The Impact of Treating Tracks as Contacts in Kinematic Data Fusion

D. J. Peters
Defence R&D Canada - Atlantic
9 Grove St., Dartmouth, NS B2Y 3Z7
(902) 426-3100 ext 350
dan.peters@drdc-rddc.gc.ca

Abstract

Level one data fusion is the process of combining data in order to estimate the kinematic state and the identity of individual entities. Its quintessential application is the tracking and identifica-tion of moving targets. The input data may be in the form of contacts, representing the minimal amount of processing of raw sensor data in order to achieve detection and parameter estimation, or in the form of tracks, indicating that some level one data fusion has already taken place.

The fusion of contact data is more straightforward than the fusion of track data, because the latter are generally cross-correlated. However, it sometimes happens that track data are received from multiple sources, while the underlying contact data are unavailable, thus necessitating the fusion of tracks. Several approaches to track-level fusion have been suggested, with varying degrees of theoretical merit and of practical feasibility. One approach that is sometimes used for track fusion is to extract the positional components of track data and to feed them into a centralised fusion algorithm that is designed for contact data. While this method has the advantage of simplicity, it fails to take into account the cross-correlations in the track data. The question remains whether one can nevertheless adopt this approach with acceptable results.

This paper documents a study to assess the impact of treating tracks as contacts on the kinematic aspect of data fusion. The simulation of air targets, detected by a generic radar with an Automatic Detection and Tracking (ADT) subsystem, is performed by a data fusion algorithm testbed that is written in Mathematica®. The performance of the tracks-as-contacts algorithm is compared with that of other methods for fusing track data (including Covariance Intersection) and with that of centralised contact-level fusion. Over the range of parameters and target behaviours considered here, the tracks-as-contacts method performs acceptably. The results suggest that this method, as applied to radar/ADT data, should be valid in practice, despite its theoretical flaws.

1. Introduction

It is typical for a tracking radar to be equipped with its own dedicated fusion node[1], such as is embodied in an Automatic Detection and Tracking (ADT) subsystem. The numerical data

[1] A fusion node is a system that creates and maintains its own set of tracks from input data that consist of contacts (as in this case), or tracks from other fusion nodes, or a combination of contacts and tracks. See the next footnote for a discussion of contacts versus tracks.

provided by the radar system are, in such cases, actually track[2] data from the ADT, while the underlying contact data may be unavailable to the operator or to whatever information systems interface with the radar. If such a radar system is being used in isolation, the fact that only the track data are being provided may not be a problem, provided the operator is willing to trust the output. But if a tactical picture is being compiled from the output of several sensors, one of which is the radar/ADT system under consideration, there arises the question of what procedure is appropriate for fusing these data with those of the other sensors.

One option is to reverse-engineer the radar/ADT system to make the contact data available for fusion. This option has the advantage that it allows for the use of many established contact-level fusion methods. If the contact data from the other sensors are commensurate with those of this sensor, in the sense that the same dimensions are being measured, then the techniques that apply to the fusion of data from a single sensor can be used, with very little modification, in order to fuse data from several sensors. However, the reverse-engineering procedure, if it is even possible, is expensive and may conflict with intellectual-property agreements with the radar's manufacturer.

A second option is to apply data fusion methods that are specifically designed for handling track data. This is in principle nearly as good as the first option, except that it assumes the availability of covariance data, which may not be provided by the ADT. Even if the covariances are available, such track-level methods are computationally intensive and more difficult to implement than contact-level methods.

A third option is to apply contact-level data fusion methods to the track data, where each track report from the ADT is treated as if it were a contact and used as input. This procedure can be made particularly straightforward by using only the positional components of the track data and discarding the velocity components. A difficulty with this approach is that contact-level methods have an underlying assumption that the errors in the input data are uncorrelated. This is clearly not the case for track data, and so the theoretical basis for the standard level one data fusion methods is undermined. Nevertheless, this method is sometimes used [Bégin *et al.*, 1994]. The present study seeks to begin to answer the question of whether we can, in practice, treat track data as contact data with acceptable results.

The data gathered in this study consist of contacts and tracks from several scenarios, each involving one or two simulated moving targets that are repeatedly detected by a simulated radar[3]. Section 2 describes the simulation environment. The contact data from the radar are fed into a fusion node, termed the *ADT node*, that simulates an ADT subsystem and produces track data as output. The study compares the performance of the following algorithms: (1) contact-level fusion using contacts as input (bypassing the ADT node), (2) track-level fusion using ADT tracks as

---

[2] Contact data represent the minimal amount of processing of raw sensor data in order to achieve detection of the target and localisation with respect to some measured parameters. Track data represent a synthesis of data from several contacts, gathered over a period of time, that are deemed to have originated from the same target. For the purpose of this study, a contact will consist only of a single measurement of an alleged target's position at a specific time, while a track will consist of estimates of both position and velocity, periodically updated with reference to an additional contact.

[3] Although multi-sensor data fusion provides the context and motivation for the question being addressed here, it is not necessary to use multiple sensors in order to investigate the effects of the tracks-as-contacts method. The point is that, with or without additional data, the tracks from the ADT node are being re-filtered.

input, and (3) contact-level fusion using ADT tracks as input. Section 3 describes the selected algorithms in more detail (see especially Figure 3). Comparisons of performance are made with respect to tracking accuracy, susceptibility to track seduction, and susceptibility to track loss. Section 4 describes the specific investigations and presents the results.

2. **The Simulation Environment**

A flexible and extensible multi-algorithm level one data fusion testbed, written in Mathematica®, was used to create all the simulated data in this study. This testbed allows the user to set several characteristics of targets, sensors, and fusion nodes. The running of a simulation within this framework consists of generating contact and track data over a given time interval. All the contact and track data remain available for subsequent analysis.

A target is defined by the time interval during which it exists, and the trajectory it follows during that time. A sensor (used in this study to simulate a generic radar) generates contact data, consisting of position measurements in spherical polar coordinates, at a user-specified time interval. The 360 degrees of bearing are divided into a number of sectors, each one being scanned after the one beside it (in a specified direction), in order to simulate radar rotation. The user can also define the probability of detection (assumed constant for all targets within the volume of coverage, which is defined by a minimum and maximum range and a minimum and maximum elevation angle) and the false alarm rate (given as an expected number of false alarms per sector per scan – the actual number is Poisson-distributed around this mean). The measurement of each polar dimension can be turned on or off (so in the present study, for example, the elevation angle measurement is turned off in order to simulate a two-dimensional radar[4]). For each contact that is generated by a successful measurement of a target, a Gaussian-distributed error is generated and added to each dimension of the measurement (the standard deviation for each dimension having been specified by the user).

A fusion node has a choice of association, track formation/deletion, and track maintenance algorithms, each with appropriate parameters. When a Kalman Filter is used for track maintenance, as in this study, its parameters include the choice between two- and three-dimensional tracking (two-dimensional in this study), the interpretation of the dimensions of the input contacts (x-y in this study[5]), the amount of error that the node attributes to the measurement (given as a standard deviation for each measured dimension), and a process model (including a transition matrix and a noise covariance matrix, both as functions of the time between updates).

The contact and track data are all stored, facilitating analysis. The true position and velocity of the target at any given time can be calculated, whenever desired, from the target definition.

3. **Scenarios and Parameter Settings**

[4] As with a real radar, the range measurement is the so-called “slant range” (i.e., the real range in three dimensions), but the trackers will interpret it as the range of the projection of the target into the horizontal plane, following common practice.

[5] The contacts originate as range-bearing but are transformed to cartesian coordinates for input to the node.

This section documents the choice of target, sensor, and node definitions, and parameter settings, that are used in the method comparisons of section 4.

3.1 *Scenario Scale; Sensor and Target Definitions*

The simulated sensor used in this study is intended to represent a radar, such as might be used on a modern frigate for the generation of the tactical picture. The 360 degrees of bearing are divided into twelve 30-degree sectors, each of which is scanned in turn, 1/12 second after the previous one, representing a 60 RPM rotation speed. The standard deviation of the error in bearing measurement is set to 0.01 radian, and the standard deviation of the error in range measurement is set to 150.0 metres. These values were not varied in this study. A maximum range of 50.0 km was also set for the sensor.

For most of this study, the probability of detection and the false alarm rate were idealized. Nevertheless, some of the fusion parameters (e.g., gate size) were sometimes set to values that take into account the possibility of more realistic measurements.

For the single-target scenarios, the target crossed the sensor's zone of coverage with a wave-like trajectory consisting of a series of circular arcs, alternating left and right, while maintaining an altitude of 1000 metres. The amplitude of the wave was kept constant at 5 km while the speed (constant with respect to time, for each target) and the curvature were varied. See Figure 1 for some of the trajectories. Table 1 shows the curvature, the speed, and the amount of time it takes to cross the zone of detection. Scenarios involving one such target will be called "class one" scenarios for the rest of this report.

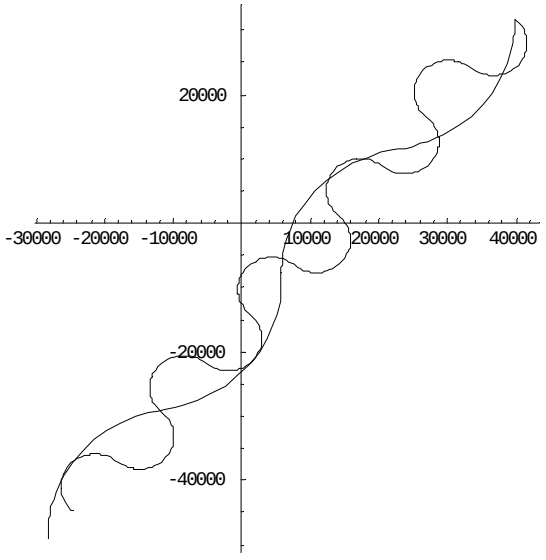

Figure 1. Sample class one scenario target trajectories (radius of curvature 5 km and 20 km)

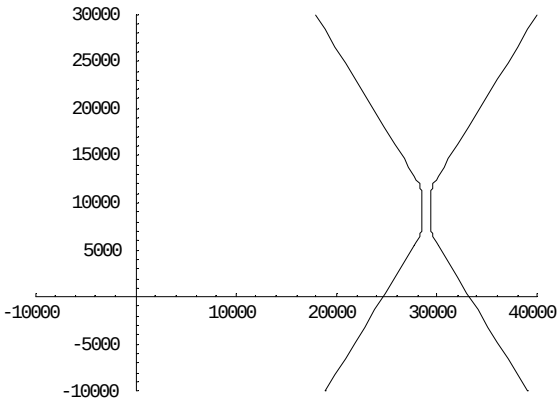

Figure 2. A class two scenario with a merge distance of 800 m

Radius of curvature	Speed

	150 m/s	300 m/s	450 m/s	600 m/s
20 km	712 s	356 s	237 s	178 s
10 km	782 s	391 s	261 s	195 s
6.67 km	875 s	437 s	292 s	219 s
5 km	1000 s	500 s	333 s	250 s

Table 1. Length of time for class one targets to cross the volume of interest

There are also scenarios (to be called "class two") involving two targets. In these scenarios, targets that start a little over 20 km apart and going at a constant speed of 400 m/s are converging with an angle of 60 degrees between their courses for 50 seconds, then they each make a turn (radius 5 km) of 30 degrees until they are side by side, then after five seconds of parallel flight they each make another such 30 degree turn in order to diverge at a relative angle of 60 degrees. See Figure 2. The merge distance (i.e., the distance between the parallel segments of the two targets) is a parameter to be varied.

3.2 ***Relationships Among Algorithms***

Three distinct fusion nodes are used in these simulations. The ADT node receives the original contact data and uses them to form its own set of tracks. (Although some of the parameters of the ADT node will be varied occasionally, the reader is invited to imagine that these parameters are fixed and are beyond our control.) Every time a track in the ADT node is updated due to association with a contact, the positional components of the updated track are sent to another fusion node (the *contact-level node* or *CL node*), whose parameters we will vary. In order to compare the results of this tracks-as-contacts (TAC) method with centralised contact-level (CCL) fusion, the CL node also receives the original sequence of contact data. See Figure 3.

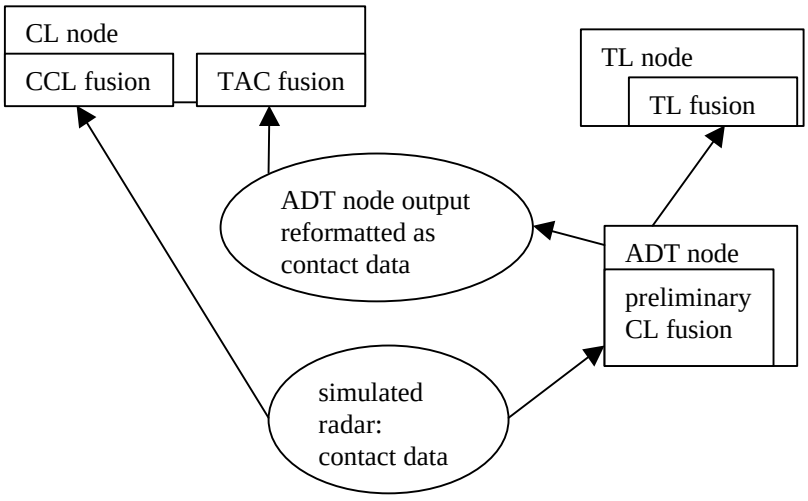

Figure 3. The flow of data in the data fusion methods under consideration

In order to provide other comparisons with the TAC method, the track output from the ADT node will, in some cases, also be fed into yet another fusion node, called the *track-level node* or *TL node*, which will apply fusion methods that make use of the full state estimate and the

associated covariance estimate from the ADT tracks. Here we will refer to these methods as track-level (TL) fusion, even when the methods in question do not properly account for cross-covariances among the data. Figure 3 shows schematically the relationships among the methods being compared in this study.

3.3 *Common Characteristics of the ADT Node and the CL Node*

In both the ADT node and the CL node, contact position data that are received in polar coordinates are converted to two-dimensional cartesian coordinates by a simple transformation (without de-biasing). It is assumed that the variance in range and the variance in bearing are constant and that there is zero covariance between range and bearing; the measurement noise covariance matrix used for tracking is derived from these assumptions, again by a simple transformation to cartesian coordinates. For purposes of this study, it is convenient to define a *measurement error ratio* (MER) for each dimension (range and bearing) as $\frac{s_{\text{believed}}}{s_{\text{true}}}$, where s_{believed} is the standard deviation that the fusion process at a given node attributes to the measurement error, and s_{true} is the actual standard deviation that was used to generate the errors in the simulated contacts. While the MER for range and the MER for bearing are independent in principle, these two MERs are kept equal to each other throughout this study, so the MER is treated as a single variable parameter for the fusion node.

A contact that is not associated with a prior track forms a tentative track. A tentative track is deleted if no contact is associated with it in the following scan. An assumed maximum target speed of 6000 m/s is used to judge the feasibility of association for this purpose. If an association is made between a contact and a tentative track, a full state estimate (x, y, and the corresponding components of velocity) and covariance are calculated from its two contacts. Such a track is called a preliminary track. A preliminary track becomes a confirmed track if at least two contacts are associated with it over the next three scans of the sensor following the upgrade to preliminary status; otherwise it is deleted. A confirmed track is deleted if no contacts are associated with it over five sensor scans in a row.

For a preliminary or confirmed track, the condition for a feasible association is $n^{\mathrm{T}}\mathbf{S}^{-1}n \le g$, where n is the innovation (i.e., the difference between the measured and predicted position), $\mathbf{S}$ is the innovation covariance (i.e., the sum of the measurement covariance and the positional part of the predicted track covariance), and g is a variable parameter. A strict nearest-neighbour rule (maximizing the number of associations, and minimizing the sum of the costs within that constraint [Blackman, 1986]) is used for association. The cost for association between a contact and a preliminary or confirmed track is $n^{\mathrm{T}}\mathbf{S}^{-1}n + \ln(\det\mathbf{S})$. The cost for association between a contact and a tentative track is a large constant (so a contact will always favour a preliminary or confirmed track over a tentative track, but all feasible associations among tentative tracks are equivalent and are in effect decided randomly).

A Kalman Filter (KF) is used for tracking. The process model is of the constant-velocity type, meaning that the target accelerations are attributed to the noise term. The process noise covariance matrix takes the form

$$\mathbf{Q}(T) = N\begin{bmatrix} \frac{T^3}{3} & 0 & \frac{T^2}{2} & 0 \\ 0 & \frac{T^3}{3} & 0 & \frac{T^2}{2} \\ \frac{T^2}{2} & 0 & T & 0 \\ 0 & \frac{T^2}{2} & 0 & T \end{bmatrix}, \quad (1)$$

where T is the elapsed time between track updates and N is a variable parameter.

3.4 ***The ADT Node***

For convenience, the ADT node was given a MER (defined in section 3.3) of unity[6]. The process noise parameter N (from equation 1) for the ADT node was set based on the premise that a real ADT system is not strongly susceptible to track loss for flight characteristics that are feasible for a military aircraft. The use of this premise will allow us to find a rough lower bound for a realistic value of N, as described in this section.

Four class one scenarios were run, ten times each. The contact data for each run were fed into five copies of the ADT node, each having a different value of N, in order to find (roughly) the threshold for track loss, given a standard 99% gate for association. (The gate size parameter g is set to 9.21. This is a "99% gate" because the probability, assuming the validity of the process model, for a contact that originates from the correct target to meet the gate criterion is 0.99. Although the false alarm rate was idealized for these runs, the use of a gate would be necessary in the realistic case of numerous false alarms.) Table 2 shows the number of runs, out of ten, in which there were two or more tracks corresponding to the target over the duration of the run. In each case the ratio $\frac{N}{a^2}$ (where N is the process noise parameter and a is the acceleration of the target's turn), which has units of time, was varied from 1 to 3 seconds, in steps of 0.5 seconds.

Scenario		N/a^2				
Curvature (km^{-1})	Speed (m/s)	1.0 s	1.5 s	2.0 s	2.5 s	3.0 s
0.05	150	5	4	1	0	0
0.1	300	5	1	0	0	0
0.15	450	8	6	4	4	2
0.2	600	7	2	0	0	0

Table 2. Number of times node suffers track loss in ten runs, for various class one scenarios

[6] While this setting in a sense idealizes the ADT node (thus perhaps, weighing the method comparisons artificially in favour of those methods that use the ADT), this idealization is balanced by the rough approach to setting the process noise parameter.

The table suggests that a value of 2 seconds for the ratio $\frac{N}{a^2}$ is very roughly where the node becomes vulnerable to track loss, given the gate condition and the measurement errors used here (although the threshold appears to be somewhat higher in the third scenario listed, for unknown reasons). For the four scenarios tested here, these thresholds correspond to noise parameter values of about 2.5 m^2/s^3, 160 m^2/s^3, 1800 m^2/s^3, and 10000 m^2/s^3. Even the last case represents flight characteristics which are within the capability of modern military aircraft, so arguably a value of 10000 m^2/s^3 for the noise parameter is realistic for an ADT system. Only two values (150 m^2/s^3 and 1000 m^2/s^3) were used in the present study, and as we will see, there appears to be little point in letting it go higher than the latter value for our purposes. A higher value of the noise parameter in effect makes the KF attach greater weight to the measurement; thus the higher this number, the more the tracks-as-contacts method will approach the centralised fusion method. The lower value of this parameter was used in order to point out the kind of differences that can be expected between these two methods, while the quantitative results that arise from using the higher value of this parameter are more realistic.

While a 99% gate was used in the above preliminary analysis, a much larger gate size was used in the investigations of sections 3.5, 4.1, and 4.2, in order to avoid complications related to track loss while investigating other results such as tracking accuracy.

For purposes of providing input to other nodes, the ADT node will send only fully-formed (preliminary or confirmed) tracks, never tentative tracks.

3.5 ***The CL Node***

The process noise parameter will be fixed at 1000 m^2/s^3 in the CL node.

The MER (defined in section 3.3) will be kept at unity in the CL node in CCL fusion. But a fair comparison between the CCL and TAC methods requires some account to be taken of the differences in the nature of their inputs[7]. In particular, it does not make sense to use the same MER for the CL node in TAC fusion as is used for the CL node in CCL fusion. The amount of measurement error that the node attributes to its inputs should in principle reflect some prior knowledge (i.e., through experimentation) of the amount of actual random error that is likely to appear. Indeed, one might expect the preliminary fusion in the ADT to have filtered out some of the random error, so the CL node's MER should be set lower for the TAC method than for the CCL method. This section documents the preliminary tests that were used to find an appropriate value of the MER to use for the CL node in the TAC method for the rest of the study[8].

---

[7] If the track covariance data from the ADT node are made available to the CL node for TAC fusion, then the positional components of the ADT track covariance can be used directly as the measurement covariance, instead of calculating the measurement covariance from a MER that is fixed for the duration of a given run. Here we will assume that the ADT track covariances are not available.

[8] In section 3.3 it was pointed out that the MER for the range and the MER for the bearing are independent in principle. The effects of changing one versus changing the other are (presumably) highly scenario-dependent, so there is little value in treating them as independent. They are kept equal to each other throughout this study.

Contact data were collected in four separate runs of the class one scenario in which the target has a speed of 300 m/s and a turn radius of 10 km. Each set of contact data was fed through the TAC fusion procedure, with a process noise parameter of 150 m^2/s^3 in the ADT node and several different values of the MER in the CL node. The MER was varied in order to find where the tracking accuracy would be optimized with respect to position error and velocity error. Since we are here concerned with tracking accuracy rather than with track loss, the gates for both the ADT node and the CL node were enlarged so that track loss would not be an issue.

Figures 4 and 5 present the time-averaged absolute position error and the time-averaged absolute velocity error, each averaged over the four runs for each of fifteen values of the MER.

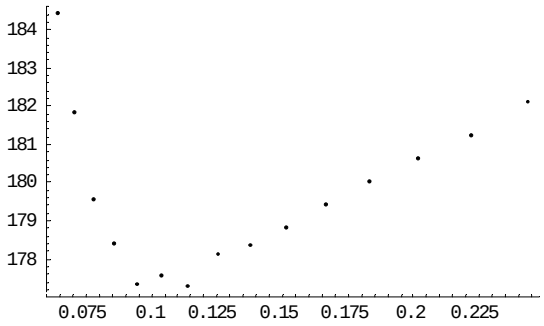

Figure 4. Position error (metres) in TAC fusion as a function of MER.

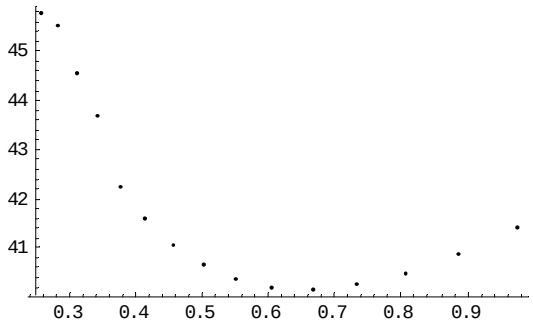

Figure 5. Velocity error (m/s) in TAC fusion as a function of MER.

It appears from these figures that a MER of about 0.10 optimizes the position error and a MER of about 0.67 optimizes the velocity error, for TAC fusion in this scenario. A similar procedure, for a process noise parameter value of 1000 m^2/s^3 in the ADT node, leads to optimal MERs of 0.25 and 1.0 (optimized respectively for position error and velocity error) in the same scenario. We will fix the MER in the CL node for TAC fusion to 0.25, representing a rough compromise among the four optimized values, for the rest of the study[9].

3.6 ***The TL Node***

In order to provide additional comparisons for the TAC method, the output of the ADT node output is also input to the TL node, which makes use of the entire state estimate and covariance of each track report. The TL node maintains its own list of (internal) tracks, which are created, updated, and deleted based on the track update reports from the ADT node.

The feasible-association condition in the TL node is $\left(\hat{\mathbf{x}}_{\text{in}} - \hat{\mathbf{x}}_{\text{pr}}\right)^{\text{T}} \left(\mathbf{P}_{\text{in}} + \mathbf{P}_{\text{pr}}\right)^{-1} \left(\hat{\mathbf{x}}_{\text{in}} - \hat{\mathbf{x}}_{\text{pr}}\right) \leq g$, where $\hat{\mathbf{x}}_{\text{in}}$ is the state estimate from the input track report, $\hat{\mathbf{x}}_{\text{pr}}$ is the predicted state estimate from the

[9] Only a rough value is of use to us, because realistically the MER in the CL node will usually be farther from its overall optimum value (however one might choose to define such a thing) in TAC fusion than in CCL fusion. In CCL fusion, all that is needed in order to determine a near-optimal value for the MER is knowledge of the characteristics of the sensor as it performs in the current environmental conditions. In TAC fusion, the optimal MER is also highly scenario-dependent in that different flight trajectories and speeds will have been handled by the ADT node with different degrees of effectiveness, and this variation in performance will be reflected in the input to the CL node.

internal track, $\mathbf{P}_{\text{in}}$ and $\mathbf{P}_{\text{pr}}$ are their respective covariances, and g is again a gate size parameter[10]. The expression on the left-hand side of the inequality above is also used as the cost when applying the nearest-neighbour rule for association. An input track report that is not associated with an internal track forms a preliminary track. Confirmation of a preliminary track, and deletion of preliminary or confirmed tracks, is handled by the same rules as in the other nodes (section 3.3).

The track updating rule followed by the TL node, in order to calculate the new track state $\hat{\mathbf{x}}_{\text{new}}$ and its covariance $\mathbf{P}_{\text{new}}$, will either be $\mathbf{P}_{\text{new}}^{-1} = \mathbf{P}_{\text{in}}^{-1} + \mathbf{P}_{\text{pr}}^{-1}$ and $\mathbf{P}_{\text{new}}^{-1}\hat{\mathbf{x}}_{\text{new}} = \mathbf{P}_{\text{in}}^{-1}\hat{\mathbf{x}}_{\text{in}} + \mathbf{P}_{\text{pr}}^{-1}\hat{\mathbf{x}}_{\text{pr}}$ or $\mathbf{P}_{\text{new}}^{-1} = w\mathbf{P}_{\text{in}}^{-1} + (1-w)\mathbf{P}_{\text{pr}}^{-1}$ and $\mathbf{P}_{\text{new}}^{-1}\hat{\mathbf{x}}_{\text{new}} = w\mathbf{P}_{\text{in}}^{-1}\hat{\mathbf{x}}_{\text{in}} + (1-w)\mathbf{P}_{\text{pr}}^{-1}\hat{\mathbf{x}}_{\text{pr}}$, where w is the real number in the interval [0,1] that minimizes the determinant of $\mathbf{P}_{\text{new}}$. The former rule neglects the effects of cross-covariance between the internal track and the input track, and is thus close in spirit to the TAC method. Here we will call it "Naï ve track-level fusion"[11]. The latter rule is one version of the Covariance Intersection (CI) method [Julier and Uhlmann, 2001]. This more sophisticated track-level algorithm does not depend on any assumptions about the cross-covariance between the internal track and the input track. The need to minimize the determinant of the new state covariance, however, means that it is relatively computation-intensive.

4. **Method Comparisons: Observations and Discussion**

The methods being compared in this section are:

- Centralised contact-level fusion (bypassing the ADT node).
- Tracks-as-contacts (TAC) fusion, using a constant MER (defined in section 3.3) of 0.25.
- Naï ve track-level fusion (used only in the tracking accuracy investigation).
- The Covariance Intersection (CI) algorithm for track-level fusion.

See sections 3.2 to 3.6 for an explanation of the methods used. The methods are compared with respect to tracking accuracy, susceptibility to track seduction, and susceptibility to track loss.

4.1 ***Tracking Accuracy***

For the tracking accuracy comparison, the sixteen class one scenarios of Table 1 were each run through the four different tracking methods. Each of the two process noise parameter values from section 3.4 were used in the ADT node. The position error and velocity error, averaged over all track updates, are respectively shown in Tables 3 and 4 for all four methods, for all sixteen scenarios. ("Low N" and "High N" refer respectively to process noise parameter values of

[10] This gate condition does not take into account cross-covariances between the internal track and the input track, but this limitation is not an issue in the present study. The effect of neglecting the cross-covariances in association is to underestimate the statistical difference between the two tracks [Bar-Shalom and Li, 1995], in effect widening the gate. This accords with the fact that we have deliberately kept the gates very large (except in section 4.3). This underestimation of the statistical difference is particularly large when previous reports from the same input track have been used to update the internal track in question. For purposes of distinguishing between multiple targets (as in section 4.2), the consequence is presumably that the association decision will be unfairly biased toward making the correct decision, which is hardly a problem! On the other hand, the cross-covariances are important where track updating is concerned.

[11] The term "track-level" is used here because the full track state and covariance are explicitly used in the update formulae. The fact that the cross-covariance is neglected, however, implies that the term is (arguably) being abused.

150 m^2/s^3 and 1000 m^2/s^3 in the ADT node.) The overall state estimate error, normalized by the track covariance (i.e., the quantity $(\hat{\mathbf{x}}-\mathbf{x})^{\mathrm{T}}\mathbf{P}^{-1}(\hat{\mathbf{x}}-\mathbf{x})$, where $\hat{\mathbf{x}}$ is the estimate, $\mathbf{x}$ is the true state vector, and $\mathbf{P}$ is the covariance) and similarly averaged, is presented in Table 5.

These results are derived only from a single run for each scenario. In order to obtain a rough measure of the statistical variation, the fourth scenario listed was subsequently run twenty times through CCL fusion. Each of the resulting standard deviations appears in the appropriate cell of its respective table.

Scenario		Fusion Method (and process noise parameter setting)						
Curvature (km^{-1})	Speed (m/s)	CCL fusion	TAC (low N)	TAC (high N)	Naïve TLF (low N)	Naïve TLF (high N)	CI TLF (low N)	CI TLF (high N)
0.05	150	170	150	180	150	170	150	170
0.05	300	150	140	150	180	150	140	150
0.05	450	150	180	150	310	170	180	150
0.05	600	200 ± 10	350	210	570	280	330	200
0.1	150	160	140	170	140	150	140	160
0.1	300	180	210	180	330	190	200	180
0.1	450	200	340	210	600	280	320	200
0.1	600	240	530	280	840	440	490	240
0.15	150	170	150	170	170	160	150	170
0.15	300	170	240	170	420	210	230	170
0.15	450	210	450	230	780	350	420	210
0.15	600	320	730	370	1030	600	670	320
0.2	150	170	150	170	180	160	150	170
0.2	300	170	280	180	510	230	262	170
0.2	450	230	540	260	890	420	500	230
0.2	600	370	730	430	950	720	660	360

Table 3. Position error (metres) for each fusion method in class one scenarios

Scenario		Fusion Method (and process noise parameter setting)						
Curvature (km^{-1})	Speed (m/s)	CCL fusion	TAC (low N)	TAC (high N)	Naïve TLF (low N)	Naïve TLF (high N)	CI TLF (low N)	CI TLF (high N)
0.05	150	41	45	62	20	32	24	41
0.05	300	40	42	57	41	35	33	40
0.05	450	51	45	58	79	54	60	51
0.05	600	73 ± 5	61	66	131	85	102	72
0.1	150	41	46	63	23	31	24	41
0.1	300	49	47	62	70	48	54	49
0.1	450	76	65	69	144	92	111	76
0.1	600	127	110	96	230	159	190	127
0.15	150	41	45	62	28	31	26	41
0.15	300	55	48	58	97	62	73	55
0.15	450	105	89	79	202	131	160	106
0.15	600	170	158	113	293	216	263	170
0.2	150	41	44	61	35	32	29	41
0.2	300	65	57	65	122	77	93	65
0.2	450	128	115	91	239	162	200	129

0.2	600	223	213	161	338	279	314	223

Table 4. Velocity error (m/s) for each fusion method in class one scenarios

For the lower ADT node process noise parameter, the TAC method estimates the positions better than the centralised fusion method in the case of the slower targets, but worse in the case of the faster targets. The higher ADT node process noise nearly eliminates the difference between these two methods. There is very little difference between TAC and CI at either setting. The Naï ve track-level fusion is much less accurate than the other methods in several cases.

The TAC method, as compared with centralised fusion, shows the opposite trend with respect to velocity error than with respect to position error. That is, the TAC is more accurate than centralised fusion for the faster targets and less accurate for the slower targets. The CI method, on the other hand, shows the same trend for velocity error as it does for position error. Naï ve track-level fusion is again noticeably less accurate than the other methods for the faster targets.

Scenario		Fusion Method (and process noise parameter setting)						
Curvature (km^{-1})	Speed (m/s)	CCL fusion	TAC (low N)	TAC (high N)	Naïve TLF (low N)	Naïve TLF (high N)	CI TLF (low N)	CI TLF (high N)
0.05	150	1.5	4.3	4.9	2.5	2.2	1.6	1.5
0.05	300	1.3	3.8	4.2	2.8	2.0	1.6	1.3
0.05	450	1.5	4.9	4.5	4.7	2.3	2.4	1.5
0.05	600	1.8 ± 0.1	8.4	5.3	8.4	3.3	3.8	1.8
0.1	150	1.4	4.0	4.7	2.4	2.1	1.5	1.4
0.1	300	1.5	5.3	4.7	4.6	2.3	2.2	1.5
0.1	450	1.8	8.6	5.5	9.0	3.4	4.1	1.8
0.1	600	2.5	13.4	6.8	13.1	5.1	6.9	2.5
0.15	150	1.4	4.2	4.8	2.7	2.1	1.5	1.4
0.15	300	1.6	6.2	4.7	6.1	2.6	2.9	1.6
0.15	450	2.3	11.5	5.9	12.3	4.4	6.0	2.3
0.15	600	3.2	19.0	9.0	15.7	7.0	9.8	3.2
0.2	150	1.5	4.2	4.7	2.9	2.1	1.7	1.5
0.2	300	1.6	7.4	4.8	7.7	2.9	3.6	1.6
0.2	450	2.7	14.6	7.0	13.9	5.5	7.7	2.7
0.2	600	4.5	18.6	11.0	15.9	9.4	11.1	4.5

Table 5. Normalized state error for each fusion method in class one scenarios

The normalized state errors show that the TAC method produces overly optimistic covariances. This result suggests that when such a method is used, any Area of Uncertainty information from the fusion node should be reinterpreted accordingly.

With the higher value (1000 m^2/s^3) of the ADT node process noise parameter, there is hardly any noteworthy differences among the centralised fusion, TAC fusion, and CI results – except that the TAC method achieves better velocity estimation for the faster targets. Note that these results (for the TAC method) depend on the believed positional errors in the "contact" input – represented in this study by the MER defined in section 3.3. The results of that section indicated that a higher

value for the MER (approaching and perhaps eventually surpassing unity) becomes appropriate as the ADT node process noise parameter is increased. But to increase these parameters can only further obscure the differences between centralised fusion and TAC fusion. Thus the claim (made in section 3.4) that there is no point, within the scope of the present study, in going beyond 1000 m^2/s^3 for the ADT node process noise parameter is justified – despite the fact that even this value is (arguably) unrealistically low.

4.2 *Track Seduction*

The next investigation uses the two-target class two scenarios described in section 3.1. For four different values of the merge distance, the centralised fusion method, the TAC method, and the CI method were each tested in twenty different runs, to see which methods were more or less likely to confuse the two targets. Both of the values of the ADT node process noise parameter that were used in earlier sections are used here for the latter two methods.

Table 6 shows the number of runs, out of twenty, in which each of these configurations ended up with the correct match between the converging and diverging parts of the tracks. The parenthesised number shows the number of runs in which every association was correct. When this number is not shown, it is equal to the previous number.

Merge distance	Fusion method (and pre-fusion process noise)				
	CCL	TAC (low N)	TAC (high N)	CI TLF (low N)	CI TLF (high N)
200 m	1 (0)	0	1 (0)	0	1 (0)
400 m	2	2 (1)	5 (3)	0	2
600 m	12	3	12	0	12
800 m	19	14	19	14	19

Table 6. Number of times track seduction is avoided for various class two scenarios

These three methods appear to perform equally when the higher value of the ADT node process noise parameter is used – except that the TAC method appears to have outperformed the others at a merge distance of 400m.

4.3 *Track Loss*

The problem of track loss was touched on in section 3.4 when the ADT node process noise parameter was set. The question now arises of how to make a meaningful comparison between the TAC method and the centralised fusion method where susceptibility to track loss is concerned. Just as a meaningful comparison of tracking accuracy between the two methods requires them to have different MERs (see section 3.5), so a meaningful comparison of susceptibility to track loss between the two methods requires them to have different gate sizes.

In the investigations thus far, the probability of detection and the false alarm rate have both been idealized. In such conditions, the use of an association gate has value only insofar as it reduces the computational load of the system by quickly rejecting some potential associations (between an internal track and an input contact or track) as unfeasible. But in the real world, the fact that the

real target may occasionally be missed by the sensor means that if a gate condition is not used, or a gate is set too large, there is the danger of a track being seduced by false alarms. This important role of an association gate belongs primarily to whichever fusion node receives the real contact data. In the case of any method that uses the ADT node output, that node will already have filtered out most of the false alarms, so the CL (or TL) node is free to use a much larger gate size, rendering this node less susceptible to track loss. However, the fact that only fully-formed tracks from the ADT node are passed on implies that, if the ADT node itself has suffered track loss, several contacts will in effect be missed by the CL (or TL) fusion node.

Here, we set the sensor's probability of detecting the target at 0.95 and set the average number of false alarms per sector per scan at 0.125 (thus producing an average of 1.5 false alarms per full rotation of the sensor). The ADT node will use a 99% gate (that is, a gate condition of $\boldsymbol{n}^{\mathrm{T}}\mathbf{S}^{-1}\boldsymbol{n} \leq 9.21$) as will the CL node in the case of centralised fusion, while the CL node will use a 99.9% gate (that is, a gate condition of $\boldsymbol{n}^{\mathrm{T}}\mathbf{S}^{-1}\boldsymbol{n} \leq 13.82$) for TAC fusion. A run is made for each of the sixteen class one scenarios used earlier. The relevant measure of performance here is the Track Continuity, which is to say the number of tracks (ideally one) that are associated at least once with the target. Both ADT node process noise parameter values are used.

Scenario		Fusion method (and pre-fusion process noise parameter setting)				
Curvature (km^{-1})	Speed (m/s)	Centralised	TAC (low N)	TAC (high N)	CI TLF (low N)	CI TLF (high N)
0.05	150	1	1	1	1	1
0.05	300	1	1	1	1	1
0.05	450	1	1	1	1	1
0.05	600	1	1	1	2	1
0.1	150	1	1	1	1	1
0.1	300	1	1	1	1	1
0.1	450	1	1	1	6	1
0.1	600	3	3	2	7	2
0.15	150	1	1	1	1	1
0.15	300	1	1	1	3	1
0.15	450	1	3	1	13	1
0.15	600	5	9	3	11	5
0.2	150	1	1	1	1	1
0.2	300	1	1	1	7	1
0.2	450	4	8	3	17	4
0.2	600	13	12	15	13	13

Table 10. Number of tracks produced to follow target in various class one scenarios

These results indicate that the TAC method is no more susceptible to track loss than is centralised fusion, given a realistic value for the ADT node process noise parameter, at least for the range of target behaviours considered here. Arguably, the association gate for the CL node as used in TAC fusion could be made even larger, and such a change could only strengthen this result.

5. **Conclusions**

The relative performance of the tracks-as-contacts method and the centralised fusion method were compared, in several simulated runs of one-target and two-target scenarios, with respect to tracking accuracy, susceptibility to track loss, and susceptibility to track seduction. The results that were obtained with the lower value (150 m^2/s^3) of the ADT node process noise parameter show that the tracks-as-contacts method might be expected, in general, to have greater errors in position estimation (at least in the case of faster targets) and to be more susceptible to track seduction than the centralised fusion method. However, the higher value (1000 m^2/s^3) of the ADT node process noise parameter is apparently high enough to erase any significant difference in performance between these two methods, even though a realistic value for this parameter would arguably be higher still. An exception to this comparison is that the velocity estimation of the tracks-as-contacts method is apparently better for faster targets and worse for slower targets than that of centralised fusion.

Two track-level fusion methods were also considered in order to provide further comparisons. There was nothing in the present set of results to indicate any significant inferiority in the tracks-as-contacts method as compared to Covariance Intersection (again, except in the velocity estimation of slower targets). The "Naï ve" track-level fusion method (considered only in the tracking accuracy investigation) was much worse than the tracks-as-contacts method, despite a similar underlying philosophy.

The practice of treating the errors in the output of an ADT as mutually uncorrelated would appear to be validated, at least for the range of simulated scenarios and parameters considered here. However, it should be noted that the tracks-as-contacts method produces overconfident covariances, as shown by the normalized state errors in Table 5.

A more thorough exploration (by simulation) of the practical implications of treating tracks as contacts would require detailed knowledge of the design and performance of the original sensor as well as of the fusion node producing the tracks. A study involving real data would of course be ideal.

6. **References**

[Bar-Shalom and Li, 1995] Yaakov Bar-Shalom and Xiao-Rong Li. *Multitarget-Multisensor Tracking: Principles and Techniques*. Yaakov Bar-Shalom, Storrs, CT, 1995.

[Bégin *et al*., 1994] F. Bégin, E. Boily, T. Mignacca, E. Shahbazian, and P. Valin. "Architecture and Implementation of a Multi-Sensor Data Fusion Demonstration Model Within the Real-Time Combat System of the Canadian Patrol Frigate", AGARD symposium on Guidance and Control for Future Air-Defence Systems, Copenhagen, 17-20 May 1994, AGARD-CP-555, pp. 28.1-28.8.

[Blackman, 1986] Samuel S. Blackman. *Multiple-Target Tracking with Radar Applications*. Artech House, Norwood, MA, 1986.

[Julier and Uhlmann, 2001] Simon Julier and Jeffrey K. Uhlmann. "General Decentralized Data Fusion with Covariance Intersection (CI)", in *Handbook of Multisensor Data Fusion*. CRC Press, Boca Raton, FL, 2001.