# A comparative study of multiple-model algorithms for maneuvering target tracking*

**Ryan R. Pitre** **Vesselin P. Jilkov** **X. Rong Li**
Department of Electrical Engineering
University of New Orleans
New Orleans, LA 70148, USA
{rrpitre, vjilkov, xli}@uno.edu, 504-280-{7399, 6179, 7416, 3950 (fax)}

**Abstract**

Many multiple-model (MM) algorithms for tracking maneuvering targets are available, but there are few comparative studies of their performance. This work compares seven MM algorithms for maneuvering target tracking in terms of tracking performance and computational complexity. Six of them are well known and widely used. They are the autonomous multiple-model algorithm, generalized pseudo-Bayesian algorithm of first order (GPB1), and of second order (GPB2), interacting multiple-model (IMM) algorithm, B-best based MM algorithm, and Viterbi-based MM algorithm. Also considered is the reweighted interacting multiple-model algorithm, which was developed recently. The algorithms were compared using three scenarios. The first scenario consists of two segments of tangential acceleration while the second scenario consists of two segments of normal acceleration. Both of these scenarios have maneuvers that are represented by one of the models in the model set. The third scenario, however, has a single maneuver that consists of a tangential and a normal acceleration. This type of maneuver is not covered by the model set and is used to see how the algorithms react to a maneuver outside of the model set. Based on the study, there is no clear-cut best algorithm but the IMM algorithm has the best computational complexity among the algorithms that have acceptable tracking errors. It also showed a remarkable robustness to model mismatching, and appears to be the top choice if the computational cost is of concern.

**Keywords**: Target tracking, Multiple model estimation, Comparison

## 1 Introduction

It is beneficial to use more than one motion model in a tracking algorithm when the dynamics of the target are not known. In order to account for the uncertainty of the target's dynamics, a multiple-model target-tracking algorithm runs a set of filters that models non-maneuver target motion along with several possible maneuvers. Then, the algorithm fuses the output of those filters for an overall estimate. Many MM target tracking algorithms have been proposed and each of them fuses its estimates differently. A comprehensive survey and an in-depth analysis of the state-of-the-art MM estimation methodology for maneuvering target tracking is provided in the survey paper [1]. For a detailed treatment of the MM algorithms, concerning their theoretical background, underlying assumptions, structures, pros and cons, implementations, applications, a vast amount of references, and more, the reader is referred to [1]. The literature concerning various applications of MM algorithms is abundant. However, few comparative studies on the performance of different MM algorithms have been reported (e.g., [2], [3], [4]), and to the best of the authors knowledge, no comprehensive study including, if not all, at least several of the most important MM techniques and algorithms is available.

The primary objective of this paper is to investigate and compare the performance of mainstream MM algorithms by simulation of some more realistic target tracking scenarios. The secondary purpose is to provide an accessible and systematic description of these MM algorithms that would facilitate practitioners in their implementation for various applications.

Seven algorithms are simulated and compared using three scenarios. The first scenario consists of two segments of tangential acceleration while the second scenario consists of two segments of normal acceleration. Both of these scenarios have maneuvers that are represented by one of the models in the model set. The third scenario, however, has a single maneuver that consists of a tangential and a normal acceleration. This type of maneuver is not covered by the model set and is used

*Research Supported in part by ARO grant W911NF-04-1-0274, NASA/LEQSF via grant (2001-4)-01, and NSFC via grant 60328305.

to see how the algorithms react to a maneuver outside of the model set. The latter scenario is perhaps the most important in judging the performance of the algorithms in practice, since in reality— especially in a non-cooperative environment—the tracker can hardly know beforehand the "models" (patterns) of the target motion.

The rest of the paper is organized as follows. Section 2 gives a self-contained summary of the MM algorithms implemented. Sections 3 describes the design of MM algorithms. The simulation scenarios, results, and their analysis are presented in Section 4. Section 5 provides a conclusion.

## 2 Summary of MM Algorithms

The MM estimation algorithms are described for a Markov jump linear system

$$x_{k+1} = F_k^{(i)} x_k + G_k^{(i)} w_k^{(i)}, \quad \text{with } w_k^{(i)} \sim \mathcal{N}(\bar{w}_k^{(i)}, Q_k^{(i)}) \tag{1}$$

$$z_k = H_k^{(i)} x_k + v_k^{(i)}, \quad \text{with } v_k^{(i)} \sim \mathcal{N}(\bar{v}_k^{(i)}, R_k^{(i)}) \tag{2}$$

where superscript $(i)$ denotes quantities pertinent to model $m^{(i)}$ in the model set $\mathbb{M} = \{m^{(1)}, m^{(2)}, \ldots, m^{(M)}\}$, and the jumps, if any, of the system mode have the following transition probabilities

$$P\{m_{k+1}^{(j)} | m_k^{(i)}\} = \pi_{ij}, \qquad \forall m^{(i)}, m^{(j)}, k \tag{3}$$

where $m_k^{(i)}$ denotes the event that model $m^{(i)}$ matches the system *mode* in effect at time $k$. $z^k = \{z_1, \ \ldots, z_k\}$ and $m^k = \{m_1^{(i_1)}, \ \ldots, m_k^{(i_k)}\}$ denote the measurement sequence and a model sequence (history), respectively, for the interval $[1, k]$. $\mathcal{N}(y; \bar{y}, P_y) = \frac{1}{|2\pi P_y|^{1/2}} \exp[-\frac{1}{2}(y-\bar{y})' {P_y}^{-1} (y-\bar{y})]$, or $\mathcal{N}(\bar{y}, P_y)$ for short, denotes the Gaussian distribution.

### 2.1 Autonomous MM (AMM) Algorithm [5]

The autonomous MM (AMM) algorithm provides the MMSE estimate $\hat{x}_{k|k} = E[x_k | z^k]$ under the assumption that the system mode is time invariant and equal to some model $m^{(i)}$, $\forall k$. This means that $\pi_{ij} = \delta_{i-j} = \begin{cases} 1, \ i = j \\ 0, \ i \neq j \end{cases}$. The AMM algorithm runs a conditional Kalman filter for each model in the model set and evaluates the posterior probability of each model. The overall fused estimate is obtained as a sum of the conditional estimates weighted by their corresponding model probabilities. The conditional filters operate independently, in an autonomous manner—no information is exchanged among the filters—and the overall estimate serves only for output. One recursive cycle $(k-1) \to k$ of AMM is given by the following three steps (see also Table 1).

Step 1. *Model-conditioned filtering.* Run a Kalman filter (see (17) in the Appendix) for each model $m^{(i)}$ with initial condition $(\hat{x}_{k-1|k-1}^{(i)}, P_{k-1|k-1}^{(i)})$.

Step 2. *Model probability update.* Evaluate the posterior probability for each model $m^{(i)}$

$$\mu_k^{(i)} = \frac{\mu_{k-1}^{(i)} L_k^{(i)}}{\sum_{j=1}^{M} \mu_{k-1}^{(j)} L_k^{(j)}}, \quad \text{with likelihood } \ L_k^{(i)} = N(\tilde{z}_k^{(i)}; 0, S_k^{(i)}) \tag{4}$$

Step 3. *Estimate fusion.* Evaluate the overall output (estimate and covariance)

$$\hat{x}_{k|k} = \sum_{i=1}^{M} \hat{x}_{k|k}^{(i)} \mu_k^{(i)}, \quad P_{k|k} = \sum_{i=1}^{M} [P_{k|k}^{(i)} + (\hat{x}_{k|k} - \hat{x}_{k|k}^{(i)})(\hat{x}_{k|k} - \hat{x}_{k|k}^{(i)})'] \mu_k^{(i)} \tag{5}$$

### 2.2 First-Order Generalized Pseudo-Bayesian (GPB1) Algorithm [6]

The first-order generalized pseudo-Bayesian (GPB1) algorithm (and all MM algorithms considered in the sequel) assume that the system mode may jump in the model set according to the Markov transition (3). GPB1 again, as AMM does, runs $M$ conditional filters; however these filters are *reinitialized* with the common overall estimate from the previous time-step. The model probability update also takes into account the possible model transitions through the transition probability matrix (3). One recursive cycle $(k-1) \to k$ of GPB1 is as follows (see also Table 1).

| AMM | GPB1 |
|---|---|
| 1. Model-conditioned filtering (for $i = 1, 2, ..., M$) | 1. Model-conditioned filtering (for $i = 1, 2, ..., M$) |
| $\{\hat{x}^{(i)}_{k-1\|k-1}, P^{(i)}_{k-1\|k-1}, z_k\} \longrightarrow \{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}, \tilde{z}^{(i)}_k, S^{(i)}_k\}$ | $\{\hat{x}_{k-1\|k-1}, P_{k-1\|k-1}, z_k\} \longrightarrow \{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}, \tilde{z}^{(i)}_k, S^{(i)}_k\}$ |
| 2. Model probability update (for $i = 1, 2, ..., M$) | 2. Model probability update (for $i = 1, 2, ..., M$): |
| $\{\mu^{(i)}_{k-1}, \tilde{z}^{(i)}_k, S^{(i)}_k\} \longrightarrow \{\mu^{(i)}_k\}$ | $\{\pi_{ij}, \mu^{(i)}_{k-1}, \tilde{z}^{(i)}_k, S^{(i)}_k\} \longrightarrow \{\mu^{(i)}_k\}$ |
| 3. Estimate fusion | 3. Estimate fusion |
| $\{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}, \mu^{(i)}_k\} \longrightarrow \{\hat{x}_{k\|k}, P_{k\|k}\}$ | $\{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}, \mu^{(i)}_k\} \longrightarrow \{\hat{x}_{k\|k}, P_{k\|k}\}$ |

Table 1: AMM and GPB1 Algorithms

Step 1. *Model-conditioned filtering.* Run a Kalman filter, (17), for each model $m^{(i)}$ with the *common initial condition* $(\hat{x}_{k-1|k-1}, P_{k-1|k-1})$.
Step 2. *Model probability update.* Evaluate the posterior probability for each model $m^{(i)}$

$$\mu^{(i)}_k = \frac{\mu^{(i)}_{k|k-1} L^{(i)}_k}{\sum_{j=1}^{M} \mu^{(j)}_{k|k-1} L^{(j)}_k}, \quad \mu^{(i)}_{k|k-1} = \sum_{j=1}^{M} \pi_{ji} \mu^{(j)}_{k-1} \quad \text{with } L^{(i)}_k = \mathcal{N}(\tilde{z}^{(i)}_k; 0, S^{(i)}_k) \tag{6}$$

Step 3. *Estimate fusion.* See (5).

Theoretically, GPB1 can be viewed as an approximation of the optimal full hypothesis tree (FHT) MM estimator that considers the possible models only at the latest time instant and merges all previous model sequences into one.

### 2.3 Second–Order Generalized Pseudo-Bayesian (GPB2) Algorithm [7]

The second-order generalized pseudo-Bayesian (GPB2) algorithm considers the possible models only at the latest two time instants and merges all previous model sequences that end at the same model. Thus GPB2 maintains $M$ initial conditional (model-matched) estimates. For each initial estimate it runs $M$ conditional filters corresponding to the possible model transitions (a total of $M^2$ conditional filtering operations), and then merges the updated estimates that end at the same model. One recursive cycle $(k-1) \to k$ of GPB2 is given by the following four steps (see also Table 2).
Step 1. *Model-conditioned filtering.* Run a Kalman filter, (17), for each model $m^{(i)}$ with initial condition $(\hat{x}^{(j)}_{k-1|k-1}, P^{(j)}_{k-1|k-1})$.
Step 2. *Model-conditioned reinitialization.* Evaluate the merging probabilities

$$\mu^{j|i}_{k-1|k} = \frac{L^{(ji)}_k \pi_{ji} \mu^{(j)}_{k-1}}{\mu^{(i)}_{k|k-1}}, \quad \mu^{(i)}_{k|k-1} = \sum_{j=1}^{M} L^{(ji)}_k \pi_{ji} \mu^{(j)}_{k-1}, \quad \text{with likelihood } L^{(ji)}_k = \mathcal{N}(\tilde{z}^{(ji)}_k; 0, S^{(ji)}_k) \tag{7}$$

and merged estimates/covariances

$$\hat{x}^{(i)}_{k|k} = \sum_{j=1}^{M} \hat{x}^{(ji)}_{k|k} \mu^{j|i}_{k-1|k}, \quad P^{(i)}_{k|k} = \sum_{j=1}^{M} [P^{(ji)}_{k|k} + (\hat{x}^{(i)}_{k|k} - \hat{x}^{(ji)}_{k|k})(\hat{x}^{(i)}_{k|k} - \hat{x}^{(ji)}_{k|k})'] \mu^{j|i}_{k-1|k}$$

Step 3. *Model probability update*

$$\mu^{(i)}_k = \frac{\mu^{(i)}_{k|k-1}}{\sum_{j=1}^{M} \mu^{(j)}_{k|k-1}} \tag{8}$$

Step 4. *Estimate fusion.* See (5).

| GPB2 | IMM |
|---|---|
| 1. Model-conditioned filtering $(i,j=1,2,\ldots,M)$ | 1. Model-conditioned reinitialization $(i=1,2,\ldots,M)$ |
| $\{\hat{x}^{(j)}_{k-1\|k-1}, P^{(j)}_{k-1\|k-1}, z_k\} \longrightarrow \{\hat{x}^{(ji)}_{k\|k}, P^{(ji)}_{k\|k}, \tilde{z}^{(ji)}_k, S^{(ji)}_k\}$ | mixing probabilities: |
| 2. Model-conditioned reinitialization (for next cycle) | $\{\pi_{ij}, \mu^{(i)}_{k-1}\} \longrightarrow \{\mu^{j\|i}_{k-1}, \mu^{(i)}_{k\|k-1}\}$ |
| merging probabilities: | mixing: |
| $\{\pi_{ij}, \mu^{(j)}_{k-1}, \tilde{z}^{(ji)}_k, S^{(ji)}_k\} \longrightarrow \{\mu^{j\|i}_{k-1\|k}, \mu^{(i)}_{k\|k-1}\}$ | $\{\hat{x}^{(j)}_{k-1\|k-1}, P^{(j)}_{k-1\|k-1}, \mu^{j\|i}_{k-1}\} \longrightarrow \{\bar{x}^{(i)}_{k-1\|k-1}, \bar{P}^{(i)}_{k-1\|k-1}\}$ |
| merging: | 2. Model-conditioned filtering $(i=1,2,\ldots,M)$ |
| $\{\mu^{(j\|i)}_{k-1\|k}, \hat{x}^{(ji)}_{k\|k}, P^{(ji)}_{k\|k}\} \longrightarrow \{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}\}$ | $\{\bar{x}^{(i)}_{k-1\|k-1}, \bar{P}^{(i)}_{k-1\|k-1}\} \longrightarrow \{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}, \tilde{z}^{(i)}_k, S^{(i)}_k\}$ |
| 3. Model probability update for $(i=1,2,\ldots,M)$ | 3. Model probability update $(i=1,2,\ldots,M)$ |
| $\{\mu^{(i)}_{k\|k-1}\} \longrightarrow \{\mu^{(i)}_{k\|k}\}$ | $\{\mu^{(i)}_{k\|k-1}, \tilde{z}^{(i)}_k, S^{(i)}_k\} \longrightarrow \{\mu^{(i)}_k\}$ |
| 4. Estimate fusion | 4. Estimate fusion |
| $\{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}, \mu^{(i)}_k\} \longrightarrow \{\hat{x}_{k\|k}, P_{k\|k}\}$ | $\{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}, \mu^{(i)}_k\} \longrightarrow \{\hat{x}_{k\|k}, P_{k\|k}\}$ |

Table 2: GPB2 and IMM Algorithms

### 2.4 Interacting MM (IMM) Algorithm [8]

The interacting MM (IMM) algorithm, like GPB1, runs $M$ conditional filters. However, each filter is *individually reinitialized* with the estimate, referred to as mixed estimate, conditioned on the model at the current time instant. After updating the filters IMM provides the overall output through estimate fusion, but maintains the conditional estimates for the next time step as GPB2 does. One cycle of IMM is given by the following four steps (see also Table 2).

Step 1. *Model-conditioned reinitialization.* Evaluate the mixing probabilities

$$\mu^{j|i}_{k-1} = \frac{1}{\mu^{(i)}_{k|k-1}} \pi_{ji} \mu^{(j)}_{k-1}, \quad \text{with } \mu^{(i)}_{k|k-1} = \sum_{j=1}^{M} \pi_{ji} \mu^{(j)}_{k-1} \tag{9}$$

and mix estimates/covariances

$$\bar{x}^{(i)}_{k-1|k-1} = \sum_{j=1}^{M} \hat{x}^{(j)}_{k-1|k-1} \mu^{j|i}_{k-1}, \quad \bar{P}^{(i)}_{k-1|k-1} = \sum_{j=1}^{M} [P^{(j)}_{k-1|k-1} + (\bar{x}^{(i)}_{k-1|k-1} - \hat{x}^{(j)}_{k-1|k-1})(\bar{x}^{(i)}_{k-1|k-1} - \hat{x}^{(j)}_{k-1|k-1})'] \mu^{j|i}_{k-1} \tag{10}$$

Step 2. *Model-conditioned filtering.* Kalman filter, (17), for each model $m^{(i)}$ with initial condition $(\bar{x}^{(i)}_{k-1|k-1}, \bar{P}^{(i)}_{k-1|k-1})$.
Step 3. *Model probability update.* See (6).
Step 4. *Estimate fusion.* See (5).

### 2.5 Reweighted IMM (RIMM) Algorithm [9]

The reweighted IMM (RIMM) algorithm is a recursive approximation of EM-based MAP state estimation for the system (1), (2). RIMM pretty much resembles the structure of IMM but it has essentially different mixing formula, and does the input mixing after model-conditioned prediction along all possible $M^2$ model transitions, i.e., reinitializes the update stage of the filters. One recursive cycle $(k-1) \to k$ of RIMM is given in by the following four steps (see Table 3).

Step 1. *Model-conditioned filter update reinitialization.* The mixing probabilities are given by (9). The model-conditioned prediction is

$$\hat{x}^{(ji)}_{k|k-1} = F^{(i)}_k \hat{x}^{(j)}_{k-1|k-1} + G^{(i)}_k \bar{w}^{(i)}_{k-1}, \quad P^{(ji)}_{k|k-1} = \frac{\pi_{ji}}{\mu^{(i)}_{k|k-1}} F^{(i)}_k P^{(j)}_{k-1|k-1} F^{(i)\prime}_k + G^{(i)}_k Q^{(i)}_{k-1} G^{(i)\prime}_k \tag{11}$$

and the input prediction mixing is given by[1]

$$\hat{x}^{(i)}_{k|k-1} = P^{(i)}_{k|k-1} \sum_{j=1}^{M} P^{(j,i)-1}_{k|k-1} \hat{x}^{(ji)}_{k|k-1} \mu^{j|i}_{k-1}, \quad P^{(i)-1}_{k|k-1} = \sum_{j=1}^{M} P^{(ji)-1}_{k|k-1} \mu^{j|i}_{k-1} \tag{12}$$

Step 2. *Model-conditioned filter update.* Kalman filter update, from (17), for each model $m^{(i)}$ with $(\hat{x}^{(i)}_{k|k-1}, P^{(i)}_{k|k-1})$.
Step 3. *Model probability update.* See (6).
Step 4. *Estimate fusion*.

$$\hat{x}_{k|k} = P_{k|k} \sum_{i=1}^{M} P^{(i)-1}_{k|k} \hat{x}^{(i)}_{k|k} \mu^{(i)}_{k}, \quad P^{-1}_{k|k} = \sum_{i=1}^{M} P^{(i)-1}_{k|k} \mu^{(i)}_{k}$$

Note that RIMM fusion formula essentially uses the conditional filter covariances along with the model weights.

Table 3: RIMM algorithm

1. Model-conditioned filter update reinitialization ($i = 1, 2, \ldots, M$)
   mixing probabilities: $\{\pi_{ij}, \mu^{(i)}_{k-1}\} \longrightarrow \{\mu^{j|i}_{k-1}, \mu^{(i)}_{k|k-1}\} \quad (j = 1, \ldots, M)$
   prediction: $\{\hat{x}^{(j)}_{k-1|k-1}, P^{(j)}_{k-1|k-1}, \mu^{(j)}_{k|k-1}\} \longrightarrow \{\hat{x}^{(ji)}_{k|k-1}, P^{(ji)}_{k|k-1}\} \quad (j = 1, \ldots, M)$
   mixing: $\{\hat{x}^{(ji)}_{k|k-1}, P^{(ji)}_{k|k-1}, \mu^{j|i}_{k-1}\} \longrightarrow \{\hat{x}^{(i)}_{k|k-1}, P^{(i)}_{k|k-1}, \mu^{(i)}_{k|k-1}\}$
2. Model-conditioned filter update ($i = 1, 2, \ldots, M$)
   $\{\hat{x}^{(i)}_{k|k-1}, P^{(i)}_{k|k-1}\} \longrightarrow \{\hat{x}^{(i)}_{k|k}, P^{(i)}_{k|k}, \tilde{z}^{(i)}_{k}, S^{(i)}_{k}\}$
3. Model probability update ($i = 1, 2, \ldots, M$)
   $\{\mu^{(i)}_{k|k-1}, \tilde{z}^{(i)}_{k}, S^{(i)}_{k}\} \longrightarrow \{\mu^{(i)}_{k}\}$
4. Estimate fusion
   $\{\hat{x}^{(i)}_{k|k}, P^{(i)}_{k|k}, \mu^{(i)}_{k}\} \longrightarrow \{\hat{x}_{k|k}, P_{k|k}\}$

### 2.6 B-Best MM (BMM) Algorithm [10]

The B-best MM (BMM) algorithm is hard decision and model sequence-based. At each time $k$ it maintains $B$ sequence-conditioned estimates selected as the "best" (in terms, e.g., of probability or likelihood) among the set of all possible model sequences $m^k$. In a recursive version, on receipt of a new measurement BMM runs $BM$ conditional filtering operations corresponding to all possible model transitions at that time and evaluates their likelihoods (resp. posterior probabilities). Then it selects the best $B$ of them (the others are discarded) and renormalizes the probability weights. The output can be the weighted sum of the survived filters, or the best of them. One recursive cycle $(k-1) \rightarrow k$ of the BMM algorithm is given in Table 4.

### 2.7 Viterbi MM (VMM) Algorithm [3]

The Viterbi MM algorithm (VMM) is also a hard decision and model sequence-based algorithm. It attempts to find the best model sequence for every model in the model set. At each time step it maintains $M$ sequence conditional estimates—the "best" one that ends at $m^{(i)}$ for each model $m^{(i)}$. On receipt of a new measurement VMM runs a number of $M^2$ conditional filters corresponding to all possible model transitions at that time and evaluates their posterior probabilities. Then, for each model in the model set it selects the best one of those ending at that model (the others are discarded). After renormalization

[1]Using inverse covariance matrices in the prediction mixing formula (12) was found to cause numerical problems if $\pi_{ji} = 0$ and $Q^{(i)}_{k-1}$ is very small ($\approx 0$) since $P^{(ji)}$ from (11) becomes $\approx 0$.

| BMM | VMM |
|---|---|
| 1. Model-conditioned filtering $(j = 1, \ldots, B;\ i = 1,\ \ldots, M)$<br>$\{\hat{x}^{(j)}_{k-1\|k-1}, P^{(j)}_{k-1\|k-1}\}^B_{j=1} \longrightarrow \{\hat{x}^{(ji)}_{k\|k}, P^{(ji)}_{k\|k}, \tilde{z}^{(ji)}_k, S^{(ji)}_k\}$ | 1. Model-conditioned filtering $(j, i = 1, \ldots, M)$<br>$\{\hat{x}^{(j)}_{k-1\|k-1}, P^{(j)}_{k-1\|k-1}\} \longrightarrow \{\hat{x}^{(ji)}_{k\|k}, P^{(ji)}_{k\|k}, \tilde{z}^{(ji)}_k, S^{(ji)}_k\}$ |
| 2. Sequence probability update $(j = 1, \ldots, B;\ i = 1,\ \ldots, M)$<br>$\{\pi_{ij}, \mu^{(j)}_{k-1}, \tilde{z}^{(ji)}_k, S^{(ji)}_k\} \longrightarrow \{\lambda^{(ji)}_k\}$ | 2. Sequence probability update $(j, i = 1, 2,\ \ldots, M)$<br>$\{\pi_{ij}, \mu^{(j)}_{k-1}, \tilde{z}^{(ji)}_k, S^{(ji)}_k\} \longrightarrow \{\lambda^{(ji)}_k\}$ |
| 3. Selection of the $B$-best (largest $\lambda^{(ji)}_k$) to form $\{\lambda^{(j)}_k\}^B_{j=1}$<br>$\{\lambda^{(ji)}_k\} \longrightarrow \{\lambda^{(j)}_k\}^B_{j=1}$ | 3. Selection of the largest $\lambda^{(ji)}_k$ $(i = 1, \ldots, M)$<br>$\{\lambda^{(ji)}_k\} \longrightarrow \{\lambda^{(i)}_k\}, \quad \lambda^{(i)}_k = \max_j \lambda^{(ji)}_k$ |
| 4. Probability renormalization<br>$\{\lambda^{(j)}_k\} \longrightarrow \{\mu^{(j)}_k\}$ | 4. Probability renormalization<br>$\{\lambda^{(i)}_k\} \longrightarrow \{\mu^{(i)}_k\}$ |
| 5. Estimate Fusion<br>$\{\hat{x}^{(j)}_{k\|k}, P^{(j)}_{k\|k}, \mu^{(j)}_k\}^B_{j=1} \longrightarrow \{\hat{x}_{k\|k}, P_{k\|k}\}$ | 5. Estimate fusion<br>$\{\hat{x}^{(i)}_{k\|k}, P^{(i)}_{k\|k}, \mu^{(i)}_k\} \longrightarrow \{\hat{x}_{k\|k}, P_{k\|k}\}$ |

Table 4: BMM and VMM Algorithms

of the probabilities the output can be the weighted sum of the survived filters, or the best of them. One recursive cycle $(k-1) \rightarrow k$ of VMM is given in Table 4.

An alternative implementation of VMM is also given in [11].

## 3 MM Tracking Algorithms' Design

### 3.1 Model Set

Each MM tracking algorithm uses nine target motion models. The model set consists of one non-maneuver model, which is the (nearly) constant velocity (CV) model, and eight maneuver models. Four of the maneuver models model a constant tangential acceleration (CTA), i.e., in the velocity direction. The CTA filters do not estimate the acceleration but rather they use a preset acceleration that acts as a constant input. The CTA models used by the filters are CTA$\left(67\ m/s^2\right)$, CTA$\left(-67\ m/s^2\right)$, CTA$\left(33\ m/s^2\right)$, CTA$\left(-33\ m/s^2\right)$, which model constant accelerations at approximately $\pm 7g$ and $\pm 3.5g$. Another set of accelerations modeled are those in the direction normal to the velocity, which model constant turns (CT). They are CT$(14\ \mathrm{deg}/s)$, CT$(-14\ \mathrm{deg}/s)$, CT$(7\ \mathrm{deg}/s)$, CT$(-7\ \mathrm{deg}/s)$. These nine models cover eight different target maneuver cases and the case with no maneuver.

All models are in the generic state-space form

$$x_k = Fx_{k-1} + Gu_{k-1} + Gw_k, \quad w_k \sim \mathcal{N}(0, Q) \tag{13}$$

with state vector $x = (\mathsf{x}, \dot{\mathsf{x}}, \mathsf{y}, \dot{\mathsf{y}})'$ for all models and specifications as given below [12].

**CV:**

$$F_{CV} = \mathrm{diag}[F_2, F_2],\ F_2 = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix};\quad G = \mathrm{diag}[G_2, G_2],\ G_2 = \begin{bmatrix} T^2/2 \\ T \end{bmatrix};\quad u_{k-1} = 0;\quad T = 1\ s \tag{14}$$

This is a non-manuever model.

**CTA:**

$$F_{CTA} = F_{CV};\quad u_{k-1} = \hat{u}_{k-1} = \frac{a_t}{\sqrt{\hat{\dot{\mathsf{x}}}_{k-1} + \hat{\dot{\mathsf{y}}}_{k-1}}} \begin{bmatrix} \hat{\dot{\mathsf{x}}}_{k-1} \\ \hat{\dot{\mathsf{y}}}_{k-1} \end{bmatrix},\quad a_t \text{ is given} \tag{15}$$

Each of the CTA models assumes a preset, tangential acceleration $a_t$ ($\approx \pm 67,\ \pm 33\ m/s^2$ respectively) and the conditional Kalman filters do not estimate acceleration. The direction of the velocity (the unit vector above) is evaluated at each iteration $k$ using the velocity $(\hat{\dot{\mathsf{x}}}_{k-1}, \hat{\dot{\mathsf{y}}}_{k-1})$ estimated at time $k-1$.

**CT:**

$$F_{CT} = \begin{bmatrix} 1 & \frac{\sin\omega T}{\omega} & 0 & \frac{1-\cos\omega T}{\omega} \\ 0 & \cos\omega T & 0 & -\sin\omega T \\ 0 & \frac{1-\cos\omega T}{\omega} & 1 & \frac{\sin\omega T}{\omega} \\ 0 & \sin\omega T & 0 & \cos\omega T \end{bmatrix}, \quad u_{k-1} = 0 \tag{16}$$

The CT models assume a nearly constant turn to the right or left. The values for the turn rate, $\omega$ ($\approx \pm 14, \ \pm 7$ deg $/s$ respectively), were determined approximately to cover $7g$, and $3.5g$ maneuvers at an expected speed of $330m/s$.

### 3.2 Process Noises

Since the Kalman filter was used for conditional filtering, the covariance $Q$ =diag$[\sigma_w^2, \sigma_w^2]$ of the process noise $w = [w_{\mathsf{x}}, w_{\mathsf{y}}]'$ are also needed. Each MM tracking algorithm used the same $\sigma_w = 2.25m/s^2$ for the CV model and $\sigma_w = 50m/s^2$for all maneuver models. These values were experimentally tuned to provide acceptable tradeoff between the errors during steady state, peak errors, and quick transition between models.

### 3.3 Model Probabilities

The model probability of the CV model was initialized to $\mu_0^{\text{CV}} = 0.99$ and the remaining model probabilities were all initialized to the value $\mu_0 = (1- \mu_0^{\text{CV}})/8 = 0.00125$.

The transition probability matrix used is given below.

$$\Pi = \begin{bmatrix} 0.99 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.00125 \\ 0.193 & 0.75 & 0.001 & 0.05 & 0.001 & 0.00125 & 0.00125 & 0.00125 & 0.00125 \\ 0.193 & 0.001 & 0.75 & 0.001 & 0.05 & 0.00125 & 0.00125 & 0.00125 & 0.00125 \\ 0.00369 & 0.00125 & 0.00001 & 0.99 & 0.00005 & 0.00125 & 0.00125 & 0.00125 & 0.00125 \\ 0.00369 & 0.00001 & 0.00125 & 0.00005 & 0.99 & 0.00125 & 0.00125 & 0.00125 & 0.00125 \\ 0.193 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.75 & 0.001 & 0.05 & 0.001 \\ 0.193 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.001 & 0.75 & 0.001 & 0.05 \\ 0.00369 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.00001 & 0.99 & 0.00005 \\ 0.00369 & 0.00125 & 0.00125 & 0.00125 & 0.00125 & 0.00001 & 0.00125 & 0.00005 & 0.99 \end{bmatrix}$$

where the models are enumerated in the following order:

$$\text{CV}, \text{CT}(14 \text{ deg}/s), \text{CT}(-14 \text{ deg}/s), \text{CT}(7 \text{ deg}/s), \text{CT}(-7 \text{ deg}/s), \text{CTA}(67\, m/s^2), \text{CTA}(-67\, m/s^2), \text{CTA}(33\, m/s^2), \text{CTA}(-33\, m/s^2)$$

## 4 Simulations

### 4.1 Test scenarios & Performance measures

**Ground truth.** The ground-truth trajectories were generated by first-order Euler dicretization of the generic 2D continuous-time curvilinear motion model from kinematics [12] as

$$\mathsf{x}_{k+1} = \mathsf{x}_k + Tv_k \cos\phi_k + w_{x_k}, \quad \mathsf{y}_{k+1} = \mathsf{y}_k + Tv_k \sin\phi_k + w_{y_k}$$
$$v_{k+1} = v_k + Ta_{t_k} + w_{v_k}, \qquad \phi_{k+1} = \phi_k + Ta_{n_k}/v_k + w_{\phi_k}$$

where $(\mathsf{x}, \mathsf{y}), v, \phi, a_t, a_n$ denote the target position (in the Cartesian coordinates), speed, heading, and tangential and normal accelerations, respectively. The sampling period is $T$ and $k$ is the time index.

For each Monte Carlo run $i = 1, 2, \ldots, M$, $M = 100$

$$\mathsf{x}_0^{(i)} \sim \mathcal{N}(\bar{\mathsf{x}}_0, \sigma_{\mathsf{x}_0}^2), \quad \mathsf{y}_0^{(i)} \sim \mathcal{N}(\bar{\mathsf{y}}_0, \sigma_{\mathsf{y}_0}^2), \quad v_0^{(i)} \sim \mathcal{N}(\bar{v}_0, \sigma_{v_0}^2), \quad \phi_0^{(i)} \sim \mathcal{N}(\bar{\phi}_0, \sigma_{\phi_0}^2);$$
$$w_{x_k}^{(i)} \sim \mathcal{N}(0, \sigma_{w_x}^2), \quad w_{y_k}^{(i)} \sim \mathcal{N}(0, \sigma_{w_y}^2), \quad w_{v_k}^{(i)} \sim \mathcal{N}(0, \sigma_{w_v}^2), \quad w_{\phi_k}^{(i)} \sim \mathcal{N}(0, \sigma_{w_\phi}^2)$$

with $\bar{\mathsf{x}}_0 = 120km, \sigma_{\mathsf{x}_0} = 5km$; $\bar{\mathsf{y}}_0 = 150km$, $\sigma_{\mathsf{y}_0} = 5km$; $\bar{v}_0 = 300m/s, \sigma_{v_0} = 5m/s$; $\bar{\phi}_0 = 30\,\text{deg}, \sigma_{\phi_0} = 1\,\text{deg}$;
$\sigma_{w_x} = 5m$, $\sigma_{w_y} = 5m$, $\sigma_{w_v} = 1m/s$, $\sigma_{w_\phi} = 0.01\,\text{deg}$

Three maneuver scenarios were investigated in the simulation: maneuvers with tangential acceleration , normal acceleration, and simultaneous tangential and normal, respectively (see Fig. 1 for quiver (velocity vector) plots with sampling period of $3s$). For each scenario the tangential and normal accelerations $a_{t_k}$ and $a_{n_k}$ were as given in Table 5.

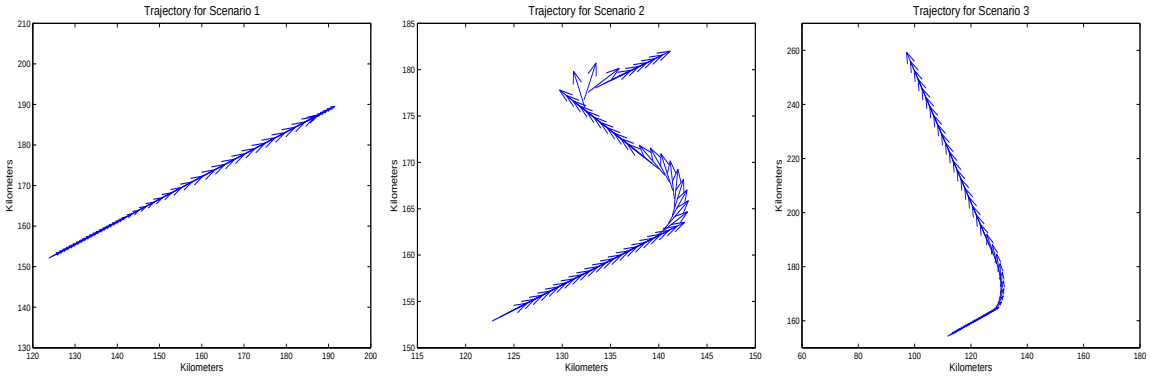


Figure 1: Ground-truth trajectories

Table 5: Maneuver Scenarios' Parameters

| | Scenario 1 | | Scenario 2 | | Scenario 3 | |
|---|---|---|---|---|---|---|
| $k$ | $a_{t_k}$ | $a_{n_k}$ | $a_{t_k}$ | $a_{n_k}$ | $a_{t_k}$ | $a_{n_k}$ |
| $1-59$ | 0 | 0 | 0 | 0 | 0 | 0 |
| $60-89$ | 20 | 0 | 0 | 20 | 30 | 30 |
| $90-119$ | 0 | 0 | 0 | 0 | 0 | 0 |
| $120-129$ | -20 | 0 | 0 | -20 | 0 | 0 |
| $130-150$ | 0 | 0 | 0 | 0 | 0 | 0 |

**Performance measures.** The *accuracy* of the algorithms was measured in terms of position and velocity root-mean-square errors (RMSE):

$$\text{RMSE}_k^{(p)} = \left( \frac{1}{M} \sum_{i=1}^{M} [(\mathsf{x}_k^{(i)} - \hat{\mathsf{x}}_{k|k}^{(i)})^2 + (\mathsf{y}_k^{(i)} - \hat{\mathsf{y}}_{k|k}^{(i)})^2] \right)^{1/2}, \quad \text{RMSE}_k^{(v)} = \left( \frac{1}{M} \sum_{i=1}^{M} [(\dot{\mathsf{x}}_k^{(i)} - \hat{\dot{\mathsf{x}}}_{k|k}^{(i)})^2 + (\dot{\mathsf{y}}_k^{(i)} - \hat{\dot{\mathsf{y}}}_{k|k}^{(i)})^2] \right)^{1/2}$$

where $(\mathsf{x}_k^{(i)}, \mathsf{y}_k^{(i)}, \dot{\mathsf{x}}_k^{(i)}, \dot{\mathsf{y}}_k^{(i)})$ and $(\hat{\mathsf{x}}_{k|k}^{(i)}, \hat{\mathsf{y}}_{k|k}^{(i)}, \hat{\dot{\mathsf{x}}}_{k|k}^{(i)}, \hat{\dot{\mathsf{y}}}_{k|k}^{(i)})$ denote the true and estimate states, respectively, in run $i$ at time $k$.

### 4.2 Results and Comparison

**Estimation accuracy.** Comparative time-plots of the position and velocity RMSEs for each scenario are shown in Figures 2, 3, and 4, respectively. The results are summarized in Tables 6–9 as follows.

Table 6 gives the "steady-state" errors during non-maneuvering obtained by time averaging over all non-maneuver trajectory segments of all scenarios. In terms of position error the algorithms rank as follows: (VMM, AMM, BMM)–best, (GPB2, IMM, RIMM)–middle, GPB1–worst. Regarding the velocity error the ranking is almost the same: (VMM, AMM, BMM)–best, (RIMM, GPB2, IMM)–middle, GPB1–worst. As expected, the "no-reinitilialization" AMM and the hard-decision-based BMM and VMM provide smallest errors. The main reason for this is that the non-maneuver (CV) Kalman filter, correct in this case, is "self-reinitialized" in these algorithms, and thus the effect of the incorrect maneuver filters is ignored in the reinitialization. The collective reinitialization of GPB2, IMM, RIMM, and GPB1, on the other hand, leads to some increase in the error since the maneuver filters still take some of the probability weight from the CV filter. While for GPB2, IMM, and RIMM this error increase tends to be acceptable, for GPB1 it appears excessively large. It seems that the GPB1 reinitialization is rather crude.

The peak dynamic errors at maneuver onset and maneuver termination are given in Tables 7, 8 for position and velocity, respectively. In these tables $\text{M}_{ij}$ denotes "Maneuver $j$ in Scenario $i$".

At maneuver onset the ranking of the algorithms with respect to the overall (averaged over all scenarios) error is: (IMM, GPB2, GPB1), (VMM, RIMM, BMM, AMM) for position; and (GPB2, IMM), (GPB1, AMM), (VMM, BMM), RIMM for velocity. The peak maneuver-on error is due to the mismatch between the true, maneuver mode of motion, and the

Table 6: Average RMSEs during Non-Maneuvering

| | AMM | GPB1 | GPB2 | IMM | RIMM | BMM | VMM |
|---|---|---|---|---|---|---|---|
| Position | 58.7 | 79.9 | 63.3 | 65.8 | 67.6 | 58.7 | 58.5 |
| Velocity | 7.4 | 18.5 | 10.4 | 11.1 | 9.3 | 7.9 | 7.4 |

nonmaneuver filter, which is still dominant in an MM configuration right at, and some time after, the maneuver onset. When this error becomes suffuciently large (for a particular MM algorithm) it forces the MM algorithm to find a more likely (dominant) model/filter in the MM configuration. Clearly, the reinitialization mechanisms of the soft-decision based GPB2, IMM, GPB1 help provide *faster* and *smoother transition* to a maneuver model (and thus cut the peak error) as compared with the hard-decision based VMM, BMM (and AMM, as well), which apparently "switch" to a maneuvering only after a larger error has built up. The RIMM also shows a rather poor response (large peak errors) to maneuver onsets. This is probably due to the direct use of the covariance information in its reinitialization—in order to abandon the nonmaneuver filter not only should its bias become large (resulting in a small likelihood, and thus model probability) but also its covariance should become sufficiently large.

Table 7: Peak Position RMSEs

| | AMM | | GPB1 | | GPB2 | | IMM | | RIMM | | BMM | | VMM | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | on | off | on | off | on | off | on | off | on | off | on | off | on | off |
| $M_{11}$ | 190.8 | 194.3 | 124.4 | 109.6 | 142.4 | 126.4 | 139.2 | 128.4 | 172.8 | 114.5 | 159.2 | 131.9 | 167.3 | 143.1 |
| $M_{12}$ | 218.1 | 195.1 | 240.4 | 214.9 | 194.2 | 102.6 | 190.9 | 103.2 | 225.9 | 126.1 | 239.9 | 99.7 | 221.9 | 108.1 |
| $M_{21}$ | 196.5 | 198.8 | 122.0 | 102.0 | 153.1 | 126.0 | 148.5 | 125.6 | 171.6 | 117.9 | 170.3 | 135.4 | 177.9 | 138.5 |
| $M_{22}$ | 207.7 | 203.3 | 213.8 | 193.3 | 181.9 | 113.4 | 179.7 | 114.1 | 198.8 | 128.5 | 221.5 | 108.4 | 208.1 | 121.1 |
| $M_{31}$ | 223.1 | 198.2 | 212.0 | 130.2 | 203.6 | 150.8 | 199.9 | 150.0 | 249.6 | 125.1 | 239.3 | 166.2 | 241.0 | 175.9 |
| Ave | 207.2 | 197.9 | 182.5 | 150.0 | 175.0 | 123.8 | 171.6 | 124.3 | 203.7 | 122.4 | 206.0 | 128.3 | 203.3 | 137.3 |

Table 8: Peak Velocity RMSEs

| | AMM | | GPB1 | | GPB2 | | IMM | | RIMM | | BMM | | VMM | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | on | off | on | off | on | off | on | off | on | off | on | off | on | off |
| $M_{11}$ | 100.8 | 107.0 | 78.2 | 51.0 | 86.6 | 75.13 | 86.9 | 77.1 | 106.5 | 60.8 | 91.0 | 77.0 | 96.6 | 87.8 |
| $M_{12}$ | 177.1 | 136.4 | 205.5 | 153.5 | 178.8 | 81.99 | 180.9 | 84.2 | 211.9 | 113.4 | 211.5 | 66.3 | 195.0 | 76.7 |
| $M_{21}$ | 99.1 | 113.8 | 76.5 | 46.5 | 87.7 | 65.35 | 87.4 | 65.8 | 104.2 | 62.4 | 92.6 | 72.9 | 95.9 | 87.4 |
| $M_{22}$ | 174.5 | 128.3 | 187.5 | 149.7 | 170.3 | 85.99 | 173.5 | 88.5 | 199.8 | 117.2 | 196.2 | 81.7 | 184.4 | 79.8 |
| $M_{31}$ | 162.0 | 125.8 | 156.1 | 80.3 | 153.2 | 141.59 | 155.3 | 129.7 | 188.2 | 125.1 | 167.1 | 143.4 | 169.2 | 157.9 |
| Ave | 142.7 | 122.2 | 140.8 | 96.2 | 135.3 | 90.01 | 136.8 | 89.1 | 162.1 | 95.8 | 151.7 | 88.3 | 148.2 | 97.9 |

Table 9 presents the average errors during maneuver obtained by time averaging over each maneuver-on trajectory segment. The peak errors (i.e., the errors at, and the two time steps after the maneuver onset) are excluded from the averaging. The behavior of the algorithms is rather specific (see also Figures 2, 3, and 4). AMM is quite good if the maneuver is represented in the model set and very bad if not presented in the model set (i.e., $M_{31}$) due to the lack of reinitialization. The same pattern (to a lesser extent) is observed also for the hard-decision based BMM and VM. GPB1 copes very well with the "small" maneuvers ($M_{i1},\ i = 1, 2$; $M_{31}$) but totally fails in the cases of the "large" maneuvers ($M_{i2},\ i = 1, 2$). The other soft-decision based, GPB2, IMM and RIMM, provide the best overall performance during maneuver. Note that RIMM is particularly good for the "mismatched" scenario $M_{31}$.

At maneuver termination (see again Tables 7, 8), the peak error of AMM is by far worse than all others. Apparently the "threshold" for the filter to switch to the non-maneuver model is excessively high. The peak errors of IMM, GPB2, BMM, and VMM are reasonably small with the VMM being more noticeably worse. GPB1 and RIMM are very insensitive to maneuver

Table 9: Average Position & Velocity RMSEs during Maneuver

| | Position | | | | | | | Velocity | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | AMM | GPB1 | GPB2 | IMM | RIMM | BMM | VMM | AMM | GPB1 | GPB2 | IMM | RIMM | BMM | VMM |
| $M_{11}$ | 105.0 | 104.1 | 103.6 | 104.8 | 113.7 | 104.6 | 105.7 | 34.4 | 52.5 | 43.3 | 43.3 | 67.0 | 43.2 | 42.5 |
| $M_{12}$ | 126.5 | 227.6 | 113.6 | 115.3 | 160.0 | 157.9 | 121.4 | 112.5 | 193.0 | 87.9 | 90.5 | 116.9 | 91.1 | 79.6 |
| $M_{21}$ | 116.2 | 98.1 | 110.9 | 111.8 | 113.3 | 115.9 | 116.2 | 47.8 | 46.9 | 53.9 | 53.7 | 67.8 | 53.2 | 54.2 |
| $M_{22}$ | 116.7 | 201.2 | 104.3 | 105.7 | 146.5 | 125.7 | 112.7 | 107.6 | 179.0 | 80.2 | 82.9 | 121.9 | 78.8 | 76.2 |
| $M_{31}$ | 211.1 | 132.7 | 151.2 | 153.1 | 115.3 | 177.0 | 167.6 | 136.8 | 86.4 | 130.3 | 128.5 | 86.8 | 140.4 | 140.4 |
| Ave | 135.1 | 152.7 | 116.7 | 118.1 | 129.8 | 136.2 | 124.7 | 87.8 | 111.6 | 79.1 | 79.8 | 92.1 | 81.3 | 78.6 |

termination—they do not develop any peak error at that moment. As a result they can hardly detect the maneuver termination event (see also the model probability plots in Figure 5). While very good at that particular moment, this is devastating for the subsequent non-maneuvering period—they cannot eliminate the non-maneuver error, but just reduce it gradually.

**Motion mode identification.** The capabilities of the MM filters to identify the true motion mode are illustrated by the time-plots of the model probabilities of each algorithm in Figure 5. BMM, GPB2, IMM identify most correctly and rapidly the true mode and are quite similar in performance. VMM is pretty close to them but with a slightly slower transition to the non-maneuver mode. AMM has difficulties (a significant delay) in detecting the terminanion of the small maneuver $M_{21}$ (at $k = 90$) and fails to detect the termination of the large maneuver $M_{22}$ (at $k = 129$) until the end of observation (at $k = 150$). GPB1 and RIMM show poorest capabilities for mode identification.

**Execution time.** The execution times for the algorithms relative to the AMM are given in Table 10. For BMM $B = 9$.

Table 10: Relative Execution Time

| AMM | GPB1 | GPB2 | IMM | RIMM | BMM | VMM |
|---|---|---|---|---|---|---|
| 1.00 | 1.09 | 7.91 | 2.30 | 4.64 | 6.67 | 7.23 |

## 5 Conclusion

Based on the study, there is no clear-cut best algorithm. Overall, BMM, VMM, GPB2 and IMM outperformed the other three algorithms – AMM, GPB1 and RIMM. In general, the hard decision-based BMM and VMM in comparison with the soft decision-based (GPB2, IMM), showed slightly better performance in some cases when the true target motion was represented in the MM set but they were worse when the true target motion was not explicitly represented in the model set. The latter case is most important in practice since the target motion "models" are most often unknown in reality. The IMM algorithm has the best computational complexity among the algorithms that have acceptable tracking errors. It also showed a remarkable robustness to model mismatching, and appears to be the top choice if the computational cost is of concern.

## 6 Appendix:

Kalman filter for model $m_k = m^{(i)}$ with initial condition $(\bar{x}_{k-1|k-1}, \bar{P}_{k-1|k-1})$:

$$
\begin{aligned}
\hat{x}^{(i)}_{k|k-1} &= F^{(i)}_{k-1}\bar{x}_{k-1|k-1} + G^{(i)}_{k-1}\bar{w}^{(i)}_{k-1} \\
P^{(i)}_{k|k-1} &= F^{(i)}_{k-1}\bar{P}_{k-1|k-1}F^{(i)\prime}_{k-1} + G^{(i)}_{k-1}Q^{(i)}_{k-1}G^{(i)\prime}_{k-1} \\
\tilde{z}^{(i)}_k &= z_k - H^{(i)}_k\hat{x}^{(i)}_{k|k-1} - \bar{v}^{(i)}_k, S^{(i)}_k = H^{(i)}_k P^{(i)}_{k|k-1}H^{(i)\prime}_k + R^{(i)}_k \\
K^{(i)}_k &= P^{(i)}_{k|k-1}H^{(i)\prime}_k S^{(i)-1}_k \\
\hat{x}^{(i)}_{k|k} &= \hat{x}^{(i)}_{k|k-1} + K^{(i)}_k\tilde{z}^{(i)}_k, \quad P^{(i)}_{k|k} = P^{(i)}_{k|k-1} - K^{(i)}_k S^{(i)}_k K^{(i)\prime}_k
\end{aligned}
\tag{17}
$$

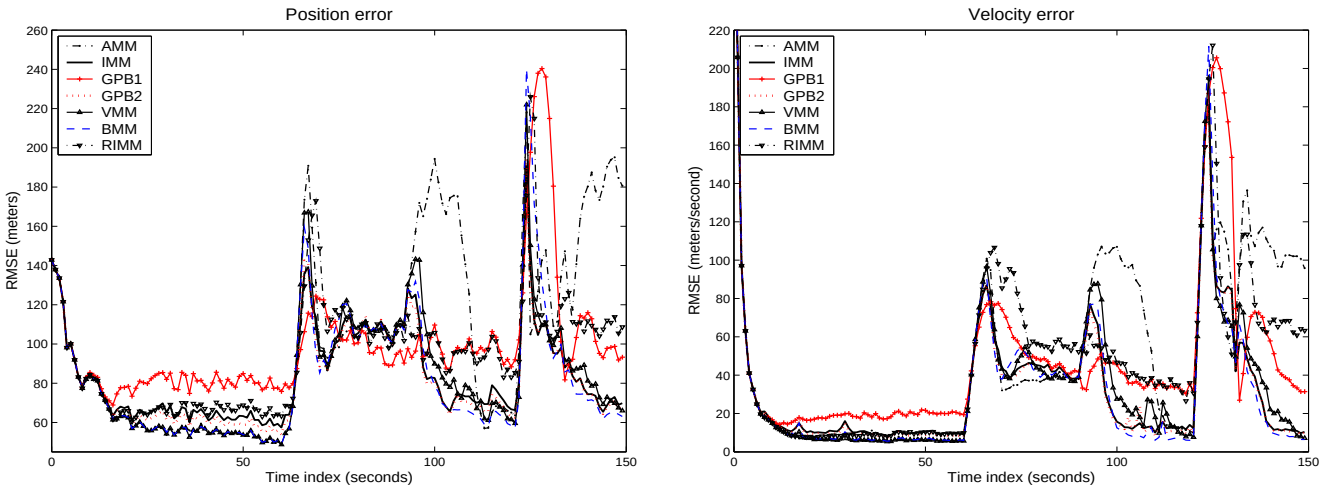


Figure 2: RMSE Scenario 1

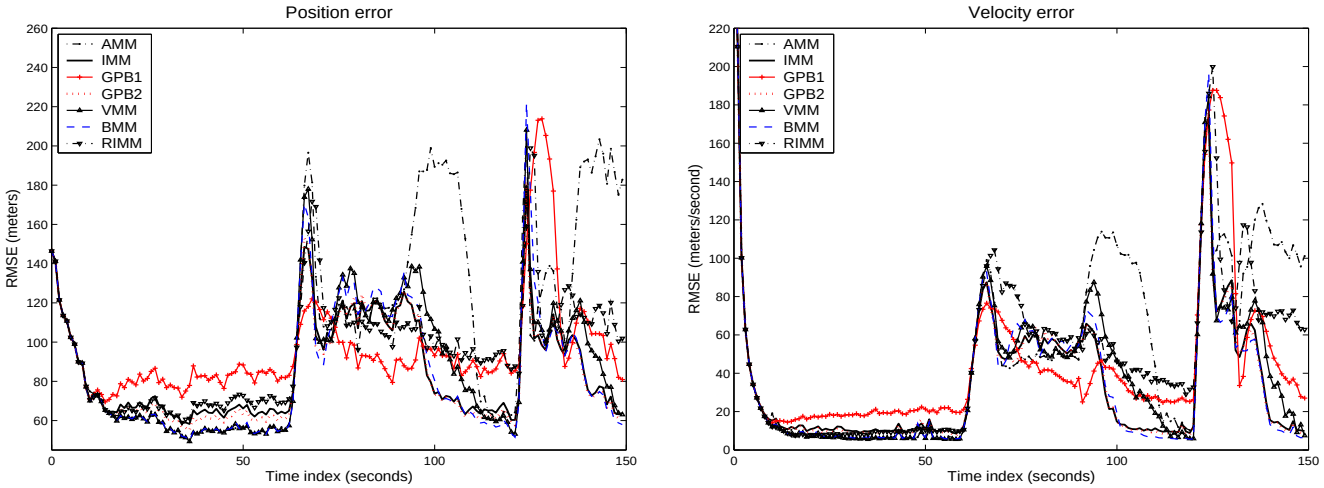


Figure 3: RMSE Scenario 2

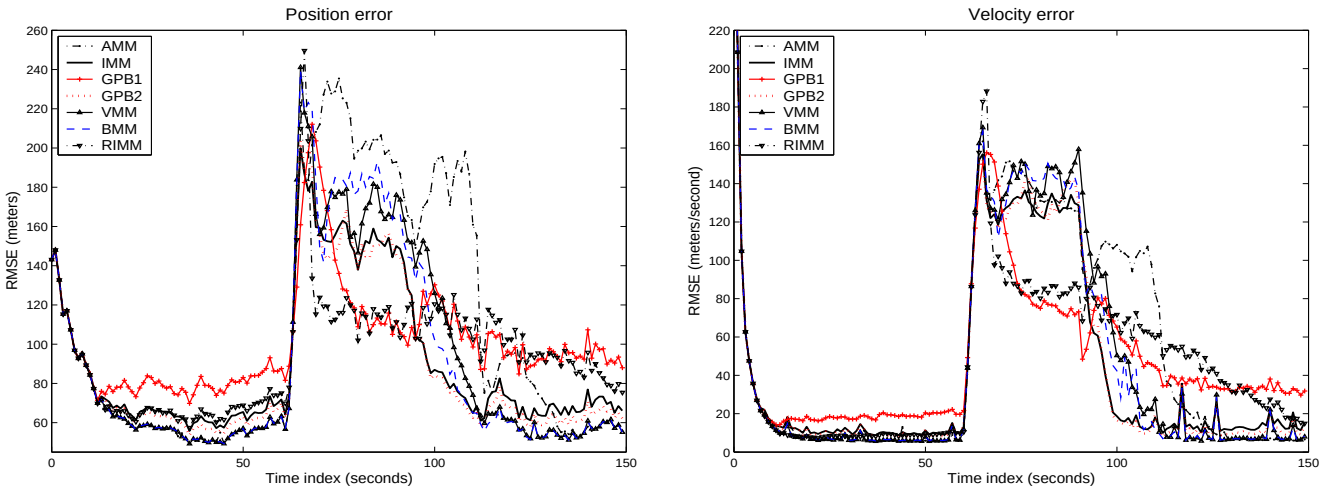


Figure 4: RMSE Scenario 3

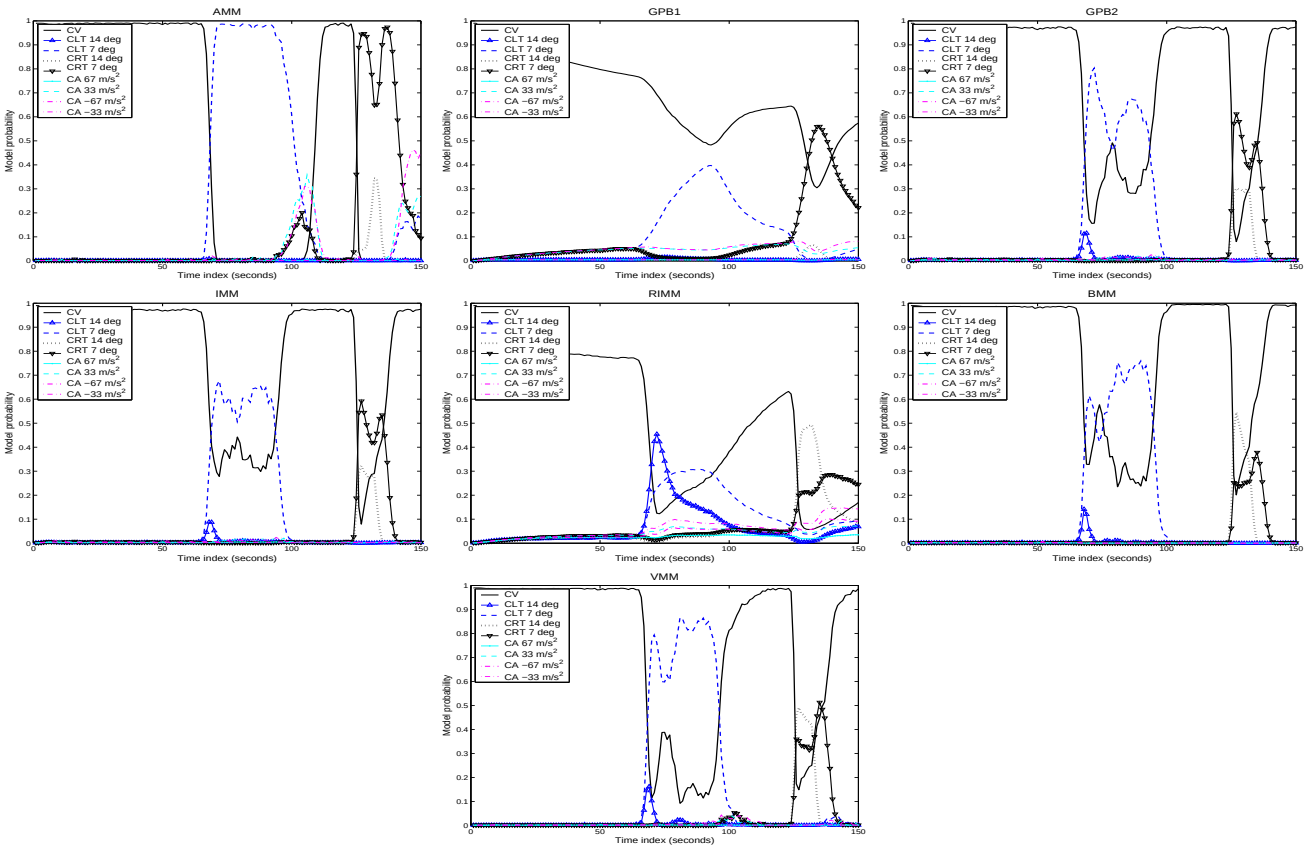


Figure 5: Model Probabilities for Scenario 2

## References


[1] X. R. Li and V. P. Jilkov. A Survey of Maneuvering Target Tracking—Part V: Multiple-Model Methods. *IEEE Trans. Aerospace and Electronic Systems*, 41(2), 2005. To appear.

[2] H. A. P. Blom and Y. Bar-Shalom. The Interacting Multiple Model Algorithm for Systems with Markovian Switching Coefficients. *IEEE Trans. Automatic Control*, AC-33(8):780–783, Aug. 1988.

[3] A. Averbuch, S. Itzikowitz, and T. Kapon. Radar Target Tracking—Viterbi versus IMM. *IEEE Trans. Aerospace and Electronic Systems*, AES-27(3):550–563, May 1991.

[4] F. Gustafsson. *Adaptive Filtering and Change Detection*. Wiley, 2001.

[5] D. T. Magill. Optimal Adaptive Estimation of Sampled Stochastic Processes. *IEEE Trans. Automatic Control*, AC-10:434–439, 1965.

[6] G. A. Ackerson and K. S. Fu. On State Estimation in Switching Environments. *IEEE Trans. Automatic Control*, AC-15(1):10–17, Jan. 1970.

[7] C. B. Chang and M. Athans. State Estimation for Discrete Systems with Switching Parameters. *IEEE Trans. Aerospace and Electronic Systems*, AES-14(5):418–425, May 1978.

[8] H. A. P. Blom. An Efficient Filter for Abruptly Changing Systems. In *Proc. 23rd IEEE Conf. Decision and Control*, Las Vegas, NV, Dec. 1984.

[9] L. A. Johnston and V. Krishnamurthy. An Improvement to the Interacting Multiple Model (IMM) Algorithm. *IEEE Trans. Signal Processing*, 49(12):2893–2908, 2001.

[10] J. K. Tugnait. Adaptive Estimation and Identification for Discrete Systems with Markov Jump Parameters. *IEEE Trans. Automatic Control*, AC-27(5):1054–1065, October 1982.

[11] G. Pulford and B. La Scala. MAP Estimation of Target Manoeuvre Sequence with the Expectation-Maximisation Algorithm. *IEEE Trans. Aerospace and Electronic Systems*, 38(2):367–377, April 2002.

[12] X. R. Li and V. P. Jilkov. Survey of Maneuvering Target Tracking—Part I: Dynamic Models. *IEEE Trans. Aerospace and Electronic Systems*, AES-39(4):1333–1364, Oct. 2003.