Master Thesis

Improved Positioning of City Buses

Centre for Mathematical Sciences
Mathematical Statistics

Lund University, Sweden

Author:
Nils Bengtsson

Supervisor:
Andreas Jakosson

January 30, 2014

Abstract

In order to build an efficient transportation system of the future, public transport needs to be an attractive alternative offering a high quality travel experience. Future digital information systems for public buses will enrich the travelers experience by delivering relevant content - accurately on the right time and place. In this master thesis, design and implementation of an improved bus positioning system is presented with the objective to meet these demands. The system is based on a bank of particle filters used to track a non-linear, non Gaussian state model of the bus. The available input signals are GPS, wheel speed, and door opening sensors. The system utilizes a GIS database containing the street network and a local traffic system database with spatial and temporal information of the buses. The implementation is done in python, powered by the GeoDjango framework and the PostgreSQL database, and has been tested in buses containing a commercial digital information system. The system improves the positioning during GPS failure and increases reliability by combining all disposable information and presenting an accurate and redundant estimate.

Populärvetenskaplig sammanfattning

Inom några år kommer många av våra stads- och regionbussar i Europa vara utrustade med skärmbaserade informationssystem. Skärmarna öppnar nya möjligheter för att kommunicera information, underhållning och reklam till resenärerna vilket många aktörer hoppas bidra till en förbättrad resandeupplevelse. För att visa rätt information vid rätt tid och plats krävs ett pålitligt och noggrant positioneringssystem i bussarna.

Idag används i hög utsträckning GPS för att positionera bussar. Det ger en noggrann skattning som fungerar över hela världen. Tyvärr kan en GPS störas av omgivningen och tappar då i noggrannhet eller i värsta fall förlora signalen helt. I stadsmiljö med många höga hus, viadukter och tunnlar är denna effekt extra tydlig. För att förbättra positionen vid mindre störningar kan enkla så kallade Kalmanfilter användas som även utnyttjar bussens hastighet och rattutslag för att skatta positionen. Dessa fungerar väl om bussen till exempel kör genom en tunnel. Vid svårare situationer med större och längre störningar räcker inte dessa metoder till för att bibehålla en robust skattning. I detta examensarbete har ett mer omfattande positioneringssystem utvecklats och implementerats vilket tillgodogör alla tillgänglig information för att tillhandahålla en robust och noggrann position.

Det finns två typer information som vägs samman i system, realtidsignaler och på förhand känd statisk information. Realtidssignalerna är GPS, hastighet, dörröppningssignal samt bussturens nummer. Den statiska informationen består av en vägkarta, bussrutternas stäckning, hållplatsernas position samt en komplett busstidtabell.

För att väga samman all information används en modellbaserad teknik där modellen består av en rörelsemodell av bussen tillsammans med vägkartan. På grund av vägnätets komplicerade uppbyggnad där varje korsning introducerar flera alternativ blir modellen svår att hantera med de enklare Kalmanfiltren. Vid t.ex. korsningar måste systemet välja alla alternativen sedan efterhand som bussen kör och mer information tillkommer avgöra vilken väg bussen valde. För detta ändamål har ett partikelfilter används.

Ett partikelfilter utnyttjar den ökade tillgången av beräkningskraft för att göra svåra skattningar. Varje partikel är en individuell skattning som i detta fall är en modellerad version av bussen. Genom att köra ett hundratal partiklar parallellt kan de t.ex. slumpmässigt göra olika vägval i en korsning och på det sättet testa alla alternativ. Partiklarna stäms sedan av mot de inkommande mätningarna från GPS, hastighet och eventuellt hållplatspositioner ifall bussen har stannat. De partiklar som bäst stämmer överens med mätningarna vägs samman och formar bussens position.

Genom att analysera inkommande signaler kan systemet automatisk växla mellan lämpliga partikelfilter för situationen. Systemet har visat sig robust mot både störningar i GPS och mot förlorade information om aktuell busstur. Det har utvecklats och testats tillsammans med aktörer i branschen och är redo för att tas i drift i kommersiell verksamhet.

Contents

Abstract i
Popularvetenskaplig sammanfattning ii
Nomenclature iii

1 Introduction **1**
1.1 Motive 1
1.2 Previous work 1
1.3 Project Objective 2

2 Geographical systems **3**
2.1 Spherical coordinates 3
2.2 Projection 3
2.2.1 Mercator projection 4
2.2.2 Google projection 4
2.2.3 UTM 5
2.2.4 SWEREF 99 TM 5

3 Mathematical Theory **7**
3.1 Models 7
3.1.1 State space models 7
3.1.2 External input and measurement output 8
3.1.3 Continuous and discrete models 9
3.1.4 Stochastic system 9
3.2 Particle Filters 10
3.2.1 Bayesian filtering 10
3.2.2 Numerical approximation 11
3.2.3 General Particle Filter 12
3.2.4 Estimation 15
3.2.5 Proposal Distribution 15
3.2.6 Resampling 17

4 Signals and available information **18**
4.1 GPS 18
4.2 Internal signals 19
4.3 Open Street Maps 19
4.4 Waylink Maps 20
4.5 Local Bus Transport system 21
4.5.1 Line, route, and tour 21
4.5.2 Stations, platform and stop 21
4.5.3 REBUS 22
4.6 Route building algorithm 22
4.6.1 Log data 22
4.6.2 Administration tool 22

4.6.3 The algorithm 22

5 Bus Positioning System **24**
5.1 System Overview 24
5.2 The model 25
5.2.1 Bus model 25
5.2.2 Transport system database 26
5.3 BPS states 26
5.4 Filter picker 27
5.5 The Filters 28
5.5.1 The particle filter algorithm 28
5.5.2 Wheel Speed 28
5.5.3 GPS position 30
5.5.4 GPS direction 31
5.5.5 Platform position 31
5.6 Implementation 32

6 Results **33**
6.1 Test environment 33
6.2 External testpoints 33
6.3 GPS performance 34
6.4 The testes 35
6.4.1 Full information 36
6.4.2 Without tour information 36
6.4.3 Without GPS 36
6.4.4 Distorted GPS postion 36

7 Conclusion **39**

8 Topics of future research **40**

Acknowledgments **41**

Bibliography **42**

1 Introduction

1.1 Motive

Buses in public transport are expected to in the next couple of years, to a large extent, be equipped with new display based digital information systems. These new systems will provide position based, real-time content to the passenger. Hence, the new information channel will depend on a robust and accurate positioning system. The GPS typically provides accurate enough positioning data for most applications in an information system. However, the GPS alone cannot guarantee a robust estimate. Disturbance caused by lost satellite connection or reflections in the environment can cause errors in the range of hundreds of meters and in worst case causes complete blackouts. Instead of using GPS, another approach is to assume that the bus will move on the known bus route and use a speedometer to sum up the distance traveled. This implies that the bus starts at a specified point and never go off-route. The requirement in a modern information system calls for a solution where different position signals, together with prior knowledge of traveling route are merged to form a robust and accurate estimate.

1.2 Previous work

The field of moving vehicle positioning by combining GPS with other vehicle information has been a subject of research for a couple of decades [1]. Good results has been presented using various filters on state space models. Both the classic Kalman filter and Extended Kalman filters give good result with moderate computational requirement when combining GPS and internal vehicle information such as odometer and steering sensor signals. To further improve the estimate a map can be used to match the position to the street network [2]. The street network is however a highly non-linear function and it stresses limits of the linear assumption in both the Kalman filter and the Extended Kalman filter. A popular alternative solution is to use a Particle Filter[3, 4], which has the ability to estimate the whole probability density function of a non-linear state space model. This makes it possible to solve multi-hypotheses tracking problems and determine the most probable

estimate.

1.3 Project Objective

The objective of this thesis is to develop a positioning system and test it on a bus equipped with a modern information system. The final product should be implemented in a programing language suitable for a commercial product and ready to be used in a live system.

2 Geographical systems

There are numerous coordinate systems used for geographical purposes, all with special purposes and backgrounds. In this chapter, some of the basic problems are summarized. Some popular coordinate system are introduced together with those relevant for this thesis. The difficulties with representing a spherical shape on a flat surface are also addressed.

2.1 Spherical coordinates

The earth is well approximated with an ellipsoid where the diameter of the equator is slightly larger than the distance between the poles. In a spherical coordinate system, the earth surface is given by longitude, latitude and height above a reference surface.

A spherical coordinate system describing an ellipsoid may be specified using a couple of different variables. The two radii from the poles and equator have to be set, the center point has to be pegged to the earth and the zero longitude has to be fixed to a point on the earth surface. A set of settings describing an ellipsoid is referred to as a *datum*. For several reasons there are many datums in use. Due to the imperfection of the earth surface there exists a variety of different datum that make a best fit in different part of the world. The earth radii measurements have developed over time, old datum become obsolete as new more precise measures have been adopted [5]. General datums exists and are used in applications where a global approximation is needed.

2.2 Projection

Projection is used to represent a spherical surface on a flat surface map by transforming the longitude and latitude coordinates onto a plane. In the process some properties of the map remain the same and some are distorted. Typical properties of a surface are distance, area, shape, and scale. Projections always distort some of the properties while others can be preserved. Many different projections are in use which all serve to preserve some specific properties throughout the transformation [6].

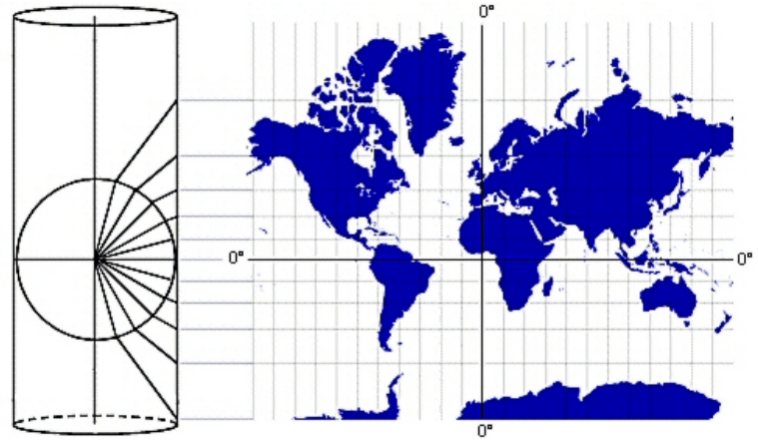

Figure 2.1: The Mercator projection.

2.2.1 Mercator projection

The Mercator Projection was presented in 1569 by Gerardus Mercator (1512 - 1594) and uses a *conformal cylindric projection* along the earth rotational axis. The key feature of a conformal projection is the preservation of the angle between two crossing lines. It also preserves the shapes of small objects across the whole map. A major drawback is the distortion of scale that is zero at the equator and goes to infinity at the poles. This is the reason countries in northern and southern hemisphere appear relatively large on maps using the Mercator projection. Because of the scaling the poles are projected to infinity and maps from the Mercator projection are therefore truncated at a north/south limit. Figure 2.1 illustrates how the latitudes are projected on to a map 'rolled' around the equator. The unfolded map is in display on the right side [5].

Two commonly used variations of the (normal) Mercator projection are the spherical (NMS) and ellipsoidal (NME) projections. They are used to form a conformal projection based on a spherical or ellipsoidal earth datum. The NMS scale equation for x and y are as follow

$$x = R\lambda \tag{2.1a}$$

$$y = R \ln \left[\tan \left(\frac{\pi}{4} + \frac{\phi}{2} \right) \right] \tag{2.1b}$$

where R is the sphere radius and λ and ϕ are the longitude and latitude. In figure 2.2, the y scale factor is plotted to illustrate the accelerating scale distortion.

2.2.2 Google projection

The Google-projection has gained popularity after the success of the Google maps service. It uses the WGS84 datum and spherical Mercator projection, NMS. Since the WGS84 is an ellipsoidal datum, the resulting Google-projection is not a conformal projection e.i., it generates a distorted direction

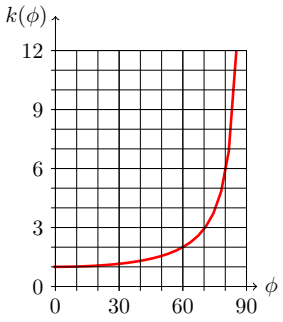

Figure 2.2: The scaling factor of the NMS [5].

scaling. The error in scale is up to 0.7%. Google's use of the spherical projection despite its distorting effect is due to computational advantages [7].

2.2.3 UTM

The Universal Transverse Mercator (UTM) is a projection built out of 60 different projections brought together to form a global map with small distortion in both scaling and shape. The globe is divided into 6°wide zones from 80°south to 84°north. Each zone is projected with a transverse Mercator projection, where the cylinder tangent is not along the equator but rather around the center of the 6 degree zone, as shown in figure 2.3.

Figure 2.3: One of 60 tiles of the UTM projection. Source: www.progonos.com

2.2.4 SWEREF 99 TM

In Sweden, the SWEREF 99 TM projection is the official coordinate system in use. It is the same projection used in UTM zone 33 but extended to cover the whole country [8].

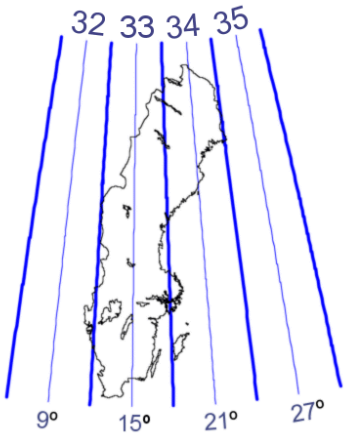

Figure 2.4: The UTM projections covering Sweden. The SWEREF99 projection uses an extension of the UTM 33 projection to cover all of Sweden.

3 Mathematical Theory

In this chapter, the reader is given the mathematical background of the particle filter. The filter is based on a stochastic state space model which is briefly introduced in the first section. In the second section a general particle filter is derived with the Bayesian filter framework. At last some different filter design alternative are discussed.

3.1 Models

The objective of mathematical modeling is to describe a system using mathematical logic. The system may come from a variety of scientific fields spanning from physics, biology, economics to social behaviors. There are various purposes for modeling e.g., to understand the behavior of a system, investigate the response of external inputs, or make predictions of future behavior. In this thesis, the system of interest is a bus driving on a street network and the purpose is to track its position in a real time environment.

3.1.1 State space models

In this thesis, the particle filter will be used to track the position of buses. The particle filter is based on the discrete stochastic state space models which here are used to model the bus. State space models are powerful and widely used in many different fields. A state space model is based on a hidden Markov model where a set of *state variables* describes the system. The system is fully determined by the states e.i., the historical information and previous states leading up to a point are all included among the current state variables.

Example 1 (A moving vehicle)
A moving vehicle may be modeled in many different ways. The method of choice depends on the purpose of the model, if the purpose is to predict its future position a simple mechanical model may be sufficient. The states in such a model can be position, heading direction and speed, as shown in figure 3.1. The current values of the states are assumed to be the only

information that determines future movement. How the vehicle ended up in its current state does not influence states to come. □

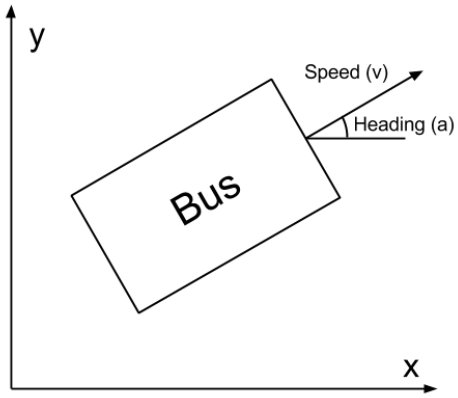

Figure 3.1: Example of a moving vehicle model. Position, speed and heading angle.

When a set of state variables x have been found that determine the system in an adequate way, modeling is the task of establishing the logic of the system dynamics. In the state space model, there are two equations to consider, the state equation and the measurement equation, as given in (3.1). The state equation $f(x)$ describes how the state variables evolve in time depending on current states. This is where the system dynamic is modeled. The measurement equation $h(x)$ describes how measurement signals are extracted from the system. It depends on the instruments it models and does not influence the state.

$$\dot{x} = f(x) \tag{3.1a}$$
$$y = h(x) \tag{3.1b}$$

3.1.2 External input and measurement output

The model system may be under influence of events and be affected by actions outside of the system boundaries. If these external influences are known, they can be modeled as input signals to the system. In most cases the state space model is only affected in the state equation by an input signal.

When a measurement is performed on the system, a signal is created based on the current state. In a state space model, the values of the state variables are given as input to a measurement function which outputs a simulated measurement.

Example 2 (Continuing example 1)
Simple mechanical laws are used to derive the logic in the vehicle model. The position will change dependent on the speed and heading direction. This makes it possible to predict future position if the current states are known. The vehicle speed can be affected by applying torque to the

wheels, in positive direction via the engine and in negative direction via the breaks. The heading direction can be changed by turning the steering wheel. If these actions are measured they can be used as input signals in the vehicle model. The speed of a vehicle is usually measured and presented as needle angle in the speedometer. This measurement can be modeled by transforming the speed state value to an angle as

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{v} \\ \dot{h} \end{pmatrix} = \begin{pmatrix} v * cos(a) \\ v * sin(a) \\ acc \\ turn \end{pmatrix} \tag{3.2a}$$

$$y = angle(v) \tag{3.2b}$$

where y is the output measurement. In the example model, x and y are the coordinates, v the speed, and h the heading direction. □

3.1.3 Continuous and discrete models

State space models can be formulated in continuous or discrete form. Both serve their purpose and complement each other in the building and usage of state space models. In the continuous time model the change in state variables is formulated as a differential equation. The discrete time model uses a difference equation, acting as a one step predictor.

Models in continuous time have some advantages in the model building phase. They are easy to derive from physical laws and other differential equations, intuitive to understand and are easy to work with analytically. Discrete models are easier to implement in computers, making them essential for numerical calculation.

3.1.4 Stochastic system

Stochastic modeling uses stochastic variables. This is a profound difference in mindset and necessary to utilize the full information of stochastic measurements. A normal variable is set to one explicit value. A stochastic variable is represented by the probability density function (PDF) describing its distribution in the value space. The two variable types are illustrated in figure 3.2. A measurement from an instrument is a stochastic process affected by noise and the encountered value can be simulated as a realization from a stochastic variable.

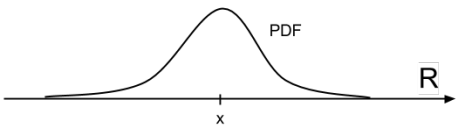

Figure 3.2: The PDF of a stochastic variable along with a normal variable x, both on the real axis.

A stochastic state space model inherit the dynamics and measurement equation from the normal state space model. The difference lies in the states which are represented by stochastic variables. The model equation then processes not only a deterministic value but the whole PDF of the states.

3.2 Particle Filters

The particle filter (PF) introduced by N. Gordon et. al. [9], use the Monte Carlo mathematics to recursively solve the non-linear filter problem of a state space model. The PF estimates the PDF of the state vector $p(x_t)$ via a set of weighted particles randomly scattered in the state space. This non-analytic approach makes it possible to filter a fully non-linear model with arbitrary colored noise input. In this section, the PF is laid out and explained with the framework of Bayesian filtering. For a deeper introduction to the PF, the reports from T. Schön [11] or M. Sanjeev [12] are recommended. The Monte Carlo mathematics are introduced by M. Skold in [10].

The state space model in (3.3) is used in this thesis together with a tracking PF to reveal the hidden states.

$$x_{k+1} = f(x_k, u_k) \tag{3.3a}$$
$$y = h(x_k) \tag{3.3b}$$

3.2.1 Bayesian filtering

The Bayesian filtering is a principal filter solution and is not analytically realizable in non Gaussian or non-linear models. It is presented to give a basic understanding of the problem of filtering and to explain the problems addressed by the PF. In the Bayesian approach, the stochastic state vector is represented by the probability density functions (PDF) $p(x_k)$. It is estimated conditioned on all available information at the time, including the measurements, $p(x_k|y_{1:k})$ [12].

Given the Bayesian dynamic state space model

$$x_{k+1} \sim p(x_{k+1}|x_k) \tag{3.4a}$$
$$y_k \sim p(y_k|x_k) \tag{3.4b}$$

the Bayesian filter may be formulated as

$$p(x_k|y_k) = \frac{p(y_k|x_k)p(x_k|y_{k-1})}{p(y|y_{k-1})} \tag{3.5a}$$
$$p(x_{k+1}|y_k) = \int_{R^{x_n}} p(x_{k+1|x_k})p(x_k|y_k)dx \tag{3.5b}$$

where

$$p(y|y_{k-1}) = \int_{R^{x_n}} p(y_t|x_t)p(x_t|y_{1:t-1})dx \tag{3.5c}$$

The first equation in (3.5) correspond to the measurement update step and follows directly from Bayes' law. The second is the model one step prediction. Together they form the recursive filter, updating the state PDF by running the model and incorporate the new measurements [13].

By representing the state with the full PDF, the Bayesian filter may be said to represent the complete solution to the filter problem. With the state PDF at hand, an optimum state estimate can be obtained [12]. In the case of a linear model and Gaussian noise, the Kalman filter (KF) provides an exact analytic solution to the Bayesian formulation. In some situations the model can be linearized and the noise pre-filtered to fit the KF but there is no formulation that covers heavily non-linear problems. The KF is limited to single point estimations and is unable to propagate PDFs with multiple peaks (hypothesis).

The PF is formulated as a numerical approach to solve the Bayesian filter. It estimates the whole state PDF with a set of particles to approximate an arbitrary shape and is limited in accuracy only by the number of particles used. In figure 3.3 the difference of a Gaussian and a particle estimation of a PDF is illustrated. If not only the mean value of the PDF but the shape is of importance, the numerical approximation offered by the PF often reassure a solution.

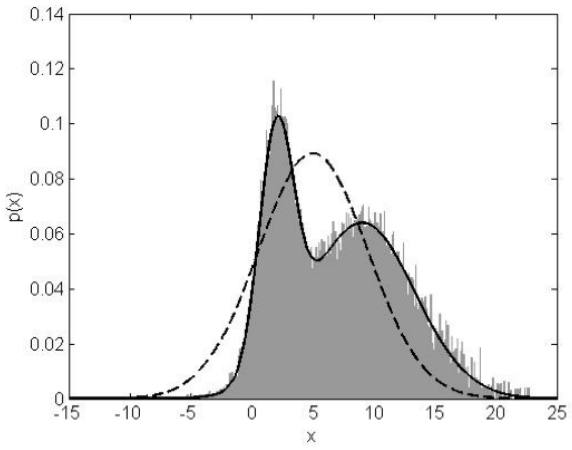

Figure 3.3: The difference in representing a PDF with Gaussian clock curve and a numerical particle approximation. Even when the PDF have a complex nature the particle approach is able to capture the shape. In the figure, the filled line is the PDF, the Gaussian approximation is the dotted line, and the histogram shows the density of particles, hence the particle approximation. Source: [11]

3.2.2 Numerical approximation

In order to understand the particle filter the numerical approximation used to estimate the PDF is introduced here. It is based on *importance sampling* where the key idea is to represent a desired target distribution $t(x)$ with

random samples x^i from a *second* distribution, called *proposal distribution*. If the samples would have been drawn from the target density itself the approximation could have been formulated as

$$\hat{t}_N(x) = \sum_{t=1}^{N} \frac{1}{N} \delta(x - x^i) \qquad i = 1, \ldots, N \tag{3.6}$$

where δ is the Dirac delta function.

The target density is usually not possible to sample from, which is why the proposal distribution $q(x)$ is used. To compensate the differences from the target distribution each sample is coupled with a *weight* describing the probability of it orienting from the target density. The weights are calculated by the fraction of the two distributions values at the sampling point using

$$w^i \propto \frac{t(x^i)}{q(x^i)}. \qquad \sum_{i=1}^{N} w^i = 1. \tag{3.7}$$

The target density function is now approximated with the following weighted sum

$$\hat{t}_N(x) = \sum_{t=1}^{N} w^i \delta(x - x^i). \qquad i = 1, \ldots, N \tag{3.8}$$

This is the idea of the numerical approximation in the particle filter, where the state PDF is the target density function. The proposal distribution have to be determined based on the filter design, model, and signal condition. This will be further discussed after a general particle filter is introduced.

3.2.3 General Particle Filter

The basic idea of the particle filter is to use random samples x^i to estimate the state PDF. These samples are called particles, hence the name particle filter. The particles are drawn from a proposal density and weighted according to the *importance sampling* theory discussed above. The two main steps in the particle filter, *time update* and *measurement update*, together samples the particles and calculates their weights. The two steps are both numerical approximations of the Bayesian filter where the whole state PDF are estimated. The PF also have a *resampling* step where the particles are re-drawn to concentrate their distribution where the state PDF is of interest. This is further discussed in section 3.2.6.

In the general filter presented here, the *proposal distribution* is not specified but presented as a function conditioned on both the state and the measurements. Because of the Markov properties of the state space model only the latest state and measurement are relevant. The state is in the particle filter represented as particles why the samples x^i are used in the general proposal distribution

$$q(x_{k+1}|x_k^i, y_{k+1}). \tag{3.9}$$

In section 3.2.5 some distributions are presented and their effect on the filter are discussed.

The state PDF is estimated according to the importance sampling estimation (**??**) and the equation is written as

$$\hat{p}(x_k|y_k) = \sum_{i=1}^{N} w_{k|k}\delta(x_k - x_k^i). \qquad i = 1, \ldots, N \tag{3.10}$$

It is valid for every iteration of the filter.

Initialization

N particles are initiated from the PDF of the initial state and the weights are given a uniformed distribution.

$$x_0^i \sim p(x_0) \qquad i = 1, \ldots, N \tag{3.11a}$$

$$w_0^i = 1/N \qquad i = 1, \ldots, N \tag{3.11b}$$

1: Time update

The time update step estimates the state PDF of the one-step prediction $p(x_{k+1}|y_k)$. It uses the proposal distribution to sample N samples and the important sampling theory to calculate the corresponding weights. The state PDF estimation is then given by

$$\hat{p}(x_{k+1}|y_k) = \sum_{t=1}^{N} w_{k+1|k}^i \delta(x_{k+1} - x_{k+1}^i) \tag{3.12}$$

where the samples are drawn from the distribution according to

$$x_{k+1}^i \sim q(x_{k+1}|x_k^i, y_{k+1}) \qquad i = 1, \ldots, N \tag{3.13}$$

The weight can then be calculated as presented in the important sampling (**??**) and results in

$$w_{k+1|k}^i = \frac{p(x_{k+1}^i|y_k)}{q(x_{k+1}^i|x_k^i, y_{k+1})} \qquad i = 1, \ldots, N \tag{3.14}$$

This way of calculating the weights assumes the target distribution $p(x_{k+1}^i|x_k^i)$ to be available. By using the Bayesian filter equation (3.5) and the state estimation (3.11a) the target distribution may be substituted as

$$p(x_{k+1}^i|y_k) = \int_{R^{x_n}} p(x_{k+1}^i|x_k)p(x_k|y_k)dx \tag{3.15a}$$

$$= \int_{R^{x_n}} p(x_{k+1}^i|x_k) \sum_{i=1}^{N} w_{k|k}\delta(x_k - x_k^i)dx \tag{3.15b}$$

$$= p(x_{k+1}^i|x_k^i)w_{k|k}^i \tag{3.15c}$$

Inserted in the equation (3.15a) the weight calculation is transformed to a weight update

$$w_{k+1|k}^i = \frac{p(x_{k+1}^i|x_k^i)}{q(x_{k+1}^i|x_k^i, y_{k+1})} w_{k|k}^i. \qquad i = 1, \ldots, N \tag{3.16}$$

After the prediction step the index k is incremented by one.

2: Measurement update

The measurement update brings in the new information from external signal into the state PDF estimation $\hat{p}(x_k|y_k)$. By using the particles from the time update only the weight has to be updated in order for estimator (3.11a) to stay valid. To derive the weight update equation the Bayesian filter (3.5) is used. Here the measurement update from the Bayesian filter is displayed

$$p(x_k|y_k) = \frac{p(y_k|x_k)p(x_k|y_{k-1})}{p(y|y_{k-1})} \tag{3.17}$$

where

$$p(y|y_{k-1}) = \int_{R^{x_n}} p(y_t|x_t)p(x_t|y_{1:t-1})dx. \tag{3.18}$$

Using the one-step prediction estimation (**??**) in the equation (3.19a) results in the approximation

$$\hat{p}(y|y_{k-1}) = \int p(y_t|x_t) \sum_{i=1}^{N} w_{k|k-1}\delta(x_k - x_k^i)dx_t \tag{3.19a}$$

$$= \sum_{i=1}^{N} p(y_t|x_t^i)w_{k|k-1}. \tag{3.19b}$$

Inserted in the Bayesian state PDF equation (**??**) and together with the one-step prediction (**??**) the measurement update equation becomes

$$\hat{p}(x_k|y_k) = \sum_{i=1}^{N} \underbrace{\frac{p(y_k|x_k)w_{k|k-1}}{\sum_{j=1}^{N} p(y_t|x_t^j)w_{k|k-1}}}_{w_{k|k}^i} \delta(x_k - x_k^i) \tag{3.20}$$

In this equation the weight update equation can be identified as a function of time update weights formulated as following

$$w_{k|k}^i = \frac{1}{c_k} p(y_k|x_k^i)w_{k|k-1}^i \qquad i = 1, \ldots, N \tag{3.21a}$$

with the normalization weight given by

$$c_k = \sum_{k=1}^{N} p(y_k|x_k^i)w_{k|k-1}^i \tag{3.21b}$$

3: Resampling

To stop the particles from diverting into states with low probability and to keep most of the particles in the region of interest, a resampling step is applied. It regenerates particles from the old set of particles based on random variables. The particles with higher weights are more likely to be regenerated. In chapter 3.2.6 this is futher discussed.

3.2.4 Estimation

The state PDF estimation is used to calculate a point estimation of the state. The minimum mean square error (MMSE) estimates the mean value of the state. The estimation equation

$$\hat{x}_k^{MV} = E\{x_k|y_k\} = \int x_k p(x_k|y_k) dx_k \tag{3.22}$$

may be approximated by inserting (3.11a), resulting in

$$\hat{x}_k^{MV} = \int x_k \sum_{i=1}^{N} w_k^i \delta x_k - x_k^i dx_k = \sum_{i=1}^{N} w_k^i x_k^i. \tag{3.23}$$

The Maximum a posteriori (MAP) estimation calculates the most likely state e.i., the peak of the state PDF.

$$\hat{x}_k^{MAP} \triangleq \arg\max p(x_k|y_{1:k}) \tag{3.24}$$

3.2.5 Proposal Distribution

From the proposal distribution $q(x)$ a new set of particles are drawn in every iteration of the PF. The choice of distribution affects the whole filter to a great extent and is an important part of the design. There are some alternatives in the literature but here three different options are presented.

In order to understand the effect of different distributions on the weights in the particle filter, the time- and measurement equations are combined. The two weight update equations, (**??**) and (3.21) may be written as

$$w_{k|k}^i \propto w_{k-1|k-1}^i \frac{p(y_k^i|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}, y_k)}. \tag{3.25}$$

Optimal distribution

The optimal distribution to sample particles from would be the PDF of the state given all knowledge. This is called the *conditional distribution* and it includes all information at hand. It may be written as

$$q(x_k|x_{k-1}^i, y_k) = p(x_k|x_{k-1}^i, y_k) = \frac{p(y_k|x_k)p(x_k|x_{k-1}^i)}{p(y_k|x_{k-1}^i)}. \tag{3.26}$$

By inserting the conditional distribution in the combined weight update equation (**??**) it becomes

$$w_{k|k}^i \propto w_{k-1|k-1}^i p(y_k|x_{k-1}^i) \tag{3.27}$$

where

$$p(y_k|x_{k-1}^i) = \int p(y_k|x_k)p(x_k|x_{k-1}^i)dx_k. \tag{3.28}$$

The complexity involved solving the integral in (**??**) makes the optimal sampling an unrealistic method in all but the linear and Gaussian case. The distribution is also hard to sample from.

The conditional distribution combines the information in the one-step prediction model and the measurements in an optimum way. It can be viewed as an optimization according to the signal to noise ratio

$$SNR \propto \frac{\parallel Q \parallel}{\parallel R \parallel} \tag{3.29}$$

where Q and R are the signals and measurements covariances. This is a good starting point to understand the two other alternatives, prior sampling and likelihood sampling. They are the two extremes where either the model or the measurements density function are used as the conditional distribution.

Prior samling

The most commonly used conditional distribution is the prior sampling. It is also the one proposed when the PF originally was presented by Gordon et. al. in [9]. It uses the one step prediction of the previous state PDF as the distribution,

$$q(x_k|x_{k-1}^i, y_k) = p(x_k|x_{k-1}^i). \tag{3.30}$$

Inserted in the combined update function (**??**) this leads to the following weight update function

$$w_{k|k}^i = \frac{1}{c_k} p(y_k|x_k^i) w_{k|k-1}^i. \tag{3.31}$$

In the equation the weights are normalized and multiplied with the measurement probability.

A major advantages of the prior sampling method over other methods is that the new particles do not need to be sampled from the proposal distribution function. The old particle from the previous iteration already have the correct distribution and weights why they can be reused.

Likelihood Sampling

When the model noise is large in comparison to the measurement noise, using likelihood sampling can be superior to using prior sampling [14]. In this case, the likelihood sampling is formulated as

$$q(x_k|x_{k-1}^i, y_k) = p(y_k|x_k). \tag{3.32}$$

The time and measurement weight update then become

$$w_{k|k}^i = p(x_{k+1}^i|x_k^i) w_{k-1|k-1}^i \tag{3.33}$$

Sampling from the likelihood function requires that the likelihood function $p(y_k|x_k)$ is integrable in respect to the state space e.i., the system needs to be observable.

3.2.6 Resampling

If the resampling step is omitted the particles will inevitably degenerate i.e., all but one particle end up with negligible weights [15]. In the resampling step, new particles are draw from the state PDF to keep a better part of the computational effort where the PDF is non-zero. There is a variety of different solutions and algorithms to perform an efficient and accurate resampling. In this thesis, the commonly used method introduced in [16, 17] is applied. The resampling is performed by duplicating or rejecting a particle based on a random variable compared to the weight. Highly weighted particles are then more likely to survive and the result is an increasing effective sample size. The resampling is formulated as generating N samples x_t^i from the old set x_t^j with replacement and the probability

$$P(x_t^i = x_t^j) = w_t^j. \qquad j = 1, \ldots, N \tag{3.34}$$

After the resampling the particles are drawn from the target density itself why the weights are uniformed according to

$$w_t^i = \frac{1}{N}. \qquad i = 1, \ldots, N \tag{3.35}$$

The resampling does not need to be applied in every iteration. To determine if the particles are degenerated the effective sample size can be measured as introduced by Kong et al. [18]. An estimate of the effective sample size $\hat{N}_{\text{eff}}$ is obtained as

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} (w_k^i)^2}. \qquad i = 1, \ldots, N \tag{3.36}$$

If the $\hat{N}_{\text{eff}}$ is smaller than some threshold value N_s a resampling algorithm is applied to the particles.

4 Signals and available information

In this chapter the signals available in the bus are presented. These signals are not unique for the specific case of this thesis but rather common in buses with a digital information system.

Another important part of the positioning system is the information known prior to the signals, including information about the street network, the location of bus stations, and the bus traveling route. In this chapter, the available databases containing the prior information are summarized.

4.1 GPS

The GPS signal is available via the central router in the bus. The router is of the model Icomera MoovBOX M220 and it provides longitude, latitude, speed, heading direction, and time with a frequency of one sample per second. The coordinates are given in the WSG84 standard format. In table 4.1 the outputs are summarized.

Table 4.1: The outputs from the GPS receiver.

Signal	Explanation
Coordinate	Longitude, latitude in WSG84 format.
Speed	Knots.
Direction	Degree.
Quality	GPS quality signal given in 0, . . . ,4 where 4 is best.
Time	Receive time stamp.

The output position signal from the MoovBOX is the result of a filter inside the device where the type and parameters of the filter are unknown. Without knowing the filter details the GPS test results presented in chapter 6 indicates that the position filter is quite well tuned for the bus positioning. The distance error to the tests reference points are small. The good results are also likely based on the fact that the device is developed for the purpose of bus positioning.

The MoovBOX filter does not integrate the GPS signal with other signals so the filter does not change the assumptions and objective of the thesis. The GPS positioning signal is used without further consideration of the MoovBOX filter.

4.2 Internal signals

The internal signals are wheel speed, door open signal, and stop button pressed. All signals are linked from sensors in the bus to a signal reader in the bus computer. The door open signal is true when any of the doors are open, the stop signal is true from the time any stop button is pressed until the door is opened.

To measure the wheel speed a cogwheel is built in to one of the wheels. A sensor is detecting when the cogs pass and then generating a *tic*. The number of cogs is unknown but are estimated to be around 8 based on the minimum speed that is detectable together with the diameter of the tire. The tics are converted into a speed signal in a computer program by measuring the time between two tics. This gives the speed in the unit tics per second. In high speed the time between the tics becomes shorter but the accuracy of the timing clock stays constant. This method for calculating the speed results in a decreasing accuracy as the speed increases and induces disturbances in the wheel speed signal. The reader software for the internal signal is compiled and out of reach in this thesis. In table 4.2 the internal signals are summarized.

Table 4.2: The internal signals.

Signal	Explanation
Wheel speed	Speed signal in odometer-ticks/second.
Door open	A Boolean signal, true if a door is open.
Stop button pressed	A Boolean signal, true when a stop button is pressed, reseted (false) when the door open is set to true.

4.3 Open Street Maps

The open street map (OSM) in an online database providing street and public transport data under an open source license. The database is accurate, rich of information and is free to download. It is available in different standard GIS formats. It uses the WSG84 datum to store geographical information and the Google projection to produce the maps.

The data is structured in three categories, *nodes*, *ways*, and *relationships*. Nodes are the base object and the one holding the geographical information. Ways connect multiple nodes to a line, circle, area or area with holes. These are not limited to streets but are also used to represent arbitrary objects on the map such as buildings, borders and water. Relations are objects

connecting multiple ways or nodes to create groups describing a logic relation such as bus lines and rivers.

In order to describe the elements of the database a tagging system is used. A tag consists of a key-value pair that gives the element a category and a property. Two examples of way-objects tags are presented in table 4.3, first a one-way residential street with the speed limit 50 km/h and below a building with the name 'sparta'.

Table 4.3: Examples of key-values pairs in the OSM database.

Key	Value
highway	residential
oneway	yes
maxspeed	50
building	apartment
name	sparta

In this thesis, only streets where buses are allowed are of interest. An import filter is used to sort out the irrelevant elements. Objects marked with the tag-key *highway* are street elements but not all are allowing motor vehicles. To filter out the pedestrian- and bicycle streets and keep the motorized, an acceptance list is used. In table 4.4 the accepts-list is presented. The java based software Osmosis is used to populate a database with the OSM data-files. Osmosis is a well maintained software and a popular choice for this task. It is capable of cropping a portion of the map and filter the objects by tags during import.

Table 4.4: List of accepted highways for the OSM import filter.

motorway	secondary	unclassified
motorway-link	secondary-link	service
trunk	tertiary	bus-guideway
trunk-link	tertiary-link	road
primary	living-street	
primary-link	residential	

4.4 Waylink Maps

The purpose of the map used in this thesis is to let a bus model drive on the street network e.i., *routing*. The OSM database is optimized to produce map pictures and is not suitable for use in the real time positioning system. Routing makes use of the information about how ways are connected, therefore a suitable street network model holds information about the intersections. A new database model was developed to fulfill this requirement.

The new model consists of two types of objects, *waylinks* and *nodes*. A waylink is a straight line connecting two nodes and is the object holding the

geographical information. The nodes holds a list of all connecting waylinks. This structure makes it possible to drive the bus along the streets and detect intersections with multiple routing alternatives in an efficient way. It is also possible to cache part of the map in local variables without dependences to objects in the database.

4.5 Local Bus Transport system

The local bus transport system is simply a database describing the system of local buses. It contains information of the departure and arriving times, the setup of stations and platforms as well as the line/route/tour relations. This section serves to give the read a clear picture of the sometimes confusing nomenclature.

4.5.1 Line, route, and tour

To make up a bus system there are usually three levels of information associated to a bus: *line*, *route*, and *tour*.

A *line* is what travelers often refer to with a number or name, example 171, 1 or Penny Line. A bus line many times has variations within itself. These variations are for example different end points, express lines skipping a stop, and small differences in the traveling route. A variant is referred to as a *route* and is specified by a number. A tour is a unique bus traveling on a route during the current period. Usually the period is a day or a week i.e., a route can be trafficked with many buses during a day which all are given a unique *tour number*. The next day a bus on the same route and with same departure/arrival times is given the same tour number.

4.5.2 Stations, platform and stop

Stations is the holder of the name of the stop that passengers often are referring to, for example the Central station, the stadium or the Main Square. Stations are not defined in a single geographical point but rather as an area covering a group of platforms. A *platform* is on the other hand tied to a single point, or at least an area small enough to be considered a point. Platforms are the objects which buses are routed between.

Example 3 (Bus line)
Line 1 connects the Central station and the University in both directions with stops along the way as well as after the University. Departing from the Central station it uses platform F, and it arrives at the University at platform A. On the way back it passes the University at platform B and finally arrives at the Central station at platform F. During one day there are 10 round trips doing all the stops and 10 express lines skipping some in-between stations.

In this example line 1 has two directions with two variants each, which adds up to a total of four routes. Each route is populated with ten tours each day. □

4.5.3 REBUS

Available for this thesis was a database from the bus administration system REBUS which holds much information about the bus system. For the purpose of building traveling routes some of the information is of interest. The line and route numbering, the bus stops and platforms geographical information, the route stops information and the tour numbering and timetable. The platform positions where given in the SWEREF 99 format.

4.6 Route building algorithm

The positioning system developed in this thesis makes use of the bus *traveling routes*. The feasible format is an ordered list of the waylinks making up the route. Unfortunately, no traveling route information was available beside the bus stop sequence and the sensor logs from the buses. A map matching algorithm converting the logs into the feasible format was developed and will be presented in this section.

4.6.1 Log data

Every log post contained sensor data from the bus GPS and wheel speed along with a time stamp. A separate log held the tour and line information. The logs were sampled every second. To cover the whole bus system, containing 20 lines with 170 routes, log data were collected from all 60 buses during a five days period. This resulted in over 3.5 million logs.

4.6.2 Administration tool

To simplify the administration process a tool was built which made it possible to automatically pick suitable logs and present them in an interactive visualization interface. When the appropriate logs were chosen they were sent to the map matching algorithm. A screenshot from the tool is displayed in picture 4.1. It shows the logs along with the result from the map matching between two platforms. The bus system contained over 600 platform to platform connections, only 15 needed manual adjustments.

4.6.3 The algorithm

The map matching is based on a deterministic iterative algorithm developed in this thesis. The proposed route starts at the waylink closest to the first platform and then follows the street network. It clones in every intersection to evaluate all alternatives. The routes receive scores based on the number of logs found nearby the waylinks they travel on. When all alternatives have either found their way to the waylink at the end platform or starved because they took a path with no logs, the iterations end. The route with the highest score is picked as the travel route.

Figure 4.1: Screenshot from the logs administration tool. The top picture contains the suggested logs picked by the automated log search. The bottom picture displays the resulting route from the map matching algorithm. If the logs are missing or the algorithm has failed the logs can be manipulated by hand to achieve a satisfying result. To assist the user the data is displayed with Open Street Map as background.

5 Bus Positioning System

The bus positioning system (BPS) refers to the complete system developed in this thesis. The main purpose is to produce a redundant position estimates based on different sets of input signals. Additionally, it publishes relevant information about the bus such as the next station and on/off route. The BPS implements the filters, the map, the transport system database and the available signals introduced in the previous chapters.

In this chapter, the reader will be introduced to the different parts of the BPS and how they work. A system overview is laid out to show how signals and information are exchanged between the different parts. The model used in the system is explained. The internal control system is explained. It decides which signals are trustable, what filter to use, and if the estimates are correct. At last the particle filter algorithm is presented along with the different signals which are utilized in the filters.

5.1 System Overview

The bus positioning system overview is shown in figure 5.1. The parts in the figure are explained below.

- The heart of the BPS is a bank of particle *filters*, tuned to different signal inputs and conditions of the bus.
- A so called *picker* selects the appropriate filter to use based on signal states and the BPS states.
- The *estimates* are outputs from the filter and are all based on the state PDF combined with the model.
- The *model* uses the *street network* to decide where the bus is allowed to travel. When tour information is available, the *transport system database* provides the model with platform and route information.
- Based on estimates the internal *BSP states* are determined.
- The *outputs* from the BPS are gathered from the estimates and the BSP states.

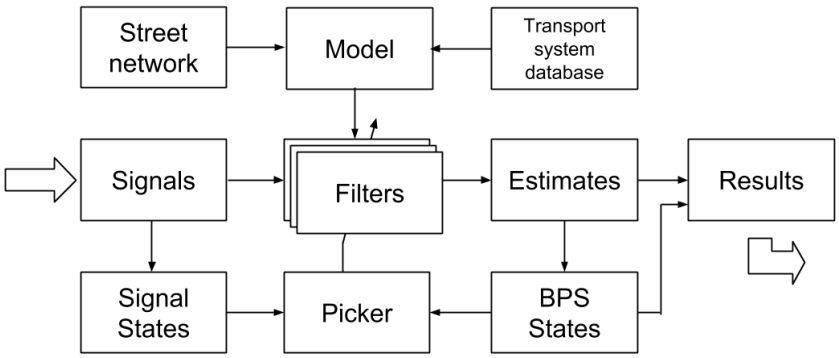

Figure 5.1: A system overview of the BPS.

5.2 The model

The model used in the filters is a non-linear stochastic state space model with non Gaussian noise. It consists of two major parts with two different objectives, a *bus model* and the *transport system database*. The purpose of the bus model is to capture the moving dynamics of the bus and use it in the filters. The bus model is based on the street map and converts the movement of the bus to a geographical position. The transport system database aims at capturing the static prior knowledge about the world of which the bus is a part. The information is utilized in the BPS, both in the bus model and in the filters. Together the two parts form a model that brings the available information into the BPS.

5.2.1 Bus model

The bus model is a combination of three components, the *street network*, a *vehicle model*, and an *intersection routing* model. The street network is the waylink map introduced in chapter 4. The bus can only appear on the streets and there is no possibility to drive off road. The vehicle model is a simple rigid body model in one dimension, obeying Newton's first law of constant velocity. It moves seamlessly across interconnected waylinks if the intermediate nodes are only connected with two ways. When the bus crosses a node connection to multiple waylinks, it is routed according to an intersection model. In principle, to cover all alternatives the intersection routing model has to choose all directions. This makes the intersection model highly non-linear and induces the need of a multi-hypotheses filter algorithm. Because the model is stochastic, the uncertainty of which way the bus selects in a crossing is modeled as probability of taking different paths. This is illustrated in figure 5.2. In the PF, the particle will be routed randomly proportional to the distribution of the intersection probability.

The bus model is formulated as a one-step prediction of the particles

$$x_{t+1}^i = f_{model}(x_t^i, u_t, e_t), \tag{5.1}$$

where u_t are the input signals and e_t is the model noise.

The state vector x consists of three variables, *pos*, *speed*, and *waylink*. In named order they describe: the position on the current waylink measured in meters from the start node, the bus speed measured in m/s, and the current waylink. With the expanded state vector the bus model is formulated as

$$x_{t+1}^i = \begin{pmatrix} speed_{t+1}^i \\ pos_{t+1}^i \\ waylink_{t+1}^i \end{pmatrix} = \begin{pmatrix} f_{speed}(speed_t^i, u_t, e_t) \\ f_{pos}(pos_t^i, speed_{t+1}^i, e_t) \\ f_{cross}(pos_{t+1}^i, u_t, e_t) \end{pmatrix} \tag{5.2}$$

where the model function f_{model} is separated in three function. Together with the map the state vector fully describes the global position of a particle, heading direction, speed and current waylink.

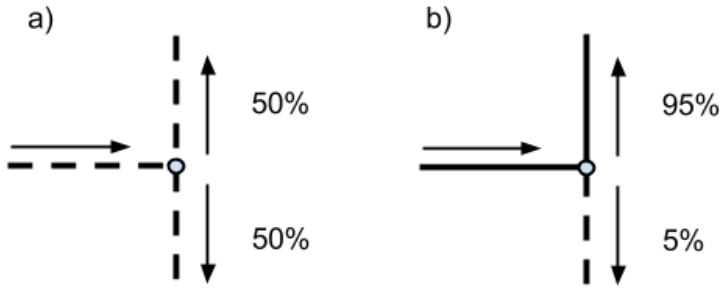

Figure 5.2: Examples of two scenarios when a bus crosses an intersection with two alternative ways. a) The bus is traveling without or outside a known route. The probabilities of a left- or right turn are equal. b) The bus is traveling on a known route which continues to the left. The probability of making a left turn and stay on the route is higher than going off route by going right.

5.2.2 Transport system database

The transport system database was introduced in section 4.5. The traveling route waylink lists are used in the intersection routing function $f_{waylink}$. When the bus travels on a confirmed route the intersection model gives higher probability to waylinks found in the route waylink-list.

The platform positions are used as a measurement y_t^{plat} when the bus is estimated to be at a platform. In section 5.5.5 this is further discussed.

5.3 BPS states

An important part of the BPS is the *BPS states*, which are not to be confused with the states of the state space model. The BPS states describe the statuses of the signals, the tour estimation and the bus movement.

To summarize the functionality of the BPS states the tables 5.1 to 5.3 lists the possible statuses of the states. A state is set to one status at the time and to switch between them a *state machine* is associated with every state. The state machines formulate the conditions given in the tables into programing code algorithms. These algorithms runs in every iteration of the filter and put the states in the corresponding status.

All input signals in the BPS have a state describing the status of the signal. The statuses and conditions are listed in table 5.1. The GPS signal is complemented with a trust state based on a trust parameter from the MoovBOX. The state is set by float values between 0 and 1.

Table 5.1: The signal state statuses and conditions. The same set is used for all signals.

Status	Condtion
No	No signal is available.
Out off bounds	Signal value is invalid.
Timed out	No signal received during a time-out period.
Ok	The signal is available and ok.

The *tour estimation state* is described in the table 5.2. The state is used in the filter picker module and have major impact on which filter is selected. E.g., if the tour is not confirmed and the tour estimation state is not set to *on route*, neither the bus traveling route nor the platform positions are used in the filter.

Table 5.2: The tour estimation state statuses and conditions.

Status	Condtion
No	No tour is received from the server.
New signal	A new tour is received.
On route	The bus position is on the route.
Off route	The bus position has deviated from the route.
Unverified	The bus position has never been on the route.
End	The bus is at the end station.

The *bus movement* state is described in table 5.3. It is also used in the picker e.g, when the bus is *at platform* the platform positioning is used in the filter. The different statuses are also used in the picker module to adjust the filter parameters.

5.4 Filter picker

The filter picker uses the BPS states to select an appropriate filter and filter parameters. Which filter it selects mainly depends on the available signals but also on the tour state. For example, if the GPS fails and the BSP switches to a dead reckoning filter where the speed and a known route are

Table 5.3: The bus movement state statuses and conditions.

Status	Condtion
No	The bus is not on a tour.
Between stops	The bus position is in-between to stops.
Closing	A transition condition when the bus enter a platform-zones.
Passing	The bus is inside a platform-zone.
At platform	The bus is inside a platform-zone and has opened the door.
Leaving platform	The bus has been at platform and closes the door.
End	The bus is at the end station.

used to continue tracking the position, there has to be a valid route estimate. The filter parameters are set based on the current chosen filter together with the bus and tour states.

5.5 The Filters

All filters in the filter bank are particle filters and use similar algorithms. They differ in which signals and type of information that are used, the parameters, and the noise distribution used in the bus model. In this section, the particle filter algorithm based on the theory in chapter 3 will be presented. The different signals and information used are discussed and how they affects the filters.

5.5.1 The particle filter algorithm

The particle filter algorithm used in the filters is based on the general particle filter together with the prior sampling principle introduced in chapter 3. It is an iterative algorithm running once every time period. In this thesis, the time period is one second. The filter is presented in Algorithm 1.

5.5.2 Wheel Speed

When the wheel speed signal is available, it is used as an input signal to the bus model. This reduces the variance of the speed state in the state space model together with the position state which is partly the result of the integrated speed.

The wheel speed signal has a decreasing resolution in higher speed due to poor clock resolution in the signal reading software, as discussed in chapter 4. The wheel speed is calculated by dividing the time difference between two ticks in the wheel signal

$$speed \sim \frac{1}{t_1 - t_0}. \tag{5.8}$$

Algorithm 1 The particle filter

1. (At startup or filter reboot) Initialize the particles from either a GPS- or a platform position in y_0^{pos} together with the wheel speed if available, else use speed:0 in y_0^{speed}. The particles are drawn from the initial measurement as

$$x_0^i \sim p(x_0|y_0) = \mathcal{N}(y_0, \sigma^y) \qquad i = 1, \ldots, N \tag{5.3a}$$
$$w_0^i = 1/N \tag{5.3b}$$

2. Increase the time t with the time difference since the last iteration.

3. Perform a one-step prediction by running the bus model according to

$$x_t^i = f_{model}(x_{t-1}^i, u_t, e_t). \qquad i = 1, \ldots, N \tag{5.4}$$

4. Update the weights by multiplying the old weight with the measurement probability density according to

$$w_t^i = p(y_t|x_t^i) w_{t-1}^i. \qquad i = 1, \ldots, N \tag{5.5}$$

5. Calculate the the effective sample size according to

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} (w_k^i)^2} \tag{5.6}$$

   and perform a resampling if the $\hat{N}_{\text{eff}}$ is smaller than the threshold value N_s. The resampling is performed by generating N samples x_t^i from the old set x_t^j with replacement and the probability

$$P(x_t^i = x_t^j) = w_t^j. \qquad j = 1, \ldots, N \tag{5.7}$$

   The weights are set to $w_t^i = 1/N$.

As the timing software approaches its lower resolution limit the resulting speed signal becomes discrete. The resolution e.i., the step size between two neighboring speed levels, is plotted in figure 5.3. As shown in the plot, the resolution degradation accelerate as the speed increases. In higher speed the level difference is over 0.5 m/s. At speed below 5 m/s the resolution is bounded by the speed variable resolution. To model this distortion a uniform noise is added to the wheel speed signal with the variance following the step between two speed levels.

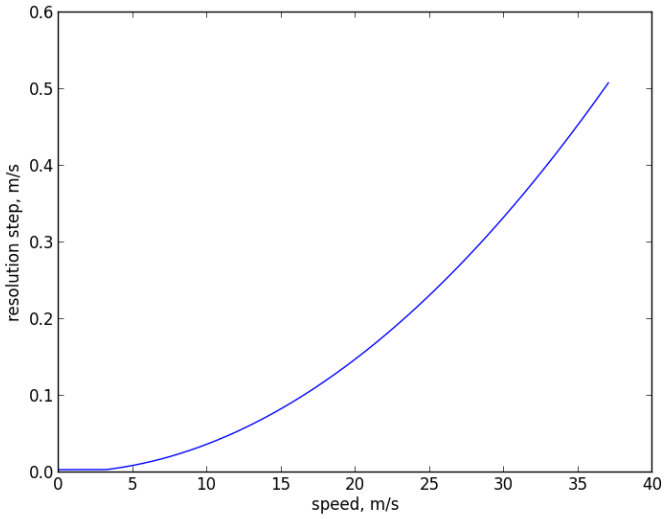

Figure 5.3: The step size between two speed levels. The wheel speed resolution degrade as the speed increases caused by the resolution in the signal reading software.

5.5.3 GPS position

The GPS position signal y^{gps} is used in the measurement update step in the filter. The measurement probability density $p(y_t^{gps}|x_t^i)$ is multiplied with the particle weight.

The GPS position error is tested in chapter 6. The error in the heading direction is shown to be approximately normal distributed. The error in the perpendicular direction i.e, sideways, is significantly smaller. For computational reasons the measurement probability density is symmetrically normal distributed. The density is then calculated based on the absolute distance from the GPS position to the individual particles as

$$p(y_t^{gps}|x_t^i) = \mathcal{N}(|y_t^{gps} - pos_t^i|; 0, \sigma^{gps}). \qquad i = 1, \ldots, N. \tag{5.9}$$

The MoovBOX GPS quality signal can be used as a warning for disturbance in the position signal. No disturbances accured during the test with reference points presented in chapter 6 but some occasions were found in

the logs where the quality signal differed from the highest state, 4. Visual examination of the position in these logs showed a lasting disturbance over a long period of time after a shorter drop in the quality signal. In figure 6.6, the GPS quality signal decreases for a couple of seconds just as the bus enter the figure but the disturbance lasts and get worse during the whole figure.

The quality signal is used to adjust the variance in the GPS position measurement noise in an ad-hoc way. The algorithm used is developed in the scope of this thesis and is presented in algorithm 2.

Algorithm 2 GPS quality dependent measurement noise variance

1. Normalize the quality signal and low-pass filter it to adjust for the slow recovery. The filter output is denoted q.

2. Calculate the quality noise variance as

$$\sigma^q = q^{max}(1 - q^8). \tag{5.10}$$

3. Add it to the GPS position variance

$$\sigma^{total} = \sigma^{gps} + \sigma^q. \tag{5.11}$$

5.5.4 GPS direction

When the speed exceeds 3 m/s the GPS direction signal is used in the measurement update function. In lower speeds the direction signal becomes to noisy and is not able to detect turning maneuvers [2]. Above the threshold the signal is assumed to be normal distributed around the unit circle, a half revolution in both positive and negative direction from the measurement. The particle heading direction dir^i is calculated from orientation of the current waylink it is associated with. The measurement density function becomes

$$p(y_t^{dir}|x_t^i) = \mathcal{N}(|y_t^{dir} - dir_t^i|; 0, \sigma^{dir}). \qquad i = 1, \ldots, N \tag{5.12}$$

5.5.5 Platform position

When there is a verified tour and the bus is estimated to be at-platform the platform positions from the database are used as a measurement signal. Each particle is coupled to its most likely platform based on position and which platforms that have passed. The measurement probability density function is assumed to be

$$p(y_t^{plat}|x_t^i) = \mathcal{N}(|y_t^{plat} - pos_t^i|; 0, \sigma^{plat}) \qquad i = 1, \ldots, N. \tag{5.13}$$

The normal distribution assumption is based on the observation that buses aim at stoping on a specific point. Together with the central limit theorem this motivates the assumption.

5.6 Implementation

All testing and implementation have exclusively been carried out in Python. Python is a popular high level programing language and was chosen primarily because it's the standard language at the company where the thesis was conducted. To power python with database functionality the GeoDjango framework has been used. GeoDjango is an extension to the popular Djagno framework and adds spatial functionality. The Django framework's main objective is to give an integration layer towards databases and an easy way to set up a web service. The BPS implementation uses the database interface in the real time filter. For the purpose of the transport system administration the web interface is used to interactively work with the spatial information such as GPS logs, platform positions and the waylink model. The database running against Django is PostgreSQL.

The particle filters and the models are implemented with the NumPy package. NumPy offers functionality and boost in performance when working with vectors, matrixes and statistical methods.

The communication with the surrounding systems are mainly signal inputs and result publications. These channels can be configured to send and receive via different interfaces. The one used in the final product is the socket based framework ZeroMQ. It offers a platform independent communication over IP. Here it is used to communicate between two processes on the same computer.

6 Results

In this chapter, the results from the performance tests of the BPS system is presented where the systems ability to handle different sets of available informations are shown. Also the GPS is tested against external test point to determine the performance of the GPS position signal. In order to carry out the tests a test environment was developed to simulate the conditions of the on-board bus computer where the signals arrive out of sync. The simulator uses log data form buses in traffic and are presented in section 6.1

6.1 Test environment

The BPS is design to run in the real-time environment of the on-board bus computer. In the test phase where the system is tried out and parameters are trimmed, the live bus computers can not be used. For this purpose a test environment was developed on a desktop computer simulating the conditions of the bus. It uses log-data from buses in traffic and simulates the out of sync arrival of the signals. All the results presented in this chapter are output from the simulator. The simulator is assumed to approximate the bus environment well the results are considered to give a correct picture of the expected performance in the finished product.

The log-data consist of the signals presented in chapter 4 which are GPS, internal signals and current tour. Incoming signals are logged and time stamped as they become available in the bus computer. This conforms with the point in the bus computer system where the BPS software will read the signals. The accuracy of the timestamp is in sub-milliseconds.

6.2 External testpoints

In order to validate the performance of the on-board GPS, external position measurements were carried out. Two routes with three tours each were chosen for the purpose. Along the two routes a total of 30 checkpoints where predetermined to cover different environments and state of the buses. A person traveling on the buses made a time notation at every checkpoint to

fix the position in time. One tour was lost because the information system of the test bus was out of order. In total, 78 checkpoints were collected.

To estimate the error introduced by the person preforming the checkpoint timings, a validation test was carried out to determine how accurate the person was able to clock an event. In the test, the same person and timing software was used. The error looked normal distributed and the result of the test has a mean of 0.09 s with the standard divination of 0.10 s.

6.3 GPS performance

The checkpoints are used to evaluate the performance of the GPS position signal. The GPS error is defined as the distance between the checkpoint and the GPS position. To minimize the effect of the GPS output frequency the GPS position is interpolated between two outputs.

To analyze if the errors are mainly in the moving direction of the bus, the vectors leading from the GPS positions to the checkpoint are projected onto a vector parallel to the moving direction. The moving direction is calculated by connecting the two closest GPS points. In this way the sideways error is eliminated. In figure 6.2, the absolute errors and the projected errors are plotted against the speed signal from the GPS devise. The projected errors are almost identical to the absolute errors which shows that most of the deviation is found in the moving direction.

Figure 6.1 show a histogram of the projected error. The majority of the errors are positive which means that the GPS position is lagging behind the bus.

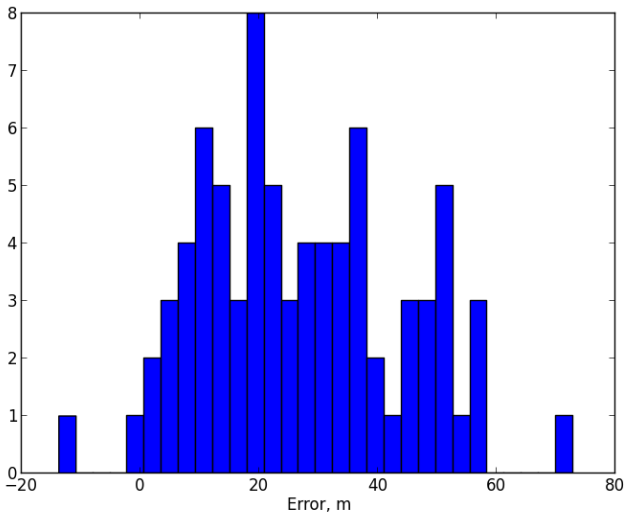

Figure 6.1: Histogram of the GPS error tested against the checkpoints.

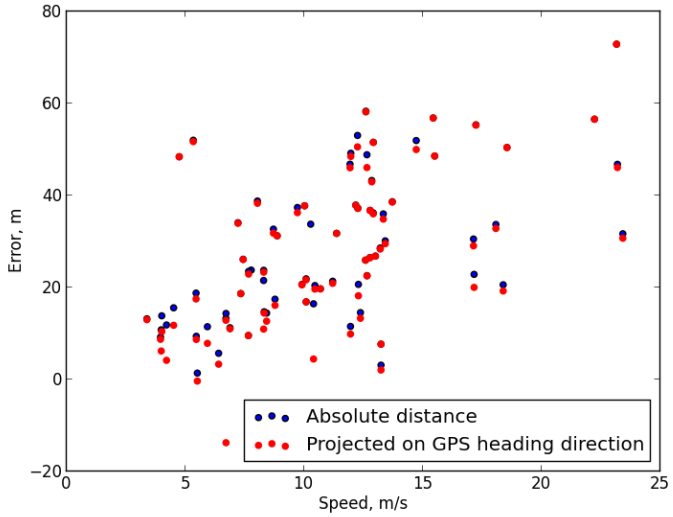

Figure 6.2: The GPS error with respect to the GPS speed, showing both the absolute error and the error in the bus moving direction.

6.4 The testes

The ability of the BPS to operate under different conditions and automatically select a proper filter is here displayed. In the simulation the signals are made unavailable or distorted which forces the BPS to detect and adapt to new conditions. In real world situations, the GPS may be out of order or heavily offset. The tour information is dependent on internet connection via the mobile network and online servers which both may be out of reach.

To simulate a lost GPS or tour signal, a log with full information and good GPS are used where the signal of interest is removed. In the case of an offset GPS a log was used that contained an offset position signal but still captured the contour of the travelled path.

The parameters used in this simulations are listed in table 6.4.

Table 6.1: Filter parametes

Parameter	Value
N	400
σ^{gps}	20 m
σ^{plat}	20
σ^{dir}	25 m
q^{max}	400 m

6.4.1 Full information

In figure 6.3, the BPS is provided with a full input signal with a low noise GPS. This includes GPS position, GPS direction, wheel speed, door open signal and tour information. The resulting position estimate follows the GPS closely but it is bounded to stay on the street network.

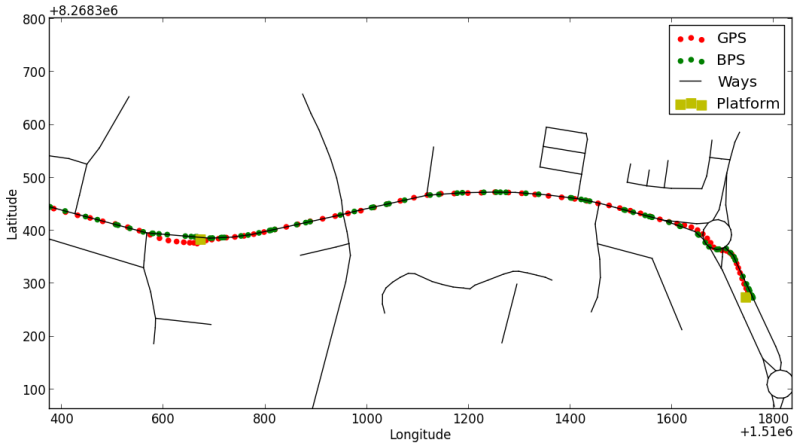

Figure 6.3: BPS performance with high precision GPS signal, wheel speed signal, and route information

6.4.2 Without tour information

In figure 6.4 the tour signal is removed simulating for example a lost internet connection. The GPS signal is still without disturbances and the wheel speed is available. Because of the lost tour signal the traveling route is unknown to the BPS. The position estimate is unaffected compared to the case where tour information was available,

6.4.3 Without GPS

To simulate a GPS blackout where the signal is not only distorted but absent, the GPS is simply removed from the simulation. This makes it possible to use the removed GPS position as a reference and benchmark the filter. Figure 6.5 shows the last minute of a five minutes blackout. The bus is traveling north and passes two stations. After the second stop, the GPS is turned back on and the distance between the two estimates, BPS and GPS, is in this simulation approximately 50 meters.

6.4.4 Distorted GPS postion

A problem that is hard to address is the situation when the GPS position is distorted but not out of boundary. An offset GPS positioning signal may be inside the allowed zone of the bus system but drift with respect to the

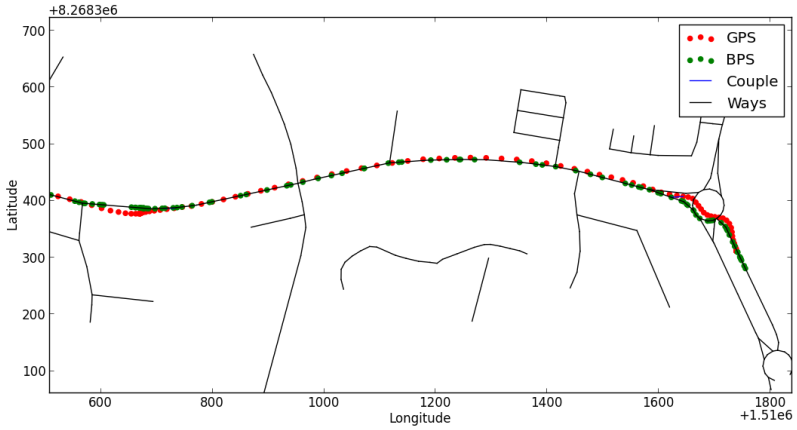

Figure 6.4: BPS performance with high precision GPS signal and wheel speed signal but without tour information

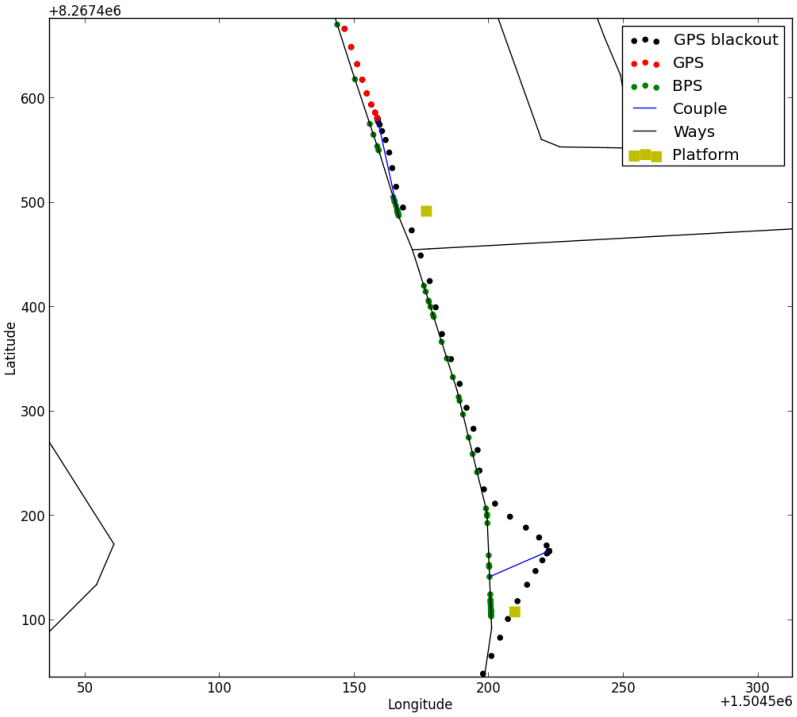

Figure 6.5: BPS performance with wheel speed signal on a verified route but with the GPS turned off. The plot shows the moment when the GPS is turned back on after a 5 minutes simulated blackout.

bus route. The ability of the PF to fuse many signals and estimate arbitrary probability density functions makes it possible to use distorted signals without failing to give an accurate estimate. In figure 6.6, a gradually failing GPS signal is fed into the simulator which estimates a correct position during the entire series. The input signals are GPS position, GPS tracking, wheel speed, door open signal and tour information. The bus travels in a westerly direction.

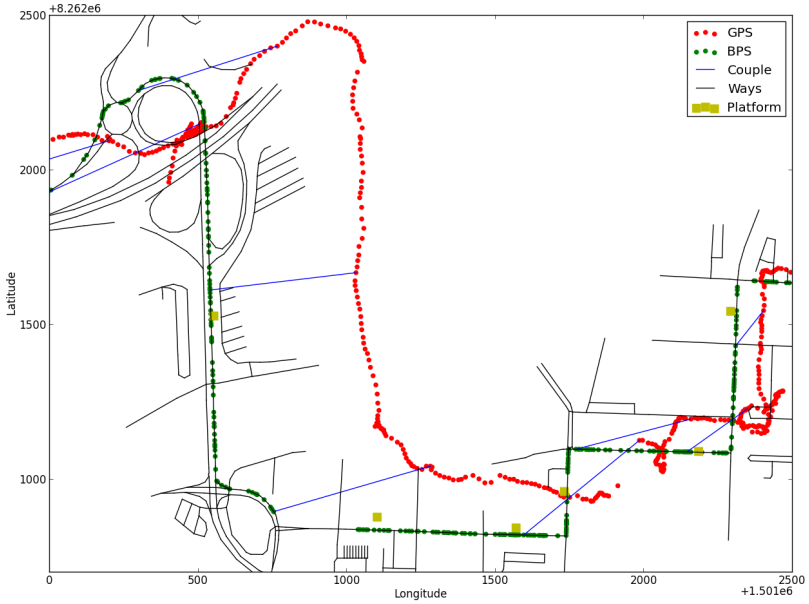

Figure 6.6: BPS performance with wheel speed signal and tour information but with a heavily distorted and offset GPS signal. The traveling route was verified earlier on the tour. The *couple* lines connect the GPS- and BPS position estimates from the same iteration.

7 Conclusion

In this master thesis, a system for improved positioning of local buses, here called the Bus Positioning System (BPS) has been developed. The positioning strategy is based on sensor fusion of GPS, on board sensors of wheel speed, and the opening of doors. A stochastic non-linear non Gaussian state space model has been developed containing both a moving vehicle model and a street map. To incorporate the prior knowledge of the transport system a database holding both spatial and temporal information has been created. The BPS uses a filter bank of particle filters to incorporate all the information and estimate the full probability density function of the state vector.

The result from the simulation with real data shows that the BPS is able to handle many different sets of signals and seamlessly change between them. With a low noise GPS, speed signals, and known tour, the system improves the position by matching the bus position to the street map. When the GPS is distorted or out of service, the BPS is able to keep the position and switch to an estimation based on the information available. The ability of the BPS to keep multiple hypotheses together with the transport system database gives it the ability to detect if the bus makes a wrong turn and leaves the route. In addition to the position estimation the BPS provides the information of the next station.

The BPS is implemented in Python and tested in a real world environment on a linux based system. Performance tests carried out on bus computers show that the computational requirements are low and that the BPS does not compromise other parts of the information system.

8 Topics of future research

A lot of improvements and further developments of the current system are possible, including extensions with new features for improved redundancy and additional functionality for improved validation.

In bigger cities, where the GPS is more distorted and unavailable for long periods of time, alternative positioning signals may be considered. The method of WiFi mapping for positioning that is being used in mobile phones and computers today may be applied in a bus positioning system in public transport. It has potential to be an attractive feature if the system is used in more dense areas.

It should be possible to validate the GPS signals by performing an online map matching with the street network. The level of consistency should then correlate with the GPS validity. This can be used to determine the trust of the GPS.

A feature with potential to improve the overall information system is a function that estimates the time of arrival at checkpoints ahead. This will improve planing of the media and in the end increase the value of the information system. This can be done by considering historical statistics or in combination with the bus model.

Acknowledgments

The work presented in this thesis was carried out at GeoSignage at Ideon Innovation incubator in Lund. I would like to thank the founder Johan Posse for the opportunity to do my Master Thesis at GeoSignage. I have had an inspiring time learning a lot in subjects far beyond the scope of thesis.

Gustav Bladh was employed at the company as the first full-time developer. He is an impressive programmer and the time I spent with him I had access to a human database of development wisdom. The major design decisions I faced and confronted Gustav with have been answered with tons of experience. Fredrik Olovsson is the sharp minded and fun head of marketing. His trust in me and my work has been encouraging throughout the entire project.

I would like to thank Prof. Andreas Jakobsson at the Centre for Mathematical Sciences at Lund University for his approval of supervising my Master Thesis and for his assistance in the course of this project.

During the time I have spent in the company it has grown from a small start-up, squeezed in on one desk in an innovation incubator to a real company with customers, investors, and employes. The environment and people working in the company have provided an entrepreneurial feeling that I will try to hold on to in my future work.

Bibliography

[1] Hirano K. Nobuta H. Itoh T Tanaka, J. Navigation System with Map-Matching Method. *ISAE Technical Paper*, (900471), 1990.

[2] WY Ochieng, M Quddus, and RB Noland. Map-matching in complex urban road networks. *Revista Brasileira de ...*, 2009.

[3] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on Signal Processing*, 50(2):425–437, 2002.

[4] AU Peker, O Tosun, and Tankut Acarman. Particle filter vehicle localization and map-matching using map topology. *Intelligent Vehicles ...*, (Iv):248–253, 2011.

[5] Peter Osborne. MERCATOR. Technical report, Edinburgh, 2013.

[6] Lantmannen. Sammanställning av kartprojektioner i alfabetisk ordning. 1(54):1–54, 2006.

[7] spatialreference.org. Epsg:3857. http://spatialreference.org/, December 2013.

[8] Lantmannen. Sweref 99. http://www.lantmateriet.se, December 2013.

[9] N J Gordon. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. 140:107–113, 1993.

[10] Martin Skold. Computer Intensive Statistical Methods. University Lecture, 2005.

[11] Thomas B Schön. Solving Nonlinear State Estimation Problems Using Particle Filters - An Engineering Perspective Abstract. Technical report, Department of Electrical Engineering Linköpings universitet, Linköpings, 2010.

[12] 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*, 50(2):174–188, 2002.

[13] K. Senne. Stochastic processes and filtering theory. *IEEE Transactions on Automatic Control*, 17, 1972.

[14] a Doucet, N De Freitas, and N Gordon. Sequential Monte Carlo Methods in Practice. *Springer New York*, pages 178–195, 2001.

[15] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. methods for Bayesian filtering. pages 197–208, 2000.

[16] Niclas Bergman. *Navigation and Tracking Applications*. Number 579. 1999.

[17] Jun S. Liu and Rong Chen. Sequential Monte Carlo Methods for Dynamic Systems. *Journal of the American Statistical Association*, 93(443):1032, September 1998.

[18] Augustine Kong, JS Liu, and WH Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American ...*, 89(425):278–288, 1994.