TARGET TRACKING

TYPES OF TARGET TRACKING

Target-tracking problems can be broadly categorized into four generic classes, as follows:

1. Sensor tracking of a single (bright) target
2. Tracking of targets that are large
3. Tracking of targets that are medium-sized
4. Tracking of targets that are small

The target sizes indicated in this list are in terms of the number of resolution elements or pixels. The primary differences in each of these problems are the algorithms used in the signal, image, and track processing. The algorithms and processing methods used for each of these problems are substantially different. Of course, the system and its hardware must be designed to be compatible with and take advantage of the appropriate processing methods for an application. These methods depend on the types of algorithms used in the processing.

Sensor Tracking of a Single Target

Examples of the Class 1 tracking problem above are a gimbals-mounted telescope following a planet or an interceptor pursuing a target. Tracking is achieved typically through signal processing to locate the target in the field of view (FOV) and then guiding gimbals of the telescope (or seeker of an interceptor) to drive the target near the center of the FOV. This type of tracker is sometimes referred to as a closed-loop tracker. The signal from the signal processor is typically temporally filtered before it is sent to the telescope gimbal driver, and this filter may be fairly simple compared with a Kalman filter. For some applications in this class, the target may be small initially and then grow in size, such as with an interceptor sensor. Note that for this class of tracking problem, there is often only a single target in the FOV, and it is bright (high contrast) relative to any false signals or background objects. Consequently, uncertainty about which are the target pixels and which are not is not a major issue. Thus, using sensor data obtained from a sequence of measurements over time points is fairly straightforward. In the future, however, as these systems are required to operate under more challenging conditions, the tracking algorithms developed for the other three tracking categories may be required. More challenging conditions might include initially tracking a dim target or a target with accompanying debris or countermeasures.

Tracking Large Targets

An example of a Class 2 tracking problem is the use of low-altitude surveillance sensors for locating or tracking ground targets. For that example, the target extent could cover many resolution elements that provide extensive detailed information about each target of interest and the other objects in the scene. With a large target, the components of the target might be identified such as wheels, tank treads, or airplane wings. Tracking and target recognition for this class of target are typically achieved through image-processing or possibly image-understanding methods. With a large target, image-processing methods could be used to determine the details of construction as well as shape of the target from only a single frame of data. With that information, normally a good probability of correctly identifying the target type for each target in the FOV with only a single frame of data is achievable. Furthermore, normally enough information exists to simply sort out and track each target over time. That is, no confusion exists about which target in one frame of data is the same target in another frame of data. Typically, image-understanding processing can be more complex than traditional image processing, but should be more versatile in handling various types of targets, including their shadows and obscurations caused by objects such as trees.

Tracking Medium-Sized Targets

An example of the Class 3 problem is use of medium altitude surveillance sensors for tracking ground vehicles. Tracking is typically achieved using either a correlation tracker or a centroid tracker. These methods are needed to deal with and possibly take advantage of the extent of the target. A target in this class might be 20 pixels in diameter. Typically, with a target of that size, not enough information useful for image processing is available yet the extent should be taken into account. For example, for each time point, the location of the target needs to be established relative to some point on the target. That point on the target is then used to track the target over time. Thus, a consistent point is needed on the target so that the estimated motion is not corrupted by use of different points on the target over time. With a *correlation tracker*, the processor finds the location that maximizes the correlation between the current image of the target and a reference target image. The appearance of the target can depend on the aspect angles, which can change over time, and that complicates the processing. The algorithms for correlation tracking usually are designed to accommodate all possible values of the aspect angles. A *centroid tracker* uses the shape and possibly the signal amplitude profile to establish a point on the target each time it is observed. For this type of target, the size and shape of the target helps in determining which target in one frame of data is the same target in another frame of data. This information can be corrupted by false signals, obscurations by other objects, similar nearby objects, and random motions of the sensor line of sight.

Tracking Small Targets

An example of the Class 4 tracking problem is the use of ground-based surveillance sensors for tracking aircraft, cruise missiles, or ballistic missiles. Tracking small targets is achieved using what is commonly referred to as multiple target tracking methods. This class of problem is often referred to as *multiple target tracking*, even though it includes both single, small target tracking with persistent clutter or false signals and the tracking of multiple small targets that may be close or crossing and with possibly persistent clutter or false signals. For this class of track-

ing problem, uncertainty can exist as to which target is responsible for a measurement, because of closely spaced measurements. This uncertainty greatly complicates the processing. The processing function that decides how to relate the current measurements to the existing target tracks or prior data is called *data association* (sometimes referred to as correlation). The data-association function deals with the ambiguity of which measurement comes from each target that was observed earlier. In most small target tracking, there is not enough information in a sensor measurement to know which target (if any) was the source of the measurement. Therefore, the wrong measurement might be used to update a target track, and this type of error is often referred to as a *misassociation.*

Impact of Target Conditions on Processing Methods

Not only does each of these classes of target-tracking problems call for different processing algorithms, but also the processing concepts and the algorithm development methodologies can be very different. For example, for the development of the small target-tracking algorithms, typically a simulation is used to generate data to test the algorithms. Many runs (instances) of a Monte Carlo simulation can then be used to obtain performance with reasonable confidence (given a sufficiently detailed simulation). On the other hand, to test image-processing algorithms for tracking large targets, usually a set of images of real scenes containing targets is used. Because typically it is difficult to obtain and test many hundreds of images, the methodology for algorithm development and performance evaluation of large target-tracking algorithms is very different from that for small targets. In addition, the type of algorithms used for image processing are very different from those used for tracking small targets.

The track-processing methods used also depend on the type of sensor or suite of sensors that provide the data for a system application. The phenomena encountered for each type of sensor can have a significant impact on the type of processing required. For example, there are methods that can be used with an active sensor, such as radar, that cannot be used with a passive sensor, such as an electro-optical sensor. Therefore, some specialized tracking techniques have been developed for some sensors that are not useful for others. In addition, multiple-sensor systems require special considerations beyond those tracking approaches used for single-sensor tracking. The type of target and its environment also have a major impact on the selection of the appropriate algorithms and the sequence of functions. The sequence of processing functions is often referred to as the *processing chain* or *algorithm architecture.* The algorithm architecture and specific algorithms appropriate to tracking ground targets can be very different from those used for surveillance of ballistic missiles.

INTRODUCTION TO SMALL TARGET TRACKING

Because each class of tracking problem poses different algorithm development issues, this article will concentrate on only one class of tracking, namely, tracking of small targets using multiple target-tracking methods. Multiple target tracking is a relatively new field. The first book dedicated exclusively to multiple target tracking was published in 1986 (1) and a number of books have been published since then (2–7). In addition to the numerous papers and reports in the open literature (too numerous to be listed here), there is an on-going series of annual SPIE conferences concerned exclusively with signal and data processing of small targets that started in 1989 (8). This article freely extracts and paraphrases material from some of the author's prior documents (9–15) and view graphs (16, 17).

For this discussion, a *small target* is characterized as one that does not provide enough data for traditional automatic target recognition (ATR) using a single frame of data (9). In contrast, a target large enough for ATR typically extends beyond a diameter of about 10 resolution elements, for example, larger than 10 by 10 pixels. Note that it is not uncommon to refer to all objects as targets whether they are of interest or not. Small targets include:

- Point source targets
- Small extended targets, including unresolved closely spaced objects
- Clusters (groups) of point source and small extended targets

The width of a typical point source target in the field of view is from 1 pixel to about 12 pixels (resolution elements), depending on the sensor design, for instance, the sensor spread function. Although the processing of point targets has been studied extensively, there are still many interesting challenges in this field. In contrast, the state of the art for processing small-extended objects and clusters is far less mature, but interest is growing. Small targets that are not point-source objects include both small-extended objects and unresolved closely spaced objects, sometimes called clumps. An *unresolved closely spaced object* (UCSO) refers to a measurement caused by a number of targets for which the location of each individual target could not be established by the signal processor because they were all too close relative to the resolution of the sensor. In many current systems, the data forwarded by the signal processor to the tracker do not give any indication of which measurement is probably a UCSO. Although UCSOs and small, extended targets provide little detailed information useful for ATR, they do exhibit some shape and size information that might be useful in tracking. In addition, an extended object may partially obscure rather than add to the background or be partially obscured. The apparent size and shape of the target can differ from sensor to sensor and over time; this may have to be taken into account. Similarly, cluster processing offers significant advantages and challenges.

Developing a tracker to follow a single small target without false signals or persistent clutter is not particularly difficult. In contrast, developing a tracker is difficult for challenging conditions with resolved or unresolved closely spaced measurements caused by false signal, persistent clutter, or close targets plus possibly countermeasures or abrupt target maneuvers. Distributed multiple sensors that exhibit platform location uncertainty and

residual sensor measurement biases pose additional challenges as do multiple sensors that exhibit different sensor phenomena, such as fusing data from radars and IR sensors.

There have been many improvements in small target processing algorithms in recent years. These advancements are, in part, because of opportunities to implement more advanced and complex algorithms because of the greatly increased capabilities of processors. Ongoing development of algorithms for new systems and upgrading existing systems is driven by improved sensors, increasingly demanding system requirements, processor and communications hardware limitations, severe operating environments, efficacious countermeasures, and challenging threat scenarios. There is growing interest in the ability to track dim targets or in a moderate to dense population of threshold exceedances caused by clutter, false signals, or targets that are close or crossing.

A common approach for processing target data from a single sensor is to partition the processing into the two major functions of signal processing and data processing, as shown in Fig. 1. The *signal processing* usually converts the sensor data into digital form; processes and thresholds the data to detect potential targets; and establishes the parameters of the measurement vector for each threshold exceedance. The type of signal processing algorithm used is highly specialized, based on the type of sensor. For systems that require detection of dim targets, multispectral sensor processing and the more complex multiple frame processing should be considered, such as the so-called track-before-detect and the velocity filter approaches. The signal processor forwards the measurements to the data processor. Measurements are sometimes referred to as reports, returns, observations, hits, plots, or threshold exceedances, depending on the type of sensor. Typically, the signal processor forwards the measurements to the data processor in the form of a sequence of frames of data. *A frame of data* is simply a collection of measurements. For radar, a frame might consist of all the measurements from a single dwell, and for an IR imaging sensor, a frame of data might be all the measurements from a single look of the imager. Note in Fig. 1 the possible use of track data at the signal processing level. There is a growing recognition of the importance of using all available information in every stage of the processing and in the feedback of information (9).

The primary functions of the *data processing* of sensor data are tracking and target classification or discrimination; however, estimation of sensor registration biases, sensor resource management, situation assessment, combat identification, target weapon assignment, and other functions may also be included. Typically, a target evolves through the three processing stages of (1) track initiation, (2) track maintenance, and (3) track termination, see Fig. 2. The basic tracking functions for each stage are data association, filtering, and the track promotion and demotion logic. As mentioned earlier, the data-association function deals with the ambiguity of which measurement is from the same target as that of a track or a sequence of prior measurements.

The *filter* uses the measurement vector to update the target state estimate, its error covariance matrix, and possibly additional information. The elements of the target state typically consist of the target position and velocity in each direction plus possibly higher derivatives and other information, such as signal signature information or target features. For this discussion, the filter is assumed to be a Kalman filter or its mathematical equivalent (1,2,5). Usually, process noise can be used in the filter model to accommodate gradual target maneuvers. If a target can make abrupt maneuvers, then, a bank of Kalman filters might be used as with the interacting multiple model filters (5, 18), which accommodates switching from one model to another. A bank of Kalman filters can also be used for multiple model problems that do not exhibit switching, sometimes called static multiple models (19, 20). An example of the use of static multiple models is for tracking a single-stage booster that might be one of a number of different types of boosters.

Note that the Kalman filter equations are not very difficult to implement; it is the selection of the structure of the model and its parameter values used to design the filter that require extensive knowledge and experience. In addition, most target-sensor mathematical models are neither linear nor Gaussian, and thus some variant of an extended Kalman filter is typically used (2, 5). Nonlinearities can introduce biases in the estimation errors (6) and, in addition, unexpected results caused by the approximations used to deal with the nonlinearities are not uncommon.

For target tracking, the accuracy of both the target state estimate and its error variance-covariance matrix (or mean square error matrix) are important. For many filtering problems other than for tracking, the accuracy of the state estimate is more important than the consistency of the error covariance matrix. *Covariance consistency* in this context refers to how well the filter computed state estimation error eovariance matrix reflects the actual variance-covariance matrix of the state estimation errors. In addition, the state estimate is somewhat adaptive to model errors, but the error covariance matrix is not, which is because the computation of the state estimate is a function of the measurements that depend on the target location and motion. Consequently, even with model errors, the computed state estimate is influenced by the true target trajectory. In contrast, the covariance matrix depends on the mathematical model used for the filter design, and in a linear system, for example, the computed filter error covariance matrix is not a function of the measurements. More emphasis on covariance consistency is expected as the processing methods for single sensor and fusion tracking matures (30).

In most tracking systems, the data-association function performance depends on the consistency of the computed filter covariance marix. Hence, the consistency of the error covariance consistency is substantially more important in tracking than in many other types of filtering applications and should be evaluated during the algorithm development process. Note that because the track error covariance matrix indicates how accurate a track is, this information might be useful for the functions downstream of the tracker. The consistency of the error covariance matrix is degraded by not only the filter design model errors

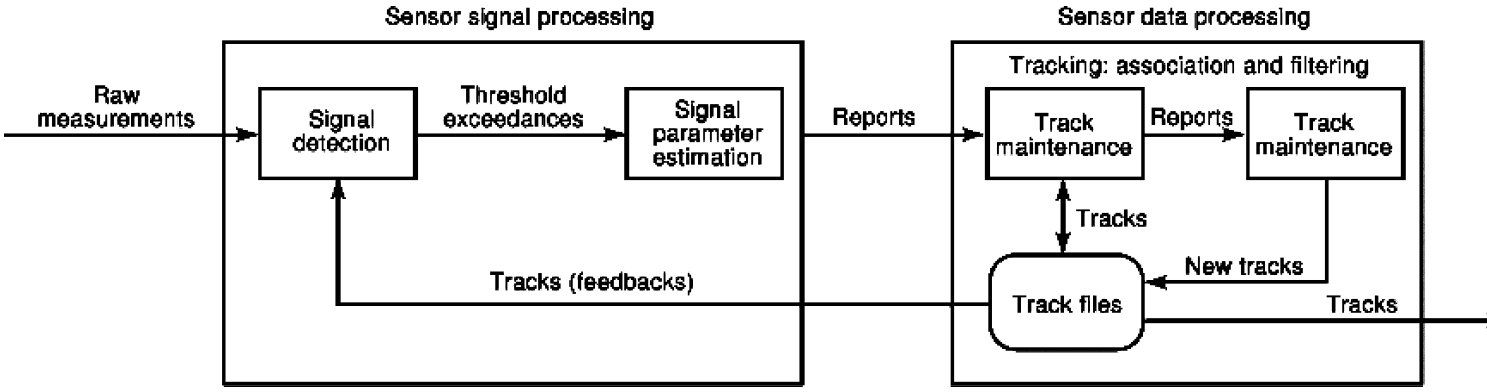

Figure 1. Block diagram of the major sensor processing functions of signal processing and data processing (9).

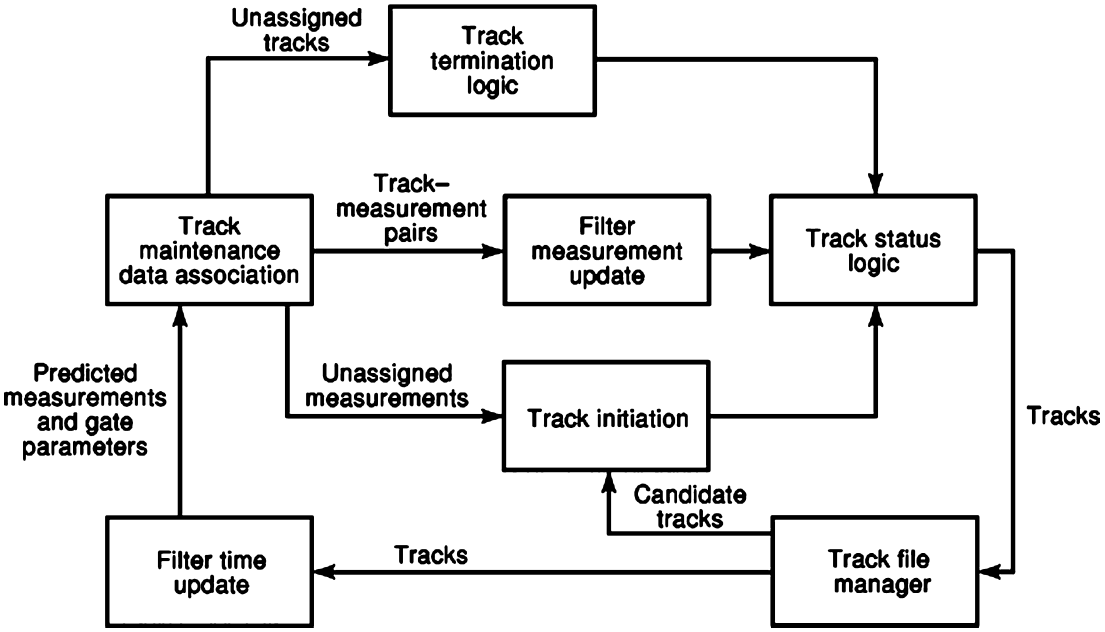

Figure 2. Block diagram of the processing functions for tracking isolated targets (17).

but also by misassociations and more so with some data-association algorithms than others.

A fundamental characteristic of small target tracking is that optimal tracking is not practical. The algorithms that would provide optimal tracking performance are too complex primarily because of the data-association function but also because most systems are neither linear nor Gaussian. For *optimal tracking* performance, each possible combination of all target tracks with all the measurements obtained up to the current time must be enumerated, and that is not practical. Consequently, a wide variety of suboptimal methods have been devised. **In algorithm development, the major trade is between tracking performance and the processor loading plus, if applicable, communications loading.** This is the major tradeoff, because improved performance can be obtained by more complex algorithms that are more hardware resource intensive.

Another fundamental characteristic of small target tracking is that it involves both discrete and continuous random variables or parameters. The measurement-track ambiguities introduce discrete random variables or hypotheses. Each *multiple-target hypothesis* represents one combination that accounts for all the apparent targets and all the measurements. The continuous random variables are the elements of the target state vectors. Most estimation theory deals with random variables from a continuous sample space and decision theory deals primarily with random variables from discrete sample space. The combination of continuous and discrete random variables can lead to unusual results compared with the more classic estimation problems. The hypotheses can cause the *a posteriori* state probability density function to be multimodal, which can lead to unexpected tracking performance and estimation errors that clearly do not exhibit a Gaussian distribution.

Because of the resulting complex nature of the estimation errors, multiple target-tracking performance evaluation and prediction are not very amenable to analysis. Therefore, usually performance is evaluated through Monte Carlo simulations followed by field-testing. Monte Carlo runs are needed because

tracker performance is data dependent because of misassociations and system nonlinearities. In addition, low-probability events might cause surprisingly poor performance that might not be uncovered with only one or a few Monte Carlo runs. The need for a simulation poses a dilemma because the performance evaluation results of simplistic simulation can be misleading and not reveal realistically all the phenomena or anomalies that will occur in the ultimate system. On the other hand, a more credible simulation of the targets, sensors, and signal-processing characteristic can be very costly, and the simulation can be complex and difficult to manage. Typically, the simulation used to design and develop a system evolves, starting from simple simulations for preliminary evaluation of well-selected critical functions to a complex high-fidelity simulation of the entire tracking system. The simulation environment needs to be flexible enough to easily accept major revisions to the tracker algorithms and the algorithms architecture, as well as new or modified sensor designs and targets.

Because optimal tracking methods are too complex to be practical, suboptimal (ad hoc) algorithms are typically devised that take advantage of the particular targets, sensors, and related conditions of the system for which the tracker is designed. Consequently, there is no universal tracker, although there might be a tracker that is developed for a particular type of scenario. Trackers are continually being developed to accommodate new requirements or target threat characteristics and to take advantage of increases in processor and communications capability as well as new or improved sensors.

Algorithm development of the trackers for a system is typically an experimental and iterative process. High-fidelity Monte Carlo simulations and hardware in the loop testing are needed to uncover undesirable performance that results from misassociations and unexpected operating conditions. Each stage of the algorithm development spiral typically includes a reevaluation of the requirements and performance obtained during the prior stage, algorithm redesign or replacement, performance evaluation testing, and study of the results. During algorithm development, it is not uncommon to have to replace some algorithms (or make major modifications to them) because of unexpected operating conditions and anomalous results. Thus, care is needed in undertaking software development of the deliverable code before the algorithm development process is complete.

In describing the various target-tracking methods, tracking with data from a single sensor is discussed before discussing tracking with multiple sensors, which is more complex. In addition, single target tracking is discussed before multiple target tracking because the tracking of multiple targets is substantially more complex than tracking a single target. The major emphasis in this article is on the data-association function, because that is the process that is unique to small target tracking relative to most other estimation tasks. The targets are assumed noncooperative in that, typically, they do not purposely communicate to the trackers their identity or location as is typical of air-traffic control. Discussion of track initiation is deferred to the section on multiple target tracking.

SINGLE TARGET TRACK MAINTENANCE WITH FALSE SIGNALS

The methods used to track a single target can be useful also for tracking multiple targets, provided they are far apart. Targets that are far apart are sometimes referred to as isolated targets (16, 17). *Isolated targets* are far enough apart so that all of the measurements in the immediate neighborhood of a target track have a very low probability of being caused by another target. In both single and multiple target tracking, a processing function is used to compute a track gate that establishes the measurements considered to be in the immediate neighborhood of a track. The processing functions for tracking isolated targets are shown in Fig. 2.

Track Gate

A track gate is also called a validation region or correlation window (1,2,6). For most tracking methods, the gate function simply serves to reduce the processing load and has little impact on performance if the track gate is big enough. To compute a simple yet practical gate, the filter function computes the expected location of the measurement from the target for a track and that location establishes the center of the gate. Then the extent of the track gate is established by computing the region around the predicted measurement that the measurement caused by the target is expected to be located with a prescribed probability (given that the target is detected). A practical value is 0.99 for the prescribed probability that the correct measurement is in the track gate region (17).

The size of the target gate region is computed using the innovations covariance matrix. The *innovations* vector is the difference between the measurement vector and the predicted measurement computed from the predicted target state vector. Thus, the innovations covariance matrix takes into account the error in the prior target state estimate, the prediction error, and the measurement errors. The innovations are sometimes called the residuals, which is not to be confused with the measurement residuals. The *measurement residual* vector is the difference between the measurement vector and the estimated measurement computed from the estimated target state vector after being updated using that measurement.

The use of two gates each with a different shape can help reduce the processing load. For two-dimensional measurements such as with a passive sensor, for example, the first gate is a rectangle that is sized to include the second gate, which is an ellipse. The rectangular gate is less computationally intensive but is not as effective. The rectangular gate eliminates most of the measurements that will not be in the elliptical gate. The elliptical gate requires more computations, but is more effective in that it produces fewer measurements in a gate for a given probability that the correct measurement is in the gate (1, 17). An elliptical gate involves the computation of the chi-square value as in Equation 2b of Table 3. In contrast, determining if a measurement is in a rectangular gate requires the computation of only a few comparisons, each proceeded by the computation of the absolute value of a difference. There are other

methods of computing a track gate, but for brevity, they are not discussed here. Note that typically in target tracking, the chi-square value does not exhibit the characteristics of a chi-square statistic because the random variable (innovations vector in this case) is seldom Gaussian because of estimation biases and misassociations, for example. Thus in tracking, the term Mahalanobis distance is moreappropriate to this computed variable than chi-square.

Data-Association Methods for Isolated Targets

For simplicity, consider single-target track maintenance, that is, assume that the target track has already been established. The tracking methods for multiple isolated targets are very similar. Track initiation, which is more computationally complex, is addressed in the section on multiple target tracking. The gate computation is the first function of the data-association process. Then, given the track gate, identify the measurements in the track gate by testing each measurement to determine first if it is in the rectangular gate and, if so, determine if it is in the elliptical gate. For a measurement vector that contains more than two elements, the rectangular and elliptical gates are replaced by their higher dimensioned counterparts. How the measurements that are in the track gate are subsequently processed depends on the chosen data-association processing method.

Data-association approaches can be classified as single-frame and multiple-frame methods (10,16,17). For single target tracking, a measurement in a track gate is either a false signal or the detected target. (Note that a measurement could be because of persistent clutter. An estimate of the location of persistent clutter should be maintained because the target might pass through its neighborhood. Accordingly, persistent clutter is equivalent to a target that is not of interest and hence is in the multiple target tracking category rather than single target tracking.) Two single-frame approaches are described for single target tracking, whereas discussions of the multiple-frame data association methods are deferred to the multiple target-tracking section.

A *single-frame association* approach typically enumerates most or all the possible hypothesis tracks for a frame of data. The hypotheses are generated based on a single prior track that remains after completing the processing for the prior frame of data. For single target tracking with M measurements in the track gate, there are $M + 1$ hypotheses, one for each measurement in the gate and one for the null hypothesis that the target was not detected in the gate for that frame of data. After the hypotheses are enumerated, the number of tracks is reduced to, at most, one track per apparent target for use with the next frame of data. Typically, the number of tracks of the current hypotheses is reduced to a single track by eliminating some, combining some, or both. The resulting single-state estimate will be referred to as the *composite state estimate*, and the composite state estimate and its coyariance matrix (plus possibly additional information) will be referred to as the *composite track* (11).

A single-frame data-association algorithm does not reprocess sensor data from prior frames, does not update the prior probabilities of the hypotheses, and carries forward in time at most one track per apparent target. As a consequence, multiple-frame data-association approaches, described later, typically perform better than single-frame approaches. To their advantage, single-frame data-association algorithms are not as complex or processing intensive as multiple-frame methods and do not require as extensive an algorithm development effort.

The two best known single-target, single-frame data-association approaches are the nearest neighbor and the probabilistic data-association filter. These approaches illustrate two very different types of decisions. The nearest neighbor approach makes *hard decisions*, that is, the association weight used for each measurement in a track gate is either zero or one. By contrast, the probabilistic data-association filter makes *soft decisions*, that is, the association weight used for each measurement in a track gate is usually between zero and one. With soft decisions, typically the sum of the weights for a track is one. The set of weights for a track include a weight for the possibility that none of the measurements in a gate are caused by the target.

Nearest Neighbor Tracking. The nearest neighbor (NN) algorithm is designed for tracking a single target, and only one track is carried forward for processing the next frame of data. This algorithm is sometimes referred to as the independent nearest neighbor (INN) algorithm to emphasize that each track is processed without regard to any other track. It *trims* (prunes) the hypotheses down to a single hypothesis by eliminating all but one hypothesis (I,2,6,16,17). For each frame of data, the INN algorithm assigns the (statistically) nearest measurement to the track. The statistical distance measure used is typically the same chi-square value that is computed for the elliptical (ellipsoidal or hyper-ellipsoidal) track gate. If the gate extent is sized appropriately, then an empty gate corresponds to the hypothesis that every measurement outside the gate is more probably a false signal than a detection of the target. Essentially, this is equivalent to finding the most probable hypothesis for each frame of data constrained by the decisions of the prior frames.

The INN algorithm is easily understood and implemented. It does not perform well, however, except with a low measurement density, such as up to an average of about 0.1 false signals in a 0.99 gate (1,2,16,17). (A 0.99 gate means that there is a 0.99 probability that the measurement caused by the target will be in the gate given that it is detected.) Note that the average number of false signals in a 0.99 track gate is a relative measure of density, not absolute, because it depends on the gate size that depends, in turn, on the accuracy of the track and measurements. With the INN algorithm, how dense the measurements can be and still provide adequate performance depends on the specific application and its characteristics, such as probability of detection, accuracy of the measurements, and number of elements in the measurement vector. The error covariance matrix for the resulting composite track is the covariance matrix of the most probable hypothesis track. That error covariance matrix of the composite track does not take into account the possibility that the most probable hypothesis track is the wrong track. In effect, this is equivalent to

neglecting the possibility that the selected hypothesis is not the correct one. Thus, the covariance consistency of the target track is degraded. The error covariance matrix for the INN track is optimistic, that is, the variance elements of the filter computed error covariance matrix are frequently smaller than the corresponding actual variances exhibited by the estimation errors.

Probabilistic Data-Association Filter. With the probabilistic data-association filter (PDAF) approach, all current hypotheses are computed and then *combined to* obtain a single track for the apparent target (1,2,6,16,17). Conceptually, the target state estimate and its error covariance matrix are computed for each hypothesis. Then the target state estimates of the hypotheses are combined into a single-composite target state estimate by computing their average using a weighted average. The weights are the probabilities of each of the hypotheses. The probabilities are a function of the same chi-square values that are used in track gating. Consequently, even with Gaussian random variables and a linear mathematical model for the measurements and target motion, the resulting composite state estimate is a non-linear function of the measurements, as is the variance-covariance matrix of its errors.

The error covariance matrix of the resulting composite state estimate is the sum over the hypotheses of the probability of each hypothesis times the error covariance matrix for the hypothesis plus the outer product of the difference between the hypothesis estimate and the composite state estimate. The equations for these computations are a special case of the ones discussed later in the multiple target-tracking section. However, If the measurement error covariance matrix is the same for all measurements and only one track is brought forward from the prior frame of data, the computations can be simplified (1,2,6). The composite track is then provided to the user, and it consists of the composite estimated state and its error covariance matrix for the apparent target. (The term *apparent target* is used because a track might not be following a target, but instead can be based on mostly, or exclusively, false signals.) This track is also used for processing the next frame of data, that is, the composite track is used instead of the hypotheses tracks as a basis to enumerate the hypotheses for the next frame of data. Accordingly, the number of hypotheses that must be enumerated for the next frame of data is greatly reduced.

Typically, the PDAF exhibits better tracking accuracy and fewer lost tracks than does an INN tracker if more than an occasional false signal in the track gate occurs (1–6). A *lost track* is one that was following a target but later was not following any single target. The mean-squared estimation errors are typically smaller for the PDAF than for the INN tracker because the weighted averaging using the association weights tends to "hedge the bets." A disadvantage of the PDAF tracker is that it is more processor intensive than the INN tracker and soft decisions might degrade features or attributes used for target classification or by the battle manager.

The PDAF-computed composite error covariance matrix is usually more realistic than that of the INN tracker and is typically consistent. The error covariance matrix of the PDAF composite estimate adapts to the sensor data because it depends on the number of measurements in the gate and how they are distributed. The variance elements of the error covariance matrix of the composite track usually will be small when there has been a sequence of frames with few false signals and large when there have been many false signals. In addition, the value of the variance elements in the covariance matrix will increase when the track gate is empty. Thus, both the PDAF composite state estimate and its covariance matrix are data dependent. The actual (true) error covariance matrix of a composite estimation problem is usually data dependent and the covariance matrix computed by the PDAF is also because it is an approximation to the actual error covariance matrix. Note that this is very different from a traditional linear, Gaussian estimation problem, for which a single Kalman filter can be used, and both the actual and computed state estimation error covariance matrix do not depend on the values of the measurements.

A Kalman filter (or the extended version) can usually be used for targets with deterministic or slightly random dynamics; such as a target with gradual maneuvers. For targets with substantially random dynamics (such as abrupt maneuvers), another filter may be needed. For abrupt maneuvers, the interacting multiple model algorithm (7, 18) might be adequate since it can accommodate multiple dynamic models and follow a target that switches abruptly from one dynamic maneuver to another.

In some tracking systems, single target-tracking approaches are used to track multiple targets. For example, the INN algorithm or the PDAF might be used to track multiple targets. As a result, each apparent target is tracked independent of the other apparent targets, that is, without the aid of information from tracks of the other apparent targets. Independent target tracking is justified with isolated targets. If some targets are closely spaced, substantially improved performance will be obtained by using a multiple target-tracking approach that coordinates the processing of the tracks by using the prior multiple track data more effectively rather than using an isolated target-tracking approach.

MULTIPLE TARGET TRACKING WITH DATA FROM A SINGLE SENSOR

A variety of single-frame and multiple-frame data-association methods have been devised for tracking multiple targets with data from a single sensor. In discussing some of these methods, it is assumed that false signals and closely spaced targets can occur. Furthermore, it is assumed that persistent clutter points are treated as targets; however, to simplify the discussion, the assumption is that no UCSOs occur. When discussing hypotheses for multiple target tracking, the term *hypothesis* refers to a multiple-target hypothesis; that is, each hypothesis accounts for all target tracks and all measurements in the applicable sensor data. The initial emphasis of this section is on track maintenance and then track initiation is discussed.

In multiple target tracking, a frame of data will normally contain measurements from many targets. Most

tracking algorithms assume that the signal processor provides measurements in a sequence of proper frames of data. *A proper frame of data* is a collection of measurements wherein no two (or more) measurements are from the same target. Performance is expected to degrade if the frames of data are smaller than is practical or are hot proper frames.

Before discussing some of these suboptimal tracking methods, it is instructive to first discuss optimal tracking. It is useful to discuss optimal tracking for at least two reasons. First, the equations of optimal tracking are also used in suboptimal tracking but in a different way. Second, optimal tracking displays important properties that are helpful in understanding the characteristics of practical, suboptimal multiple target-tracking methods and in designing those methods.

Optimal Tracking of Multiple Targets

There is no single method for optimal tracking because different optimization criteria lead to different optimal tracking algorithms even for a linear, Gaussian problem (15). This characteristic of target tracking is very different from the more traditional linear, Gaussian estimation problem, for which the Kalman filter is optimal for most optimization criteria. Other complexities unique to the multiple target estimation task also exist that muddy the issue of what is meant by the optimal estimate, but that issue need not be explored here (15). To limit this discussion, a simple scenario is addressed and with but two optimization criteria.

Suppose that at time zero the tracking system receives ahandoff of tracks from another system that has (somehow) established a set of tracks for all the targets that includes consistent error covariance matrices for all the target tracks. Furthermore, the handoff estimation errors for each target exhibit a Gaussian probability distribution and are not cross-correlated from target to target. The system has a linear sensor with measurement errors that exhibit a Gaussian probability distribution. Furthermore, the target motion is described by linear vector state equation and, if process (state) noise is applicable, it exhibits a Gaussian probability distribution. Note that this is not only a "nice" linear, Gaussian problem, but the number of targets is known, which greatly simplifies the problem.

In optimal tracking, all hypotheses and all their tracks must be retained for use in processing the subsequent frames of data, A bank of Kalman filters can be used to compute the state estimates for each target track for each hypothesis. Fortunately, a target track based on a specific sequence of measurements is used in more than one hypothesis so that some economy of processing is obtained by taking advantage of that fact. Equations for computing the probability for each hypothesis are given in Table 1. These equations apply to optimal multiple target tracking for most optimization criteria. The notation used here is consistent with typical Kalman filter notation, except that the estimates are also conditioned on the hypothesis, as can be seen from Eq. (Id). The optimization criterion determines how the estimates of the hypotheses are processed to establish the single best track for each target. For the minimum mean-square error (MMSE) criterion, the equations for the optimal composite estimate are given in Table 2. Table 3 amplifies on the equations used to compute the probability of the innovations used to compute the probability of each hypothesis. All the current hypotheses are retained and used as a basis for computing the hypotheses' tracks when the next frame of data becomes available. In contrast, the composite tracks are recomputed after every frame of data becomes available, based on the tracks of all the current hypotheses and their probabilities.

Note that the equations in Table 1 permit the targets' state vectors to be handled in two different ways. If any of the random variables related to one target are cross-correlated with those of another target, then state vectors of all the targets are concatenated in to a single "system state vector," which consists of all the state vectors for all the targets. The equations of Table 2 are treated this way. Note from Eq. (2d) of Table 2 that the individual target tracks of the composite estimate are cross-correlated because of the last term, that is, the outer product in that equation. The second method for handling the target states applies if no target-to-target cross-correlation exists or can be neglected [see Eq. (lw)].

To illustrate that two different optimization criteria lead to different optimal multiple target-tracking algorithms, Table 4 provides the optimal composite estimate for the joint maximum a posteriori probability (JMAP) criterion (11,15–17). Note that Eq. (2d) in Table 4 shows that any estimate that is not the same as optimal MMSE estimate will have a larger actual error covariance matrix (11,16,17). Also, note from that equation that any suboptimal hypothesis estimate will exhibit cross-correlations between the individual target hypothesis tracks because of the outer product term.

The optimal single track (in the minimum mean square sense) for an apparent target at any one time is a composite track that is the appropriately weighted sum of tracks for that apparent target contained in all the multiple-target hypotheses. To illustrate the magnitude of the complexity for optimal tracking without missed signals or false signals, N_T targets and N_F frames of data would require the enumeration of $(\boldsymbol{N_T}!)^{N_{F-1}}$ hypotheses (assuming no tracks based on prior data are available)

It is the retention of all the hypotheses and all their tracks that makes optimal tracking impractical. Clearly, optimal tracking is a multiple-frame data-association approach with the number of frames in the data association equal to number of frames of data available. The so-called "gated optimal" tracking is optimal tracking except that gates are used to eliminate unlikely track-measurement pairs (16, 17). The gating process reduces processing complexity, but, because it is a trimming process, the results are suboptimal.

Single-Frame Data-Association for Track Maintenance

Single-frame data-association approaches for multiple target-track maintenance include the global nearest neighbor algorithm and joint probabilistic data association.

Global Nearest Neighbor Tracking. The global nearest neighbor (GNN) tracker uses a single-frame data-association algorithm that makes hard decisions. It is an

Table 1. Block 2 Optimal Multiple Target Estimation Equations (16, 17)

Block 2	
Compute	(1a)
	(1b)
$\omega_{k_n}(n)\alpha\ p[k_n, z(n)\|Z(n-1)]$	(1c)
$\hat{x}_{k_n}(n) = E[x(n)\|k_m Z(n)]$ = hypothesis estimate	(1d)
$P_{k_n}(n)$ = hypothesis estimate covariance	(1e)
$v_{k_n}(n)$ = hypothesis innovations	(1f)
$S_{k_n}(n)$ = hypothesis innovations covariance	(1g)
k_n = hypothesis index	(1h)
$\beta_F = P_{FP}/A_P$	(1i)
β_F = false signal density	(1j)
	(1k)
	(1l)
P_D = probability of target signal detection	(1m)
P_G = probability that target is in gate	(1n)
n_F = number of false signals in gate	(1o)
$\beta_{NT} = P_{NT}/A_P$	(1p)
β_{NT} = new target density	(1q)
	(1r)
	(1s)
n_D = number of targets detected	(1t)
n_T = number of targets	(1u)
n_{NT} = number of new targets	(1v)
If the random variables are independent from target to target, that is, the system innovations covariance matrix is target, block diagonal, then:	
$P[v_{k_n}(n)] = \Pi_i\ p[v_{ij}(n)]$	(1w)
where:	
i = index of target tracks	(1x)
j = index of measurements, a function of i and k_n	(1y)
$n_{ij}(n)$ = innovations vector for track i and measurement j	(1z)

Table 2. Block 1 Multiple Target Equations for optimal MMSE Estimation (15-17)

Block 1	
	(2a)
	(2b)
	(2c)
	(2d)
where	
$\omega_{k_n}(n) = p[k_n\|Z(n)]$ = hypothesis probability	(3a)
$\hat{x}(n) = E[x(n)\|Z(n)]$ = composite estimate	(3b)
$P(n)$ = composite estimate covariance	(3c)

Table 3. Hypothesis Innovations Probability Equations for Optimal Multiple-Target Estimation (15-17)

For linear, Gaussian conditions:	
	(1)
	(2a)
	(2b)
	(3)
$S = H(n)P(n\|n-1)H^{\mathrm{T}}(n) + R(n)$	(4)
where	
[T$_{\mathrm{E}}$Xnical Error]	(5)
$\hat{v}(n) = z(n) - H(n)\hat{n}(n)$	(6)

extension of the INN tracker for use with multiple targets. There is a number of different implementation approaches to GNN tracking. One version of the GNN finds the most probable (multiple-target) hypothesis for each frame of data constrained by the decisions of the prior frames. This version will be referred to as (multiple-target) single-frame most probable hypothesis (SF-MPH) tracking. Only one track per apparent target is carried forward for processing the next frame of data. Rather than enumerate all the hypotheses, typically an *optimal, unique, 2-D assignment algorithm* is used to find the most probable hypothesis, and that greatly reduces the amount of computations (1). The assignment algorithm assigns measurements to tracks. The term *unique* in this context means that no track is assigned to more than one measurement and no measurement is assigned to more than one track.

Table 4. Block 1 Multiple-Target Equations for Optimal JMAP Estimation (15-17)

Block 1. Maximum joint a posteriori estimate (JMAP estimate)	
	(2a)
	(2b)
$\hat{x}_{\text{JMAP}}(n) = \hat{x}_{k_n}(n)$	(2c)
	(2d)
where	
$\hat{k}_n$ = JMAP hypothesis decision	(3a)
$\hat{x}_{\text{JMAP}}(n)$ = JMAP estimate	(3b)
	(3c)
$\hat{x}_{\text{MS}}(n)$ = MMSE estimate	(3d)
	(3e)

Table 5. Qualitative Comparison of Fusion Algorithm Architectures (12)

	Report Responsibility	Fusion without Track Feedback	Track Fusion with Feedback	Measurement Fusion	Hybrid Fusion
Track accuracy, false/missed tracks	1	2	3.5*	3.5*	5
• Increase effective sampling rate	1	2.5	2.5	5	4
• Utilize diversity-geometrical/accuracy	1	3.5	3.5	3.5	3.5
Extend detection range	3	3	3	3	3
Extend field of view (FOV)	3	3	3	3	3
Communication load	5	4	2	1	3
Inaccuracy and misassociations due to residual registration biases	3	4	4	1	3
Need changes to sensor processor	5	2.5	2.5	1	4
For single-platform tracking	1	2	3.5*	3.5*	5
For multiple-platform tracking	2	3	4	1	5

* Key: 5 (or Largest Number) Best.

Note: In ordering, values adjusted so that the sum of each row is 15.

The 2-D qualifier refers to two dimensions because there are two data sets that are involved, namely, measurements and tracks, and therefore the assignment cost array is a matrix. Note that although finding the optimal (minimum cost) solution to a two-dimensional assignment problem is tractable, it turns out that a higher dimensioned assignment problem is not. Also note that an optimal unique assignment algorithm does not provide optimal tracking.

In the past, suboptimal assignment algorithms were used to further reduce the amount of computations. However, there is little advantage to using a suboptimal assignment algorithm because now very fast optimal 2-D assignment algorithms are available. These algorithms are fast because they take advantage of the sparseness of the cost matrix. The sparseness occurs because not every measurement is in every track gate. A unique assignment algorithm is able to find the most probable hypothesis because of the basic structure of the equation for the probability of a hypothesis, provided the target-track-to-target-track cross-correlations are neglected.

Figure 3 displays a block diagram of the data-association functions for use of a 2-D assignment algorithm. The gate search function determines which measurements are in the rectangular track gates (or its higher dimensioned version). A simplistic algorithm should not be used for this function if many targets occur in any one region. For example, If two loops were used (one for measurements and one for tracks) for the gate search function, then 100 targets and 100 measurements would require 10,000 evaluations to determine which measurements are in each track gate. More ingenuous methods will greatly reduce this number. The likelihood function serves to perform the elliptical (or hyper-ellipsoidal) gate process and also to compute the cost value for use in the assignment matrix. The chi-square calculation of Eq. (2) in Table 3 is often used for the cost in the assignment matrix (1,16,17). *Singletons are* simply obvious measurement-track pairs for which there is no contention and thus need not be included in the assignment algorithm (16, 17), If many targets occur, then there may be an advantage to partitioning the tracks into what are called track clusters may exist. Tracks are partitioned so that no measurement in the gate of a track in one cluster is also in the gate of a track that is in another cluster. Track clusters should not be confused with target clusters, which are groups of targets whose state vectors are approximately the same. The purpose of partitioning tracks is to reduce the processing load but will not necessarily reduce processing if a state-of-the-art assignment algorithm is used.

The last function in Fig. 3 is the assignment algorithm, which uniquely assigns measurements to tracks. This block diagram is applicable with modification to other data-association approaches. The unassigned measurements are normally forwarded to the track-initiation function, and the unassigned tracks are tested for possible termination (see Fig. 2). the advantage of the GNN approach is that it does take into account multiple targets by using the multiple tracks and all the measurements in a frame of data (or partition). In addition, it is relatively easy to implement, compared with other data-association methods and is not

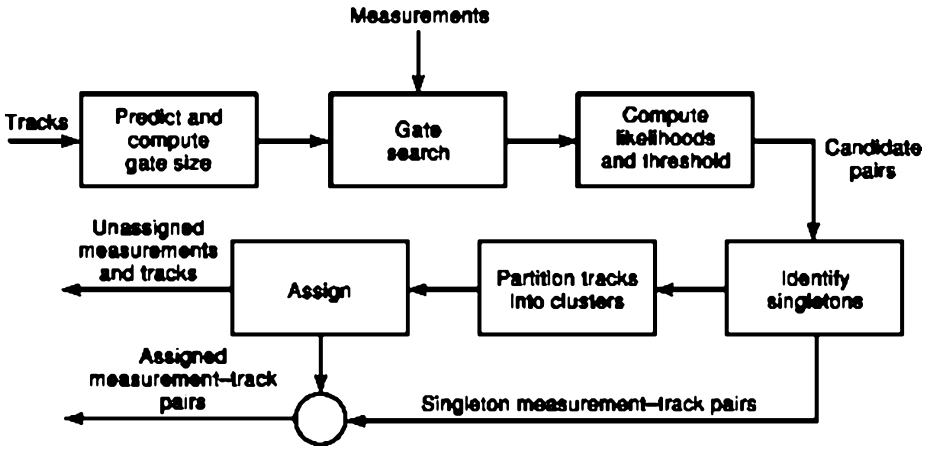

Figure 3. Typical block diagram of functions for single-frame data association (16, 17).

very processing intensive. In addition, this approach tends to adjust for shifts in the data from frame to frame because of interframe jitter. The disadvantage is that false signals arid new targets tend to degrade performance. This might be an appropriate tracking approach for implementation early in the algorithm development stage for a new system and then followed later by a more advanced and complex tracking approach, if needed, when the systemcharacteristics are better understood and established.

Joint Probabilistic Data Association. The joint probabilistic data-association (JPDA) tracker uses a single-frame data-association approach that makes soft decisions. It is an extension of the PDAF tracker for use with multiple targets (1,2,6). In the JPDA approach, all current hypotheses are computed and then combined to obtain a single composite track for each apparent target. These composite tracks are used to provide to the user the estimated state and its covariance matrix for each apparent target. In addition, for processing the next frame of data, the composite tracks are used instead of the hypothesis tracks. Accordingly, the number of hypotheses that must be enumerated for the next frame of data is greatly reduced compared with optimal tracking. The equations of Table 1 are applicable to JPDA, but normally the target-track-to-target-track cross-correlations are neglected. A JPDA tracker is expected to perform better than the GNN tracker as the number of false signals or new targets increases. The JPDA is more processing intensive and requires more complex computer programming and algorithm development effort than the GNN approach. In addition, the soft decisions of the JPDA tracker might degrade features or attributes used for target classification or by the battle manager. Fortunately, the composite error covariance matrix consistency of a JPDA tracker is usually substantially better than that of the GNN tracker.

A unique feature of the JPDA approach is that it permits the computation of the probability that a specific measurement is caused by the same target that a specific track is following. Similarly, the probability that a measurement is a false signal or caused by a new target can also be computed. These probabilities can be used for a variety of purposes beyond simply computing the composite state estimate and the error covariance matrix for each track. These probabilities are sometimes referred to as marginal association probabilities, and the details for computing them depend in part on a number of assumptions, such as the mathematical model used for the false signals (1,2,6).

Multiple-Frame Data Association for Track Maintenance

Multiple-frame data-association approaches include the algorithm by Singer, Sea, and Housewright, multiple-hypothesis tracking, Poore's tracker, and other algorithms (10, 11). Typically multiple-frame data-association approaches enumerate (and also carry forward to the next frame) more candidate tracks than do single-frame approaches. Consequently, multiple-frame data-association approaches typically provide better estimation accuracy and fewer false and missed tracks than single-frame approaches. However, multiple-frame data-association algorithms are more complex and processing intensive, and they require substantially more effort in design, development, and test.

Singer, Sea, and Housewright Tracker. The Singer, Sea, and Housewright (SSH) tracker was designed for tracking a single target with false signals (21), but has been extended to multiple target tracking. It is sometimes referred to as the "n-scan back" approach (2) and is similar to the JPDA approach, except that it maintains more than one track per target for processing the next frame of data. The algorithm uses local combining to reduce the number of tracks. Tracks are selected to be combined if they use the same measurements in the latest N frames of data and the same target track $N + 1$ scans back. The number of frames, N, used in the combining process is a design parameter. Although the SSH algorithm can be classified as a multiple-frame, probabilistic data-association approach, it uses local rather than global combining and uses measurements as the criteria for forming local composite tracks. Because it usually retains more than one candidate track per target from frame to frame, it should perform better than JPDA but increases the processor load and requires more extensive algorithm and software development.

Multiple-Hypothesis Tracking. Multiple-hypothesis tracking (MHT) typically carries more than one hypothesis track per apparent target forward for processing the next frame of data. Many different versions of MHT have been developed since its original conception (22). In MHT, for practical reasons the number of hypotheses is limited by both eliminating and combining some hypotheses and tracks (1,7,22). In the original MHT, the typical combining (merging) process is local rather than global. Given four hypotheses' tracks for a single apparent target, for example, two similar hypotheses tracks might be combined (merged) to form one hypothesis track, a "local" composite. As a result, the four hypotheses tracks would be reduced to three. Then one of these three, the one with the smallest hypothesis probability, might be eliminated so that only two tracks would be forwarded for processing with the next frame of data for that apparent target. In order to compute the needed probabilities, all the current hypotheses are enumerated and the (multiple target) hypotheses' probabilities computed. The computations for these probabilities are similar to those in Table l.

MHT should perform better than the other tracking approaches just discussed. Improved performance is obtained at the expense of an increase in processing load, computer programming, and algorithm-development effort. Many organizations either have developed or are developing MHT or similar trackers and some are on their second-or third-generation (incarnation) multiple-frame tracker. Some of the more recent MHT approaches use a sliding window of multiple frames of data,which is similar to the method discussed in Section 4.3.3. In MHT, however, there is additional pruning of unlikely candidate tracks and combining of similar candidate tracks for a target to reduce the number of multiple target hypotheses that must be enumerated.

Multiple-Frame Most Probable Hypothesis Tracker. Poore's tracker is similar to MHT but does not use any local combining or trimming; it uses deferred global trimming.His tracker employs a sliding window of $M-1$ frames of data (23). The window also includes tracks based on data up to and including M frames back, that is, the tracks are based on all data except the latest $M-1$ frames of data. The tracker then uses an M-D assignment algorithm to seek the most probable hypothesis for the M-1 frames of data given the tracks M frames back. This is a multiple-frame most probable hypothesis (MF-MPH) tracker. The M frames back tracks are then updated just one frame of data using the measurements in frame M–1 back that are paired with those tracks in the most probable hypothesis.

It is not practical for most systems, however, to find the optimal solution to the M-D assignment algorithm with M greater than 2. To circumvent this problem, search for the optimal solution in Poore's M-D assignment algorithm is stopped when the current solution is close enough to the optimal assignment solution relative to the uncertainty caused by the random variables. His assignment algorithm is able to determine bounds on how close the current solution is to the optimal solution. This tracker makes a firm decision on the measurements in $M-1$ frames back and tentative decisions on all subsequent measurements so that the current target state estimates can be computed for all apparent targets. *A firm decision* is an irreversible decision and a *tentative decision* is one that may be revisited and changed at a later time. After this processing is completed, the window is moved forward one frame of data and the process repeated.

Related Comments. There are also other multiple-frame data-association algorithms that have been devised, and some employ *retrodicted probabilities* (10, 11), which are "smooth" decisions that are analogous to smoothing of continuous random variables in Kalman filtering. (Note that in Kalman filtering, it seems that it might be more appropriate to refer to a "smoothed" estimate as a *retrodicted estimate*, i.e., an estimate of the state for a specific time given subsequent measurements.) More complex track processing can be expected in the future as the processing capabilities of computers continue to improve and thus permit tracking to approach optimal performance more closely.

Note that the hypotheses' probabilities of all the suboptimal tracking approaches discussed above, that is, except for optimal tracking, are not truly probabilities but "pseudo-probabilities." A pseudo-probability is an approximation to the probability that the measurements assigned to the tracks for a hypothesis are the correct ones. The pseudo-probabilities are approximations because all previous hypotheses have not been maintained and used in the computations. The pseudo-probabilities are usually computed as if the deleted hypotheses were not possible and that no loss of information results from combining hypotheses.

Some target-tracking approaches partition the processing into the three major stages of (1) track initiation (formation), (2) track maintenance (extension or continuation), and (3) track termination, as in Fig. 2, Each track is started in the track-initiation processing and then continued in track-maintenance processing until terminated by the track-termination logic. The optimal, MHT, and MF-MPH approaches typically integrate all three phases in. the process of enumerating hypotheses and generating tracks (1,3,4). In contrast, the INN, PDAF, JPDA, GNN, and SSH approaches are typically track-maintenance approaches that must be augmented by separate track-initiation and track-termination processing functions (1,4,17). The algorithm for the track-termination function might be as simple as to terminate tracks that are not updated L frames in a row, or possibly not updated L frames out of J frames, where L and J are selected using Markov chain analysis. Markov chain analysis can be used to trade off the number of tracks incorrectly terminated versus the number of false tracks that are not terminated soon enough. This type of analysis can be used also to establish parameters for track initiation (1).

Track Initiation

Typically, a sequence of more than two measurements is needed to initiate a track. Fortunately, tracks do not have to be initiated very often. For tracking approaches that do not integrate the track initiation and maintenance processing, measurements not used by track maintenance are usually

forwarded for use by the track-initiation function. With a very sparse population of measurements, it may be sufficient to initiate tracks by using the same INN algorithm that was described for track maintenance. The first measurement used to start a new track is called an *initiator*. An initiator starts a candidate initial track that is updated using the INN algorithm as appropriate measurements are provided to the track-initiation function. A score based on chi-square values can be updated as a candidate track is updated. When the score exceeds a prescribed threshold, the candidate initial track is promoted to a mature track and processed by the track-maintenance function thereafter.

Note that in track initiation, not enough information exists to compute the first gate using only one measurement. After an initiator is identified, the first gate (and possibly more) is computed using *a priori* information on the velocity (and possibly higher derivatives) because the data of the initiator does not include complete velocity information, if any.

If more than just a few measurements exist in a region that are forwarded to the track-initiation function, there can be contention for measurements by a nuimber of different candidate initial tracks. One approach that addresses this issue is to use binary linear programming or an optimal, unique M-D assignment algorithm to resolve the contentions and find all the appropriate sequences of measurements for promotion to mature tracks (24). If this is too processing intensive, then it might be sufficient to use a unique suboptimal M-D assignment algorithm, such as the so-called greedy algorithm. A variety of other methods have been developed for track initiation. Track initiation is complex because usually more than just a few frames of data are needed to initiate tracks with reasonable confidence.

MULTIPLE-SENSOR (FUSION) ALGORITHM ARCHITECTURES

There are many different ways that data from multiple sensors can be combined. The differences between the various multiple sensor approaches may not be important with respect to performance for tracking with a sparse population of measurements. With challenging conditions of a moderate to dense population of measurements the difference between the various tracking approaches can have a significant impact on both performance and required hardware capacity. In designing an algorithm architecture for multiple-sensor tracking, ultimately, the major considerations are typically cost, communication load, processor load, survivability, and performance. Performance considerations typically include estimation accuracy, number of false tracks, number of missed tracks, number of missed tracks, covariance matrix consistency, and robustness. There are virtually an infinite number of possible processing and data distribution methods for multiple target tracking with multiple sensors,The understanding of the fusion options is simplified if the considerations are divided into "how" the processing is done and then "where" the processing components are located. One view of the different types of fusion algorithm architectures limited to "how" the processing without regard to "where" the processing is located is summarized in the following section and then compared.

Alternative Fusion Algorithm Architectures

Four pure generic types of algorithm architectures for track maintenance and for track initiation have been identified. This classification of algorithm architectures is based primarily on how the association processing is performed over time and over the ensemble of sensors (10,12,17). The four types of track maintenance algorithm architectures are as follows:

- *Type I:* Independent sensor algorithm architecture
- *Type II:* Track fusion algorithm architecture
- *Type III:* Composite-measurement fusion algorithm architecture
- *Type IV:* Measurement fusion algorithm architecture

In the *independent sensor algorithm architecture* (Type I), the tracks are processed for each sensor without use of the data from the other sensors. Frame-to-frame data association and filtering are performed without any sensor-to-sensor processing, feach user obtains tracks based on a single sensor. Note that each measurement is subjected to only one association process, but single-sensor tracks need to be retained in track files for each sensor. In addition, there is no improvement in the track quality because of the existence of multiple-sensor data.

In the *track fusion algorithm architecture* (Type II), tracks are first processed for each sensor without use of data from the other sensors. Sensor-to-sensor processing follows single-sensor frame-to-frame association and filtering. Single-sensor tracks are fused using track-to-track association followed by filtering to form multiple-sensor (global) tracks. Note that each measurement is subjected to two association processes. Multiple-sensor tracks as well as single-sensor tracks for each sensor are retained in track files. This process is sometimes called hierarchical or distributed algorithm architecture and is complicated by the property that typically sensor-level tracks are cross-correlated with the global-level tracks. Feedback of the multiple-sensor global tracks to the single-sensor track processing can be employed. The vanilla architecture without feedback to the lower levels is designated Type IIa. Feedback to the lower levels usually improves the track accuracy at both that level and the higher levels, and that architecture is designated Type IIb. In systems where there are multiple sensors on each platform and each platform is at a different location, it is common to have three processing levels: (1) sensor-level tracking, (2) platform-level tracking, and (3) global-level tracking.

There are a number of methods for dealing with the track-to-track error cross-correlation in track fusion. In some methods, the track data are distributed in the form of a tracklet *A tracklet* is defined as a track computed so that its errors are not cross-correlated with any other data distributed in the system for the same target (12). Tracklets can be computed by decorrelating the sensor tracks (25) or

formed from a sequence of measurements (12). The term track fusion is used here to refer to a system that distribute tracks or a system that dbtributes tracklets from the local track processor to the fusion processor. One of the major benefits of track fusion compared with the other types of fusion is that the communications load can be greatly reduced by not distributing the track data after every measurement is obtained for a target. The *tracktet interval*, the time between when tracklets are distributed by a sensor for a target, can often be from 5 to 30 measurement sampling periods, depending on the application. Thus, data compression is obtained with little loss of information provided the target dynamics are deterministic. The original tracklet methods were designed for non-maneuvering targets, and those methods might not provide adequate performance if the targets are maneuvering, because with maneuvers those tracklet methods do not provide lossless information (26). If the possibility of misassociations exist at the local or fusion level, then a number of considerations need to be addressed in deciding whether to distribute target tracks or tracklets (27).

In the *composite-measurement fusion algorithm architecture* (Type III), multiple-sensor processing of the measurements from all sensors is first employed. The processing of measurements consists of associating measurements from one frame of data from all sensors and computing an improved estimate of a projection of the state vector for each target, such as estimated position. Note that normally for accurate fusion with this approach, either the sensors must obtain measurements at the same time or the targets and sensors must be moving slowly relative to the frame period. These composite measurements are then used in frame-to-frame association and filtering. Serisor-to-sensor processing precedes frame-to-frame processing. Note that each measurement is subjected to two association processes, but only one set of multiple-sensor tracks need be retained in track files.

In the *measurement fusion algorithm architecture* (Type IV), measurement-to-track association is followed by filtering using the prior multiple-sensor tracks. This architecture is sometimes referred to as central-level fusion (1, 7). In its simpler form, the data-association processing uses the multiple-sensor tracks and one frame of data from a sensor; the tracks are updated and then a frame of data from another sensor along with the updated multiple-sensor tracks are processed. Note that each measurement is subjected to only one association process and only one set of multiple-sensor tracks need be retained in track files.

In addition to the pure generic methods for track maintenance is one more type of fusion approach that is not a pure approach, namely, a hybrid approach. One devised hybrid approach is flexible and adaptive because it permits the distribution of tracklets, composite measurements, or measurements for each apparent target, depending on the needs of the system for data on that target at the current time (12).

Report responsibility is a multiple-sensor, multiple target-tracking algorithm architecture that is popular in the radar community. This approach might be viewed as a special case of the Type II, track fusion algorithm architecture but it is not fusion. That is, data from more than one sensor is not combined to form a multiple-sensor track. While it does not fusion data, it may produce tracks containing a sequence of Segments of tracks for which each segment uses data from a different sensor than the prior segment. Thus, report responsibility could be considered in a class by itself. It is discussed here for completeness in preparation for a qualitative comparison of algorithm architectures.

In report responsibility, each sensor tracker is responsible for providing the tracks for a subset of all the targets. The intent is for one and only one sensor tracker to broadcast a track for a target. The sensor tracker that provides the besttrack for a target is responsible for broadcasting the track for that target on the network to the users and all the other sensor trackers; no other sensor tracker is supposed to broadcast a track for that target. Consequently, the issue of track cross-correlation does not apply to this approach. A number of approaches on how to coordinate the decisions to achieve the intent of report responsibility exist. Depending on how report responsibility is coordinated, transients can exist with more than one sensor tracker broadcasting a track for a target. Some advantages of report responsibility include very low communications rate and use of the best sensor track (or one of the better sensor tracks for a target). In report responsibility, however, since the sensor tracks for a target from multiple sensors are *not* combined, fall advantage is not taken of the capabilities of fusion and so-called "geographic diversity." Typically, each sensor is more accurate in one direction than the other(s) so that combining data from distributed sensors can decrease the standard deviation of the estimation errors by substantially more then the square root of the number sensors.

Comparison of Fusion Algorithm Architectures

It would be very desirable to be able to compare algorithm architectures rigorously. The state of the art of target tracking and target typing is such that apparently no one can afford the cost of a comprehensive comparison of algorithm architectures or of the algorithms for each of the tracking functions. Performance is data dependent and requires simulations for evaluation. Comparing the fault tolerance of the various possible system designs is certainly not easy, and a comparison of the hardware required for alternative system designs can be extremely complex.

In lieu of an extensive quantitative comparison, a subjective qualitative comparison of some algorithm architectures has been made (12). This comparison, shown in Table 5, is only an initial effort, assumes no process noise, and compares only five algorithm architectures. It must be stressed that this is a qualitative comparison in that a rating of 4 might be substantially better than a rating of 3 or only slightly better than a rating of 3 if measured quantitatively. In addition, this comparison does not explicitly include all the dimensions or trade issues listed in the beginning of this section. The comparison of Table 5 is only intended to indicate some of the critical issues inselecting an algorithm architecture. This table is more applicable to track maintenance than to track initiation because the properties of some sensor combinations require special

consideration for track initiation. For example, two sensors may not have much information in common with only a few frames of data, such as an active sensor with relatively inaccurate angle data and an accurate passive sensor with no range data.

A big influence in Table 5 is the relative location of the sensors. With alt sensors at one location (on one platform), the communications load is not an issue and measurement fusion might be preferred. Communication between distant participating units is a major consideration and, so with distributed platforms, track fusion might be preferred in order to reduce the communications load. In addition, sensor location and orientation biases plus sensor measurement biases are extremely important in the fusion of multiple-sensor data and tyjpicaliy must be addressed. The residual biases appear to cause more misassociations with measurement fusion than with track fusion.

The asterisks in Table 5 indicate that for best tracking accuracy, the selection of the best algorithm architecture depends heavily on how different the participating sensor characteristics are, the size of the residual biases, and the types of targets. For example, for best tracking accuracy, very similar sensors may make measurement fusion preferred whereas track fusion may be preferred for disparate sensors. Two benefits of measurement fusion is its data timeliness, which is critical for highly maneuverable targets and the aspect that each measurement goes through one association process. However, the number of misassociations exhibited by track fusion and measurement fusion can be very different. With diverse sensors and a very different number of targets observed by each sensor, measurement fusion might introduce many more misassociations than would track fusion.

Another consideration in the selection of a fusion approach is the impact on the existing hardware. Some sensor processors provide only sensor tracks and do not provide measurements. If a measurement fusion approach were chosen, then the processors would have to be changed, which could be expensive. In addition, some existing trackers do hot provide the track error covariance matrices. The error covariance matrices are not needed for some approaches to report responsibility, but are required to compute the tracklet if the tracks are to be decorrelated for track fusion, and the expense of this hardware change should be considered. Hybrid fusion that distributes tracklets or measurement data exhibits the best characteristics of both measurement fusion and track fusion because the choice of what is distributed can be based on the needs at any one time.

Discussion of Fusion Systems

A clear distinction should be made between the functional (logical) algorithm architecture (discussed in Section 5.2) and the physical distribution of the processing. With multiple platforms and onboard processing, each function of an algorithm architecture can be physically distributed in many ways over the sensor platforms and a centralized processing station, if applicable. In addition, each of the generic algorithm architectures can be implemented in many ways.

An important example of a specific combination of both the algorithm architecture and the physical distribution of the processing is what could be called measurement fusion with distributed data association (distributed measurement fusion). Consider distributed sensor platforms with a fusion processor on each platform. In addition, there might be a user of fused tracks on each (or most) sensor platform plus possibly platforms with users and fusion processors but no sensors. For track maintenance, each platform is responsible for the assignment of its measurements to the fusion (network) tracks, and then each measurement is tagged with its assigned fusion track number. Each measurement with its fusion track tag is distributed to all the other platforms. When a platform receives a tagged measurement from another platform, the data-association function can be bypassed and the measurement is sent to the filter function for use to update the track with the track number as indicated by the measurement's tag. The track initiation function assigns new track numbers to new tracks, and processing is needed to attempt to identify and eliminate redundant tracks from being proliferated. In order for the distributed users to coordinate their actions, all platforms need to exhibit the same information (including fusion track number) for each target. This property is sometimes called *single integrated air picture* (SIAP). The distributed measurement fusion approach is designed to exhibit SIAP. With centralized measurement fusion and distributed users, a number of methods have been devised to achieve SIAP, for example, the centralized fusion tracks could be distributed to all user platforms but that would require an increase in communications capacity.

There are also four track-initiation architectures that are conceptually the same as the track-maintenance architectures summarized above. The type of track-initiation architecture need not be the same as the selected type of track maintenance. Note that for a number of fusion approaches, no simple obvious approach exists for upgrading from single-frame data association to multiple-frame data association as exists for tracking with data from a single sensor. One exception to this challenge is centralized measurement fusion at a single ground station because it is very similar to processing with: data from a single sensor.

In some multiple sensor systems, data is available in addition to simple kinematic measurements. The additional data might be features and attributes that are useful in target classification and combat identification or target typing and discrimination, depending on the type of target. A distinction is made between features and attributes because they are each processed differently.

Features are measurement data useful in target classification whose random components are from continuous sample space. Features such as target size, radar cross section, and signal strength might be processed much the same way that target location is processed. Attributes are measurement data useful in target classification that are drawn from discrete sample space. Attributes such as number of engines of an aircraft are processed very differently compared to kinematic information and can be processed using discrete probabilities and likelihoods. Attributes and features could be processed after the kinematic data association is complete for a frame of data or could be included

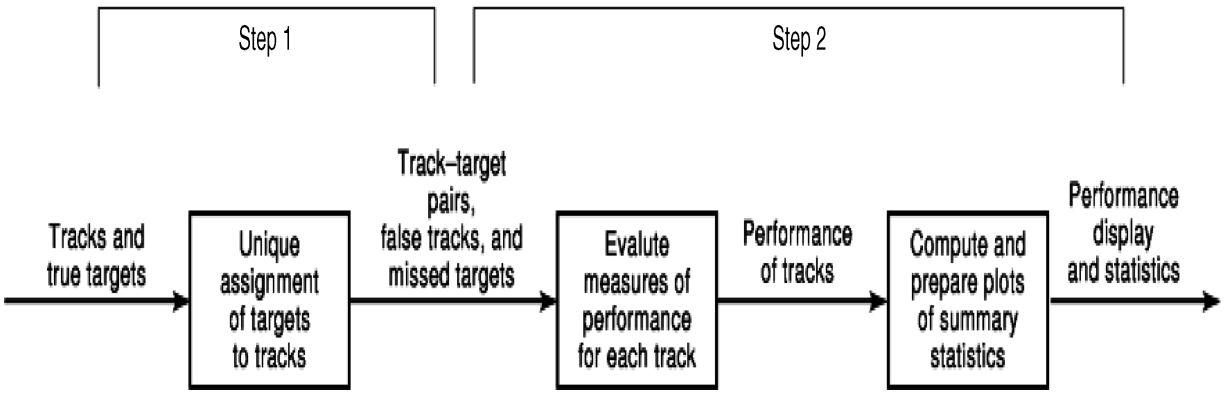

Figure 4. Diagram of a two-stage performance-evaluation methodology (17).

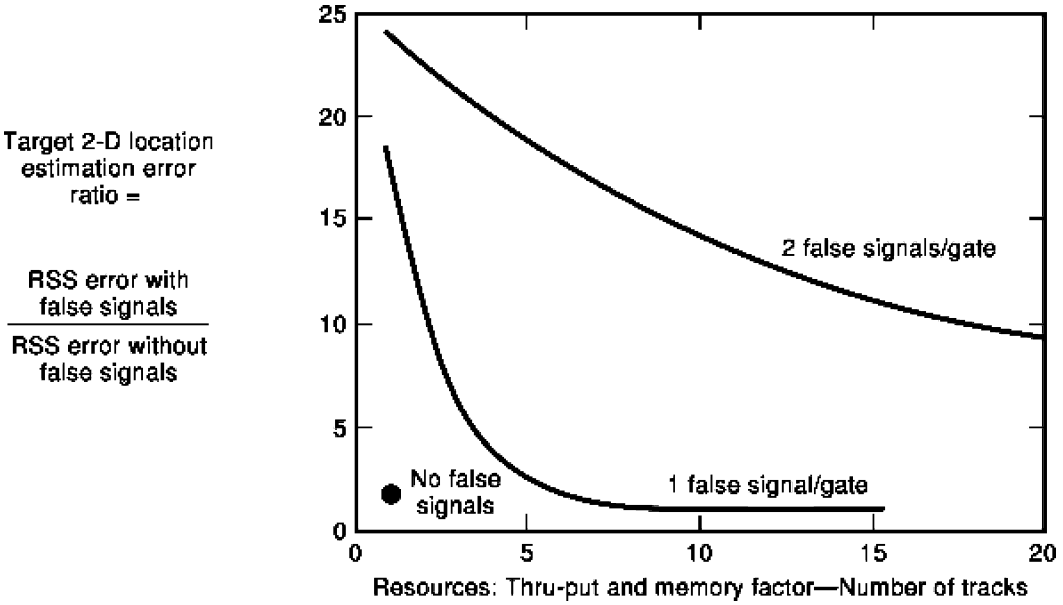

Figure 5. Illustration of major trade-off parameters for single-target tracking (14).

in the data association processing. The later approach is normally referred to as feature aided tracking (31).

PERFORMANCE EVALUATION

Ultimately, the performance of tracking algorithms is judged by the success of the system that they support. Evaluation of tracking performance serves as an intermediate measure of system effectiveness, to diagnose the algorithms, and to predict performance for use in system studies. However, ambiguities can occur in evaluating performance because of misassociations (13). Misassociations can cause missed targets and false tracks such as redundant, spurious, switched, and lost tracks. As a result, it may not be clear which target a track is following, if any. Measures of performance cannot be evaluated with the aid of a simulation (or through field tests) without first designating which target each track is following; There are a number of evaluation methodologies that have been proposed to address this problem (28). Care is needed not to use a methodology that gives unfair advantage to one tracking approach over another.

One methodology for resolving these ambiguities is to use an assignment algorithm to uniquely assign the tracks to targets (13, 15). The use of the statistical distances between targets and tracks for the cost elements in the assignment matrix tends to treat the alternative tracking algorithms fairly. Then the tracking errors and other measures of performance can be computed given these unique track-target assignments. This two-stage methodology is shown in Fig. 4. Some of the common measures of performance include the root mean sum square of the error biases, the position errors and the velocity errors; covariance consistency; the number of misassociations; track purity and duration; average time to initiate tracks; and the number of missed, switched, and false tracks (29). If the system involves multiple platforms, then performance metrics may also be needed to determine if all platforms exhibit the same information about the threat and friendly forces (SIAP). For a tracking application, no single critical performance metric exists that can be used to evalu-

ate one or more trackers. For one reason, a collection of metrics is needed because usually the tracker parameters could be adjusted to favor one metric at the expense of others.

Both tracking performance and required hardware capacity should be evaluated. As mentioned earlier, choices of the algorithm architecture, algorithms and locations for each function, and the algorithm parameters will impact both performance and required processor capacity (and communications load, if applicable). An example of this tradeoff between performance and required hardware resources is shown in Fig. 5. This figure summarizes results of the simulation of tracking a single target with data (that included false signals) from a single passive sensor. The results are shown after seven frames of data have been processed. The tracking algorithm was similar to a single-target version of Poore's tracker. The number of frames in the sliding window was varied from 1 to 6 so that the curve in the figure was obtained. The values for the horizontal and vertical axes have been normalized by dividing by the corresponding value that is exhibited by tracking without false signals. Note that the results for the INN algorithm are at the left end of each of the two curves. This figure illustrates the major tradeoff between performance and required processor capacity (processing time or required memory) for tracking with a single sensor.

BIBLIOGRAPHY

1. S. S. Blackman, *Multiple Target Tracking With Radar Applications*, Denham, MA: Artech House, 1986.
2. Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*, San Diego, CA: Academic Press, 1987.
3. Y. Bar-Shalom, ed., *Multitarget-MuHisensor Tracking: Advanced Applications*, Norwood, MA: Artech House, 1990.
4. Y. Bar-Shalom, ed., *Multitarget-Multisensor Tracking: Applications and Advances*, Vol.**2**, Norwood, MA: Artech House, 1992.
5. Y. Bar-Shalom and X. R. Li, *Estimation and Tracking: Principles, Techniques and Software*, Boston, MA: Artech House, 1993.
6. Y. Bar-Shalom and X. R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, Los Angeles, CA: OPAMP Tech. Books, 1995.
7. S. S. Blackman and R. F. Popoli, *Design and Analysis of Modern Tracking Systems*, Norwood, MA: Artech House, 1999.
8. O. E. Drummond, ed., *Signal and Data Processing of Small Targets 1997, Proc. SPIE*, **3163**; (1997).
9. O. E. Drummond, ed.,Introduction, *Signal and Data Processing of Small Targets 1997, Proc. SPIE*, **3163**: ix (1997).
10. O. E. Drummond, Multiple sensor tracking with multiple frame, probabilistic data association, Signal and Data Processing of Small Targets 1995, *Proc. SPIE*, **2561**: 322–336 (1995).
11. O. E. Drummond, Target tracking with retrodicted discrete probabilities, Signal and Data Processing of Small Targets 1997, *Proc. SPIE*, **3163**: 249–268 (1997).
12. O. E. Drummond, A hybrid sensor fusion algorithm architecture and tracklets, Signal and Data Processing of Small Targets; 1997, *Proc. SPIE*, **3163**: 485–502 (1997).
13. O. E. Drummond, and B. E. Fridling, Ambiguities in evaluating performance of multiple target tracking algorithms, Signal and Data Processing of Small Targets 1992, *Proc. SPIE*, **1096**: 326–337 (1992).
14. O. E. Drummond and S. S. Blackman, Challenges of developing algorithms for multiple sensor, multiple target tracking, Signal and Data Processing of Small Targets 1989, *Proc. SPIE*, **1096**: 244–256 (1989).
15. O. E. Drummond, *Multiple-object Estimation*, Ph.D dissertation, Univ. of California at Los Angeles, Los Angeles, CA, 1975, Xerox Univ. Microfilms No. 75–26, 954.
16. O. E. Drummond, *Multiple Sensor, Multiple Target Tracking, SPIE Short Course SC56, April 1998 and earlier versions, SPIE*, Bellingham, WA.
17. O. E. Drummond, *Multiple Target Tracking Lecture Notes*, Los Angeles, CA: Technical Book Company, 1998.
18. H. A. P. Blom and Y. Bar-Shalom, The interacting multiple model algorithm for systems with Markovian switching coefficients, *IEEE Trans. Autom. Control*, **33**: 780–783 (1988).
19. D. T. Magill, Optimal adaptive estimation of sampled stochastic processes, *IEEE Trans. Autom. Control*, **10**: 434–439 (1965).
20. F. L. Sims and D. G. Lainiotis, Recursive algorithm for the calculation of the adaptive Kalman filter weighting coefficients, *IEEE Trans. Autom. Control*, **14**: 215–217 (1969).
21. R. A. Singer, R. G. Sea, and K. Housewright, Derivation and evaluation of improved tracking filters for use in dense multitarget environments, *IEEE Trans. Inf. Theory*, **20**: 423–432 (1974).
22. D. B. Reid, An algorithm for tracking multiple targets, *IEEE Trans. Autom. Control*, **24**: 843–854 (1979).
23. A. B. Poore and N. Rijavec, Multiarget and multidimensional assignment problems, Signal and Data Processing of Small Targets 1991, *Proc. SPIE*, **1481** 345–356 (1991).
24. C. L. Morefield, Application of 0–1 integer programming to multi-target tracking problems, *IEEE Trans. Autom. Control*, **22**: 302–312 (1977).
25. G. Frenkel, *Multisensor tracking of ballistic targets, Signal and Data Processing of Small Targets 1995, Proc, SPIE*, **3561**: 337–346 (1995).
26. O. E. Drummond, et al., Performance Assessment and Comparison of Various Tracklet Methods for Maneuvering Targets, Signal Processing, Sensor Fusion, and Target Recognition XII, *Proc. SPIE*, **5096**: 514–539 (2003).
27. O. E. Drummond, Track and Tracklet Fusion Filtering, Signal and Data Processing of Small Targets 2002, *Proc. SPIE*, **4728**: 176–195 (2002).
28. O. E, Drummond, Methodologies for Performance Evaluation of Multitarget Multisensor, Signal and Data Processing of Small Targets 1999, *Proc. SPIE*, **3809**: 355–369 (1999).
29. R. L. Rothrock and O. E. Drummond, Performance Metrics for Multiple-Sensor, Multiple-Target Tracking, Signal and Data Processing of Small Targets 2000, *Proc. SPIE*, **4048**: 521–531 (2000).

30. O. E. Drummond, "Tracking and Classification with Attribute Data from Legacy Sensors," *Workshop on Multiple Sensor Target Tracking, A Tribute to Oliver E. Drummond Key West, FL, GTRI*, Atlanta, GA, (2004).

31. O. E. Drummond, et al., "On Target Track Covariance Consistency," Signal and Data Processing of Small Targets 2006, *Proc. SPIE* Vol.**6236**, Paper 623615 (2006).

OLIVER E. DRUMMOND
Consulting Engineer,
CyberRnD, Inc.
Culver City, CA