IMPLEMENTATIONS OF A MULTISENSOR MULTITARGET TRACKING ALGORITHM

Gîrniță Marian-Silviu

Military Technical Academy Blvd. George Coşbuc, 81-83, Bucureşti, ROMÂNIA

Abstract: Parallel and sequential implementations of the Multi-sensor Joint Probabilistic Data Association (MSJPDA) tracking algorithm are analyzed and compared. The sequential implementation is shown to be exponentially less computationally complex as the number of sensors increases. Simulation results suggest that the sequential method also yields better tracking performance on the average. This is primarily due to the fact that better filtered estimates are available after processing each sensor's data. Thus, while sequential and parallel implementations are equivalent in multisensor filtering when no data association routine is needed, the sequential implementation gives superior tracking performance when data association is required.

Keywords: *multitarget, tracking , multisensor*

1. INTRODUCTION

The computational requirements for Kalman filtering in multisensor systems have been studied in (Speyer, 1979; Willner, 1976). For linear systems, parallel and sequential Kalman filtering of measurements from multiple sensors are equivalent and optimum (Willner, 1976). When multiple sensors are used for tracking the states of multiple objects in a cluttered environment, a data association process is often necessary to assign measurements to the objects they represent. When data association algorithms are used, the equivalence of parallel and sequential implementations no longer hold and it is not obvious which method would yield better tracking performance. We analyze both parallel and sequential implementation of a Multisensor Joint Probabilistic Data Association (MSJPDA) algorithm (Bar-Shalom, 1988; O'Neill, 1993), and we present simulation results showing the performance differences of the two implementations.

2. MULTISENSOR MULTITARGET TRACKING

The multisensor multitarget tracking problem is to track T targets in clutter with N_s sensors. Measurements (also called reports or returns) from the sensors are received by a central processor at discrete time intervals. Each measurement can originate from at most one target. Some sensors may not provide measurements at every interval. Some of the measurements arise from targets, and some arise from clutter; some targets may not yield any measurements at all in a particular time interval or for a particular sensor. Measurement errors due to measurements from one sensor are assumed to be independent of those from another sensor. The target dynamics and the measurements are assumed to obey the following linear equations:

$$\mathbf{x}^t(k+1) = \mathbf{F}^t(k)\mathbf{x}^t(k) + \mathbf{G}^t(k)\mathbf{w}^t(k) \quad (1)$$

$$\mathbf{z}^t_{i,l}(k) = \mathbf{H}_i(k)\mathbf{x}^t(k) + \mathbf{v}^t_i(k) \quad (2)$$

where $x^t(k)$ $(1 < t < T)$ denote the state vectors of each target t at the kth time interval, and $\mathbf{z}^t_{i,l}(k)$ $(1 < i < N_s)$ $(1 \leq l \leq M_{i,k})$ denote the target originated measurements at the kth interval. The matrices $\mathbf{F}^t(k)$, $\mathbf{G}^t(k)$ and $\mathbf{H}_i(k)$ are assumed to be known. Each $\mathbf{w}^t(k)$ and $\mathbf{v}^t_i(k)$ is a zero mean Gaussian noise vector uncorrelated with all other noise vectors, and the covariance matrices of the noise vectors are known. The number of reports from each sensor i at the kth time interval is denoted by $M_{i,k}$. Assuming a pre-correlation gating process is used to eliminate some of the returns (Bar-Shalom, 1988), let $m_{i,k}$ denote the number of validated returns from sensor i at time k. For a given target t and sensor i, it is not known which measurement l ($1 \leq l \leq M_{i,k}$) originates from the target. That is the problem of data association whereby it is necessary to determine which measurements originate from which targets. Let Z^k denote the sequence of the first k observations.

3. MULTISENSOR JPDA

3.1 Single sensor JPDA

We first recapitulate the JPDA algorithm for a single sensor since this gives us a basis to easily describe both the parallel and the sequential MSJPDA implementations. For single-sensor tracking, $N_s = 1$ and the goal is to associate the T targets with the $m_{1,k}$ measurements based on the current estimates of the target states and to update those estimates. The actual association being unknown, the conditional estimate is determined by taking a weighted average over all possible associations. For $1 \leq t \leq T$ and $0 \leq l \leq m_{1,k}$ let $\theta_l^k(k)$ denote the event that return l is the true measurement from target t. The conditional probabilities $\beta_l^t(k)$ of the events $\theta_l^k(k)$ given Z^k are then the association probabilities:

$$\beta_l^t(k) = P\{\theta_l^t(k) \mid Z^k\} \tag{3}$$

Let $\hat{\mathbf{x}}_l^t(k|k)$ denote the estimate of $\hat{\mathbf{x}}^t(k|k)$ given by the Kalman filter on the basis of the previous estimate and the association of the tth target with the lth return:

$$\hat{\mathbf{x}}_l^t(k|k) = \hat{\mathbf{x}}^t(k|k-1) + \mathbf{K}^t(k)[\mathbf{z}_l - \mathbf{H}(k)\hat{\mathbf{x}}^t(k|k-1)] \tag{4}$$

where $\hat{\mathbf{x}}^t(k|k-1) = \mathbf{F}^t(k)\hat{\mathbf{x}}^t(k-1|k-1)$ is the prediction of $\mathbf{x}^t(k)$ and $\mathbf{K}^t(k)$ is the filter gain for the tth target estimate. The conditional estimate $\hat{\mathbf{x}}^t(k \mid k)$ for $\mathbf{x}^t(k)$ given Z^k is

$$\hat{x}^t(k \mid k) = \sum_{l=0}^{m_{1,k}} \beta_l^t(k)\hat{x}_l^t(k|k) \tag{5}$$

3.2 Parallel MSJPDA

In this section, we review a parallel implementation of the MSJPDA algorithm (O'Neil, 1993; Pao, 1994), where all the measurements from all N_s sensors are taken into account in one pass through the multisensor data association and filtering routines (see Figure 1). For multisensor tracking, the T targets now have to be associated with the $m_{i,k}$ measurements for each of the N_s sensors.

For $1 \leq t \leq T$ şi $\mathcal{L} = (l_1, l_2, \dots, l_{Ns})$ where $0 \leq l_1 \leq m_{1,k}$, …, $0 \leq l_{Ns} \leq m_{Ns,k}$ let $\theta_{\mathcal{L}}^t(k)$ denote the event that l_i is the true measurement from sensor i for the kth observation. Let $\beta_{\mathcal{L}}^t(k)$ denote the conditional probability of $\theta_{\mathcal{L}}^t(k)$ given Z^k. The desired multisensor event probability, $\beta_{\mathcal{L}}^t(k)$, is given by

$$\beta_{\mathcal{L}}^t(k) = \prod_{i=1}^{N_s} \beta_{l_i,i}^t(k) \tag{6}$$

where the $\beta_{l_i,i}^t(k)$ are just the single-sensor event probabilities described above. The conditional estimates $\hat{\mathbf{x}}^t(k|k)$ for the MSJPDA algorithm is given by

$$\hat{\mathbf{x}}^t(k|k) = \sum_{\mathcal{L}} \beta_{\mathcal{L}}^t(k)\hat{\mathbf{x}}_{\mathcal{L}}^t(k|k) \tag{7}$$

where the sum is over all possible sets of associations $\mathcal{L}$ with target t. The estimate $\hat{\mathbf{x}}_{\mathcal{L}}^t(k|k)$ of $\mathbf{x}^t(k)$ is based on the prediction $\hat{\mathbf{x}}^t(k|k-1)$ and the association of the tth target with the set of $\mathcal{L}$ returns from the N_s sensors:

$$\begin{aligned}\hat{\mathbf{x}}_{\mathcal{L}}^t(k|k) = \hat{\mathbf{x}}^t(k|k-1) + \sum_{i=1}^{N_s} \mathbf{K}_i^t(k)[z_{l_i}^i(k)] - \\ - \mathbf{H}_i(k)\hat{\mathbf{x}}^t(k|k-1)\end{aligned} \tag{8}$$

where $\mathbf{K}_i^t(k)$ is the filter gain for measurements from the ith sensor for a multisensor system. The covariance update corresponding to $\hat{\mathbf{x}}_{\mathcal{L}}^t(k|k)$ is computed by

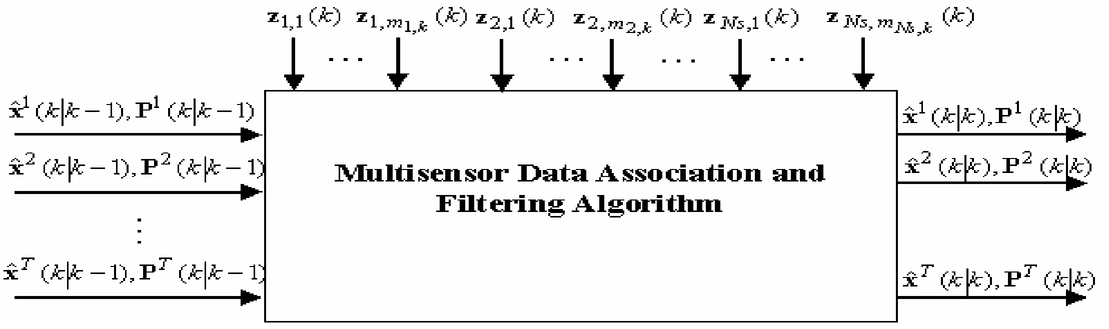
$$\begin{aligned}\mathbf{P}^t(k|k) = \sum_{\mathcal{L}} \beta_{\mathcal{L}}^t(k)[\mathbf{P}_{\mathcal{L}}^t(k|k) + \hat{\mathbf{x}}_{\mathcal{L}}^t(k|k)\hat{\mathbf{x}}_{\mathcal{L}}^t(k|k)^T]^T - \\ - \hat{\mathbf{x}}^t(k|k)\hat{\mathbf{x}}^t(k|k)\end{aligned} \tag{9}$$

where $\mathbf{P}_{\mathcal{L}}^t(k|k)$ are covariances corresponding to $\hat{\mathbf{x}}_{\mathcal{L}}^t(k|k)$.

3.3 Sequential MSJPDA

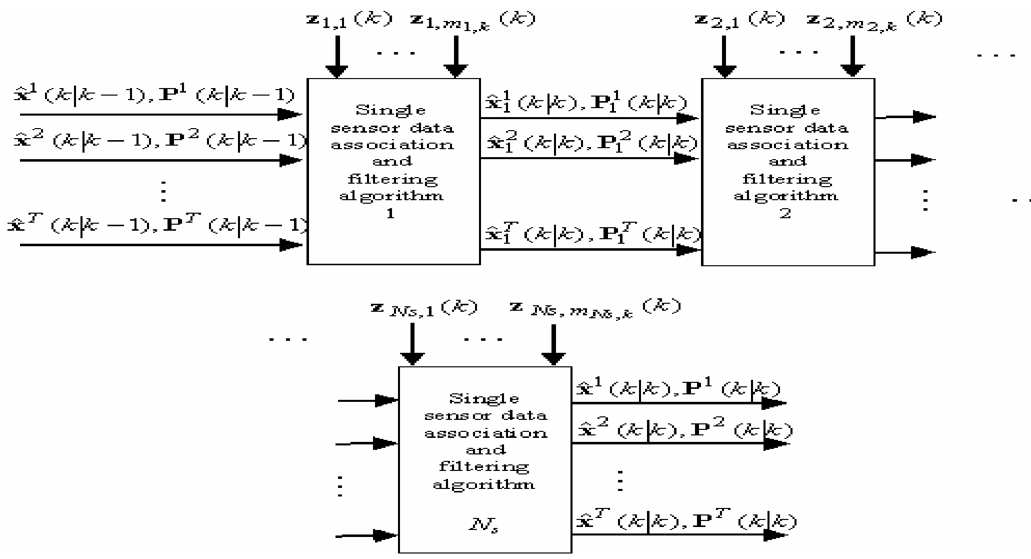
Another way of implementing the MSJPDA algorithm is to process the measurements from each sensor one sensor at the time, as shown in Figure 2. The measurements of a first sensor are used to compute a first intermediate state estimate $\hat{\mathbf{x}}_1^t(k|k)$ and the corresponding covariance $\mathbf{P}_1^t(k|k)$ for each of the targets. The performed computation is equivalent to the one described for the single-sensor case (section 3.1). The measurements of the next sensor are then used to further improve this intermediate state estimate, again using the single-sensor JPDA filter. If we let $\hat{\mathbf{x}}_i^t(k|k)$ and $\mathbf{P}_i^t(k|k)$ denote the state estimate and covariance respectively after processing the data of the ith sensor, the update equations are

$$\hat{\mathbf{x}}_i^t(k|k) = \hat{\mathbf{x}}_{i-1}^t(k|k) + \mathbf{K}_i^t(k) \sum_{l_i=0}^{m_{i,k}} \beta_{l_i,i}^t(k)[z_{l_i}^i(k)] - $$
$$- \mathbf{H}_i(k)\hat{\mathbf{x}}_{i-1}^t(k|k) \qquad (10)$$



$\mathbf{z}_{1,1}(k)$ $\mathbf{z}_{1,m_{1,k}}(k)$ $\mathbf{z}_{2,1}(k)$ $\mathbf{z}_{2,m_{2,k}}(k)$ $\mathbf{z}_{Ns,1}(k)$ $\mathbf{z}_{Ns,m_{Ns,k}}(k)$
$\hat{\mathbf{x}}^1(k|k-1), \mathbf{P}^1(k|k-1)$
$\hat{\mathbf{x}}^2(k|k-1), \mathbf{P}^2(k|k-1)$
$\hat{\mathbf{x}}^T(k|k-1), \mathbf{P}^T(k|k-1)$
Multisensor Data Association and Filtering Algorithm
$\hat{\mathbf{x}}^1(k|k), \mathbf{P}^1(k|k)$
$\hat{\mathbf{x}}^2(k|k), \mathbf{P}^2(k|k)$
$\hat{\mathbf{x}}^T(k|k), \mathbf{P}^T(k|k)$

Fig. 1. Parallel implementation of multisensor tracking algorithm



$\mathbf{z}_{1,1}(k)$ $\mathbf{z}_{1,m_{1,k}}(k)$
$\mathbf{z}_{2,1}(k)$ $\mathbf{z}_{2,m_{2,k}}(k)$
$\hat{\mathbf{x}}^1(k|k-1), \mathbf{P}^1(k|k-1)$
$\hat{\mathbf{x}}^2(k|k-1), \mathbf{P}^2(k|k-1)$
$\hat{\mathbf{x}}^T(k|k-1), \mathbf{P}^T(k|k-1)$
Single sensor data association and filtering algorithm 1
$\hat{\mathbf{x}}_1^1(k|k), \mathbf{P}_1^1(k|k)$
$\hat{\mathbf{x}}_1^2(k|k), \mathbf{P}_1^2(k|k)$
$\hat{\mathbf{x}}_1^T(k|k), \mathbf{P}_1^T(k|k)$
Single sensor data association and filtering algorithm 2
$\mathbf{z}_{Ns,1}(k)$ $\mathbf{z}_{Ns,m_{Ns,k}}(k)$
Single sensor data association and filtering algorithm N_s
$\hat{\mathbf{x}}^1(k|k), \mathbf{P}^1(k|k)$
$\hat{\mathbf{x}}^2(k|k), \mathbf{P}^2(k|k)$
$\hat{\mathbf{x}}^T(k|k), \mathbf{P}^T(k|k)$

Fig. 2. Sequential implementation of multisensor tracking algorithm.

where $\hat{\mathbf{x}}_0^t(k|k) = \hat{\mathbf{x}}^t(k|k-1)$ and $\hat{\mathbf{x}}_{Ns}^t(k|k) = \hat{\mathbf{x}}^t(k|k)$. With $\mathbf{P}_0^t(k|k) = \mathbf{P}^t(k|k-1)$ and $\mathbf{P}_{Ns}^t(k|k) = \mathbf{P}^t(k|k)$, the update of the covariance matrices is

$$\mathbf{P}_i^t(k|k) = \beta_{0,i}^t(k)\mathbf{P}_{i-1}^t(k|k) + [1-\beta_{0,i}^t(k)][\mathbf{I} - \mathbf{K}_i^t(k)\mathbf{H}_i(k)]\mathbf{P}_{i-1}^t(k|k) +$$
$$+ \mathbf{K}_i^t(k)[\sum_{l_i=0}^{m_{i,k}} \beta_{l_i,i}^t(k)\mathbf{z}_{l_i}^i(k)\mathbf{z}_{l_i}^i(k)^T -$$
$$- \sum_{l_i=0}^{m_{i,k}} \beta_{l_i,i}^t(k)\mathbf{z}_{l_i}^i(k) \sum_{l_i=0}^{m_{i,k}} \beta_{l_i,i}^t(k)\mathbf{z}_{l_i}^i(k)^T]\mathbf{K}_i^t(k)^T \qquad (11)$$

It is important to notice that the intermediate state estimates $\hat{\mathbf{x}}_{i-1}^t(k|k)$ and the covariances $\mathbf{P}_{i-1}^t(k|k)$ are used to compute the association probabilities $\beta_{l_i,i}^t$ for the *i*th sensor. These association probabilities still have the same meaning as for the parallel MSJPDA filter but they have different values.

4. COMPUTATIONAL COMPLEXITY

The computational complexity becomes important as the number of sensors and the clutter density grow. An addition/subtraction and multiplication/division is counted as one operation. The variable s_T denotes the number of elements in the target state vector and s_i denotes the size of the measurement vector of the ith sensor. In simplifying the expressions for approximating the operation counts, some terms were dropped for the following reasons. First, since we are trying to track targets in a cluttered environment, the number of measurements $m_{i,k}$ is assumed to be larger than the number of targets *T*. Second, $m_{i,k}$ is also assumed to be much greater than both s_T and s_i. Finally, it is assumed that sensors are not capable of giving data for all the quantities of the target states, that is, $s_i < s_T$. The computational complexity of both algorithms is equivalent except in the covariance update routines. The quantities $m_{i,k}$ grow linearly with λ (clutter density). Therefore the complexity for the sequential covariance update grows linearly with the number of sensors and the clutter density. However, in the parallel implementation, the computational complexity of the covariance update routine grows exponentially with the number of sensors and polynomially (power N_s) with the clutter density. Thus, as the clutter density and the number of sensors increase, the sequential implementation becomes significantly more computationally efficient compared to the parallel implementation.

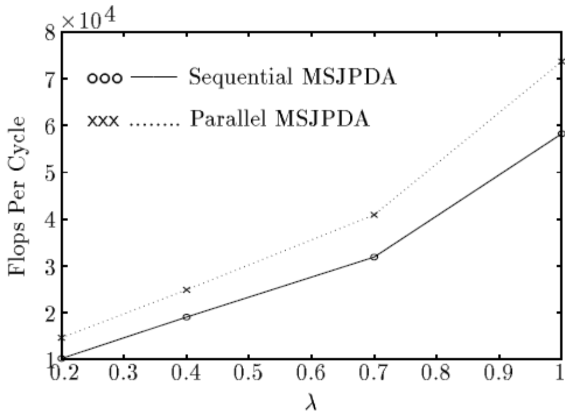


Fig.3. Variation of the computational complexity with clutter density.

For the Monte Carlo simulations to be presented in section 5, Figure 3 shows how the average number of flops (floating-point operations) per cycle varies with clutter density. As the analytical comparison above suggests, the computational complexity increases with λ and is lower for the sequential MSJPDA algorithm.

5. SIMULATION RESULTS

I have run simulations comparing the parallel and sequential implementation of the MSJPDA algorithm described in sections 3.2 and 3.3. Simulations have been run for the tracking of two targets. The configuration for these simulations are the same as in (Pao, 1993) with the exception that the noise covariances chosen here are $r = q = 0.0144$. Two measures of tracking performance are used: average RMS position error and average track lifetime (Salmond, 1990). The clutter density λ was varied from 0.2 to 1.0. The expected number of false measurements per gate, using steady-state Kalman filter covariances, varies then from 0.38 to 1.92.

To evaluate tracking performance, 100 Monte Carlo runs were performed for various values of A (indicated by the x's and o's in the figures). The algorithms were run until both tracks were lost. The track lifetimes were averaged over both targets and across all 100 runs, and the RMS position errors were averaged over all filter estimates (prior to track loss) of both targets and across all 100 runs.

Figure 4 shows the average track lifetimes for parallel and sequential implementations of the MSJPDA algorithm as the clutter density is varied. As expected, the average track lifetimes decrease as the clutter density increases. The figure also shows that, on the average, the sequential MSJPDA yields longer track lifetimes than the parallel MSJPDA implementation.

Figure 5 presents similar results for the average RMS position errors. It is not surprising that the average RMS position errors increase as the clutter density increases. We also see that the average RMS position error is lower for the sequential implementation of the MSJPDA.In figure 5 results for additional runs $\lambda = 10^{-10}$ are also shown, and we see that the parallel and sequential results at $\lambda = 10^{-10}$ are very close to each other. They lie slightly above the theoretical value for the steady state RMS of the equivalent Kalman filters under perfect data association (dash-dotted line); this is primarily due to the uncertain measurement association when there are multiple interfering targets rather than due to clutter.

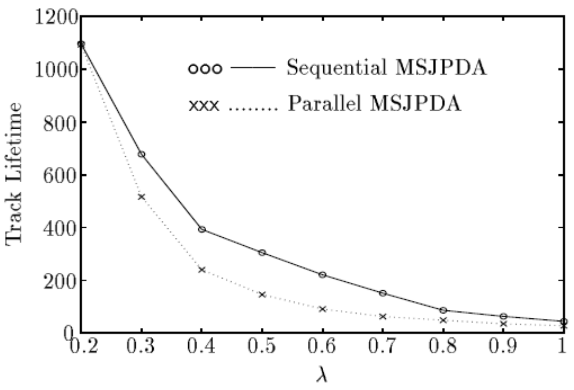


Fig.4. Variation of the track lifetime with clutter density.

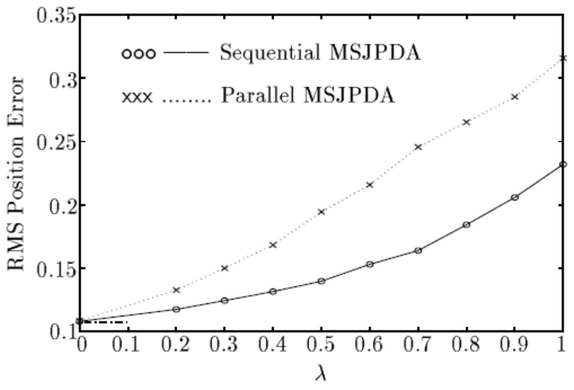


Fig.5. Variation of average position RMS with clutter density.

A heuristic explanation of why the sequential implementation yields better performance is as follows. The parallel implementation uses the predicted measurements and covariances based on the state estimates and co-variances of the previous interval for data association and filtering of measurements from all sensors in the current interval, whereas the sequential implementation only uses this information for data association and filtering of measurements for the first sensor. After processing data from this first sensor, better estimates are available which are then used for data association and filtering of measurements from the next sensor. Thus, successively better estimates are used for data association and filtering for each subsequent sensor.

Because of the statistical nature of data association, the sequential implementation is better only in the average. A close comparison of the simulations run reveal that there are indeed a few runs where the parallel implementation yield better performance than the sequential implementation.

6. CONCLUSIONS

We have compared the performance and computational complexity of parallel and sequential implementations of the MSJPDA tracking algorithm. From simulation results, the performance (based on RMS position error and track lifetime metrics) of the sequential implementation is better on the average than the parallel implementation. Analytical and simulation results further show that the sequential implementation is increasingly more computationally efficient as the clutter density and number of sensors increase.

For multiple sensors with different characteristics, it is necessary to investigate the order in which sensor information should be processed in the sequential implementation. (We bypassed this issue in the current study by using identical sensors in our simulations). Our conjecture is that using the "best" sensors first should lead to better performance as well as lower complexity, since this would yield better intermediate state estimates and smaller gate sizes for the processing of the measurements from the remaining sensors.

REFERENCES

Bar-Shalom, Y. and T. E. Fortmann (1998). *Tracking and Data Association*, Academic Press, San Diego.

O'Neil, S. D. and L. Y. Pao (1998). *Multisensor Fusion Algorithms for Tracking*, *Proc. 1998 American Control Conference,* San Francisco, CA, pp. 859-863.

Pao, L. Y. (1999). *Multisensor Multitarget Mixture Reduction Algorithms for Tracking*, *Proc. AIAA Guidance, Navigation and Control Conf.,* Monterey, CA, pp.28-37.

Pao, L. Y. (1997) *Centralized Multisensor Fusion Algorithms for Tracking Applications*, *IFAC J. Control Engineering Practice,* 2(5), 1997.

Speyer, J.L. (1999). *Computation and Transmission Requirements for a Decentralized Linear-Quadratic-Gaussian Control Problem*, *IEEE Trans. Automatic Control,* 24(2): 266-269.

Willner, D., C. B. Chang and K. P. Dunn (1986), *Kalman Filter Algorithms for a Multi-Sensor System*, *Proc. IEEE Conf. Decision and Control,* pp. 570-574.

Salmond, D. J. (2000). *Mixture Reduction Algorithms for Target Tracking in Clutter*, *SPIE Signal and Data Processing of Small Targets,* 1305: 434-445.