Expectation Maximization Algorithm for Calibration of Ground Sensor Networks using a Road Constrained Particle Filter

Marek Syldatk, Egils Sviestins
Data Fusion, Command and Control Systems
Security and Defence Solution
Saab AB, Järfälla, Sweden
Email: {marek.syldatk},{egils.sviestins}@saabgroup.com

Fredrik Gustafsson
Department of Electrical Engineering
Linköping University
Linköping, Sweden
Email: fredrik.gustafsson@liu.se

***Abstract*—Target tracking in ground sensor networks requires an accurate calibration of sensor positions and orientations, as well as sensor offsets and scale errors. We present a calibration algorithm based on the EM (expectation maximization) algorithm, where the particle filter is used for target tracking and a non-linear least squares estimator is used for estimation of the calibration parameters. The proposed algorithm is very simple to use in practice, since no ground truth of the target position and time synchronization are needed. In that way, opportunistic targets can also be used for calibration. For road-bound targets, a road-constrained particle filter is used to increase the performance. Tests on real data shows that a sensor position accuracy of a couple of meters is obtained from only one passing target.**

I. Introduction

Bias estimation (encompassing the concepts sensor calibration, registration, localization and alignment) is a crucial element of multisensor tracking. Reliable tracking requires the sensor data to be aligned and consistent, otherwise there is an obvious risk for degraded performance or even ghost tracks.

There are many previous works dealing with bias estimation (for example [10] or [2]). Here we focus on methods applicable to a ground sensor network. The different biases, here denoted bias parameters, may include location errors, orientation errors, range measurements etc., depending on sensor type.

There are two main groups of techniques for calibrating the sensors: using reference targets, or using targets of opportunity. Both can be performed on-line or off-line.

In the ground sensor applications, the reference targets could typically be one or more vehicles carrying satellite navigation equipment, e.g. GPS. One must make sure that enough measurements are generated for each sensor with the target at different locations. Problems to consider are how to make the GPS data available to the fusion centre, and also the fact, that the GPS data may be corrupt, especially in an urban environment. Once the data has been collected, one can apply e.g. a Maximum Likelihood (ML) method, to estimate the bias parameters that make observations of the target fit the reference data best.

In case no reference targets can be used, one needs to track targets of opportunity for the purpose of bias estimation. This may however be hard, as tracking may not work well without having the bias estimates. One approach is to simultaneously estimate track states and biases, which can be accomplished by forming augmented state vectors that combine target state estimates together with bias estimates [2]. With a large number of targets and sensors this will hardly be tractable due to computational requirements etc. There are however suboptimal but very efficient techniques to decouple target state and bias estimation process [4] or [10], fully feasible for on-line operation.

This paper examines an *off-line* approach to bias estimation with targets of opportunity. As the number of measurements suitable for bias estimation can be quite low, it is desirable to find a method that uses them as effectively as possible. To this end, the method considered in this paper is based on Expectation Maximization (EM) [9], with additional performance enhancing features, in particular using road constraints together with particle filtering and smoothing. In EM, the entire set of measurements is processed iteratively to provide both state estimates and bias parameters [5]. This will be described in detail in Section III after the formal definition in Section II. The techniques are developed further in Section IV and Section V with particle filtering and smoothing for on-road and off-road motion. Finally, Section VI provides experimental results.

II. Problem definition

A. Sensor and Motion Models

A ground target is detected by a number of sensors, resulting in a set of observations $Y_K = \{y_k\}_1^K$ of a target states $X_K = \{x_k\}_1^K$, where y_k is a single measurement, or set of measurements stacked as a vector, and x_k is target state at time k. The measurements are affected by a number of bias parameters and by a measurement noise. All the bias parameters, for all sensors, are collected in a single vector b.

The measurement model for each sensor is assumed to be a known, nonlinear function $h(x_k, b)$ of the state x_k, with measurement bias b, and is defined as

$$y_k = h(x_k, b) + \nu_k \tag{1}$$

where ν_k is an additive noise with known distribution $p_{\nu_k}(\cdot)$.

In general we assume that observed target moves according to a nonlinear motion model

$$x_{k+1} = f(x_k, \eta_{k+1}) \tag{2}$$

with motion model $f(\cdot)$ assumed to be known and with η_k being a process noise with known distribution $p_{\eta_k}(\cdot)$.

There is a number of sensor types used in ground target tracking, from which we can distinguish four basic types:

a) Linear range (TOA, time of arrival): The time τ of a signal arrival (transmission time) is measured, and by using the known speed v in the propagation media (speed of light/sound in air/water etc.) one can estimate range to the target. The measurement function is defined as

$$h_{TOA}(x_k, b) = |p_k - (s + b_p)| + b_r \tag{3}$$

where p_k is target positions vector at time k, s is the sensor position vector, b_p is the sensor position bias vector and b_r is an additive range measurement bias.

b) Linear range difference (TDOA, time difference of arrival): The transmission time is here unknown, but by using two sensors one can estimate a relative time of arrival, thus relative distance to the target. The measurement function takes then a form

$$h_{TDOA}(x_k, b) = |p_k - (s^1 + b_p^1)| - |p_k - (s^2 + b_p^2)| + b_d \tag{4}$$

where s^n is the n-th sensor position vector, b_p^n is the n-th sensor position bias vector and b_d is an additive range difference measurement bias.

c) Bearing measurements (DOA, difference of arrival): The sensor provides bearing (direction, angle) to the target, with measurement function

$$h_{DOA}(x_k, b) = \text{atan2}\left(\frac{p_k^x - (s_x + b_x)}{p_k^x - (s_y + b_y)}\right) + b_\theta \tag{5}$$

where p_k^x and p_k^y are target positions, s_x and s_y are sensor positions, b_x and b_y are sensor position biases in x and y direction respectively and b_θ is called north alignment bias.

d) Logarithmic range (power measurements).: The sensor measures the strength of a received signal, that can be of a different kind, e.g. acoustic or electromagnetic wave. The measurement function is then defined as

$$h_{RSS}(x_k, b) = \log(1 - b_G) + P_k^{LOG} - \beta \log(r) \tag{6}$$

where $r = |p_k - (s + b_p)|$ is the distance to the target, P_k^{LOG} is a logarithm of power emitted by the target at time k, β is called path loss constant and b_G is a gain bias related to miscalibration of sensors.

The list of sensor models and bias parameters above is not exhaustive, but can be considered as a good illustration of the calibration problem considered herein. Further, sensor modalities can be mixed arbitrarily in our framework. Also combined sensors can be included easily, such as a radar sensor, providing both range and bearing measurements.

B. General Estimation Framework

The estimation approach consists in finding the bias parameters b that maximize the marginal likelihood

$$p_b(Y_K) = \int p_b(X_K, Y_K) dX_K \tag{7}$$

where b indicates dependence on bias parameters. The EM algorithm computes the maximum likelihood estimate by iteratively solving an filtering problem to get X_k given an estimate of the bias, and an estimation problem to get b_k, given an estimate of X_k.

A general framework on the EM algorithm for estimating parameters in a nonlinear dynamic system is provided in [9].

III. Expectation Maximization algorithm

The ideas behind EM, as applicable to our problem, can be summarized as follows. It is an iterative method used for finding the Maximum Likelihood (ML) or Maximum Posterior (MAP) estimates of parameters, when a model depends on latent (hidden) values. In case of bias estimation, where we based on set of measurements Y_K want to estimate the bias vector b, the latent variables are the sequence of state vectors X_K. The two steps described below are repeated until convergence. The first step is an Expectation (E) step. Introducing the notation

$$\ln p_b(X_K, Y_K) = L_b(X_K, Y_K) \tag{8}$$

one computes the $Q(b, \hat{b})$, that is a minimum variance estimate [9] of above log likelihood, made use of the available data set $\{y_k\}_1^K$ and an assumption $\hat{b}$ of the true value of bias vector b. Function is defined as

$$\begin{aligned} Q(b, \hat{b}) &= E_{\hat{b}}\{L_b(X_K, Y_K)|Y_K\} \\ &= \int L_b(X_K, Y_K) p_{\hat{b}}(X_K|Y_K) dX_K \end{aligned} \tag{9}$$

The core idea behind the above procedure is that it should be much easier to maximize the complete likelihood $p_b(X_K, Y_K)$ than $p_b(Y_K)$ in eq. (7).

The second step is Maximization (M) step, where one calculates new bias estimate

$$\hat{b}_{n+1} = \arg\max_b p_b(Y_K) = \arg\max_b Q(b, \hat{b}_n) \tag{10}$$

This vector is used as an input for the next iteration of the two steps. As a result, with further iterations, the algorithm delivers a sequence of estimates $\hat{b}_n$ that are increasingly, with each iteration, better approximations of the ML estimate. As an initial value b_0 one can use zeros or any other data one may have available. The EM algorithm is presented in **Algorithm 1** and details of the Expectation and Maximization steps are given in the following sections.

Algorithm 1 - EM Algorithm

```
(1) Set n = 0 and initialize $\hat{b}_n$
(2) (E)xpectation step:
        Calculate: $Q(b, \hat{b}_n)$
(3) (M)aximization step:
        Calculate: $\hat{b}_{n+1} = \arg\max_b Q(b, \hat{b}_n)$
(4) Update n = n+1 until convergence.
```

The important problem in the EM algorithm is evaluation of the distribution $p_{\hat{b}}(X_K|Y_K)$ in (9), which is a smoothing problem. It should be discussed a bit wider due to the fact that there exist multiple ways to obtain estimates of b through EM.

In case of a linear measurement model and a linear motion model with normally distributed noise, one obtains a direct solution to the estimation problem by using a Kalman Smoother in the Expectation step together with Weighted Least Squares algorithm to obtain bias estimates in the Maximization step. It is then a wellknown property of the EM algorithm that it will converge to a local maximum of the likelihood function.

In the nonlinear case we need to use approximative algorithms. One general solution to the nonlinear filtering problem is provided by the Particle Smoother, that can obtain an arbitrarily good approximation of the $Q(b, \hat{b}_n)$ function. Then, one can use a gradient method to solve for b (which is equivalent to Nonlinear Least Squares problem). This method in detail is presented in [9].

Another way is to use linearization of the motion model and/or measurement model. With a linearized model, one can use the Extended Kalman Smoother to obtain smoothed state estimates [5]. Linearization allows for an iterative solution where the model is linearized at each new iterate.

In this paper, the method with linearized measurement function and particle smoother is used to obtain an approximation of $p_{\hat{b}}(X_K|Y_K)$.

A. Expectation step

We now focus on how to calculate $Q(b, \hat{b}_n)$ defined in (9). Using the Bayes' rule, the log likelihood (8) can be expressed as

$$L_b(X_K, Y_K) = \ln p_b(Y_K|X_K) + \ln p_b(X_K) \tag{11}$$

Actually, only the first term depends on the biases related to the measurement function, so under the assumption of known initial state distribution $p(x_1)$, the second term can be considered as a constant. Thus keeping only terms dependent on b, and including other terms into the constant term, one can write (8) in an extended form as

$$L_b(X_K, Y_K) = \sum_{k=1}^{K} \ln p_b(y_k|x_k) + const \tag{12}$$

Assuming that measurements are obtained from true target states x_k through a known, nonlinear measurement function

$$y_k = h(x_k, b) + \nu_k \tag{13}$$

defined in (1), with b being the bias vector and ν_k assumed to follow a Normal distribution with a known covariance R, we then have

$$L_b(X_K, Y_K) = -\frac{1}{2}\sum_{k=1}^{K}(h^T(x_k, b)R^{-1}h(x_k, b) - 2y_k^T R^{-1} h(x_k, b)) + const \tag{14}$$

where the terms independent of the bias b were again considered as constants.

Due to its nonlinearity, the measurement function needs to be linearized in order to obtain a closed form solution for bias estimates. The best point for linearization is around the state estimate $\hat{x}_{k|K} = E_{\hat{b}}\{x_k|Y_K\}$. A motivation to use this value comes directly from the fact, that computing of $Q(b, \hat{b}_n)$ requires calculation of this value. It is due to expectation over Y_k in (9), which requires computing $p_{\hat{b}}(X_K|Y_K)$. As will be seen below, linearization around the smoothed state estimate will simplify the final solution (by making some terms disappear).

In this paper, the particle filter together with a particle smoother will be applied to obtain smoothed state estimates. First, in the filtering step one obtains a set of particles $\{x_k^i\}_{i=1}^M$ together with their weights $\{w_k^i\}_{i=1}^M$, where M is the number of particles, that are used in the smoothing step to compute the smoothed estimate $\hat{x}_{k|K}$. Detailed description of these algorithms will be provided in Section 3 and 4 respectively.

To proceed, the measurement function is linearized around the smoothed estimate $\hat{x}_{k|K}$ and bias estimate $\hat{b}_n$ using a Taylor 1^{st} order expansion, resulting in

$$h(x_k, b) \approx h(\hat{x}_{k|K}, \hat{b}_n) + H_k^b(b - \hat{b}_n) + H_k^x(x_k - \hat{x}_{k|K}) \tag{15}$$

with Jacobians

$$H_k^x = \frac{\partial h(x_k, b)}{\partial x_k}\bigg|_{\substack{x_k = \hat{x}_{k|K} \\ b = \hat{b}_n}} \quad \text{and} \quad H_k^b = \frac{\partial h(x_k, b)}{\partial b}\bigg|_{\substack{x_k = \hat{x}_{k|K} \\ b = \hat{b}_n}}$$

Substituting (15) into (14) and applying the expectation operator $E_{\hat{b}}\{\cdot|Y_K\}$ to (14), in order to calculate $Q(b, \hat{b}_n)$ (cf. (9)), and by again throwing away terms that do not depend on b, together with terms that are linear in $x_k - \hat{x}_{k|K}$, as their expectation value vanishes (as a direct result of linearization around smoothed state estimate). What remains is

$$Q(b, \hat{b}) = -\frac{1}{2}\sum_{k=1}^{K}\big(-2(y_k - h(\hat{x}_{k|K}, \hat{b}) + H_k^b \hat{b})^T R^{-1} H_k^b b + b^T (H_k^b)^T R^{-1} H_k^b b\big) + const \tag{16}$$

B. Maximization step

In the maximization step, we compute new bias estimate $\hat{b}_{n+1}$ from (10). The maximum can be found by solving

$$\frac{\partial Q(b, \hat{b}_n)}{\partial b} = 0 \tag{17}$$

for b and using the property of symmetry of measurement covariance matrix $R = R^T$. Because (16) has a form of $Ab + b^T Bb + const$, where

$$A = \sum_{k=1}^{K} (y_k - h(\hat{x}_{k|K}, \hat{b}) + H_k^b \hat{b})^T R^{-1} H_k^b \tag{18}$$

$$B = \frac{1}{2} \sum_{k=1}^{K} (H_k^b)^T R^{-1} H_k^b \tag{19}$$

with B as mentioned symmetric and $const$ independent of b, the result is simply $\hat{b}_{n+1} = -B^{-1}A/2$, or explicitly

$$\hat{b}_{n+1} = \left(\sum_{k=1}^{K} (H_k^b)^T R^{-1} H_k^b \right)^{-1} \cdot \left(\sum_{k=1}^{K} (H_k^b)^T R^{-1} (y_k - h(\hat{x}_{k|K}, \hat{b}) + H_k^b \hat{b}) \right) \tag{20}$$

C. Comments on observability and convergence

For the maximization step to work it is necessary that B is invertible (or rather, that $\hat{b}_{n+1} = -B^{-1}A/2$ is a well-conditioned problem). Typically this is not the case: there are too many degrees of freedom for the bias parameters. The freedom can be restricted in different ways: fixing certain parameters or using road constraints can be very helpful, but not always enough. There are also requirements on the set of measurements. If there are too few measurements, of if e.g. the target has not moved in the measurement set, it will in most cases also be impossible to find the parameters. Using priors is also another way to ensure observability.

The proof of a convergence of the EM algorithm can be found in literature in [9] or [1].

IV. PARTICLE FILTERING

As mentioned in the previous section, to obtain bias estimates $\hat{b}_n$ through the EM algorithm, we require smoothed state estimates $\hat{x}_{k|K}$. To obtain those estimates, we will use the particle smoother. To be able to apply smoothing, we first need to run a particle filter, that will provide a set of particles, together with their weights, further used in a smoothing step. In this section, a simple SIR (Sequential Importance Resampling) algorithm will be used to obtain the required particles and weights, followed by its modification with applied road constraints.

A. Particle Filter

In general, we assume that the observed target moves according to a nonlinear motion model defined in (2)

$$x_{k+1} = f(x_k, \eta_{k+1}) \tag{21}$$

with motion model $f(\cdot)$ assumed to be known and with η_k being a process noise with known distribution $p_{\eta_k}(\cdot)$. The measurement function used to obtain target observations was previously defined in (13). A simple SIR particle filter algorithm [3] [7] is presented in **Algorithm 2**.

Algorithm 2 - SIR Particle Filter

(1) **Initialize:** Set $k = 1$ and initialize particles

$$\{x_0^i\}_{i=1}^M \sim p(x_0) \tag{22}$$

(2) **Prediction:** Predict the particles by drawing M i.i.d. samples according to

$$\tilde{x}_k^i \sim p(\tilde{x}_k | x_{k-1}^i), \qquad i = 1, \ldots, M \tag{23}$$

(3) **Update:** Compute the importance weights $\{w_k^i\}_{i=1}^M$,

$$w_k^i = w(\tilde{x}_k^i) = \frac{p_b(y_k | \tilde{x}_k^i)}{\sum_{j=1}^M p_b(y_k | \tilde{x}_k^j)}, \qquad i = 1, \ldots, M \tag{24}$$

(4) **Resampling:** For each $j = 1, \ldots, M$ draw a new particle x_k^i with replacement (resample) according to

$$P(x_k^j = \tilde{x}_k^i) = w_k^i, \qquad i = 1, \ldots, M \tag{25}$$

(5) Increment $k = k + 1$ while $k < K$ and return to step 2, otherwise terminate.

B. Road Constrained Particle Filter

When the target moves on a road, and a map of the road is available, it is natural to try to use this extra knowledge as a constraint in order to obtain more accurate estimates. In this section a road constrained particle filter (where the target moves only on the road map) will be described in detail.

There are many ways to apply road constraints to the particle filter. In this paper it will be assumed, that the road map is represented by a road network defined as J_{RN}. The road network is a set of definitions of straight segments together with a description of their attributes. Each segment is build of 2 points (a start point and an end point). Points common for 3 or more segments are considered to be junctions.

In general, the target state is defined in some coordinate system, let us call it global for a purpose of this paper. For the road constrained filtering there exists a need of using an extra coordinate system, that will describe target position in road coordinates. The state at time k in this system will be denoted as x_k^r, where r denotes road coordinate system.

The on-road state vector $x_k^r = [z_k \ l_k]^T$ is combined of two vectors: $z_k = [s_k \ v_k]^T$ describing a one-dimensional motion model, where s_k is a total distance traveled on the road since $k = 1$, v_k is a speed, and l_k informs about the road segment the target is on, location on the segment and direction of movement.

The motion model for the on road case is, as in general case, also nonlinear and defined [6] as

$$x_{k+1}^r = f^r(x_k^r, J_{RN}, \eta_{k+1}^r, \nu_{k+1}^r) \tag{26}$$

where η_k^r is a process noise with known distributions $p_{\eta_k^r}(\cdot)$ and ν_k^r is a discrete process noise, determining the choice of next road segment with known distributions $p_{\nu_k^r}(\cdot)$

Because the measurement function $h(x_k, b)$ is usually defined in global coordinates, there is a need to be able to convert target state from road to global coordinates. Thus we assume, there exists a transformation that allows exact conversion from global coordinates to road coordinates and opposite. Let $\Gamma^{r2g}(\cdot)$ be the function converting road coordinates to global coordinates and let $\Gamma^{g2r}(\cdot)$ be the transformation from global coordinates to road coordinates. In the on-road case, the measurement function (13) takes the form

$$y_k = h(\Gamma^{r2g}(x_k^r), b) + \nu_k. \tag{27}$$

The procedure for on road filtering is analogous to the one presented in Algorithm 2, but with a few modifications. In the initialization step, particles can be initialized directly on the road (in road coordinates) or projected onto road in the case when the initial distribution is only known in global coordinates. Then initial samples (particles) need to be projected using some known projection function. In general, equation (22) in Algorithm 2, for road constrained case, takes the form

$$\{x_0^{r,i}\}_{i=1}^M \sim p(x_0^r) \tag{28}$$

Next, in prediction step one needs to consider the on road motion model together with the probability of choosing one of the next road segments (in case of junctions or end of the road [8]). The likelihood in (23), which we sample from in step (2) of Algorithm 2, is then

$$\begin{aligned} p(x_k^r|x_{k-1}^r) &= p(z_k, l_k|z_{k-1}, l_{k-1}) \\ &= \frac{p(z_k, l_k, z_{k-1}, l_{k-1})}{p(z_{k-1}, l_{k-1})}, \end{aligned} \tag{29}$$

where Bayes theorem was used. By using the property of independence of z_{k-1} and l_{k-1}, and Bayes theorem again, (29) can be rewritten as

$$\begin{aligned} p(x_k^r|x_{k-1}^r) &= \frac{p(l_k, l_{k-1}|z_k, z_{k-1})}{p(l_{k-1})} \frac{p(z_k, z_{k-1})}{p(z_{k-1})} \\ &= \frac{p(l_k|l_{k-1}, z_k, z_{k-1})p(l_{k-1}|z_k, z_{k-1})}{p(l_{k-1})} p(z_k|z_{k-1}) \end{aligned} \tag{30}$$

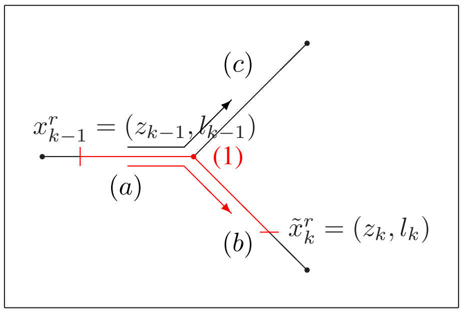
Using the fact that $p(l_k|z_k, z_{k-1}) = p(l_k)$, terms from nominator and denominator disappear and we finally get

$$p(x_k^r|x_{k-1}^r) = p(z_k|z_{k-1})p(l_k|l_{k-1}, z_k, z_{k-1}) \tag{31}$$

As was mentioned before, in the update step of Algorithm 2 there might be a need to convert particles from road to global coordinates. To calculate $p_b(y_k|\tilde{x}_k^i)$ we should use the coordinate conversion function, as in (27).

Modified version for road constrained particle filter is presented in **Algorithm 3**.

Now we focus more deeply on sampling from the distribution $p(\tilde{x}_k^r|x_{k-1}^{r,i})$ in (33), with respect to known motion model and noise distribution. According to the first term in the equation, $p(\tilde{z}_k|z_{k-1}^i)$, we sample with respect to the on-road motion model. The second term depends on the road map, and is called junction selection likelihood.



(c)
$x_{k-1}^r = (z_{k-1}, l_{k-1})$
(1)
(a)
(b)
$\tilde{x}_k^r = (z_k, l_k)$

Fig. 1. Junction example

Algorithm 3 - Road constrained Particle Filter

(1) **Initialize:** Set $k = 1$ and initialize particles

$$\{x_0^{r,i}\}_{i=1}^M \sim p(x_0^r) \tag{32}$$

(2) **Prediction:** Predict the particles by drawing M i.i.d. samples according to (31)

$$\tilde{x}_k^{r,i} \sim p(\tilde{x}_k^r|x_{k-1}^{r,i}), \qquad i = 1, \dots, M \tag{33}$$

(3) **Update:** Compute the importance weights $\{w_k^i\}_{i=1}^M$,

$$w_k^i = w(\tilde{x}_k^{r,i}) = \frac{p_b(y_k|\Gamma(\tilde{x}_k^{r,i}))}{\sum_{j=1}^M p_b(y_k|\Gamma(\tilde{x}_k^{r,j}))}, \qquad i = 1, \dots, M \tag{34}$$

(4) **Resampling:** For each $j = 1, \dots, M$ draw a new particle $x_k^{r,i}$ with replacement (resample) according to

$$P(x_k^{r,j} = \tilde{x}_k^{r,i}) = w_k^i, \qquad i = 1, \dots, M \tag{35}$$

(5) Increment $k = k + 1$ while $k < K$ and return to step 2, otherwise terminate.

The particle $x_{k-1}^{r,i}$ at time $k-1$ is located on a certain segment, and its location is described by the vectors z_{k-1} and l_{k-1} (as presented in Figure 1).

The first step is to sample a new $\tilde{z}_k$ from distribution $p(\tilde{z}_k|z_{k-1}^i)$. Having a new sample $\tilde{z}_k = [\tilde{s_k}\ \tilde{v_k}]^T$, we need to calculate the distance made by particle on road, $d = \tilde{s}_k - s_{k-1}$, which is a difference between total distance s at time k and $k-1$. If the distance d is smaller or equal to the remaining distance to the junction (1) (end of a segment (a)), particle does not cross the junction and stays on segment (a) with probability 1, so $\tilde{l_k} = l_{k-1}$.

In case, when the distance d is larger than the remaining distance to the junction (1), the particle changes road segment. As we can see on the example in Figure 1, the particle has 2 possible paths to follow (red to segment (b) or black to segment (c)). In this paper, the distribution $p_{\nu_k^r}(\cdot)$, determining the choice of the next road segment, is assumed to be uniform, so particle changes its location to one of two possible segments ((b) or (c)) with an equal probability $\frac{1}{m}$, where m is a number

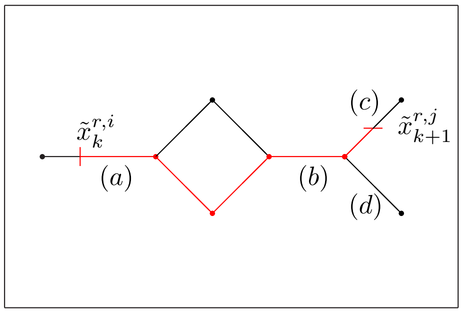


Fig. 2. Multiple junction example

of possible segments to move onto (in this example $m = 2$).

In case where at one time step the particle crosses more than one junction, we repeat the above procedure until the distance d is smaller than the remaining distance to the end of the segment the particle is on.

V. Smoothing

As was mentioned before, to calculate smoothed state estimates we need to run first the particle filter and store all predicted particles $\{\tilde{x}_k^i\}_{i=1}^M$ and corresponding weights $\{w_k^i\}_{i=1}^M$ for all $k = 1, \dots, K$ and then run Algorithm 4 [9].

Algorithm 4 - Particle Smoother

(1) **Initialization:** Set filtered terminal weights $\{w_k^i\}$ to be initialized smoothed weights at time $k = K$ as

$$w_{K|K}^i = w_K^i, \qquad i = 1, \dots, M \tag{36}$$

(2) **Smoothing:** Use filtered weights $\{w_k^i\}$ and sets of stored particles $\{\tilde{x}_k^i, \tilde{x}_{k+1}^i\}_{i=1}^M$ to compute smoothed weights $\{w_{k|K}^i\}_{i=1}^M$ using formulas below:

$$w_{k|K}^i = w_K^i \sum_{j=1}^M w_{k+1|K}^j \frac{p(\tilde{x}_{k+1}^j|\tilde{x}_k^i)}{v_k^j} \tag{37}$$

where

$$v_k^j = \sum_{i=1}^M w_k^i p(\tilde{x}_{k+1}^j|\tilde{x}_k^i) \tag{38}$$

(3) While $k > 0$ decrease $k = k - 1$ and return to step (2), otherwise terminate.

In case of road constrained algorithm, the above procedure needs to be modified, analogously to the filtering case. The main difference is in calculating the probability $p(\tilde{x}_{k+1}^j|\tilde{x}_k^i)$, between 2 particles, as in step (2) and (3) of Algorithm 4. In the road constrained case it follows, as defined in (31)

$$p(\tilde{x}_{k+1}^{r,j}|\tilde{x}_k^{r,i}) = p(\tilde{z}_{k+1}^j|\tilde{z}_k^i)p(\tilde{l}_{k+1}^j|\tilde{l}_k^i, \tilde{z}_{k+1}^j, \tilde{z}_k^i) \tag{39}$$

There are two main problems regarding evaluation of this probability.

The first term of equation (39) is directly related to the one-dimensional on-road motion model, as defined in (26). The second term is related to the road network. In a simple case, when particles $\tilde{x}_{k+1}^{r,j}$ and $\tilde{x}_k^{r,i}$ are located on the same segment, the second term is equal to 1. In case when particles are located on corresponding segments (problem analogous to the one presented on Figure 1), the probability is equal to $\frac{1}{m}$, where m is the number of possible segments the particle can choose.

In general case, when the on-road path between 2 particles crosses more than one junction (as presented on Figure 2), one needs to consider all the possible trajectories between two particles. As seen in the figure, particle moving from segment (a), through (b) to (c), can pick two possible trajectories. Thus, the second term in (39), for trajectories between (a) and (b) is equal to 1, and between (b) and (c) to $\frac{1}{m}$ with $m = 2$. That makes the junction selection probability between segment (a) and (c) equal to $1 \cdot \frac{1}{2} = \frac{1}{2}$.

A second problem is that the probability in (39) needs to be computed for each pair of particles in each smoothing step k, thus the complexity of one step is equal to M^2, where M is the number of particles. Because computation of the likelihood requires finding all possible paths between two points on-road, it is computationally demanding. This is considered as an important part to be improved in future.

Having the set of particles $\{\tilde{x}_k^i\}_{i=1}^M$ and their smoothed weights $\{w_{k|K}^i\}_{i=1}^M$, the minimum mean square error estimate of the target state can be computed as

$$\hat{x}_{k|K} = \sum_{i=1}^M w_{k|K}^i \tilde{x}_k^i, \tag{40}$$

and used to obtain bias estimate $\hat{b}_n$ through equation (20). Because the EM algorithm is, in general, defined in global coordinates, there is a need to convert estimates in case when road constraints are applied. In this paper particles in road coordinates are converted first to global coordinates using the Γ^{r2g} mapping, and together with the smoothed weights are used to obtain the state estimate in global coordinates through

$$\hat{x}_{k|K} = \sum_{i=1}^M w_{k|K}^i \Gamma^{r2g}(\tilde{x}_k^{r,i}). \tag{41}$$

Because, in the above, we are using particles converted to global coordinates it might happen that the estimate will not be on-road. A natural remedy is to compute the particle that minimizes the mean square error,

$$\hat{x}_{k|K} = \arg\min_{\tilde{x}_k^j} \|\tilde{x}_k^i - \sum_{i=1}^M w_{k|K}^i \tilde{x}_k^i\|^2. \tag{42}$$

Since each particle is constrained to be on-road, this procedure will guarantee that also the estimate is on-road.

VI. Experimental results

The EM algorithm with applied road constraints will be compared with the version without constraints in the experiment, where a single target is moving on-road and is being observed by a number of sensors measuring (unknown) acoustic power emitted by the target. Evaluation will be performed on real data. In both versions of the EM algorithm, a constant velocity (CV) motion model [6] was considered with an extra state representing logarithm of acoustic power P^{log} emitted by the target. The motion model in global coordinates, defined in (21), was represented by a two-dimensional CV model defined as

$$\begin{bmatrix} p^x_{k+1} \\ p^y_{k+1} \\ v^x_{k+1} \\ v^y_{k+1} \\ P^{log}_{k+1} \end{bmatrix} = \begin{bmatrix} I_2 & TI_2 & 0_{2\times1} \\ 0_2 & I_2 & 0_{2\times1} \\ 0_{1\times2} & 0_{1\times2} & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \\ P^{log}_k \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2}I_2 & 0_{2\times1} \\ TI_2 & 0_{2\times1} \\ 0_{1\times2} & 1 \end{bmatrix} \eta^g_{k+1} \tag{43}$$

where $T = 1$, p^x_k and p^y_k represent x and y position at time k respectively, v^x_k and v^y_k represent x and y velocities at time k, P^{log}_k is the emitted acoustic power at time k and η_k is a three-dimensional Gaussian process noise with zero mean and covariance $Q = diag([25\ 25\ 0.1])$.

The on-road motion model is defined as

$$\begin{bmatrix} s_{k+1} \\ v_{k+1} \\ P^{log}_{k+1} \\ l_{k+1} \end{bmatrix} = f^r\left(\begin{bmatrix} s_k \\ v_k \\ P^{log}_k \\ l_k \end{bmatrix}, J_{RN}, \nu^r_{k+1}, \eta^r_{k+1} \right) \tag{44}$$

where

$$\begin{bmatrix} s_{k+1} \\ v_{k+1} \\ P^{log}_k \end{bmatrix} = \begin{bmatrix} 1 & T & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^r_k \\ v^r_k \\ P^{log}_k \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & 1 \end{bmatrix} \eta^r_{k+1} \tag{45}$$

and where s_k and v_k represent a one-dimensional distance and velocity at time k respectively and η^r_k is a two-dimensional Gaussian process noise with zero mean and covariance $Q^r = diag([0.01\ \ 0.1])$ and ν^r_k is a discrete process noise with uniform distribution.

In this experiment, measurements are obtained from N identical acoustic sensors (microphones). The measurement function for the n-th sensor, as defined in (6), is

$$y^n_k = h^n_{RSS}(x_k, b^n) + \nu^n_k = log(1 - b^n_g) + P^{log}_k - \beta \log(r_n) + \nu^n_k \tag{46}$$

where $r_n = \sqrt{((p^x_k - (s^n_x - b^n_x))^2 + (p^y_k - (s^n_y - b^n_y))^2)}$ is a distance from the sensor to the target, $\beta = 2$ is the path loss exponent, the bias vector for the n-th sensor is defined as $b^n = [b^n_x\ \ b^n_y\ \ b^n_g]^T$, where b^n_x, b^n_y and b^n_g represent position biases in x and y sensor position and gain error respectively, s^n_x and s^n_y is a sensor position, p^x_k and p^y_k are positions of the target in global coordinates at time step k and ν^n_k represents a scalar Gaussian noise with zero mean and variance $\sigma^2_n = 0.2^2$.

A. Scenario description

In the considered scenario, $N = 12$ acoustic sensors are located as in Table I.

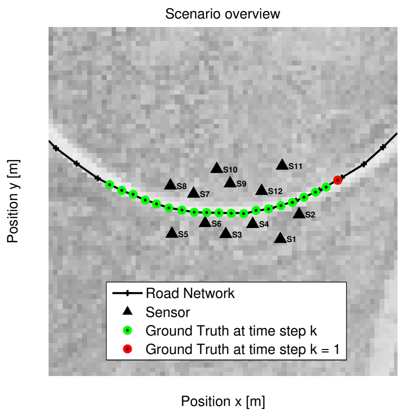


Fig. 3. Target trajectory and sensor positions

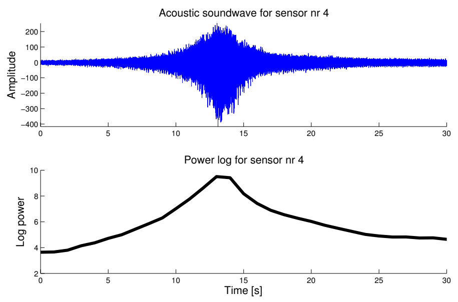


Fig. 4. Acoustic soundwave and logarithmic power measurements

TABLE I
SENSOR POSITIONS

n	1	2	3	4	5	6
s^n_x [m]	77.0	88.6	43.3	59.9	10.0	30.6
s^n_y [m]	7.0	22.2	10.0	16.3	10.2	16.8
n	7	8	9	10	11	12
s^n_x [m]	23.4	9.1	46.0	37.7	78.1	65.5
s^n_y [m]	34.9	40.0	41.4	50.3	52.3	36.7

A single target is moving on the road and $K = 30$ measurements are collected for every n-th sensor. The measurements are related to the target positions x_k at time steps $k = 1 : K$ with units in seconds. The scenario is presented on Figure 3. Acoustic power measurements are obtained by first taking the square of the sound signal from each sensor and then averaging it for each $T = 1$ second. As an example, the raw sound data and the acoustic power measurements generated from it, for sensor nr 6, are presented in Figure 4. The ground truth reference target state is obtained using GPS.

B. Results

In the experiment, position biases are added to the sensors by simply switching the positions of two pairs of sensors (3–4 and 7–8, respectively). Also the prior knowledge about biases is introduced for each n-th sensor as zero mean $b^n_\pi = [b^n_{x,\pi}\ \ b^n_{y,\pi}\ \ b^n_{g,\pi}]^T = [0\ \ 0\ \ 0]^T$ with covariance $P^n_\pi = diag([5^2\ \ 5^2\ \ 0.1^2])$. A total number of $Mc = 70$ Monte Carlo runs of the particle filter are performed for each case, with $I = 30$ number of EM iterations. The number of particles in each case is equal to $M = 500$. The results are described below.

The RMS error based on Monte Carlo runs is presented in Figure 5. Numerical results for selected biases are presented

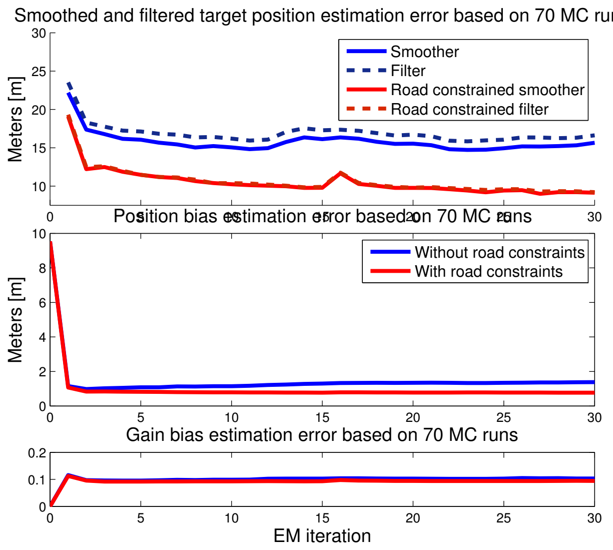


Fig. 5. Mean RMS error based on Monte Carlo runs

in Table II. Figure 6 presents true and predicted logarithmic

TABLE II
MONTE CARLO RESULTS FOR BIAS ESTIMATES IN SIMULATED SCENARIO

Bias index	True bias	EM algorithm Without constraints	EM algorithm With constraints
b^3_x	16.67	12.64 ± 3.24	11.03 ± 2.37
b^3_y	6.25	1.48 ± 4.23	4.13 ± 0.89
b^3_g	0	0.01 ± 0.04	-0.04 ± 0.01
b^4_x	-16.67	-15.3 ± 3.79	-16.42 ± 3.49
b^4_y	-6.25	-3.53 ± 3.7	-6.16 ± 1.29
b^4_g	0	0.04 ± 0.04	0.02 ± 0.01
b^7_x	-14.27	-10.26 ± 3.22	-11.44 ± 2.43
b^7_y	5.08	6.88 ± 6.09	4.59 ± 0.99
b^7_g	0	-0.01 ± 0.03	0.01 ± 0.01
b^8_x	14.27	16.09 ± 4.21	16.15 ± 3.4
b^8_y	-5.08	-9.74 ± 5.34	-10.11 ± 2.15
b^8_g	0	-0.16 ± 0.04	-0.15 ± 0.03

power measurements for sensor 4 before and after calibration for both unconstrained and road constrained version of the EM algorithm for one of the MC runs. Figure 7 presents filtered and smoothed estimates for $I = 30^{th}$ iteration of the EM algorithm (after calibration) compared to ground truth.

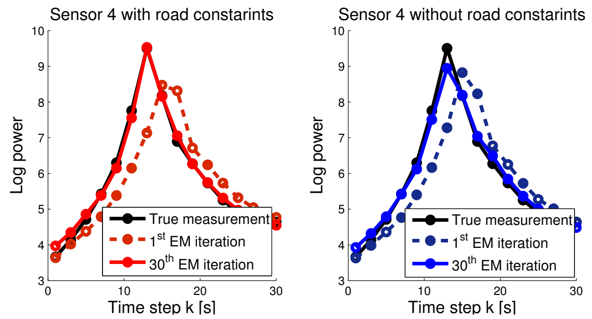


Fig. 6. Predicted power measurements (obtained from smoothed state estimates) before and after calibration

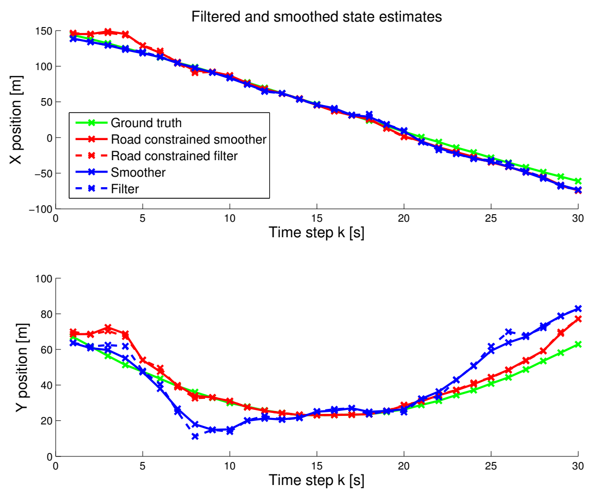


Fig. 7. Filtered and smoothed estimates after calibration together with ground truth

VII. CONCLUSION

In this paper, the EM algorithm was presented as a general solution to calibrate ground sensor network without special equipment, both natural and opportunistic targets can be used. Application of road constraints to the particle filter provides improvement in bias estimation quality, especially in terms of the standard deviation of the bias estimates. It provides also much better state estimates for the target in the scenario with missed detections.

ACKNOWLEDGMENT

The research leading to these results has received funding from the EUs Seventh Framework Programme under grant agreement no 238710. The research has been carried out in the MC IMPULSE project: https://mcimpulse.isy.liu.se/.

REFERENCES

[1] C.M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
[2] S. Dhar. Application of a recursive method for registration error correction in tracking with multiple sensors. In *American Control Conference, 1993*, pages 869 –874, june 1993.
[3] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107 –113, apr 1993.
[4] M. Ignagni. An alternate derivation and extension of friendland's two-stage kalman estimator. *Automatic Control, IEEE Transactions on*, 26(3):746 – 750, jun 1981.
[5] Z. Li, H. Leung, and L. Tao. Simultaneous registration and fusion of radar and esm by em-eks. In *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, pages 1130 –1134, july 2010.
[6] F. Gustafsson, U. Orguner, T.B. Schon, P. Skoglar, and R. Karlsson "Navigation and tracking of road-bound vehicles," in Handbook of Intelligent Vehicles, A. Eskandarian, Ed. Springer, Mar. 2012.
[7] B. Ristic, S. Arulampalam, and N. Gordon. *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, 2004.
[8] D. Salmond, M. Clark, R. Vinter, and S. Godsill. Ground target modelling, tracking and prediction with road networks. In *Information Fusion, 2007 10th International Conference on*, pages 1 –8, july 2007.
[9] T.B. Schön, A. Wills, and B. Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, January 2011.
[10] E. Sviestins. Online bias estimation for multisensor tracking. In *Proceedings of Information Decision and Control 99*, pages 221 –226, 1999.