Recursive estimation of three-dimensional aircraft position using terrain-aided positioning

Per-Johan Nordlund

Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden

WWW: <http://www.control.isy.liu.se>

Email: perno@isy.liu.se

November 9, 2001

Report no.: LiTH-ISY-R-2403

Technical reports from the Automatic Control group in Linköping are available at <http://www.control.isy.liu.se/publications>.

Recursive estimation of three-dimensional aircraft position using terrain-aided positioning

Per-Johan Nordlund
Department of Electrical Engineering
Linköping University
SE-58183 Linköping, Sweden
Email: perno@isy.liu.se

November 9, 2001

Abstract

As a part of aircraft navigation three-dimensional position (horizontal position and altitude) must be computed continuously. For accuracy and reliability reasons several sensors are usually integrated together, and here we are dealing with dead-reckoning integrated with terrain-aided positioning. Terrain-aided positioning suffers from severe nonlinear structure, meaning that we have to solve a nonlinear recursive Bayesian estimation problem. This is not possible to do exactly, but recursive Monte Carlo methods, also known as particle filters, provide a promising approximate solution.

To reduce the computational load of the normally rather computer intensive particle filter we present algorithms which take advantage of linear structure. These algorithms are all based on a Rao-Blackwellisation technique, i.e. we marginalise the full conditional posterior density with respect to the linear part, which here is altitude. The algorithms differ in the way the linear part is estimated, but the principle is to use multiple Kalman filters. The particle filter is then used for estimating horizontal position only. Simulations show that the computational load is reduced significantly.

Contents

1	Introduction	3
2	Problem formulation	3
3	The particle filter	4
4	Rao-Blackwellization	7
4.1	Altitude estimation I	7
4.2	Horizontal position estimation I	8
4.3	Altitude estimation II	9
4.4	Horizontal position estimation II	11
5	Filter improvements	12
6	Simulations	12
7	Conclusions	15

1 Introduction

The purpose of this report is to provide optimal, yet tractable, methods for estimation of three-dimensional aircraft position (horizontal position and altitude), under the assumption that measurements are provided by a dead-reckoning system and a terrain-aided positioning system.

The integration of dead-reckoning with terrain-aided positioning is a nonlinear recursive Bayesian state estimation problem, nonlinear mainly due to the highly nonlinear nature of terrain-aided positioning and recursive because we need the estimates on-line. The classical approach to this type of problems is to use the extended Kalman filter. However, for systems with severe nonlinear and/or non-Gaussian structure the extended Kalman filter technique is not adequate. Another way to deal with nonlinear estimation problems is to approximate the sought probability density. This can be achieved by using for example grid based methods, or point mass filter [2], where the target density is discretized deterministically. The disadvantage is that they suffer from the curse of dimensionality, i.e. the complexity increases exponentially with the dimension. A third approach is to use simulation based methods, also referred to as sequential Monte Carlo methods, or particle filters [6, 4, 11, 5]. Theoretically they do not suffer from the curse of dimensionality, although in practice it can be seen that more particles are needed when the dimension of the problem increases.

The outline of the report is as follows. In Section 2 the estimation problem is described in detail. Section 3 gives a brief introduction to particle filtering. Section 4 deals with methods for reducing the computational load, for two different assumptions on the measurement noise. A discussion about filter improvements is given in Section 5 and Section 6 provide simulation results. Finally, in Section 7 conclusions are drawn.

2 Problem formulation

The principle for a dead-reckoning system is to start with an initial position and then update it by integrating (cumulating in discrete time) measured movements. For example, the inertial navigation system measures acceleration, which is integrated once to provide velocity and twice to provide position. Terrain-aided positioning uses the terrain, more specifically the terrain height variation, to extract the aircraft position. Measured terrain height, provided by taking the difference between altitude over sea-level and ground clearance, is compared to the terrain height obtained from a database. The model of the system is, in discrete time with sampling period 1 sec,

$$\begin{aligned}x_{t+1} &= x_t + u_t + v_t^x \\z_{t+1} &= z_t + v_t^z \\y_t &= h(x_t) + z_t + e_t,\end{aligned}\tag{1}$$

where x_t represent two-dimensional horizontal position and z_t represents pressure altitude error. The horizontal movement provided by the dead-reckoning system is denoted u_t . Note that (1) does not require a u_t^z because z_t represents altitude error. To be able to concentrate on position only the error inherent in

measured movement, $v_t = (v_t^x, v_t^z)^T$ is considered to be uncorrelated over time. In practice this is often not true, e.g. the INS velocity error is highly correlated, see [12] how to deal with this. Moreover, in (1), y_t represents measured terrain elevation and $h(\cdot)$ is the terrain elevation database. The measurement noise, e_t , can be described by a Gaussian mixture with two modes. The beam from the radar measuring ground clearance can, particularly when flying over dense forest, be reflected by either the ground or the tree tops. Each of these two cases is represented by one of the two modes in the Gaussian mixture. More precisely, the model of the measurement noise is

$$p_{e_t}(\cdot) = \sum_{k=0}^1 P(\gamma_t = k) \mathcal{N}(e_t; m_t^k, R_t^k), \quad (2)$$

where γ_t is a binary random variable with

$$P(\gamma_t = 0) = p_0, \quad P(\gamma_t = 1) = p_1.$$

In (2), $p_{e_t}(\cdot)$ is the probability density for e_t and $\mathcal{N}(\cdot; m, R)$ is the Gaussian density with mean m and covariance R .

The aim is to estimate the filtering posterior density $p(x_t, z_t | Y_t)$, where $Y_t = \{y_0, \dots, y_t\}$. Due to the not only nonlinear, but also nonanalytical, function $h(\cdot)$ the extended Kalman filter is practically out of the question. A feasible way to proceed is to apply the particle filter, described in Section 3. As will be shown, applying it on the entire problem will lead to an unnecessarily high computational load. By considering (1) one sees that the altitude state enters the estimation problem linearly. This implies that it should be possible to estimate z_t using standard linear methods (the Kalman filter), and thereby reducing the dimension of the problem on which the particle filter is applied. Intuitively, this should in turn decrease the computational load caused by the normally rather computer intensive particle filter.

3 The particle filter

Recursive Monte Carlo filtering methods, also referred to as particle filters, provide a solution to the general nonlinear, non-Gaussian filtering problem. Consider the state space model

$$\begin{aligned} x_{t+1} &= f(x_t) + v_t, \\ y_t &= h(x_t) + e_t, \end{aligned} \quad (3)$$

where the process and measurement noises, $v_t \sim p_{v_t}(\cdot)$ and $e_t \sim p_{e_t}(\cdot)$ respectively, are assumed independent with known but arbitrary densities. Furthermore, the state of the system, $X_t = \{x_0, \dots, x_t\}$, is a Markov process,

$$p(X_t) = \prod_{k=0}^t p(x_k | x_{k-1}),$$

where $p(x_0 | x_{-1}) = p(x_0)$. The observations, $Y_t = \{y_0, \dots, y_t\}$, are conditionally independent given the states,

$$p(Y_t | X_t) = \prod_{k=0}^t p(y_k | x_k).$$

The aim is to recursively estimate the filtering density $p(x_t|Y_t)$, using the time and measurement recursions

$$\begin{aligned} p(x_{t+1}|Y_t) &= \int p(x_{t+1}|x_t)p(x_t|Y_t)dx_t \\ p(x_{t+1}|Y_{t+1}) &= \frac{p(y_{t+1}|x_{t+1})p(x_{t+1}|Y_t)}{p(y_{t+1}|Y_t)}, \end{aligned} \quad (4)$$

where $p(y_{t+1}|Y_t) = \int p(y_{t+1}|x_{t+1})p(x_{t+1}|Y_t)dx_{t+1}$.

In the general case there do not exist closed-form expressions for estimating $p(x_t|Y_t)$. Instead we are forced to use numerical methods. The principle for recursive Monte Carlo methods is to discretize the density, in a stochastic manner, utilizing a large number of samples. Suppose we have a set of independent samples $\{x_t^{(i)}\}_{i=1}^N$ with associated weights $\{w_t^{(i)}\}_{i=1}^N$, which together represent a Monte Carlo approximation of $p(x_t|Y_t)$. We can write this down as

$$p(x_t|Y_t) \approx \frac{\sum_{i=1}^N w_t^{(i)} \delta_{x_t^{(i)}}(x_t)}{\sum_{i=1}^N w_t^{(i)}} = \sum_{i=1}^N \bar{w}_t^{(i)} \delta_{x_t^{(i)}}(x_t), \quad (5)$$

where

$$\delta_{x_t^{(i)}}(x_t) = \begin{cases} 1 & \text{if } x_t = x_t^{(i)} \\ 0 & \text{otherwise.} \end{cases}$$

Initially, at time $t = 0$, we have

$$\begin{aligned} x_0^{(i)} &\sim p(x_0), \quad i = 1, \dots, N \\ w_0^{(i)} &= \frac{1}{N}, \quad i = 1, \dots, N. \end{aligned}$$

The set of samples with associated weights is said to be properly weighted with respect to $p(x_t|Y_t)$ if

$$\lim_{N \rightarrow \infty} \frac{\sum_{i=1}^N w_t^{(i)} g(x_t^{(i)})}{\sum_{i=1}^N w_t^{(i)}} = E_{p(x_t|Y_t)}(g(x_t)),$$

for any integrable function g . Note that if all the weights are equal ($= 1/N$) we have a set of samples that can be looked upon as drawn directly from $p(x_t|Y_t)$. Plugging (5) into the recursion formulas (4) we obtain

$$p(x_{t+1}|Y_{t+1}) \approx \frac{p(y_{t+1}|x_{t+1})}{p(y_{t+1}|Y_t)} \sum_{i=1}^N \bar{w}_t^{(i)} p(x_{t+1}|x_t^{(i)}). \quad (6)$$

From (6), probably the most obvious way to recursively create a new set of independent, properly weighted samples is to draw from $p(x_{t+1}|x_t^{(i)})$, i.e.

$$x_{t+1}^{(i)} \sim p(x_{t+1}|x_t^{(i)}). \quad (7)$$

This corresponds to that the associated weights are updated according to

$$w_{t+1}^{(i)} = p(y_{t+1}|x_{t+1}^{(i)})\bar{w}_t^{(i)}, \quad \bar{w}_{t+1}^{(i)} = \frac{w_{t+1}^{(i)}}{\sum_{i=1}^N w_{t+1}^{(i)}}. \quad (8)$$

We now have a new set of independent, properly weighted samples, and we can perform the recursions all over again.

There are several more ways to recursively update the particles and their associated weights. Another example is to draw new particles from

$$x_{t+1}^{(i)} \sim p(x_{t+1}|x_t^{(i)}, y_{t+1}), \quad (9)$$

which implies that the weights are updated according to

$$w_{t+1}^{(i)} = \frac{p(y_{t+1}|x_{t+1}^{(i)})p(x_{t+1}^{(i)}|x_t^{(i)})\bar{w}_t^{(i)}}{p(x_{t+1}^{(i)}|x_t^{(i)}, y_{t+1})} = p(y_{t+1}|x_t^{(i)})\bar{w}_t^{(i)} \quad (10)$$

It can be shown [4] that using the update formulas according to (9)-(10) we obtain a set of samples with better properties than by using (7)-(8). However, with a nonlinear measurement equation, $p(y_{t+1}|x_t)$ is not possible to compute analytically.

The weights represent the deviation or mismatch between the density which the particles are drawn from and the target density, here the posterior filtering density. For example, drawing from the prior causes the particles to spread out more and more. At the same time, the target density becomes more and more concentrated as more and more information is cumulated. This means that most of the weights will tend to zero, and only a few will be significant for representing the approximation of the target density.

To avoid this sample impoverishment we resample among the particles at regular intervals. Resampling can be done in many different ways, but one strategy known to have nice properties is residual resampling. The principle is to first multiply/discard particles according to $\lfloor N\bar{w}_t^{(i)} \rfloor$, and then to pick randomly, with replacement, amongst the rest, $M_t = N - \sum_{i=1}^N \lfloor N\bar{w}_t^{(i)} \rfloor$. In the last step, the probability for picking particle i is $M_t^{-1}(N\bar{w}_t^{(i)} - \lfloor N\bar{w}_t^{(i)} \rfloor)$. See [11, 8] for alternative resampling strategies.

A standard method for choosing when to resample is to use an approximative expression for the effective sample size of the filter [9]

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^N (\bar{w}_t^{(i)})^2}.$$

When the effective sample size falls below a threshold, $\hat{N}_{\text{eff}} < N_h$ the algorithm performs a resampling. Here, in the three-dimensional position application we use $N_h = N/3$.

The minimum mean square (MMS) estimate of x_t and its covariance are computed according to

$$\hat{x}_t^{\text{MMS}} = E_{p(x_t|Y_t)}[x_t] \approx \sum_{i=1}^N \bar{w}_t^{(i)} x_t^{(i)} \quad (11)$$

$$\begin{aligned} P_t^{\text{MMS}} &= E_{p(x_t|Y_t)}[(x_t - E_{p(x_t|Y_t)}[x_t])^2] \\ &\approx \sum_{i=1}^N \bar{w}_t^{(i)} (x_t^{(i)} - \hat{x}_t^{\text{MMS}})(x_t^{(i)} - \hat{x}_t^{\text{MMS}})^T. \end{aligned} \quad (12)$$

4 Rao-Blackwellization

The purpose of Rao-Blackwellization is to try to reduce the number of particles for a given estimation precision. Intuitively, if it is possible to estimate a part of the state vector using closed-form expressions, e.g. the Kalman filter, it should be possible to reduce the variance of the particle filter.

The trick here is to apply the particle filter for estimation of horizontal position (x_t) only. Consider for the moment the stacked vector of position history, $X_t = \{x_0, \dots, x_t\}$. The posterior density for X_t and z_t can be factorized, using Bayes' rule, according to

$$p(X_t, z_t | Y_t) = p(z_t | X_t, Y_t) p(X_t | Y_t). \quad (13)$$

From (13) we see that if there exist a closed-form expression for $p(z_t | X_t, Y_t)$ and if there is a method for recursively updating $p(X_t | Y_t)$ we can reduce the dimension on which we apply the particle filter, and thereby we should be able to reduce the number of particles needed for a certain accuracy. Note that the same approximation as in (5) is also valid for the entire position history,

$$p(X_t | Y_t) \approx \sum_{i=1}^N \bar{w}_t^{(i)} \delta_{X_t^{(i)}}(X_t). \quad (14)$$

For the filtering density for altitude bias, we can use (13) and (14) to obtain

$$\begin{aligned} p(z_t | Y_t) &= \int p(X_t, z_t | Y_t) dX_t = \int p(z_t | X_t, Y_t) p(X_t | Y_t) dX_t \\ &\approx \int p(z_t | X_t, Y_t) \sum_{i=1}^N \bar{w}_t^{(i)} \delta_{X_t^{(i)}}(X_t) = \sum_{i=1}^N \bar{w}_t^{(i)} p(z_t | X_t^{(i)}, Y_t). \end{aligned} \quad (15)$$

How to estimate $p(z_t | X_t^{(i)}, Y_t)$ in (15) is described in Section 4.1 and 4.3, assuming Gaussian measurement noise in the first case and Gaussian mixture measurement noise in the second. How to recursively update the approximation of $p(X_t | Y_t)$ in (14) is described in Section 4.2 and 4.4, following the same assumptions on the measurement noise as for the altitude estimation.

4.1 Altitude estimation I

Suppose, for the moment, that the sequence $X_t = \{x_0, \dots, x_t\}$ is given. In practice, it is the sequence $X_t^{(i)}$ obtained from the particle filter that is given, but we leave the index out in the following for notational convenience. With $\xi_t = y_t - h(x_t)$ the part of (1) applicable for z_t is

$$\begin{aligned} z_{t+1} &= z_t + v_t^z \\ \xi_t &= z_t + e_t. \end{aligned} \quad (16)$$

Assume that $p_{v_t^z}(\cdot) = \mathcal{N}(v_t^z; 0, Q_t^z)$, where $\mathcal{N}(\cdot; m, Q)$ represents the Gaussian density with mean m and covariance Q . If $p(z_0)$ and $p_{e_t}(\cdot)$ are Gaussian distributed as well we have a Gaussian posterior density $p(z_t | X_t, Y_t) = p(z_t | \Xi_t)$, with $\Xi_t = \{\xi_0, \dots, \xi_t\}$. The optimal solution is then be given by the Kalman filter, i.e.

$$p(z_t | \Xi_t) = \mathcal{N}(z_t; \hat{z}_{t|t}, P_{t|t}^z),$$

where

$$\begin{aligned} K_{t+1} &= (P_{t|t}^z + Q_t^w)(P_{t|t}^z + Q_t^w + R_{t+1})^{-1} \\ \hat{z}_{t+1|t+1} &= \hat{z}_{t|t} + K_{t+1}(\xi_{t+1} - \hat{z}_{t|t}) \\ P_{t+1|t+1}^z &= (I - K_{t+1})(P_{t|t}^z + Q_t^w). \end{aligned} \quad (17)$$

Note that it is not necessary to assume $p(z_0)$ Gaussian distributed. It is the distribution for $z_0^{(i)}$ that we need to model as Gaussian distributed. The distribution for z_0 itself can then be approximated arbitrarily well by

$$\sum_{i=1}^N \mathcal{N}(z_0; \hat{z}_{0|-1}^{(i)}, P_{0|-1}^{z,(i)}). \quad (18)$$

As mentioned in Section 2 the measurement noise e_t is not Gaussian, but rather a Gaussian mixture with two modes, see (2) and Figure 2. Given the sequence of modes $\Gamma_t = \{\gamma_0, \dots, \gamma_t\}$ and using the law of total probability we can rewrite $p(z_t|\Xi_t)$ according to

$$p(z_t|\Xi_t) = \sum_{k=1}^{2^t} p(z_t, \Gamma_t^k|\Xi_t) = \sum_{k=1}^{2^t} p(z_t|\Gamma_t^k, \Xi_t) P(\Gamma_t^k|\Xi_t). \quad (19)$$

In (19), $p(z_t|\Gamma_t^k, \Xi_t)$ is Gaussian and therefore the optimal solution is given by the Kalman filter. However, because of the exponential growth of possible mode sequences, Γ_t , the exact analytical solution is intractable.

The simplest approach is to approximate the Gaussian mixture with a single Gaussian, i.e.

$$p_{e_t}(\cdot) \approx \mathcal{N}(e_t; m_t^e, R_t^e), \quad (20)$$

where

$$\begin{aligned} m_t^e &= \mathbb{E}_{p_{e_t}}[e_t] = \sum_{k=0}^1 p_k m_t^k \\ R_t^e &= \mathbb{E}_{p_{e_t}}[(e_t - m_t^e)^2] = \sum_{k=0}^1 p_k (R_t^k + (m_t^k - m_t^e)^2) \end{aligned}$$

and we are back to the solution according to (17).

4.2 Horizontal position estimation I

Returning to (13), left to compute is $p(X_t|Y_t)$. This density can be rewritten recursively, using Bayes' rule repeatedly, according to

$$\begin{aligned} p(X_t|Y_t) &= \frac{p(y_t|X_t, Y_{t-1})p(X_t|Y_{t-1})}{p(y_t|Y_{t-1})} \\ &= \frac{p(y_t|X_t, Y_{t-1})p(x_t|\overbrace{X_{t-1}, Y_{t-1}}^{x_{t-1}})}{p(y_t|Y_{t-1})} p(X_{t-1}|Y_{t-1}) \\ &= \frac{p(y_t|X_t, Y_{t-1})p(x_t|x_{t-1})}{p(y_t|Y_{t-1})} p(X_{t-1}|Y_{t-1}). \end{aligned} \quad (21)$$

In (21), the time propagation density is given by

$$p(x_t|x_{t-1}) = p_{v_t^x}(x_t - x_{t-1} - u_{t-1}),$$

not necessarily, but usually, assumed Gaussian distributed. Regarding $p(y_t|X_t, Y_{t-1})$, let us assume here that e_t is Gaussian distributed (or approximated by a single Gaussian). The density can be rewritten according to

$$\begin{aligned} p(y_t|X_t, Y_{t-1}) &= \int p(y_t, z_t|X_t, Y_{t-1})dz_t \\ &= \int p(y_t|z_t, X_t, Y_{t-1})p(z_t|X_t, Y_{t-1})dz_t \\ &= \int p(y_t|z_t, x_t)p(z_t|\Xi_{t-1})dz_t. \end{aligned} \quad (22)$$

Thus the expression for $p(y_t|X_t, Y_{t-1})$ is

$$\int \mathcal{N}(y_t; h(x_t) + z_t + m_t^e, R_t^e) \mathcal{N}(z_t; \hat{z}_{t|t-1}, P_{t|t-1}^z) dz_t. \quad (23)$$

Completing the squares for z_t in (23) results in

$$p(y_t|X_t, Y_{t-1}) = \mathcal{N}(y_t; h(x_t) + \hat{z}_{t|t-1} + m_t^e, R_t^e + P_{t|t-1}^z). \quad (24)$$

To summarize, we can estimate $p(X_t|Y_t)$ using the particle filter algorithm described in Section 3, only changing the weight update from (8) to

$$\begin{aligned} w_t^{(i)} &= p(y_t|X_t^{(i)}, Y_{t-1})\bar{w}_{t-1}^{(i)} \\ &= \mathcal{N}(y_t; h(x_t^{(i)}) + \hat{z}_{t|t-1}^{(i)} + m_t^e, R_t^e + P_{t|t-1}^{z, (i)})\bar{w}_{t-1}^{(i)}. \end{aligned} \quad (25)$$

4.3 Altitude estimation II

Returning to the estimation of altitude, to account for the fact that the measurement noise consists of two Gaussian modes we can try to apply the generalized pseudo-Bayesian method of the first order (GPB1) [1, 7], or the interacting multiple model (IMM) [1, 3, 7].

Equation (19) can be rewritten according to

$$\begin{aligned} p(z_t|\Xi_t) &= \sum_{k=0}^1 p(z_t|\gamma_t = k, \Xi_t)P(\gamma_t = k|\Xi_t) \\ &= \sum_{k=0}^1 \frac{p(z_t|\gamma_t = k, \Xi_t)p(\xi_t|\gamma_t = k, \Xi_{t-1})}{p(\xi_t|\Xi_{t-1})} \sum_{l=0}^1 \pi_{kl}P(\gamma_{t-1} = l|\Xi_{t-1}) \end{aligned} \quad (26)$$

where $\pi_{kl} = P(\gamma_t = k|\gamma_{t-1} = l)$ is an element in the transition probability matrix

$$\pi = \begin{bmatrix} \pi_{00} & \pi_{01} \\ \pi_{10} & \pi_{11} \end{bmatrix}. \quad (27)$$

The approximate GPB1 solution is given by

$$p(z_t|\Xi_t) \approx \sum_{k=0}^1 \mathcal{N}(z_t; \hat{z}_{t|t}^k, P_{t|t}^{z, k}) \bar{\alpha}_{t|t}^k, \quad (28)$$

where

$$\begin{aligned}
\hat{z}_{t|t-1} &= \hat{z}_{t-1|t-1} \\
P_{t|t-1}^z &= P_{t-1|t-1}^z + Q_{t-1}^z \\
S_t^k &= P_{t|t-1}^z + R_t^k \\
\hat{z}_{t|t}^k &= \hat{z}_{t|t-1} + P_{t|t-1}^z (S_t^k)^{-1} (\xi_t - m_t^k - \hat{z}_{t|t-1}) \\
P_{t|t}^{z,k} &= P_{t|t-1}^z - P_{t|t-1}^z (S_t^k)^{-1} P_{t|t-1}^z
\end{aligned} \tag{29}$$

and

$$\begin{aligned}
\alpha_{t|t}^k &= \mathcal{N}(\xi_t; \hat{z}_{t|t-1} + m_t^k, S_t^k) \sum_{l=0}^1 \pi_{kl} \bar{\alpha}_{t-1|t-1}^l \\
\bar{\alpha}_{t|t}^k &= \frac{\alpha_{t|t}^k}{\sum_{k=0}^1 \alpha_{t|t}^k}.
\end{aligned} \tag{30}$$

Finally the result from the two Kalman filters is merged according to

$$\begin{aligned}
\hat{z}_{t|t} &= \sum_{k=0}^1 \bar{\alpha}_{t|t}^k \hat{z}_{t|t}^k \\
P_{t|t}^z &= \sum_{k=0}^1 \bar{\alpha}_{t|t}^k (P_{t|t}^{z,k} + (\hat{z}_{t|t}^k - \hat{z}_{t|t})(\hat{z}_{t|t}^k - \hat{z}_{t|t})^T).
\end{aligned} \tag{31}$$

The principle for the interacting multiple model (IMM) approach is very similar to GPB. The difference is that the two hypothesis are used to split and combine right before the measurement update

$$p(z_{t-1}|\gamma_t = k, \Xi_{t-1}) = \frac{\overbrace{\sum_l \pi_{kl} P(\gamma_{t-1} = l|\Xi_{t-1}) p(z_{t-1}|\gamma_{t-1} = l, \Xi_{t-1})}^{\approx \bar{\alpha}_{t-1|t-1}^k}}{\underbrace{\sum_l \pi_{kl} P(\gamma_{t-1} = l|\Xi_{t-1})}_{\approx \bar{\alpha}_{t|t-1}^k}}, \tag{32}$$

where π_{kl} is given by (27). The approximation here is that the two densities $p(z_{t-1}|\gamma_t = k, \Xi_{t-1})$ are approximated by single Gaussians, which are then updated using two Kalman filters matched to either $\gamma_t = 0$ or $\gamma_t = 1$. The weights are updated using

$$\begin{aligned}
\alpha_{t|t}^k &= \mathcal{N}(\xi_t; \hat{z}_{t|t-1}^k + m_t^k, S_t^k) \sum_{l=0}^1 \pi_{kl} \bar{\alpha}_{t-1|t-1}^l \\
\bar{\alpha}_{t|t}^k &= \frac{\alpha_{t|t}^k}{\sum_{k=0}^1 \alpha_{t|t}^k}.
\end{aligned} \tag{33}$$

The purpose of the IMM filter is to simulate GPB2, while having approximately the same computational load as GPB1. Intuitively, the more correlated

the current mode is to the previous, the better you should be able to predict $p(z_{t-1}|\gamma_t = k, \Xi_{t-1})$. On the other hand, in the case where

$$\pi = \begin{bmatrix} p_0 & p_0 \\ p_1 & p_1 \end{bmatrix}, \quad (34)$$

one can easily show that the current mode is independent of the previous, i.e. $p(\gamma_t, \gamma_{t-1}) = p(\gamma_t)p(\gamma_{t-1})$. Using (34) in (32) shows that the two solutions $p(z_{t-1}|\gamma_t = k, \Xi_{t-1})$ are the same independently of k . In other words, with (34) the IMM filter degenerates to GPB1.

4.4 Horizontal position estimation II

In Section 4.2 we showed how to estimate $p(X_t|Y_t)$ under the assumption that e_t is Gaussian distributed (or approximated by a single Gaussian). Here, we show how to estimate $p(X_t|Y_t)$ when e_t is a Gaussian mixture as in (2). From (21) we have

$$p(X_t|Y_t) = \frac{p(y_t|X_t, Y_{t-1})p(x_t|x_{t-1})}{p(y_t|Y_{t-1})}p(X_{t-1}|Y_{t-1}). \quad (35)$$

and the question is how to compute $p(y_t|X_t, Y_{t-1})$. Similar to (22), using the mode indicator γ_t we can write

$$\begin{aligned} p(y_t|X_t, Y_{t-1}) &= \\ \int \sum_{k=0}^1 p(y_t|z_t, \gamma_t = k, x_t)p(z_t|\gamma_t = k, \Xi_{t-1})P(\gamma_t = k|\Xi_{t-1})dz_t &= \\ \sum_{k=0}^1 P(\gamma_t = k|\Xi_{t-1}) \int p(y_t|z_t, \gamma_t = k, x_t)p(z_t|\gamma_t = k, \Xi_{t-1})dz_t & \end{aligned} \quad (36)$$

Using the GPB1 approximation we have

$$\begin{aligned} p(y_t|X_t, Y_{t-1}) &\approx \\ \sum_{k=0}^1 \bar{\alpha}_{t|t-1}^k \int \mathcal{N}(y_t; h(x_t) + z_t + m_t^k, R_t^k) \mathcal{N}(z_t; \hat{z}_{t|t-1}, P_{t|t-1}^z) dz_t &= \\ \sum_{k=0}^1 \mathcal{N}(y_t; h(x_t) + \hat{z}_{t|t-1} + m_t^k, R_t^k + P_{t|t-1}^z) \sum_{l=0}^1 \pi_{kl} \bar{\alpha}_{t-1|t-1}^l &= \\ \sum_{k=0}^1 \mathcal{N}(\xi_t; \hat{z}_{t|t-1} + m_t^k, S_t^k) \sum_{l=0}^1 \pi_{kl} \bar{\alpha}_{t-1|t-1}^l, & \end{aligned} \quad (37)$$

where we have used $\xi_t = y_t - h(x_t)$ and $S_t^k = R_t^k + P_{t|t-1}^z$ in the last step. Comparing (37) with (30) we see that

$$p(y_t|X_t, Y_{t-1}) \approx \sum_{k=0}^1 \alpha_{t|t}^k. \quad (38)$$

In other words, $p(y_t|X_t, Y_{t-1})$ is approximated by the sum of the unnormalized weights, $\alpha_{t|t}^k$.

For the IMM case, the solution is identical to (38), but with $\alpha_{t|t}^k$ taken from (33) instead.

To summarize, we can estimate $p(X_t|Y_t)$ using the particle filter algorithm described in Section 3, only changing the weight update from (8) to

$$w_t^{(i)} = p(y_t|X_t^{(i)}, Y_{t-1})\bar{w}_{t-1}^{(i)} \approx \sum_{k=0}^1 \alpha_{t|t}^{k,(i)} \bar{w}_{t-1}^{(i)}. \quad (39)$$

5 Filter improvements

Convergence of the particle filter is of course of major practical importance. To make the particle filter converge when using a rather small amount of particles, throughout the estimation, the initialization procedure, i.e. the initial convergence of the filter, has to be dealt with. This is especially true when the initial uncertainty is large.

The simplest approach would be to just use as many particles as needed to be sure that the filter converges. After a while, the number is decreased appropriately. To keep the computational load at a constant level, the update rate is lower when the number of particles is higher. Another way is to use some additional process noise, a so called jittering noise, see [5]. The variance of the jittering noise is preferably made larger initially, to be able to quickly cover the gaps that inherently occur when discretizing the continuous density. In [10], a discount factor technique is used. The principle is to use an additional process noise component whose variance is equal to a fraction of the estimated covariance

$$v_t^{\text{model}} = v_t^{\text{system}} + v_t^{\text{jitter}}, \quad v_t^{\text{jitter}} \sim \mathcal{N}(0, k \cdot P_t^{\text{MMS}}), \quad (40)$$

where P_t^{MMS} is given by (12) and k is typically in the order of $0.01 - 0.001$.

6 Simulations

The whole idea of trying to solve as much of the problem as possible using closed-form methods is of course to reduce the computational load required for a given accuracy. For the particle filter the computational load is proportional to the number of particles you use. Moreover, for the same number of particles the computational load for the combined particle and GPB1 filter is higher than the stand-alone particle filter. The reason for this is that, for each particle, two Kalman filters are running in parallel. For each of these $2N$ Kalman filters we need to estimate the state and the state error covariance. However, the Kalman filters are all one-dimensional implying a modest increase of the computational load. For the combined particle and Kalman filter the state error covariance is the same for all particles, meaning that the computational load should be approximately the same as for the stand-alone particle filter, for the same number of particles.

We performed a number of simulations to compare the different methods developed in previous chapters:

1. The particle filter applied on 3-dimensional position (Section 3).

2. The particle filter applied on 2-dimensional horizontal position (Section 4.2) and the Kalman filter on altitude (Section 4.1).
3. The particle filter applied on 2-dimensional horizontal position (Section 4.4) and the generalized pseudo Bayesian (GPB1) filter on altitude (Section 4.3).

Because of the chosen mode transition probabilities, π_{kl} (see Table 1), according to Section 4.3 the IMM filter will degenerate to GPB1. Therefore, in this case, there is no point in evaluating the IMM filter also, it will provide exactly the same result as the GPB1 filter.

All the simulations were performed using the trajectory depicted in Figure 1. Note that the terrain elevation data is not simulated, it is taken from a commercial database. The database contains terrain elevation sampled in a 50 by 50 meter grid, and intermediate points are obtained using bilinear interpolation. As one can deduce from Figure 1 the terrain variation for the chosen trajectory is good during the first half, to become first worse then better again during the second half. The remaining simulation parameters are given in Table 1. The

Figure 1: Terrain elevation profile.

filter parameters for the process noise and initial error distribution differ slightly compared to the corresponding simulation parameters. For all filters the model of the process noise for each of the two horizontal channels is extended to consist

Table 1: Simulation parameters

Sampling period	1 sec
Track length	120 sec
u_t	$[-100 \ 100 \ 0]^T$ m/s
$p_{e_t}(\cdot)$	$0.5 \cdot \mathcal{N}(e_t; 0, 3^2) + 0.5 \cdot \mathcal{N}(e_t; 12, 6^2)$ (see Figure 2)
$p_{v_t^x}(\cdot), p_{v_t^z}(\cdot)$	$\mathcal{N}\left(v_t^x; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 2^2 & 0 \\ 0 & 2^2 \end{bmatrix}\right), \mathcal{N}(v_t^z; 0, 0.2^2)$
$p(x_0)$	$\begin{cases} 1/2000^2 & \text{for } x_0^1 < 1000, x_0^2 < 1000 \\ 0 & \text{otherwise} \end{cases}$
$p(z_0)$	$\begin{cases} 1/(200\sqrt{3}) & \text{for } z_0 < 100 \cdot \sqrt{3} \\ 0 & \text{otherwise} \end{cases}$
π_{kl}	0.5 for $k, l = 0, 1$

of not just the system noise but also additional jittering noise

$$v_t^{x,\text{model}} = \underbrace{v_t^{x,\text{system}}}_{\sim \mathcal{N}(0, 2^2 \cdot I_2)} + \underbrace{v_t^{x,\text{add}}}_{\sim \mathcal{N}(0, k \cdot P_t^{x,\text{MMS}})}$$

following the discussion in Section 5 about filter improvements. The factor k is here set to 0.001. For the initial error distribution, the two filters estimating the vertical channel separately using Kalman filters assumes the initial altitude error to be Gaussian distributed. Of course we can for each particle draw the initial altitude from a Gaussian distribution with a small variance and a mean value drawn from the uniform distribution, and thereby approximating the true initial distribution. On the other hand, this practically means that in the limit, by letting the variance of the Gaussian distributions tend to zero, we are back to the 3-dimensional particle filter, i.e. initially, when it is needed the most, we do not take advantage of the fact that we are applying Kalman filters to the vertical channel and thereby reducing the dimension of the particle filter to two. Therefore, we simply assume that the initial altitude is Gaussian distributed, i.e for the particle filter/Kalman filter and the particle filter/GPB1 we use

$$\begin{aligned} z_{0|-1}^{(i)} &= 0, \quad i = 1, \dots, N \\ P_{0|-1}^{z,(i)} &= 100^2, \quad i = 1, \dots, N. \end{aligned}$$

This will also give an indication on how robust the Kalman and GPB1 filters are with respect to $p(z_0)$.

The result from the simulations is shown in Table 2, where the result is based on 100 Monte Carlo simulations for each filtering method and number of particles. The simulations were performed in Matlab on a Sun station (Ultra 10). The table shows that by using the particle and GPB1 filter it is enough to use less than half the number of particles compared to the stand-alone particle filter for the same accuracy (23-24 m and 1.0 m for horizontal position and altitude respectively). Less than half the number of particles in this case means that the computational load is about 65% of the load for the stand-alone particle filter. Furthermore, when using the particle and Kalman filter the computational load

Figure 2: Histogram for $p_{e_t}(\cdot) = \frac{1}{2}N(0, 3^2) + \frac{1}{2}N(12, 6^2)$.

is reduced to 25% of the load for the stand-alone particle filter. On the other hand, due to the Gaussian approximation of the measurement noise, the accuracy is worse (30 m and 1.2 m for horizontal position and altitude respectively). In Table 2, convergence is defined as the case where the MMS estimate does not differ more than three times the estimated standard deviation from the true state, for the last 60 seconds. The result according to Table 2 is also confirmed

Table 2: Number of converged runs, required simulation time and $\sqrt{(\frac{1}{61} \sum_{t=60}^{120} \text{RMSE}(\cdot)^2)}$ for $\hat{x}_t^{\text{MMS}}$ and $\hat{z}_t^{\text{MMS}}$.

	N	Time	Conv	Position	Altitude
Particle filter	10000	24.6 s	97	26.1 m	0.98 m
	11000	26.6 s	100	24.4 m	0.97 m
	12000	29.7 s	100	24.2 m	1.00 m
Particle / GPB1 filter	4000	14.3 s	99	23.5 m	0.96 m
	5000	17.7 s	100	23.1 m	0.96 m
	6000	20.9 s	100	23.0 m	0.96 m
Particle / Kalman filter	2000	4.6 s	99	30.3 m	1.23 m
	3000	6.9 s	100	30.7 m	1.24 m
	4000	8.5 s	100	30.4 m	1.24 m

by Figures 3, 4 and 5, which shows typical behaviour of the RMSE of horizontal position and altitude over time for the three filters.

7 Conclusions

In this report we have given a number of ways to estimate three-dimensional position given a dead-reckoning system and a terrain-aided positioning system. All of the methods are based on the particle filter, what differs is the way we

estimate altitude. We have shown that the particle filter is well suited for this highly nonlinear problem, but the computational load is unnecessarily high. By using the fact that altitude enters linearly we can use Rao-Blackwellisation technique to reduce the dimension on which we apply the particle filter, and thereby reduce the computational load significantly.

References

- [1] Y. Bar-Shalom and X-R. Li. *Estimation and Tracking: Principles, Techniques and Software*. Artech House, Boston, London, 1993.
- [2] N. Bergman. *Recursive Bayesian Estimation, Navigation and Tracking Applications*. Phd thesis 579, Department of Electrical Engineering, Linköping University, Linköping, Sweden, 1999.
- [3] H. A. P. Blom and Y. Bar-Shalom. The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8):780–783, August 1988.
- [4] A. Doucet, S.J. Godsill, and C. Andrieu. On sequential simulation-based methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.
- [5] P. Fearnhead. *Sequential Monte Carlo methods in filter theory*. Phd thesis, University of Oxford, 1998.
- [6] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings on Radar and Signal Processing*, volume 140, pages 107–113, 1993.
- [7] F. Gustafsson. *Adaptive Filtering and Change Detection*. John Wiley & Sons Inc, 1st edition, 2000.
- [8] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.
- [9] A. Kong, J.S. Liu, and W.H. Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(524):278–288, March 1994.
- [10] J. Liu and M. West. Combined parameter and state estimation in simulation-based filtering. In *Sequential Monte Carlo methods in Practice*. Addison-Wesley, 2001.
- [11] J.S. Liu and R. Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443), September 1998.
- [12] P-J. Nordlund and F. Gustafsson. Sequential Monte Carlo filtering techniques applied to integrated navigation systems. In *Proceedings of the 2001 American Control Conference*, volume 6, pages 4375–4380, 2001.

Figure 3: RMSE of $\hat{x}_t^{\text{MMS}}$ and $\hat{z}_t^{\text{MMS}}$ (solid lines) together with estimated covariances (dashed lines) based on 100 Monte Carlo simulations for the stand-alone particle filter (11000 particles).

Figure 4: RMSE of $\hat{x}_t^{\text{MMS}}$ and $\hat{z}_t^{\text{MMS}}$ (solid lines) together with estimated covariances (dashed lines) based on 100 Monte Carlo simulations for the Rao-Blackwellized particle filter (5000 particles) using GPB1 technique to estimate altitude bias.

Figure 5: RMSE of $\hat{x}_t^{\text{MMS}}$ and $\hat{z}_t^{\text{MMS}}$ (solid lines) together with estimated covariances (dotted lines) based on 100 Monte Carlo simulations for the Rao-Blackwellized particle filter (3000 particles) using standard Kalman filter technique to estimate altitude bias.