cellular radio systems. When distributing transmission power to handsets in the nearby area, the base stations in the network use rough position estimates based on measurements on the incoming signals. This information can be forwarded to the handset and used e.g. in a yellow-pages service or for navigation purposes. Contemporary systems provide estimates with very low accuracy, but future generations are expected to improve this considerably. An overview of various techniques can be found in [5].

### 2.6 Sensor Fusion

From the discussion in the previous sections, it is obvious that no single sensor can provide completely accurate position information. Thus it is desirable to have a multisensor configuration and merge the available information in order to increase the overall performance.

The idea of using several different sensors and extracting the combined information from the observations is referred to as sensor fusion. There are several advantages with the fusing process, e.g. increased accuracy, increased degree of confidence or reduced ambiguity. This is of course valid only if the sensor models used describe the true properties of the physical sensors.
A common statistical approach to sensor fusion is to use Kalman filtering. A detailed description of this can be found in [9].

2.7 Map Matching

An important part of vehicle navigation is to present the current position in a satisfactory fashion to the driver. Usually this involves displaying the vehicle location on a digital road map. To provide this kind of information, an accurate vehicle position obtained from the positioning module is required. However, since there are errors associated with the position estimates, the actual position does not necessarily agree with the measured position. These errors are often of cumulative nature and thus the deviation is likely to increase with time.

To deal with this discrepancy some kind of map-matching algorithm is often employed to correct the measured position so that it fits on the map. If properly done, this will enhance the accuracy of the position estimate.

The basic idea of conventional map matching is to compare the output from the positioning module, i.e. the trajectory of the vehicle, with the shape of nearby roads and choose the closest match. There exist both semi-deterministic and probabilistic map matching algorithms.

2.7.1 Semi-Deterministic Algorithms

Semi-deterministic map matching is based on the presumption that the vehicle is traveling on the road network. The matching procedure utilizes geometric and topological information to correct the errors in the position estimate. A very simple example of a semi-deterministic match is to snap the position estimate to the nearest road in the network. However, this is only feasible if the estimate and the network model is very accurate. More sophisticated algorithms can be found in [2].

2.7.2 Probabilistic Algorithms

To improve the performance of map-matching algorithms some information about the accuracy of the position estimate should be incorporated with the spatial information of the map. This idea is adopted in probabilistic map-matching algorithms. Based on models of the sensor errors, confidence regions can be defined for the estimate. If these regions are superimposed on the road network it is possible to determine the most probable road segment from which the estimate originates. A major enhancement of this method
over the semi-deterministic method is that the vehicle does not necessarily have to be confined to the road. If no road segment candidate intersects the confidence region the algorithm leaves the position unmatched, i.e. the fix provided by the positioning module is not corrected.

For further discussions about conventional map-matching algorithms see [12], which also describes some alternative methods.

2.8 Map Representation

In order to implement a high performance positioning algorithm, which relies on map information, it is of vital importance that the representation of the map is efficient and accurate.

There are in general two major ways to represent map information in a computer. One is to digitize the paper map using a scanner to produce a raster encoded structure, which will more or less be an exact copy of the original. Another way is to convert the information into a vector-encoded structure. This procedure will extract only the most relevant features of the map resulting in a more efficient and flexible representation. In a navigation application the vector-encoded format is superior to the raster-encoded format concerning the possibility to use mathematical models and calculations as well as the lower memory requirements. Therefore this report will focus on the vector-encoded representation.
3 Map-Aided Positioning

The basic idea in this thesis is to use the information in a digital map together with a standard dead-reckoning system to solve the vehicle positioning problem. This is tackled from a statistical viewpoint using the concept of sensor fusion. Since the approach is quite different from conventional map matching the term *map-aided positioning* is used here. In the following sections, the problem will be formulated mathematically.

3.1 Problem Specification

Let $x_t$ denote the position of the vehicle and let $u_t$ denote the relative movement measured by relative sensors (such as gyroscopes and wheel sensors). If the drift in the relative sensors is modeled by random walk, the state transition equation can be written as

$$x_{t+1} = x_t + u_t + v_t,$$

where $v_t$ is white noise with density function $p_v(\cdot)$. To integrate map information in the model one can introduce measurements, denoted by $y_t$. The relation between these measurements and the vehicle position is modeled as

$$y_t = h(x_t) + e_t,$$

where $h(\cdot)$ is some nonlinear function and $e_t$ is white measurement noise with density function $p_e(\cdot)$. The processes $e_t$ and $v_t$ are statistically independent. A measurement of particular interest is the minimum Euclidean distance from an arbitrary point in the state space, to the road network. The function $h(\cdot)$ thus represents the corresponding distance obtained from the map. Let $\Omega_R$ denote the road network then

$$h(x_t) = \min_{x \in \Omega_R} \|x_t - x\|_2.$$  

The equations (3.1) and (3.2) define a nonlinear recursive estimation problem

$$\begin{cases} x_{t+1} = x_t + u_t + v_t \\ y_t = h(x_t) + e_t, \end{cases}$$

(3.4)

to which Bayesian estimation theory will be applied in Section 4.2.
3.2 Limitations

In the first study of the positioning problem only the absolute position of the vehicle is considered unknown. This means that the absolute heading of the vehicle is assumed to be known, perhaps from measurements by a magnetic fluxgate compass.

Another assumption is that the vehicle is always located on a road. This constraint means that the measurement of the minimum distance from the vehicle to the road is simply zero plus some measurement noise \( \epsilon_t \). The road network, \( \Omega_R \), defines a subspace to the state space, which here is \( \mathbb{R}^2 \). The spatial structure of the road network contains valuable information for positioning.

3.3 Methods

The estimation problem will be studied from a Bayesian viewpoint [1] using two methods to approximate the optimal analytical solution. In the next section a brief review of recursive Bayesian estimation is presented.
4 Bayesian Estimation

4.1 Basics

State space estimation is the procedure of gaining information about the state of a system by observing some kind of related quantity. In a Bayesian framework the state as well as the measurement are treated as stochastic vectors. Let the likelihood of the measurement $y$, given the state $x$ be denoted by $p(y|x)$. By considering the prior knowledge about the state, $p(x)$, i.e. what is known before obtaining the measurement, the joint density of the state and the observation

$$p(x, y) = p(y|x)p(x)$$  \hspace{1cm} (4.1)

defines a statistical model for the estimation problem. Applying Bayes’ rule (Appendix A) yields

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}.$$  \hspace{1cm} (4.2)

This relation describes how the knowledge about the state is updated by the observation. The function $p(x|y)$ is called the posterior probability density function (PDF) and summarizes everything there is to know about the state after the observation. The denominator of (4.2) can be obtained by marginalizing out $x$ in equation (4.1)

$$p(y) = \int p(y|x)p(x) \, dx$$  \hspace{1cm} (4.3)

and thus it can be seen as a normalizing constant.

4.2 Recursive Estimation

In many applications the measurements are obtained in a sequential fashion and if real-time (on-line) performance is demanded, the estimation has to be handled recursively.

The estimation problem stated in Section 3.1 is recursive since the state transition kernel is a Markovian process and the measurements are conditionally independent of earlier observations, given the current state [1]. Let $Y_t = \{y_i\}_{i=0}^t$ denote the available set of measurements at time $t$. If $p(x_t|Y_{t-1})$ is assumed known, the update of the PDF with the new measure-
ment, $y_t$, is obtained by using Bayes’ rule

$$p(x_t|Y_t) = p(x_t|y_t, Y_{t-1}) = \frac{p(y_t|x_t, Y_{t-1})p(x_t|Y_{t-1})}{p(y_t|Y_{t-1})}$$

$$= \frac{p(y_t|x_t)p(x_t|Y_{t-1})}{p(y_t|Y_{t-1})}, \quad (4.4)$$

where the last equality is due to the fact that the new measurement is conditionally independent given the current state.

To complete the recursion one has to calculate the time update of the pdf, i.e. how it is affected by the state transition (3.1). This can be achieved by observing that

$$p(x_{t+1}, x_t|Y_t) = p(x_{t+1}|x_t, Y_t)p(x_t|Y_{t-1}) = p(x_{t+1}|x_t)p(x_t|Y_t), \quad (4.5)$$

which follows from (4.1) and the Markovian property of the state. Marginalization with respect to $x_t$ yields

$$p(x_{t+1}|Y_t) = \int_{\mathbb{R}^x} p(x_{t+1}|x_t)p(x_t|Y_t) \, dx_t. \quad (4.6)$$

Using the state-space model according to (3.4), the following can be noticed

$$p(x_{t+1}|x_t) = p_v(x_{t+1} - x_t - u_t)$$

$$p(y_t|x_t) = p_e(y_t - h(x_t)).$$

The Bayesian solution to the problem (3.4) is thus given by the following expressions

$$p(x_t|Y_t) = \frac{1}{c_t} p_e(y_t - h(x_t))p(x_t|Y_{t-1}) \quad (4.7)$$

$$p(x_{t+1}|Y_t) = \int_{\mathbb{R}^2} p_v(x_{t+1} - x_t - u_t)p(x_t|Y_t) \, dx_t, \quad (4.8)$$

where $c_t$ is a normalizing constant. From these two equations it is possible to calculate the a posteriori probability density function, $p(x_t|Y_t)$, recursively given the measurements $Y_t$. The recursion is initialized with $p(x_0) = p(x_0|Y_{-1})$.

The posterior density summarizes all information about the state $x_t$ given by the measurements and the initial state $x_0$. Thus it is possible to calculate various point estimates given the posterior density. For example the
conditional mean

\[ \hat{x}_t^{\text{MMS}} = \int_{\mathbb{R}^2} x_t p(x_t|Y_t) \, dx_t, \quad (4.9) \]

which is also referred to as the minimum mean square (MMS) estimate. Another possible point estimate is the maximum a posteriori (MAP) estimate

\[ \hat{x}_t^{\text{MAP}} = \arg \max_{\hat{x}_t} p(x_t|Y_t) \quad (4.10) \]

It should be noted that these two point estimates have quite different characteristics. The MMS works best when the posterior density is unimodal and the MAP is very sensitive to outliers.

Choosing any of the point estimates above, the estimation error correlation matrix can be calculated according to

\[ P_t = \int_{\mathbb{R}^2} (x_t - \hat{x}_t)(x_t - \hat{x}_t)^T p(x_t|Y_t) \, dx_t, \quad (4.11) \]

which, if the MMS is used, coincides with the estimation error covariance matrix. This is due to the fact that the MMS is the first moment of \( p(x_t|Y_t) \) and thus

\[ \mathbb{E}(x_t - \hat{x}_t|Y_t) = 0. \quad (4.12) \]

In general it is not possible to evaluate the Bayesian solution analytically. However, there are some well-known special cases when there exist an explicit analytical solution, e.g., if the problem is linear and the noise distribution is Gaussian the Kalman filter can be applied. If the model is nonlinear, but can be locally linearized, the extended Kalman filter (EKF) provides an analytical solution to the linearized problem.

If it is not possible to linearize the model locally or if the noise distribution is non-Gaussian, an alternative approach is to approximate the solution instead of the model, i.e., to approximate the posterior density globally. In this thesis, two approximate solutions to the Bayesian estimation problem will be studied. The first one is a deterministic method, based on numerical integration, and the second one is a stochastic, simulation-based method involving Monte Carlo integration.

### 4.3 Grid Based Methods

#### 4.3.1 The Point-Mass Filter

To approximate the integrals in the Bayesian solution by numerical integration, some kind of quantization of the state space is required. An appealing
straightforward quantization is to apply a uniform grid to the state space and evaluate the PDF in these grid points. Each of the grid points will carry a weight, a sample of the original continuous PDF, and is therefore referred to as point masses. The implementation of this approximation is known as the point-mass filter (PMF) [1].

The grid is characterized by a matrix containing the point masses, a resolution and a reference vector. Usually not all of the grid points are occupied by a nonzero point mass and thus the matrix will be rather sparse. Figure 4.1 shows an example of how this might look like.

Assume that the grid resolution is $\delta$ and the total number of grid points (in $\mathbb{R}^2$) is $N$, then the point-mass filter is described by the following equations (see e.g. [1]):

Algorithm 1 (The Point-Mass Filter)

\[
\begin{align*}
 c_t &= \sum_{n=1}^{N} p_{e_t}(y_t - h(x_t(k)))p(x_t(k)|Y_{t-1})\delta^2 \quad (4.13a) \\
p(x_t(k)|Y_t) &= \frac{1}{c_t} p_{e_t}(y_t - h(x_t(k)))p(x_t(k)|Y_{t-1}) \quad (4.13b) \\
 \hat{x}_t^{\text{MMS}} &= \sum_{n=1}^{N} x_t(n)p(x_t(n)|Y_t)\delta^2 \quad (4.13c) \\
P_t &= \sum_{n=1}^{N} (x_t(n) - \hat{x}_t)(x_t(n) - \hat{x}_t)^T p(x_t(n)|Y_t)\delta^2 \quad (4.13d) \\
x_{t+1}(k) &= x_t(k) + u_t, \quad k = 1, 2, \ldots, N \quad (4.13e) \\
p(x_{t+1}(k)|Y_t) &= \sum_{n=1}^{N} p_{v_t}(x_{t+1}(k) - x_t(n) - u_t)p(x_t(n)|Y_t)\delta^2 \quad (4.13f)
\end{align*}
\]

The equations (4.13a)-(4.13d) is often referred to as the measurement update. During these steps the point-masses are re-weighted according to the information in the new measurement and the estimate and its error covariance are calculated. The next two steps, equations (4.13e) and (4.13f), are called the time update. First the grid points are translated according to the relative movement $u_t$ and then the prior density is convolved with the density of $v_t$, i.e. the prior density is smoothed somewhat due to the uncertainty in the relative movement. A more detailed presentation of the PMF is given in [1].
4.3 Grid Based Methods

Figure 4.1. The grid is characterized by a sparse matrix $P$, containing the point masses, the grid resolution $\delta$ and the reference vector $x_0$.

4.3.2 Grid Adaptation

During the measurement update the point-mass values will increase in parts of the state space where the likelihood is high and decrease in areas where it is unlikely to find the true position. Through the convolution operation in the time update the grid support will be increased. Because of implementational issues, such as computational load, and to ensure high filter performance, it would be desirable to keep the number of point-masses at approximately the same level. This implies that some kind of grid adaptation should be used. Low point-mass values should be removed from the grid and the grid resolution should be adjusted when the number of point-mass values falls outside certain limits.

One way to adjust the resolution is to resample the grid when needed, e.g. the resolution can be decreased by decimation of the grid point matrix and increased by bilinear interpolation. The idea of utilizing an adaptive grid resolution can be justified by the following discussion: When the uncertainty about the vehicle position is high, e.g. when the algorithm is initialized, it is not interesting to have a dense grid. After some iterations the prior density has more or less vanished on larger areas, and the grid resolution should be increased gradually to concentrate on more likely areas. The denser the grid gets, the better does the PMF approximate the Bayesian solution. However if the grid resolution is very high, it would require a lot of grid points, and at a certain level it would be necessary to decrease the resolution due to the computational load. Note that it is important to frequently normalize the PDF to suppress approximation errors.
If $N$ is the number of nonzero grid points, a simple adaptation procedure can be described by the following steps:

- If $N > N_1$, remove every second row and column from the matrix and set the resolution to $2\delta$.
- If $N < N_2$, use bilinear interpolation to increase the number of elements in the matrix and set the resolution to $\delta/2$.

### 4.4 Monte Carlo Methods

An alternative to deterministic methods, such as the PMF, is to use a stochastic (simulation-based) approach.

#### 4.4.1 Monte Carlo Integration

One class of simulation-based methods are the Monte Carlo methods. They aim at approximating integrals of the kind

$$ I = \int_{\mathbb{R}^n} f(x) \pi(x) \, dx, \quad (4.14) $$

where

$$ \int_{\mathbb{R}^n} \pi(x) \, dx = 1, \quad \pi(x) > 0 \quad \forall x \quad (4.15) $$

by the following sum

$$ \hat{I} = \sum_{i=1}^{N} f(\tilde{x}_i). \quad (4.16) $$

The set $\{\tilde{x}_i\}_{i=1}^{N}$ is i.i.d. (independent identically distributed) samples from $\pi(x)$. Some of the integrals of interest in the Bayesian solution, like the MMS estimate (4.9) and the error covariance (4.11), is of the kind in (4.14). In general it is not possible to sample directly from the distribution $\pi(x)$. However this can be tackled e.g. by using the following technique: Let $q(x)$ be a distribution from which samples are easily generated. Then an importance weight can be defined for each sample from $q(x)$ as

$$ w(x_i) = \frac{\pi(x_i)}{q(x_i)} \quad (4.17) $$

Hence the density function $q(x)$ is usually referred to as importance function. The integral in (4.14) can thus be rewritten as
4.4 Monte Carlo Methods

\[ I = \int_{\mathbb{R}^n} f(x) \frac{\pi(x)}{q(x)} q(x) \, dx = \int_{\mathbb{R}^n} f(x)w(x)q(x) \, dx \]  
(4.18)

This integral can be approximated by

\[ \hat{I} = \sum_{i=1}^{N} f(\tilde{x}_i)w(\tilde{x}_i), \]  
(4.19)

where \( \{\tilde{x}_i\}_{i=1}^{N} \) are i.i.d. samples from \( q(x) \).

The importance weights can be seen as correction factors when the value of the density function \( q(x) \) does not agree with value of the true density function \( \pi(x) \). This sampling technique is called importance sampling and the performance is of course dependent on the choice of importance function.

In the Bayesian framework the target distribution is given by

\[ \pi(x) = p(x|y) \propto p(y|x)p(x). \]  
(4.20)

If the prior distribution, \( p(x) \), is used as importance function, the unnormalized weights is given by \( w(x) = p(y|x) \). A more detailed presentation of Monte Carlo methods are given in [1] and [4].

4.4.2 Sequential Monte Carlo Methods

If Monte Carlo integration is used in a recursive estimation problem, the target distribution, \( \pi(x) \), is time dependent

\[ \pi_t(x_t) = p(x_t|Y_t). \]  
(4.21)

In general there exists no explicit analytical expression for this function. This means that recursive, or sequential, Monte Carlo methods have to deal with approximation and propagation of the posterior density function as well. This is achieved by representing it with a set of i.i.d. samples drawn from (4.21) (or approximately by using e.g. importance sampling). These samples are often called particles, because of their similarity with a swarm or cloud of small particles (e.g. dust) that evolves with time, when propagated according to the Bayesian solution. The filters based on sequential Monte Carlo methods are thus often referred to as particle filters.

A major advantage of Monte Carlo methods over numerical integration is that the complexity of the problem does not increase as much with the state
dimension. This can be motivated by the fact that in Monte Carlo integration the samples of the posterior distribution are automatically chosen in the parts of the state space that are important for the integration result. This means that the grid is adaptive in an efficient way and that the user does not have to bother about how to discretize the state space.

4.4.3 Particle Filters

One algorithm that utilizes the sequential Monte Carlo ideas is the Bayesian bootstrap or Sampling Importance Resampling (SIR) filter [1, 4, 8] summarized in Algorithm 2.

Algorithm 2 (Bayesian bootstrap (SIR))

1. Set $t = 0$ and sample $N$ times from $p(x_0)$ to generate the set $\{x_0^i\}_{i=1}^N$.

2. Calculate normalized weights $w_i = \frac{p_{i}(y_n|h(x_i^t))}{\sum_{j=1}^{N} p_{i}(y_n|h(x_j^t))}$.

3. Calculate position estimate $\hat{x}_MMS^t = \sum_{i=1}^{N} w_i x_i^t$.

4. Generate a new set $\{x_0^i\}_{i=1}^N$ by sampling with replacement $N$ times from the old set. The probability for resampling particle $x_i^t$ should be equal to $w_j$.

5. Generate $v_i^t \sim p_{v}(v), i = 1, \ldots, N$
   and predict each particle $x_{t+1}^i = x_i^t + u_t + v_i^t, i = 1, \ldots, N$

6. Set $t = t + 1$ and iterate from 2.

The algorithm is initialized by generating samples from the prior distribution, $p(x_0)$. The samples are then weighted with the information provided by the measurement. These weights are used to estimate the position and the error covariance. The next step is to propagate the posterior distribution. This is done by resampling with replacement (the bootstrap step) from the present set of samples. The probability of resampling a certain particle is equal to its importance weight. This means that particles with a small weight will most likely not be resampled. Such particles are “killed” in the resampling step, while particles with a large weight probably will be resampled several times, and thus new particles will be “born” in areas where it is likely to find the true position. The recursion is ended with a time update or prediction step, where every particle is translated according to the state transition equation. The SIR procedure is illustrated in Figure 4.2.
Figure 4.2. An example of how the SIR procedure is applied to a one-dimensional problem. The particles are weighted according to the importance distribution $p(y|x)$. Particles with large weight are likely to be resampled several times, while those with small weight are unlikely to be resampled at all. After the resampling step all particles are assigned equal weights. Each new particle is then predicted according to the state transition equation.

The most computationally expensive step in the SIR algorithm is the resampling procedure. In general it is not necessary to resample in every recursion of the algorithm. Instead the weights could be updated recursively by the algorithm, and the resampling step could be performed when the level of degeneracy of the weights is too high. Another advantage of this procedure is that the algorithm becomes less sensitive to outliers. This modification to the SIR algorithm, usually referred to as the Sequential Importance Sampling (SIS) algorithm \cite{1, 4}, is summarized in Algorithm 3.

**Algorithm 3 (Sequential Importance Sampling (SIS))**

1. Set $t = 0$ and sample $N$ times from $p(x_0)$ to generate the set $\{x_0^i\}_{i=1}^N$ and set $w_i^0 = \frac{1}{N}, \; i = 1, \ldots, N$

2. Update the normalized weights $w_i^t = \frac{p_e(y_t - h(x_i^t))w_i^{t-1}}{\sum_{j=1}^N p_e(y_t - h(x_j^t))w_j^{t-1}}$.

3. Calculate position estimate $\hat{x}_t^{\text{MMS}} = \sum_{i=1}^N w_i^t x_i^t$.

4. At every $k$:th iteration, generate a new set $\{x_0^{i*}\}_{i=1}^N$ by sampling with replacement $N$ times from the old set. The probability for resampling particle $x_i^t$ should be equal to $w_i^t$. Reset the weights $w_i^t = \frac{1}{N}, \; i = 1, \ldots, N$. 
5. Generate $v_i^t \sim p_v(v)$, $i = 1, \ldots, N$
and predict each particle $x_{i+1}^t = x_i^t + u_t + v_i^t$, $i = 1, \ldots, N$

6. Set $t = t + 1$ and iterate from 2.

Instead of inserting the bootstrap step on regular basis (every k:th iteration) one could do this only when a significant degeneracy of the weights is observed. A measure for determining the level of degeneracy is the effective sample size [1, 4] which can be estimated by

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{j=1}^{N} w_j^2}$$  \hspace{1cm} (4.22)

If all the particles have the same weight this expression yields the true sample size ($N$). When the algorithm starts to degenerate the variance of the weights increases and this is reflected in the denominator of (4.22). The bootstrap step is applied when $\hat{N}_{\text{eff}}$ falls below a certain threshold, $N_{\text{thres}}$, defined by the user.
5 Vehicle Positioning

The filters presented in the previous sections provide approximations to the optimal Bayesian solution. The next step is to implement and evaluate these filters when applied to the vehicle positioning problem.

When initializing the filters it is assumed that some knowledge about the initial position is available, i.e. one knows that the true initial position is within some limited region of the map. This information can be obtained e.g. by restoring a saved position or by using a single position fix from an external system like the cellular radio network. In either case the accuracy of this initial estimate is allowed to be rather poor.

Furthermore, the true orientation of the vehicle is assumed to be known. This means that the input from the dead-reckoning system can be treated as a relative displacement vector in the coordinate system of the map, see (3.4).

5.1 Implementation

5.1.1 The Point-Mass Filter

The PMF has been implemented in the MATLAB™ [10] environment, using a sparse matrix representation, i.e. only non-zero grid points and their indices are stored and operated on. The implementation used is almost identical with the one suggested in [1].

The performance of the filter depends on several factors. Since it is desirable to achieve both low approximation errors and low computational load, it will be necessary to make a trade-off. As discussed in Section 4.3.2 the density of the grid is a tuning parameter for this purpose. Other aspects that have to be considered are for example:

- The choice of prior distribution.
- Should the grid be adaptive with time? How should it be adapted? Is there a significant performance gain?
- The choice of noise distributions and variance.
- The memory requirements, i.e. how large support can the grid have and how dense can it be?
- Robustness against e.g. measurement errors and sensor drifts.
5.1.2 The Particle Filter

Several versions of the particle filter were implemented in MATLAB\textsuperscript{TM} including some modifications discussed further in Section 5.2.3, though the basic algorithm was the standard SIR algorithm.

An interesting implementational issue is the number of particles used in the algorithm. Either one can choose to have a fixed number throughout the whole run, or to utilize an adaptive sample size. The latter choice can be justified by the fact that a large number of particles is in general required initially, but when the algorithm has converged a smaller number will suffice. In [3] a lower bound for the number of particles required to assure a certain quality of the approximation is derived.

Even though there is a possibility to decrease the number of particles during the execution of the algorithm, it is not obvious that one should do that. If the filter is to be implemented in a real-time environment it is often better to have a fixed sample size for scheduling and memory allocation reasons.

5.2 Performance Evaluation

The PMF and the particle filter have been simulated on both fictitious and real measurements.

Preliminary simulations were performed on a simple, imaginary map using simulated measurement data. The PMF implementation seemed to work very well with a fixed grid resolution. The particle filter on the other hand showed a rather disappointing performance. The frequency of filter divergence was far too high; in some cases over 20%. Even after convergence the filter could suddenly loose the track of the vehicle, resulting in a divergence. Figure 5.1 shows how typical divergence of the particle filter might look like (in this case when applied to real measurement data).

Due to these discouraging results, effort was put into how to improve the particle filter implementation. In Section 5.2.2 these issues are discussed in detail.

5.2.1 Measurement Data Acquisition

The measurement data used for the simulations were collected from test drives in suburban and dense residential areas. The sensor equipment included ABS signals (wheel speed) available through the CAN bus, a rate-gyro
Figure 5.1. The estimation error of 100 different simulations on the same sequence of measurement data, using the particle filter (sIR with a fixed number of particles, N=2000). In this case 22 of the runs did not converge, which is far too many.

oriented along the vertical axis and a three-dimensional accelerometer cluster. However, the accelerometer signals were not used here due to their sensitivity to road inclination. A GPS-receiver was used as absolute position reference along the test path. The measurements were collected by a hardware platform mounted in the test vehicle, a Volvo V40, and downloaded to a laptop.

In this report, two different data sets were used for evaluation purposes. The first one, in the sequel referred to as S1, is a 560 seconds long sequence collected in a suburban district (Figure 5.2(a)). The second one was collected in a denser residential area with several crossings (Figure 5.2(b)). This sequence will be referred to as S2.

The signals were sampled with 20 Hz, but in the simulations they were resampled with 2 Hz. The gyro offset was removed off-line since the main objective here is not to test the robustness of the filters.

To provide the filters with maps of the test drive areas, a simple digitizing GUI was implemented in MATLAB™. The road network was then encoded from high resolution (1 m/pixel) aerial photos.
5.2.2 Particle Clustering

Simulations of the particle filter show that it has a tendency of occasionally losing the track. The frequency of lost tracks increases when the initial number of particles is small. This is partly caused by a phenomenon here referred to as particle clustering, i.e. when the initial distribution does not spread the particles well on the road. This introduces clusters of particles with gaps between them (see Figure 5.3). These gaps can cause the filter to fail detecting a turn, resulting in a lost track. This indicates that the particle filter is very sensitive to the initial particle distribution. The frequency of lost tracks is an important measure of the filtering performance.

The particle clustering problem is amplified by the fact that the interesting part of the state-space, i.e. the road network, occupies a very small fraction of the complete state-space and that the likelihood (i.e. $p_{e_t}(\cdot)$) has a very narrow support. The consequence of this is that only a few particles will have a significant non-zero weight and are likely to be resampled.

One idea how to reduce the particle clustering is to use the knowledge about the car being somewhere on the road in a more efficient way. This can be
done e.g. by only sampling particles on the road initially, which means that the prior distribution density is basically zero everywhere outside the road. The problem is that there is no way of sampling directly from this density, so one have to consider using e.g. a Monte Carlo sampling method [1, 4], which can be rather inefficient, or a semi-deterministic sampling method. Both alternatives have been tested, but the latter was used for the results presented in this report.

Another idea is to use the process noise to smear the particles in the direction of the road while running the algorithm. This can be done by introducing some correlation between the process noise components, so that the density get more support in the direction of travel than perpendicular to it. This will increase the uncertainty about the location of the car along the road when traveling in the same direction for a longer period of time, but hopefully it will also reduce the particle clustering. An alternative to using correlated process noise is to introduce the velocity of the vehicle as a new state in the model, but that will increase the complexity of the problem.

There exist several articles that suggests improvements to the original SIR algorithm. In [6] and [8] a method called jittering or roughening is presented. The basic idea is to prevent that the filter degenerates to only a few distinct
sample values (i.e. to prevent clusters) by adding some noise to each sample generated in the time update step. This is particularly efficient when the process noise is low.

Another method is prior editing [6, 8], which uses an acceptance test after the sampling step. This means that only samples with a reasonable chance of being resampled is accepted. To achieve this one needs to have access to the next measurement, which will delay the estimate one sample.

5.2.3 Improvements of the Particle Filter Implementation

A solution to the particle clustering problem discussed in the previous section is to introduce some correlation in the noise distribution, which is used in the state evolution equation (3.1), $v_t$. The aim is to spread more particles along the road rather than away from it. If one assume that the absolute heading of the vehicle, i.e. the Euler yaw angle $\psi_t$, is known and Gaussian noise is used, the noise vector expressed in the vehicle-body frame,

$$v_t' = \begin{bmatrix} v_{lat}^t \\ v_{long}^t \end{bmatrix}$$

(5.1)

can be transformed to the global frame

$$v_t = \begin{bmatrix} \cos \psi_t & -\sin \psi_t \\ \sin \psi_t & \cos \psi_t \end{bmatrix} v_t'$$

(5.2)

By choosing the variance of $v_{lat}^t$ and $v_{long}^t$, the desired particle distribution can be tuned.

Yet another modification to improve the particle convergence is to introduce a prior editing step in the time update of the particle filter. The prior editing procedure involves an acceptance test after the bootstrap step as described by Algorithm 4. Note that this modifies item 4 and 5 in Algorithm 2 and 3.

Algorithm 4 (Prior Editing)

1. Set $i=1$.

2. Generate a sample $x_{t+1}^i$ in the bootstrap step (i.e. $x_{t+1}^i$ is assumed to be an i.i.d. sample from $p(x_t|y_t)$) and predict $x_{t+1}^i = x_{t+1}^i + u_t + v_t$, $v_t \sim p_\alpha(v)$.

3. Calculate the residual $e_{t+1}^i = y_{t+1} - h(x_{t+1}^i)$
4. Accept this sample and set $i = i + 1$ if $|e_t| < k_{pe} \sqrt{\text{Var}(e_t)}$, where $k_{pe}$ is a user defined parameter.

5. Iterate from item 2 while $i \leq N$

Note that the prior editing requires the next measurement $y_{t+1}$ and that this would delay the filter one sample. However, since the measurement in this case is rather fictitious and considered to be known for all time instances (see Section 3.2), no time lag is introduced in the algorithm.

The relative number of rejected samples in the prior editing step is a measure of how well the particles are spread onto the road. When the vehicle is making a sharp turn the prior editing ratio will increase slightly, because most particles are spread in the direction of the tangent to the trajectory.

### 5.2.4 Evaluation Using Real Measurements

The PMF was simulated on data from $S1$, using a fixed grid resolution, $\delta = 5$ m, and $N = 40000$ grid points initially. The position error compared to the GPS is shown in Figure 5.4. Also included in the figure is the square root of the scalar mean square error, which is defined by

$$
E\left(\|x_t - \hat{x}_t\|_2^2\right) = \text{tr} \ P_t,
$$

which can be thought of as the standard deviation of the estimation error [9]. The “true” position, $x_t$, was obtained from dead reckoning using offline corrected sensor signals, i.e. the sensor errors were tuned so that the trajectory of the vehicle matched the current driving path on the map.

The increase in the error after about 280 seconds is caused by off-road driving. The large error in the GPS-position in some regions arises when driving under foliage. The position estimate from the PMF is better than the GPS-position almost everywhere during the simulation.

A simulation on the same sequence, performed with the particle filter using prior editing, is shown in Figure 5.5. The relative number of rejected samples in the prior editing step is shown in Figure 5.6.
Figure 5.4. Simulation results of the PMF using data from sequence S1.

Figure 5.5. Simulation results of the particle filter using data from sequence S1.
Figure 5.6. The prior editing ratio for the simulation in Figure 5.5.
The particle filter yields approximately the same error level as the PMF. An interesting feature is that the sharp turns made during the test drive is shown as peaks in the prior editing ratio.

If the filter loses track of the vehicle position this will probably result in that a lot of particles are spread off the road, since the estimated trajectory no longer matches the heading of the vehicle. This makes it possible to use the prior editing ratio to detect lost tracks. Figure 5.7 shows how this might look like. After 380 seconds the track is lost, which is indicated by the increased error. This is followed by a heavy increase in the prior editing ratio (> 1000%), which causes the algorithm to make a soft reset of the filter. After about 20 seconds the estimate is back on track. The reaction time of the filter is deliberately chosen to be slower than necessary for the sake of clarity.

![Figure 5.7. Simulation results of the particle filter when the track is lost.](image)
5.2.5 Monte Carlo Simulations

To evaluate the performance of the particle filter, extensive simulations are required because of its stochastic nature. However, by applying the algorithm several times to a single data set, each time with a different noise realization, a similar effect can be achieved. This kind of simulation is called Monte Carlo simulation [9].

Suppose that $M$ Monte Carlo runs are generated from one set of measurement data, i.e. $M$ different sequences of position estimates are available. Then it is possible to calculate the root mean square error (RMSE) from these simulations

$$\text{RMSE}(t) = \left( \frac{1}{M} \sum_{j=1}^{M} \left\| x_t - \hat{x}_t^{(j)} \right\|_2^2 \right)^{1/2}$$

which is a measure of the length (magnitude) of the estimation error, i.e. the distance between the estimated and the true position. A common navigation performance parameter is the median of this error, the circular error probable (CEP).

Results from 100 Monte Carlo simulations of the particle filter (sIR with correlated process noise and prior editing) using data from test drive SI are shown in Figure 5.8. The number of particles used in each recursion was chosen adaptively between 200 and 1000, based on the uncertainty in the estimate. The CEP was calculated for the sequence starting after a burn-in time, i.e. in this case after about 150 samples, and compared to the corresponding value obtained from the GPS. Since the GPS-receiver has trouble with foliage at the end of the sequence, the particle filter yields significantly better performance.

It should be noted that the GPS receiver has an initial transient as well, although not shown in the results. For the particular receiver model used here the initialization process takes approximately 45 seconds.

The particle filter can be configured in several ways, using the ideas discussed in Section 5.2.3. However, this report will only present a selection of these. Two different configurations have been simulated on measurement data from the driving scenarios depicted in Figure 5.2. The first one uses the

---

3This can be motivated by Figure 5.1; the true performance of the filter is not revealed by a single simulation.

4The transient of the estimation error.
sis algorithm with an adaptive number of particles and conditional resampling (i.e. to use the effective sample size $N_{\text{eff}}$ to decide when to resample). Furthermore prior editing was used to ensure high performance when the particle number drops. The second configuration uses the sis algorithm with a fixed number of particles and conditional resampling.

The results from 100 Monte Carlo simulations is shown in Figure 5.9. From this it can be noted that the filter using a large fixed number of particles has a slightly faster convergence and lower error. However, the number of floating point operations required for each run is about three times higher. The test drive from the residential area, $S2$, yields a slower convergence in a characteristic staircase fashion, due to the larger ambiguity in the road network. Each “step” in the staircase is caused by a road change.

The area where particles is sampled initially, i.e. the area where the vehicle is likely to be when the filter is started, was chosen to be $1000 \times 1000$ m in $S1$ and $500 \times 500$ m in $S2$. 

**Figure 5.8.** RMSE for the particle filter based on 100 Monte Carlo simulations. Also shown is the RMSE plus one standard deviation and the maximum error. The GPS error is included as a reference. Note that the x-axis label is sample number.
5.2 Performance Evaluation

(a) Test drive $S_1$.

(b) Test drive $S_2$.

Figure 5.9. Two different particle filter configurations applied to driving scenario $S_1$ and $S_2$ respectively. The result was obtained from 100 Monte Carlo simulations on each sequence.
5.3 Discussion

Both the PMF and the particle filter solve the vehicle positioning problem with satisfying performance. The particle filter occasionally diverges, but the frequency of these events can be reduced by some improvements, like e.g. correlated process noise, semi-deterministic prior distribution and prior editing. The latter of these modifications also provides a method of detecting filter divergence, so that the algorithm can be re-initialized.

A major drawback of the filters discussed so far is that the heading angle must be provided by an external source. Hence, the robustness against angular errors are low. If e.g. there is a drift in the heading angle, the algorithms will most certainly diverge.

The simulations discussed above assume that the initial knowledge about the position is very uncertain. In general, when the filters are started, it is possible to retrieve a stored position and thus the initial estimate is much better than the ones considered here.

Since the map-aided approach heavily relies on the accuracy of the map, an important question is how the map should be modeled to yield optimal performance of the algorithms. Although this thesis does not focus on these aspects, one should keep in mind that map modeling errors have a great influence on the performance of this method.
6 Heading Angle Estimation

The algorithms presented in the previous section solve the two-dimensional positioning problem, assuming that the heading angle is known. A natural extension would be to explore the possibilities to remove that assumption in order to get an independent absolute position estimator.

The extended estimation problem includes estimation of the heading angle and thus the state-space model has to be modified.

6.1 Model Extension

The two-dimensional state space model assumes that the absolute heading of the vehicle is known and thus this information has to be provided from an external source. However, since the map contains a lot more information than used in the current model, there is a possibility to estimate the heading angle as well. To be able to do this, an extension of the model is needed. Let the heading angle be denoted by $\psi_t$ and introduce it as a new state

\[
\begin{align*}
    x_{t+1} &= x_t + f(\psi_t, \Delta r) + v^x_t \\
    \psi_{t+1} &= \psi_t + \Delta \psi + v^\psi_t \\
    y^1_t &= h(x_t) + e^1_t \\
    y^2_t &= \phi(x_t, \psi_t) + e^2_t
\end{align*}
\]

where the input from the relative sensors are split up into a linear and an angular displacement, i.e.

\[
\begin{align*}
    \Delta r &= T_s v_x \\
    \Delta \psi &= T_s v^\psi
\end{align*}
\]

The nonlinear function $f(\psi_t, \Delta r)$ is given by

\[
f(\psi_t, \Delta r) = \begin{bmatrix} -\Delta r \sin \psi_t \\ \Delta r \cos \psi_t \end{bmatrix}
\]

The state evolution is no longer linear, but consists of a nonlinear update of $x_t$ and a linear update of $\psi_t$. A new measurement, $y^2_t$, is added as well. It consists of the nonlinear function, $\phi(x_t, \psi_t)$, that yields the orientation of
the closest road minus the current heading angle, plus some measurement noise.

The Bayesian solution to the three-dimensional problem is of course somewhat different compared to the one presented in Section 4.2. However, the derivation is almost identical (see Appendix B) and thus the solution is just summarized here as

\[
p(x_t|y_t) = \frac{1}{c_t} p_e(y_t^1 - h(x_t), y_t^2 - \phi_t(x_t) + \psi_t) p(x_t|y_{t-1})
\]

(6.5)

\[
p(\bar{x}_{t+1}|y_t) = \int_{\mathbb{R}^3} p_v(x_{t+1} - x_t - f(\psi_t, \Delta r), \psi_{t+1} - \psi_t - \Delta \psi) p(\bar{x}_t|y_t) dx_t,
\]

(6.6)

where

\[
\bar{x}_t = \begin{pmatrix} x_t \\ \psi_t \end{pmatrix}, \quad \bar{y}_t = \{y_t^1, y_t^2\}_{i=0}^t.
\]

Note that \(p_v\) is a three-dimensional PDF, while \(p_e\) is two-dimensional.

### 6.2 Implementation

Since the extended estimation problem introduces an extra state in the model, the complexity of the problem increases as well. Therefore, due to its special properties concerning high-dimensional problems, the particle filter is chosen for this purpose. Although the performance of Monte Carlo methods does not explicitly depend on the dimension of the problem, the number of particles usually has to be increased, especially during the initial phase. To relieve the computational load due to the increased sample size, the SIS algorithm is utilized here. It should be noted that the original SIR algorithm is a special case of SIS when the bootstrap step is included in every iteration.

### 6.3 Performance Evaluation

The filter was simulated on sensor data collected in both suburban areas and dense residential areas. No initial knowledge about the orientation is assumed, i.e., the initial heading angle is uniformly distributed between \(0^\circ\) and \(360^\circ\).
6.3 Performance Evaluation

6.3.1 Monte Carlo Simulations

To evaluate the performance of the three-dimensional particle filter, Monte Carlo simulations were performed on the same data sets that were used in the two-dimensional case ($S1$ and $S2$), see Figure 6.1. The wheel speed signals were used without modifications, but the gyro signal was preprocessed in order to reduce the effects of the offset error (analogous to the two-dimensional case, see Section 5.2.1). The bound for the initial position was $1000 \times 1000 \text{ m}$ in $S1$ and $500 \times 500 \text{ m}$ in $S2$. In Table 6.1 the simulation results are summarized.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Burn-In Time [s]</th>
<th>Particle Filter CEP [m]</th>
<th>GPS CEP [m]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S1$</td>
<td>85</td>
<td>7.7</td>
<td>19.9</td>
</tr>
<tr>
<td>$S2$</td>
<td>110</td>
<td>9.4</td>
<td>12.4</td>
</tr>
</tbody>
</table>

Table 6.1. Simulation results from 100 Monte Carlo simulations of the three-dimensional particle filter. The CEP was calculated for the sequence after a specified burn-in time.

The simulations in the residential area have a slower convergence due to the facts discussed for the two-dimensional case earlier. Also, the standard deviation as well as the maximum error for the Monte Carlo simulations are larger here. Furthermore, six of the runs diverged, indicating that this is a particularly tough scenario for the algorithm.

6.3.2 Robustness

The simulation results presented in Section 5.2.5 and 6.3.1 were accomplished with preprocessed sensor signals, i.e. some of the errors was removed by off-line operations before passed on to the particle filter. However, it would be interesting to analyze also the robustness of the filter, to see if it is possible to use erroneous sensor data as input.

Another interesting issue is whether it is feasible to implement the system without utilizing the rate-gyro, which can be achieved by computing the yaw rate signal from the wheel sensor signals, as in (2.6). The reason why one would like to do this is that rate-gyros are non-standard components in today’s cars, while ABS-sensors are more common. There is, however, an increasing use of inertial sensors in automotive applications, like e.g. anti-skid systems, anti-rollover systems or intelligent airbags.

The system was simulated using raw sensor signals in two combinations:
Figure 6.1. 100 Monte Carlo simulations of the three-dimensional particle filter applied to driving scenario $S1$ and $S2$ respectively.
• Longitudinal velocity calculated from the ABS-sensors and yaw rate provided by the gyro.

• Both longitudinal velocity and yaw rate calculated from the ABS-sensors.

Since the gyro signal is associated with an offset error, the heading angle will suffer from linear drift. With the current error model this error can not be completely eliminated. Hence, the estimated heading angle will include a small offset (Figure 6.2). One way to deal with this is to estimate the drift, i.e. to include the drift term as a new state in the model. That will of course increase the dimension of the problem, but in this case there are ways to simplify the computations [11].

![Figure 6.2. Estimated heading angle when the gyro signal was used (sequence S1). Note that there is a small offset after burn-in.](image)

If one instead uses the ABS-sensor signals to calculate the yaw rate, the drift is not as significant and thus the estimate is less biased (Figure 6.3). However, this is only true when the rear wheel radii are approximately the same, which may not be the case if the tire pressure difference is significant. This difference between the two sensor configurations has no significant
influence on the accuracy of the position estimate, but the frequency of divergence is higher when using the rate gyro in this case.

![Graph](image)

**Figure 6.3.** Estimated heading angle when the wheel-speed signals were used to compute the yaw rate (sequence $S1$). The offset is smaller compared to Figure 6.2.

It is of course not recommendable to use the sensor signals without any preprocessing. More preferable is to remove some of the errors by using sensor fusion on the different sensors before applying the positioning filters [9]. A very simple operation to reduce the offset error of the gyro is to calibrate the signal when the vehicle comes to a standstill (i.e. when the yaw rate is zero).

### 6.4 Discussion

In the extended model (6.1), an extra measurement, $y^2_t$, is used. It would be interesting to find out whether that is necessary, especially when the information obtained from the observation is not very useful when the vehicle is in the middle of a crossroad. Tests without the additional observation indicate a slower convergence, but it seems to be feasible to use this simplification whenever a sharp turn is detected.

One can of course ask if it is not possible to use the PMF on the extended
problem, when it seemed to work so well for the two-dimensional case. Theoretically, the corresponding procedure is to define a three-dimensional grid and apply the point-mass approximation. This means that a two-dimensional grid has to be used for each discretized value of the extended state variable (the heading angle). As can be imagined this entails a heavy computational burden and large memory requirements.
7 Conclusions

In this thesis we have studied map-aided vehicle positioning using Bayesian estimation. Sequential Monte Carlo methods as well as a numerical integration method (the point-mass filter) have been successfully applied to the basic two-dimensional positioning problem (assuming that the heading angle is known).

The extended problem, in which the heading angle is estimated as well as the position, has been solved using a particle filter with satisfying results.

The conclusion of this work is that it is possible to use the concept of map-aided positioning, as defined in this report, in a vehicle positioning application. However, it needs a lot more research to be fully functional.

8 Future Work

The natural aim for the future work on this project is to implement the positioning algorithm in a real-time environment for online evaluation. However, there are several other issues that should be further considered, e.g. the map representation. In this thesis the maps have been digitized manually by the author. This procedure is of course not feasible in the long run and therefore it is necessary to find a supplier of the desired map information.

Another aspect that should be investigated further is the robustness of the filters. This requires extensive simulations and a lot of effort put on tuning. If the algorithm is to be implemented online, research on the computational requirements for different filter configurations has to be performed.

There are several possibilities for how the positioning algorithm could be used in a navigation system. One example is to use map-aided positioning together with a cellular radio positioning system. Another alternative is to add a magnetic compass to this configuration, and use the map-aided algorithm on the two-dimensional problem only. Yet another possibility is to use the map-aided positioning module as a complementary system to a GPS-based navigator, in order to increase the overall performance.

Apart from navigation applications, the map-aided positioning system can be used for several other purposes. For example the heading angle estimation yields an improved yaw rate estimate, which is of great importance in anti-skid systems and many other automotive applications. The system can also be used for on-line sensor calibration, improved absolute velocity estimation and tire pressure indicators.
The ideas of map-aided positioning are of course not restricted only to road vehicles (cars, busses, trucks, motorcycles), but can also be applied to other areas where map information is available, such as e.g. trains and autonomous industrial robots. The special requirements and problems associated with this remains to be sorted out.

To summarize, the guideline in this work has been to use low cost sensor equipment and digitally stored maps to provide high performance positioning for road vehicles. The nonlinear filtering problem that arises can be efficiently tackled by Bayesian estimation methods, as indicated by the results presented in this report. Thus the first stepping-stone has been completed and the next phase of this project can be initialized.

References


Appendix A: Bayes’ Rule

Let $x$ and $y$ denote two stochastic variables with probability density functions $p(x)$ and $p(y)$ respectively. The joint density can thus be written as

$$p(x, y) = p(x|y)p(y), \quad (A.1)$$

or equivalently as

$$p(x, y) = p(y|x)p(x). \quad (A.2)$$

Combining (A.1) and (A.2), and solving for the conditional probability $p(x|y)$ yields

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}. \quad (A.3)$$

This expression is usually referred to as Bayes’ rule or Bayes’ theorem. The denominator $p(y)$ can be obtained by marginalizing out $x$ in (A.2)

$$p(y) = \int_{\mathbb{R}^n} p(x, y) \, dx = \int_{\mathbb{R}^n} p(y|x)p(x) \, dx. \quad (A.4)$$
Appendix B: Bayesian Solution to the 3D Problem

Let the extended state and measurement vectors be summarized as
\[
\begin{align*}
\bar{x}_t &= \begin{pmatrix} x_t \\ \psi_t \end{pmatrix} \\
\bar{y}_t &= \begin{pmatrix} h(x_t) \\ \phi(x_t, \psi_t) \end{pmatrix} + \begin{pmatrix} e^1_t \\ e^2_t \end{pmatrix}
\end{align*}
\]

and let \( \mathbf{Y}_t = \{ \bar{y}_t \}_{t=0}^t \) denote the available set of measurements at time \( t \). If \( p(\bar{x}_t|\mathbf{Y}_{t-1}) \) is assumed known, the update of the PDF with the new measurement, \( \bar{y}_t \), is obtained by using Bayes’ rule

\[
p(\bar{x}_t|\mathbf{Y}_t) = \frac{p(\bar{y}_t|\bar{x}_t, \mathbf{Y}_{t-1})p(\bar{x}_t|\mathbf{Y}_{t-1})}{p(\bar{y}_t|\mathbf{Y}_{t-1})} = \frac{p(\bar{y}_t|\bar{x}_t)p(\bar{x}_t|\mathbf{Y}_{t-1})}{p(\bar{y}_t|\mathbf{Y}_{t-1})}. \tag{B.1}
\]

The time update is obtained by observing that

\[
p(\bar{x}_{t+1}|\bar{x}_t, \mathbf{Y}_t) = p(\bar{x}_{t+1}|\bar{x}_t, \mathbf{Y}_t)p(\bar{x}_t|\mathbf{Y}_t) = p(\bar{x}_{t+1}|\bar{x}_t)p(\bar{x}_t|\mathbf{Y}_t). \tag{B.2}
\]

Marginalizing with respect to \( \bar{x}_t \) yields

\[
p(\bar{x}_{t+1}|\mathbf{Y}_t) = \int_{\mathbb{R}^n} p(\bar{x}_{t+1}|\bar{x}_t)p(\bar{x}_t|\mathbf{Y}_t) d\bar{x}_t. \tag{B.3}
\]

Using the extended state-space model according to (6.1), the following can be noticed

\[
\begin{align*}
p(\bar{x}_{t+1}|\bar{x}_t) &= p\psi_t(x_{t+1} - x_t - f(\psi_t, \Delta r), \psi_{t+1} - \psi_t - \Delta \psi) \\
p(\bar{y}_t|\bar{x}_t) &= p\psi_t(y^1_t - h(x_t), y^2_t - \phi(x_t) + \psi_t).
\end{align*}
\]

The Bayesian solution to this problem is thus given by the following expressions

\[
\begin{align*}
p(\bar{x}_t|\mathbf{Y}_t) &= \frac{1}{c^t} p\psi_t(y^1_t - h(x_t), y^2_t - \phi(x_t) + \psi_t)p(\bar{x}_t|\mathbf{Y}_{t-1}) \tag{B.4} \\
p(\bar{x}_{t+1}|\mathbf{Y}_t) &= \int_{\mathbb{R}^3} p\psi_t(x_{t+1} - x_t - f(\psi_t, \Delta r), \psi_{t+1} - \psi_t - \Delta \psi)p(\bar{x}_t|\mathbf{Y}_t) dx_t \tag{B.5}
\end{align*}
\]
where $c_t$ is a normalizing constant. From these two equations it is possible to calculate the a posteriori probability density function, $p(\tilde{x}_t|\bar{Y}_t)$, recursively given the measurements $\bar{Y}_t$. The recursion is initialized with $p(\tilde{x}_0) = p(\tilde{x}_0|\bar{Y}_{-1})$. 