

# Position Error Modeling Using Gaussian Mixture Distributions With Application to Comparison of Tracking Algorithms

Lidija Trailović and Lucy Y. Pao

Department of Electrical and Computer Engineering  
University of Colorado at Boulder, Boulder CO 80309-0425  
Lidija.Trailovic@Colorado.EDU and Lucy.Pao@Colorado.EDU

## Abstract

In this paper Gaussian mixtures are used to model the distribution of position error in tracking algorithms. An expectation maximization algorithm is constructed to estimate parameters of a  $k$ -component Gaussian mixture based on a sample set obtained from a tracking simulator. The modeling and parameter estimation approach is applied to position error data generated by several tracking algorithms including multi-target multi-sensor joint probabilistic data association and particle filters. The Gaussian mixture model yields significantly better likelihood of position error compared to the single Gaussian distribution. It is further shown how the mixture models can be used to efficiently compare tracking algorithms in terms of the root mean squared position error.

## 1 Introduction

Our goal is to efficiently evaluate tracking algorithms when the performance metric is the root mean squared position error (RMSE). It is frequently assumed that position error in tracking algorithms has a Gaussian distribution [13, 15]. The RMSE represents the standard deviation of the position error data collected from a tracking algorithm. Under the Gaussian distribution assumption, variance estimation and ranking methods that use properties of the Chi-squared and  $F$ -distribution [18] can be applied for ranking tracking algorithms in terms of the RMSE [15]. However, in many tracking problems it has been observed that the distribution of position error is more complex [8, 9] and therefore the standard tools for variance estimation and ranking do not apply.

In this paper we model the tracking position error using Gaussian mixture distributions [5]. To apply this modeling approach to efficient performance evaluation of tracking algorithms, we need (1) a method to estimate Gaussian mixture parameters from position error data, and (2) tools for variance estimation and ranking applicable to Gaussian mixtures.

It has been shown [16, 17] that a variance estimate based on data samples from a zero-mean Gaussian mixture distribution has the same properties as the distribu-

tion of a variance estimate based on a *reduced* number of samples from a single zero-mean Gaussian distribution. As a result, the tools for variance ranking based on properties of the  $F$ -distribution can still be applied to data originating from a zero-mean Gaussian *mixture* distribution simply by taking into account the reduced number of samples. The reduction factor has been found as a function of the Gaussian mixture parameters [16, 17].

In this paper, we focus on the estimation of Gaussian mixture parameters from position error data. Mixture learning algorithms are frequently based on the Expectation Maximization (EM) algorithm. The approach described in this paper is based on extensions of the ideas presented in [11, 19], where parameters of a  $k$ -component mixture are estimated through  $k$  successive applications of the EM algorithm.

The paper is organized as follows. The position error and performance evaluation of tracking algorithms in terms of RMSE are discussed in Section 2. The EM algorithm for estimation of Gaussian mixture parameters is described and simulation results based on synthetic data are presented in Section 3. Variance estimation for Gaussian mixture distributions is briefly discussed in Section 4. Application examples including (a) optimization of sensor processing order in the sequential multi-target multi-sensor joint probabilistic data association (MSJPDA) algorithm and (b) comparison of particle filters are presented in Section 5.

## 2 Position Error in Tracking Algorithms

To define the problem of tracking, let us consider the evolution of the state-space model of a target defined by a sequence  $\{\mathbf{x}(\tau)\}$ ,

$$\mathbf{x}(\tau) = \mathbf{f}_\tau(\mathbf{x}(\tau-1), \mathbf{w}(\tau-1)), \quad (1)$$

where  $\mathbf{f}_\tau(\cdot)$  is a possibly time-varying nonlinear function of the state  $\mathbf{x}(\tau)$ ,  $\mathbf{w}(\tau)$  is an i.i.d. process noise sequence, and  $\tau$  is the discrete time index. The tracking objective is to recursively estimate  $\mathbf{x}(\tau)$  based on measurements

$$\mathbf{z}(\tau) = \mathbf{h}_\tau(\mathbf{x}(\tau), \mathbf{v}(\tau)), \quad (2)$$

where  $\mathbf{h}_\tau(\cdot)$  is a possibly time-varying nonlinear function of the state, and  $\mathbf{v}(\tau)$  is an i.i.d. measurement noise sequence. In particular, tracking is seeking filtered estimates of  $\mathbf{x}(\tau)$  based on the set of all available measurements  $\mathbf{z}(1:\tau) = \{\mathbf{z}(i), i = 1, \dots, \tau\}$  up to time  $\tau$ .

---

This work was supported in part by the Office of Naval Research (Grant N00014-02-1-0136) and a University of Colorado Faculty Fellowship.

The estimate error  $\mathbf{e}(\tau)$  defined as the difference between the true state  $\mathbf{x}(\tau)$  and the state estimate  $\hat{\mathbf{x}}(\tau|\tau)$ ,

$$\mathbf{e}(\tau) = \mathbf{x}(\tau) - \hat{\mathbf{x}}(\tau|\tau), \quad (3)$$

is the random variable of interest. In the case of target tracking [13] the position error  $\mathbf{e}(\tau) = [e_x \ e_y]'(\tau)$  is often used to gauge tracking quality.

The traditional RMSE measure is used to quantify performance. Other performance metrics have been used in target tracking algorithms (*e.g.*, covariance control [9]), but the advantages of the RMSE are that it is an easily computed scalar and results for new algorithms can be readily compared to previously reported work.

Let  $e_\tau$  be a sample of the position error produced by a tracking algorithm at time step  $\tau$ . The collection of position errors  $\{e_1, \dots, e_n\}$  forms a data set. The standard deviation  $\sigma$  of the position error data is the *true* RMSE of a given tracking algorithm. The samples  $e_\tau$  present a discrete random variable. After a simulation run that generates  $n_{pts}$  data points we have

$$\bar{e} = \frac{1}{n_{pts}} \sum_{\tau} e_\tau \quad \text{and} \quad \hat{\sigma} = \widehat{\text{RMSE}} = \sqrt{\frac{1}{n_{pts} - 1} \sum_{\tau} e_\tau^2}, \quad (4)$$

where  $\bar{e}$  is the mean of that random variable and  $\hat{\sigma}$  is the standard deviation *estimate*. We have assumed that the position error is zero-mean,  $\bar{e} \approx 0$ . The standard deviation  $\hat{\sigma}$  equals the *measured* RMSE that a tracking algorithm produces as a sample of the performance metric for the given design and the set of parameters [15].

If we perform  $n$  simulation runs and update  $\hat{\sigma}$  after each run, by the strong law of large numbers,

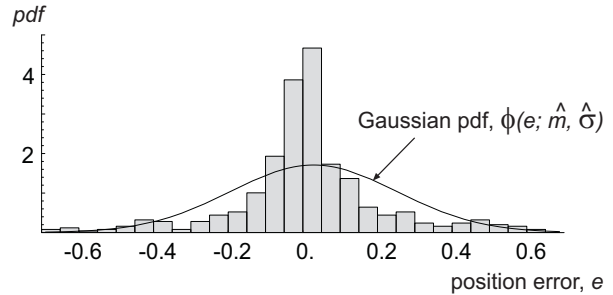
$$\sigma = \lim_{n \rightarrow \infty} \hat{\sigma} \quad (5)$$

with probability 1. This means that the measured standard deviation is asymptotically close to the true RMSE as the number  $n$  of simulation runs increases.

The collection of position errors  $\{e_1, \dots, e_n\}$  forms a data set that can further be used to estimate distribution parameters. A typical histogram of position error data obtained by a sequential MSJPDA algorithm is shown in Fig. 1. The error histogram is compared with the pdf of the Gaussian distribution with the mean  $\hat{m}$  and the standard deviation  $\hat{\sigma}$  computed from the data. The histogram shows high concentration of the data points around zero and a significant number of data points far from zero. Clearly, the Gaussian pdf does not model the data well. In Section 3 we show that a much better model can be achieved by using Gaussian mixture distributions.

### 3 Error Modeling Using Gaussian Mixtures

Finite mixture distributions provide a framework for modeling heterogeneous data [12] and have been popular in data clustering and pattern recognition. The assumption is that the data originates from a fixed number  $k$  of Gaussian densities. A random variable  $e$  is said to have



**Fig. 1:** A typical histogram of the position error obtained from the sequential MSJPDA simulator and the pdf of the Gaussian distribution with the mean  $\hat{m}$  and the standard deviation  $\hat{\sigma}$  computed from the data.

a  $k$ -component Gaussian mixture pdf

$$f_k(e) = \sum_{j=1}^k w_j \phi(e; m_j, \sigma_j), \quad \sum_{j=1}^k w_j = 1, w_j \geq 0 \quad (6)$$

where the number of components  $k$  is fixed, each  $\phi(e; m_j, \sigma_j)$  is a Gaussian pdf parameterized by the mean  $m_j$  and the standard deviation  $\sigma_j$ , and  $w_j$  are the mixing weights.

Given a training set  $\{e_1, \dots, e_n\}$  of i.i.d. data points sampled from the mixture, the modeling problem is to estimate the mixture parameters

$$\{w_j, m_j, \sigma_j\}, \quad j = 1, \dots, k \quad (7)$$

in order to maximize the log-likelihood

$$\Lambda_k = \log \prod_{i=1}^n f_k(e_i) = \sum_{i=1}^n \log f_k(e_i). \quad (8)$$

#### 3.1 Estimation of Gaussian Mixture Parameters

The problem of Gaussian mixture parameter estimation is frequently addressed using the well known expectation maximization (EM) algorithm [5, 11, 19]. Given a training set of data points  $\{e_1, \dots, e_n\}$ , the  $\ell$ -th iteration in the EM algorithm is performed as follows [19]:

$$p_{ji} = \frac{w_j^{\ell-1} \phi(e_i; m_j^{\ell-1}, \sigma_j^{\ell-1})}{f_k^{\ell-1}(e_i)}, \quad i=1, \dots, n, j=1, \dots, k \quad (9)$$

$$w_j^\ell = \frac{1}{n} \sum_{i=1}^n p_{ji}, \quad j = 1, \dots, k \quad (10)$$

$$m_j^\ell = \frac{\sum_{i=1}^n p_{ji} e_i}{\sum_{i=1}^n p_{ji}}, \quad j = 1, \dots, k \quad (11)$$

$$\sigma_j^\ell = \sqrt{\frac{\sum_{i=1}^n p_{ji} (e_i - m_j^\ell)^2}{\sum_{i=1}^n p_{ji}}}, \quad j = 1, \dots, k. \quad (12)$$

In (9),  $p_{ji}$  is the conditional probability that the data point  $e_i$  is sampled from the  $j$ -th Gaussian component, given the mixture parameters  $\{w_j^{\ell-1}, m_j^{\ell-1}, \sigma_j^{\ell-1}\}$ . The mixture parameters are then updated to the new values  $\{w_j^\ell, m_j^\ell, \sigma_j^\ell\}$  according to (10)-(12).

It has been shown [5] that the EM algorithm produces a monotonically non-decreasing log-likelihood sequence,

$$\Lambda_k^\ell \geq \Lambda_k^{\ell-1}. \quad (13)$$

Thus, from an initial guess for the mixture parameters,  $\{w_j, m_j, \sigma_j\} = \{w_j^0, m_j^0, \sigma_j^0\}$ , under certain assumptions [5], the algorithm converges to a set of parameters that produces a local maximum of the log-likelihood.

The main difficulties in applying the EM algorithm for the mixture parameter estimation are: (a) the true number of mixing components  $k$  is unknown, (b) there is no widely accepted method for parameter initialization, and (c) the algorithm can converge to one of many possibly inferior local maxima of the likelihood. Our approach in addressing these issues is based on modifications of the EM algorithm for multivariate Gaussian mixture learning [11, 19] to take advantage of the previously observed properties of the position error data generated by target tracking algorithms. The mixture components are expected to have significant overlaps, with means that are relatively close to zero. It should be noted that, in general, parameter estimation is more difficult in mixtures where the Gaussian pdf's have poor separation [4].

The EM algorithm presented here is performed by iteratively adding one component to the mixture. The algorithm starts with  $k = 1$ , where

$$f_1(e) = \phi(e; m, \sigma) \quad (14)$$

is the Gaussian pdf with the unbiased estimates of the mean  $m$  and the standard deviation  $\sigma$  of the single Gaussian distribution for the given training data set

$$m = \frac{1}{n} \sum_{i=1}^n e_i, \quad \sigma = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (e_i - m)^2}. \quad (15)$$

The algorithm proceeds by adding one component to the mixture. Assuming that the parameters of a  $k$ -component mixture  $f_k(e)$  have been found, a new component  $\phi(e; m_{k+1}^0, \sigma_{k+1}^0)$  is added to form the mixture

$$f_{k+1}(e) = (1 - a) f_k(e) + a \phi(e; m_{k+1}^0, \sigma_{k+1}^0). \quad (16)$$

The initial values of the mixture weight  $a = w_{k+1}^0$  and standard deviation  $\sigma_{k+1}^0$  for the added component are

$$a = \begin{cases} 0.5, & \text{if } k = 1 \\ \frac{2}{k+1}, & \text{if } k \geq 2 \end{cases} \quad (17)$$

$$\sigma_{k+1}^0 = b \min_{1 \leq j \leq k} \sigma_j, \quad (18)$$

where  $0 < b < 1$ . The motivation for this initialization is an observation that adding a component with a narrower pdf improves the likelihood. The mean of the added component is initialized using the single-parameter optimization

$$m_{k+1}^0 = \arg \max_m \Lambda_{k+1}^0(m), \quad (19)$$

where the global search for  $m$  that maximizes the likelihood is performed over the range

$$\min_{1 \leq j \leq k} m_j - \max_{1 \leq j \leq k} \sigma_j \leq m \leq \max_{1 \leq j \leq k} m_j + \max_{1 \leq j \leq k} \sigma_j. \quad (20)$$

With the parameter initialization given by (16)-(20), the EM iteration (9)-(12) is performed to converge to the updated parameters  $\{w_j, m_j, \sigma_j\}$  of the  $(k+1)$ -component

mixture. The algorithm stops when  $\Lambda_{k+1} \leq \Lambda_k$ , *i.e.*, when the log-likelihood does not improve with the added component, or when a predetermined maximum number of components  $k_{max}$  is reached.

### 3.2 Simulation Results Using Synthetic Data

The proposed EM algorithm was evaluated through a variety of simulations using synthetic data generated by sampling Gaussian mixtures with known parameters and here we present some representative results.

A training set of  $n = 200$  data points was generated from a Gaussian mixture with  $k = 3$  components and known parameters. These parameters produce a pdf with a shape resembling that of a typical position error histogram obtained from the sequential MSJPDA tracking algorithm. Fig. 2 compares the training data histogram with the true and estimated 3-component mixture pdf's.

For comparison, using the same training data set, we performed a Monte Carlo test repeating the EM iteration 1000 times for  $k = 3$ , each starting with a random set of initial parameters  $\{w_j^0, m_j^0, \sigma_j^0\}$ . The initial parameter values were sampled uniformly in the following ranges:

$$0 < w_j^0 < 1, \quad \sum_{j=1}^k w_j = 1, \quad (21)$$

$$\begin{aligned} \min_{1 \leq j \leq 3} m_{j,true} - \max_{1 \leq j \leq 3} \sigma_{j,true} &< m_j^0 \\ &< \max_{1 \leq j \leq 3} m_{j,true} + \max_{1 \leq j \leq 3} \sigma_{j,true}, \end{aligned} \quad (22)$$

$$0.3 \min_{1 \leq j \leq 3} \sigma_{j,true} < \sigma_j^0 < 3 \max_{1 \leq j \leq 3} \sigma_{j,true}. \quad (23)$$

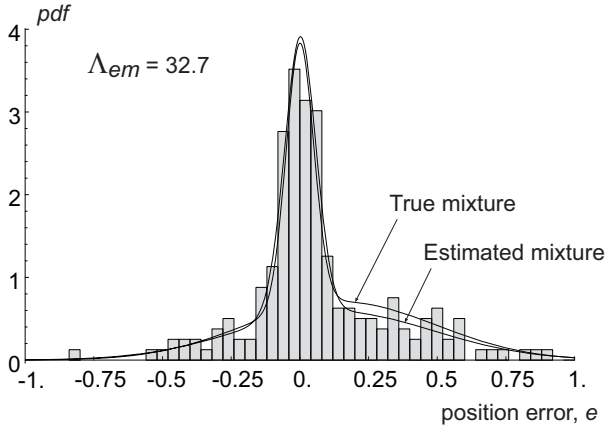
The histogram of all 1000 log-likelihood values from the Monte Carlo test is shown in Fig. 3. The maximum log-likelihood is  $\Lambda_{max} = 37.6$ . The log-likelihood  $\Lambda_{em} = 32.7$  obtained by the proposed EM algorithm is in the top 20% of the values obtained by the long Monte Carlo search. Note that the EM iteration initialized with random parameters as in the Monte Carlo test can produce significantly smaller log-likelihoods compared to the proposed EM algorithm (9)-(12) initialized with (16)-(20).

### 4 Variance Ranking

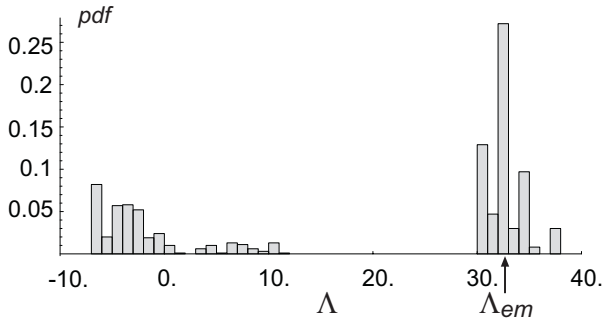
Chi-squared and  $F$ -distributions are used for ranking variances of data originating from Gaussian distributions [15, 18]. For data originating from  $k$ -component zero-mean Gaussian mixture distributions similar tools were developed and reported in [16, 17]. It was shown that for large  $n$  the distribution of the variance estimate based on  $n$  samples of data from a zero-mean Gaussian mixture distribution has *the same properties* as the distribution of the variance estimate based on  $n^* = \bar{\gamma} n$  samples from a zero-mean single Gaussian distribution, where the reduction factor  $\bar{\gamma}$  can be found as a function of the mixture parameters as [16, 17]:

$$\bar{\gamma} = \frac{2}{3 \left( \sum_{j=1}^k w_j \sigma_j^4 \right) - 1}. \quad (24)$$

Note that  $\bar{\gamma}$  does not depend on  $n$ . Therefore, variance ranking tools for samples originating from a Gaussian



**Fig. 2:** Training data histogram and pdf's of the true and estimated Gaussian mixtures with  $k = 3$  components.



**Fig. 3:** Histogram of the log-likelihood values obtained in 1000 runs of the EM algorithm randomly initialized.

distribution can be applied to the class of zero-mean Gaussian mixture distributions simply by taking into account the reduction factor  $\bar{\gamma}$  in the degrees of freedom.

Let us consider two stochastic processes  $\theta_u$  and  $\theta_v$  that produce random variables  $e_u$  and  $e_v$  with zero-mean Gaussian mixture distributions. Given the samples  $\{e_{u1}, \dots, e_{un_u}\}$  from  $\theta_u$  and  $\{e_{v1}, \dots, e_{vn_v}\}$  from  $\theta_v$ , the respective variance estimates  $\hat{\sigma}_{eq,u}^2$  and  $\hat{\sigma}_{eq,v}^2$  are obtained from data. If  $\hat{\sigma}_{eq,u} < \hat{\sigma}_{eq,v}$ , the problem of computing the confidence level, *i.e.*, the *a posteriori* probability  $p_{uv}$  that  $\sigma_{eq,u} < \sigma_{eq,v}$  is finding

$$p_{uv} = P \left\{ \left( \frac{\sigma_{eq,v}}{\sigma_{eq,u}} \right)^2 > 1 \mid \hat{\sigma}_{eq,u}, \hat{\sigma}_{eq,v} \right\}. \quad (25)$$

Assuming that  $n_u$  and  $n_v$  are large, the confidence level  $p_{uv}$  can be computed as

$$\begin{aligned} p_{uv}^* &= 1 - \text{CDF} \left( n_u^*, n_v^*, 1/\xi_{uv}^2 \right) \\ &= \int_{1/\xi_{uv}^2}^{\infty} f(n_u^*, n_v^*, x) dx \end{aligned} \quad (26)$$

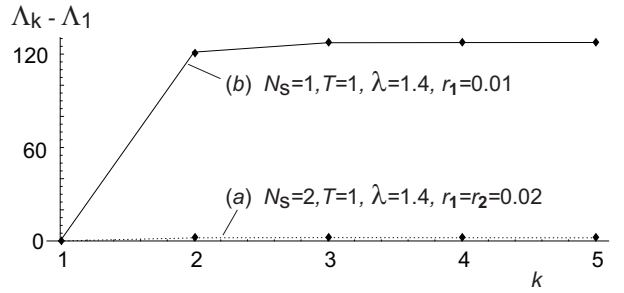
where

$$n_u^* = \bar{\gamma}_u n_u, \quad n_v^* = \bar{\gamma}_v n_v, \quad (27)$$

$f(n_u^*, n_v^*, x)$  is the pdf of the random variable having  $F_{n_u^*-1, n_v^*-1}$  distribution with  $(n_u^* - 1)$  degrees of freedom in the numerator and  $(n_v^* - 1)$  degrees of freedom in the denominator,

$$\xi_{uv} = \frac{\hat{\sigma}_{eq,v}}{\hat{\sigma}_{eq,u}}, \quad (28)$$

and  $\text{CDF}(\cdot)$  is the cumulative density function of the  $F$ -distribution random variable.



**Fig. 4:** Excess log-likelihood vs. the number of components in the Gaussian mixture distributions for two sets of data generated by the sequential MSJPDA tracking simulator.

## 5 Comparison of Tracking Algorithms

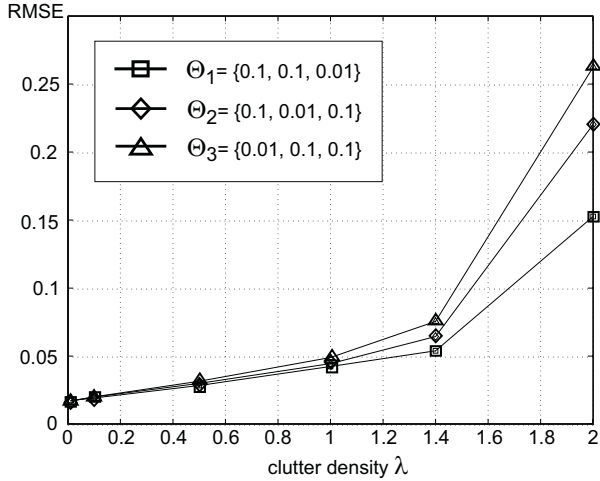
The position error modeling of Section 3 and the variance ranking tools of Section 4 were used in two application examples: (a) optimization of sensor processing order in the sequential MSJPDA algorithm and (b) comparison of particle filters. Our results are summarized below.

### 5.1 Sequential MSJPDA Tracking Algorithm

Considering the tracking model (1)-(2), the sequential MSJPDA [7] is characterized by linear time-invariant tracking, Gaussian process and measurement noises with zero-means and known covariances  $\mathbf{Q}$  (same for all targets) and  $\mathbf{R}_s$  (different for each sensor  $s$ ), and state estimates produced using a Kalman filter. Simulator parameters are as in [13],  $\mathbf{Q} = q\mathbf{I}$ , and  $\mathbf{R}_s = r_s\mathbf{I}$ . Nonlinearities in the algorithm are caused by the existence of clutter (false measurements uniformly distributed with density  $\lambda$ ) and data association (that resolves uncertainties about the origin of the received measurements) [2, 3].

The number of components  $k$  in a Gaussian mixture needed to model the position error measurements of the tracking algorithm was determined by observing the sequence of log-likelihoods  $\Lambda_k$  generated by the EM algorithm of Section 3. Fig. 4 shows log-likelihood as a function of  $k$  for two different tracking scenarios. Case (a) is an example of very good tracking with  $N_s = 2$  sensors ( $r_1 = r_2 = 0.02$ ) tracking  $T = 1$  target in clutter with density  $\lambda = 1.4$ . Position error measurements in this case are very well modeled with a single Gaussian pdf ( $k = 1$ ). The increase of Gaussian mixture components ( $k = 2, 3, 4, 5$ ) does not increase the log-likelihood significantly. In case (b), which is an example with a loss of track,  $N_s = 1$  sensor ( $r_1 = 0.01$ ) is tracking  $T = 1$  target in the same clutter. To model position error in this case, Gaussian mixtures provide much better likelihoods than a single Gaussian pdf. A 2-component Gaussian mixture models the data quite well, and further increase in the number of mixture components does not increase the log-likelihood significantly.

Let us further consider the sequential MSJPDA algorithm with  $N_s = 3$  sensors tracking  $T = 4$  targets. The compared designs are defined as ordered sets of sensor noise parameters (two identical sensors with  $r = 0.1$  and



**Fig. 5:** RMSE of the sequential MSJPDA algorithm as a function of clutter density  $\lambda$  for three designs.

$\lambda$	0.01	0.1	0.5	1.0	1.4	2.0
$\theta_b$	$\theta_2$	$\theta_1$	$\theta_1$	$\theta_1$	$\theta_1$	$\theta_1$
$N_{runs}$	100	100	93	85	21	15
conf. level	0.43	0.86	0.98	0.98	0.98	0.99

**Table 1:** The selected best ranking design  $\theta_b$ , the total number of simulation runs  $N_{runs}$ , and the confidence level as functions of clutter density  $\lambda$ . The maximum allowed number of runs is  $N_{max} = 100$ .

one better sensor with  $r = 0.01$ ). The objective is to find the best design in terms of the RMSE for various clutter densities  $\lambda$ . The three designs are

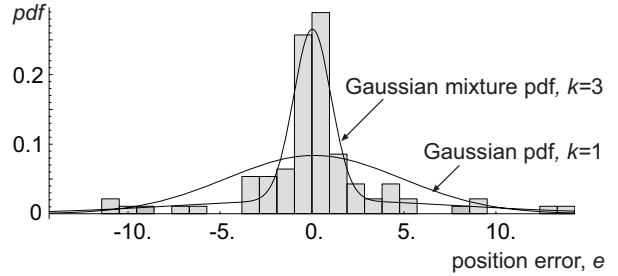
$$\begin{aligned} \theta_1 &= \{0.1, 0.1, 0.01\}, \quad \theta_2 = \{0.1, 0.01, 0.1\}, \\ \theta_3 &= \{0.01, 0.1, 0.1\}. \end{aligned} \quad (29)$$

The variance ranking tools [16, 17] reviewed in Section 4 were used in the optimization algorithm of [15] to yield the results in Fig. 5 and Table 1. For low clutter densities  $\lambda$ , the order of sensor processing does not affect the tracking performance. Since the RMSE's do not differ significantly, the best ranking design  $\theta_b$  is selected with a small confidence level when the maximum number of simulation runs is exhausted. For higher  $\lambda$ , the order of sensor processing affects the tracking more significantly. The best ranking design is consistently the one where the best sensor is processed last. These conclusions are consistent with findings of [13] where simpler 2-sensor systems were analyzed.

## 5.2 Comparison of Particle Filters

Particle filters [1, 6] are sequential Monte Carlo methods based on point mass (or ‘‘particle’’) representations of probability densities, which can be applied to any (possibly nonlinear and non-Gaussian) system model, and which generalize traditional Kalman filtering methods.

Consider a one-dimensional random process with pdf  $p(x)$ . If  $N_p$  particles  $x^i$ ,  $i = 1, \dots, N_p$  are drawn from



**Fig. 6:** Position error histogram for the SIR particle filter and the estimated Gaussian mixture pdf's for  $k = 1$  and 3.

the process, a discrete approximation of  $p(x)$  is

$$p(x) \approx \sum_{i=1}^{N_p} w^i \delta(x - x^i), \quad (30)$$

where  $\delta(\cdot)$  is the Dirac delta function and  $w^i$  is the weight of the  $i$ -th particle. A particle filter operates by recursively propagating a set of  $N_p$  particles through the model (1)-(2). Variations in particle filters differ in the details of numerical procedures used to recursively propagate the set of particles. Three popular particle filters considered here are [1]: (a) Sampling Importance Resampling (SIR) particle filter, (b) Auxiliary Sampling Importance Resampling particle filter (APF), and (c) Regularized particle filter (RPF).

Our objective is to use the tools from Sections 3 and 4 to efficiently compare the performance of the particle filter algorithms in the following illustrative example:

$$x_\tau = f_\tau(x_{\tau-1}, \tau) + v_\tau \quad (31)$$

$$z_\tau = \frac{x_\tau^2}{20} + n_\tau \quad (32)$$

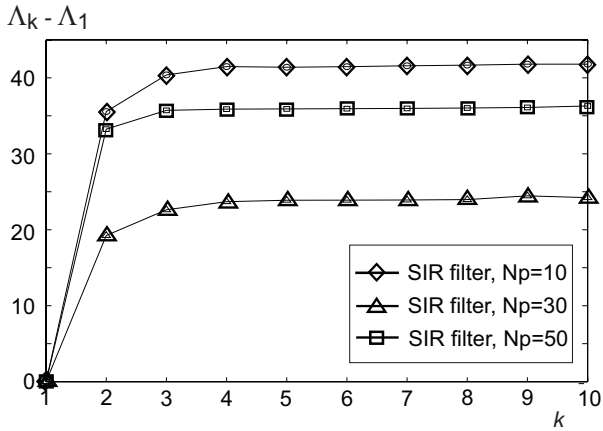
where

$$f_\tau(x_{\tau-1}, \tau) = \frac{x_{\tau-1}}{2} + \frac{25x_{\tau-1}}{1 + x_{\tau-1}^2} + 8 \cos(1.2\tau) \quad (33)$$

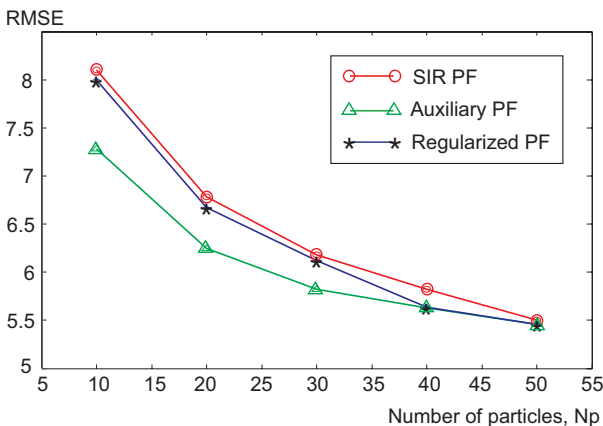
and where  $v_\tau$  and  $n_\tau$  are zero-mean Gaussian random variables with variances  $Q_\tau$  and  $R_\tau$ , respectively. In the presented example  $Q_\tau = 10$  and  $R_\tau = 1$ . This tracking problem has been considered as a benchmark example in [1, 10], where it was shown that classical Kalman or extended Kalman filters do not perform well.

Fig. 6 shows a state error histogram of a representative run of the SIR particle filter with the pdf's of a single Gaussian and a 3-component Gaussian mixture. The mixture parameters are estimated using the EM algorithm of Section 3. It is evident that the 3-component Gaussian mixture is a much better fit to the position error histogram than the single Gaussian. Fig. 7 shows log-likelihood as a function of the number of components  $k$  in a Gaussian mixture used to model data for the SIR particle filter. It can be observed that a 3-component Gaussian mixture models the data quite well.

The computational effort in particle filters is directly proportional to the number of particles  $N_p$  sequentially propagating through the filter. It is therefore of interest to compare the filters using the number of particles as a parameter. Again, the tools of Section 4 and [17] were



**Fig. 7:** Excess log-likelihood as a function of components  $k$  in the Gaussian mixtures for three sets of data generated by SIR particle filters with  $N_p = 10, 30,$  and  $50$  particles.



**Fig. 8:** RMSE as a function of the number of particles for the SIR, auxiliary, and regularized particle filters.

used to perform the ranking, and the results are shown in Table 2 and Fig. 8. In this example, the APF outperforms the RPF and the basic SIR filter. The difference in performance is more visible when a smaller number of particles  $N_p$  is used in the filters. As  $N_p$  increases, the performance difference diminishes and all three filters perform almost the same for sufficiently large  $N_p$ .

## 6 Conclusion

Gaussian mixture distributions have been shown to more accurately model position error in tracking algorithms. An expectation maximization (EM) algorithm is constructed to estimate parameters of a Gaussian mixture with  $k$  components based on data from a target tracking simulator. The developed method is applied to model position error distributions in examples from a sequential multi-sensor multi-target data association algorithm and three variations of particle filters. Results show that Gaussian mixture distributions produce significantly better likelihoods compared to single Gaussian distributions. Furthermore, we show how to efficiently compare different tracking algorithms in terms of the root mean squared position error.

Algorithm	RMSE for $N_p$				
	10	20	30	40	50
SIR	8.10	6.78	6.18	5.82	5.50
APF	7.28	6.25	5.82	5.63	5.46
RPF	8.00	6.67	6.12	5.64	5.46
conf. level	99.0	95.2	95.7	61.7	40.1
$N_{runs}$	395	440	846	5000	5000

**Table 2:** Ranking of particle filters in terms of the RMSE for different numbers of particles  $N_p = 10, 20, 30, 40,$  and  $50$ . The table shows the estimated RMSE, the confidence level, and the total number of simulation runs  $N_{runs}$  performed, with the maximum allowed number of runs  $N_{max} = 5000$ .

## References

- [1] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *IEEE Trans. Sig. Proc.*, Feb. 2002.
- [2] Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association*, Academic Press Inc., 1988.
- [3] Y. Bar-Shalom and E. Tse, "Tracking in a Cluttered Environment with Probabilistic Data Association," *Automatica*, Sept. 1975.
- [4] S. Dasgupta, "Learning Gaussian Mixtures," *Proc. IEEE Symp. on Foundations of Computer Science*, Oct. 1999.
- [5] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum-Likelihood from Incomplete Data Via the EM Algorithm," *J. Royal Stat. Soc. Ser. B*, 1977.
- [6] A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*, Springer-Verlag Inc., 2001.
- [7] C. W. Frei and L. Y. Pao, "Alternatives to Monte-Carlo Simulation Evaluations of Two Multi-sensor Fusion Algorithms," *Automatica*, Jan. 1998.
- [8] L. Hong and S. Cong, "Bias Phenomenon and Compensation in Multiple Target Tracking Algorithms," *Mathematical and Computer Modelling*, 2000.
- [9] M. Kalandros and L. Y. Pao, "Covariance Control for Multi-sensor Systems," *IEEE Trans. Aero. Elec. Sys.*, Oct. 2002.
- [10] G. Kitagawa, "Non-Gaussian State-Space Modeling of Non-stationary Time Series," *J. Amer. Stat. Assoc.*, Dec. 1987.
- [11] J. Q. Li and A. R. Barron, "Mixture Density Estimation," *Advances in Neural Information Processing Systems 12*, The MIT Press, 2000.
- [12] G. J. McLachlan and D. Peel, *Finite Mixture Models*, Wiley, 2000.
- [13] L. Y. Pao and L. Trailović, "The Optimal Order of Processing Sensor Information in Sequential Multi-sensor Fusion Algorithms," *IEEE Trans. Aut. Ctrl.*, Aug. 2000.
- [14] D. L. Alspach and H. W. Sorenson, "Nonlinear Bayesian Estimation Using Gaussian Sum Approximation," *IEEE Trans. Aut. Ctrl.*, Aug. 1972.
- [15] L. Trailović and L. Y. Pao, "Computing Budget Allocation for Optimization of Sensor Processing Order in Sequential Multi-sensor Fusion Algorithms," *Proc. Amer. Ctrl. Conf.*, June 2001.
- [16] L. Trailović and L. Y. Pao, "Variance Estimation and Ranking for Gaussian Mixture Distributions in Target Tracking Applications," *Proc. Conf. Decision and Ctrl.*, Dec. 2002.
- [17] L. Trailović, *Ranking and Optimization of Target Tracking Algorithms*, Ph.D. thesis, University of Colorado at Boulder, 2002.
- [18] S. B. Vardeman, *Statistics for Engineering Problem Solving*, IEEE Press, 1994.
- [19] N. Vlassis and A. Likas, "A Greedy EM Algorithm for Gaussian Mixture Learning," *Neural Proc. Letters*, Feb. 2002.