Distributed Fusion Architectures and Algorithms for Target Tracking

MARTIN E. LIGGINS II, MEMBER, IEEE, CHEE-YEE CHONG, MEMBER, IEEE,
IVAN KADAR, SENIOR MEMBER, IEEE, MARK G. ALFORD, MEMBER, IEEE,
VINCENT VANNICOLA, MEMBER, IEEE, AND STELIOS THOMOPOULOS, SENIOR MEMBER, IEEE

Invited Paper

Modern surveillance systems often utilize multiple physically distributed sensors of different types to provide complementary and overlapping coverage on targets. In order to generate target tracks and estimates, the sensor data need to be fused. While a centralized processing approach is theoretically optimal, there are significant advantages in distributing the fusion operations over multiple processing nodes. This paper discusses architectures for distributed fusion, whereby each node processes the data from its own set of sensors and communicates with other nodes to improve on the estimates. The information graph is introduced as a way of modeling information flow in distributed fusion systems and for developing algorithms. Fusion for target tracking involves two main operations: estimation and association. Distributed estimation algorithms based on the information graph are presented for arbitrary fusion architectures and related to linear and nonlinear distributed estimation results. The distributed data association problem is discussed in terms of track-to-track association likelihoods. Distributed versions of two popular tracking approaches (joint probabilistic data association and multiple hypothesis tracking) are then presented, and examples of applications are given.

I. INTRODUCTION

In recent years there has been increasing emphasis on using multiple data sources for detection, tracking, classification, situation assessment, etc. The synergistic use of overlapping and complementary data sources provides information that is otherwise not available from individual sources. For example, radar provides accurate range but poor ange data while infrared provides accurate angle but poor range data. Even for similar sensor types, the different viewing angles from multiple physically distributed sensors can be exploited to provide better location data. Furthermore, multiple sensors provide more robust performance due to the inherent redundancy.

Fusion is necessary to integrate the data from the different sensors and extract the relevant information on the targets. The traditional architecture for fusion is centralized. Data from multiple sensors are sent to a single location where the data are fused and the results distributed to various users. Recent advances in computing and communication have made other architecture options feasible. These include hierarchical architectures where the fusion nodes are arranged in a hierarchy with the lowest level nodes processing sensor data and sending results to higher level nodes to be combined. When the higher level node collects processing results only periodically, significant savings in communication can be achieved.

In a fully distributed architecture, there is no fixed superior/subordinate relationship. Each node can communicate with other nodes subject to connectivity constraints. Communication can be adaptive and dependent on the information content and needs of the individual nodes. In an extreme case, each sensor may have its own processor to fuse the local data and cooperate with other sensor nodes.

The centralized architecture is theoretically optimal if the communication bandwidth is high enough to send the sensor data to the fusion node, which has enough computer resources to process the data. It is also conceptually simpler. On the other hand, a distributed (including hierarchical) fusion architecture has the following advantages: lighter processing load at each fusion node due to the distribution over multiple nodes; no need to maintain a large centralized database since each node has its own local database; lower communication load since data does not have to be sent to/from a central processing site; faster user access to fusion results since there is less communication delay; and

Manuscript received May 8, 1996; revised October 18, 1996.

M. E. Liggins, M. G. Alford, and V. Vannicola are with the Rome Laboratory, Surveillance and Photonics Directorate, Rome, NY 13441-4514 USA (e-mail: liggins@rl.af.mil).

C.-Y. Chong is with the Booz, Allen and Hamilton, Inc., Sunnyvale, CA 94089 USA (chong_chee-yee@bah.com).

I. Kadar is with the Northrop Grumman Corporation, Advanced Technology, and Development Center, Bethpage, NY 11714–3582 USA (ivan_kadar@atdc.northgrum.com).

S. Thomopoulos is with the INTELNET Inc., State College, PA 16803 USA (e-mail: scat@intelnet.intelgate.com).

Publisher Item Identifier S 0018-9219(97)00773-1.

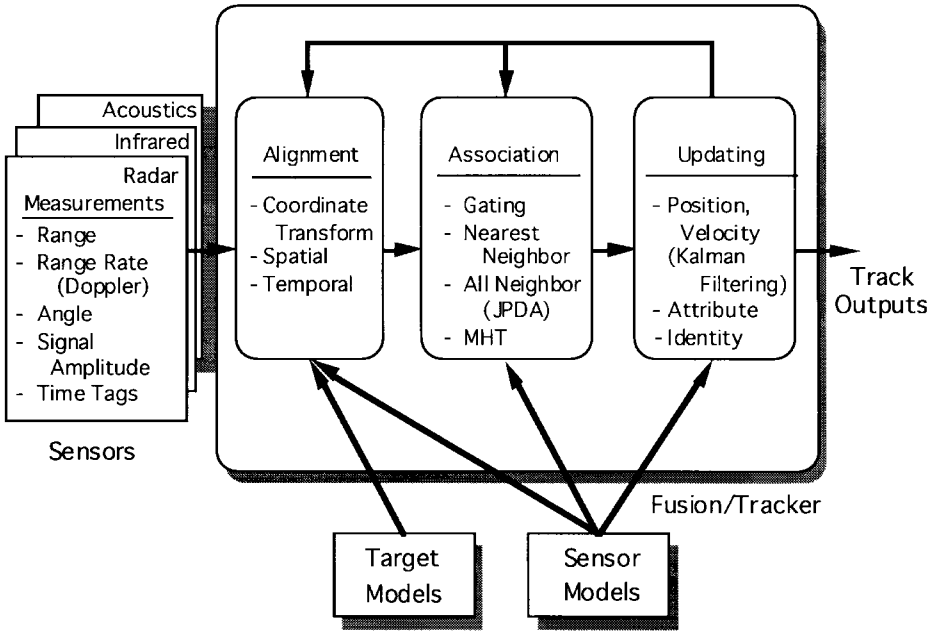

Fig. 1. Functional architectures for tracking/fusion.

higher survivability since there is no single point of failure associated with a central fusion node. The distributed fusion architecture is also a necessity since many fusion systems have to be built with existing (legacy) fusion systems as components.

While there are many advantages to distributed fusion, there are also technical issues on system architecture and algorithms that need to be addressed before high performance systems can be developed. Some important technical issues include the following items.

- *Architecture*: how the nodes should share the fusion responsibility, e.g., which sources or sensors should report to each node, and the targets that each node should be responsible for.
- *Communication*: how the nodes should communicate, e.g., connectivity and bandwidth of the communication network, information push or pull, and communicating raw data versus processing results.
- *Algorithms*: how the nodes should fuse data for high performance results and select their communication actions (who, when, what, and how).

Compared to centralized fusion on which much research has been performed [1], [2], distributed fusion is a much less mature area of research. The goal of this paper is to review the current status of research, and present an approach for analyzing distributed architectures and developing distributed fusion algorithms.

There are two main types of fusion problems: fusion for making a decision on a hypothesis such as detecting the presence of targets or classifying a signal, and fusion for target tracking. The distributed version of the first type of problems is usually known as distributed decision/detection fusion [3], [4], and will not be the subject of this paper. Instead, we will focus on the tracking problem [5]–[8] and how it can be distributed.

Since the main components of tracking are estimation and data association, distributed fusion for tracking will involve distributed estimation and distributed association. For distributed estimation, most research has assumed a hierarchical fusion architecture. Most results on distributed linear Gaussian estimation start with the various forms of Kalman filters [9] to derive the distributed equations [10]–[23]. Distributed nonlinear results generally follow the same philosophy [24], [25]. The information graph approach for modeling the information flow in arbitrary distributed fusion architectures and developing fusion equations was introduced in [26]–[29]. The linear and nonlinear fusion equations for hierarchical architectures can also be derived using this approach.

For the distributed data association problem, early work dealt with the track to track association problem [30], [31]. Since then distributed versions of centralized algorithms such as joint probabilistic data association (JPDA) [32], [33], or multiple hypothesis tracking (MHT) [34], [35], have been developed [36], [27]–[29].

The structure of this paper is as follows. In Section II we first review the key functional components in a fusion system used for tracking. Section III presents different fusion architectures, the information graph approach to represent the information flow, and what we mean by optimal distributed fusion. A methodology for developing distributed estimation algorithms is then discussed in Section IV and related to standard linear and nonlinear results. The effects of dynamic states and process noise are also considered. Section V discusses the distributed version of the data association problem. Some examples of applications are given in Section VI.

II. Fusion for Target Tracking

The sensor/data fusion problem [1], [2] is present in many different applications including surveillance, robotics,

manufacturing automation, etc. In each application area, even though the general goal is to utilize the available data to improve the understanding of some state of the world, the specific nature of the problem may be quite different. For example, parts inspection in manufacturing automation generally involves a known number of cooperative static objects in a controlled environment, while fusion in surveillance has to deal with an unknown number of noncooperative dynamic objects in an adverse environment. In this paper, we focus on the problem of determining the location, velocity, type, and other attributes of multiple moving objects from sensor data. Such problems, sometimes called tracking, are prevalent in surveillance applications such as defense or air traffic control, and also occur in robotics applications.

A common characteristic in fusion for tracking is that the sensors do not provide information on the origins of the measurements. For example, a radar scan provides multiple detections of targets and clutter but the detections are not identified with the targets. Thus a key function in fusion is the association of the measurements to the targets before any estimates can be made from the measurements. Fig. 1 shows an architecture for tracking/fusion which is adapted from [37]. The architecture consists of the following functions.

A. Alignment

Data alignment is the first step in the fusion process. Typically data alignment deals with the spatial registration or temporal prediction of the target tracks based on the inputs of the sensor suite. Converting the data of each sensor to a common coordinate system is necessary to fuse target data from dissimilar sensors. Although conversion provides a convenient frame of reference, it may create biases when compensating for measurement error that needs correcting [8]. Temporal alignment extrapolates the tracks to the same time as the measurements so that they can be compared with each other. It first links one or more predetermined target kinematic models to the track; then the temporal prediction portion of the Kalman filter is used to estimate the location or state of the target at the time of the next scheduled sensor report.

B. Association

Association is responsible for partitioning the measurements into sets that could have originated from the same targets. In a recursive processing algorithm, this involves associating measurements to the set of aligned tracks. Validation gates are first computed from the extrapolated track and sensor errors and used to reduce the number of possible associations to a track. Then association is performed using one of a number of possible approaches.

The *nearest neighbor* approach assumes that the measurement closest to the center of the validation gate represents the target. All other detections in the gate are then labeled as false alarms. Although this simple approach requires minimal computational resources, it may not work well in difficult situations with high clutter, poor detection or dense targets.

In contrast to the simple nearest neighbor approach, the JPDA algorithm [32], [33] makes the assumption that the nearest detection from the extrapolated target estimate may not always originate from the target and that other detections which are farther away may be the real measurement. To account for this, the JPDA algorithm assigns a weight to each detection within the target validation gate to represent its likelihood of being associated with the target. All the detections are then used to update the target state estimate, the contribution of each detection being proportional to their association weights. The probabilistic association results from the fact that all detections are used with weights that depend on their association likelihoods. This technique represents an all neighbor approach. One advantage of the JPDA approach is that it readily allows for the assignment of a single sensor report to multiple tracks, thus reducing the difficulty of track conflicts during track crossings. Furthermore, memory and computational requirements are usually kept fixed as compared to other more complicated approaches. The primary disadvantage is that excessive clutter can pull the tracker away from the true target position.

The MHT approach [34], [35] delays making a firm association decision when the situation is unclear. Generally, the hypotheses are formed at two levels—track and scene. Track-level hypotheses are formed from possible associations of current detections to previous tracks, and their likelihoods are computed by comparing the extrapolated track state estimates with the detections. Each scene level hypothesis is a consistent set of track hypotheses (e.g., no two tracks share the same measurements when a measurement can only result from a single target). Scene level hypotheses from the same set of detections are mutually exclusive and represent alternate ways of associating the data. Scene level hypotheses are also scored using track likelihoods and probabilities of previous hypotheses. Even with pruning and other hypothesis management techniques, the MHT approach requires more computational resources but can produce better results in more complex situations.

The different approaches are related. In fact, the JPDA algorithm can be viewed as a special case of the MHT by combining all the hypotheses after each update. The specific choice of techniques depends on the scenario such as target and sensor parameters, the desired performance and the computation resources available. In other words, application dictates the specific choice of the association approach.

C. Updating

Given a particular association, the estimates of the states for each track are updated with the sensor measurements. When the measurement models are linear or can be approximated by linearized models, e.g., measurements of position and velocity, the Kalman filter measurement update equations [9] are used to combine the associated measurements with the predicted states. For attributes that are nonlinear in

nature, e.g., target type, Bayes rule or other approaches such as Dempster–Shafer may be used. The updated information is then used again to support future alignment, gating and association when new measurements are received.

III. Distributed Fusion Architectures and Approach

Many existing fusion systems have a centralized architecture with all data processed by a single fusion node. The availability of distributed computing and the need to deal with bigger problems, however, will imply distributed fusion systems with multiple fusion nodes processing data from their own sensors and communicating with other nodes to improve upon the local results.

The presence of multiple data sources and fusion nodes provides many choices in the architecture, i.e., how the sensors or data sources report to each fusion node and the connectivity among the nodes. Fig. 2 shows four possible architectures: centralized, hierarchical without feedback, hierarchical with feedback, and fully distributed. In a centralized architecture, there is a single track database and the data from multiple sensors are aligned and associated with the tracks in this database to update the tracks and their state estimates. In the hierarchical architectures, the fusion nodes are arranged in a hierarchy with the higher level nodes processing results from the lower level nodes and possibly providing some feedback. The hierarchical architecture is natural for many applications, e.g., a fusion node for radar data, another node for infrared sensor data, and a node that combines the radar and infrared tracks. Considerable savings in communication can be achieved if data is communicated to the higher levels at a rate slower than the sensor observation rate. Hierarchical architecture with feedback is equivalent to periodic broadcast when the feedback goes to every node. In a fully distributed architecture, there is no pre-determined superior/subordinate relationship, each node can communicate with any other node subject to connectivity constraints, and the communication can be asynchronous.

For architectures that are not centralized, a key problem is how to combine the results from two fusion nodes. Standard centralized techniques such as those described in Section II can no longer be used since they usually assume that the errors in the data to be fused are independent. This assumption is usually violated in distributed fusion. In fact, the *rumor propagation* or *chicken little phenomenon* arises when information arriving by multiple paths has to be fused. For example, A may tell both B and C that he saw a target with medium confidence. When D gets this information from both B and C, he may not recognize the redundant information and increase the confidence to high because he gets the same conclusions from both B and C. This phenomenon can produce biased and erroneous results at the fusion nodes.

A. Information Graph

A model of the information flow among the components in a fusion system is needed to analyze and design distributed fusion systems and algorithms. The information graph has been proposed [28], [29] as a way to represent the information events in a distributed fusion system and their interactions. Using the information graph the common information shared by arbitrary estimates can be identified so that any redundant information can be removed in the fusion process. This is especially useful when the communication structure among the fusion nodes is complicated since the identification of common information may not be easy.

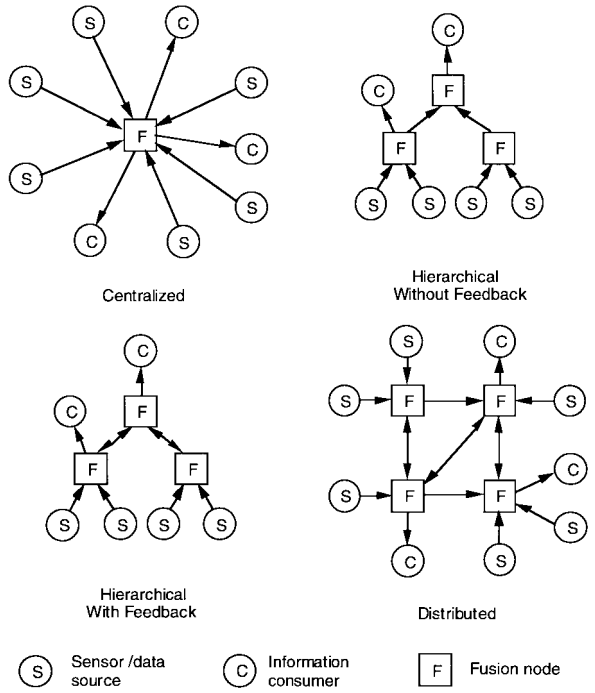

Fig. 2. Fusion architectures.

The information graph contains four types of nodes to represent the following information events:

- sensor observation nodes I_{ST};
- sensor data reception nodes I_{SR};
- communication transmission nodes I_{CT};
- communication reception nodes (fusion nodes) I_{CR}.

The directed links in the graph represent communication. A directed path from *Node A* to *Node B* means that there is communication from *Node A* to *Node B*. Each node has access to the information of the predecessor nodes and the maximum information at a node consists of all predecessor observation nodes. A node that is a common predecessor to two nodes contains the common information of those two nodes. Thus the information graph can be used to find the common information of any two or more nodes by identifying their common predecessors.

Fig. 3–6 show four examples of fusion architectures and their information graphs. In each example, time goes from left to right. It is obvious that the multiple information path problem does not exist for centralized fusion (Fig. 3). For hierarchical fusion, without or with feedback (Fig. 4 and 5), multiple information paths exist, but identifying the redundant information from the information graph is

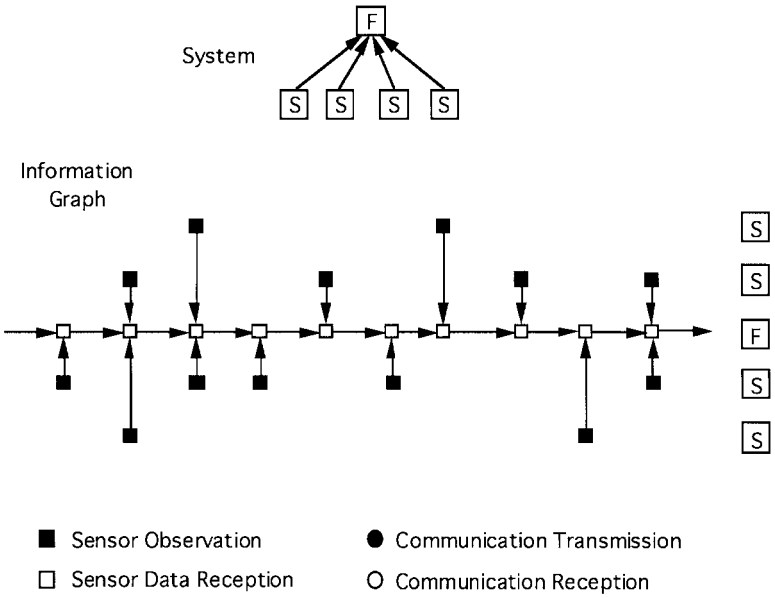

Fig. 3. Information graph: centralized fusion.

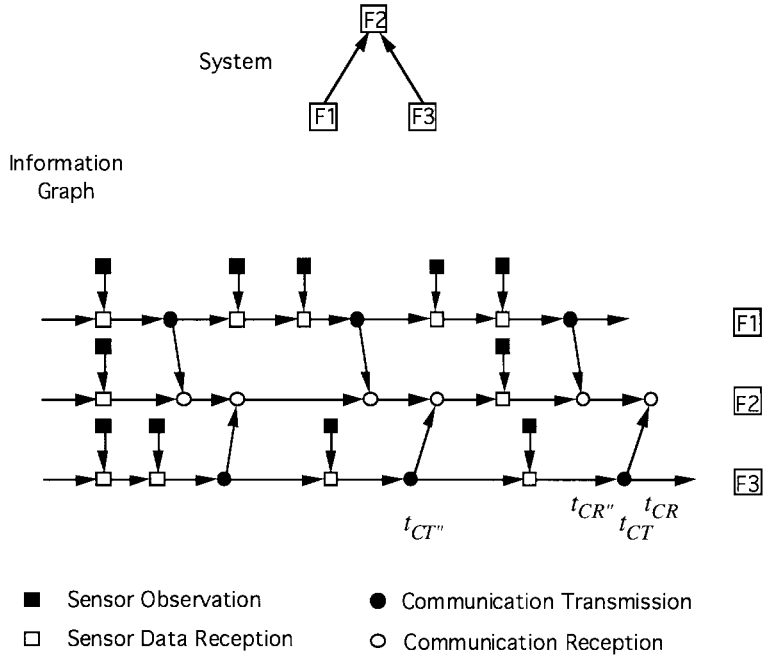

Fig. 4. Information graph: Hierarchical fusion without feedback.

relatively straightforward. At each fusion node, one only needs to determine the single common ancestor for the nodes whose information is to be fused. For hierarchical without feedback, this ancestor is at the lower level of the hierarchy, i.e., F1 or F3. For hierarchical with feedback, the ancestor is at the higher level of the hierarchy, i.e., F2.

For cyclic communication, each fusion node collects data from its own sensor, forms the local estimate and then communicates it to another node in a cyclical manner. The receiving node then fuses the incoming estimate with the local estimate. Fig. 6 shows that determining the common information can be quite complicated. The information graph can be used recursively to determine the common source of information for any two nodes. In particular, the information at the fusion node A fuses the estimates at nodes B and C, whose common information consists of that at the common predecessor nodes D and E. The common information of D and E consists of that at the nodes F and G, which is equivalent to the node H.

B. Distributed Fusion Approach

The distributed fusion algorithm attempts to duplicate the results of a centralized algorithm as if the actual sensor data were communicated instead of the outputs of the tracker/correlators. Fig. 7 illustrates the philosophy of the approach. On the right side of this figure, Nodes 1 and 2

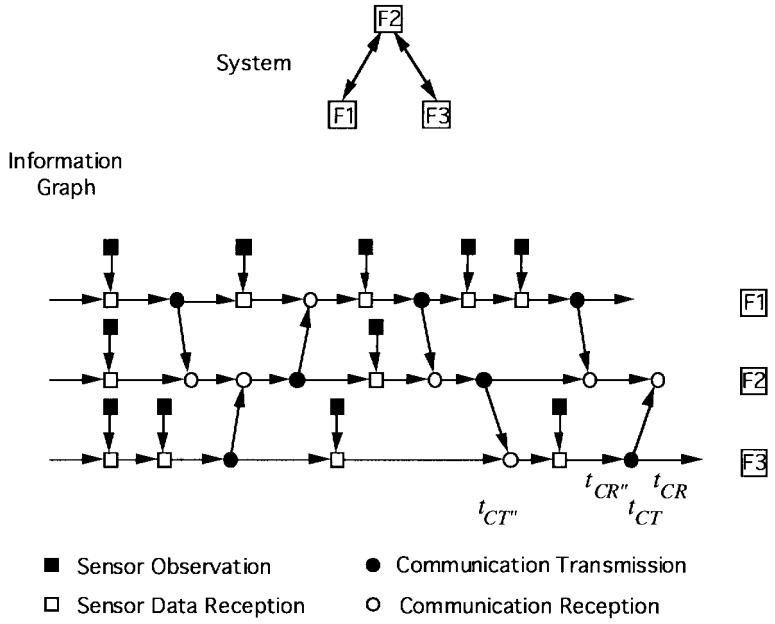

Fig. 5. Information graph: Hierarchical fusion with feedback.

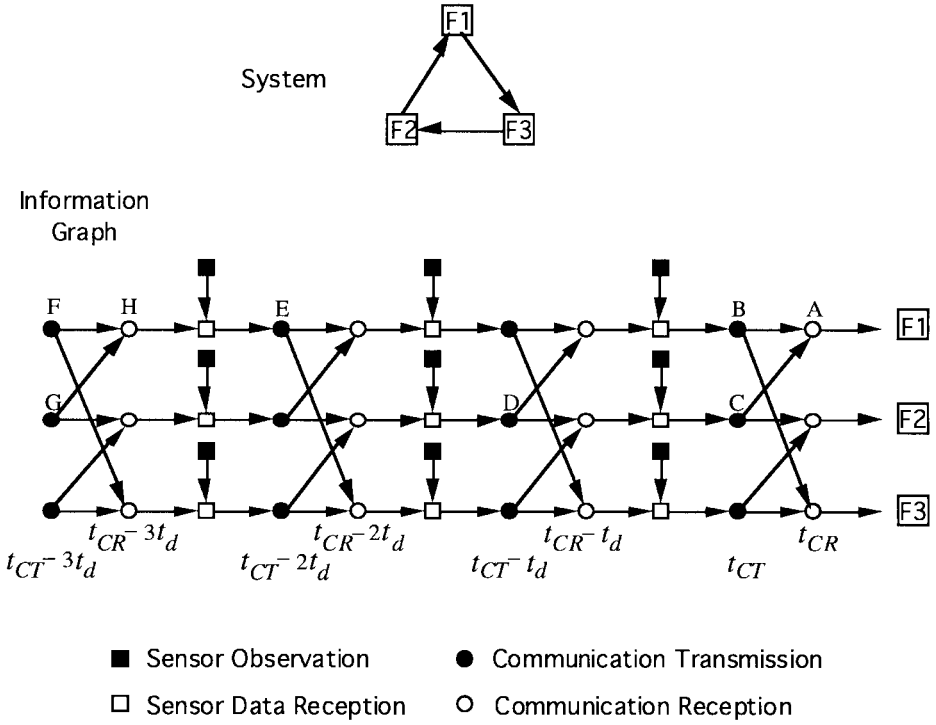

Fig. 6. Information graph: cyclic fusion.

do not process the data but just communicate them to Node 3, which then generates the optimal (centralized) estimate given the data. On the left side of the figure, each node generates the best estimate given its data and sends it to the fusion node (Node 3) which then combines the local estimates to obtain the optimal centralized estimate.

In order for the fused estimate to be the same as the optimal (centralized estimate), the information communicated by each node has to contain all the information needed to reconstruct the optimal estimate (sufficient statistics). Depending on the system architecture, this may include processing history and previous estimates. When this information cannot be communicated, the resulting fused estimate will be suboptimal.

IV. Distributed Estimation

As discussed before in Section II, a key part of target tracking is estimating the target state given the associated measurements. This target state may involve continuous variables such as position and velocity and discrete vari-

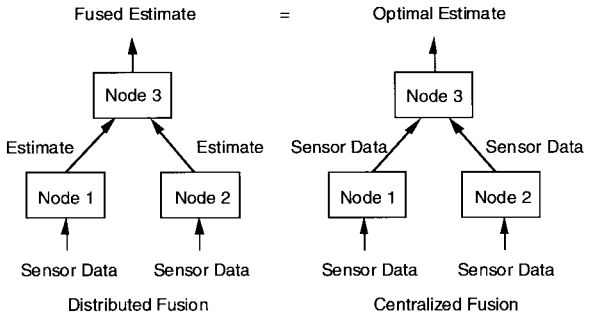

Fig. 7. Philosophy of fusion approach.

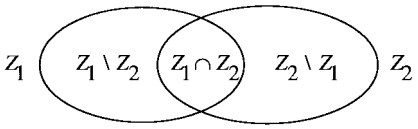

Fig. 8. Fusion by removing redundant information.

ables such as target type. In a distributed situation, the main problem is how to combine the estimates to arrive at the best estimate.

A. Basic Algorithm

Let x be the state (e.g., the location or type of a target) to be estimated and the measurements for the two nodes, one and two, be $z_{11}, z_{12}, z_{13}, \cdots$ and $z_{21}, z_{22}, z_{23}, \cdots$. These measurements may come from multiple sensors at different times or the same time. Assume the individual measurements are conditionally independent given the state x, i.e., $p(z_{ij}, z_{mn}|x) = p(z_{ij}|x)p(z_{mn}|x)$. This assumption is valid if the measurement errors are independent across sensors and over time. The total information available to the two nodes are then: $Z_1 = \{z_{11}, z_{12}, z_{13}, \cdots\}$ and $Z_2 = \{z_{21}, z_{22}, z_{23}, \cdots\}$. The conditional independence assumption implies:

$$p(Z_1 \cup Z_{12}|x) = \frac{p(Z_1|x)p(Z_2|x)}{p(Z_1 \cap Z_2|x)} \tag{1}$$

where $\cup$ and $\cap$ are the set union and intersection operations respectively.

Then the basic fusion equation is given by

$$\begin{aligned} p(x|Z_1 \cup Z_2) &= \frac{P(Z_1 \cup Z_2|x)p(x)}{p(Z_1 \cup Z_2)} \\ &= C^{-1}\frac{p(x|Z_1)p(x|Z_2)}{p(x|Z_1 \cap Z_2)} \end{aligned} \tag{2}$$

where $p(x|Z_1)$ is the estimate by node one, $p(x|Z_2)$ is the estimate by node two, $p(x|Z_1 \cup Z_2)$ is the estimate based on the joint information, $p(x|Z_1 \cap Z_2)$ is the estimate based on the common information, and C is a normalization constant. Basically each of the estimates contains the common information $p(x|Z_1 \cap Z_2)$, which has to be removed (divided) from the product of the two estimates to avoid any double-counting. This equation can be viewed as the distributed version of Bayes rule. Fig. 8 illustrates the concept behind this approach.

For estimation problems that are linear and Gaussian, the fusion equation becomes

$$P_{1\cup 2}^{-1} = P_1^{-1} + P_2^{-1} - P_{1\cap 2}^{-1} \tag{3}$$

and

$$P_{1\cup 2}^{-1}\hat{x}_{1\cup 2} = P_1^{-1}\hat{x}_1 + P_2^{-1}\hat{x}_2 - P_{1\cap 2}^{-1}\hat{x}_{1\cap 2} \tag{4}$$

where P_i and $\hat{x}_i$ are the covariance and mean for the Gaussian distribution $p(x|Z_i)$, and $i \cup j$ and $i \cap j$ correspond to union and intersection of the data sets. Equations (3) and (4) can be derived from (2) or directly from the following standard equations for linear estimation

$$P_i^{-1} = P_x^{-1} + \sum_k H_{ik}^T R_{ik}^{-1} H_{ik} \tag{5}$$

$$P_i^{-1}\hat{x}_i = P_x^{-1}\overline{x} + \sum_k H_{ik}^T R_{ik}^{-1} H_{ik}(z_{ik} - H_{ik}\overline{x}) \tag{6}$$

where $\overline{x}$ and P_x are the mean and covariance of x; the measurements $z_{ik}, i = 1, 2, k = 1, 2, \cdots$ are related to the state by

$$z_{ik} = H_{ik}x + v_{ik} \tag{7}$$

and the measurement errors v_{ik} are independent zero-mean with covariance R_{ik}.

The state x does not have to be static, i.e., it can be the state of a dynamic system, as long as the conditional independence assumption is valid. Under these assumptions the local estimates can be generated by nonlinear filters or Kalman filters and fusion involves only a static combination of the local estimates.

B. Hierarchical Fusion Algorithm

The basic fusion equations (2)–(4), can be used with the information graph to identify the common information and develop the fusion equation for any arbitrary fusion architecture. For the hierarchical architectures in Figs. 4 and 5, finding the common information is quite straightforward and the fusion equations are:

Without feedback:

$$p(x|Z(t_{CR}, 2)) = C^{-1}p(x|Z(t_{CR''}, 2))\frac{p(x|Z(t_{CT}, 3))}{p(x|Z(t_{CT''}, 3))}. \tag{8}$$

With feedback:

$$p(x|Z(t_{CR}, 2)) = C^{-1}p(x|Z(t_{CR''}, 2))\frac{p(x|Z(t_{CT}, 3))}{p(x|Z(t_{CT'}, 2))}. \tag{9}$$

Note that in each case the communication does not have to be synchronized and the local nodes can process many measurements before communicating with the high level node. The last term in each equation represents the new information received from the lower level fusion node. When there is no feedback, the new information is the difference between the current and previous estimates from the lower level. When there is feedback, the new information is the difference between the new estimate and

the last feedback from the higher level. The distributed nonlinear fusion equations in [24], [25] can be derived by including more nodes at the lower level. For example, assuming communication at each instant and no feedback, the fusion equation for the fusion node F with N lower level nodes is given by [24]

$$p(x|Z(t,F)) = C^{-1}p(x|Z(t-1,F))\prod_{i=1}^{N}\frac{p(x|Z(t,i))}{p(x|Z(t-1,i))}. \tag{10}$$

For linear observations with Gaussian random variables, fusion equations can be developed using a similar approach. Different fusion equations for linear systems have been developed in [10]–[23]. The following two common forms, first appearing in [11], were based upon the information form of the Kalman filter.

Without feedback:

$$\begin{aligned} P^{-1}(t|t) &= P^{-1}(t|t-1) + \sum_{i=1}^{N}[P_i^{-1}(t|t) \\ &\quad - P_i^{-1}(t|t-1)] \end{aligned} \tag{11}$$

$$\begin{aligned} P^{-1}(t|t)\hat{x}(t|t) &= P^{-1}(t|t-1)\hat{x}(t|t-1) \\ &\quad + \sum_{i=1}^{N}[P_i^{-1}(t|t)\hat{x}_i(t|t) \\ &\quad - P_i^{-1}(t|t-1)\hat{x}_i(t|t-1)]. \end{aligned} \tag{12}$$

With feedback:

$$P^{-1}(t|t) = \sum_{i=1}^{N} P_i^{-1}(t|t) - (N-1)P^{-1}(t|t-1) \tag{13}$$

$$\begin{aligned} P^{-1}(t|t)\hat{x}(t|t) &= \sum_{i=1}^{N} P_i^{-1}(t|t)\hat{x}_i(t|t) \\ &\quad - (N-1)P^{-1}(t|t-1)\hat{x}(t|t-1). \end{aligned} \tag{14}$$

In the above, $\hat{x}_i(t|t), \hat{x}_i(t|t-1), P_i(t|t)$, and $P_i(t|t-1)$ denote the updated and predicted estimates and error covariances for the lower level node i while the nonsubscripted quantities denote those at the higher level. As in the nonlinear case, (11) and (12) show that the higher level node fuses only the incremental information when there is no feedback. When there is feedback, the fusion node has to remove its own previously sent information before combining the local estimates. The process of removing the common estimates can be viewed as decorrelation to make the local estimates independent again.

C. Algorithm for Arbitrary Architecture

Identifying the common information so that it can be removed in fusion is relatively simple in a hierarchical fusion architecture. For more complicated distributed fusion architectures, identifying the common information is not as straightforward. The information graph introduced in Section III-A is useful for this purpose. When used in conjunction with the basic fusion equation (2), it can be used to derive fusion equations for arbitrary architectures [26]–[29]. For example, in the cyclic communication of Fig. 6, when the fusion equation is used a number of times, with the help of the predecessor nodes identified before, the following fusion equation can be obtained:

$$\begin{aligned} &p(x|Z(t_{CR},1)) \\ &\quad = C^{-1}\frac{p(x|Z(t_{CT},1))p(x|Z(t_{CT},2))}{p(x|Z(t_{CR}-2t_d,1))p(x|Z(t_{CR}-t_d,2))} \\ &\quad \cdot p(x|Z(t_{CR}-3t_d,1)). \end{aligned} \tag{15}$$

In this particular case, the common information shared by the two nodes is represented not by one estimate (or probability distribution) but by three.

For a general distributed fusion architecture characterized by an information graph, the fusion equation is given by

$$p(x|\bigcup_{i\in I} Z^{(i)}) = C^{-1}\prod_{j\in J} p(x|Z^{(j)})^{\alpha(j)} \tag{16}$$

where $Z^{(i)}; i \in I$ are the nodes (in the information graph) to be fused by the fusion node, J is a subset of information nodes that are predecessors of the set $I, \alpha(j) = \pm 1$ is an integer-valued function defined on J, and C is a normalizing constant. In this equation, $\alpha(j) = +1$ means that the information at the node is to be added (multiplication of probability) while $\alpha(j) = -1$ means that the information has to be removed (division of probability) because of redundancy. Equations for the means and covariances can be derived for Gaussian distributions to be of the form:

$$P^{-1} = \sum_{j\in J}\alpha(j)P_j^{-1} \tag{17}$$

$$P^{-1}\hat{x} = \sum_{j\in J}\alpha(j)P_j^{-1}\hat{x}_j. \tag{18}$$

In order to identify the common information using the information graph and generate the set J, the processing history (or pedigree) also needs to be communicated in addition to the estimate. Using this additional information, the fusion node can extract the relevant parts of the information graph and construct the optimal fused estimate on the fly. This approach supports fusion in arbitrary architectures with asynchronous communication. For some architectures with complicated information flow, the optimal fusion equation may involve many previous estimates. However, computational experience indicates that not much optimality is sacrificed if only the recent history is communicated. Also, the choice of the architecture and the design of the fusion algorithm should be considered together. A distributed fusion architecture that requires a complicated fusion algorithm may not be a good choice.

D. Dynamic States and Process Noise

The optimality of the fusion equations in reproducing the centralized estimates depends on the conditional independence of the measurements given the target state. For static target states, e.g., the state of a stationary target, the measurements are conditionally independent and the fusion equations always yield the optimal estimates.

For dynamic states, e.g., when the state represents the position and velocity of a moving target, the conditional independence assumption holds only when the dynamic model does not involve any process noise. For these so called deterministic processes [28], the distributed fusion equations produce the optimal estimates from the local estimates. In particular, in the hierarchical architecture, each local tracker can process many scans of sensor data before sending the results to the higher level to be fused [21]. This periodic communication reduces the amount of communication needed between the levels.

For dynamic states with process noise, the conditional independence assumption no longer holds unless synchronized communication takes place after each sensor observation time. The loss in conditional independence is due to the correlation from propagating the process noise. Since communication at every observation instant requires high bandwidth, hierarchical communication with periodic update is more desirable. In this case the fusion results produce only approximate results [19], [20]. Computational experience indicates that the approximation is quite good when the process noise is small.

E. Implementation Issues

The fusion equations in this section prescribe how local estimates should be combined to obtain the optimal estimates. In many instances, optimal fusion requires the use of other estimates communicated earlier so that redundant information can be removed. Distributed fusion imposes additional communication, memory, and computation requirements but it is possible to trade one for the other and between nodes. Consider the hierarchical fusion equations (11) and (12). These equations can be implemented in two different ways. The first is for the local nodes to compute the difference between the local updated and predicted estimates and send the new information to the higher level fusion node. The local nodes, rather than the higher level node, perform the decorrelation function. The second approach is for the local nodes to send the best estimates. The high level node stores the estimates from previous time, predict to the current time, find the difference with the updated estimates, and then computes the fused estimates. This approach increases the memory and computation requirements at the fusion node. In general, the specific implementation of the fusion equations depends on communication, computation, and memory considerations.

In some situations, the local nodes and fusion node may be interested in different parts of the state space. For example, the local nodes may collect data on local areas while the fusion node has responsible over a bigger area. For efficient computation, the local fusion node may use lower dimensional models than the fusion node. Conditions under which the optimal estimate can still be reconstructed have been investigated in [12] and [25].

V. Distributed Tracking and Correlation

In distributed fusion systems for target tracking, a fusion node or local tracker processes the local sensor data to generate target tracks and estimates on locations, velocities, type, etc. Target tracks from different nodes are then fused to improve on the tracks and state estimates. Fusion generally involves two operations (Fig. 9): associating the tracks from the different nodes to determine whether they correspond to the same targets, and combining the state estimates for those tracks that are associated. Note that this process is similar to association and updating in centralized fusion (Fig. 1).

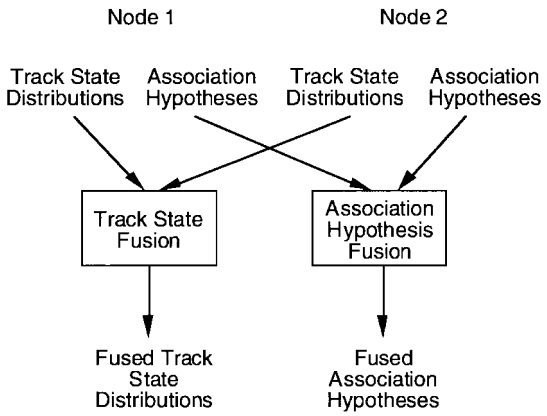

Fig. 9. Track state and association hypothesis fusion.

The problem of track state fusion given associated tracks has been extensively investigated and discussed in Section IV. By comparison, much less work has been done in distributed data association. This is partly due to the fact that data association is already complicated in its centralized form, with different techniques applicable to different scenarios. In the following we review several approaches for distributed data association.

A. Track-to-Track Association Likelihoods

A metric for determining whether two tracks should be associated is the likelihood of two tracks from different nodes being associated with the same target. Given a table of such metrics for all tracks, different association approaches such as optimal assignment or other suboptimal approaches can be used to make the association decision.

Suppose a track τ is formed by associating pairs of tracks, τ_j^1 and τ_k^2, one from each of the nodes 1 and 2. The information graph model can be used to derive the likelihood of the track τ [21]. Let the set J and the function $\alpha(j) = \pm 1$ be defined as in Section IV-C, and $p(x|Z^{(j)}, \tau^{(j)})$ be the state distribution of a track $\tau^{(j)}$ given the cumulative information $Z^{(j)}$. Then the likelihood of the track τ formed by associating τ_j^1 and τ_k^2 is given by [29]

$$L(\tau|Z) = \int \prod_{j \in J} p(x|z^{(j)}, \tau^{(j)})^{\alpha(j)} \mu(x). \qquad (19)$$

The state distribution of the fused track given the association is

$$p(x|Z, \tau) = C^{-1} \prod_{j \in J} p(x|Z^{(j)}, \tau^{(j)})^{\alpha(j)}. \qquad (20)$$

In the above, $\int \cdots \mu(x)$ is used to represent a generalized integral that may include summation for discrete variables

such as the type of a target in addition to integration over continuous variables such as position and velocity. If the two nodes do not share any common information at all, then the likelihood of associating two tracks is just the product of the track state distributions. The likelihood is high when there is much overlap in the state distributions of the tracks and low otherwise. Ideally the metric should involve the states of the tracks at all sensor observation times since good overlap of track states at a single instant may not represent a single target (e.g., tracks from two distinct targets that happen to converge). Using the states at all observation times satisfies the conditional independence assumption and makes the likelihood formula exact.

Since using the entire track history is sometimes not practical, most track association metrics consider only the most recent time. Under the Gaussian assumption, with only two nodes and prior density that is relatively flat, the likelihood becomes:

$$L(\tau|Z) = (\det(2\pi(P_1+P_2)))^{-1/2} \times \exp\cdot(-\tfrac{1}{2}(\hat{x}_1-\hat{x}_2)^T(P_1+P_2)^{-1}(\hat{x}_1-\hat{x}_2)) \quad (21)$$

where $\hat{x}_i$ and P_i are the means and covariances for the two tracks. This expression is similar to the standard equations for computing track-to-track association metrics and is valid when the conditional independence assumption holds.

When the conditional independence assumption is not satisfied, such as in a nondeterministic dynamic system with process noise, (21) is no longer exact. Expressions have been developed to approximate the association likelihood as well as the fusion of the track state estimates by introducing additional terms to compensate for the correlation in the tracks [30], [31].

Recently, a general track association metric based on (19) has been developed for nondeterministic processes [38] and hierarchical architecture. This metric is given by

$$L(\tau|Z_1,Z_2) = \frac{l(\tau|Z)}{l(\tau_1|Z_1)l(\tau_2|Z_2)} \quad (22)$$

where τ_1 and τ_2 are two tracks based on the data sets Z_1 and Z_2, τ is the track obtained by fusing τ_1 and τ_2 and Z is the fused data. The likelihoods on the right hand side are the track likelihoods given the data and are computed recursively using all the observation data. In linear Gaussian cases, they can be evaluated by Kalman filtering type operations. This metric uses the track data at all observation instants and has been shown to perform better than the traditional static distance.

B. Distributed JPDA

The distributed version of the JPDA algorithm for a hierarchical architecture is given in [36]. In this case, the state estimate for a particular target after association and fusion is given by

$$E\{x|Z^1,Z^2\} = \sum_{j=0}^{m_1}\sum_{l=0}^{m_2} E\{x|\chi_j^1,\chi_l^2,Z^1,Z^2\} \cdot P\{\chi_j^1,\chi_l^2|Z^1,Z^2\} \quad (23)$$

where $m_i, i=1,2$, are the numbers of measurements for the latest measurement sets of sensors one and two, $Z^i, i=1,2$ are the cumulative data sets, and χ_j^i is the association event (hypothesis) that z_j^i is the correct measurement, i.e., from the target. The first term on the right hand side is computed from the distributed estimation algorithm given the associations as described in Section IV. The second term is computed from the individual association probabilities as follows:

$$P(\chi_j^1,\chi_l^2|Z^1,Z^2) = \sum_{\chi^1}\sum_{\chi^2} P(\chi^1,\chi^2|Z^1,Z^2)\hat{\omega}_j^1(\chi^1)\hat{\omega}_l^2(\chi^2) \quad (24)$$

$$P(\chi^1,\chi^2|Z^1,Z^2) = \frac{1}{c}P(\chi^1|Z^1)P(\chi^2|Z^2)\gamma(\chi^1,\chi^2) \quad (25)$$

where χ^i are the joint feasible events involving all measurements and targets and $\hat{\omega}_j^i(\chi^i)$ are binary measurement-target association indicators.

Even though the product term implies that high individual association probabilities result in a high joint association probability, the additional term $\gamma(\chi^1,\chi^2)$ depends on the individual correlation events and reflects the influence of the actual measurement locations on the combined joint events. These equations were derived assuming communication after every observation and are only approximate with less frequent communication and in the presence of process noise.

C. Distributed MHT

The distributed version [27], [29] of the MHT has a structure similar to the distributed JPDA. Consider the case when a fusion node needs to fuse two sets of hypotheses and tracks (one can be local and the other coming from another site). Suppose the two sets of hypotheses and tracks are represented by $H^i(Z^i)$ and $T^i(Z^i), i=1,2$, and the probabilities of the hypotheses λ_j^i and state distributions for the tracks τ_j^i are given by $P(\lambda_j^i|Z^i)$ and $p(x|Z^i,\tau_j^i)$. The maximum information available to the fusion node is $Z = Z^1 \cup Z^2$. The goal of fusion is to obtain the hypothesis set $H(Z)$, track set $T(Z)$, hypothesis probabilities $P(\lambda|Z)$, and track state distributions $p(x|Z,\tau)$. Fusion consists of the following two steps:

1) *Hypothesis formation*. For each pair of hypotheses, λ_j^1 and λ_k^2, that can be fused, a track τ is formed by associating pairs of tracks, τ_j^1 and τ_k^2, one from each node, that could have originated from the same target. The result is a set of fused hypotheses $H(Z)$ and tracks $T(Z)$. Fig. 10 shows the distributed hypothesis and track formation process.
2) *Hypothesis evaluation*. The probability of each association hypothesis and the state estimate of each fused track are then computed. The distributed estimation algorithm is used to compute the likelihoods of possible associations and the resulting estimates for a given association. Using the information graph model, the

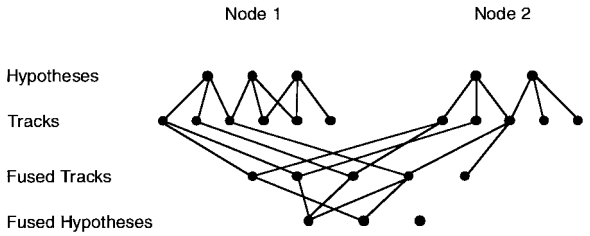

Fig. 10. Distributed hypothesis formation.

probability of each fused hypothesis is given by

$$P(\lambda|Z) = C^{-1} \prod_{j \in J} P(\lambda^{(j)}|Z^{(j)})^{\alpha(j)} \prod_{\tau \in \lambda} L(\tau|Z) \quad (26)$$

where C is a normalization constant. The likelihood of each fused track and the state estimates are given by (19) and (20). These expressions can be simplified under Gaussian assumptions.

VI. Examples of Applications

Distributed fusion techniques are applicable to many real world problems where surveillance performance can be enhanced by cooperation among sensing and processing nodes. Much of the early work was sponsored by Advanced Research Projects Agency (ARPA) under the Distributed Sensor Networks Program. A testbed for distributed acoustic tracking of air targets was developed at Lincoln Laboratory [39], [40], and the need for removing redundant information was noted. A distributed MHT approach to this problem was developed and demonstrated with both simulated and real data in [41].

Recent applications include the use of multiple platforms for airborne surveillance. Each platform has a suite of sensors and the sensors on different types of platforms have complementary capabilities. For example, a sensor (e.g., radar) on one platform has good location accuracy but poor classification performance, while another platform has a sensor (e.g., electronic support measure) that is good at classification but poor at location. Furthermore, some of the sensors may be good at wide-area surveillance while others are suitable for getting high resolution data. Thus by using a sensor from one platform to cue a sensor on another platform and sharing local processing results, distributed fusion and resource management provides much better performance than with the platforms working independently. A system based on this concept has been described in [42].

VII. Conclusion

In many applications distributed fusion is a better architecture than centralized fusion because of advantages such as distribution of processing load, lower communication bandwidth, local authority, higher reliability, etc. The availability of low-cost and high performance computers has also made distributed fusion feasible. However, distributed fusion also presents technical challenges that must be overcome before high performance systems can be developed. These include the proper choice of distributed architectures and the design of distributed fusion algorithms.

Over the past decade and a half significant advances have been made in the theory of distributed fusion for tracking. In particular, many distributed estimation algorithms have been developed for both linear and nonlinear systems, and hierarchical as well as general distributed architectures. These algorithms can be viewed as distributed extensions of linear and nonlinear estimation theory. To address the data association problem in tracking, distributed versions of standard approaches such as JPDA and MHT have also been developed.

Distributed fusion ideas are beginning to show up in many applications. Besides research prototypes, some current or planned surveillance systems have incorporated concepts of distributed fusion. The application efforts will provide stimulus for further research on issues such as impact of limited communication bandwidth, design of communication strategies [43], fusion of outputs from nodes using different models or algorithms [44], and distributed fusion management.

References

[1] E. Waltz and J. Llinas, *Multisensor Data Fusion*. Norwood, MA: Artech House, 1990.
[2] D. L. Hall, *Mathematical Techniques in Multisensor Data Fusion*. Norwood, MA: Artech House, 1992.
[3] R. R. Tenney and N. R. Sandell, Jr., "Detection with distributed sensors," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-17, pp. 501–510, July 1981.
[4] Z. Chair and P. K. Varshney, "Optimal data fusion in multiple sensor detection systems," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-22, pp. 98–101, Jan. 1986.
[5] Y. Bar-Shalom, "Tracking methods in a multitarget environment," *IEEE Trans. Automat. Contr.*, vol. AC-33, no. 4, pp. 618–626, 1978.
[6] S. S. Blackman, *Multiple-Target Tracking with Radar Application*. Norwood, MA: Artech House, 1986.
[7] Y. Bar-Shalom and T. E. Fortman, *Tracking and Data Association*. San Diego, CA: Academic, 1988.
[8] Y. Bar-Shalom and X. Li, *Multitarget—Multisensor Tracking: Principles and Techniques*. New York: YBS, 1995.
[9] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Englewood Cliffs, NJ: Prentice-Hall, 1979.
[10] J. L. Speyer, "Computation and transmission requirements for a decentralized linear-quadratic-Gaussian control problem," *IEEE Trans. Automat. Contr.*, vol. AC-24, pp. 266–269, April 1979.
[11] C. Y. Chong, "Hierarchical estimation," in *Proc. MIT/ONR Workshop on C3*, Monterey, CA, 1979.
[12] A. Willsky *et al.*, "Combining and updating of local estimates along sets of one-dimensional tracks," *IEEE Trans. Automat. Contr.*, vol. AC-27, pp. 799–813, Aug. 1982.
[13] B. C. Levy, D. A. Castanon, G. C. Verghese, and A. S. Willsky, "A scattering framework for decentralized estimation problems," *Automatica*, vol. 19, no. 4, pp. 373–384, 1983.
[14] H. R. Hashemipour, S. Roy, and A. J. Laub, "Decentralized structures for parallel Kalman filtering," *IEEE Trans. Automat. Contr.*, vol. AC-33, pp. 88–93, Jan. 1988.
[15] H. F. Durrant-Whyte, B. S. Y. Rao, and H. Hu, "Toward a fully decentralized architecture for multi-sensor data fusion," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1990.
[16] N. A. Carlson, "Federated square root filter for decentralized parallel processes," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 26, pp. 517–525, May 1990.
[17] T. H. Kerr, "Comments on 'federated square root filter for decentralized parallel processes' and author's reply," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 27, pp. 946–951, Nov. 1991.
[18] B. S. Y. Rao, H. F. Durrant-Whyte, and J. A. Sheen, "A fully decentralized multi-sensor system for tracking and surveillance," *Int. J. Robot. Res.*, vol. 12, no. 1, pp. 20–44, Feb. 1993.

[19] B. Belkin, S. L. Anderson, and K. M. Sommar, "The pseudo-measurement approach to track-to-track data fusion," in *Proc. 1993 Joint Service Data Fusion Symp.*, pp. 519–538, 1993.
[20] R. Lobbia and M. Kent, "Data fusion of decentralized tracker outputs," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 30, pp. 787–799, July 1994.
[21] O. E. Drummond, "Feedback in track fusion without process noise," in *Proc. SPIE, Signal and Data Process. of Small Targets*, vol. 2561, pp. 369–383, 1995.
[22] G. Frenkel, "Multisensor tracking of ballistic targets," in *Proc. SPIE, Signal and Data Process. of Small Targets*, vol. 2561, 1995.
[23] O. E. Drummond, "Track fusion with feedback," in *Proc. SPIE, Signal and Data Process. of Small Targets*, vol. 2759, pp. 342–360, 1996.
[24] D. Castanon and D. Teneketzis, "Distributed estimation algorithms for nonlinear systems," *IEEE Trans. Automat. Contr.*, vol. AC-30, pp. 418–425, May 1985.
[25] A. Alouani, "Distributed estimators for nonlinear systems," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 1078–1081, Sept. 1990.
[26] C. Y. Chong, S. Mori, and E. Tse, "Distributed estimation in networks," in *Proc. 1983 Amer. Contr. Conf.*, San Francisco, CA, June 1983.
[27] C. Y. Chong, S. Mori, and K. C. Chang, "Information fusion in distributed sensor networks," in *Proc. 1985 Amer. Contr. Conf.*, 1985, pp. 830–835.
[28] ——, "Adaptive distributed estimation," in *Proc. 26th IEEE Conf. Decision and Contr.*, Los Angeles, CA, Dec. 1987, pp. 2233–2238.
[29] ——, "Distributed multitarget multisensor tracking," in *Multitarget Multisensor Tracking: Advanced Applications*, Y. Bar-Shalom, Ed. Norwood, MA: Artech House, 1990, pp. 247–295.
[30] Y. Bar-Shalom, "On the track-to-track correlation problem," *IEEE Trans. Automat. Contr.*, vol. AC-26, no. 2, pp. 571–572, Apr. 1981.
[31] Y. Bar-Shalom and L. Campo, "The effect of the common process noise on the two-sensor fused track covariance," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-22, pp. 803–805, Nov. 1986.
[32] Y. Bar-Shalom, T. E. Fortmann, and M. Scheffe, "Joint probabilistic data association for multiple targets in clutter," in *Proc. 1980 Conf. Inform. Sci. Syst.*, Princeton Univ., Princeton, NJ, 1980.
[33] T. E. Fortman, Y. Bar-Shalom, and M. Scheffe, "Multi-target tracking using joint probablistic data association," in *Proc. 19th IEEE Conf. Decision Contr.*, Albuquerque, NM, 1980, pp. 807–812.
[34] D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. Automat. Contr.*, vol. AC-24, pp. 843–854, Dec. 1989.
[35] S. Mori, C. Y. Chong, E. Tse, and R. P. Wishner, "Tracking and classifying targets without a priori identification," *IEEE Trans. Automat. Contr.*, vol. AC-31, pp. 401–409, May 1986.
[36] K. C. Chang, C. Y. Chong, and Y. Bar-Shalom, "Joint probabilistic data association in distributed sensor networks," *IEEE Trans. Automat. Contr.*, vol. AC-31, no. 10, pp. 889–897, Oct. 1986.
[37] T. Kurien and M. Liggins, "Report-to-track assignment in multisensor multitarget tracking," in *Proc. 27th IEEE Conf. Decision and Contr.*, Austin, TX, Dec. 1988.
[38] S. Mori, K. A. Demetri, W. H. Barker, and R. N. Lineback, "A theoretical foundation of data fusion—General track association metric," in *Proc. 7th Joint Service Data Fusion Symp.*, Johns Hopkins Univ., Baltimore, MD, Oct. 1994, pp. 585–594.
[39] "Distributed sensor networks," Final Rep., MIT Lincoln Lab., Lexington, MA, Sept. 1986.
[40] R. T. Lacoss, "Distributed mixed sensor aircraft tracking," in *Proc. Amer. Contr. Conf.*, Minneapolis, MN, 1987, pp. 1827–1830.
[41] C. Y. Chong, K. C. Chang, S. Mori, and D. S. Spain, "Tracking multiple air targets with distributed acoustic sensors," in *Proc. Amer. Contr. Conf.*, Minneapolis, MN, June 1987.
[42] J. S. Jones and M. E. Liggins, "Off-board cueing and real-time sensor resource management for Joint STARS," in *Proc. 8th Natl. Symp. on Sensor Fusion*, vol. 1, Mar. 1995.
[43] I. Kadar, Ed., "Optimal communications strategies with bandwidth constraints in distributed fusion," to appear in *Signal Processing, Sensor Fusion and Target Recognition VI, Proc. SPIE 7505*, Apr. 1997.
[44] ——, "Adaptive sensor fusion," in *Signal Processing, Sensor Fusion and Target Recognition V*, I. Kadar and L. Vibeke, Eds.., *Proc. SPIE 2484*, pp. 75–82, Apr. 1995.

Martin E. Liggins II (Member, IEEE) received the B.S.E.E. degree from the University of Texas, Austin, and the M.S.E. degree in electrical engineering from California State University, Northridge.

In 1986 he joined Rome Laboratory, Rome, NY, where he has worked on developing a number of fusion and tracking testbeds for analysis and application. From 1982 to 1986 he was an Avionics Flight Test Engineer for radar and navigation at Edwards Air Force Base, CA. His research interests have included detection, tracking, multisensor and data fusion, information fusion, and system resource mangement.

Mr. Liggins is a senior member of the National Symposium on Sensor and Data Fusion, and was its chairman in 1995. He is a member of Eta Kappa Nu.

Chee-Yee Chong (Member, IEEE) received the S.B., S.M., and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, MA.

Since 1991 he has been with Booz, Allen and Hamilton, Inc., Mountain View, CA (the firm acquired Advanced Decision Systems (ADS), where he had beeen working since 1980). From 1973 to 1980 he was on the faculty of the School of Electrical Engineering, Georgia Institute of Technology, Atlanta, GA. His current research interests include centralized and distributed tracking and fusion, resource planning and scheduling, reasoning with uncertainty, distributed decision making, integration of system techniques, and artificial intelligence. He was an Associate Editor for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL and published many papers and book chapters on tracking and fusion and control theory.

Dr. Chong is a member of Tau Beta Pi and Eta Kappa Nu.

Ivan Kadar (Senior Member, IEEE) received the B.E.E. degree from the City College of City University of New York, the M.S.E.E. degree from Columbia University, New York, and the Ph.D. degree in electrical engineering from the Polytechnic Institute of New York.

In 1963, he joined Grumman Corp. (now Northrop Grumman) where he is now a Principal Scientist in the Advanced Technology and Development Center. During 1984–1985 he was Technical Advisor to the Director of Systems Engineering and Advanced Technology Department at Grumman. In addition, during 1979–1981, he worked with IBM's T. J. Watson Research Center in Systems Analysis and Algorithms of Computer Sciences. He also served as Grumman's Principal Investigator and Manager for the Rome Laboratory Pre-detection Fusion program. His research interests include integration of surveillance systems technologies, multisensor data association and information integration/fusion, wireless communications, distributed systems and networks, resource management, decision aid and situation/threat assessment systems, uncertainty models and knowledge representation, automated decision making, robust knowledge-based signal/image processing, and pattern recognition. He has authored more than 60 papers and edited two books.

Dr. Kadar is the Industry Chair of the Technology Committee of the Automatic Target Recognizer Working Group. Since 1988 he has been invited conference organizer and chair of SPIE's Signal Processing, Sensor Fusion, and Target Recognition I-VI conferences. He received the IEEE Region I Award, the IEEE Aerospace and Electronics Systems Society's M. Barry Carlton Best Paper Award, and the AIAA Space Shuttle Flag Award. He is a member of Sigma Xi.

Mark G. Alford (Member, IEEE) received the B.S. degree in electrical engineering from the State University of New York, Buffalo, in 1983, and the M.S. degree in electrical engineering from Syracuse University, Syracuse, NY, in 1988.

In 1981 he joined the Intelligence and Reconnaissance Directorate at Rome Laboratory, Rome, NY, as an Engineering Aide, and during 1983–1986 he was with its Signal Intelligence Division. From 1986 to 1989 he was with General Electric's Aerospace Electronics Division, Utica, NY. In 1989, he returned to Rome Laboratory, where he has been working in the Advanced Concepts and Analysis Branch of the Surveillance Technology Division of the Surveillance and Photonics Directorate. His current research interests include detection, tracking, and identification of weak signal targets, multisensor multispectral fusion, information fusion, systems analysis, signal processing, and artificial intelligence.

Mr. Alford is a member of Tau Beta Pi.

Vincent Vannicola (Member, IEEE) received the B.A. degree in physics from the University of Connecticut, Storrs, and the M.S.E.E. and E.E. degrees from Syracuse University, Syracuse, NY, in 1956, 1969, and 1973, respectively. In 1982 he received the Ph.D. degree in electrical and computer engineering from the same university.

From 1956 to 1962 he was a product design engineer with the Light Military Electronics Laboratory. In 1963 he joined Rome Laboratory, where his work focused on high-power microwave components, dielectric gases, high-resolution microwave networks, detection, adaptive radar algorithms, optical signal processing, polarization processing, and the modeling and estimation of random processes for frequency and phase instability in signal sources. His current research interest is in the use of AI methods toward enhancing signal processing in radar and communication systems.

Stelios Thomopolous (Senior Member, IEEE) received the Diploma in electrical and mechanical engineering from the National Technical University of Athens, Greece, in 1978, and the M.S. and Ph.D. degrees in electrical engineering from the State University of New York at Buffalo, in 1981 and 1983, respectively.

In 1995 he co-founded INTELNET Inc., State College, PA, of which he is the President and CEO. He is principal designer of INTELNET's SmartBoard, Ver-i-Fus, and Fus-A-Fis systems, and real-time information tracking software. He also co-founded Intelnet Consulting in 1984. Prior to his present position, he held faculty positions at the Pennsylvania State University (1989–1996) and Southern Illinois University (1983–1989). he has lectured short courses on data fusion in the United States and abroad. He was responsible for the design, implementation, and validation of the distributed fusion system, RobCFAR, under the Rome Lab Predetection Fusion Program. In late 1993 he was invited by the French CNRS as an international expert and adviser on data fusion to audit the French activities on data fusion. he has published extensively in the areas of distributed decision and estimation theory, data fusion, adaptive control, neural networks, and data networks.

Dr. Thomopolous organized and chaired the 1993 Conference on Data Fusion for Substance Identification and Security, in Innsbruck, Austria.