Particle Filters for Positioning, Navigation, and Tracking

Fredrik Gustafsson, Fredrik Gunnarsson, Niclas Bergman, Urban Forssell, Jonas Jansson, Rickard Karlsson, and Per-Johan Nordlund

***Abstract*—A framework for positioning, navigation, and tracking problems using particle filters (sequential Monte Carlo methods) is developed. It consists of a class of motion models and a general nonlinear measurement equation in position. A general algorithm is presented, which is parsimonious with the particle dimension. It is based on marginalization, enabling a Kalman filter to estimate all position derivatives, and the particle filter becomes low dimensional. This is of utmost importance for high-performance real-time applications.**

Automotive and airborne applications illustrate numerically the advantage over classical Kalman filter-based algorithms. Here, the use of nonlinear models and non-Gaussian noise is the main explanation for the improvement in accuracy.

More specifically, we describe how the technique of map matching is used to match an aircraft's elevation profile to a digital elevation map and a car's horizontal driven path to a street map. In both cases, real-time implementations are available, and tests have shown that the accuracy in both cases is comparable with satellite navigation (as GPS) but with higher integrity. Based on simulations, we also argue how the particle filter can be used for positioning based on cellular phone measurements, for integrated navigation in aircraft, and for target tracking in aircraft and cars. Finally, the particle filter enables a promising solution to the combined task of navigation and tracking, with possible application to airborne hunting and collision avoidance systems in cars.

I. Introduction

RECURSIVE implementations of Monte Carlo-based statistical signal processing [19] are known as *particle filters*; see [13] and [14]. The research has, since [21], steadily intensified; see the recent first article collection [13]. The particle filters may be a serious alternative for real-time applications classically approached by model-based Kalman filter techniques [29], [24]. The more nonlinear model, or the more non-Gaussian noise, the more potential particle filters have, especially in applications where computational power is rather cheap and the sampling rate is moderate.

The paper describes a general framework for a number of applications, where we have implemented the particle filter. The problem areas are the following.

- *Positioning*, where one's own position is to be estimated. This is a filtering problem rather than a static estimation problem, when an inertial navigation system is used to provide measurements of movement.
- *Navigation*, where, besides the position, velocity, attitude and heading, acceleration, and angular rates are included in the filtering problem.
- *Target Tracking*, where another object's position is to be estimated based on measurements of relative range and angles to one's own position.

Another related application fitting this framework, not explicitly included here, is *robot localization*; see, for instance, [43] and [44]. The problems listed above are related in that they can be described by quite similar state-space models, where the state vector contains the position and derivatives of the position. Traditional methods are based on linearized models and Gaussian noise approximations so that the Kalman filter can be applied [1]. Research is focused on how different state coordinates or multiple models can be used to limit the approximations. In contrast to this, the particle filter approximates the optimal solution numerically based on a physical model, rather than applying an optimal filter to an approximate model. A well-known problem with the particle filter is that its performance degrades quickly when the dimension of the state dimension increases. A key theoretical contribution here is to apply marginalization techniques [36], adopted and extended from [12], leading to where the Kalman filter can be used to estimate (or eliminate) all position derivatives, and the particle filter is applied to the part of the state vector containing only the position. Thus, the particle filter dimension is only 2 or 3, depending on the application, and this is the main step to get real-time high-performance algorithms.

The applications we will describe are the following.

- *Car Positioning by Map Matching*: A digital road map is used to constrain the possible positions, where a dead-reckoning of wheel speeds is the main external input to the algorithm. By matching the driven path to a road map, a vague initial position (order of kilometers) can be improved to a meter accuracy. This principle can be used as

Manuscript received Janury 30, 2001; revised October 16, 2001. R. Karlsson and P.-J. Nordlund were supported by the competence center ISIS at Linköping University. The associate editor coordinating the review of this paper and approving it for publication was Prof. Simon J. Godsill.

F. Gustafsson is with the Department of Electrical Engineering, Linköping University, Linköping, Sweden (e-mail: fredrik@isy.liu.se).

F. Gunnarsson is with the Department of Electrical Engineering, Linköping University, 58183 Linköping, Sweden. He is also with Ericsson Radio, Linköping, Sweden (e-mail: fred@isy.liu.se).

N. Bergman is with SaabTech Systems, Järfälla, Sweden (e-mail: ncbe@systems.saab.se).

U. Forssell is with NIRA Dynamics AB, Linköping, Sweden (e-mail: urban.forssell@niradynamics.se).

J. Jansson is with the Department of Electrical Engineering, Linköping University, Linköping, Sweden. He is also with Volvo Car Corporation, Linköping, Sweden (e-mail: jansson@isy.liu.se).

R. Karlsson is with the Department of Electrical Engineering, Linköping University, Linköping, Sweden. He is also with Saab Dynamics, Linköping, Sweden (e-mail: rickard@isy.liu.se).

P.-J. Nordlund is with the Department of Electrical Engineering, Linköping University, Linköping, Sweden. He is also with Saab Gripen, Linköping, Sweden (e-mail: perno@isy.liu.se).

Publisher Item Identifier S 1053-587X(02)00554-8.

a supplement to, or even replacement for, a global positioning system (GPS).

- *Car Positioning by Radio Frequency (RF) Measurements*: The digital road map above can be replaced by, or supplemented by, measurements from a terrestrial wireless communications system. For handover (to transfer a connection from one base station to another) operation, the mobile stations (MS) monitor the received signal powers from a multitude of base stations and report regularly to the network. These measurements provide a power map that can be used in a similar manner as above. Mobile stations in a near future will, moreover, provide the possibility of monitoring the traveled distance of the radio signals from a number of base stations [16]. Such measurements can also be utilized in the same manner as with the power measurements.
- *Aircraft Positioning by Map Matching* or *Terrain Navigation*: A geographical information system contains, among other information, terrain elevation. The aircraft is equipped with sensors such that the terrain elevation can be measured. By map matching, the position can be deducted [5].
- *Integrated Navigation*: The aircraft's inertial navigation system (INS) uses dead reckoning to compute navigation and flight data, i.e., position, velocity, attitude, and heading. The INS is regarded as the main sensor for navigation and flight data due to being autonomous and having high reliability [10]. However, small offsets cause drift, and its output has to be stabilized. Here, terrain navigation is used.
- *Target Tracking*: A classical problem in signal processing literature is target tracking, where an IR sensor measures relative angle, or a radar measures relative angle, range and possibly range rate, to the object [4]. For the case of a bearings-only measuring IR sensor, either the state dynamics or measurement equation is very nonlinear, depending on the choice of state coordinates; therefore, the particle filter is particularly promising.
- *Combined Navigation and Tracking*: Because the target tracking measurements are relative to one's own platform, positioning is an important subproblem. Since the sensor introduces a cross-coupling between the problems, a unified treatment is tempting.
- *Car Collision Avoidance*: This is very similar to the target tracking problem. Here, we are interested in predicting the own car's and other objects' future position; see [40]. Based on the prediction, collision avoidance actions such as warning, braking, and steering are undertaken when a collision is likely to happen. In order to have enough time to warn the driver, the prediction horizon needs to be quite long. Therefore, utilizing knowledge about road geometry and infrastructure becomes important. One way to improve the prediction of possible maneuvers is to use information in a digital map. Thus, this is a specific project including all aspects of the problems listed above.

The outline is as follows. We will start with a general framework of models covering all of our applications in Section II. Then, a general algorithm is presented, covering all applications, where special attention is paid to practical problems as divergence test, initialization, and real-time requirements. Each application in the list above is devoted its own section, and conclusions and open questions of general interest are discussed in Section VIII.

TABLE I
INTERESTING SIGNALS IN NAVIGATION AND TRACKING APPLICATIONS. INDEX (1) AND (2) INDICATES SIGNALS RELATED TO ONE'S OWN AND ANOTHER PLATFORM, RESPECTIVELY. ALL QUANTITIES CAN BELONG TO EITHER ONE-, TWO-, OR THREE-DIMENSIONAL SPACES, DEPENDING ON THE APPLICATION

Object	Position	Velocity	Acceleration
Own	$p^{(1)}$	$v^{(1)}$	$a^{(1)}$, $\delta a^{(1)}$ acc. bias
Other	$p^{(2)}$	$v^{(2)}$	-

II. MODELS

Central for all navigation and tracking applications is the motion model to which various kind of model based filters can be applied. Models that are linear in the state dynamics and nonlinear in the measurements are considered:

$$x_{t+1} = Ax_t + B_u u_t + B_f f_t \tag{1a}$$

$$y_t = h(x_t) + e_t. \tag{1b}$$

Here

x_t	state vector;
u_t	measured inputs;
f_t	unmeasured forces or faults;
y_t	measurements;
e_t	measurement error.

We assume independent distributions for f_t, e_t and x_0, with known probability densities p_{e_t}, p_{f_t} and p_{x_0}, respectively, not necessarily Gaussian. Motion models (1a) are further discussed in Section II-A. These are to a large extent similar in all applications and standard in the literature. The model (1) takes only movements into account, and we do not attempt to model for instance mechanical dynamics in the platform. That is, the equations in (1) have no model parameters. The difference between the applications mainly lies in the availability of measurements. Section II-B provides an extensive list of possible measurement equations (1b).

A. Motion Models

The signals of primary interest in navigation and tracking applications are related to position, velocity, and acceleration as summarized in Table I.

Newton's law relates known and unknown external forces on the platforms to acceleration. From the differential equations $\dot{p}_t = v_t$ and $\dot{v}_t = a_t$, we obtain relations like $p_t = p_0 + v_0 t$ if velocity is assumed constant and $p_t = p_0 + v_0 t + a_0 t^2/2$ if acceleration is assumed constant. If we here plug in the sample period T_s, we get a discrete-time model for motion between two consecutive measurements, as will be frequently used in the sequel.

Depending on whether the signals are measurable or not, they may be components of either the state vector x_t or the input signal u_t. The ambition here is to discuss models through which the applications are naturally related. In specific applications, however, other parameterizations might provide better understanding of design variables and algorithm tuning.

In positioning and navigation applications, the signals related to the own platform are of interest. If the velocity $v_t^{(1)}$ is assumed measurable (and thus part of the input signal), the state dynamics can be modeled as

$$p_{t+1}^{(1)} = \underbrace{p_t^{(1)}}_{x_t} + \underbrace{T_s v_t^{(1)}}_{B_u u_t} + \underbrace{T_s f_t^{(1)}}_{B_f f_t}. \tag{2a}$$

In several navigation applications, such as airborne, measurements of the acceleration are used instead of velocity. These are typically biased, and the true acceleration can be expressed as

$$a_{\text{true},\, t}^{(1)} = a_t^{(1)} + \delta a_t^{(1)}$$

where $a_t^{(1)}$ is the measured acceleration, and $\delta a_t^{(1)}$ is the bias. The position is extracted by dead reckoning of the measured acceleration, and therefore, the presence of acceleration bias is critical. The natural thing to do is to include the bias in the state vector and the measured acceleration in the input signal. The resulting motion model is

$$\begin{pmatrix} p_{t+1}^{(1)} \\ v_{t+1}^{(1)} \\ \delta a_{t+1}^{(1)} \end{pmatrix} = \underbrace{\begin{pmatrix} I & T_s \cdot I & T_s^2/2 \cdot I \\ 0 & I & T_s \cdot I \\ 0 & 0 & I \end{pmatrix}}_{A^{(1)}} \begin{pmatrix} p_t^{(1)} \\ v_t^{(1)} \\ \delta a_t^{(1)} \end{pmatrix}$$
$$+ \underbrace{\begin{pmatrix} T_s^2/2 \cdot I \\ T_s \cdot I \\ 0 \end{pmatrix}}_{B_u^{(1)}} a_t^{(1)} + \underbrace{\begin{pmatrix} T_s^3/6 \cdot I \\ T_s^2/2 \cdot I \\ T_s \cdot I \end{pmatrix}}_{B_f^{(1)}} f_t^{(1)}. \tag{2b}$$

Analogously, a bias in any other measured signal [e.g., a bias in the velocity in (2a)] can be considered by incorporating it in the state equation.

Thus far, the focus has been on the own platform. In a simple model of the movements of the other platform, the assumption is that its velocity $v^{(2)}$ is subject to an unknown acceleration. This yields

$$\begin{pmatrix} p_{t+1}^{(2)} \\ v_{t+1}^{(2)} \end{pmatrix} = \underbrace{\begin{pmatrix} I & T_s \cdot I \\ 0 & I \end{pmatrix}}_{A^{(2)}} \begin{pmatrix} p_t^{(2)} \\ v_t^{(2)} \end{pmatrix} + \underbrace{\begin{pmatrix} T_s^2/2 \cdot I \\ T_s \cdot I \end{pmatrix}}_{B_f^{(2)}} f_t^{(2)}. \tag{2c}$$

In the target tracking literature, a popular choice of motion model is given by the "coordinated turn"-model [4].

When considering tracking of another platform, while moving the own platform, joint navigation and target tracking can be employed. Essentially, the total motion model comprises the motion models (2b) and (2c):

$$\begin{pmatrix} p_{t+1}^{(1)} \\ v_{t+1}^{(1)} \\ \delta a_{t+1}^{(1)} \\ p_{t+1}^{(2)} \\ v_{t+1}^{(2)} \end{pmatrix} = \begin{pmatrix} A^{(1)} & 0 \\ 0 & A^{(2)} \end{pmatrix} \begin{pmatrix} p_t^{(1)} \\ v_t^{(1)} \\ \delta a_t^{(1)} \\ p_t^{(2)} \\ v_t^{(2)} \end{pmatrix}$$
$$+ \begin{pmatrix} B_u^{(1)} \\ 0 \end{pmatrix} a_t^{(1)} + \begin{pmatrix} B_f^{(1)} & 0 \\ 0 & B_f^{(2)} \end{pmatrix} \begin{pmatrix} f_t^{(1)} \\ f_t^{(2)} \end{pmatrix}. \tag{2d}$$

B. Measurement Equations

The main difference between the considered applications is the measurements available. Basically, the measurements are related to the positions of one's own platform $p^{(1)}$ and to the other object $p^{(2)}$. Therefore, the measurement equations can be categorized as depending on $p^{(1)}$ only or depending on both $p^{(1)}$ and $p^{(2)}$:

$$y_t^{(1)} = h^{(1)}\left(p_t^{(1)}\right) + e_t^{(1)} \tag{3a}$$

$$y_t^{(2)} = h^{(2)}\left(p_t^{(1)}, p_t^{(2)}\right) + e_t^{(2)} \tag{3b}$$

where the measurement noise contributions $e_t^{(1)}$ and $e_t^{(2)}$ are characterized by their distributions. If not explicitly mentioned, a Gaussian distribution is used.

In the studied applications, measurements from at least one of the categories above are available. It is important to note that any combination of the sensors is possible. The presented applications are just a few examples.

1) Measurements of Relative Distance: As always, any position has to be related to a coordinate system and a reference position. Several types of sensors (e.g., GPS, RF) basically measure the distance relative to that reference point. One possibility is distance measurements of the own position relative to points of known positions $p_i, \quad i = 1, \ldots, M$, which yields M measurement equations with

$$h_{a,i}^{(1)}\left(p_t^{(1)}\right) = \left|p_i - p_t^{(1)}\right|, \qquad i = 1, \ldots, M. \tag{3c}$$

This is also applicable when the position of another object is related to one's own position (e.g., radar, sonar, ultrasound):

$$h_b^{(2)}\left(p_t^{(1)}, p_t^{(2)}\right) = \left|p_t^{(2)} - p_t^{(1)}\right|. \tag{3d}$$

Some sensors do not measure the relative distance explicitly but rather a quantity related to the same. One example is sensors that measure the received radio signal power transmitted from a known position p_i. This received power typically decays as $\sim K_1/|p_i - p_t^{(1)}|^\alpha$, $\alpha \in [2, 5]$, where K_1 and α depend on the radio environment, antenna characteristics, terrain, etc. In a logarithmic scale, the measurements are given by

$$h_{c,i}^{(1)}\left(p_t^{(1)}\right) = K - \alpha \log_{10}\left|p_i - p_t^{(1)}\right|, \quad i = 1, \ldots, M \tag{3e}$$

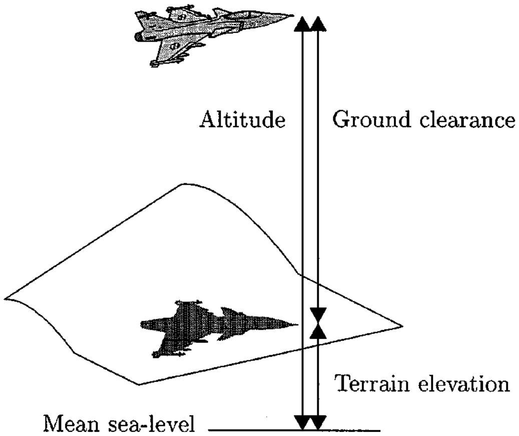

Fig. 1. Aircraft measures absolute altitude and height over ground from which terrain height y is computed.

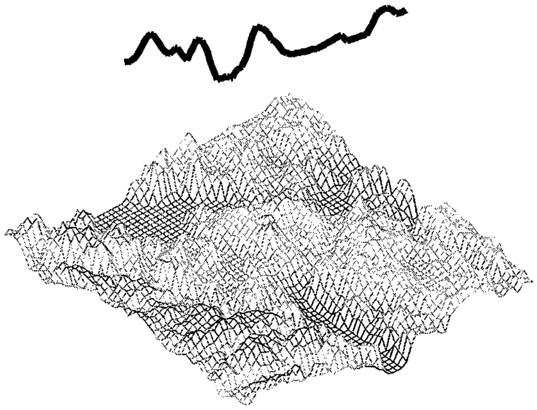

Fig. 2. Measured terrain elevation y together with measured velocity can be seen as the profile above the terrain elevation map $h(p^{(1)})$.

where $K = \log_{10} K_1$ [26]. Analogously, we can consider the situation when we focus on the power or intensity transmitted or reflected from an object and received at one's own position. The measurement is thus modeled by

$$h_d^{(2)}\left(p_t^{(1)}, p_t^{(2)}\right) = K - \alpha \log_{10} \left|p_t^{(1)} - p_t^{(2)}\right|. \qquad \text{(3f)}$$

2) Measurements of Relative Angle: Similarly, the sensors can measure the relative angle between two positions (e.g., radar, IR, sonar, ultrasound). Given points of known positions $p_i, \;\; i = 1, \ldots, M,$ the relative angle measurements can be described by

$$h_{e,i}^{(1)}\left(p_t^{(1)}\right) = \text{ angle } \left\{p_i, p_t^{(1)}\right\}, \qquad i = 1, \ldots, M. \quad \text{(3g)}$$

When relating the angle of an object to one's own position, we have

$$h_f^{(2)}\left(p_t^{(1)}, p_t^{(2)}\right) = \text{ angle } \left\{p_t^{(1)}, p_t^{(2)}\right\}. \qquad \text{(3h)}$$

3) Measurements of Relative Velocity: Some sensors (e.g., radar) typically measure the Doppler shift of signal frequencies to estimate the magnitude of the relative velocity. This is essentially only applicable when relating the velocity of an object to one's own velocity. The measurements are categorized by

$$h_{g,i}^{(2)}\left(v_t^{(1)}, v_t^{(2)}\right) = \left|v_t^{(2)} - v_t^{(1)}\right|. \qquad \text{(3i)}$$

4) Map-Related Measurements: Fig. 1 illustrates how ground altitude is computed from radar measurements of height over ground and barometric measurements from which altitude is computed. The measured terrain height together with relative movement from the INS build up a height profile, as illustrated in Fig. 2, and the task is to find the current position and orientation of this profile on the map.

The measurement in terrain navigation is the measured ground height, and $h_h(p^{(1)})$ is the height at point $p^{(1)}$ according to the Geographical Information System (GIS). Much effort has been spent on modeling the measurement error $e_t^{(1)}$ in a realistic way. It has turned out that a Gaussian mixture with two modes works well. One mode has zero mean, and the other has a positive mean that corresponds to radar echoes from the tree tops. The ground type in GIS can be used to switch the mean and variances in the Gaussian mixture. For instance, over sea there is only one mode with a small variance.

For map matching in the car-positioning case, there is no real measurement. Instead, $h_j^{(1)}(p_t^{(1)})$ denotes the distance to the nearest road, and the measurement

$$y_t^{(1)} = h_j^{(1)}\left(p_t^{(1)}\right) + e_t^{(1)}$$

should therefore be equal to zero. A simple and relevant noise model is white and zero mean Gaussian noise.

C. Applications

The applications discussed briefly in Section I are explored in further detail in the sequel. Typical state vectors, input signals, and available (nonlinear) sensor information are summarized in Table II. Motivations and more elaborative discussions regarding the applications and appropriate models are found in Sections IV–VII.

III. Particle Filter

A. Recursive Bayesian Estimation

Consider systems that are described by the generic state space model (1). The optimal Bayesian filter in this case is given below. For further details, consult [5].

Denote the set of available observations at time t by

$$Y_t = \{y_0, \ldots, y_t\}.$$

The Bayesian solution to compute the posterior distribution $p(x_t|Y_t)$ of the state vector, given past observations, is given by [5]

$$\begin{aligned} p(x_{t+1}|Y_t) &= \int p(x_{t+1}|x_t)p(x_t|Y_t)dx_t \\ &= \int p_{f_t}\left(B_f^{\dagger}(x_{t+1} - Ax_t - B_u u_t)\right) p(x_t|Y_t)dx_t \end{aligned} \qquad \text{(4a)}$$

$$\begin{aligned} p(x_t|Y_t) &= \frac{p(y_t|x_t)p(x_t|Y_{t-1})}{p(y_t|Y_{t-1})} \\ &= \frac{p_{e_t}(y_t - h(x_t))p(x_t|Y_{t-1})}{c_t} \end{aligned} \qquad \text{(4b)}$$

TABLE II
LIST OF CONSIDERED APPLICATIONS WITH THE CORRESPONDING STATE VECTOR (cf. TABLE I), INPUT SIGNAL, AND SENSOR INFORMATION

Application	State vector	Input	Measurement equations
Car positioning	$p_t^{(1)}$	$v_t^{(1)}$	Road map $h_j(p_t^{(1)})$, possibly GPS or base station distances $h_{a,i}^{(1)}(p_t^{(1)})$, base station powers $h_{c,i}^{(1)}(p_t^{(1)})$
Aircraft positioning	$p_t^{(1)}$	$a_t^{(1)}$	Altitude map $h_j(p_t^{(1)})$, GPS or other reference beacons $h_{a,i}^{(1)}(p_t^{(1)})$
Navigation in aircraft	$p^{(1)}, v^{(1)}, \delta a_t^{(1)}$	$a_t^{(1)}$	Altitude map $h_j(p_t^{(1)})$, GPS or other reference beacons $h_{a,i}^{(1)}(p_t^{(1)})$
Tracking	$p_t^{(2)}, v_t^{(2)}$		distance $h_b^{(2)}(p_t^{(1)}, p_t^{(2)})$, bearing $h_f^{(2)}(p_t^{(1)}, p_t^{(2)})$, Doppler $h_g^{(2)}(p_t^{(1)}, p_t^{(2)})$, intensity $h_d^{(2)}(p_t^{(1)}, p_t^{(2)})$
Navigation and tracking in aircraft	$p_t^{(1)}, v_t^{(1)}, \delta a_t^{(1)}, p_t^{(2)}, v_t^{(2)}$	$a_t^{(1)}$	Altitude map $h_j(p_t^{(1)})$, GPS or other reference beacons $h_{a,i}^{(1)}(p_t^{(1)})$ distance $h_b^{(2)}(p_t^{(1)}, p_t^{(2)})$, bearing $h_f^{(2)}(p_t^{(1)}, p_t^{(2)})$, Doppler $h_g^{(2)}(p_t^{(1)}, p_t^{(2)})$, intensity $h_d^{(2)}(p_t^{(1)}, p_t^{(2)})$
Navigation and tracking in cars	$p_t^{(1)}, v_t^{(1)}, \delta a_t^{(1)}, p_t^{(2)}, v_t^{(2)}$	$a_t^{(1)}$	Road map $h_j(p_t^{(1)})$, possibly GPS or base station distances $h_{a,i}^{(1)}(p_t^{(1)})$, base station powers $h_{c,i}^{(1)}(p_t^{(1)})$ distance $h_b^{(2)}(p_t^{(1)}, p_t^{(2)})$, bearing $h_f^{(2)}(p_t^{(1)}, p_t^{(2)})$, Doppler $h_g^{(2)}(p_t^{(1)}, p_t^{(2)})$, intensity $h_d^{(2)}(p_t^{(1)}, p_t^{(2)})$

$$\hat{x}_t^{\mathrm{MMS}} = \int x_t p(x_t|Y_t)\,dx_t \tag{4c}$$

$$P_t^{\mathrm{MMS}} = \int \left(x_t - \hat{x}_t^{\mathrm{MMS}}\right)\left(x_t - \hat{x}_t^{\mathrm{MMS}}\right)^T p(x_t|Y_t)\,dx_t \tag{4d}$$

where

$\dagger$	Moore–Penrose pseudo-inverse;
c_t	normalization constant;
$\hat{x}_t^{\mathrm{MMS}}$	minimum mean square (MMS) estimate.

If the noise distributions are independent, white, and zero mean Gaussian with $\mathrm{E}(e_t e_t^T) = R$, $\mathrm{E}(f_t f_t^T) = Q$ and the measurement equation is linear in the state, i.e., $h(x_t) = Cx_t$, the optimal solution is given by the Kalman estimator [29]. Table III summarizes the time and measurement update equations for the Kalman estimator.

B. Particle Filter Implementation

A numerical approximation to (4) is given in the following algorithm.

Algorithm 1: **Particle Filter***:*

1) *Initialization*: Generate $x_0^i \sim p_{x_0}$, $i = 1, \ldots, N$. Each sample of the state vector is referred to as a *particle*.

2) *Measurement Update*: Update the weights by the likelihood (more generally, any importance function; see [13])

$$w_t^i = w_{t-1}^i p\left(y_t \left| x_t^i\right.\right) = w_{t-1}^i p_{e_t}\left(y_t - h\left(x_t^i\right)\right)$$
$$i = 1, 2, \ldots, N$$

and normalize to $w_t^i := w_t^i / \sum_i w_t^i$. As an approximation to (4c), take

$$\hat{x}_t \approx \sum_{i=1}^{N} w_t^i x_t^i.$$

3) *Resampling*:

a) *Bayesian Bootstrap*: Take N samples with replacement from the set $\{x_t^i\}_{i=1}^N$, where the probability to take sample i is w_t^i. Let $w_t^i = 1/N$. This step is also called *sampling importance resampling (SIR)*.

b) *Importance Sampling*: Only resample as above when the effective number of samples is less than a threshold N_{th}

$$N_{\text{eff}} = \frac{1}{\sum\limits_i \left(w_t^i\right)^2} < N_{th}$$

see [5], [14], [34], and [35]. Here, $1 \leq N_{\text{eff}} \leq N$, where the upper bound is attained when all particles have the same weight, and the lower bound when all probability mass is at one particle. The threshold can be chosen as $N_{th} = 2N/3$.

4) *Prediction*: Take a $f_t^i \sim p_{f_t}$, and simulate

$$x_{t+1}^i = Ax_t^i + B_u u_t + B_f f_t^i, \qquad i = 1, 2, \ldots, N.$$

5) Let $t := t + 1$, and iterate to item 2).

The key point with resampling is to prevent high concentration of probability mass at a few particles. Without this step, some w_t^i will converge to 1, and the filter would brake down to a pure simulation. The resampling can be efficiently implemented using a classical algorithm for sampling N ordered independent identically distributed variables [5], [39].

It can be shown analytically [11] that under some conditions, the estimation error is bounded by g_t/N. The function g_t grows with time but does not depend on the dimension of the state space. That is, in theory, we can expect the same good performance for high-order state vectors. This is one of the key reasons for the success of the particle filter compared with other numerical approaches such as the point mass filter (a numerical integration technique that can be seen as a deterministic particle filter) [5] and filter banks [24]. The computational steps are compared with the Kalman filter in Table III. Note that the most time consuming step in the Kalman filter is the Riccati recursion of the matrix P, which is not needed in the particle filter. The time update of the state is the same. Let n_x denote the dimension of the state vector and similar definitions for n_y and n_f. As a first-order approximation for large n_x, the Kalman

TABLE III
COMPARISON OF KF AND PF: MAIN COMPUTATIONAL STEPS

Algorithm	Kalman filter	Particle filter
Time update	$x := Ax + B_u u$ $P := APA^T + B_{f_t} Q B_{f_t}^T$	$f^i \sim p_f$ $x^i := Ax^i + B_u u + B_f f^i$
Measurement update	$K = PC^T(CPC^T + R)^{-1}$ $x := x + K(y - Cx)$ $P := P - KCP$	$w^i := w^i p_e(y - h(x^i))$

filter is $\mathcal{O}(2n_x^3)$ from the matrix times matrix multiplication AP, whereas the particle filter is $\mathcal{O}(Nn_x^2)$ from the matrix times vector multiplication Ax. This indicates that the particle filter is about 100 times slower in an application with $n_x \approx 5$ and $N \approx 10^3$. The difference becomes less when n_y increases, in which case, the measurement update becomes more complex. The nonlinear function evaluation (preferably implemented as a table lookup) of $h(x)$ in the particle filter has a counterpart of computing the gradient $C = dh(x)/dx$ in the Kalman filter or any other linearization that is needed. In a multisensor application, the matrix inversion $(CPC^T + R)^{-1}$ may no longer be negligible. All in all, a precise comparison is hard to make, although it is worth pointing that the particle filter runs in real time, even in Matlab in several of the applications presented here.

C. Sample Impoverishment

When the particle filter is used in practice, we often wish to minimize the number of particles to reduce the computational burden. For many applications using recursive Monte Carlo methods, depletion or sample impoverishment may occur, i.e., the effective number of samples is reduced. This means that the particle cloud will not reflect the true density. Several different methods are proposed in the literature to reduce this problem.

By introducing an additional noise to the samples, the depletion problem can be reduced. This technique is called *jittering* in [17], but a similar approach was introduced in [21] under the name *roughening*. In [15], the depletion problem is handled by introducing an additional Markov chain Monte Carlo (MCMC) step to separate the samples.

In [21], the so-called *prior editing* method is discussed. The estimation problem is delayed one time-step; therefore, the likelihood can be evaluated at the next time step. The idea is to reject particles with sufficiently small likelihood values since they are not likely to be resampled. The update step is repeated until a feasible likelihood value is received. The roughening method could also be applied before the update step is invoked. The *auxiliary particle filter* [37] is constructed in such a way that we will simulate from particles associated with large predictive likelihoods directly. A two stage resampling may be used by this method.

D. Rao–Blackwellization

Despite the theoretical independence of accuracy on the particle dimension, it is well-known that the number of particles needs to be quite high for high-dimensional systems; see, for instance, Section VI for an illustration. To be able to use a small N and to reduce the risk of divergence, a procedure known as Rao–Blackwellization can be applied. The idea is to use the Kalman filter for the part of the state space model that is linear and the particle filter for the other part. As a motivation, the state vector in inertial navigation can have as many as 27 states, and here, the Kalman filter can be used for the 24 states, whereas the particle filter applies on the 3-D position state. The extra workload here is minor.

The motion models given in Section II can actually be rewritten in the form

$$\begin{pmatrix} x_{t+1}^{\text{pf}} \\ x_{t+1}^{\text{kf}} \end{pmatrix} = \begin{pmatrix} I & A^{\text{pf}} \\ 0 & A^{\text{kf}} \end{pmatrix} \begin{pmatrix} x_t^{\text{pf}} \\ x_t^{\text{kf}} \end{pmatrix} + \begin{pmatrix} B_u^{\text{pf}} \\ B_u^{\text{kf}} \end{pmatrix} u_t + \begin{pmatrix} B_f^{\text{pf}} \\ B_f^{\text{kf}} \end{pmatrix} f_t \tag{5a}$$

$$y_t = h(x_t^{\text{pf}}) + e_t \tag{5b}$$

where x_t^{pf} (where pf is short for particle filter) and x_t^{kf} (where kf is short for Kalman filter) is a partition of the state vector with f_t assumed Gaussian. The e_t and x_0^{pf} can have arbitrarily given distributions. As the indices indicate, the Kalman filter will be applied to one part and the particle filter for the other part of the state vector.

For a derivation of the algorithm, see the Appendix or [36]. A similar result is presented in [12] for the general case, where the state space equation is linear and Gaussian, but one observes a z_t instead of y_t, where the relation $p(z_t|y_t)$ is known. An algorithmically similar approach is given in [5], as an approximate solution to an altitude offset in terrain navigation. The result is a particle filter with N particles estimating x_t^{pf}. The difference to the standard particle filter algorithm (Algorithm 1) here is that the prediction step is done using

$$x_{t+1}^{\text{pf},i} \sim \text{N}\left(x_t^{\text{pf},\,i} + A^{\text{pf}} \hat{x}_{t|t-1}^{\text{kf},\,i} + B_u^{\text{pf}} u_t,\right.$$
$$\left. A^{\text{pf}} P_{t|t-1}^{\text{kf}} (A^{\text{pf}})^T + B_f^{\text{pf}} Q_t \left(B_f^{\text{pf}}\right)^T\right).$$

Moreover, for each particle, one Kalman filter estimates $\{x_{t+1|t}^{\text{kf},\,i}; i = 1, \ldots, N\}$ using

$$K_t = P_{t|t-1}^{\text{kf}} (A^{\text{pf}})^T \left(A^{\text{pf}} P_{t|t-1}^{\text{kf}} (A^{\text{pf}})^T + B_f^{\text{pf}} Q_t \left(B_f^{\text{pf}}\right)^T\right)^{-1}$$

$$\hat{x}_{t+1|t}^{\text{kf},i} = \overline{A}^{\text{kf}} \left(\hat{x}_{t|t-1}^{\text{kf},i} + K_t \left(z_t^i - A^{\text{pf}} \hat{x}_{t|t-1}^{\text{kf},i}\right)\right) + B_u^{\text{kf}} u_t + B_f^{\text{kf}} \left(B_f^{\text{pf}}\right)^{\dagger} z_t^i$$

$$P_{t+1|t}^{\text{kf}} = \overline{A}^{\text{kf}} \left(P_{t|t-1}^{\text{kf}} - K_t A^{\text{pf}} P_{t|t-1}^{\text{kf}}\right) \left(\overline{A}^{\text{kf}}\right)^T$$

where $\overline{A}^{\text{kf}} = A^{\text{kf}} - B_f^{\text{kf}} (B_f^{\text{pf}})^{\dagger} A^{\text{pf}}$ († denotes the Moore–Penrose pseudo-inverse), and $z_t^i = \hat{x}_{t+1}^{\text{pf},\,i} - \hat{x}_t^{\text{pf},\,i}$.

Remark 1: The covariance $P_{t|t-1}^{\text{kf}}$ and the Kalman gain K_t are the same for all particles, implying a very efficient implementation of the N parallel Kalman filters, where the P and K updates in Table III are done only once per time step.

Remark 2: The distribution for x_0^{kf} does not necessarily have to be Gaussian. We can approximate $p(x_0^{\text{kf}})$ arbitrarily well by $\{N(\hat{x}_{0|-1}^{\text{kf},\,i}, P_{0|-1}^{\text{kf}}); i = 1, \ldots, N\}$.

Remark 3: The derivation still holds if an additional nonlinear term $g(x_t^{\text{pf}})$ enters the state dynamics for x_t^{pf}.

Remark 4: The Kalman filter here applies to a state vector of dimension $n_x - n_y$, which is an improvement compared with

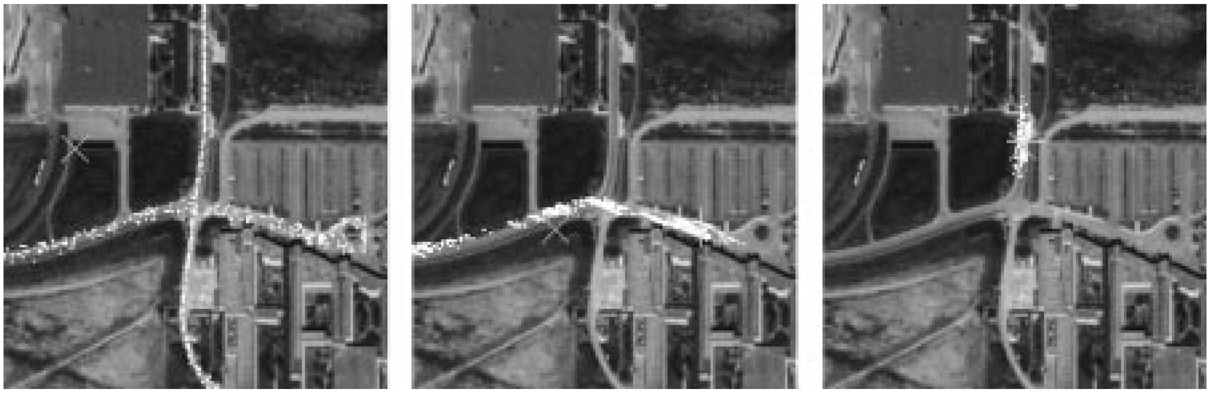

Fig. 3. Car positioning: Sequence of illustrations of particle clouds (white dots) plotted on a flight image for visualization. The center point "$+$" shows the true position and "x" the estimate.

dimension $\dim x$ as to where the derivation in [12] leads. For large n_x, the reduction in complexity is approximately

$$\mathcal{O}\left(\frac{(n_x - n_y)^3}{n_x^3}\right).$$

The estimate of the particle filter part is computed in the normal way, and for the Kalman filter part, we can take the MMS estimate (4c)

$$\hat{x}_t^{\mathrm{kf}} \approx \sum_{i=1}^{N} w_t^i \hat{x}_{t|t-1}^{\mathrm{kf},\, i}$$

with covariance (4d)

$$P_t^{\mathrm{kf}} \approx P_{t|t-1}^{\mathrm{kf}} + \sum_{i=1}^{N} w_t^i \left(\hat{x}_{t|t-1}^{\mathrm{kf},\, i} - \hat{x}_{t|t-1}^{\mathrm{kf}}\right)\left(\hat{x}_{t|t-1}^{\mathrm{kf},i} - \hat{x}_{t|t-1}^{\mathrm{kf}}\right)^T.$$

IV. Car Positioning

Wheel speed sensors in ABS are available as standard components in the test car (Volvo V40). From this, yaw rate and speed information are computed, as described in [22]. Therefore, the velocity vector $v_t^{(1)}$ is considered available as an input signal, and the motion model in (2a) with measurement equation given by (3a) is thus appropriate. The initial position is either marked by the driver or given from a different system, e.g., a terrestrial wireless communications system, where crude position information is available today [16] or GPS. The initial area should cover an area not extending more than a couple of kilometers to limit the number of particles to a realizable number. With infinite memory and computation time, no initialization would be necessary.

The car positioning with map matching has been implemented in a car, and the particle filter runs in real time with sampling frequency 2 Hz on a modern laptop with a commercial digital road map. This corresponds to a measurement equation specified by $h_j^{(1)}(p_t^{(1)})$ in Section II-B4.

Fig. 3 shows a sequence of images of the particle cloud on a flight image of the local area. The driven path consists of a number of 90° turns. Initially, the particles are spread uniformly over all admissible positions, that is, on the roads, covering an area of about 1 km^2. After the first turns, a few clouds are left.

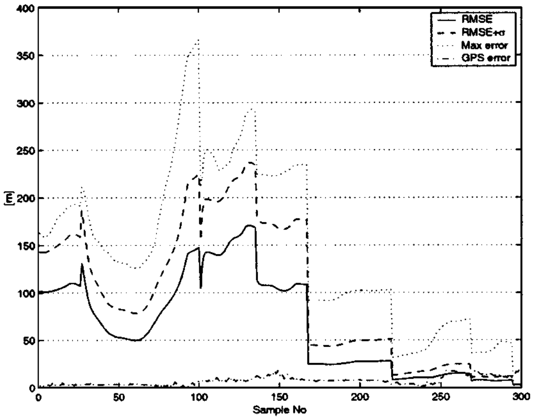

Fig. 4. Car positioning: RMSE for particle filter and GPS, respectively.

After four to five turns, the filter essentially has converged. One can note that the state evolution on the straight path extends the cloud along the road to take into account unprecise velocity information. Details of the implementation are found in [23] and [25], whereas some comments on the divergence problem are given in the conclusions.

GPS is used as a reference positioning system. It provides reliable position estimates in rural areas but is hampered in non-line-of-sight situations and when the signals are attenuated by foliage, etc. After convergence, the map-matching particle filter is seen equal to, or even slightly better than, GPS in terms of performance; see Fig. 4. However, in test drives along forests, close to high buildings, and tunnels, the GPS performance deteriorates quickly. Furthermore, the GPS has a convergence time of about 45 s when turned on; this is not shown in Fig. 4.

For comparison, the particle filter using map matching and filters based on measurements from a fictive terrestrial wireless communications system are applied to data from a simulation setup mimicking the real case above. The area is essentially covered by one macro cell, but yet another base station is assumed within measurable distance.

The base stations in a terrestrial wireless communications system act as beacons by transmitting pilot signals of known power. The mobile station monitors the M [in Global System for Mobile Communications (GSM), $M = 5$] strongest signals and reports regularly (or event-driven) the list to the network.

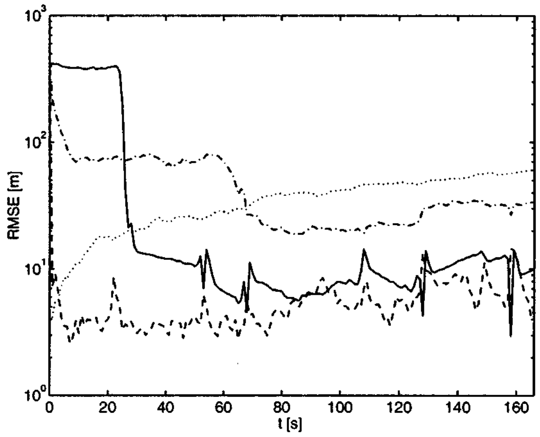

Fig. 5. RF positioning: Monte Carlo performance over time in the simulated scenario. The map matching (solid) needs some 25 s to converge, but after this burn-in time, the algorithm provides RMSE $= 8.7$ m. This is almost as good as with ideal distance measurements to two base stations (dashed) with RMSE $=$ 7.0 m. For comparison, power measurements (dash-dotted) yield RMSE $= 36$ m and dead reckoning (dotted) a steadily increasing error with RMSE $= 50$ m.

Based on these lists, the network centrally transfers connections from one base station to another (hand-over) when the mobile is moving during the service session. According to the empirical model by Okumura–Hata [26], this provides M measurement equations as in (3e), one for each available base station (in this simulation, $M = 2$), and $p_e(e) \in \mathrm{N}(0, \sigma_e^2)$, where $\sigma_e = 6$ dB. Similar measurements, but with a different motion model (the velocity is unknown), are used in [28]. Point-mass implementation of estimators based on RF measurements is also discussed in [9].

To provide more accurate positioning via RF measurements, future mobile stations will be able to estimate the traveled distance of radio signals from a multitude of base stations. In the ideal case, the signals have traveled without reflections to the mobile station (line-of-sight situation), and the estimates describe the distance to the base stations. The M (M is typically 1–3) measurement equations can thus be modeled by (3c), and they represent a rather ideal situation. Moreover, the noise is modeled as $p_e(e) \in \mathrm{N}(0, \sigma_e^2)$, where $\sigma_e = 3$ dB. The received power measurements discussed above are available today but are of worse accuracy due to unmodeled power variations.

A third alternative is to simply integrate the relative movements provided by the ABS (dead-reckoning). Monte Carlo simulations based on these different approaches are summarized in Fig. 5. It is interesting to note that map matching provides a position accuracy of roughly the same accuracy as with accurate distance measurements (which would almost never be the case in a real situation), without relying on external signals. Furthermore, integrating the ABS signals directly yields an increasing error over time.

V. Terrain Elevation Matching

The air fighter JAS 39 Gripen is equipped with an accurate radar altimeter and a digital map. This corresponds to the measurement equation characterized by $h_h(p^{(1)})$ in Section II-B4. The velocity vector is obtained by integrating the acceleration provided by the inertial navigation system. Since $v_t^{(1)}$ is available as an input signal, the motion model in (2a) is appropriate.

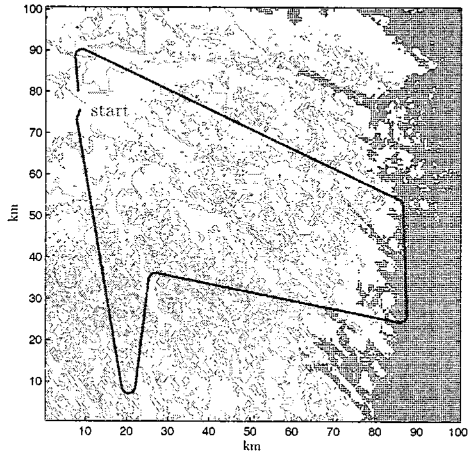

Fig. 6. Terrain navigation: Test track over a part of southeastern Sweden.

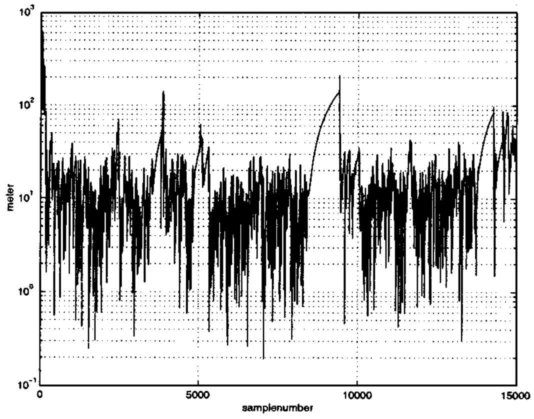

Fig. 7. Terrain navigation: Estimation error relative a GPS reference, as a function of sample number. Note the growth in error over open water.

The particle filter has been applied to a number of flight tests on the fighter JAS 39 Gripen, and Fig. 6 shows the path in one of them. In these tests, differential GPS (DGPS) is taken as the true position, and the resulting position error is shown in Fig. 7. The accuracy beats the first-generation system and comes down to the value of the point mass filter described in [8]. Since the point mass filter satisfies the Cramér–Rao lower bound (see [6]), there is no better filter. The advantage of the particle filter over the point mass filter is first a much less complex algorithm occupying only some 30 lines of code (Ada) and, second, the possibility to extend the functionality by including other parameters such as barometric height offset in the state vector (that is, increasing the particle dimension). Saab has evaluated the deterministic particle filter in Gripen in parallel with the first generation system with superior results, whereas the particle filter described herein, so far, is run offline.

VI. Integrated Navigation Systems

As a simplified study to illustrate the Rao–Blackwellization procedure, a 2-D navigation model with six states is used according to (2b), and the measurement of position is taken from

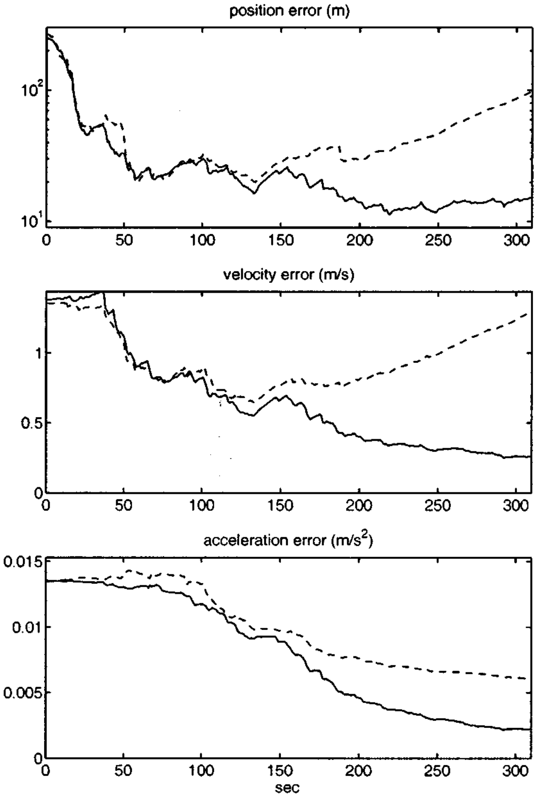

Fig. 8. RMSE based on 100 Monte Carlo simulations for the particle filter using 60 000 particles (dashed lines) and the Rao–Blackwellized filter using 4000 particles (solid lines).

the terrain navigation algorithm according to Section II-B4. It should be noted that the 2-D navigation model is valid only when the earth is modeled as flat. As soon as one accounts for the curvature of the earth, the model becomes more complicated; see [10]. In practice, there also exist gyro sensor errors that further complicate the problem.

In Fig. 8, the result is shown for the particle filter when using $N = 60\,000$ particles (dashed lines). The performance is pretty bad, and it quickly deteriorates even more when the number of particles is decreased. In particular, the transient requires many particles. The basic problem is high dimensionality and small process noise. On the other hand, following the Rao–Blackwellization procedure, we partition the state vector and rewrite the motion model according to (5) with

$$x_t^{\mathrm{pf}} = p_t^{(1)}, \quad x_t^{\mathrm{kf}} = \begin{pmatrix} v_t^{(1)} \\ \delta a_t^{(1)} \end{pmatrix}.$$

The result from applying this Rao–Blackwellized filter using only $N = 4000$ particles is also shown in Fig. 8 (solid lines), and the performance enhancement is significant.

VII. Target Tracking

The standard approach to target tracking is based on (extended) Kalman filters [3], [42]. Bearings-only target tracking was introduced as the illustration of particle filters in [21]. Since then, bearings-only target tracking has been used in many investigations; see, for instance, several of the chapters in [13]. A more realistic scenario is investigated in [31]. Here, the case of radar measurements where range is also available is discussed, which occurs in different applications, such as air traffic control (ATC) and collision avoidance. Linear models such as (1) can often be used, but nonlinear state equations are also used. For instance, when the tracking object is moving in straight paths or on circular segments, different variations of the so-called coordinated turn model [4] can be utilized. For maneuvering targets, multiple models are used to enhance tracking performance. The interacting multiple model (IMM) [4] is one classical multiple model algorithm based on the interaction of several extended Kalman filters [1]. Hard constraints on system states, such as velocity and acceleration boundaries or obstacles from the terrain, may introduce nonlinearities in many applications, which could degrade performance if not handled by the tracking filter. Two different applications will be presented in more detail below. It is important to note that realistic measurements (3g) can easily be used, modeling the radar loob in the angle noise distribution, and (3c), with uniform range noise distribution.

TABLE IV
Target Tracking: RMSE Comparison for ATC Monte Carlo Simulations

	APF	Bootstrap	IMM-3	Measurements
RMSE	34.03	40.84	42.20	63.96

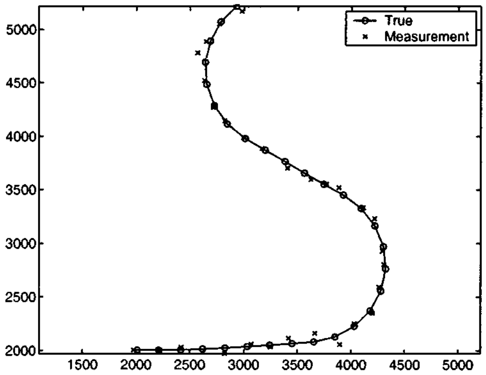

Fig. 9. Target tracking: Trajectory for the 100 Monte Carlo simulations, 800 particles.

An important aspect of target tracking is *association* [3], [42]: To which object should a certain measurement be associated? This is a discrete problem, and attempts to include this in a particle filter framework are described in [2], [7], [18], [20], [27], [32], [38], and [41].

A. Air Traffic Control (ATC)

In [30], a simple nearly coordinated turn model [4] was used for an ATC radar application. In the simulation study presented in Table IV, two different simulation-based methods are compared to the state-of-the-art IMM method. The particle filter algorithms tested are the Bayesian bootstrap method (3a) and APF [37]. The particle filters are here extended to the multiple model case, where target maneuvers are according to a Markov chain. Three different turn assumptions were made (right/left turn or straight flying) in the simulations presented. The true path projected in the horizontal plane is viewed in Fig. 9. It was generated with a true turn rate value chosen as an intermediate value

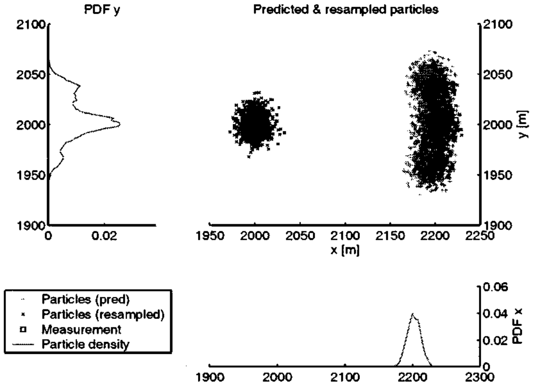

Fig. 10. Target tracking: Particle cloud and density.

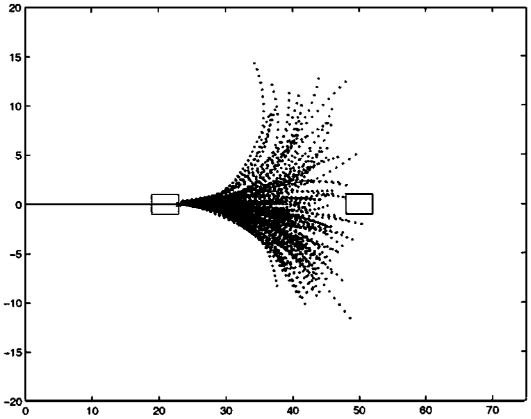

Fig. 11. Collision avoidance: The left rectangle is the own car, which is approaching rapidly the right rectangle. The trajectories indicate 31-step ahead prediction using 100 particles. There are still possible trajectories avoiding collision, of which the driver will most probably choose one. Thus, no active control is needed at this stage.

of the turn rate used in the multiple model conditioning, thus allowing the IMM to mix between models, and the particle filter process noise to perform the turn interaction. The incorporation of hard constraints on the velocity is also straightforward for the particle filter case. The radar sensor used in the application measures range, azimuth, and elevation at a rather low update rate to emulate a track-while-scan (TWS) behavior. In Table IV, the IMM method is compared with the particle filters and measurements only, viewing the position RMSE for 100 Monte Carlo simulations. For the Bayesian bootstrap case, two simulations diverged. Depending on the choice of process noise, the slight difference between the IMM and the Bayesian bootstrap may change. The marginalized density is also shown in Fig. 10 together with the particle cloud.

B. Collision Avoidance

The coordinated turn model can be used for collision avoidance to track the car position and predict future positions. The goal of the prediction in this case is not necessarily to get as good a point estimate as possible. Instead, we are interested in the whole distribution of possible maneuvers. Fig. 11 shows a simulation where the collision is still avoidable. This would not be obvious from just looking at the point estimate.

The main contributions to the process noise come from the driver's action via steering wheel, gas, and brake. A lot of effort has to be spent on how to choose the process noise so that it corresponds to the driver's behavior and the physical limitations of the car. The vehicle and driver behaviors change significantly for different speeds of the vehicle. Thus, in order to get a good prediction with this model, it is necessary to let the process noise f_t change with different speeds. It is also important in this application to incorporate knowledge about the environment to improve the prediction. For example, it is likely that the car will travel on the road and if there are some hard boundaries like rails or other stationary objects. These are hard constraints on the car's movement.

VIII. Conclusions and Discussion

We have given a general framework for positioning and navigation applications based on a flexible state space model and a particle filter. Five applications illustrate its use in practice. Evaluations in real-time, off-line, on real data and in simulation environments show a clear improvement in performance compared with existing Kalman filter-based solutions, where the new challenge is to find nonlinear relations, state constraints, and non-Gaussian sensor models that provide the most information about the position. Thus, *modeling* is the most essential step in this approach, compared with the various *implementations* of the Kalman filter found in this context (linearization issues, choice of state coordinates, filterbanks, Gaussian sum filters, etc.).

General conclusions from the implementations are as follows: A choice of state coordinates making the state equation linear is beneficial for computation time and opens up the possibility for Rao–Blackwellization. This procedure enables a significant decrease in the particle state dimension. The evaluation of the likelihood one step ahead before resampling (APF, prior editing) is, together with adding extra state noise (jittering, roughening), crucial for avoiding divergence and implies that the number of particles can be decreased further. Our implementations run in real time (2 Hz), even in Matlab, and have some 2000 particles.

Open questions for further research and development are as follows.

- *Divergence Tests*: It is essential to have a reliable way to detect divergence and to restart the filter (for the latter, see the transient below). For car positioning, the number of resamplings in the prior editing step turned out to be a very good indicator of divergence. Another idea used in the terrain navigation implementation where the sampling rate is higher than necessary, is to split up the measurements to a filterbank so that particle filter number i, $i = 1, 2, \ldots, n$ gets every nth sample. The result of these n particle filters are approximately independent, and voting can be used to restart each filter. This has turned out to be an efficient way to remove the outliers in data.
- *Transient Improvement*: The time it takes until the estimate accuracy comes down to the stationary value (the Cramér–Rao bound) depends on the number of particles. Given limited computational time, it may be advantageous

to increase the number of particles N after a restart and discard samples in such a way that N/T_s is constant.

- Since the particle filter has shown good improvement over linearization approaches, it is tempting to try even more accurate nonlinear models. In particular, the flight dynamics of one's own vehicle is known and, indeed, used in model-based control but is very rare in navigation applications; see [33] for one attempt in this direction. In that study, it seems that the computational burden and linearization errors imply no gain in total performance. As a possible improvement, the particle filter may take full advantage of a more accurate model, where parts of the nonlinear dynamics from driver/pilot inputs are incorporated.

APPENDIX

For the derivation of the Rao–Blackwellized algorithm given in Section III-D, suppose first that the particle filter part of the state vector is known. That is, the sequence $X_t^{\rm pf} = \{x_0^{\rm pf}, \ldots, x_t^{\rm pf}\}$ is known. We can, temporally, consider $z_t = x_{t+1}^{\rm pf} - x_t^{\rm pf}$ as the measurement. The state space model here is

$$\begin{aligned} x_{t+1}^{\rm kf} &= A^{\rm kf} x_t^{\rm kf} + B_u^{\rm kf} u_t + B_f^{\rm kf} f_t \\ z_t &= A^{\rm pf} x_t^{\rm kf} + B_u^{\rm pf} u_t + B_f^{\rm pf} f_t. \end{aligned}$$

Since this model is linear and Gaussian, the optimal solution is provided by the Kalman filter. We then know that $p(x_t^{\rm kf}|X_t^{\rm pf})$ is Gaussian; therefore

$$x_t^{\rm kf} \left| X_t^{\rm pf} = x_t^{\rm kf} \right| Z_{t-1} \sim N\left(\hat{x}_{t|t-1}^{\rm kf}, P_{t|t-1}^{\rm kf}\right)$$

where $\hat{x}_{t|t-1}^{kf}$ and $P_{t|t-1}^{kf}$ are given by the Kalman filter equations adjusted for correlated noise [24]

$$\begin{aligned} K_t &= P_{t|t-1}^{\rm kf}(A^{\rm pf})^T \left(A^{\rm pf} P_{t|t-1}^{\rm kf} (A^{\rm pf})^T + B_f^{\rm pf} Q_t \left(B_f^{\rm pf}\right)^T\right)^{-1} \\ \hat{x}_{t+1|t}^{\rm kf} &= \overline{A}^{\rm kf} \left(\hat{x}_{t|t-1}^{\rm kf} + K_t \left(z_t - A^{\rm pf} \hat{x}_{t|t-1}^{\rm kf}\right)\right) \\ &\quad + B_u^{\rm kf} u_t + B_f^{\rm kf} \left(B_f^{\rm pf}\right)^{\dagger} z_t \\ P_{t+1|t}^{\rm kf} &= \overline{A}^{\rm kf} \left(P_{t|t-1}^{\rm kf} - K_t A^{\rm pf} P_{t|t-1}^{\rm kf}\right) \left(\overline{A}^{\rm kf}\right)^T \end{aligned}$$

where $\overline{A}^{\rm kf} = A^{\rm kf} - B_f^{\rm kf}(B_f^{\rm pf})^{\dagger} A^{\rm pf}$ († denotes the Moore–Penrose pseudo-inverse).

Now, to compute $p(x_t|Y_t) = p(x_t^{\rm pf}, x_t^{\rm kf}|Y_t)$, note that

$$p\left(X_t^{\rm pf}, x_t^{\rm kf} \middle| Y_t\right) = p\left(x_t^{\rm kf} \middle| X_t^{\rm pf}\right) p\left(X_t^{\rm pf} \middle| Y_t\right).$$

We only have to compute $p(X_t^{\rm pf}|Y_t)$. Repeated use of Bayes' rule gives

$$p\left(X_t^{\rm pf} \middle| Y_t\right) = \frac{p\left(y_t|x_t^{\rm pf}\right) p\left(x_t^{\rm pf} \middle| X_{t-1}^{\rm pf}\right)}{p(y_t|Y_{t-1})} p\left(X_{t-1}^{\rm pf} \middle| Y_{t-1}\right).$$

We have a nonlinear and non-Gaussian measurement equation; therefore, to solve the measurement update, the particle filter will be used to approximate this distribution. The particle predictions $p(x_{t+1}^{\rm pf}|X_t^{\rm pf})$ are given by

$$x_{t+1}^{\rm pf} \left| X_t^{\rm pf} = x_t^{\rm pf} + A^{\rm pf} x_t^{\rm kf} \right| X_t^{\rm pf} + B_u^{\rm pf} u_t + B_f^{\rm pf} f_t$$

so that $p(x_{t+1}^{\rm pf}|X_t^{\rm pf})$ is given by

$$N\left(x_t^{\rm pf} + A^{\rm pf} \hat{x}_{t|t-1}^{\rm kf} + B_u^{\rm pf} u_t \right. \\ \left. A^{\rm pf} P_{t|t-1}^{\rm kf} (A^{\rm pf})^T + B_f^{\rm pf} Q_t \left(B_f^{\rm pf}\right)^T\right).$$

Finally, note that the derivation does not change if we use the fictitious measurement $z_t = x_{t+1}^{\rm pf} - g(x_t^{\rm pf})$ for an arbitrary nonlinear function, which is Remark 3.

ACKNOWLEDGMENT

The authors would like to thank the competence center ISIS at Linköping University, which has brought all of the authors together. The authors are very grateful to C. Andrieu and A. Doucet for their fruitful discussions on the theoretical subjects. They want to acknowledge their gratitude to the masters students M. Ahlström and M. Calais, who have implemented the terrain navigation filter, P. Hall, who implemented the car positioning filter, and the supporting companies SAAB Dynamics and NIRA Dynamics.

REFERENCES

[1] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Englewood Cliffs, NJ: Prentice-Hall, 1979.

[2] D. Avitzour, "Stochastic simulation Bayesian approach to multitarget tracking," *Proc. Inst. Elect. Eng., Radar, Sonar Navigat.*, vol. 142, no. 2, 1995.

[3] Y. Bar-Shalom and T. Fortmann, "Tracking and data association," in *Mathematics in Science and Engineering*. New York: Academic, 1988, vol. 179.

[4] Y. Bar-Shalom and X. R. Li, *Estimation and Tracking: Principles, Techniques, and Software*. Norwell, MA: Artech House, 1993.

[5] N. Bergman, "Recursive Bayesian estimation: Navigation and tracking applications," Dissertation 579, Linköping Univ., Linköping, Sweden, 1999.

[6] ——, "Posterior Cramér–Rao bounds for sequential estimation," in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds. New York: Springer-Verlag, 2001.

[7] N. Bergman and A. Doucet, "Markov chain Monte Carlo data association for target tracking," in *Proc. IEEE Conf. Acoust., Speech, Signal Process.*, 2000.

[8] N. Bergman, L. Ljung, and F. Gustafsson, "Terrain navigation using Bayesian statistics," *IEEE Contr. Syst. Mag.*, vol. 19, no. 3, pp. 33–40, 1999.

[9] J. Blom, F. Gunnarsson, and F. Gustafsson, "Estimation in cellular radio systems," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Phoenix, AZ, Mar. 1999.

[10] K. R. Britting, *Inertial Navigation Systems Analysis*. New York: Wiley-Interscience, 1971.

[11] D. Crisan and A. Doucet, "Convergence of sequential Monte Carlo methods," Signal Process. Group, Dept. Eng., Univ. Cambridge, Cambridge, U.K., Tech. Rep. CUED/F-INFENG/TR381, 2000.

[12] A. Doucet and C. Andrieu, "Particle filtering for partially observed Gaussian state space models," Dept. Eng., Univ. Cambridge, Cambridge, U.K., Tech. Rep. CUED/F-INFENG/TR393, Sept. 2000.

[13] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag, 2001.

[14] A. Doucet, S. J. Godsill, and C. Andrieu, "On sequential simulation-based methods for Bayesian filtering," *Statist. Comput.*, vol. 10, no. 3, pp. 197–208, 2000.

[15] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Trans. Signal Processing*, vol. 49, pp. 613–624, Mar. 2001.

[16] C. Drane, M. Macnaughtan, and C. Scott, “Positioning GSM telephones,” *IEEE Commun. Mag.*, vol. 36, July 1998.

[17] P. Fearnhead, “Sequential Monte Carlo methods in filter theory,” Ph.D. dissertation, Univ. Oxford, Oxford, U.K., 1998.

[18] S. Geman and D. Geman, “Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 721–741, 1984.

[19] W. Gilks, S. Richardson, and D. Spiegelhalter, *Markov Chain Monte Carlo in Practice*. London, U.K.: Chapman & Hall, 1996.

[20] N. J. Gordon, “A hybrid bootstrap filter for target tracking in clutter,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 33, pp. 353–358, Apr. 1997.

[21] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, “A novel approach to nonlinear/non-Gaussian Bayesian state estimation,” *Proc. Inst. Elect. Eng., Radar Signal Process.*, vol. 140, pp. 107–113, 1993.

[22] F. Gustafsson, S. Ahlqvist, U. Forssell, and N. Persson, “Sensor fusion for accurate computation of yaw rate and absolute velocity,” in *Proc. SAE*, Detroit, MI, 2001.

[23] F. Gustafsson, U. Forssell, and P. Hall, “Car positioning system,” Swedish patent application SE0 004 096-4, 2000.

[24] F. Gustafsson, *Adaptive Filtering and Change Detection*. New York: Wiley, 2000.

[25] P. Hall, “A Bayesian approach to map-aided vehicle positioning,” Master thesis LiTH-ISY-EX-3104 (in Swedish), Dept. Elect. Eng. Linköping Univt, Linköping, Sweden, 2001.

[26] M. Hata, “Empirical formula for propagation loss in land mobile radio services,” *IEEE Trans. Veh. Technol.* , vol. 29, May 1980.

[27] C. Hue, J. P. Le Cadre, and P. Pérez, “Tracking multiple objects with particle filtering,”, Tech. Rep. IRISA, 1361, Oct. 2000.

[28] H. Jwa, S. Kim, X. Cho, and J. Chun, “Position tracking of mobiles in a cellular radio network using the constrained bootstrap filter,” in *Proc. Nat. Aerosp. Electron. Conf.*, Dayton, OH, Oct. 2000.

[29] T. Kailath, A. H. Sayed, and B. Hassibi, “Linear estimation,” in *Information and System Sciences*. Upper Saddle River, NJ: Prentice-Hall, 2000.

[30] R. Karlsson and N. Bergman, “Auxiliary particle filters for tracking a maneuvering target,” in *Proc. IEEE Conf. Decision Contr.*, Sydney, Australia, Dec. 2000.

[31] R. Karlsson and F. Gustafsson, “Range estimation using angle-only target tracking with particle filters,” in *Proc. Amer. Contr. Conf.*, Arlington, VA, June 2001.

[32] ——, “Monte Carlo data association for multiple target tracking,” in *Proc. IEE Target Tracking: Algorithms Appl.*, Amsterdam, The Netherlands, Oct. 2001.

[33] M. Koifman and I. Y. Bar-Itzhack, “Inertial navigation system aided by aircraft dynamics,” *IEEE Trans. Contr. Syst. Technol.*, vol. 7, pp. 487–493, Sept. 1999.

[34] A. Kong, J. S. Liu, and W. H. Wong, “Sequential imputations and Bayesian missing data problems,” *J. Amer. Stat. Assoc.*, vol. 89, no. 425, pp. 278–288, 1994.

[35] J. S. Liu, “Metropolized independent sampling with comparison to rejection ampling and importance sampling,” *Statist. Comput.*, vol. 6, pp. 113–119, 1996.

[36] P.-J. Nordlund and F. Gustafsson, “Sequential Monte Carlo filtering techniques applied to integrated navigation systems,” in *Proc. Amer. Contr. Conf.*, Arlington, VA, June 2001.

[37] M. K. Pitt and N. Shephard, “Filtering via simulation: Auxiliary particle filters,” *J. Amer. Statist. Assoc.*, vol. 94, no. 446, pp. 590–599, June 1999.

[38] C. Rago, P. Willett, and R. Streit, “A comparison of the JPDAF and PMHT tracking algorithms,” in *Proc. IEEE Conf. Acoust., Speech, Signal Process.*, vol. 5, 1995, pp. 3571–3574.

[39] B. D. Ripley, *Stochastic Simulation*. New York: Wiley, 1988.

[40] S. Rohr, R. Lind, R. Myers, W. Bauson, W. Kosiak, and H. Yen, “An integrated approach to automotive safety systems,” *Autom. Eng. Int.*, Sept. 2000.

[41] D. J. Salmond, D. Fisher, and N. J. Gordon, “Tracking in the presence of intermittent spurious objects and clutter,” in *Proc. SPIE Conf. Signal Data Process. Small Targets*, 1998.

[42] S. S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Norwood, MA: Artech House, 1999.

[43] S. Thrun, D. Fox, F. Dellaert, and W. Burgard, “Particle filters for mobile robot localization,” in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds. New York: Springer-Verlag, 2001.

[44] O. Wijk, “Triangulation based fusion of sonar data with application in mobile robot mapping and localization,” Ph.D. dissertation, Roy. Inst. Technol., Linköping, Sweden, 2001.

Fredrik Gustafsson received the M.S. degree in electrical engineering in 1988 and the Ph.D. degree in automatic control in 1992, both from Linköping University, Linköping, Sweden.

He is Professor of communication systems with the Department of Electrical Engineering, Linköping University. His research is focused on statistical methods in signal processing, with applications to automotive, avionic, and communication systems.

Dr. Gustafsson is an associate editor of IEEE TRANSACTIONS ON SIGNAL PROCESSING.

Fredrik Gunnarsson received the M.Sc. degree in applied physics and electrical engineering in 1996 and the Ph.D. degree in electrical engineering in 2000, both from Linköping University, Linköping, Sweden.

He is a research associate with Communications Systems, Department of Electrical Engineering, Linköping University. His research interests include control and signal processing in terrestrial wireless communications systems and automotive applications.

Niclas Bergman received the M.Sc. degree in applied physics and electrical engineering in 1995 and the Ph.D. degree in electrical engineering in 1999, both from Linköping University, Linköping Sweden.

He is with SaabTech Systems, Järfälla, Sweden. He is currently working with research and development in the areas of target tracking and navigation and is responsible for the coordination of data fusion activities within the Saab Group.

Urban Forssell received the M.Sc. degree in applied physics and electrical engineering in 1995 and the Ph.D. degree in automatic control in 1999, both from Linköping University, Linköping, Sweden.

He is President and CEO of NIRA Dynamics AB, Linköping. The company focuses on advanced signal processing and control in vehicles.

Jonas Jansson has, since 1999, been pursuing the Ph.D. degree at Linköping University, Linköping, Sweden.

He is with Volvo Car Corporation, Gothenberg, Sweden, developing a collision avoidance system. He His current research interests focus on particle filter implementation of navigation and tracking systems.

Rickard Karlsson has, since 1999, been pursuing the Ph.D. degree at Linköping University, Linköping, Sweden.

He has worked at SAAB Dynamics, Linköping, in target tracking and sensor fusion since 1997. His current research interests focus on particle filter implementation of target tracking algorithms with radar and/or IR sensors.

Per-Johan Nordlund has, since 1999, been pursuing the Ph.D. degree at Linköping University, Linköping, Sweden.

He has worked at SAAB Gripen, Linköping, while developing a new version of the navigation system for the fighter JAS 39 Gripen. His current research interests focus around particle filter implementation of integrated navigation systems with particular attention to complexity aspects and fault detection.