Target Tracking With Particle Filters Under Signal Propagation Delays

Umut Orguner and Fredrik Gustafsson

Umut Orguner and Fredrik Gustafsson, Target Tracking With Particle Filters Under Signal Propagation Delays, 2011, IEEE TRANSACTIONS ON SIGNAL PROCESSING, (59), 6, 2485-2495.
http://dx.doi.org/10.1109/TSP.2011.2122260

Postprint available at: Linköping University Electronic Press
http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-68818

Target Tracking with Particle Filters under Signal Propagation Delays

Umut Orguner, *Member, IEEE,* and Fredrik Gustafsson, *Senior Member, IEEE*

***Abstract*—Signal propagation delays are hardly a problem for target tracking with standard sensors such as radar and vision due to the fact that the speed of light is much higher than the speed of the target. This contribution studies the case where the ratio of the target and the propagation speed is not negligible, as in the case of sensor networks with microphones, geophones or sonars for instance, where the signal speed in air, ground and water causes a state dependent and stochastic delay of the observations. The proposed approach utilizes an augmentation of the state vector with the propagation delay in a particle filtering framework to compensate for the negative effects of the delays. The model of the physics rules governing the propagation delays is used in interaction with the target motion model to yield an iterative prediction update step in the particle filter which is called the *propagation delayed measurement particle filter* (PDM-PF). The performance of PDM-PF is illustrated in a challenging target tracking scenario by making comparisons to alternative particle filters that can be used in similar cases.**

***Index Terms*—Propagation delay, state estimation, target tracking, constrained estimation, implicit constraints, stochastic sampling, sequential Monte Carlo, particle filter.**

I. Introduction

THE conventional sensors such as radar, vision (EOR, IR) etc. used in target tracking [1–3] generally observe emitted (passive sensors) or reflected (active sensors) energy from the target. The observation delay in these sensors is negligible since the speed of light (electromagnetic waves) is much larger than the speed of the target. However, one trend in sensor networks is to use standard low cost sensors as microphones and geophones on land, and sonar in water. The assumption of negligible target speed compared to the speed of the media cannot always be made here. In a general scenario where the target moves swiftly, even if the sensor is stationary and collects uniformly sampled (in time) measurements of the target, the actual time instants that the target is observed are in fact non-uniform (in time) due to the propagation delays and this leads to unexpected errors in the estimation algorithms.

The physics rules governing the propagation delays are generally well known and hence a model for the propagation delays is usually available for target tracking applications. However, since the actual target state at the delayed time instant is also uncertain, the propagation delay model must be used in interaction with the target state model to form an implicit equation characterizing the propagation delay (which we call "the implicit delay constraint" or simply "the delay constraint" below) which is difficult to handle. This type of problem has been investigated in a novel framework in the authors' series of earlier work [4–6] and the current work can be considered as an improved version of [6]. The solution presented originally in [4] is a Bayesian algorithm that adds the involved propagation delay into the state vector of the target. The idea of using the delay constraint as an information source was hence first proposed in [4] which utilizes an iterative procedure in a deterministic sampling (such as unscented Kalman filter (UKF) [7]) based framework. Later, the single sensor algorithm presented in [4] was generalized to multiple sensors in [5] using the largest ellipsoid algorithm (LEA) [8, 9] in a distributed scenario.

The studies [4] and [5] both used UKF-type algorithms (which was called *propagation delayed measurement* (PDM) *filter* (or PDMF) in [5]) to include the information of the implicit delay constraint into the estimation process, which limits their applicability to only linear or slightly nonlinear models with reasonably high signal to noise ratio. In fact, it was empirically observed already in [6] that PDMF of [4] based on UKF is sensitive to a high level of measurement noise. The work in this paper is instead concerned with using the implicit delay constraints along with particle filters (PFs) [10–12] enabling the use of almost any nonlinear model in the estimation even in low signal to noise ratio environments.

As mentioned above, an earlier version of this work was presented in [6] and the current work presents a more general convergence proof and simple specific sufficient statistics for (the main recursion involved in) the algorithm along with improved simulation results. We use the same Bayesian methodology as [4–6] to include the time delays into the state vector of the target. Then, each particle in the particle filter keeps a different hypothesis about the propagation delay together with its corresponding kinematic state vector. The implicit constraint information is incorporated into the particles in the prediction update using recursions that combine the physics rules governing the propagation delays and the target state dynamics. The measurement update of the particle filter is a standard one. Due to the fact that the time stamp of the kinematic state component of the augmented state is stochastic, an additional non-conventional prediction step has to be utilized for the output purposes. In this paper, for the sake of simplicity, we consider only the single sensor case leaving the multiple sensor generalization as a future work.

Consideration of implicit delay constraints in the estimation process makes this work highly related to the area of

U. Orguner and F. Gustafsson are with the Division of Automatic Control, Department of Electrical Engineering, Linköping University, Linköping, Sweden (e-mail: {umut,fredrik}@isy.liu.se).

Manuscript received August 30, 2010.

constrained state estimation. The existing solutions which can handle equality constraints in the estimation cycles consist of two main approaches. One and possibly the more popular of these is the use of constraints as fictitious (pseudo) measurements where the constraint equation is considered as a (possibly noisy) information source about the state [13–15]. A conceptually similar method which uses the so called three block Kalman filter of [16] is given in [17]. The second approach of handling constraints is the projection type methods in which the unconstrained estimates are projected onto the constraint surface (manifold) at the end of each estimation cycle [18]. Another more useful form of projection based method has been proposed in [19]. In [20], a comparison of these two types of algorithms is presented along with some cases which yield equivalent results. A conceptually different method which can be related to both approaches is to search for optimal filter gains that yield estimates satisfying the constraints. An application of this is presented for estimation problems using quaternions (vectors with unity norm) in [21].

This study, although being closer to projection type approaches, is different from these existing methodology in two aspects. The first one is that implicit constraints can be handled with our methodology whereas the existing ones are more suitable for explicit constraints. The second and the more important aspect that differentiates our study from the existing ones is that we consider the inclusion of the constraints in the prediction update of the estimation process. This alleviates the performance reducing effects of unconstrained prediction. In order to observe this, we make a comparison of our method with other particle filters that compensate for the propagation delays in output calculation or measurement update steps (both of which are quite late in the estimation cycle) in Section V.

Although this work can be counted as a continuation of the previous conference papers [4–6], it is self-contained. The remaining parts are organized as follows. We illustrate the state estimation problem involved in this paper via a simplified example in Section II. Section III makes a formal problem definition. Section IV gives the main result of the paper, a PF algorithm that compensates for the propagation delay effects. We call the resulting algorithm as the *propagation delayed measurement particle filter* (PDM-PF). In Section V, we present the results of a simulation study on a single sensor target tracking scenario along with some comparisons to alternative particle filters that can be used with propagation delays. Conclusions are drawn in Section VI.

II. Simplified Example

Leaving the general and formal problem formulation to Section III, we here make a simplified introduction to the problem. Consider a case where we have perfect knowledge of the state vector $x_{t_{k-1}}$ of a target at time t_{k-1} and that the target position $\mathsf{p}_{t_{k-1}}$ is a subset of the state vector. One sensor shown as s in Figure 1 gets an observation related to $x_{t_k-\Delta_k}$ at time t_k. The unknown delay Δ_k can be described as a function of the unknown position (and hence the state) of the target at the (unknown) time $t_k-\Delta_k$ using the physics

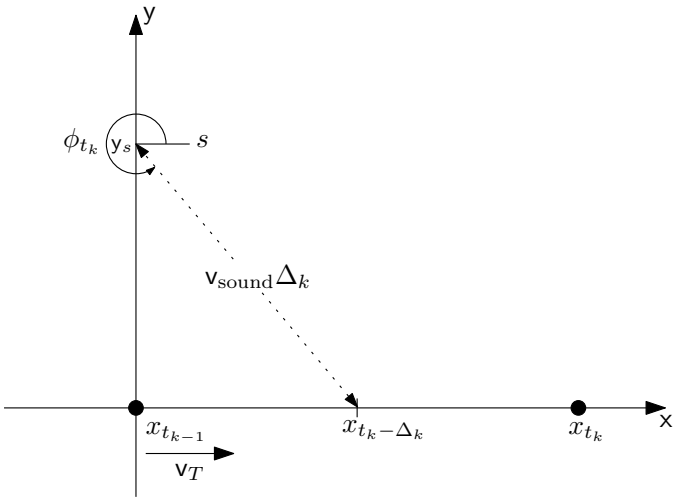

Fig. 1. Simplified example where a constant speed target on the x-axis is observed with a microphone array acquiring bearing information.

rules of signal propagation in the medium as

$$\Delta_k = d_{t_k}(x_{t_k-\Delta_k}). \tag{1}$$

On the other hand, using an assumed or known target dynamics, we can obtain a prediction of $x_{t_k-\Delta_k}$ from perfectly known $x_{t_{k-1}}$ as

$$x_{t_k-\Delta_k} = f_{t_k-\Delta_k,t_{k-1}}(x_{t_{k-1}}).$$

So the model does not store the previous knowledge. Instead, it uses the current knowledge to predict the previous knowledge .....

Consider, for instance, the case in Figure 1 wh
a target whose scalar state $x_t \triangleq \mathsf{p}_t$ is the po
axis. The target has the known state $x_0 = 0$ a
and moves with a known constant speed v_T along the x-axis. Notice that in a practical application, the velocity v_T would actually be unknown and a part of the state, however, in this example, we treat it as a known parameter in the scalar state dynamics for the sake of simplicity. At time t_k, a sound sensor s (microphone array) positioned on the y-axis value y_s collects the bearing ϕ_{t_k} of the target (corresponding to time $t_k-\Delta_k$). Then, the specific models corresponding to (1) and (2) would be

$$\Delta_k = d_{t_k}(x_{t_k-\Delta_k}) \triangleq \frac{1}{\mathsf{v}_{\text{sound}}}\sqrt{\mathsf{y}_s^2 + x_{t_k-\Delta_k}^2}, \tag{3}$$

$$x_{t_k-\Delta_k} = \underbrace{x_{t_{k-1}}}_{\triangleq 0} + \mathsf{v}_T(t_k-\Delta_k) = \mathsf{v}_T(t_k-\Delta_k) \tag{4}$$

respectively, where $\mathsf{v}_{\text{sound}}$ is the speed of sound. These two equations, substituting $x_{t_k-\Delta_k}$ of (4) into (3) and then squaring both sides of (3), have a solution $\Delta_k > 0$ satisfying

$$\mathsf{v}_T^2(t_k-\Delta_k)^2 + \mathsf{y}_s^2 = (\mathsf{v}_{\text{sound}}\Delta_k)^2 \tag{5}$$

as shown in Figure 1. Now, instead of solving the parabolic equation for Δ_k, one can define a recursion for Δ_k with the initial value e.g. $\Delta_k(0) = 0$ by just substituting (4) into (3) to get

$$\Delta_k(m+1) = \frac{1}{\mathsf{v}_{\text{sound}}}\sqrt{\mathsf{y}_s^2 + [\mathsf{v}_T(t_k-\Delta_k(m))]^2} \tag{6}$$

which can be shown to converge to the positive root of (5) if $\mathsf{v}_T < \mathsf{v}_{\text{sound}}$. The case $\mathsf{v}_T \geq \mathsf{v}_{\text{sound}}$ can also be handled by simply running the recursion (6) backwards but since this case causes difficulties in the general case, it will not be considered in this work.

Just as in the case of this simple example, in a more general setting, (1) and (2) together define an implicit equation for Δ_k (like (5)) whose solution can be obtained with iterative techniques similar to (6). This type of implicit and in general nonlinear constraints and their inclusion into the estimation process is the main subject of this work. In this simplified scenario, we have neglected three sources of uncertainty:

- The initial state $x_{t_{k-1}}$ is actually random.
- A process noise term must be added into the simplified description (2).
- The propagation time itself through (1) might be uncertain due to possible reasons such as the unmodeled non-line-of-sight (NLOS) effects or the uncertain (own) position of the sensor (y_s in the simplified example above).

In this work we propose a solution that covers all the uncertainties mentioned above for the single sensor case and is based on including the delay Δ_k in the state vector while processing the observation taken at time t_k. Since the measurements observe the delayed state values $x_{t_k-\Delta_k}$, the augmented state is formed as $[x_{t_k-\Delta_k}^{\mathrm{T}}, \Delta_k]^{\mathrm{T}}$. While making estimation, the form of the states $[x_{t_k-\Delta_k}^{\mathrm{T}}, \Delta_k]^{\mathrm{T}}$ forces one to design a special output estimate generation mechanism. This is because the state element $x_{t_k-\Delta_k}$ in the estimate has an ambiguous time stamp that depends on the Δ_k component of the state.

III. Problem Definition

We consider the following discrete-time nonlinear state space model

$$x_{t_{k+1}} = f_{t_{k+1},t_k}(x_{t_k}) + w_{t_{k+1},t_k} \tag{7}$$

where $\{x_{t_k} \in \mathbb{R}^{n_x}\}$ is the state sequence with initial distribution $x_{t_0} \sim p_0(x_{t_0})$. We here adopt an implicit simplified notation such that the system state dynamics given by (7) is a discretized version of a corresponding continuous time dynamics

$$\dot{x}_t = f(x_t) + w_t. \tag{8}$$

In (7), $t_k \in \mathbb{R}$ is an arbitrary time value and $f_{t_{k+1},t_k}(\cdot)$ is the state transition function transforming x_{t_k} to $x_{t_{k+1}}$ according to continuous time dynamics $f(\cdot)$. We also assume that the time sequence $\{t_k\}_{k=0}^{\infty}$ is non-decreasing and therefore the transformation involved in (7) is not necessarily invertible. $\{w_{t_{k+1},t_k} \in \mathbb{R}^{n_x}\}$ is a white process noise sequence with distribution $w_{t_k,t_{k-1}} \sim p^w_{t_k,t_{k-1}}(\cdot)$. Here it is important to emphasize that w_{t_{k+1},t_k} models the lumped effects of a continuous independent increment process noise w_t between the time instants t_k and t_{k+1}.

The discrete delayed measurements $\{y_k \in \mathbb{R}^{n_y}\}$ of this system are collected by a sensor s as

$$y_k = h_k(x_{t_k-\Delta_k}) + v_k \tag{9}$$

where $h_k(\cdot)$ is in general a nonlinear function of the state; Δ_k is the amount of delay in the measurement y_k and $\{v_k \in \mathbb{R}^{n_y}\}$ is a white measurement noise sequence independent from the process noise with distribution $v_k \sim p^{v_k}(\cdot)$. We here assume that we have knowledge about the time delay Δ_k in the form of an implicit equation (which we call c_k) as follows

$$c_k: \quad \Delta_k = d_{t_k}(x_{t_k-\Delta_k}) + \tau_k \tag{10}$$

where $d_{t_k}(\cdot)$ is in general a nonlinear function of the delayed state value $x_{t_k-\Delta_k}$ and $\{\tau_k \in \mathbb{R}\}$ is a white noise sequence independent from the process and measurement noise with distribution $\tau_k \sim p^{\tau_k}(\cdot)$. Our main motivation for selecting such an expression for the time delay sequence Δ_k is the case of a sound sensor whose delay expression is given as

$$\Delta_k = \frac{\|\mathsf{p}_{t_k-\Delta_k} - \mathsf{p}_{t_k}^{\text{sensor}}\|_2}{\mathsf{v}_{\text{sound}}} + \tau_k \tag{11}$$

where $\mathsf{p}_{t_k-\Delta_k}$ is the position of the target at time $t_k - \Delta_k$ (which is a function of the delayed state $x_{t_k-\Delta_k}$ and hence the form of (10)), $\mathsf{p}_{t_k}^{\text{sensor}}$ is the position of the sensor s at time t_k and $\mathsf{v}_{\text{sound}}$ is the speed of sound. The noise term τ_k then represents unmodeled effects in the transmission of the sound (pressure) wave in the environmental conditions such as NLOS and multipath effects or the possible uncertainty in the position of the sensor. In this work, we consider each constraint c_k of (10) as a piece of information to include into the estimation process and we show the cumulative information of constraints up to and including time n as $c_{0:n} \triangleq \{c_k\}_{k=0}^{n}$. Although the constraints themselves are not random variables, in the following, we are going to use them as the arguments of the probability density functions in given conditions and Bayes rules and these must be interpreted information-wise. Note that this type of notation is unconventional in the literature and here adopted for ease of notation. Throughout the study, we assume that all measurement acquisition times $\{t_k\}_{k=0}^{\infty}$ and auxiliary variables (like sensor positions etc.) used in the constraint evaluation (of course, other than the state component $x_{t_k-\Delta_k}$ in (10)) are known. Therefore, the constraints $c_{0:\infty}$ are known beforehand but their availability to the estimators is limited for the purpose of recursive estimation.

Problem Definition: Given the state dynamics (7) and measurement relation (9), find (possibly approximately) the density $p(x_{t_k}|y_{0:k}, c_{0:k})$.

IV. Propagation Delayed Measurement Particle Filter (PDM-PF)

This section is divided into three subsections the first of which gives the derivation of PDM-PF where a theorem plays the key role and ends with a summary of the main algorithm. The second subsection examines a special case of the theorem that facilitates the use in practical applications. The third subsection mentions shortly about possible alternative approaches.

A. PDM-PF Derivation

As mentioned in Section I, we keep the summary statistics of the propagation delayed measurement filter in terms of the augmented state vector $\xi_k \triangleq [x_{t_k-\Delta_k}^{\mathrm{T}}, \Delta_k]^{\mathrm{T}}$. There are two

merits and one difficulty involved with this. The two merits are

1) This selection makes the measurement update of the required filters quite easy.
2) Delay constraint information can be incorporated into the filter in a structured way.

On the other hand, there is one important difficulty in working with such an augmented vector and that is the fact that the state component of the augmented vector has a stochastic time stamp (whose uncertainty is determined by the delay components of the augmented state) that has to be taken care of before a meaningful output (an estimate that has a deterministic time stamp) can be produced.

Making this augmented state definition, a recursive Bayesian filter needs to calculate the posterior density $p(x_{t_k-\Delta_k}, \Delta_k | y_{0:k}, c_{0:k})$. In a particle filtering framework, we are going to approximate this posterior density shown equivalently as $p(\xi_k | y_{0:k}, c_{0:k})$ as follows

$$p(\xi_k | y_{0:k}, c_{0:k}) \approx \sum_{i=1}^{N} \pi_k^{(i)} \delta_{\xi_k^{(i)}}(\xi_k) \tag{12}$$

where $\{\xi_k^{(i)}\}_{i=1}^N$ is the set of particles; $\{\pi_k^{(i)}\}_{i=1}^N$ is the set of corresponding weights and the notation $\delta_\xi(\cdot)$ denotes the Dirac delta function positioned at ξ.

Having such a particle based density approximation at time $k-1$, we are going to define the classical updates in a Bayesian filter as follows.

- **Prediction Update:** Obtain the predicted density $p(\xi_k | y_{0:k-1}, c_{0:k})$ from the previous sufficient statistics $p(\xi_{k-1} | y_{0:k-1}, c_{0:k-1})$ by using the process model (7) and the current delay constraint information c_k (10).
- **Measurement Update:** Obtain the measurement updated density $p(\xi_k | y_{0:k}, c_{0:k})$ from the predicted density $p(\xi_k | y_{0:k-1}, c_{0:k})$ by using the current measurement y_k that has the model (9).

As can be easily seen above, once one has the predicted particles $\{\xi_{k|k-1}^{(i)}\}_{i=1}^N$ and corresponding weights $\{\pi_{k|k-1}^{(i)}\}_{i=1}^N$ that represent the predicted density $p(\xi_k | y_{0:k-1}, c_{0:k})$, obtaining the measurement updated density is a matter of computing the weights as

$$\pi_k^{(i)} \propto p(y_k | \xi_{k|k-1}^{(i)}) \pi_{k|k-1}^{(i)} \tag{13}$$

for $i = 1, \ldots, N$ which should be normalized to get $\sum_{i=1}^N \pi_k^{(i)} = 1$. The measurement updated particles are just set to the predicted particles, i.e., $\xi_k^{(i)} = \xi_{k|k-1}^{(i)}$. Note that the term $p(y_k | \xi_{k|k-1}^{(i)})$ in (13) is readily available because the measurement y_k is actually a direct function of ξ_k. This is a direct manifestation of the augmented state definition above on the measurement update.

In the prediction update, however, the process model and the constraint information have to be used at the same time to obtain the predicted density $p(x_{t_k-\Delta_k}, \Delta_k | y_{0:k-1}, c_{0:k})$ from the previous updated density $p(x_{t_{k-1}-\Delta_{k-1}}, \Delta_{k-1} | y_{0:k-1}, c_{0:k-1})$. There is actually a vicious circle inherent in this problem in that in order to predict $x_{t_k-\Delta_k}$ from $x_{t_{k-1}-\Delta_{k-1}}$ and Δ_{k-1} using the system model (7), we would need Δ_k; on the other hand, in order to calculate Δ_k from the delay constraint (10), we need $x_{t_k-\Delta_k}$. Such a circular dependence of augmented state elements is what makes the prediction update difficult. However, if we remember the recursion (6) that we set up in Section II, we can possibly use the system model (7) and the delay constraint (6) iteratively by starting with a reasonable initial condition to actually converge to a meaningful prediction solution. In other words, we can actually exploit the vicious circle described above to our advantage. The main building block of such an approach is provided by the following theorem whose earlier versions were first stated in [4–6]. The version in this work is the most general of the earlier versions and contains also a much more rigorous proof which does not need heuristic assumptions involved in the earlier versions. For the sake of keeping generality in the state variables, we first define the operators $\mathrm{pos}(\cdot)$, $\mathrm{vel}(\cdot)$ and $\mathrm{acc}(\cdot)$ of the target state x_t which give the Cartesian position, velocity and acceleration of the target in the Euclidean space at time t respectively. Expressed mathematically

$$\mathrm{pos}(x_t) \triangleq \begin{bmatrix} \mathsf{p}_t^{\mathsf{x}} \\ \mathsf{p}_t^{\mathsf{y}} \end{bmatrix} \quad \mathrm{vel}(x_t) \triangleq \begin{bmatrix} \mathsf{v}_t^{\mathsf{x}} \\ \mathsf{v}_t^{\mathsf{y}} \end{bmatrix} \quad \mathrm{acc}(x_t) \triangleq \begin{bmatrix} \mathsf{a}_t^{\mathsf{x}} \\ \mathsf{a}_t^{\mathsf{y}} \end{bmatrix} \tag{14}$$

where $\{\mathsf{p}_t^{\mathsf{x}}, \mathsf{p}_t^{\mathsf{y}}\}$, $\{\mathsf{v}_t^{\mathsf{x}}, \mathsf{v}_t^{\mathsf{y}}\}$ and $\{\mathsf{a}_t^{\mathsf{x}}, \mathsf{a}_t^{\mathsf{y}}\}$ are Cartesian position, velocity and acceleration variables of the target with state x_t defined over the 2D Euclidean $\mathsf{x} - \mathsf{y}$ plane. Notice that the state variable x_t might be defined in any other coordinates (like polar, spherical etc.) and the operators $\mathrm{pos}(\cdot)$, $\mathrm{vel}(\cdot)$ and $\mathrm{acc}(\cdot)$ represent the corresponding transformations from the state vectors to Cartesian position, velocity and acceleration spaces respectively. Extensions to 3D are straightforward. Note that the acceleration operator is not needed for the theorem but will be necessary for a corollary of it.

Theorem 1. *Let $d_{t_k}(\cdot)$ be a function of the target position $\mathsf{p}_{t_k-\Delta_k} \triangleq \mathrm{pos}(x_{t_k-\Delta_k})$ only and let it be a Lipschitz continuous function i.e.,*

$$|d_{t_k}(x) - d_{t_k}(x')| \leq K_d \| \mathrm{pos}(x) - \mathrm{pos}(x') \|_2 \tag{15}$$

for all x and x' where $K_d \geq 0$ is the corresponding Lipschitz constant. Let $\breve{x}, \breve{w} \in \mathbb{R}^{n_x}$ and $\breve{\Delta}, \tau \in \mathbb{R}$ be constants.

Consider the recursion[1]

$$\Delta_k(m+1) = d_{t_k}(x_{t_k-\Delta_k(m)}) + \tau \tag{16}$$

where $x_{t_k-\Delta_k(m)}$ is calculated using

$$x_{t_k-\Delta_k(m)} = f_{t_k-\Delta_k(m), t_{k-1}-\breve{\Delta}}(\breve{x}) + \breve{w}. \tag{17}$$

The initialization is done with

$$\Delta_k(0) = d_{t_k}(\breve{x}) + \tau. \tag{18}$$

The sequence $\Delta_k(m)$ converges exponentially to a fixed point $\Delta_k(\infty)$ satisfying

$$\Delta_k(\infty) = d_{t_k}(x_{t_k-\Delta_k(\infty)}) + \tau \tag{19}$$

[1] See Figure 2 for a Matlab® pseudo-code of the recursion.

```
function Δ=Recursion(x̆, Δ̆, w̆, τ, t_k, t_{k-1}, Threshold)
    N_max=100; %maximum number of iterations
    Δ=zeros(1, N_max); %delay array
    Δ(1)=d_{t_k}(x̆)+τ; %equation (18)
    m=1;
    difference=inf;
    while (difference>Threshold)&&(m<N_max)
        m=m+1;
        x=f_{t_k-Δ(m-1),t_{k-1}-Δ̆}(x̆)+w̆; %equation (17)
        Δ(m)=d_{t_k}(x)+τ; %equation (16)
        difference=Δ(m)-Δ(m-1);
    end
```

Fig. 2. A Matlab® pseudo-code of the recursion of Theorem 1. Notice that in order to be able to give an always stable function, we here included a constraint on the maximum number of iterations.

if $K_d \mathsf{v}_{\max}(t_{k-1}-\breve{\Delta}, t_k, \breve{x}) < 1$ *where the function* $\mathsf{v}_{\max}(\cdot,\cdot,\cdot)$ *is defined as*

$$\mathsf{v}_{\max}(t', t'', x) \triangleq \max_{t' \le t \le t''} \|\mathrm{vel}\,(f_{t,t'}(x))\|_2. \tag{20}$$

for all $x \in \mathbb{R}^{n_x}$*, and* $t', t'' \in \mathbb{R}$ *with* $t'' \ge t'$. □

Proof: Proof is given in Appendix A for the sake of clarity. □

In simple words, Theorem 1 states that finding the solution of the implicit delay constraint is possible using a recursion which converges to the solution exponentially. The condition for the convergence is dependent on the delay and state models along with the quantities $\breve{x}$ and $\breve{\Delta}$. The Lipschitz condition (15) on $d_{t_k}(\cdot)$ assumes that according to the propagation delay model, the propagation delay values for close target positions are similar which is natural and can be considered to be a fairly weak assumption. It is still important to note that this condition would be difficult to satisfy with multi-path and non-line-of-sight propagation models.

We are going to use Theorem 1 in updating the particles $\xi_{k-1}^{(i)} \triangleq [(x_{t_{k-1}-\Delta_{k-1}}^{(i)})^{\mathrm{T}}, \Delta_{k-1}^{(i)}]^{\mathrm{T}}$ to their predicted versions $\xi_{k|k-1}^{(i)} \triangleq [(x_{t_k-\Delta_k|k-1}^{(i)})^{\mathrm{T}}, \Delta_{k|k-1}^{(i)}]^{\mathrm{T}}$. For this purpose, one can initiate the recursion in Theorem 1 with the selection

$$\breve{x} = x_{t_{k-1}-\Delta_{k-1}}^{(i)} \qquad \breve{\Delta} = \Delta_{k-1}^{(i)} \qquad \breve{w} = w^{(i)} \qquad \tau = 0 \tag{21}$$

in the case that the noise term τ_k in (10) is identically zero i.e., $\tau_k \equiv 0$. Here, we sample $w^{(i)}$ from the density $p^w_{t_k,t_{k-1}}(\cdot)$, which is equivalent to making the approximation

$$p^w_{t_k-\Delta_k^{(i)},t_{k-1}-\Delta_{k-1}^{(i)}}(\cdot) \approx p^w_{t_k,t_{k-1}}(\cdot). \tag{22}$$

After running the recursion for several iterations (until a convergence criterion is satisfied), the resulting predicted delay $\Delta_{k|k-1}^{(i)}$ is selected as the converged fixed point $\Delta_k^{(i)}(\infty)$ and the predicted delayed state is found by

$$x_{t_k-\Delta_k|k-1}^{(i)} = f_{t_k-\Delta_k^{(i)}(\infty),t_{k-1}-\breve{\Delta}}(\breve{x}) + \breve{w}. \tag{23}$$

In the more general case where $\tau_k \sim p^{\tau_k}(\cdot)$, one should simply replace the above selection of the quadruple $(\breve{x}, \breve{\Delta}, \breve{w}, \tau)$ with

$$\breve{x} = x_{t_{k-1}-\Delta_{k-1}}^{(i)} \qquad \breve{\Delta} = \Delta_{k-1}^{(i)} \qquad \breve{w} = w^{(i)} \qquad \tau = \tau^{(i)} \tag{24}$$

where $\tau^{(i)} \sim p^{\tau_k}(\cdot)$ is a sample from the density of τ_k.

Once the predicted particles are obtained, the measurement update step follows:

$$x_{t_k-\Delta_k}^{(i)} = x_{t_k-\Delta_k|k-1}^{(i)} \tag{25}$$

$$\Delta_k^{(i)} = \Delta_{k|k-1}^{(i)} \tag{26}$$

for $i = 1, \ldots, N$. The updated weights are given by

$$\pi_k^{(i)} \propto p(y_k | x_{t_k-\Delta_k}^{(i)}), \tag{27}$$

normalized such that $\sum_{i=1}^N \pi_k^{(i)} = 1$. This is simply because of the fact that we use the bootstrap version [10] of the PF, i.e., the proposal density is the augmented system model $p(\xi_k|\xi_{k-1}, c_k)$.

It is interesting to see in the PF described above that each particle $\xi_k^{(i)}$ actually holds a state component $x_{t_k-\Delta_k}^{(i)}$ whose time stamp $t_k - \Delta_k$ is different from the other particles. The time stamp of each such component is determined by the delay component $\Delta_k^{(i)}$ of the corresponding particle. Hence forming the classical mean point estimate using the formula

$$\hat{\xi}_{k|k} = \sum_{i=1}^N \pi_k^{(i)} \xi_k^{(i)} \tag{28}$$

would actually average the state components that belong to different time instants of the target which would make the estimate meaningless. Hence, before obtaining the mean estimates, one has to predict the state components to a common deterministic time instant. Suppose that common time instant is selected to be the last measurement time t_k. Then the mean estimate $\hat{x}_{t_k}$ and covariance P_{t_k} of the PDM-PF can be calculated as follows.

$$\hat{x}_{t_k} = \sum_{i=1}^N \pi_k^{(i)} f_{t_k,t_k-\Delta_k^{(i)}}(x_{t_k-\Delta_k}^{(i)}) \tag{29}$$

$$P_{t_k} = \sum_{i=1}^N \pi_k^{(i)} \left[\left(x_{t_k}^{(i)} - \hat{x}_{t_k}\right)\left(x_{t_k}^{(i)} - \hat{x}_{t_k}\right)^{\mathrm{T}} + Q_{t_k,t_k-\Delta_k^{(i)}} \right] \tag{30}$$

where $Q_{t_k,t_k-\Delta_k^{(i)}}$ is the covariance of $w_{t_k,t_k-\Delta_k^{(i)}} \sim p^w_{t_k,t_k-\Delta_k^{(i)}}(\cdot)$. This is the specific output calculation step required by the PDM-PF filter.

We give a summarized description of one step of the PDM-PF filter in the algorithm below.

Algorithm 1. ***PDM-PF***

Suppose we have the summary statistics $\{\xi_{k-1}^{(i)} \triangleq [(x_{t_{k-1}-\Delta_{k-1}}^{(i)})^{\mathrm{T}}, \Delta_{k-1}^{(i)}]^{\mathrm{T}}\}_{i=1}^N$ *and* $\{\pi_{k-1}^{(i)}\}_{i=1}^N$*, below we give the steps to obtain the new summary statistics* $\{\xi_k^{(i)} \triangleq [(x_{t_k-\Delta_k}^{(i)})^{\mathrm{T}}, \Delta_k^{(i)}]^{\mathrm{T}}\}_{i=1}^N$ *and* $\{\pi_k^{(i)}\}_{i=1}^N$.

- ***Resampling:*** *Obtain the resampled particles* $\{\bar{\xi}_{k-1}^{(i)} \triangleq [(\bar{x}_{t_{k-1}-\Delta_{k-1}}^{(i)})^{\mathrm{T}}, \bar{\Delta}_{k-1}^{(i)}]^{\mathrm{T}}\}_{i=1}^{N}$ *such that*

$$P(\bar{\xi}_{k-1}^{(i)} = \xi_{k-1}^{(j)}) = \pi_{k-1}^{(j)} \tag{31}$$

for $j = 1, \ldots, N$.
- ***Prediction Update:*** *For* $i = 1, \ldots, N$
  - *Set* $\breve{x}$, $\breve{\Delta}$ *and* $\breve{w}$ *as*

$$\breve{x} = \bar{x}_{t_{k-1}-\Delta_{k-1}}^{(i)} \qquad \breve{\Delta} = \bar{\Delta}_{k-1}^{(i)} \qquad \breve{w} = w^{(i)} \tag{32}$$

  where $w^{(i)} \sim p_{t_k,t_{k-1}}^{w}(\cdot)$.
  - *Set* τ *as*

$$\tau = \begin{cases} 0, & \text{if } \tau_k \equiv 0 \\ \tau^{(i)}, & \text{otherwise} \end{cases} \tag{33}$$

  where $\tau^{(i)} \sim p^{\tau_k}(\cdot)$.
  - *Run the recursion of Theorem 1 with the above selection of the quadruple* $(\breve{x}, \breve{\Delta}, \breve{w}, \tau)$ *until convergence to obtain the fixed point* $\Delta_k^{(i)}(\infty)$.
  - *Set the predicted delayed state* $x_{t_k-\Delta_k|k-1}^{(i)}$ *and predicted delay* $\Delta_{k|k-1}^{(i)}$ *as*

$$x_{t_k-\Delta_k|k-1}^{(i)} = f_{t_k-\Delta_k^{(i)}(\infty),t_{k-1}-\breve{\Delta}}(\breve{x}) + \breve{w} \tag{34}$$

$$\Delta_{k|k-1}^{(i)} = \Delta_k^{(i)}(\infty). \tag{35}$$

- ***Measurement Update:***
  - *Set the updated particles as*

$$x_{t_k-\Delta_k}^{(i)} = x_{t_k-\Delta_k|k-1}^{(i)} \tag{36}$$

$$\Delta_k^{(i)} = \Delta_{k|k-1}^{(i)} \tag{37}$$

  for $i = 1, \ldots, N$.
  - *Set the updated weights as*

$$\pi_k^{(i)} \propto p(y_k | x_{t_k-\Delta_k}^{(i)}) \tag{38}$$

  such that $\sum_{i=1}^{N} \pi_k^{(i)} = 1$.
- ***Output Calculation:***
  - *Calculate the output state estimate* $\hat{x}_{t_k}$ *and covariance* P_{t_k} *as*

$$\hat{x}_{t_k} = \sum_{i=1}^{N} \pi_k^{(i)} f_{t_k,t_k-\Delta_k^{(i)}}(x_{t_k-\Delta_k}^{(i)}) \tag{39}$$

$$P_{t_k} = \sum_{i=1}^{N} \pi_k^{(i)} \Big[\left(x_{t_k}^{(i)} - \hat{x}_{t_k}\right) \left(x_{t_k}^{(i)} - \hat{x}_{t_k}\right)^{\mathrm{T}} + Q_{t_k,t_k-\Delta_k^{(i)}} \Big]. \tag{40}$$

□

B. Constant Propagation Speed Case

A common case in target tracking is the case of constant speed of signal propagation where simple sufficient conditions for the convergence of the recursion of Theorem 1 can be found. We give such conditions in the following corollary which enables the easy and fast use of the theorem in practical applications.

Corollary 1. *With a propagation model that assumes a constant speed of signal propagation, the condition for the exponential convergence of the recursion in Theorem 1 becomes*

$$\mathsf{v}_{\max}(t_{k-1} - \breve{\Delta}, t_k, \breve{x}) < \mathsf{v}_{\text{propagation}} \tag{41}$$

where $\mathsf{v}_{propagation}$ *is the speed of signal propagation. Furthermore, if in addition a (nearly) constant velocity (speed) model is used for target tracking, the condition becomes*

$$\| \operatorname{vel}(\breve{x}) \|_2 < \mathsf{v}_{\text{propagation}}. \tag{42}$$

If instead a (nearly) constant acceleration model is used, the condition becomes

$$\max\left(\| \operatorname{vel}(\breve{x}) \|_2, \| \operatorname{vel}(\breve{x}) + (t_k - t_{k-1} + \breve{\Delta}) \operatorname{acc}(\breve{x}) \|_2 \right) < \mathsf{v}_{\text{propagation}}. \tag{43}$$

□

Proof: Proof is given in Appendix B for the sake of clarity.□

The specific sufficient conditions of Corollary 1 are much simpler than the one in Theorem 1 and one can quite efficiently check that the exponential convergence of the recursion of Theorem 1 is guaranteed. The conditions are fairly weak in that they require the range of predicted target speeds to be smaller than the speed of signal propagation in the medium. This is a reasonable assumption because if the target of interest moves with a speed larger than the speed of signal propagation, doing the estimation based on such a signal is not a good idea in the first place. Considering for example a target moving directly towards the sensor with a larger speed than the signal propagation speed, the target would arrive at the sensor location even before its signal reaches the sensor hence there is practically no way of tracking the target with a good performance.

C. Alternative Ideas

Another idea for handling the implicit delay constraints is to run a numerical root-finding algorithm [22, Section 5.1] in order to obtain the solution such as Newton-Raphson or modified Newton [22, Section 5.4] algorithms, see [22, Chapter 5] for many other alternatives. In the case of the simplified example in Section II, Newton-Raphson method, for example, would yield the solution in a single iteration thanks to the quadratic cost function. In the general case treated in this section, we might not have such attractive properties that can be associated to the related cost functions. Nevertheless, one can still use a numerical root-finding algorithm on each

particle if the solutions can be obtained more quickly. An advantage of our method compared to some others like the modified Newton method [22, Section 5.4] is that it does not require a step-size selection mechanism.

V. Simulations

In this section, the performance of the PDM-PF is going to be compared on a simulated target tracking scenario with other possible approaches which are

- A PF which totally neglects that there is a delay in sensing.
- The deterministic sampling approach of [4].
- An alternative PF which tries to compensate for the delay by using its last estimate (particles).
- Another particle filter which uses a random walk model for the delays and incorporates the information of delay constraint using pseudo-measurements in the measurement update.

For this purpose, we choose to consider an exaggerated two-dimensional bearing only tracking problem with a single maneuvering sensor to clearly illustrate the detrimental effects of the signal propagation delays. The single target in the scenario makes a clockwise coordinated turn of radius 500m with a speed about 200km/h beginning in y-direction with the initial position $[-500\text{m}, 800\text{m}]$ for 45 seconds. The tracking sensor called S_1 acquires bearing data of the target corrupted by a Gaussian measurement noise with zero mean and standard deviation of 0.01 radians with sampling period $T = 1$s beginning at $t_0 = 4$ seconds. The difference of this scenario from the one that was used in [4] is that the measurement noise standard deviation used here is more realistic than that of [4] which was 0.001 radians. The sensor gets a total number of 42 measurements in the interval [4secs,45secs]. The sensor trajectory is selected to lie on the curve $\mathsf{y} = 100\sin(\frac{2\pi}{200}\mathsf{x})$ when x ranges in the interval $[-200\text{m}, 200\text{m}]$ beginning at $x = -200$ meters at time $t_0 = 4$ seconds with constant x-speed. The true target trajectory and the sensor positions used in the example are illustrated in Figure 3.

The target motion is modeled with a discretized coordinated turn model with an unknown constant turn rate (i.e., the turn rate is also a state variable) and with Cartesian velocity. Therefore, the state of the target is given as $x_k = [\mathsf{p}_k^{\mathsf{x}}, \mathsf{p}_k^{\mathsf{y}}, \mathsf{v}_k^{\mathsf{x}}, \mathsf{v}_k^{\mathsf{y}}, \omega_k]^{\mathrm{T}}$ where p, v and ω variables denote the position, velocity and turn rate respectively and the motion model is

$$x_{k+1} = \begin{bmatrix} 1 & 0 & \frac{\sin(\omega_k T_{k+1})}{\omega_k} & -\frac{1-\cos(\omega_k T_{k+1})}{\omega_k} & 0 \\ 0 & 1 & \frac{1-\cos(\omega_k T_{k+1})}{\omega_k} & \frac{\sin(\omega_k T_{k+1})}{\omega_k} & 0 \\ 0 & 0 & \cos(\omega_k T_{k+1}) & -\sin(\omega_k T_{k+1}) & 0 \\ 0 & 0 & \sin(\omega_k T_{k+1}) & \cos(\omega_k T_{k+1}) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} x_k + w_{k+1} \tag{44}$$

where $T_{k+1} = t_{k+1} - t_k$ is the length of the time period between x_k and x_{k+1} and w_k is a Gaussian noise with zero mean and covariance Q_k given as

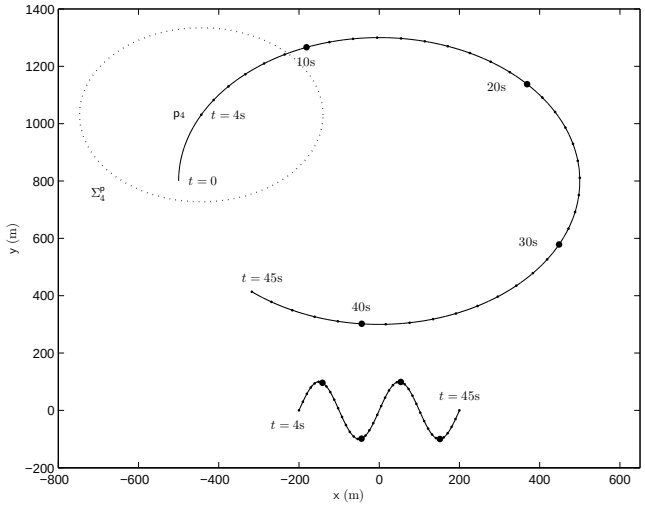

Fig. 3. The target and sensor trajectory used in the example. The target and sensor positions at measurement times $t_k = 4, 5, \ldots 45$s are emphasized with black dots. The filter initialization parameters p_4 and Σ_4^{p} are explained the main text.

$$Q_k = \begin{bmatrix} \sigma_{\dot{v}}^2 T_k^3/3 & 0 & \sigma_{\dot{v}}^2 T_k^2/2 & 0 & 0 \\ 0 & \sigma_{\dot{v}}^2 T_k^3/3 & 0 & \sigma_{\dot{v}}^2 T_k^2/2 & 0 \\ \sigma_{\dot{v}}^2 T_k^2/2 & 0 & \sigma_{\dot{v}}^2 T_k & 0 & 0 \\ 0 & \sigma_{\dot{v}}^2 T_k^2/2 & 0 & \sigma_{\dot{v}}^2 T_k & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\dot{\omega}}^2 T_k \end{bmatrix}. \tag{45}$$

In all simulations, we selected the standard deviations for the turn rate and speed as $\sigma_{\dot{\omega}} = 0.01\text{rad/s}^2$ and $\sigma_{\dot{v}} = 1\text{m/s}^2$ respectively. The measurement model is given as

$$y_k = \arctan \frac{\mathsf{p}_k^{\mathsf{y}} - \mathsf{p}_k^{\mathsf{y},S_1}}{\mathsf{p}_k^{\mathsf{x}} - \mathsf{p}_k^{\mathsf{x},S_1}} + \nu_k^S \tag{46}$$

where the superscript S_1 denotes the sensor related quantities.

The delay expression used in the simulations is given by (11) with $\mathsf{v}_{\text{sound}} \approx 344$m/s and $p^{\tau_k}(\tau) \triangleq \delta_0(\tau)$, i.e., $\tau_k \equiv 0$.

Five different algorithms are tested with the following brief descriptions (and abbreviations).

- **PF:** A standard particle filter which does not compensate for the propagation delay at all. In other words, it just uses the measurement equation
$$y_k = h_k(x_{t_k}) + v_k. \tag{47}$$
- **PDMF:** The deterministic sampling based propagation delay compensator proposed in [4].
- **PDM-PF:** The propagation delayed measurement compensating particle filter proposed in this work. In the prediction step, the recursions of Theorem 1 are applied to particles with a stopping rule
$$\left|\Delta_k^{(i)}(m) - \Delta_k^{(i)}(m-1)\right| < 10^{-10}\text{s}. \tag{48}$$
Since the target model used in the filter (coordinated turn) is a nearly constant speed model the convergence of the recursion on the ith particle is guaranteed by the condition $\|\operatorname{vel}(x_k^{(i)})\| < \mathsf{v}_{\text{sound}}$ according to Corollary 1. The

recursion is carried out only on the particles satisfying this condition and if there are any particles not satisfying the condition their weight is set to zero and hence they are completely disregarded in the estimation.

- **PFD:** This algorithm uses the same measurement equation (47) as PF and hence does not take care of the delay in the prediction or measurement updates. However, it knows that what it estimates is $x_{t_k-\Delta_k}$. Therefore, for each of its particles it calculates a delay estimate $\Delta_k^{(i)}$ from (11). Then, it extrapolates its particles as PDM-PF does to calculate an output estimate. This is the most straightforward estimation solution that comes to mind but it ignores the dynamics of Δ_k and therefore its estimate of $x_{t_k-\Delta_k}$ is wrong which can cause unpredictable errors.
- **PFD-RW:** An algorithm that assumes that the delays Δ_k satisfy a random walk (represented by the abbreviation RW) model given as follows

$$\Delta_k = \Delta_{k-1} + \gamma_k \tag{49}$$

where the standard deviation of the zero-mean Gaussian noise γ_k has been taken to be 0.13s which was calculated from the true delays in the example. This algorithm runs a particle filter that keeps the same number of states $x^{(i)}_{t_k-\Delta_k}$ and delays $\Delta_k^{(i)}$ as PDM-PF. In each prediction step (suppose we are at time $k-1$), first the delays $\{\Delta_{k-1}^{(i)}\}_{i=1}^N$ are predicted using the model (49) to obtain $\{\Delta_k^{(i)}\}_{i=1}^N$. Then, the states $x^{(i)}_{t_{k-1}-\Delta_{k-1}}$ are predicted as

$$x_{t_k} = f_{t_k-\Delta_k^{(i)},t_{k-1}-\Delta_{k-1}^{(i)}}(x_{t_{k-1}-\Delta_{k-1}}) + w^{(i)} \tag{50}$$

where $w^{(i)} \sim p^w_{t_k,t_{k-1}}(\cdot)$ as in PDM-PF. If the measurement update was done in exactly the same way as PDM-PF, such an algorithm would not use the delay constraint. Instead, we here use the delay constraint in the measurement update as a pseudo measurement [13–15] by defining the augmented measurement vector $\bar{y}_k$ as

$$\bar{y}_k \triangleq \begin{bmatrix} y_k \\ 0 \end{bmatrix} = \begin{bmatrix} h_k(x_{t_k-\Delta_k}) + v_k \\ d_{t_k}(x_{t_k-\Delta_k}) - \Delta_k + v_k^{\Delta} \end{bmatrix} \tag{51}$$

where the distribution of v_k^{Δ} has been selected to be $\mathcal{N}(v_k^{\Delta}; 0, 0.2^2)$. The measurement update is done similarly to PDM-PF after replacing the likelihood $p(y_k|x^{(i)}_{t_k-\Delta_k})$ in (38) with $p(\bar{y}_k|x^{(i)}_{t_k-\Delta_k}, \Delta_k^{(i)})$. The output calculation step is the same as PDM-PF. Note that this type of methodology was also described in [4, Section IV] for deterministic sampling based implementations.

All the particle filters to be run have been initialized with randomly generated particles $x_4^{(i)} \sim \mathcal{N}(\mu_4, \Sigma_4)$ where $\Sigma_4 = \mathrm{diag}(100^2, 100^2, 10^2, 10^2, 0.05^2)$ and the mean μ_4 of the initial particles is also selected randomly such that $\mu_4 \sim \mathcal{N}(x_4, \Sigma_4)$ where x_4 is the true target state. Notice here that it is not the initial particles $x_4^{(i)}$ but their mean μ_4 which is distributed around the true target state. The initial state estimate and covariance of PDMF are set to the random mean μ_4 and the covariance Σ_4 respectively. The position component $\mathsf{p}_4 \triangleq \mathrm{pos}(x_4)$ of x_4 and the 99 percent confidence ellipse of the position partition Σ_4^{p} of the covariance Σ_4 are also illustrated in Figure 3. In order to be fair to alternative particle filters, the weights of the particles not satisfying the condition $\|\mathrm{vel}(x_k^{(i)})\| < \mathsf{v}_{\text{sound}}$ are also set to zero in PF, PFD and PFD-RW. All PFs have used 10000 particles.

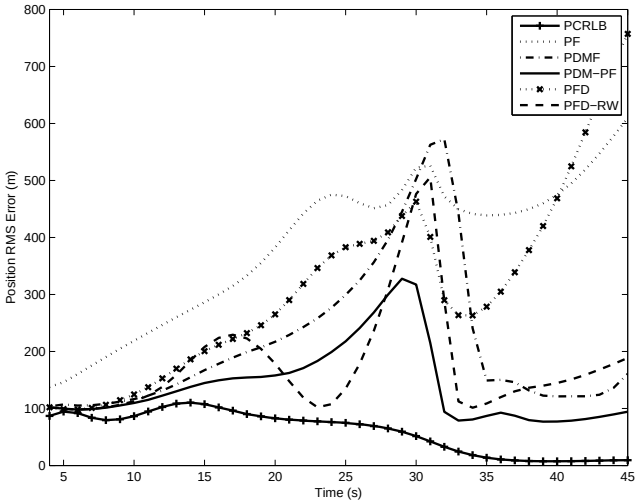

Fig. 4. RMS-position errors of the algorithms. The clairvoyant parametric Cramer-Rao lower bound (PCRLB) (calculated with known delays) is also illustrated.

A total number of 10000 Monte-Carlo runs have been made by changing the realization of the measurement noise and the initial particles/state estimate in each one. PDMF, as reported also in [4], turned out to be quite sensitive to measurement noise, and it has obtained speed estimates larger than $\mathsf{v}_{\text{sound}}$ in 31 Monte-Carlo runs where the recursions of Theorem 1 cannot be applied. In these runs, PDMF has been declared as divergent and such runs were not included in the averages calculated about PDMF. It has been observed that on average approximately 9 recursions for each particle were made to satisfy the stopping rule (48) in the prediction update of PDM-PF. Since one more prediction is necessary for the output calculation of PDM-PF, at each cycle of PDM-PF, approximately 10 standard prediction updates and one standard measurement update were performed. Compared to the standard algorithm PF, this means an approximate extra computation load of 9 prediction updates. RMS position and velocity errors of the algorithms are shown along with the clairvoyant parametric Cramer-Rao lower bounds (PCRLBs) (calculated with known delays) in Figures 4 and 5 respectively.

In both position and velocity estimation, PDM-PF seems to be better than all the other filters and the closest to PCRLB. PDM-PF is outperformed only by PFD-RW during a negligible time period which appears to be scenario dependent. In order to explain the performance differences of the other algorithms, we show the true propagation delay sequence for the example along with the average estimated propagation delays for the delay compensating algorithms in Figure 6. One standard deviation variations over the Monte-Carlo runs are also shown in the figure. Although PDMF diverges in some runs as mentioned above it has considerably reasonable average propa-

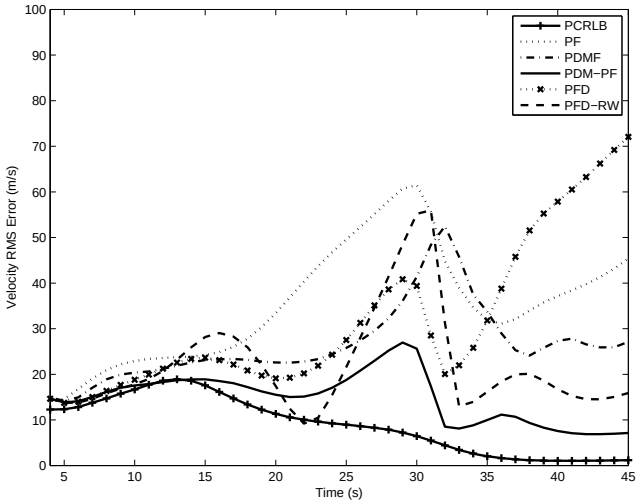

Fig. 5. RMS-velocity errors of the algorithms. The clairvoyant parametric Cramer-Rao lower bound (PCRLB) (calculated with known delays) is also illustrated.

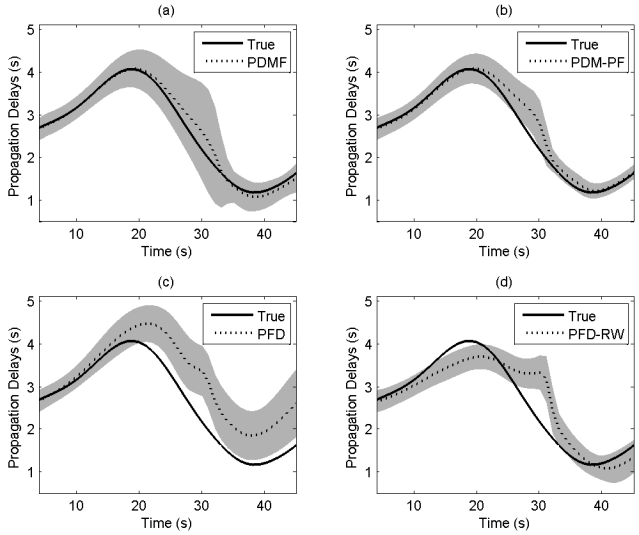

Fig. 6. The true and average estimated propagation delays for PDMF in (a), PDM-PF in (b), PFD in (c) and PFD-RW in (d). The variations of one standard deviation over the Monte-Carlo runs are also illustrated with grey clouds around the estimated delays.

gation delay estimates. The average delay estimates of PDM-PF are similar but with a smaller variation. An interesting observation is that in the short time period where PFD-RW outperforms PDM-PF, the average propagation delay estimate curve of PFD-RW intersects the true delay curve which shows that this behavior is scenario dependent as mentioned above.

In Figures 4 and 5 it is shown that PFD mostly performs better than PF which neglects the delay. This shows that the delay compensating strategy of PFD is effective to some extent. On the other hand, after $t = 35$s, the performance of PFD degrades to be significantly below that of PF. Note that the delay compensated particles of PFD are just extrapolated particles of PF with the delays calculated from the state of each particle. Since PF does not take into account the delays, these state values turn out to be wrong which leads to biased

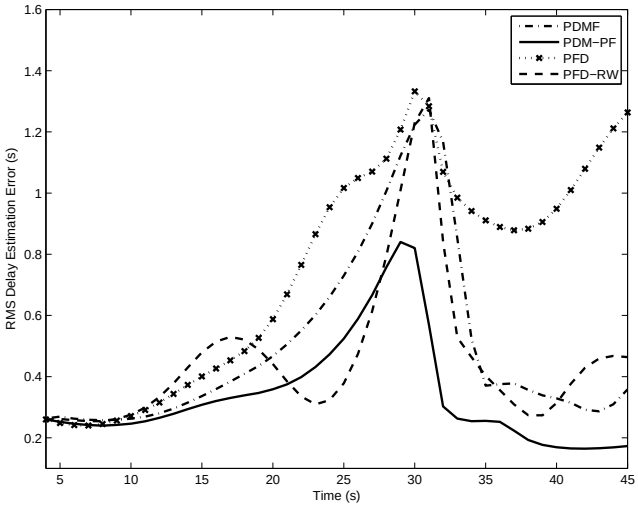

Fig. 7. RMS delay estimation errors of the delay compensating algorithms.

delay calculation of PFD as shown in Figure 6. When the true delays are large, this bias can be thought of as negligible and this manifests itself in the better performance of PFD in such regions. However, towards the end of the scenario, the biases which do not get smaller start becoming important while the true delay shrinks. As a result, the wrong state values begin to be extrapolated with significantly wrong delay values which results in the worse performance of PFD than PF.

PDM-PF compensates for the delays in its prediction update. PFD does this after the measurement update and outside the estimation loop which can cause even worse results than PF. PFD-RW, on the other hand, incorporates the delay constraint information in the measurement update as a compensation. In this measurement update, the particles with delay–state pairs that do not satisfy the delay constraint would get very small weights and hence disappear due to the additional pseudo-measurement. Computation resources on such particles are wasted and the particle depletion in the particle filter could get worse, i.e., the number of effective particles decreases. The advantage of PDM-PF compared to PFD-RW lies in that it never generates a particle with a delay–state pair that does not satisfy the delay constraint thanks to the recursion of Theorem 1. As another point, the process noise of (49) and pseudo-measurement noise v_k^{Δ} of (51) are artificially imposed into the information structure of the problem to enable the stability with a particle filter and they would cause performance loss.

The delay compensation methodologies of PFD and PDF-RW are still useful to some extent compared to PF when the delays do not change much which is the case when the target is far away and moves orthogonally to the range vector to the sensor, but when this is not the case, as seen towards the end of the simulation, they might cause even more (PFD) or similar (PFD-RW) errors. When the delays change quite fast (between 20s–35s) the PDM-PF performance is also affected, however, the estimates can quickly recover after this period ends. This is more clearly illustrated with the RMS delay estimation errors of the delay compensating algorithms shown

in Figure 7. PFD-RW and also PDMF (to a lesser extent) are also capable of recovering a level of reasonable performance after this challenging period which is evident also from their propagation delay estimation performance at the end of the scenario.

VI. Conclusions

We have proposed a particle filter called PDM-PF that compensates for the signal propagation delays in the prediction update of the particle filter. Compared to the earlier deterministic sampling based method, this filter has the capability to work with general stochastic nonlinear models and is much less sensitive to higher noise levels. Simulation results show that, in a quite challenging scenario, PDM-PF can beat two other PFs which compensate for the delays in measurement update or output calculation steps. This shows that the early compensation of the propagation delays is useful while working with propagation delayed measurements. Especially, the delay compensation outside the estimation loop in PFD was observed sometimes to potentially give worse results than even neglecting the delays. Compensating for the delays in a measurement update in the form of a pseudo-measurement also appeared as a feasible stable alternative though there might be important performance and computation resource losses compared to PDM-PF.

Another potential application that one can consider for the method proposed in this work might be (the fine tuning of) tracking and/or prediction of star trajectories based on visual bearing and azimuth information in which case the speed of light plays the role of signal propagation speed and the order of magnitude of the delays might be (light) years.

The assumption of target speed being smaller than the signal propagation speed used in this work precluded the use of the algorithm presented here in some practical applications such as [23] where the concern is on supersonic targets (bullets) sensed by microphones. In fact, as we hinted in Section II, it is theoretically possible to use the recursion (16) backwards to solve the implicit delay constraint. However, such an approach would involve possibly problem specific and constraining function invertibility requirements and hence excluded from the current work.

Appendix A
Proof of Theorem 1

The difference between two consecutive $\Delta_k(m)$ values is given as

$$\begin{aligned}
&|\Delta_k(m+1) - \Delta_k(m)| \\
&\quad = |d_{t_k}(x_{t_k-\Delta_k(m)}) + \tau - d_{t_k}(x_{t_k-\Delta_k(m-1)}) - \tau| \\
&\quad = |d_{t_k}(x_{t_k-\Delta_k(m)}) - d_{t_k}(x_{t_k-\Delta_k(m-1)})| \qquad (52)\\
&\quad \leq K_d \| \operatorname{pos}\left(x_{t_k-\Delta_k(m)}\right) - \operatorname{pos}\left(x_{t_k-\Delta_k(m-1)}\right) \|_2 \qquad (53)
\end{aligned}$$

where we used the Lipschitz property (15) of $d_{t_k}(\cdot)$. Noticing that the position difference (distance) $\| \operatorname{pos}\left(x_{t_k-\Delta_k(m)}\right) - \operatorname{pos}\left(x_{t_k-\Delta_k(m-1)}\right) \|_2$ between the states that can be obtained via the prediction relation

$$x_{t_k-\Delta_k} = f_{t_k-\Delta_k, t_{k-1}-\breve{\Delta}}(\breve{x}) + \breve{w}. \qquad (54)$$

for two different delay values ($\Delta_k(m)$ and $\Delta_k(m-1)$) is limited by the relation

$$\begin{aligned}
&\| \operatorname{pos}\left(x_{t_k-\Delta_k(m)}\right) - \operatorname{pos}\left(x_{t_k-\Delta_k(m-1)}\right) \|_2 \\
&\quad \leq \mathsf{v}_{\max}\Big(\min\big(t_k-\Delta_k(m-1), t_k-\Delta_k(m)\big) \\
&\qquad\qquad , \max\big(t_k-\Delta_k(m-1), t_k-\Delta_k(m)\big), \breve{x}\Big) \\
&\qquad \times |\Delta_k(m) - \Delta_k(m-1)| \qquad (55)\\
&\quad \leq \mathsf{v}_{\max}\left(t_{k-1} - \breve{\Delta}, \max\big(t_k-\Delta_k(m-1), t_k-\Delta_k(m)\big), \breve{x}\right) \\
&\qquad \times |\Delta_k(m) - \Delta_k(m-1)| \qquad (56)\\
&\quad \leq \mathsf{v}_{\max}\left(t_{k-1} - \breve{\Delta}, t_k, \breve{x}\right) |\Delta_k(m) - \Delta_k(m-1)| \qquad (57)
\end{aligned}$$

where the function $\mathsf{v}_{\max}(\cdot,\cdot,\cdot)$ is defined in (20) and it finds the maximum velocity of the target that might be predicted by the state space model when initialized with $\breve{x}$ at time $t_k - \breve{\Delta}$.

Now substituting (57) into (53), we get

$$\begin{aligned}
|\Delta_k(m+1) - \Delta_k(m)| \leq & K_d \mathsf{v}_{\max}\left(t_{k-1} - \breve{\Delta}, t_k, \breve{x}\right) \\
& \times |\Delta_k(m) - \Delta_k(m-1)| \qquad (58)
\end{aligned}$$

which proves that the overall transformation from $\Delta_k(m)$ to $\Delta_k(m+1)$ is a contraction if $K_d \mathsf{v}_{\max}\left(t_{k-1} - \breve{\Delta}, t_k, \breve{x}\right) < 1$. The result of the theorem is then given by the famous Banach fixed-point theorem (a.k.a. contraction mapping theorem) [24].

Appendix B
Proof of Corollary 1

With constant speed of propagation, we have

$$d_{t_k}(x) \triangleq \frac{\| \operatorname{pos}(x) - \mathsf{p}_{t_k}^{\text{sensor}} \|_2}{\mathsf{v}_{\text{propagation}}} + \tau_k. \qquad (59)$$

Then

$$\begin{aligned}
&|d_{t_k}(x) - d_{t_k}(x')| \\
&\qquad = \frac{\left| \| \operatorname{pos}(x) - \mathsf{p}_{t_k}^{\text{sensor}} \|_2 - \| \operatorname{pos}(x') - \mathsf{p}_{t_k}^{\text{sensor}} \|_2 \right|}{\mathsf{v}_{\text{propagation}}} \qquad (60)\\
&\qquad \leq \frac{1}{\mathsf{v}_{\text{propagation}}} \| \operatorname{pos}(x) - \operatorname{pos}(x') \|_2 \qquad (61)
\end{aligned}$$

where we used the reverse triangle inequality while writing (61) from (60). Identity (61) means that a Lipschitz constant for $d_{t_k}(x)$ is given as $K_d \triangleq \frac{1}{\mathsf{v}_{\text{propagation}}}$. Substituting K_d into the condition $K_d \mathsf{v}_{\max}(t_{k-1} - \breve{\Delta}, t_k, \breve{x}) < 1$ of Theorem 1, we obtain the first condition (41) of Corollary (1).

The second condition (42) of the corollary can be easily proven from the first condition (41) by noting that

$$\mathsf{v}_{\max}(t_{k-1} - \breve{\Delta}, t_k, \breve{x}) = \operatorname{vel}(\breve{x}) \qquad (62)$$

for all (nearly) constant velocity (speed) target state models.

Proving the third condition (43) is a little more involved.

$$\begin{aligned}
\mathsf{v}_{\max}(t_{k-1} - \breve{\Delta}, t_k, \breve{x}) &\triangleq \max_{t_{k-1}-\breve{\Delta} \leq t \leq t_k} \left\| \operatorname{vel}\left(f_{t, t_{k-1}-\breve{\Delta}}(\breve{x})\right) \right\|_2 \qquad (63)\\
&= \sqrt{\max_{t_{k-1}-\breve{\Delta} \leq t \leq t_k} \left\| \operatorname{vel}\left(f_{t, t_{k-1}-\breve{\Delta}}(\breve{x})\right) \right\|_2^2}. \qquad (64)
\end{aligned}$$

Now assume that we use a (nearly) constant acceleration model for target tracking. Then we have

$$\operatorname{vel}\left(f_{t,t_{k-1}-\breve{\Delta}}(\breve{x})\right) = \operatorname{vel}(\breve{x}) + (t - t_{k-1} + \breve{\Delta})\operatorname{acc}(\breve{x}). \quad (65)$$

Substituting (65) into (64), we get

$$\begin{aligned}&\mathsf{v}_{\max}(t_{k-1} - \breve{\Delta}, t_k, \breve{x}) \\ &\quad = \sqrt{\max_{t_{k-1}-\breve{\Delta} \le t \le t_k} \left\| \operatorname{vel}(\breve{x}) + (t - t_{k-1} + \breve{\Delta})\operatorname{acc}(\breve{x}) \right\|_2^2}.\end{aligned} \quad (66)$$

Now, the objective function of the $\max$-operator on the right hand side of (66) is a parabola in t and the coefficient of t^2 is $\|\operatorname{acc}(\breve{x})\|_2^2 \ge 0$. Hence, the maximum should be attained at one of the boundary points $t = t_{k-1} - \breve{\Delta}$ and $t = t_k$ giving

$$\begin{aligned}&\mathsf{v}_{\max}(t_{k-1} - \breve{\Delta}, t_k, \breve{x}) \\ &= \sqrt{\max\left(\|\operatorname{vel}(\breve{x})\|_2^2, \|\operatorname{vel}(\breve{x}) + (t_k - t_{k-1} + \breve{\Delta})\operatorname{acc}(\breve{x})\|_2^2\right)} \\ &= \max\left(\|\operatorname{vel}(\breve{x})\|_2, \|\operatorname{vel}(\breve{x}) + (t_k - t_{k-1} + \breve{\Delta})\operatorname{acc}(\breve{x})\|_2\right)\end{aligned} \quad (67)$$

Substituting the result (67) into the first condition (41), we obtain the third condition (43). Notice also that the third condition (43) reduces into the second condition (42) when $\operatorname{acc}(\breve{x}) = \mathbf{0}$.

Acknowledgments

The authors gratefully acknowledge fundings from the Swedish Research Council VR in the Linnaeus Center CADICS, and Swedish Foundation for Strategic Research in the center MOVIII. The strategic motivation and practical relevance for this contribution stem from the Vinnova, SSF (Swedish Foundation for Strategic Research) and KKS *Institute Excellence Centre* for Advanced Sensors, Multisensors and Sensor Networks (FOCUS). The first author thanks Henrik Ohlsson of Linköping University for proofreading an early version of the manuscript.

References

[1] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York: Wiley, 2001.

[2] Y. Bar-Shalom and X. R. Li, *Multitarget-Multisensor Tracking: Principles, Techniques*. Storrs, CT: YBS Publishing, 1995.

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

[4] U. Orguner and F. Gustafsson, "Target tracking using delayed measurements with implicit constraints," in *Proceedings of 11th International Conference on Information Fusion (FUSION 2008)*, Cologne, Germany, Jul. 2008.

[5] ——, "Distributed target tracking with propagation delayed measurements," in *Proceedings of 12th International Conference on Information Fusion (FUSION 2009)*, Seattle, Washington, Jul. 2009.

[6] ——, "Particle filtering with propagation delayed measurements," in *Proceedings of IEEE Aerospace Conference 2010*, Mar. 2010.

[7] S. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Trans. Autom. Control*, vol. 45, no. 3, pp. 477–482, Mar. 2000.

[8] A. Benaskeur, "Consistent fusion of correlated data sources," in *Proceedings of the 28th Annual Conference of the Industrial Electronics Society (IECON 02)*, vol. 4, Nov. 2002, pp. 2652–2656.

[9] Y. Zhou and J. Li, "Data fusion of unknown correlations using internal ellipsoidal approximation," in *Proceedings of the 17th IFAC World Congress*, Jul 2008.

[10] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "A novel approach to nonlinear/non-Gaussian Bayesian state estimation," in *IEE Proceedings on Radar and Signal Processing*, vol. 140, 1993, pp. 107–113.

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

[12] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*. Springer Verlag, 2001.

[13] M. Tahk and J. L. Speyer, "Target tracking problems subject to kinematic constraints," *IEEE Trans. Autom. Control*, vol. 35, no. 3, pp. 324–326, Mar. 1990.

[14] A. T. Alouani and W. D. Blair, "Use of a kinematic constraint in tracking constant speed, maneuvering targets," *IEEE Trans. Autom. Control*, vol. 38, no. 7, pp. 1107–1111, Jul. 1993.

[15] J. De Geeter, H. Van Brussel, J. De Schutter, and M. Decréton, "A smoothly constrained Kalman filter," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 10, pp. 1171–1177, Oct. 1997.

[16] R. Nikoukhah, A. S. Willsky, and B. C. Levy, "Kalman filtering and Riccati equations for descriptor systems," *IEEE Trans. Autom. Control*, vol. 37, no. 9, pp. 1325–1342, Sep. 1992.

[17] L. S. Wang, Y. T. Chiang, and F. R. Chang, "Filtering method for nonlinear systems with constraints," *IEE Proceedings on Control Theory and Applications*, vol. 149, no. 6, pp. 525–531, Nov. 2002.

[18] D. Simon and T. Chia, "Kalman filtering with state equality constraints," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 1, pp. 2774–2784, Jan. 2002.

[19] S. Julier and J. J. LaViola, Jr., "On Kalman filtering with nonlinear equality constraints," *IEEE Trans. Signal Process.*, vol. 55, no. 6, pp. 2774–2784, Jun. 2007.

[20] G. Nachi, "Kalman filtering in the presence of state space equality constraints," in *Proceedings of Chinese Control Conference, 2007. CCC 2007*, Jun. 2007, pp. 107–113.

[21] A. J. Calise, "Enforcing an algebraic constraint in extended Kalman filter design," in *Proceedings of AIAA Guidance, Navigation and Control Conference and Exhibit*, Aug. 2007.

[22] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis*, 2nd ed. New York: Springer-Verlag, 1993.

[23] D. Lindgren, O. Wilson, F. Gustafsson, and H. Habberstad, "Shooter localization in wireless microphone networks," in *Proceedings of 12th International Conference on Information Fusion (FUSION 2009)*, Seattle, Washington, Jul. 2009.

[24] E. Kreyszig, *Introductory Functional Analysis with Applications*. New York: John Wiley & Sons, 1978.

Umut Orguner (M'00) received the B.Sc., M.Sc., and Ph.D. degrees, all in electrical engineering, from the Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey, in 1999, 2002, and 2006 respectively.

He is currently an Assistant Professor at the Division of Automatic Control at the Department of Electrical Engineering of Linköping University, Linköping, Sweden, where he previously held a postdoctoral position between 2007 and 2009. His research interests include estimation theory, multiple-model estimation, target tracking, and information fusion.

Fredrik Gustafsson (S'91-M'93-SM'05) received the M.Sc. degree in electrical engineering and the Ph.D. degree in automatic control, both from Linköping University, Linköping, Sweden, in 1988 and 1992, respectively.

From 1992 to 1999, he held various positions in automatic control, and from 1999 to 2005 he had a professorship in communication systems at Linköping University, where he has been a Professor in sensor informatics at the Department of Electrical Engineering since 2005. His research interests are in stochastic signal processing, adaptive filtering, and change detection, with applications to communication, vehicular, airborne, and audio systems. His work in the sensor fusion area involves design and implementation of nonlinear filtering algorithms for localization, navigation, and tracking of all kind of platforms, including cars, aircraft, spacecraft, UAV's, surface and underwater vessels, cell phones, and film cameras for augmented reality. He is a co-founder of the companies NIRA Dynamics and Softube, developing signal processing software solutions for automotive and music industry, respectively.

Dr. Gustafsson was an Associate Editor for the IEEE TRANSACTIONS OF SIGNAL PROCESSING from 2000 to 2006 and is currently Associate Editor for the EURASIP Journal on Applied Signal Processing and the International Journal of Navigation and Observation. In 2004, he was awarded the Arnberg prize by the Royal Swedish Academy of Science (KVA), and in 2007 he was elected member of the Royal Academy of Engineering Sciences (IVA).