Abstract – To improve target tracking algorithms, supervised learning of adaptive interacting multiple model (SLAIMM) is compared to other interacting multiple model (IMM) methods. Based on the classical IMM tracking, a trained adaptive acceleration model is added to the filter bank to track behavior between the fixed model dynamics. The results show that the SLAIMM algorithm 1) improves kinematic track accuracy for a target undergoing acceleration, 2) affords track maintenance through maneuvers, and 3) reduces computational costs by performing off-line learning of system parameters. The SLAIMM method is compared with the classical IMM, the Munir Adaptive IMM, and the Maybeck Moving-Bank multiple-model adaptive estimator (MBMMAE).

Keywords: Interactive Multiple Model

1 Introduction

A single model state estimator is sufficient for tracking targets in constant linear or curvilinear motion. However, for targets with varying or multiple kinematic behaviors, (e.g. maneuvering targets) multiple motion models are utilized to capture diverse behavior. In the classical interacting multiple model (IMM) tracking, each model is assigned a fixed deterministic state (e.g. acceleration).\[1, 2\] The probability of each model being true is found using a likelihood function and the movement between models is performed using a transition probability. The IMM outperforms switching schemes because a smooth (i.e. mixing) transition is achieved between models. Although system accuracies can be improved by increasing the number of models, it also increases computational complexity, degrades performance when the true system dynamics lie between the fixed models, and limits efficiency if all models are simultaneously run whether or not there is a large model contribution to the target state.

Two algorithms which improve upon the classical IMM are the adaptive IMM (AIMM) [3] and Moving-Window IMM (MW-IMM). Munir and Atherton [4] estimate the target’s acceleration and send the result to a bank of IMM filters. An entire bank of filters is centered on the acceleration estimate and the bank moves in acceleration space. The algorithm requires fewer models to cover the acceleration space. 2) Another adaptive multiple model estimator is the so-called moving-bank multiple model adaptive estimator (MBMMAE) presented by Maybeck in [5, 6]. Because the bank of filters is fixed and does not move, perhaps a more appropriate name might be the moving-window multiple model estimator (MWE). [7] Using the IMM framework, models are assumed fixed and the MWE sets a window around a subset of the filter bank which is centered on the estimated acceleration. Only the filters that fall within the window propagate the updated estimates and everything falling outside the window is turned off.

As shown in Figure 1, assume a target is accelerating such that the maneuver requires acceleration analysis. To increase algorithm performance, we seek to reduce complexity and increase accuracy. We propose to reduce the filter number with general motion filters and a single adaptive acceleration model is added to capture target dynamics when its behavior falls between the fixed models [7] and use supervised learning to train the acceleration model parameters. Through learning, the estimated acceleration of a given target can be used to better estimate the parameters of the acceleration space.

The IMM algorithm has been assessed in the literature as a contemporary approach for maneuver target tracking as shown by Semerdjiev et al. [8, 9, 10, 11]. Various approaches to tracking include the multiple-hypothesis tracker (MHT), IMM, and the joint probability data association filter (JPDAF). [12] Different approaches seek to classify the target based on its maneuver [13, 14] such as determining the intent of a target [15]. In addition to the above methods, structure [16] and learning techniques have been applied to complement the IMM: including the neural-net extended Kalman Filter (NEKF) IMM [17] and
the Genetic Algorithm [18]. In this paper, we use supervised reinforcement learning.

Recent applications for target tracking compare JPDAF for electro-optical [19], particle filter (PF) methods for robots [20], unscented KF (UKF) to IMM for adaptive cruise control [21], and fast-adaptive methods for GPS/INS [22]. The most common application of the IMM is for radar target tracking.

Moving target detection with radar can be completed with synthetic aperture radar (SAR) [23] and high-range resolution (HRR) [24, 25] radar. HRRR IMM applications include tracking the signature of a target [26], group tracking [27], and simultaneous target track and identification (STID) [28]. For STID, combinations of joint-belief probability data association (JPBDAF) [29] and IMM/MHT [30] have been applied. Recent efforts include IMM for phased-array radar [31, 32]. One open issue is determining the pose change in the target [33] and subsequent model selection and the detection of acceleration maneuvers and adapting the IMM [34].

Model selection can be done through learning of the target behaviors and we propose a supervised learning adaptive IMM (SLAIMM). As shown, we found that our AIMM outperforms both the classical IMM and other AIMMs with reduced computational complexity. The research focused on further increasing the accuracy without increasing computational complexity. The probabilities associated with the model transitions were learned off-line to improve model mixing/switching over existing IMM methods.

Section 2 introduces the basic theory used in a classical IMM and Section 3 presents the adaptive IMM approaches. Comparisons of the classical, adaptive, and supervised learning IMM methods is shown in Section 4, where we show that the supervised learning adaptive IMM reduces computations in improves track accuracy as compared to the classical and adaptive IMM approaches. Finally, Section 5 presents our conclusions and discusses future research directions.

2 Interactive Multiple Model

Figure 2 shows the architecture for the classical IMM. In IMM tracking, it is assumed that the target behavior obeys one of a finite number of fixed models. As shown in Figure 2, the estimate and covariance that is sent to the filter at time \( k - 1 \) is the weighted Gaussian mixture of the filters estimate at \( k - 1 \). The process is usually referred to as interaction, or mixing of the estimates [1]. The output estimate, usually referred to as the combined estimate, is the weighted Gaussian mixture of each of the IMM models.

We assume that the fixed models exhibit nearly constant velocity with a bias corresponding to acceleration. The stochastic difference equation describing the system behavior for model \( M_j \) : \( j = 1, ..., N \) is given by:

\[
x_j(k + 1) = \Phi x_j(k) + \Gamma [ a_j(k) + w(k) ]
\]

where \( x_j(k) = [p_x(k) \ v_x(k) \ p_y(k) \ v_y(k)]^T \) is the state vector \((x, y)\) position and velocity, \( a_j(k) = [a_x(k) \ a_y(k)]^T \) is the acceleration vector, and \( w(k) \in \mathbb{R}^4 \) is the white noise vector such that \( w(k) \sim N(0, Q) \). The matrices \( \Phi \in \mathbb{R}^{4 \times 4} \) and \( \Gamma \in \mathbb{R}^{2 \times 4} \) are the state transition matrix and the measurement matrix, respectively:

\[
\Phi = \begin{bmatrix}
1 & T & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & T \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
\Gamma = \begin{bmatrix}
\frac{1}{2} T^2 & 0 \\
T & 0 \\
0 & \frac{1}{2} T^2 \\
0 & 0 \\
\end{bmatrix}
\]

Figure 2. Classical IMM architecture.

Each IMM model studied assumes a different acceleration with nine accelerations as shown in Figure 3. A Kalman filter is designed for each of these models. For simulation, noise corrupted measurements of \((x, y)\) position, are modeled as:

\[
z(k) = H x(k) + v(k)
\]

where \( v(k) \in \mathbb{R}^2 \) is a white noise vector \( v(k) \sim N(0, R) \) and \( H \in \mathbb{R}^{2 \times 4} \) is the measurement matrix:

\[
H = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
\end{bmatrix}
\]

Figure 3. IMM model location in acceleration space.
2.1 Interaction/Mixing of the Estimates

The mixing probability that model \( M_j \) is in effect at \( k-1 \) given that \( M_j \) is in effect at \( k \), conditioned on the measurement history up to \( k-1 \), denoted \( z^{k-1} \), is:

\[
\mu_{ij} (k-1 | k-1) = \frac{p_{i} \mu_{ij} (k-1)}{\sum_{i=1}^{N} p_{i} \mu_{ij} (k-1)}
\]

(5)

where \( p_{i} \) is the a priori (known) mode transition probabilities, \( \mu_{ij} \) is the model probability, and the denominator is a normalizing constant.

The mixed estimate, \( \hat{x}_j (k-1 | k-1) \), and covariance, \( p_j (k-1 | k-1) \) for the filter associated with model \( M_j \) are:

\[
\hat{x}_j (k-1 | k-1) = \sum_{i=1}^{N} \mu_{ij} (k-1 | k-1) \hat{x}_i (k-1 | k-1)
\]

(6)

\[
p_j (k-1 | k-1) = \sum_{i=1}^{N} \mu_{ij} (k-1 | k-1) \big( P_i (k-1 | k-1) + A A^T \big)
\]

(7)

where \( A = [\hat{x}_i (k-1 | k-1) - \hat{x}_j (k-1 | k-1)] \).

2.2 Kalman Filtering

The updates for the \( j \text{th}, \{j = 1, .., N\} \), filter model are computed via the Kalman filter equations using the mixed estimate and covariance:

\[
x_j (k | k) = \Phi \hat{x}_j (k-1 | k-1) + \Gamma a_j (k)
\]

(8)

\[
P_j (k | k) = \Phi \hat{P}_j (k-1 | k-1) \Phi^T + \Gamma \Gamma^T + R_j
\]

(9)

\[
S_j (k) = H P_j (k | k-1) H^T + R_j
\]

(10)

\[
v_j (k) = z(k) - H \hat{x}_j (k | k-1)
\]

(11)

\[
K_j (k) = P_j (k | k-1) H^T S_j^{-1}
\]

(12)

\[
\hat{x}_j (k | k) = \hat{x}_j (k | k-1) + K_j (k) v_j (k)
\]

(13)

where \( x_j (k | k) \in \mathbb{R}^{4} \) is the state estimate, \( P_j (k | k) \in \mathbb{R}^{4x4} \) is the covariance of the state estimate error, \( v_j (k) \in \mathbb{R}^{4} \) is the innovation process, \( S_j (k | k) \in \mathbb{R}^{4x4} \) is the covariance of the innovation process, and \( K_j (k | k) \in \mathbb{R}^{4x2} \) is the Kalman filter gain.

2.3 Model Probability

The likelihood function of model \( M_j; \{j = 1, .., N\} \) at time \( k \) under the linear Gaussian assumption is:

\[
\Lambda_j (k) = p[z(k) | Z^{k-1}, M_j] = p[v_j (k)]
\]

(14)

The posterior probability of model \( j \) being correct is obtained recursively and normalized:

\[
\mu_j (k) = \frac{\Lambda_j (k) \sum_{i=1}^{N} p_{ij} \mu_i (k-1)}{\sum_{i=1}^{N} \left( \Lambda_j (k) \sum_{i=1}^{N} p_{ij} \mu_i (k-1) \right)}
\]

(15)

where \( j = 1, .., N \).

The algorithm begins at \( \mu_j (0) \), specified a priori.

2.4 Estimate and Covariance Combination

The estimated probability density function of the system state is the weighted Gaussian mixture of the \( N \) model estimates, i.e.

\[
p[x(k) | Z^k] = \sum_{j=1}^{N} \mu_j (k) N[\hat{x}_j (k | k), P_j (k | k)]
\]

(16)

which has a mean: \( \hat{x}(k | k) = \sum_{j=1}^{N} \mu_j (k) \hat{x}_j (k | k) \)

and covariance given by:

\[
P(k | k) = \sum_{j=1}^{N} \mu_j (k) \left[ P_j (k | k) - B \Gamma B^T \right]
\]

(17)

where, \( B = [\hat{x}_j (k | k) - \hat{x}(k | k)] \).

2.5 Acceleration Filtering

Several methods for estimating acceleration are possible: 1) numerical derivative of the combined velocity estimate which suffers from noise; 2) second numerical derivative of the combined position estimate which also exhibits degradation due to noise; and 3) second position derivative with estimate filtering using both a moving-average window filter and a moving median window filters. The filters smooth the estimate, but results in measurement delays.

A better technique, the one used for simulation, utilizes a separate Kalman filter for estimating the bias (acceleration) that is learned off-line. The acceleration is estimated by the following bias filter:

\[
x_b (k) = [p_a (k) v_a (k) a_a (k) p_j (k) v_j (k) a_j (k)]^T
\]

(18)

\[
x_b (k | k-1) = \Phi_b x_b (k-1 | k-1)
\]

(19)

\[
P_b (k | k-1) = \Phi_b P_b (k-1 | k-1) \Phi_b^T + \Gamma_b \Gamma_b^T
\]

(20)

\[
S_b (k) = H_b P_b (k | k-1) H_b^T + R_b
\]

(21)

\[
v_b (k) = z(k) - H_b \hat{x}_b (k | k-1)
\]

(22)

\[
K_b (k) = P_b (k | k-1) H_b^T S_b^{-1}
\]

(23)
\[ P_b(k|k) = P_b(k|k-1) - K_b(k) H_b P_b(k|k-1) \quad (24) \]
\[ \hat{x}_b(k|k) = \hat{x}_b(k|k-1) + K_b(k) v_b(k) \quad (25) \]

where

\[
\Phi = \begin{bmatrix}
1 & T & \frac{1}{2} T^2 & 0 & 0 & 0 \\
0 & 1 & T & 0 & 0 & 0 \\
0 & 0 & 1 & T & \frac{1}{2} T^2 & 0 \\
0 & 0 & 0 & 1 & T & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix} \quad \Gamma = \begin{bmatrix}
\frac{1}{2} T^2 & 0 \\
T & 0 \\
1 & 0 \\
0 & 1 & 0 \\
0 & \frac{1}{2} T^2 & 0 \\
0 & T & 0 \\
0 & 0 & 1 & 0 \end{bmatrix}
\]

and

\[
H = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0
\end{bmatrix}
\]

3 Adaptive IMM Techniques

Adapting the IMM models to various target maneuvers supports a tailored approach to target tracking. The approaches discussed include adaptive, moving-window, and supervised learning IMM methods.

3.1 Adaptive IMM

The Adaptive IMM method [3] moves the entire filter bank to center on the acceleration estimate, Figure 4.

3.2 Moving-Window Adaptive IMM

The moving-window adaptive IMM [5,6] uses a subset of the nine filters - those filters contained in a window centered on the acceleration estimate as shown in Figure 5. All other filters outside the window are turned off. When the window moves, new filters are brought on line and initialized by the previous combined state estimate and covariance. Without accuracy improvement beyond the classical IMM, the MWE provides a significant reduction in computation.

3.3 Supervised Learning Adaptive IMM

There are various forms of learning that can be employed: search, reinforcement, and supervised. The standard IMM models suffer from heuristic initializations of probabilities and learning of target maneuvers can improve the adaptive IMM methods for tracking of maneuvering targets. Learning assumes some level of intelligence (i.e. supervised) and is shown to be better than the standard heuristic or random search approaches for the initializations of probabilities as well as reduces the required longitudinal historical data needed for reinforce learning techniques.

Specifically, Equation 5 uses a symmetric \( P_T \) matrix as the \( a \) priori (known) mode transition probabilities:

\[
P_T = \begin{bmatrix}
a & b & \cdots & n \\
b & c & \cdots & n-1 \\
\cdots & \cdots & \cdots & \cdots \\
n & n-1 & \cdots & m
\end{bmatrix} \quad (26)
\]

where \( m \) is a transition probability for the \( m \)th model. Learning the optimum transition probabilities can be performed off-line and updated on-line to increase the accuracy of the AIMM model. If the learning takes place off-line, then the computational requirements are not increased during execution, but performance is enhanced.

The supervised-learning adaptive IMM is illustrated in Figure 6. It is based on having a number of fixed IMM models and a single learned adaptive model to capture behavior that falls in between the fixed model dynamics. The learned model can be determined \( a \) priori based on estimated target types, learning over the behaviors most likely to be exhibited by a few target types and capabilities, or learned from the history of a unknown but observable target.

To reduce computations associated with learning the target maneuver (i.e. acceleration maneuvers), the number of fixed motion models was reduced so that the acceleration space is covered at coarser levels. The analysis that follows is based on only five fixed models. The models are trained based on the maneuvers of the target (i.e. supervised) and can be adaptive on-line based on the observations of the target maneuvers.
The critical notion is whether the on-line learning suffers from computational burdens. Reinforcement learning tries to achieve a specified goal, but suffers from combinatorial explosion; a random search learns the optimum transition probabilities but is inefficient; and supervised learning, selected for the analysis, directs the learning process and exhibits less computational burdens and may be adapted for on-line analysis.

4 Simulation Results

The performance of the SLAIMM algorithm is evaluated over the a) classic IMM algorithm, b) Mun-Ath Adaptive IMM, b) supervised-learning adaptive IMM algorithm, and c) moving-window IMM algorithm. In these simulations, we used the target track shown in Figure 7, where the target performs three maneuvers at time \( t = \{10, 20, 30\} \) seconds.

The nine model IMM is used for algorithm comparison. To reduce computations, the adaptive IMM is designed using only six models (five fixed and one adaptive). Six models are chosen to provide a fair comparison with the classical IMM. If an adaptive model is added to the nine model IMM, we would have had ten models in our AIMM. The comparison would not be fair as better performance might be obtained at the cost of additional computations. We also implemented the AIMM presented by Munir-Atherton [2] and the Moving-Window AIMM by Maybeck [5,6] to determine how they compare to our AIMM.

For each of the algorithms, we computed the average RMS errors in the \( x \) and \( y \) axis for 50 Monte Carlo runs. The results are shown in Table 1. Further we compared the number of Matlab FLOPS required by each algorithm to determine which was most efficient.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>FLOPs</th>
<th>RMSx</th>
<th>RMSy</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMM</td>
<td>33802500</td>
<td>5.4916</td>
<td>3.9259</td>
</tr>
<tr>
<td>Mun-Ath</td>
<td>38580000</td>
<td>5.3912</td>
<td>3.7839</td>
</tr>
<tr>
<td>Mov-Win</td>
<td>26569872</td>
<td>5.1089</td>
<td>4.3241</td>
</tr>
<tr>
<td>SLAIMM</td>
<td>25040000</td>
<td>3.5511</td>
<td>3.4323</td>
</tr>
</tbody>
</table>

From Table 1, we see that the Adaptive IMM method has comparable average RMS errors as the classic IMM. Also, note that the moving-widow model has better performance in \( x \), but not \( y \) indicating that a fixed model could be beneficial. However, the average RMS errors of the SLAIMM are much smaller than those obtained for the other methods.

SLAIMM requires the fewest FLOPS of all the algorithms considered. The results show that the FLOPS count is largest in the Mun-Ath Adaptive IMM method, which acknowledges that additional computations over the classical IMM when estimating the acceleration. Also note that the FLOPS count of the moving-window method is less than the classic IMM method due to the fact that only 4 models are active at any given time. Using one model in the SLAIMM is efficient and improved through a priori learning off-line the target behaviors and adaptive learning of on-line current model parameters.

4.1 RMS Errors

Figures 8-11 show plots of the RMS errors over 50 Monte Carlo runs for each method. By examining Figure 8 you can see that the RMS error for the classic IMM is increased when the target is doing a maneuver.

In Figure 9 we see that the Munir-Atherton model attempts to move its center acceleration model to where the maneuver occurs. Hence, the RMS error is large during
the move and decreases as the acceleration is identified. Note that the second maneuver is larger than the first and the RMS is larger.

![Figure 9. RMS Errors of MA Adaptive IMM.](image)

Figure 9 shows us that the RMS errors of the moving-window method exhibit a large transient when the window moves as a result of a maneuver. The reason for this maybe due to the fact that the target does a swift transition from one set of acceleration to another. We believe that this model is unable to handle such a fast transition. (We made many attempts to correct this problem without finding a satisfactory solution.)

![Figure 10. RMS Errors of Moving-Window IMM.](image)

Figure 10 shows that SLAIMM provided the best overall performance with the least number of computations. The reason for this is simple. The adaptive model is able to capture the target’s dynamics when the accelerations fall between the fixed model dynamics and by learning the optimum transition probabilities, the algorithm accounts for model switching.

![Figure 11. RMS Errors for SL-Adaptive IMM.](image)

4.2 Transition Probabilities

Figures 12-15 show the plots of the probability of each model in each algorithm. When looking at the plot for the Munir method note that the “center” model is always moving to match the target acceleration. This is demonstrated by the probability of the center model rising quickly during a transition in acceleration. The dark line in the subsequent Figures indicates the acceleration model.

![Figure 12. Classic-IMM Model probabilities.](image)

Notice in Figure 13 that the moving-window method is unable to handle the transition from one quadrant to another.

![Figure 13. Adaptive IMM Model probabilities.](image)
Finally in Figure 15 it is observed that the SLAIMM performs well when the target changes maneuver. More specifically we see that our learned adaptive method captures the maneuver well when the maneuver is not matched exactly by one of the fixed models. It is this feature that makes our method better than the traditional IMM and others.

Figure 15. Model probabilities for our adaptive IMM.

4.3 Simulation and Learning

The heuristic method just chooses the best guess of the programmer. The examples from the text are on-diagonal elements $= 0.8$ and off-diagonal elements $= 0.025$, in Equation 5. In Figure 16, the search algorithm finds the optimal approach, but after much computational time, which makes it unsuitable for online implementation. The supervised learning algorithm learns the optimal transition probabilities in a reasonable amount of time and shows possibility for on-line implementation. The learned parameters are then used in the acceleration model embedded in the adaptive IMM algorithms. Table 2 shows that the learning results for the different methods compared to approaches.

![Figure 16. Model Errors(x) for SLAIMM.](image)

![Figure 17. Model Errors(y) for SLAIMM.](image)

Table 2. FLOPs and the Ave. RMS errors.

<table>
<thead>
<tr>
<th></th>
<th>Heuristic</th>
<th>Iterative</th>
<th>Learn</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-diagonal</td>
<td>0.800</td>
<td>0.500</td>
<td>0.4</td>
</tr>
<tr>
<td>Off-diagonal</td>
<td>0.025</td>
<td>0.005</td>
<td>0.6</td>
</tr>
<tr>
<td>FLOPs</td>
<td>620700</td>
<td>248280</td>
<td>312560</td>
</tr>
<tr>
<td>RMSx</td>
<td>967</td>
<td>195</td>
<td>121</td>
</tr>
<tr>
<td>RMSy</td>
<td>650</td>
<td>138</td>
<td>118</td>
</tr>
</tbody>
</table>

Note that learning can be used not only for determining the features of the target motion, but also features for target identification [28].

5 Conclusion

The paper provides an empirical analysis of adaptive IMM schemes and how learning can reduce the computational time with the supervised-learning adaptive Interactive Multiple Model (SLAIMM) algorithm. The results provide evidence that a learning adaptive IMM outperforms the classical IMM and other adaptive IMM algorithms by increasing effective accuracy while increasing efficiency by reducing computations. Learning provides an assurance of into the stability of the IMM to the simulated target tracking system. Future work will focus on the processing real data obtained from high fidelity sensor models to fully assess the applicability for using the SLAIMM algorithm for tracking maneuvering targets.
6 References


