0 8 DEC 1994

Ref: 94-F-2210

Professor Joseph Wilder
Center for Computer Aids for Industrial Productivity
Rutgers University
Piscataway, NJ 08855-1390

Dear Professor Wilder:

This responds to your October 6, 1994, Freedom of Information Act (FOIA) request pertaining to a 1989 presentation by Michael Kovacich titled *Application of Bayesian Networks to Midcourse Multi-target Tracking*; and, a 1989 Institute for Defense Analysis Paper by Barry Fridling titled *Survey of Strategic Defense Initiative Tracking Algorithms* (IDA Paper P-2284). Our October 18 interim response refers.

The Ballistic Missile Defense Organization has provided the enclosed records as responsive to your request. After providing 100 pages of reproduction at no charge, the remaining chargeable cost consists of 165 pages of office copy reproduction at $0.15 per page ($24.75). However, in this instance, all charges have been waived.

Sincerely,

SIGNED

A. H. Passarella
Acting Director
Freedom of Information
and Security Review

Enclosures:
As stated

Prepared by Kahn:4F2210L1gr:12/8/94:DFOI:X71160:gr✓pk__yl__wh__

IDA PAPER P-2284

SURVEY OF STRATEGIC DEFENSE INITIATIVE TRACKING ALGORITHMS

Gabriel Frenkel
Task Leader
Barry E. Fridling
Principal Author

November 1989

Prepared for
Strategic Defense Initiative Organization

INSTITUTE FOR DEFENSE ANALYSES
1801 N. Beauregard Street, Alexandria, Virginia 22311-1772

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 1989	Final--Oct 1988 to Sept 1989

4. TITLE AND SUBTITLE
Survey of Strategic Defense Initiative Tracking Algorithms

5. FUNDING NUMBERS
C - MDA 903 89 C 0003
T - T-R2-597.1

6. AUTHOR(S)
Gabriel Frenkel, Barry E. Fridling

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Institute for Defense Analyses
1801 N. Beauregard St.
Alexandria, VA 22311

8. PERFORMING ORGANIZATION REPORT NUMBER
IDA Paper P-2284

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Strategic Defense Initiative Organization
Washington, DC 20301-7100

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution limited to U.S. Government agencies and their contractors; Adminstrative/Operational Use, 19 December 1989. Other requests must be referred to Strategic Defense Initiative Organization, The Pentagon, Washington, DC 20301-7100.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

This paper presents an overview of tracking methods and issues written for the nonexpert and an overview of tracking algorithm architectures. We summarize the algorithms surveyed to facilitate the understanding of survey responses and underscore the algorithms' general features, information flow, and calculational techniques.

The paper endeavors to serve both the nonexpert and the expert. For the nonexpert, the survey of methods explains tracking problems and solutions to provide both a summary of the state of the art and the technical background for the algorithm survey. For the expert, the survey of algorithms provides a catalog of approaches and results in tracking and a community of designers with whom to interact.

14. SUBJECT TERMS
Multiple target tracking, Kalman filters, Tracking Algorithm architecture, Data fusion, Passive sensors, Closely Spaced Objects

15. NUMBER OF PAGES
175

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR

DEFINITIONS

IDA publishes the following documents to report the results of its work.

Reports

Reports are the most authoritative and most carefully considered products IDA publishes. They normally embody results of major projects which (a) have a direct bearing on decisions affecting major programs, (b) address issues of significant concern to the Executive Branch, the Congress and/or the public, or (c) address issues that have significant economic implications. IDA Reports are reviewed by outside panels of experts to ensure their high quality and relevance to the problems studied, and they are released by the President of IDA.

Group Reports

Group Reports record the findings and results of IDA established working groups and panels composed of senior individuals addressing major issues which otherwise would be the subject of an IDA Report. IDA Group Reports are reviewed by the senior individuals responsible for the project and others as selected by IDA to ensure their high quality and relevance to the problems studied, and are released by the President of IDA.

Papers

Papers, also authoritative and carefully considered products of IDA, address studies that are narrower in scope than those covered in Reports. IDA Papers are reviewed to ensure that they meet the high standards expected of refereed papers in professional journals or formal Agency reports.

Documents

IDA Documents are used for the convenience of the sponsors or the analysts (a) to record substantive work done in quick reaction studies, (b) to record the proceedings of conferences and meetings, (c) to make available preliminary and tentative results of analyses, (d) to record data developed in the course of an investigation, or (e) to forward information that is essentially unanalyzed and unevaluated. The review of IDA Documents is suited to their content and intended use.

The work reported in this document was conducted under contract MDA 903 89 C 0003 for the Department of Defense. The publication of this IDA Paper does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that Agency.

This Paper has been reviewed by IDA to assure that it meets the high standards of thoroughness, objectivity, and appropriate analytical methodology and that the results, conclusions and recommendations are properly supported by the material presented.

IDA PAPER P-2284

SURVEY OF STRATEGIC DEFENSE INITIATIVE TRACKING ALGORITHMS

Gabriel Frenkel
Task Leader
Barry E. Fridling
Principal Author

November 1989

INSTITUTE FOR DEFENSE ANALYSES

Contract MDA 903 89 C 0003
Task Order T-R2-597.01

PREFACE

This paper constitutes a deliverable to Task T-R2-597.01, "SDI Battle Management/ C^3 Studies" in accordance with Section 5.0, "Schedule," of the task order dated 1 October 1988. It presents an overview of tracking methods and issues writen for the nonexpert, and an overview of tracking algorithm architectures. We summarize the algorithms surveyed to facilitate the understanding of survey responses and underscore the algorithms' general features, information flow, and calculational techniques.

The paper endeavors to serve both the nonexpert and the expert. For the nonexpert, the survey of methods explains tracking problems and solutions to provide both a summary of the state of the art and the technical background for the algorithm survey. For the expert, the survey of algorithms provides a catalog of approaches and results in tracking and a community of designers with whom to interact.

CONTENTS

EXECUTIVE SUMMARY

A. ALGORITHM SURVEY

Table 1 provides one summary of the 19 algorithms surveyed in this report. The general information about an algorithm that we felt should be immediately available includes: What phase of a ballistic missile's trajectory does the algorithm track?; Does the algorithm use single or multiple sensors?; Does it track clusters of objects as well as individual objects?; Does the algorithm rely on track information being handed over by another source? and What is the status of the algorithm? Other information of a more technical nature is provided in Table 2: In what manner does the algorithm share information among multiple sensors?; Does the algorithm perform measurement-to-track association by assignment or multiple hypotheses? For a detailed description of these terms and summary of algorithms please see Chapters 2-4.

The tables illustrate that tracking algorithm activities, at least those we surveyed, are concentrated in boost and midcourse phases, mostly multiple sensor tracking of individual objects. In addition, a few organizations are developing algorithms for tracking closely spaced objects or clusters. The manner in which the tracking information is processed varies, as can be seen from the different types of algorithm architectures implemented. Last, and most important, both major approaches to the association problem are being addressed.

B. CRITICAL ISSUES[1]

SDI tracking algorithms will face demonstration and evaluation milestones in the near future. Solutions to the most difficult tracking problems considered both in isolation and as part of a surveillance system remain to be successfully demonstrated. In particular, the future activities in SDI tracking algorithms must focus on five critical issues:

---

1 This section draws heavily on discussions by the SDI Panels on Tracking and from Drummond, O.E., "Multiple Target Tracking Lecture Notes," 18 March 1988, Technology Training Corporation, Torrance, CA.

Table 1.

	Phase			Sensor Number		Subject			Status	
	boost	mid-course	terminal	single	multiple	object	group	Conceptual	Feasibility Testing	Extensive Testing
Advanced Systems Architecture	√	√			√	√	√	√		
Alphatech		√			√	√				√
Ball Systems Engineering		√		√		√			√	
Caltech/Jet Propulsion Laboratory	√				√	√				√
ESL		√		√		√				√
Hughes Aircraft Company		√		√	√		√		√	
Lockheed Missiles and Space Company	√	√			√	√	√		√	
MIT Lincoln Laboratory		√			√	√	√			√
McDonnell Douglas Space Systems		√			√	√			√	
MindGate Technologies		√		√		√	√	√		
MITRE: Ballistic Tracker		√			√	√				√
MITRE: Boost-phase Tracker	√				√	√				√
MITRE: Multiple Sensor/Target	√				√	√				√
Raytheon: BMEWS Upgrade	√	√		√		√				√
Raytheon: Ground-Based Radar		√	√	√		√				√
Space Computer	√	√		√		√	√		√	
Systems Control Technology	√	√		√		√				√
TITAN Systems									√	
TRW		√		√		√	√			√

Table 2.

	Track Initiation		Track Initiation Algorithm Architecture				Track Maintenance Algorithm Architecture				Association		Track before detect
	cold	warm	I	II	III	IV	I	II	III	IV	Assignment	Multiple Hypothesis	
Advanced Systems Architecture	√				√					√		√	
Alphatech	√			√				?	?	?	√	√	
Ball Systems Engineering		√					?					√	
Caltech/Jet Propulsion Laboratory	√		√					√				√	
ESL		?					√					√	
Hughes Aircraft Company		√					√	√	√	√	?	?	
Lockheed Missiles and Space Company	√	√		√						√		√	
MIT Lincoln Laboratory	√			√						√	√		
McDonnell Douglas Space Systems	√				√					√	√		
MindGate Technologies		√					√				√		
MITRE: Ballistic Tracker		√								√	√		
MITRE: Boost-phase Tracker	√				√				√		√		
MITRE: Multiple Sensor/Target	√				√				√		√		
Raytheon: BMEWS Upgrade	√		√				√				√	√	
Raytheon: Ground-Based Radar	√	√	√				√				√	√	
Space Computer	√												√
Systems Control Technology	√												√
TITAN Systems													
TRW	√		√				√				√		

- Cluster tracking
- Scan-to-scan association
- Performance evaluation and prediction methods
- Testbed
- Signal processing.

1. Cluster Tracking

Cluster tracking methods are currently under development at a few organizations, as demonstrated by Table 3.

Table 3. Cluster Tracking Activities

Organization	Description of Algorithm
Advanced Systems Architectures	Tracks spawned from a common source are combined to form a cluster track. Cluster tracks are also formed from objects with similar tracks. No mention of unresolved objects.
Hughes Aircraft Company	State-of-the-art multiple sensor tracking of unresolved or resolved clusters of objects. Group-to-object transition included.
Lockheed Missiles & Space Company	State-of-the-art multiple sensor tracking of unresolved or resolved clusters of objects. Group-to-object transition included.
MIT Lincoln Laboratory	Individual object tracks for members of a group of closely spaced objects (CSO) are initiated from a cluster track that is generated by the edges of the cluster.
MindGate Technologies	Pattern matching of clusters of co-moving objects.
Space Computer Corporation	Track-before-detect approach for determining clusters in velocity space. Potentially very useful as a method for performing computationally affordable individual object track initiation.
TRW	Cluster tracking to implement a pattern-matching track initiation algorithm.

The many conceptual difficulties in approaches to cluster tracking are just beginning to be addressed. Much more work in this area is needed. However promising they may be, none of the algorithms listed in Table 3 have been demonstrated to be a solution to the cluster tracking problem. The SDI Panels on Tracking are actively pursuing this issue.

Individual object tracking will be impossible, too difficult, or too expensive to be practical, or not necessary during portions of the ballistic missile defense engagement. Shortly after deployment from post-boost vehicles, reentry vehicles and decoys may be so closely spaced as to be unresolvable to tracking sensors presenting as point or extended objects. Even when resolved, RVs and decoys could be so closely spaced that the computational resources required for individual object tracking would be prohibitively expensive because of scan-to-scan association difficulties. A sensor's ability to resolve objects depends on more than its optical qualities and signal processing. Resolution is also a function of the viewing geometry and range. For this reason, as the sensors move along their orbits, the closely spaced objects may *unresolve as well as resolve*. Therefore, cluster tracking will play a very important role in any tracking algorithm architecture.

There are critical operational requirements for maintaining tracks on individual targets, including discrimination of RVs from decoys, threat assessment, and attack execution. As the threat resolves, cluster tracks spawn individual object tracks, that is, individual object tracks are initialized from the cluster track. Therefore, cluster tracking should be evaluated, in part, on the quality of the initial estimates for the spawned individual object tracks.

2. Scan-to-Scan Association

There have been no full-scale demonstrations and evaluations of the two principal competing conceptual approaches to this problem. Much more work is needed in this area.

The great challenge of SDI tracking results from the high density of target and clutter observations reporting out of the sensor's signal processor. A high density means that the association of tracks to measurements cannot be made without significant uncertainty or error. At large but achieveable computational cost, tracks can be assigned to one scan's worth of observations in an optimal fashion. Incorrect assignments can lead the tracking system into estimating the quality of track predictions as better than they actually are. Since track prediction affects the assignment of weapons to targets and the ability of weapons to autonomously locate their targets, poor performance in this regard would have critical implications for the management of the ballistic missile defense engagement. Misassociations may also result in the loss of track as the filter follows an incorrect sequence of observations.

Misassignments that may occur in a high-density environment may result in tracks based on measurements from more than one target. Impure tracks over the course of many

scans randomly mix measurements from several targets. In this case, any phenomenology used to discriminate between classes of objects, such as RVs and decoys, that depends on repeated measurements will be of limited usefulness. In other words, poor track purity performance because of misassignments limits multiple scan discrimination techniques.

Assignment algorithms are computationally affordable but may not provide the necessary performance. On the other hand, multiple hypothesis algorithms should provide superior performance but their computational requirements may not be affordable. Rigorous, full-scale testing is critical.

3. Performance Evaluation and Prediction Methods

Much more work is needed in developing methodologies for fairly scoring tracking algorithms that use different conceptual and mathematical approaches. Until recently, there was no such scoring methodology. One scoring methodology has been agreement developed by the SDI Panel on Tracking Parameters. Development of appropriate scoring criteria must keep pace with new approaches to tracking.

The complexity of complete SDI tracking algorithms is such that analytic track performance predictions do not exist. More work must be done in this area. The only alternative is to run computer simulations, which are costly and sometimes difficult to interpret.

Much more work also must be done on determining the required computational resources of tracking algorithms before expensive simulations are run. Computer throughput and memory demands will play a critical role in selecting tracking algorithms.

4. Testbed

There is a critical need for a *portable* testbed that can be used in the *development* of tracking algorithms, not just for evaluation purposes. Contractors are naturally reluctant to bring their algorithms to a central testbed facility during development to avoid revealing proprietary details and embarrassing algorithm performance, which is to be expected during development.

The portable testbed should consist of a complete set of library modules for those functions that support a tracking algorithm. Library modules would consist of accepted, standardized models for such things as possible threat trajectories and signatures,

background signatures, and signal processing.[2] The tracking algorithm designer should be able to select individual modules to plug into his sensor system in support of developing a tracking algorithm.

A modest step in this direction is threat data developed by members of the SDI Panel on Track Parameters that has been distributed to 13 organizations by IDA. A second generation of threat data is nearly ready for distribution. An approved signal processing model could be used to supply measurements for the tracking algorithm. In particular, realistic, accepted models of background clutter in which tracking algorithms are expected to operate need to be made available as soon as possible.

Currently, each algorithm design team must provide their own testbeds. This is done at great cost. The government pays for the redundant effort of contractors developing their own testbeds with their own supporting models.

The efforts of some contractors could be greatly handicapped by poor fidelity supporting models. This could, perhaps, prevent good ideas from getting adequate testing. In this case, the government would suffer the opportunity cost of losing a good idea.

The government pays an additional cost in that low fidelity support function models used by tracking algorithm designers may lead to misleading performance results that might not be discovered until late in the development cycle and at the great expense of full-scale computer simulations.

A portable testbed is critical for the ability to run full-scale simulations during development to provide tracking algorithm designers an opportunity to study the problems they face and to adequately exercise different approaches.

5. Signal Processing

There may be large potential for tremendous growth in tracking performance by better or new, innovative approaches to signal processing. For instance, if a signal processor could greatly decrease the high density of observations passed to the SDI tracking system by removing stars, persistent background, and decoys, the processing load for a space surveillance and tracking system would be substantially lessened. Although such an approach has been hypothesized, we have seen no evidence of such a bulk filter.

---

2 In addition, modules with alternative levels of fidelity will expand the usefulness of the testbed to various purposes and stages in the development process.

Signal processing is critical to discrimination. Measurements in high-density environments can be corrupted as signals from neighbors are mixed together. Impure measurements, like impure tracks, limit discrimination.

Signal processing lies at the heart of the cluster tracking or closely-spaced-object problem. Much of the work in this area is highly proprietary and was not available to us.

New, innovative signal processing approaches are under investigation. One in particular, known as a velocity filter, may provide a computationally affordable approach to track initiation in the dense SDI environment that is otherwise hugely expensive because of the scan-to-scan association problem. Velocity filters are one example of a class of approaches, known as track-before-detect, that have been found to hold great potential for tracking low observable objects.

I. INTRODUCTION

The Institute for Defense Analyses (IDA) was tasked by the Strategic Defense Initiative Organization (SDIO) to monitor, evaluate, and facilitate the development of tracking algorithms. Among other things, IDA was asked to survey tracking algorithms under development for or applicable to SDI in order to ascertain the status of activities in this critical area. This paper reports the results of IDA's survey.

As part of IDA's overall task, three working panels were established to provide a forum for addressing problems in tracking. These panels,[3] which meet bimonthly for three consecutive days, are staffed by tracking algorithm designers from Federal Contract Research Centers, many companies, and each of SDI's sensor program elements. The SDI Panel on Critical Issues in Tracking developed a common survey format for describing tracking algorithms to ensure that key questions were answered in a succinct manner and to simplify the process of understanding the details of the activities. One part of that format was a decomposition of all tracking algorithms into architectures consisting of four-track initiation and four track maintenance generic processing chains. It was expected that all responses would conform to the survey format and algorithm architectures.

To a large extent, the accomplishment of the survey was dependent on the tracking algorithm community responding to IDA's request for information. Given that replying to such a survey is not a contractual obligation, the number of answers received is gratifying and IDA expresses its appreciation to those who took the time to carefully and thoughtfully respond. This survey report does not contain algorithms for some SDI development efforts because of classification or proprietary restrictions.

The survey benefited greatly from the collective expertise of the SDI Panels on Tracking. A list of the contributors from these panels is included in Appendix C. The

---

3 The SDI Panel on Tracking Parameters, the SDI Panel on Critical Issues in Tracking, and the SDI Panel on Advanced Concepts. For a description of their activities see the Proceedings of the SDI Panels on Tracking.

survey also benefited greatly from three useful expositions on multiple-target tracking.[4] The reader desiring more depth on this subject is directed there.

The paper consists of an executive summary, four chapters, and three appendices. Chapter 2 presents an overview of tracking methods written for the non-expert. Chapter 3 presents an overview of tracking algorithm architectures developed by the SDI Panels on Tracking. Chapter 4 contains summaries of the surveyed algorithms that are intended to facilitate the understanding of the unprocessed responses and underscore the algorithms' general features, information flow, and calculational techniques. The survey form and the responses can be found in the appendices. Our conclusions and recommendations are contained in the Executive Summary.

The paper endeavors to serve both the nonexpert and the expert. For the nonexpert, the survey of methods explains tracking problems and solutions to provide both a summary of the state of the art and the technical background for the algorithm survey. For the expert, the survey of algorithms provides a catalog of approaches and results in tracking and a community of designers with whom to interact.

---

4 Samuel S. Blackman, *Multiple-Target Tracking with Radar Applications*, Artech House, Inc., 1986; O.E. Drummond, "Multiple Target Tracking Lecture Notes," UCLA October 1985, Revised 18 March 1988, Technology Training Corporation, Torrance, CA.; and Yaakov Bar-Shalom and Thomas E. Fortmann, *Tracking and Data Association*, Academic Press, Inc.

II. OVERVIEW OF TRACKING METHODS

Before summarizing the results of the algorithm survey, it is worthwhile to provide the nonexpert reader with an overview of tracking methods. We begin by explaining the tracking process in a general, introductory fashion to acquaint the reader with terminology and fundamental concepts. The difficulties of SDI tracking are described next. Finally, the principal tracking approaches are described in depth but at a level appropriate for the nonexpert.

A. INTRODUCTION

1. What is Tracking?

A track is an estimate, based on sensor measurements, of the kinematic components (position, velocity, and acceleration) that describe the motion of an object. These components are collectively referred to as the **state of the object**; the state's evolution in time describes the object's motion. Therefore, a track is an estimate of the object's state derived from sensor measurements of it.[5]

The tracking process as generally practiced today consists of the interrelated functions of **association** and **estimation**. A (radar, optical, acoustic) sensor system uses its **observations** of an object's reflections or emissions to derive **measurements** of the object's state. The portion of the state that is measured depends on the sensor type. For instance, a passive optical observation cannot provide range measurements because it relies entirely on emissions. In general, the measurements are some (not necessarily linear) function of the state.

Association is the decision process of **linking** observations or tracks of a **common origin**. Links can be made observation to observation, observation to track, or track to track. Observations taken at (nearly) the same time by multiple platforms or from one platform's different sensor systems can be linked together as assumed to have a common origin for the purpose of **sensor fusion**.

---

5 The state could also include quantities other than kinematic components, such as temperature.

A sensor's observations can be linked across frames[6] to form a **time sequence** of measurements. This sequence without further processing is a type of track, one that lacks the ability to predict the future and is limited to the measurements rather than the full state. A time sequence of linked observations, however, can be processed, that is, **statistically filtered**, to transform a measurement sequence into an estimate of the full state's evolution as a function of time. Statistical filtering is the **estimation** part of tracking.

In the next section we will discuss in detail the Kalman filter, which currently is the most general and powerful track estimation method commonly used. Here it suffices to note that the Kalman filter refers to the algorithm that produces the statistical estimate of the state and covariance.

A major task in tracking algorithms is the linking of observations to tracks. A **gate** is a region in the sensor's field of view, determined in part by the prediction from the Kalman filter, where the subsequent track measurement is likely to fall. As a rule, only observations in the gate are considered for association with that track, thus greatly reducing the number of computations. Often in SDI applications there will be more than one observation within a gate and, therefore, several possible observation-track pairings. Since the output from a Kalman filter participates in determining the size of the gate, the association and estimation functions are interrelated. We will discuss this further below.

Last, as with observations, tracks from multiple sensors can be linked as assumed to have a common origin, also for the purpose of sensor fusion.

2. What are the Difficulties in SDI Tracking?

The essential difficulties in SDI tracking are the large number of objects to be tracked, the high density of observations, and the inability of sensors to resolve individual objects from closely spaced neighbors.

The massive number of objects that have to be tracked in SDI scenarios requires huge computing resources. To understand this, consider the computational burden from processing one Kalman filter for each object being tracked. Since all the tracks update on each frame, every few seconds the information processor must perform the necessary, highly nontrivial, update calculations of the Kalman filter. We see that this computer

---

6 A frame is defined as one data collection survey of the surveillance region. In this form, the definition is independent of whether the sensor surveils by mechanically sweeping the field of view with detectors or surveils electronically with staring detectors.

burden scales linearly with the number of tracks. Faster, more efficient Kalman filters can mitigate this huge computer load. Unfortunately, this is the least of the difficulties.

In common SDI scenarios, the density of observations originating from targets and clutter is likely to be high. Recall from our discussion above, the observation-to-track association function establishes a gate in the sensor's field of view around where it expects to find the observation for the track. A high density means that there is likely to be more than one observation in a gate. When this occurs, the tracking system cannot know with certainty which observation if any originates with the target. Handling this association problem is the most computationally intensive aspect of tracking. **Incorrect observation-to-track association can lead to poor track performance, loss of track, and tracking errors far worse in reality than those predicted by the Kalman filter.** This is the most critical SDI tracking difficulty.

One approach to managing the high density threat is to temporarily forego tracking individual targets and rely instead on tracking the group or cluster in which they are traveling. There are substantial computational advantages to this approach. A track can be established for some carefully chosen parameters of the group, such as the group centroid and extent. This saves on the computer resources required when many targets are so close that it is not practical to process many individual tracks and circumvents the problem of large numbers of misassociations likely to occur if the individual tracks were maintained.

Unresolvable closely spaced objects cause another major difficulty. The sensor's ability to resolve neighboring objects, of course, depends on the sensor, the range, and the viewing geometry. If unresolved, a group of closely spaced objects may appear as a relatively large (compared to the signal from individual objects) extended object on the sensor's detectors. A track, however, can be established on the extended object.

Another major tracking problem caused by CSOs is that the resolution can be unstable from frame to frame, e.g., two targets may be resolved on one frame and not resolved on the next and possibly resolved on the following. The instability of the measurements stresses the association and estimation processes.

B. ESTIMATION, ASSOCIATION, AND DECISION

1. Estimation: The Kalman Filter[7]

A statistical estimation filter consists of three parts: models for the dynamics and measurement processes, statistical assumptions, and an optimality criterion. The result is an algorithm for transforming measurements of a state into an estimate of it. The model for the object's dynamics describes its time evolution, which may contain elements, referred to as process noise, that are unknown or unpredictable, except for their statistics. One example is the random changes in acceleration typical of rocket boosters. A measurement model must both specify the relation between the state and the measurements and account for the generally random inaccuracies, referred to as measurement noise, always present in measurements. For instance, we may wish to determine the object's position, velocity, and acceleration from noisy position measurements alone. The most simplifying assumption is that the system noises are statistically independent, white processes.

Optimality criteria establish a measure for the "goodness" of an estimate. Once selected, an optimality criterion, such as maximizing the likelihood function or minimizing the mean square error between truth and estimate, leads to a procedure for transforming a set of measurements into an estimate for the state. When the random processes in the filter are assumed gaussian, or we require the estimate to be linear in the data, all optimality criteria lead to the same estimator.

The term filter can be thought of as a generic term for the **process** of recovering information from noisy measurements. Statistical filtering develops information from noisy measurements by assuming that the desired signal and unwanted noise can be distinguished by their statistical properties.

The term filter also refers to a type of information processing that is distinct from two related types, smoothing and prediction. Filtering means the recovery at some particular time, t_k, of information about the system using measurements up to and including that time. Smoothing differs in that the information about the system need not become available at t_k, and measurements derived later than t_k can be used in obtaining information

---

7 This section is based on material in the following publications: Andrew P. Sage and James L. Melsa, *Estimation Theory With Applications to Communications and Control*, McGraw-Hill Book Company, 1971; Brian D.O. Anderson and John B. Moore, *Optimal Filtering*, Prentice-Hall Inc., 1979.

about the system at t_k. Prediction is the forecasting type of information processing in which the aim is to obtain information at t_k about the state of the system at a later time.

The Kalman filter is the mathematically optimal estimator for deriving at some particular time, t_k, an estimate of the state and its covariance from measurements of the states.

The filtered estimate of the state is processed in two stages: a time update and a measurement update. The first step, the time update, is the filtered estimate of the state at the preceding time, t_{k-1}, predicted ahead one step. The measurement update involves the difference between the associated measurement and the predicted measurement, referred to as the **innovation**, multiplied by the **Kalman gain**. Calculated by the filter, the gain determines the weight given to the new measurement information. The prediction of the state at some future time is computed from the present filtered estimate, without employing the innovation process.

The filter also provides for the time and measurement update of the covariance of the state estimate. In addition, the algorithm calculates the covariance of the innovation process, which is the measurement prediction uncertainty, indicating the quality of the prediction.

2. Association: The Concept and Role of a Gate

The **innovation chi-square** is derived from the innovation and its covariance matrix. It specifies an elliptical volume in measurement space known as a **gate** that is an indication of the track prediction uncertainty. The gate establishes an acceptance or validation region into which observations considered for association with the track must fall. Landing within the gate is a necessary but not sufficient condition for an observation to be considered as having originated from the track because incorrect measurements may also fall in the gate or the target may not have been detected in the gate. The sole purpose of the gate is to decrease the processing load by decreasing the number of possible observation-to-track association pairs by limiting the number of candidates. Observations within the gate are often called validated.

The size of the gate is determined by fixing the probability that the correct measurement will fall within its volume. Since this probability is set to less than one there is a nonzero probability that the correct observation will not be a candidate for association. A larger ellipse enjoys a higher probability of capturing the measurement that originates

with the target but at the expense of possibly increasing the number of association candidates and thereby increasing the processing burden.

If a gate is empty, there is no measurement to update that particular track. In this case, the Kalman filter, which predicted the center of this gate, predicts ahead an additional frame but now without the benefit of measurement update information. The measurement prediction uncertainty is increased; thus, the gate is larger for the next frame.

There is some measurement origin uncertainty even if there is one measurement in the gate because the target may not be detected in the gate and the observation could be from a different target or a false signal. When updated, the measurement prediction uncertainty of the Kalman filter typically decreases regardless of the true source of the updating measurement because it fails to account for its origin uncertainty. In other words, the measurement prediction uncertainty typically decreases after the update *regardless* of the origin of the measurement.

For this reason, misassociations unless compensated for cause the filter to estimate the quality of its prediction as better than actual. Misassociations may also cause poor track performance, that is, a large gap between estimated and true tracks, and loss of track as the filter follows an incorrect sequence of observations. Discrimination and battle management is also adversely affected.

3. Decision: Strategies for Managing Measurement Origin Uncertainty

The fundamental multiple target tracking dilemma is deciding which measurement to use, if any, in updating a track. An observation has three possible sources: an individual target, clutter, or a set of unresolved targets. Following Bar-Shalom and Fortmann,[8] we define clutter as observations from background stationary objects, interference, environmental anomalies, false alarms, etc., that are generally random in number, location, and intensity.

The difficulties caused by not knowing the source of a measurement are shown by considering the tracking situations in Figure 1. In the single-target case, there are two measurements within the gate, each, or none, possibly originating from the target. The correct decision is not obvious. In the multiple-target case, the gates overlap, with measurement 3 falling in both. It is usually assumed that one measurement cannot be

---

8 Yaakov Bar-Shalom and Thomas E. Fortmann, *Tracking and Data Association*, Academic Press, Inc. p. 153.

simultaneously assigned to more than one track, that is, each observation is uniquely assigned. If we decide that measurement 3 is associated with track 2, then measurement 1 is the only candidate for association with track 1. **Thus, associations over multiple targets are interdependent.**

The manner in which the problem of uncertain measurement origin is managed can be used to categorize approaches to multiple target tracking. *Assignment* methods make a definitive decision, typically at each frame, on the origin of the measurement. One measurement (or none) from those within the gate is selected as having originated from the target.

Instead of selecting a single observation, the *probabilistic data association* (PDA) approaches avoid selecting by averaging over all decisions. The track is updated, using all measurements in the gate, weighted by the probability that they are correct. This is referred to by Blackman as an all-neighbors approach.

Multiple hypothesis tracking (MHT) algorithms defer a decision on the origin of the measurement. Multiple alternatives are retained as distinct tracks until later information improves the probability of the correct measurement-to-track association.

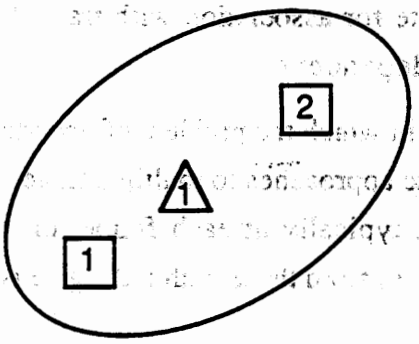
C. MULTIPLE TARGET TRACKING APPROACHES: ASSIGNMENT, PDA, MHT

1. Assignment

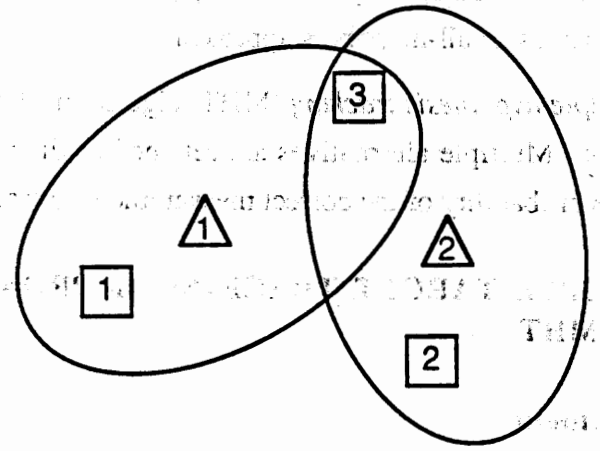
The simplest association scheme, known as the nearest-neighbor algorithm, assigns a track to the measurement "nearest" to the predicted measurement, such as calculated by the innovation chi-square. All other observations in the gate are disregarded.

One approach to multiple target tracking is to run a nearest-neighbor algorithm for each track, independent of all other tracks. This is referred to as uncoordinated nearest neighbor. In the situation shown in Figure 1b, the processor would decide for track 1 whether to update with measurement 1 or 3, or nothing, independent of track 2 assignments.

We have commented that this is unsatisfactory for high-density environments for the reason that associations over multiple targets are interdependent when we require unique assignments. This requirement is fundamental to simplifying the calculation



a. Single Target Case



b. Multiple Target Case

Predicted Measurement for Track 1

Actual Measurement

Figure 1.

because its effect is to make the different states statistically independent. Therefore, a nearest-neighbor algorithm is generally executed in a coordinated manner as follows.

A cost matrix is defined by all possible track assignments, including that the measurement is from a new target or false alarm, and all possible candidate measurements, including the case that the correct track measurement is not detected in the gate. The cost matrix entries are proportional to the probabilities of the assignments. These can involve the innovation chi-square for the measurement-track association pair, the probability of detection, the probability of the gate, the probability that the observation is from a new source, and the probability of choosing no observation for association with the track.

An algorithm, such as the Munkres algorithm, is run that assigns measurements to tracks in a coordinated fashion by maximizing the sum of matrix entries *subject* to the constraints that no track is updated by more than one measurement and one measurement is not assigned to more than one track. The results of such an algorithm is a unique pairing of tracks to observations.

Assignment algorithms are used not only to associate one list of observations with one list of tracks as just described. They can also be used to associate two lists of observations or two lists of tracks. Furthermore, there are assignment algorithms that can be used to associate data among more than two data lists, for instance linking several frames worth of observations to tracks. In this manner, assignment algorithms can generate multiple hypotheses, in the sense that more than one viable alternative per track is retained over a number of frames. The general distinguishing characteristic, however, of assignment algorithms is that of a definitive decision. For examples and discussion of multiple frame assignment algorithms see the Alphatech survey.

2. Probabilistic Data Association[9]

The Probabilistic Data Association Filter (PDAF) applies to a single target and is strictly a method for handling the problem of multiple observations within the gate of an established track. The fundamental idea is to exploit the **association probabilities** of the complete set of observations within the gate for the target.

In the PDAF, each observation in turn is considered as originating from the target. Also, the case that the observation originating from the target is not detected is considered.

---

9 This section borrows heavily from Bar-Shalom and Fortmann, ibid.

An association hypothesis consists of assigning one observation (or none) to the track and considering all others as statistically independent clutter.

The PDAF procedure first multiplies the probability of each association hypothesis with the updated state estimate that assumes that hypothesis is true; thereby weighting that state estimate. Then the final PDAF state estimate is formed as the weighted average, that is, the sum over the association hypotheses of the weighted average of the state estimates for each hypothesis. The estimate can be shown to be equal to the predicted state plus the weighted average of the individual innovation chi-squares multiplied by the standard Kalman gain.

The covariance of the final PDAF state estimate follows immediately as the average over the covariances for each hypothesis. This can be seen to be equal to a sum of three terms. One term is the prediction covariance multiplied by the probability that no observation originated with the target. A second term consists of the covariance of the state updated with the correct measurement multiplied by the probability that the target-originated observation is available. The last term increases the covariance of the total updated state to account for the uncertainty in the origin of the observation.

The Joint Probabilistic Data Association Filter (JPDAF) extends the fundamental idea of the PDAF to multiple targets by computing the association probabilities jointly across all targets rather than for each individual track. The final JPDA state estimate is calculated as before as an average over the association hypotheses.

To summarize, the PDA state estimate is an average over observation-to-track association hypotheses. Each hypothesis consists of an unique assignment of the track to an observation. The total state estimate is an average over the many feasible assignments for the one track.

The five principal distinguishing characteristics of the PDA approaches are the assignment of one track to many observations, one per hypothesis, the exploitation of association probabilities, the calculation of state estimates as averages over association hypotheses, an adjustment to the covariance for source uncertainty, and a lack of organic track initiation logic. The association probabilities are calculated with Bayes Theorem from Probability Theory. For this reason, PDA approaches are one member of a class of tracking approaches referred to as **Bayesian Tracking Algorithms**. We will discuss other members of this class shortly.

3. Multiple Hypotheses Tracking

An intuitive approach to managing multiple observations in a gate is to **split** the original track into many tracks, one for each validated observation. This process is known as **track splitting.** Each track is updated with the associated observation and carried forward to the next frame in the standard fashion. The fate of these many tracks depends on whether the associated observations arise from clutter or targets.

If the source of the associated observation is clutter, then subsequent observations for this track will be randomly detected and located. The quality of the track, therefore, is expected to decrease markedly. For this reason, a pruning mechanism is usually contained in track splitting algorithms to drop low-quality tracks.

An observation originating from targets may have four sources. First, the observation could be from the original target. In this manner, this algorithm ensures maintaining the original track. "Extra" observations within the gate could be from objects just released by a common carrier vehicle, such as reentry vehicles and decoys released from a post-boost vehicle. Also, the new observations could be newly resolved, closely spaced objects from what had been a single unresolved CSO. In these two cases, tracks are split, also referred to as **spawned**, by initializing the new tracks with the original track's state estimate on the previous frame.

Last, the extra observations could be from new targets just moving into detection range or field of view that happen to fall within the gate. Track splitting may provide poor estimates in this case because the original state estimate may have very little to do with the newly detected targets, except for their location on the sensor focal plane.

A track splitting algorithm has two limitations. First, the algorithm disregards all observations that fall outside the gate. For this reason, a separate track initiation algorithm must be included in the battle management system that uses this approach.

The second and major limitation of the track splitting algorithm is that association over multiple tracks is performed in an uncoordinated manner. There is no conflict resolution logic that manages the problem of observations within multiple gates.

Track splitting algorithms are distinguished by the assignment of multiple observations in the gate to one track, the deferring of difficult assignment decisions until more information is generated, and the absence of association probabilities based on global hypotheses.

Deferring difficult assignment decisions is prototypical of multiple hypothesis tracking algorithms but other MHT approaches use association probabilities based on global hypotheses. Recall that PDA algorithms use the probabilities of the multiple association hypotheses, each one consisting of unique assignments of the *latest* set of observations to the tracks from the prior frame.

Both the PDA and track splitting algorithms assign many observations to one track, but very differently. **Track splitting generates one candidate track for each validated observation assignment. PDA yields a final state estimate that is averaged over all candidate hypotheses.**

Two avenues for generalization are suggested. First, use association probabilities based on global hypotheses but maintain individual state estimates rather than average state estimates. Second, extend the association hypotheses over many frames rather than just the most recent. An Optimal Bayesian Tracking algorithm would generate hypotheses across all frames from the first through to the current.

Bar-Shalom and Fortmann[10] relate the PDA single frame approaches to an optimal Bayesian algorithm as follows. Consider a time sequence of observations, one observation per frame, from the initial to the present time. Such a sequence forms one possible target history, that is, one possible track. Consider all possible such sequences. The set of all possible assignments at the current frame can be decomposed into tracks at the previous frame associated with some observation from the current frame. A few moments thought reveals this is to be an abstract description of track splitting.

An association probability for each observation sequence, that is, a probability for each track, can be calculated, conditioned on the entire set of observations. As in PDA, the conditional probability for each hypothesis multiplied by the state estimate that assumes that hypothesis is true is summed over all possible hypotheses. Thus, the updated state estimate for a track is an average over the different possible association hypotheses.

Optimal PDA associates over all frames, not just the most recent. Its computational expense may be prohibitive. A suboptimal approach looks back N frames, referred to as N-backscan, rather than all the way to the initial frame. The original PDA is the zero-backscan suboptimal version.

---

10 Ibid.

The optimal PDA shares the four principal distinguishing characteristics of the zero-backscan PDA approaches. It generates **track-oriented hypotheses**, that is, every observation is considered for association with each track from the previous frame. This is the reason for the absence of organic track initation logic: no observation is considered for association with a track that did not exist on the previous frame, that is, a new target.

Reid's[11] multiple hypothesis tracking algorithm remedies the absence of organic track initiation logic by generating **observation-oriented hypotheses**. Each observation is associated with a false alarm, as a feasible continuation of a previous track, or as a new target, in the following manner.

Start with the hypotheses generated on the previous frame. Consider the first new observation. Generate a new hypothesis *for each possible assignment* of the observation: as a false alarm, as a feasible continuation of a previous track, or as a new target. Take this new set of hypotheses and repeat this procedure with the second observation, except that more than one observation cannot be assigned to one track. Continue in this way until every current observation has been assigned.

For instance, Reid's algorithm applied to Figure 1 would generate eight hypotheses in the single target case and 30 hypotheses in the multiple target case. Reid refers to these as cluster hypotheses. See Table 4.

Table 4. Results of Reid's Algorithm Applied to Figure 1a.

Cluster Hypothesis	Measurement 1	Measurement 2
1	FA	FA
2	T1	FA
3	T2	FA
4	FA	T1
5	T2	T1
6	FA	T3
7	T1	T3
8	T2	T3

Notes: FA = false alarm;
T1 = the original track;
T2 = a possible new track originating with observation 1;
T3 = a possible new track originating with observation 2.

---

11 Donald B. Reid, "An Algorithm for Tracking Multiple Targets," *IEEE Trans. Auto. Control*, Vol. AC-24, No. 6, December 1989, pp. 843-854.

While the total number of cluster hypotheses generated can be quite large, the number of track assignment hypotheses is relatively few. This is important because the number of computations in this approach can be greatly reduced, as we now demonstrate.

Observe from Table 4 that a cluster hypothesis consists of one possible set of track assignment hypotheses. The original track, T1, is either associated with measurement 1 or 2 or with no measurement. New track 2, T2, may or may not be generated by measurement 1 and similarly for new track 3 and measurement 2. This yields seven track-oriented hypotheses. A similar calculation shows that there are ten track-oriented hypotheses in the multiple target case in Figure 1b.

A particular track assignment hypothesis can appear in many different cluster hypotheses. Each track assignment hypothesis is followed by a Kalman filter update computation. If the track update computations were performed for each cluster hypothesis, then the same filter update computation would be repeated many times. Instead, association probabilities are calculated over alternative target assignment hypotheses and then mapped onto the larger set of cluster hypotheses. It must be reemphasized that each association hypothesis assumes unique track assignments so that the association probabilities are calculated over statistically independent states.

The optimal implementation of Reid's algorithm would require ever-increasing computer memory as more hypotheses are generated on each frame. A practical version must limit the number of hypotheses. One method is to divide the set of tracks and observations into independent groups, which Reid calls clusters, requiring conflict resolution. Hypotheses are also limited by pruning and merging. Hypotheses considered unlikely, say those below some threshold, are dropped while those that are "similar" according to some criteria are combined. These operations are suggestive of track splitting but in that case there were no association probabilities and there were multiple assignments of tracks to observations. The Reid algorithm generates individual state estimates that are scored by association probabilities. It is formulated in optimal and suboptimal versions, which can be well implemented. Clustering, pruning, and merging can be adjusted to fit hypothesis growth to track density and throughput and memory computer restrictions.

The major limitation of Reid's algorithm is that it does not include multiple assignments of tracks to observations, such as may occur in merged measurements, or observations to tracks, such as may occur in track spawning. The fundamental reason for this is the manner in which the association probabilities are calculated. One association

hypothesis consists of a set of unique assignments. The probability of the assignment hypothesis, calculated by Bayes rule, when the states are statistically independent, decomposes into the products of probabilities for the individual tracks. Unique assignments under each hypothesis ensure the statistical independence of the states.

Kovacich of Lockheed Missiles and Space Company recently described a Bayesian multiple hypothesis tracking algorithm that remedies the defects in Reid's approach to produce the most advanced MHT algorithm to date.[12] The key idea is to use a Bayesian network architecture (also known as influence diagrams) to provide a calculus to represent and manipulate joint probability distributions such as those that occur in multiple target tracking. Rather than decompose the association-to-track problem into assignment hypotheses, the fundamental unit in Lockheed's approach is the scene which is defined as the joint set of observation-oriented hypotheses, track-oriented hypotheses, and track spawning outcomes for different clusters. The probability for each individual possible outcome is calculated by the Bayesian network.

D. GROUP/CLUSTER TRACKING

Thus far the discussion has been limited to tracking of individual objects. In Section 2.1 of this chapter we pointed out that one approach to managing the high density SDI threat is to forego tracking individual objects and instead track groups. In this section we will describe the issues and methods of such an approach.

1. Definitions

To begin, we need to define what we mean by group and cluster. In the previous section the term clustering referred to collecting interacting observations and tracks, that is, track clusters. In the discussion of group tracking, the term cluster is defined differently and is used in reference to nearby objects, that is, target clusters. Before stating the definition, a comment on what is to be included. Consider two simple examples that mark the extremes: (1) a long line of equally spaced objects; (2) a sphere of objects. In the second example, each object is within some radius of the center of the sphere. In contrast, each object in the first example is within some metric distance of at least one other object.

---

[12] Michael Kovacich, "Application of Bayesian Networks to Midcourse Multi-Target Tracking," presentation to the SDI Panel on Advanced Concepts in Tracking, *Proceedings of the SDI Panels on Tracking*, No. 4, 1989.

Another issue that often generates confusion is whether the objects in the collection are based on truth or measurement data.

We will follow the lead of the SDI Panels on Tracking in our definitions.[13]

Cluster:	A maximal collection of objects each of which is within some metric distance of at least one other object in the collection.
Group:	To be used at the author's discretion but to include a cluster.
Group track:	Track established to represent a cluster of objects.
Clump:	A single observation arising from two or more objects.

2. Types of Group Tracking

Following Drummond[14] we identify individual and group tracking as the endpoints of a spectrum:

- Group tracking without individual target tracks
- Group tracking with simple individual target tracks
- Individual target tracks supplemented with simple group information
- Individual target tracking without group tracks.

This ordering is suggestive of a logical sequence of operations that might occur in midcourse tracking. During deployment of reentry vehicles (RVs) and decoys from post-boost vehicles (PBVs), the threat initially consists of closely spaced objects. The objects may resolve with increasing time from deployment as the threat cloud disperses.[15] As the sensors move along their orbits, however, the resolution of objects is a function of the sensor resolution and the viewing geometry and range. For this reason, the threat could be resolving or unresolving during the course of the sensor's observations.

These considerations lead to the conclusion demonstrated below that the information processing in group tracking algorithms must begin by deciding, based on data, computational, communication, and operational considerations what combination of group and individual target tracking to execute.

---

13 *Proceedings of the SDI Panels on Tracking.*

14 Oliver E. Drummond, Hughes Aircraft Company, presentation to the SDIO Panels on Tracking, *Proceedings of the SDI Panels on Tracking.*

15 This is not to suggest that the threat density will not or cannot be increased later in the flight.

It is not possible to establish individual object tracks on members of an unresolved CSO. Extended object tracking must be executed. Only when the objects in a CSO are resolved is individual object tracking possible. The density of observations and the high rate that new objects are resolved, however, may compel group tracking as the only practical alternative because of great computational expenses in track initiation and misassociation. If objects unresolve, the tracking architecture must extrapolate individual object tracks or establish group or extended object tracks.

There are critical operational requirements for maintaining tracks on individual targets, including discrimination of RVs from decoys, threat assessment, and attack execution. Group tracking is performed when individual object tracking is impossible or too expensive. As the threat resolves, group tracks spawn individual object tracks, that is, individual object tracks are initialized by the group track. Therefore, group tracking should be evaluated based on its relatively inexpensive computation and communication requirements and the quality and the processing load required of the initial estimates for the spawned individual object tracks.

3. Single and Multiple Sensor Group Tracking

Blackman[16] describes a single sensor group tracking algorithm that tracks the group centroid position and velocity. A gating logic that is a generalization of the gate for an individual object track determines which observations will be considered for updating the group track. A conflict resolution logic is required for all observations that satisfy multiple group track gates. All observations assigned to a group track are used to form a group observation consisting of a measurement centroid and dispersion ellipse. The measurement centroid updates the group centroid state in the standard manner of Kalman filtering. Tracks for objects splitting off the group are initialized by the group centroid state.

Drummond, Blackman, and Hell,[17] have extended these ideas to multiple sensor group tracking. The principal difficulty in multiple sensor group tracking is that the size, shape, and composition of the group varies from sensor to sensor. For this reason, multiple sensor group tracking must have more information than just the location of the

---

16 Samuel S. Blackman, *Multiple-Target Tracking with Radar Applications*, Artech House, Inc., 1986, Chapter 11.

17 O.E. Drummond, S.S. Blackman, K.C. Hell, "Multiple Sensor Tracking of Clusters and Extended Objects," *Technical Proceedings 1988 Tri-Service Data Fusion Symposium*, Laurel, Maryland, May 1988.

group. Drummond et al.'s approach is to model the group as an ellipsoid in three dimensions. Separate filters are established for the group centroid and the ellipsoid extent parameters.

The group centroid state estimate initializes tracks for objects that split away from the group, as before. The ellipsoid extent state estimate permits sensors in different locations to associate groups and facilitates handing over group data to other sensor systems.

E. CONCLUSIONS

We have reviewed the major difficulties and methods in SDI tracking. We have chosen to organize the discussion around the critical problem of uncertain measurement origin that arises during the association of observations to tracks. This is sometimes referred to as the frame-to-frame association problem. We have not discussed other types of association, such as observation-to-observation and track-to-track, that are typically performed in sensor fusion, even though real-world SDI tracking algorithms involve multiple sensors. We have said little about filtering beyond defining it and developing its relationship with observation-to-track association.

There are many components in a complete tracking system. Before tracks can be maintained by association of validated observations and updating they must be initiated. Track initiation refers to the formation of the first or initial estimate of the state of an object. We have found that, for the nonexpert, following the information flow among these many components is often one of the most significant impediments to understanding particular tracking algorithms. For this reason, the next chapter presents an overview of tracking algorithm architectures developed by the SDI Panels on Tracking.

III. OVERVIEW OF TRACKING ALGORITHM ARCHITECTURES

The SDI Panels on Tracking have standardized on a small set of algorithm architectures as high-level descriptions of the logical flow of information in tracking algorithms. It is expected that most algorithms can be decomposed in terms of these algorithm architectures, first identified by Drummond.[18].

A natural taxonomy of multiple sensor, multiple target tracking algorithms is organized by dividing the problem into track initiation and track maintenance algorithm architectures for individual objects and clusters. First consider the algorithm architectures for individual object track maintenance contained in Figures 2-5.

Figure 2 establishes the basic functions of single sensor track maintenance algorithms. Figure 3 represents an architecture in which individual sensor tracks are fused together. Instead of combining tracks, a frame's worth of measurements from multiple sensors could be combined before being filtered, as in Figure 4. Figure 5 differs from Figure 3 in that the individual sensors no longer maintain individual tracks. The system uses only central track files.

Figures 6-9 depict individual object track initiation architectures. Figure 6 establishes the basic functions of single sensor track initiation algorithms. Figures 7, 8, and 9 are similar to Figures 3, 4, and 5, demonstrating that tracks or one frame's worth of measurements could be combined and that the system could use individual sensor tracks or centralized tracks.

Cluster tracking algorithm architectures involve more than the basic individual object functions. The data emerging from the signal processor and requirements from the battle manager determine the type of tracking to be performed. This is depicted in Figure 10. Figures 11-18 are essentially equivalent to Figures 2-9.

---

18 O.E. Drummond, "Multiple Target Tracking Lecture Notes," UCLA October 1985, Revised 18 March 1988, Technology Training Corporation, Torrance, CA.

Figures 2-18 establish the functions that make up tracking algorithms. The descriptions in the figures use terminology that is indicative of but not specific to techniques in particular architectures. This was done in order to remain at a sufficiently high level of generality so that these architectures were applicable to most algorithms. Hence, the terms association, filter, and track were used in their most general sense.

There are two major areas of detail lacking in these figures. First, specific definitions and descriptions of techniques for each function of the process. Second, the distribution and location of computer resources for carrying them out.

The SDI Panels on Tracking have moved to describe some of the specific techniques, which are the fundamental tracking algorithms, most of which are described in Chapter 2. For instance, Figure 19 summarizes track maintenance functions and Figure 20 the elements essential to the description of measurement-to-track association. Finally, Figure 21 indicates specific algorithms for this purpose. Figures 22 and 23 repeat this for track initiation.

The overview of methods and tracking architectures presented in the last two chapters should prepare the nonexpert reader for the algorithm survey in the remainder of this report.

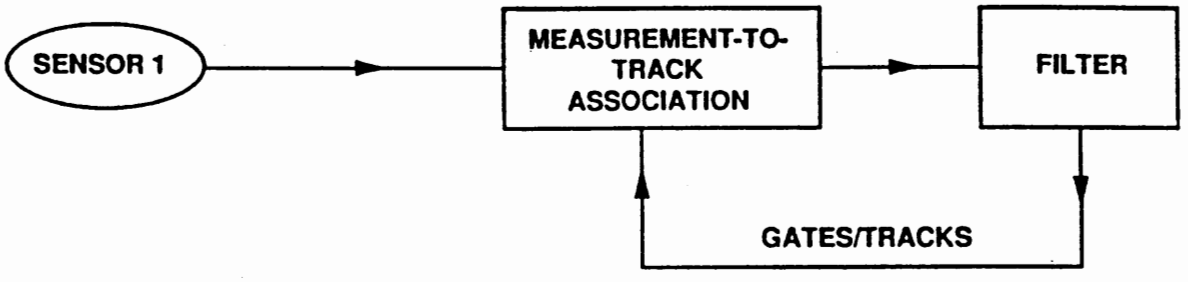


Figure 2. Single Sensor Track Maintenance (Type I)

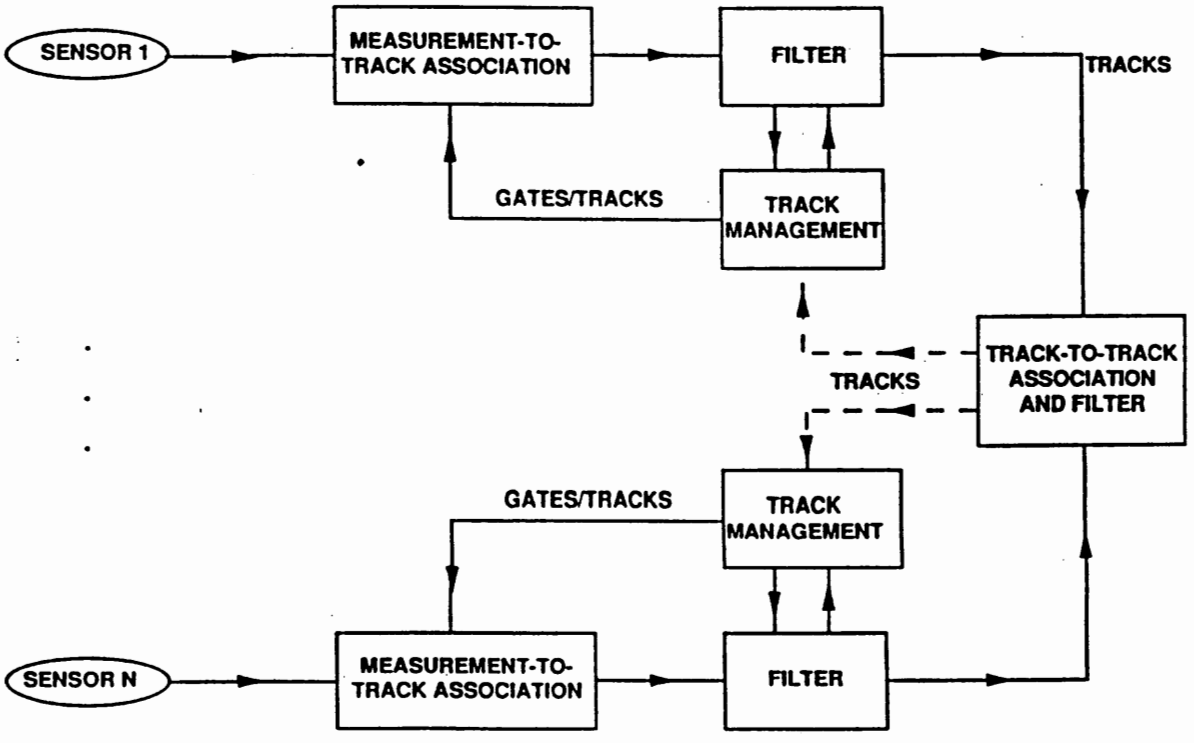


Figure 3. Single Sensor Then Fuse Track Maintenance (Type II)

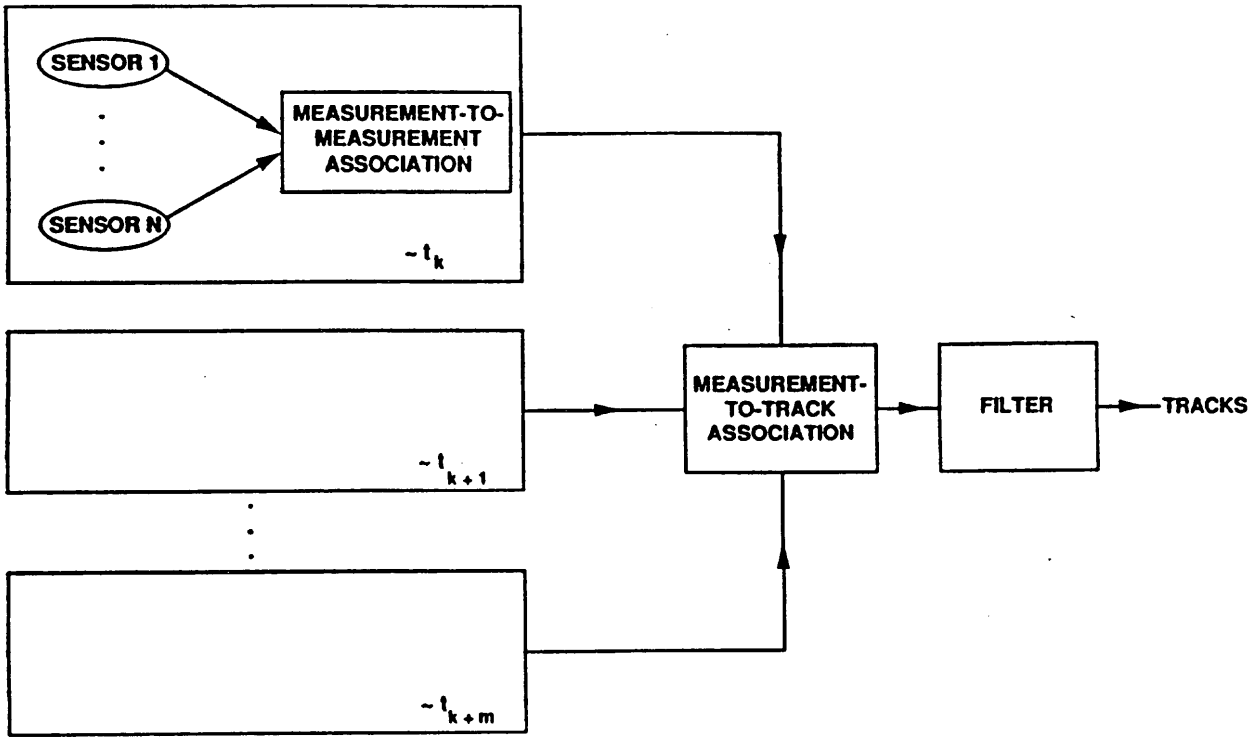


Figure 4. Sensor-to-Sensor Then Scan-to-Scan Track Maintenance (Type III)

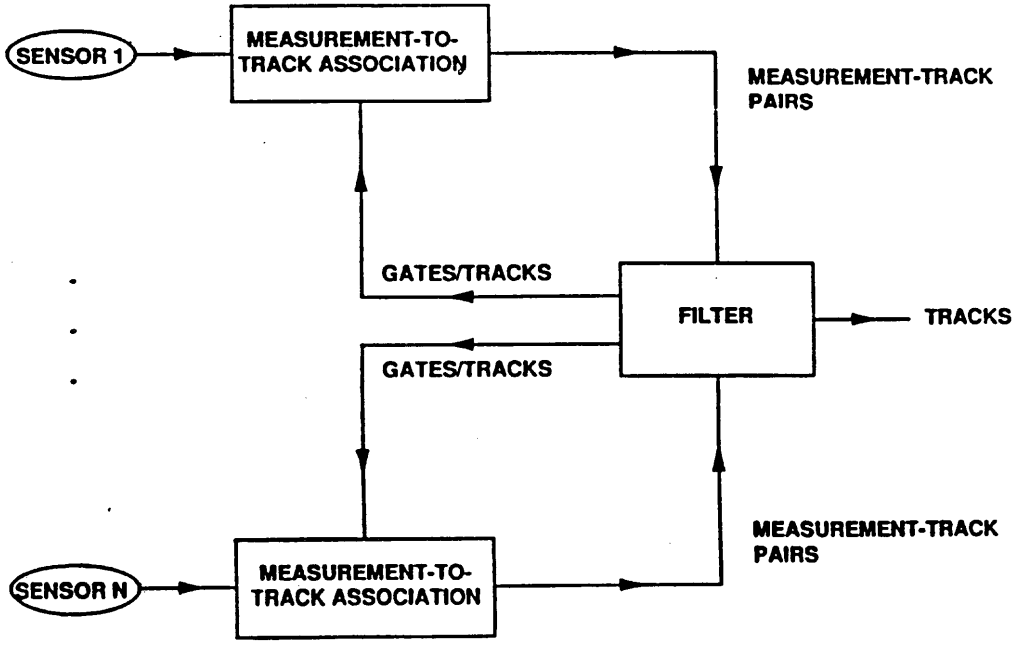


Figure 5. Centralized Sensor-by-Sensor Track Maintenance (Type IV)

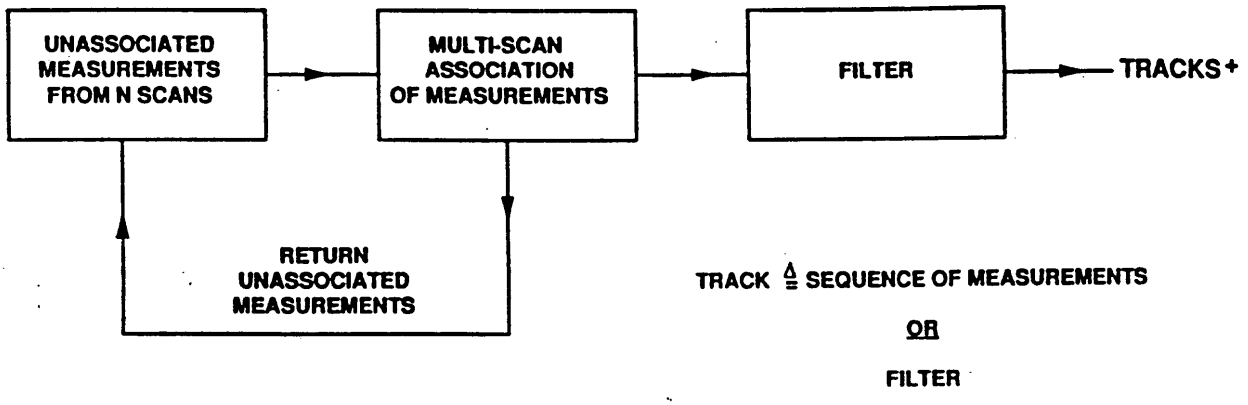


Figure 6. (Cold Start) Single Sensor Track Initiation (Type I)

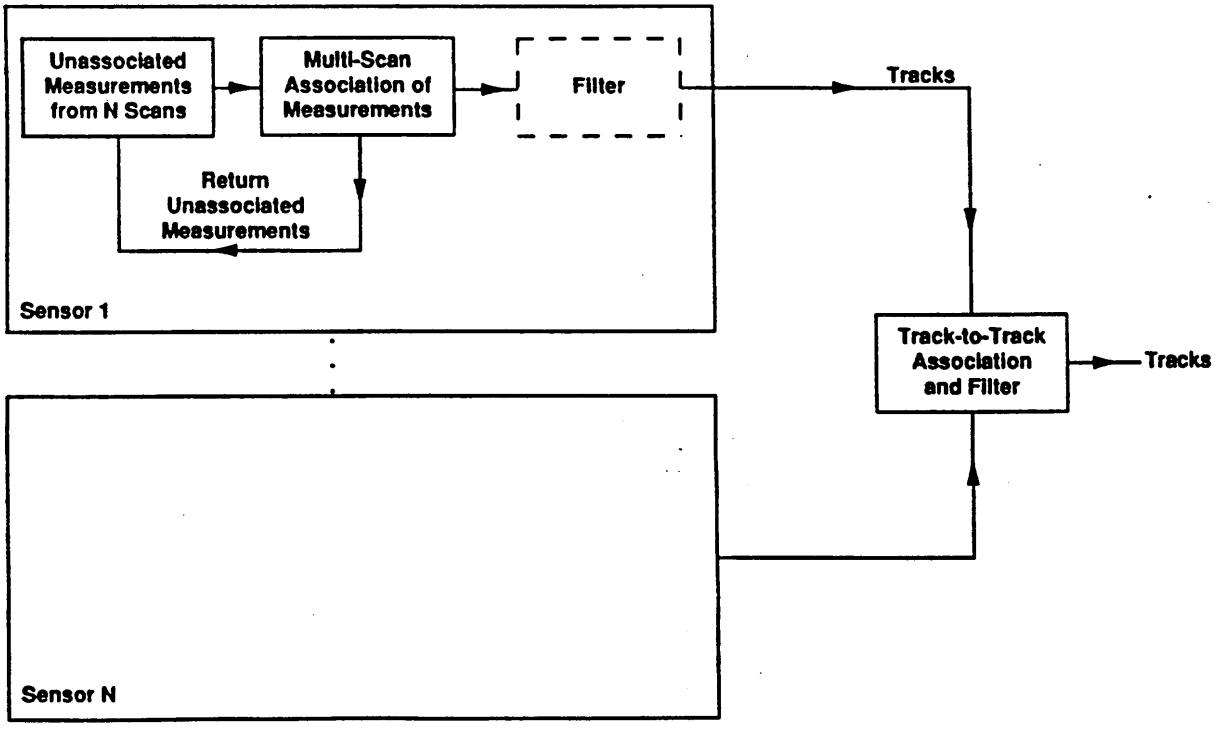


Figure 7. (Cold Start) Scan-to-Scan then Sensor-to-Sensor Track Initiation (Type II)

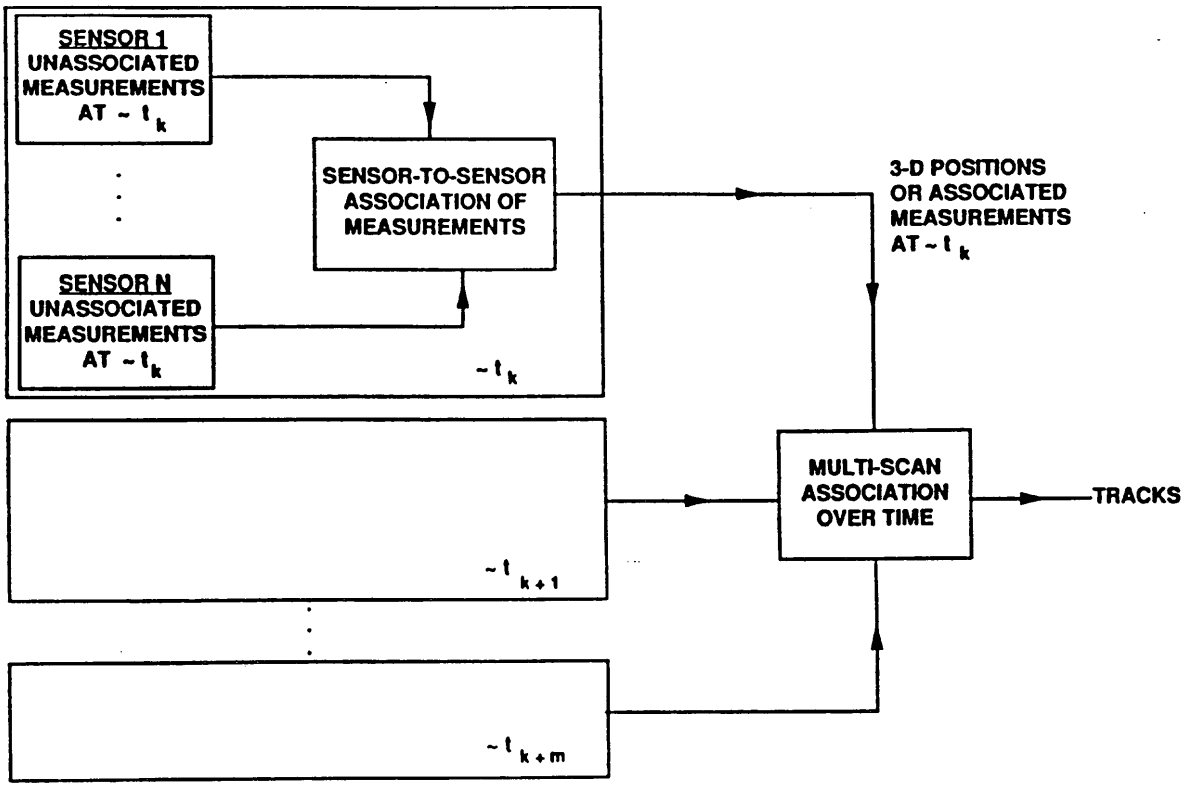


Figure 8. (Cold Start) Sensor-to-Sensor Then Scan-to-Scan Track Initiation (Type III)

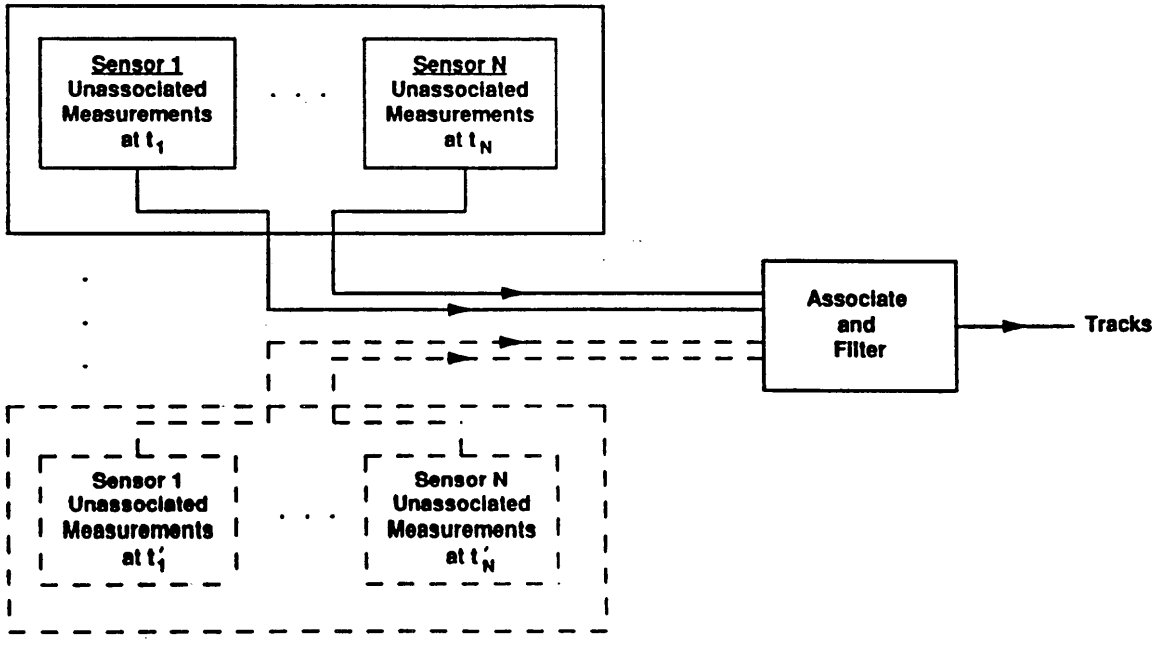


Figure 9. (Cold Start) Centralized Track Initiation (Type IV)

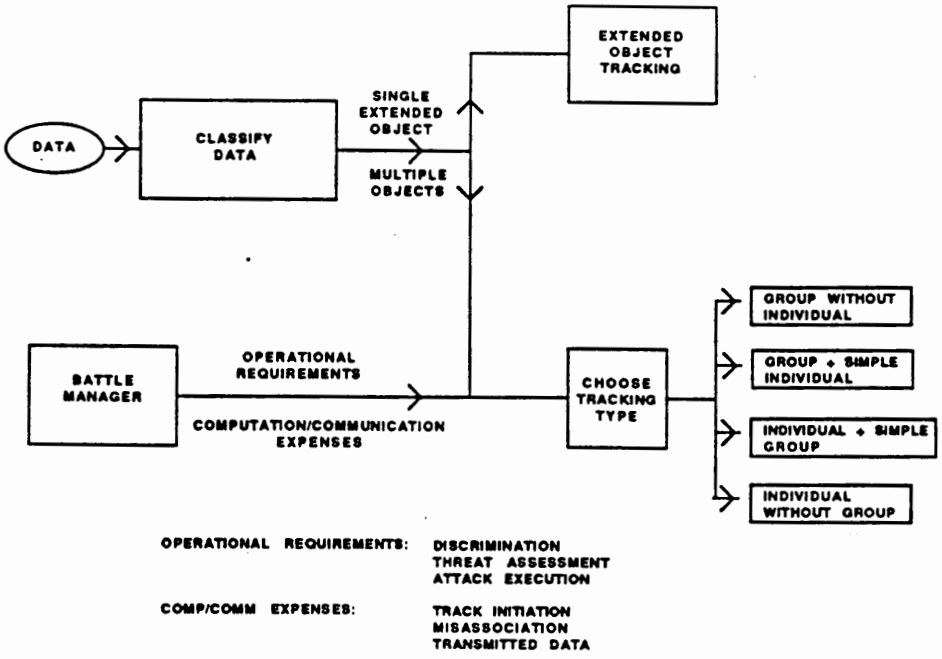


Figure 10. Group Tracking Hierarchy

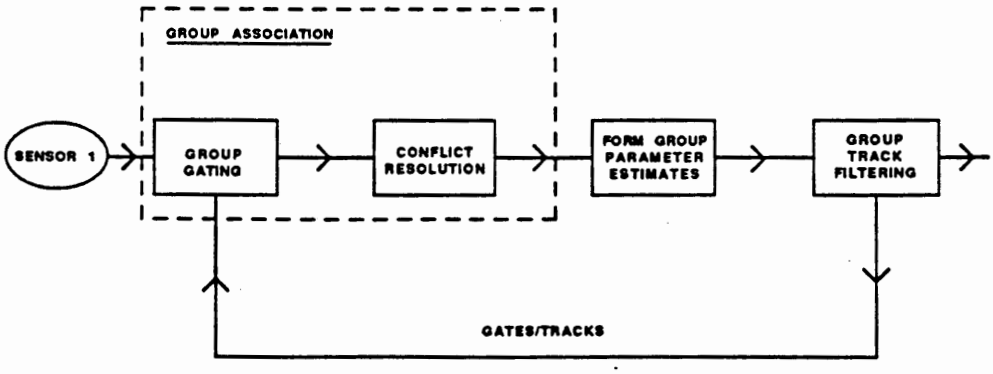


Figure 11. Single Sensor Group Track Maintenance (Type I)

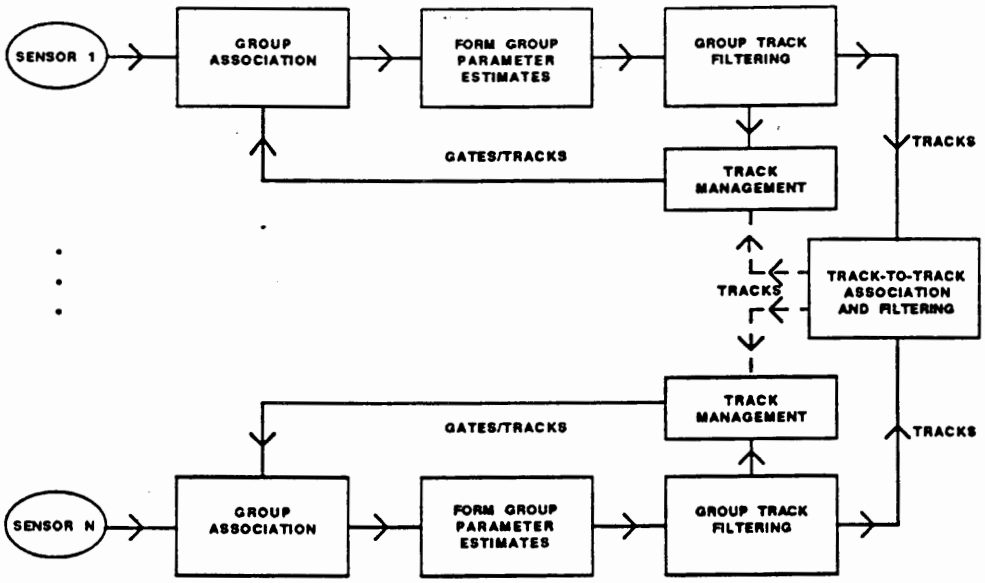


Figure 12. Single Sensor Then Fuse Group Track Maintenance (Type II)

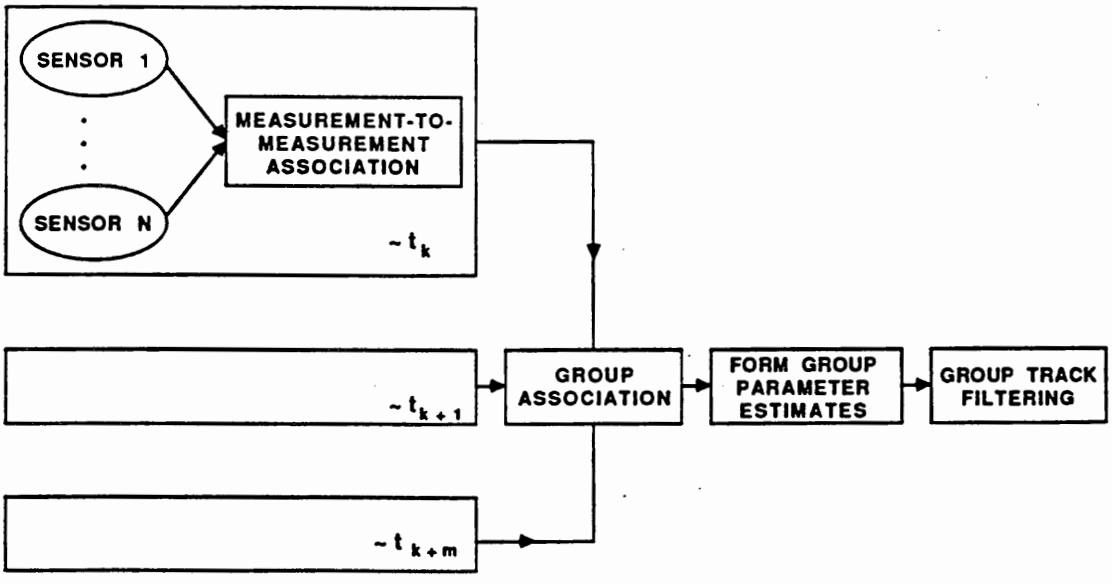


Figure 13. Sensor-to-Sensor Then Scan-to-Scan Group Track Maintenance (Type III)

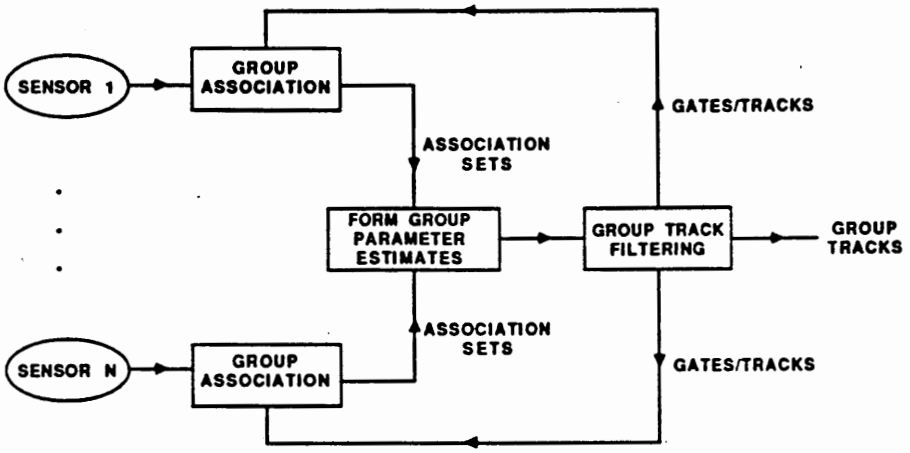


Figure 14. Centralized Sensor-by-Sensor Group Track Maintenance (Type IV)

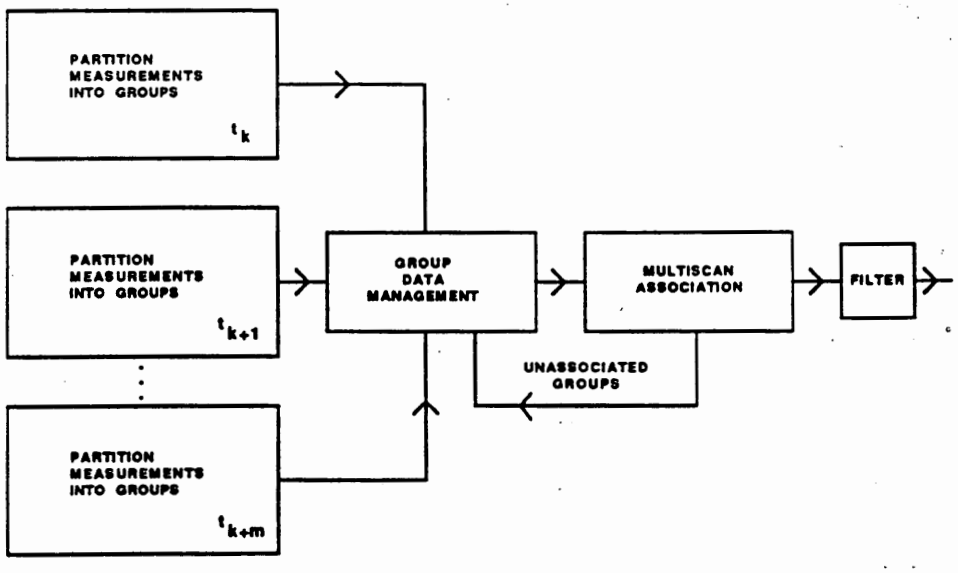


Figure 15. (Cold Start) Single Sensor Group Track Initiation (Type I)

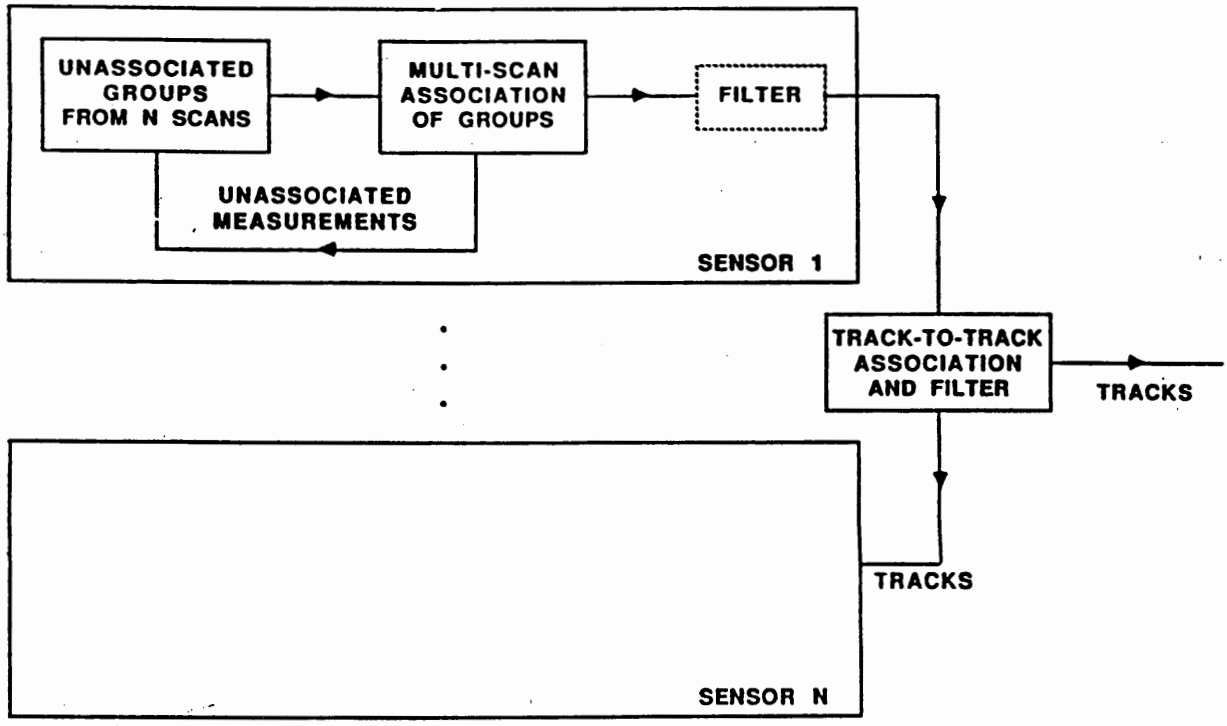


Figure 16. (Cold Start) Scan-to-Scan Then Sensor-to-Sensor Group Track Initiation (Type II)

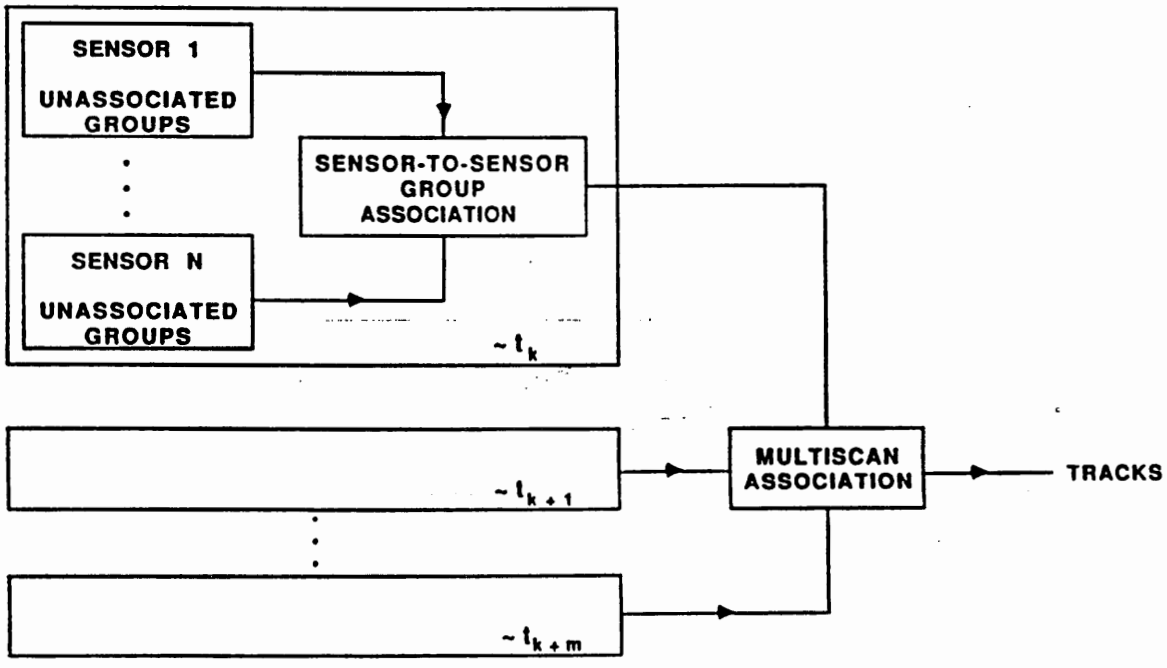


Figure 17. (Cold Start) Sensor-to-Sensor Then Scan-to-Scan Group Track Initiation (Type III)

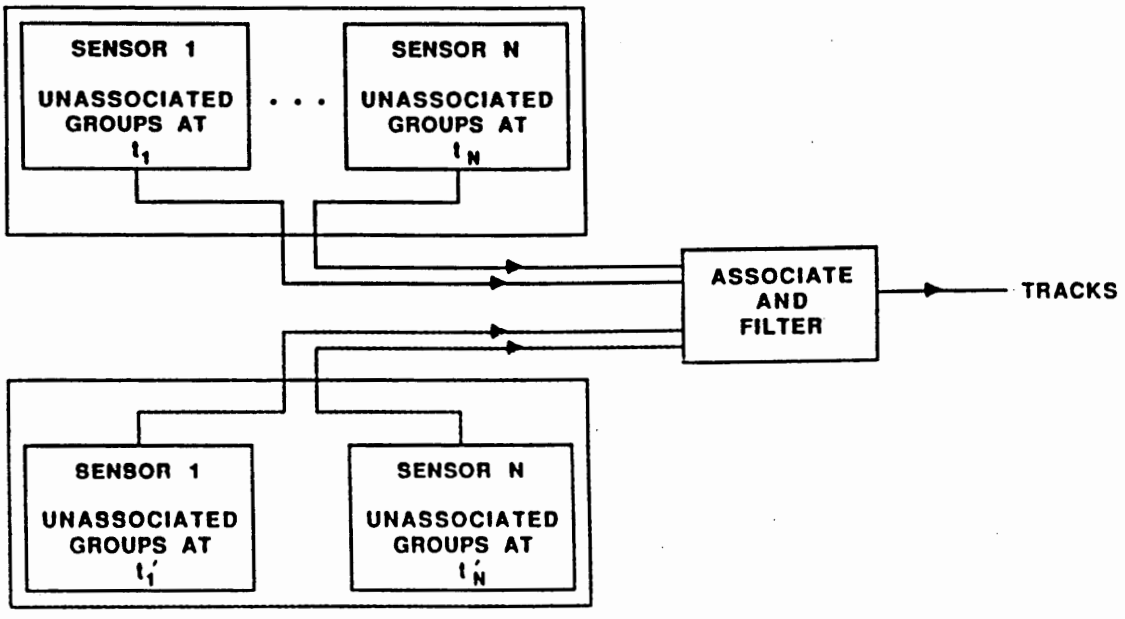


Figure 18. (Cold Start) Centralized Group Track Initiation (Type IV)

- MEASUREMENT-TO-TRACK ASSOCIATION
- FILTERING
- TRACK-TO-TRACK ASSOCIATION AND FILTERING
- MEASUREMENT-TO-MEASUREMENT ASSOCIATION
- TRACK MANAGEMENT

Figure 19. Track Maintenance Functions

- IS THE ASSOCIATION PERFORMED IN AN INDEPENDENT MANNER OR IN A COORDINATED MANNER?
- IS THE ASSOCIATION OVER ONE SCAN OR MULTIPLE SCANS?
- WHAT IS BEING TRACKED?
  - OBJECTS
  - CLUSTERS
- TO WHAT EXTENT IS THE SIGNAL PROCESSING AND TRACKING SEPARATED VERSUS INTEGRATED?

Figure 20. Measurement-to-Track Association Descriptors

- FOR OBJECT TRACKING AND INDEPENDENT ASSOCIATION
  - SINGLE SCAN
    - NEAREST NEIGHBOR
    - PDAF
  - MULTIPLE SCANS
    - SINGER, SEA, HOUSEWRIGHT
    - DEFERRED PDAF
    - SINGLE TARGET MULTIPLE HYPOTHESIS TRACKING, e.g., TRACK BRANCHING
- FOR OBJECT TRACKING AND COORDINATED ASSOCIATION
  - SINGLE SCAN
    - OPTIMAL ASSIGNMENT (e.g., MUNKRES)
    - SUBOPTIMAL ASSIGNMENT (e.g., GREEDY)
    - JPDAF
  - MULTIPLE SCANS
    - REID'S MULTIPLE HYPOTHESIS TRACKING
    - EXTENSIONS OF SINGLE SCAN ASSIGNMENTS
    - DEFERRED JPDAF

Figure 21. Algorithms for Measurement-to-Track Association for Object Track Maintenance

- MULTI-SCAN ASSOCIATION OF MEASUREMENTS
  - SINGLE SENSOR (TYPE I, II) VS. MULTIPLE SENSOR (TYPE IV)
  - SUBJECT OF ASSOCIATION
  - COORDINATION VS\ INDEPENDENT ASSOCIATION
- SENSOR-TO-SENSOR OBSERVATION ASSOCIATION
  - SINGLE SCAN OBSERVATION-TO-OBSERVATION
  - MULTIPLE SCAN POSITION-TO-POSITION
- TRACK INITIATION FILTERING
  - OBSERVATION-TO-2D STATE
  - OBSERVATION-TO-3D STATE (INCLUDES PASSIVE RANGING)
- TRACK-TO-TRACK ASSOCIATION AND FILTER

Figure 22. Functions and Descriptors of (Cold Start) Track Initiation

- SINGLE SENSOR
  - INDEPENDENT
    - NEAREST NEIGHBOR
    - N OUT OF M
    - VELOCITY FILTER
    - TRACK SPLITTING AND PRUNING
  - COORDINATED
    - N-SCAN OPTIMAL ASSIGNMENT (e.g., MOREFIELD)
    - N-SCAN SUBOPTIMAL ASSIGNMENT
    - PATTERN MATCHING
    - REID'S MHT
- MULTIPLE SENSOR
  - COORDINATED
    - N-SCAN OPTIMAL ASSIGNMENT

Figure 23. Algorithms for Multiple Scan Observation Association in Object Track Initiation

IV. SURVEY OF TRACKING ALGORITHMS

This chapter contains a summary of the algorithm survey responses to facilitate the understanding of the responses and to succinctly point out the algorithms' general features, information flow and calculational techniques.

The SDI Panel on Critical Issues in Tracking developed a common survey format for describing tracking algorithms to ensure that key questions were answered in a succinct manner and to simplify the process of understanding the details of the activities. One part of that format was a decomposition of all tracking algorithms into a taxonomy consisting of four track initiation and four track maintenance generic processing chains. It was expected that most responses would conform to the survey format and processing chains. The survey forms and responses are contained in Appendix A and B, respectively.

SUMMARY OF ALGORITHMS

1. **Organization:** Advanced System Architectures Ltd.

   Algorithm: An Object-Oriented Architecture for Sensor Data Fusion/ Tracking in Dense Threat Environments.

   Submitter: Edward J. G. Goodchild

   Description: The algorithm is designed to perform multiple sensor, birth-to-death, three-dimensional tracking of individual objects and clusters. Single sensor scan-to-scan association and track-forming functions are excluded. Unassociated two-dimensional observations are fused across multiple sensors. Tracks are initiated directly into three dimensions from three two-dimensional unassociated observations. New tracks are formed by track spawning of boost phase tracks and are assigned to members of a cluster. New tracks formed by spawning during the post-boost phase are assigned to members of the cluster. Cluster tracks are also formed later by association of a number of non-cluster member tracks, all having near identical trajectories. Cluster tracks determine rectangular gates that are used to partition new observation data. A six-state Kalman filter is used for individual target track state estimation and a second six-state Kalman filter is used to maintain the cluster track trajectory and extent. Track pruning tests implemented and

IV. SURVEY OF TRACKING ALGORITHMS

This chapter contains a summary of the algorithm survey responses to facilitate the understanding of the responses and to succinctly point out the algorithms' general features, information flow and calculational techniques.

The SDI Panel on Critical Issues in Tracking developed a common survey format for describing tracking algorithms to ensure that key questions were answered in a succinct manner and to simplify the process of understanding the details of the activities. One part of that format was a decomposition of all tracking algorithms into a taxonomy consisting of four track initiation and four track maintenance generic processing chains. It was expected that most responses would conform to the survey format and processing chains. The survey forms and responses are contained in Appendix A and B, respectively.

SUMMARY OF ALGORITHMS

1. **Organization:** Advanced System Architectures Ltd.

   Algorithm: An Object-Oriented Architecture for Sensor Data Fusion/ Tracking in Dense Threat Environments.

   Submitter: Edward J. G. Goodchild

   Description: The algorithm is designed to perform multiple sensor, birth-to-death, three-dimensional tracking of individual objects and clusters. Single sensor scan-to-scan association and track-forming functions are excluded. Unassociated two-dimensional observations are fused across multiple sensors. Tracks are initiated directly into three dimensions from three two-dimensional unassociated observations. New tracks are formed by track spawning of boost phase tracks and are assigned to members of a cluster. New tracks formed by spawning during the post-boost phase are assigned to members of the cluster. Cluster tracks are also formed later by association of a number of non-cluster member tracks, all having near identical trajectories. Cluster tracks determine rectangular gates that are used to partition new observation data. A six-state Kalman filter is used for individual target track state estimation and a second six-state Kalman filter is used to maintain the cluster track trajectory and extent. Track pruning tests implemented and

planned include: consistent and sensible track behavior, behavior of the error covariance, and continued updating with new sensor data. The algorithm is in a concept-proving stage.

2. **Organization:** Alphatech

Algorithm: Multiple Information Set Tracking Correlator (MISTC)

Submitter: Robert B. Washburn, Jr.

Description: This work investigated eight different data association algorithms focused on the midcourse problem of multiple sensor individual object track initiation and track maintenance. Tracks are initiated as two-dimensional tracks until multiple sensor associations are made; three-dimensional tracks are then initialized. A single scan of observations from multiple sensors (two or three), containing an assumed 100,000 objects, was partitioned into spatially separated groups assumed to consist of about 100 objects. After partitioning, each data group was processed by the tracker/association algorithm. Of the eight association algorithms, three were zero-scan, pairwise approaches and five were N-scan, multiple hypotheses approaches. The same association algorithm performed scan-to-scan and sensor-to-sensor association. Ten hypotheses were permitted per target, with each hypothesis scored by likelihood ratios. Extended Kalman filters for state estimation assume the targets travel along Keplerian trajectories. The algorithms did not handle track spawning. Algorithms have been implemented for sequential, off-line processing in FORTRAN. Plans include incorporating CSO tracking and resolution into the algorithms and implementing them on different parallel processors.

3. **Organization:** Ball Systems Engineering (VERAC, Incorporated) and Daniel H. Wagner, Associates

Algorithm: SDI Midcourse Tracker/Correlator Algorithm

Submitter: Larry Filippelli

Description: This single sensor, midcourse individual object tracking algorithm does not perform cold start track initiation. Instead, warm start track initiation of a six-state Kalman filter is accomplished by handover of boost phase information. The algorithm relies on the assumption that objects tend to cluster into spatially inseparable groups to reduce the combinatorial explosion that results from high target density. Clutter is assumed removed by the signal processor and not passed to the information processor.

Multiple observation-to-track association hypotheses are formed by track splitting and scored using a Bayesian approach that takes into account probability of detection, probability of false alarm, false and new target densities, and observation-to-track association scores. Hypotheses are deleted if their score is below a percentage of the score of the best hypothesis. There is also a maximum number of hypotheses that can be saved. A second-generation algorithm has been completed and is in the testing phase. The algorithm has been installed at NRL and LANL.

4. **Organization:** CALTECH Jet Propulsion Laboratory

Algorithm: CALTRAX: The Tracking Program for Simulation-88

Submitter: James Ortolf (Applied Research Associates)

Developer: Thomas D. Gottschalk

Description: This algorithm tracks individual objects in the boost and post-boost phases from multiple sensors. Single-sensor, two-dimensional tracks are initiated by a three-scan batch processor. Two-dimensional tracks are maintained by a four-state Kalman filter and track splitting. Tracks are deleted on the occurrence of a single empty gate (probability of detection is assumed equal to one) and merged if of common history. Mature tracks are propagated to a common time and exchanged with a stereo partner sensor to determine the three-dimensional state vector in earth-centered inertial coordinates. The two sets of tracks are associated by a modified nearest neighbor algorithm to initiate three-dimensional tracks, which are used to solve for launch parameters according to a powered flight model. Once initialized, launch parameters are updated on subsequent scans by means of extended Kalman filters. Individual sensor observations are associated with three-dimensional tracks by a global modified nearest-neighbor algorithm. Any observations unassociated with a three-dimensional track on one sensor, if part of a mature two-dimensional track, are associated with those on the stereo partner sensor to initiate new three-dimensional tracks. The design of the three-dimensional tracker is based entirely on parameterized trajectories, with all updates of existing tracks done using extended Kalman filters for assumed trajectory models. The next generation tracker will perform parameter estimations on arbitrary flight models.

5. **Organization:** ESL

Algorithm: Tracking Algorithm for Project Swat

Submitter: Jack Liu

Description: The algorithm, produced to evaluate the applicability of DARPA's MOSAIC architecture developed by ESL, performs single platform tracking of individual objects during midcourse. The survey response provides no specific information on track initiation procedures. Track maintenance is accomplished by a single target, multiple hypotheses scheme: nearest neighbor observation-to-track association in a rectangular gate updates the track; remaining observations in the gate are used to split new tracks; and detections within overlapping gates are utilized by all affected tracks. Tracks are scored by the log-likelihood function of the detection relative to the prediction plus maintenance bias. The scores are cumulative. When a set track redundancy is reached, the low score tracks are deleted. Tracks that are very close in terms of estimated object state and uncertainty are merged in a probabilistic fashion based on their track scores. A six-state extended Kalman filter is the track estimator with an earth gravity model selectable up to J6. Multiple platform tracking will be addressed in the near future.

6. **Organization:** Hughes Aircraft Company

Algorithm: Multiple Sensor Cluster Tracking

Submitter: Oliver E. Drummond

Developers: Oliver E. Drummond and Samuel S. Blackman

Description: This algorithm, for use in the early midcourse phase, accomplishes multiple sensor tracking of multiple clusters. Cluster tracks can be used to initiate individual object tracks as the closely spaced objects become resolved, by track spawning based on the estimated PBV track. This permits a smooth transition from PBV cluster deployment to individual target tracking. The filtering segment of the algorithm estimates the cluster centroid position and velocity in inertial space and the cluster extent, the second central moment in inertial space, of the objects in the group. The extent, which establishes the cluster size and shape in inertial space, not just relative to a particular sensor, is used to determine which observation belongs to which group. Based on the projection of the predicted extent on to the field of view of a sensor, a gate is computed for a cluster. The filtering is composed of two filters, one for the state of the centroid and

another for the extent. The filter for the 6-D centroid state is a simplified extended Kalman filter; the extent state has 6 elements and the filter is a pseudo-linear filter. Feasibility tests have been conducted successfully for cluster tracking under realistic conditions and further testing is under way.

7. Organization: Lockheed Missiles and Space Company

Algorithm: SSTS Tracking and Association Algorithm

Submitter: Michael Kovacich

Description: This algorithm accomplishes multiple sensor, birth-to-death, group-to-object, midcourse tracking of groups, clumps, and objects in the presence of clutter, including stars, nuclear redout, and a structured background. Initial coarse three-dimensional track estimates are produced by each sensor after four-to-six updates using an iterated maximum likelihood passive ranging algorithm. Precision ECI tracks are initiated by multiple sensor triangulation. The track initiation process completes with the formation of precision tracks. Data association and track maintenance are accomplished by a multiple hypotheses approach known as a Bayesian network architecture. Pruning, merging, and clustering are used to control the combinatoric explosion. Hypothesis scoring accounts for clutter density, new track density, missed detections, and cumulative chi-squares. Thresholds based on a fraction of the best track score are used for track promotion and deletion. A variant of the A* search algorithm is used to find likely hypotheses. Tracks are merged and the covariance matrix adjusted accordingly, in a manner similar to PDAF. The Bayesian network approach includes the multiple assignment of observations to tracks and the multiple assignment of tracks to observations. An extended Kalman filter performs track estimation. The algorithm is currently being implemented in ADA in preparation for a 1989 demonstration.

8. Organization: M.I.T. Lincoln Laboratory

Algorithm: A Mid-Course Track Initiation and Maintenance Algorithm

Submitter: Ming J. Tsai

Developers: M.J. Tsai, K.P. Dunn, L.C. Youens, and C.B. Chang

Description: This midcourse, individual object and cluster track initiation and track maintenance algorithm accomplishes single sensor track initiation of clusters by forming track files for edges of clusters. Tracks on cluster members are then initiated by

assuming that targets within the same cluster travel in parallel. Tracks initiated by two sensors are merged and track maintenance performed in a sensor-by-sensor centralized fashion. Tracks are maintained by an extended Kalman filter and nearest neighbor observation-to-track association. The algorithm has been implemented, tested, and run in a number of simulated threat/sensor scenarios. Currently, it is being integrated with discrimination algorithms and radar tracking functions.

9. **Organization:** McDonnell Douglas Space Systems Company

Algorithm: Integrated Correlation, Track

Submitter and Developer: Thomas R. Blackburn

Description: This algorithm was developed for cold start track initiation and track maintenance of individual objects in the late midcourse phase of their trajectory, when objects are resolvable but not undergoing the influence of the atmosphere. Two sensor, sensor-to-sensor, observation association is performed before track initiation in order to initiate three-dimensional tracks. Sensor-to-sensor association is performed by matching the pseudo-elevation angle generated from interpolated line-of-sight measurements taken from two frames of data. The track file is initiated with a square-root information filter. A nearest-neighbor observation-to-track association is used in track maintenance to provide measurement updates to a six-state Kalman filter. The algorithm is in the late conceptual development stage and has been tested and debugged running against threats consisting of about 600 objects.

10. **Organization:** MindGate Technologies, Inc.

Algorithm: Cluster Map Tracking

Submitter and Developer: Lawrence M. Beyl

Description: The crux of this midcourse tracking algorithm is the supposition that there are patterns within a threat that are naturally formed by the objects dispersed from the same PBV and that are heading toward the same target. A collection of co-moving objects is termed a cluster. The pattern of angle measurements of objects in a cluster can be traced from one scan to another by using the previous scan's two-dimensional map of the cluster as a pattern for the next scan's data associations. Thus, this is a pattern-matching algorithm. The interlocking of the cluster to its source data permits each map to be used as a filter to remove the cluster's new measurements from the field-of-view for the sensor

with the extracted data set then used to replace the existing set as the new cluster map. Each time a cluster map is updated, the associated cluster centroid state is updated via a six-state extended Kalman filtering, where the data used in the update is a calculated pseudo-measurement created from the collection of angle measurements that define the cluster map. The transition to individual object tracking is accomplished by forming an initial state vector and covariance for each object within the cluster through the centroid state, the angular separation of the objects within the cluster from the centroid, the cluster spread and spread rate, and the angular measurement accuracy. Splitting, merging, and other phenomena are handled within the individual object tracking environment, but are always restricted to the domain of the cluster. A cluster's centroid state can be estimated initially based on the PBV state at the time of deployment from boost phase information.

11. Organization: MITRE Corporation, Bedford

Algorithm: The MITRE Experimental Version Prototype (EVP) Ballistic Tracker

Submitter and Developer: J. A. Krajewski

Description: This algorithm performs individual object tracking during post-boost and midcourse using sensor-by-sensor centralized track maintenance. There is no cold start track initiation capability. Instead, the algorithm relies on warm start track initiation from handover of boost-phase tracks and spawning of RV/decoy tracks (assumed distinguishable from PBV tracks). The data are partitioned based on a user defined angular distance threshold and a distance matrix calculated for each group. Hungarian and Greedy-type algorithms are used for observation-to-track association. For each association pair, the distance is tested against a threshold and, when greater, the association is suppressed. A six-state extended Kalman filter is used to estimate the Keplerian three-dimensional ballistic tracks. The state is updated using two-dimensional observational data from one SSTS satellite at a time. All algorithms have been coded in FORTRAN and are being written in Ada. Test cases are currently being run and analyzed.

12. Organization: MITRE Corporation, Bedford

Algorithm: The MITRE Experimental Version Prototype (EVP) Boost-Phase Tracker.

Submitter and Developer: J. H. Latimer

Description: This is a boost phase, individual object, two sensor, track initiation and track maintenance algorithm that develops three-dimensional position estimates by associating observations from the asynchronous sensors before initiating and updating tracks. Cold start track initiation is performed from a single three-dimensional position estimate and the *a priori* assumptions of a three-degree-of-freedom, reference rocket trajectory model. A Newton's method iterative procedure is used to fit the reference trajectory to the three-dimensional target position. The value of the reference trajectory velocity and acceleration at the point of the fit is used to initialize the track state estimate. The data from two sensors are paired by mapping the data from one sensor onto the focal plane of the other. The position estimate-to-track association is performed by a Greedy-type algorithm. Tracks are estimated by a nine-state Kalman filter for the position, velocity, and acceleration in Cartesian earth-centered inertial coordinates. The algorithm does not handle stars, false alarms, or other forms of stationary clutter. All algorithms have been coded in FORTRAN and are being written in Ada. Test cases are currently being run and analyzed.

13. Organization: MITRE Corporation, Bedford

Algorithm: The MITRE Multi-Sensor, Multiple Target Tracker.

Submitters and Developers: R. Varad and J. T. McKernan

Description: This is a boost phase, individual object, three sensor, track initiation and track maintenance algorithm that associates two sets of stereo-associated data. Three sensors are divided into two pairs and observations are associated for each pair. Then the two pairs of associated observations are fused. Range is determined from the common sensor once target lists from each pair are formed and then associated based on hinge angles, in-plane angles, and estimates of the baseline ranges determined from each pair. A track is initialized when association can be obtained in hinge angle, in-plane angle and range for data from two consecutive scans. After initialization, rates for hinge angle, in-plane angle, and range are calculated for each track and predictions of target coordinates for the next scan are made. The track state consists of position and velocity estimates in three dimensions. The algorithm is fully designed, developed, and implemented in Pascal on a VAX/VMS system.

14. Organization: Raytheon Company

Algorithm: BMEWS Phased Array Radar Upgrade

Submitter: Fred Daum

Description: This boost and midcourse single-sensor, cold-start track initiation and track maintenance algorithm processes phased array radar measurements of range, elevation, azimuth and target amplitude to estimate position and velocity vectors in a six-state Kalman filter. A multiple hypothesis track initiation scheme is used with pulse-pair track initiation and track maintenance. The observation-to-track association is performed by a nearest neighbor chi-square test.

15. Organization: Raytheon Company

Algorithm: Ground-Based Radar (GBR)

Submitter: Fred Daum

Description: This midcourse single-sensor cold or warm start track initiation and track maintenance algorithm processes phased array radar measurements of range, elevation, azimuth, target amplitude, and phase to estimate position, velocity, and higher order rotational dynamics (for discrimination). A multiple hypothesis track initiation scheme is used with pulse-pair track initiation and track maintenance. The observation-to-track association is performed by a nearest neighbor chi-square test. Three Kalman filters are maintained: six state, seven state, and sixteen state.

16. Organization: Space Computer Corporation

Algorithm: Velocity Filter Algorithm for SDI Detection and Tracking

Submitter: William J. Jacobi

Description: Both boost and midcourse applications of this cold start track initiation algorithm have been investigated. Operating with either resolved or unresolved objects, the velocity algorithm performs a combination of signal-to-noise enhancement and scan-to-scan association functions utilizing a "track-before-detect" approach. A bank of filters matched to different vector velocities provides correlated object positions and velocities for track initiation. The vector velocities to which the filters are tuned are derived from cross-correlation of successive input image frames. When the data contains "velocity clusters," the algorithm is inherently robust against object proliferation, merging/crossing tracks, background clutter, and temporary loss of data.

17. **Organization:** Systems Control Technology

Algorithm: Dynamic Programming Algorithm (DPA)

Submitter: Kenneth Kessler

Developer: Yair Barniv

Description: Another "track-before-detect" approach, the dynamic programming algorithm is a practical and feasible alternative to replace exhaustive search techniques for detecting and locating entire target trajectories inside a sensor's field of view over some time interval. This is accomplished by batch processing data over a small number of frames through a bank of matched filters, where each filter represents a single possible two-dimensional trajectory. The algorithm produces simultaneous detection and two-dimensional tracking of targets because its output consists of detected targets and their associated hit strings. The analysis and software development has been ongoing for over seven years.

18. **Organization:** TITAN Systems

Algorithm: Knowledge-Based Sensor Fusion (KBSF)

Submitter: Timothy E. Brockwell

Description: This is a tracking algorithm to the extent that it is not entirely possible to decouple tracking from discrimination. The program was initiated to determine whether rule-based techniques could be applied to strategic sensor fusion, specifically to demonstrate a rule-based approach to multi-sensor discrimination. The goal is to host and evaluate competing algorithms, primarily those designed for platform-to-platform association and track maintenance, by building a machine that automatically selects the "best" algorithm for fusing multiple or single platform, multiple sensor track data. Experiments are being conducted with fuzzy techniques for determining "degrees of membership" in the set of valid tracks that is maintained by the system, and for uncertainty management in general. Currently, a variation on the Sequential Probability Ratio Test for determining "degrees of membership" in the set of valid tracks is employed.

19. **Organization:** TRW, Huntsville

Algorithm: ADOP Scan-to-Scan and Track Algorithm Set

Submitter: J. T. Lawson

Description: This is a midcourse, single sensor, individual object and cluster cold start track initiation and track maintenance algorithm that uses pattern matching in the data association and derives object velocity estimates from three-color correlation processing. In the track initiation phase, association is performed over six frames. The algorithm partitions the observations in a scan according to a distance separation threshold. For each partitioned set, the group centroid and group velocity are calculated, as are the predicted centroid for the next scan, a centroid gate, and a group azimuth-elevation extent gate. The grouped observations are associated by pattern matching between scan 1 and scan 2 and between scan 2 and scan 3; thereafter the association is performed in an independent, nearest-neighbor manner. Objects are assumed to follow non-maneuvering Keplerian ballistic trajectories. During the angle-only phase of tracking, a cubic-fit least squares filter in azimuth and elevation is used. A precision track mode uses a six-state Kalman filter. An iterative batch filter is used to transition between the modes: an initial state estimate is obtained by the Gauss algorithm for orbit determination given three angle-only sightings; the batch filter is then applied over the previous observation history to start the Kalman filter. A track is declared lost and no longer maintained after two consecutive empty gates. All clutter is assumed removed by the signal processor and not passed to the tracker. Versions of the algorithm's software (Pascal) are available at the USASDC Advanced Research Center for both the Honeywell and DEC VAX configurations. No significant improvements have been made since 1985.

APPENDIX A

ALGORITHM SURVEY FORM

TRACKING ALGORITHM SUMMARY

Submitter's Name: _____

Submitter's Company/Organization: _____

Submitter's Phone: _____

Address of Submitter: _____

_____

Author's Company/Organization: _____

Author's Phone: _____

Address of Author: _____

_____

TITLE OF ALGORITHM:

SPONSOR:

DEVELOPER:

_____

(Complete as appropriate)

NOTES AND INSTRUCTIONS

(The Summary/Abstract of the algorithm should be limited to half a page. The basic answers to the questions below should be limited to three pages. Additional information, referenced to the section numbers, should be included in a separate appendix that should be limited to four pages. Classified information should be included in a separate supplement. The total information including Abstract/Summary, basic answers, appendix, and classified supplement should not exceed eight pages. If some of the algorithms or details are proprietary, indicate what is proprietary and discuss only the nonproprietary aspects.)

ABSTRACT/SUMMARY

(Describe, in general terms, how the algorithm works and, if applicable, provide a high-level flow diagram.)

1. CONTEXT

(What processing chain characterizes your algorithm? [see Figs. 1-4]. What functions within the chain are covered by your algorithm, e.g., Track Initiation, Track Maintenance, etc? To what phase(s) of SDI is your algorithm applicable - Boost, Post-Boost, Midcourse, Terminal? What are your inputs, e.g., single sensor or multiple sensors, clustered data or not clustered data, etc.? What are your outputs, e.g., 2-d tracks or 3-d tracks, launch parameters?)

2. NOTABLE FEATURES

(List any features that distinguish the algorithm.)

3. SENSOR ARCHITECTURE AND THREAT SCENARIO

(Characterize the scenario that drives the algorithm design. What is the threat size and density? What are the assumptions about background clutter? Is all the clutter removed prior to tracking? What is the constellation size and orbit? What are the measurement errors/biases? What is the target kinematic model? Does the target maneuver? Is its signature related to aspect angle?)

4. SENSOR MODEL/PROCESSING

(What is the model used for sensor/signal processing? What is the probability of detection, false alarm? What is the clutter density and model for the clutter density, e.g., uniform density in regions with a Poisson model for clutter returns? What is the sensor? Is it a scanning or staring sensor? What is the frame time? Is it variable? How many wavebands of data are available? What is the measurement noise, bias, resolution? How are CSOs modeled? What are the attitude/navigation errors/bias assumed? What is the precision of your input data? Specify the interface between the sensor/signal processor and the tracker, e.g., [time, az, el, snr, extended object indicator, covariance].)

5. TRACK INITIATION

(Specify the method used to initiate tracks. How does it work? Are tracks initiated as 2-d tracks or 3-d tracks? Is cold start initiation performed, i.e., is handover data assumed for all tracks? Are tracks processed individually or in a batch or both? How is the initial precision track state estimate generated? Is track initiation performed sequentially or in a batch mode or both? How long does it take to initiate a track (scans or seconds)? How are cluster tracks initiated? If applicable, provide a high-level data flow. If Track Initiation and Track Maintenance are not separable in the algorithm design, so indicate and describe the Track Initiation and Maintenance algorithms in Section 5 and omit Section 6.)

6.1 TRACK MAINTENANCE - DATA ASSOCIATION

(Specify the data association approach for scan to scan, similar sensor platform to platform association and dissimilar sensors. Are multiple hypotheses generated? How is the combinatoric explosion controlled: pruning, merging, clustering? What is the branching factor in the hypothesis tree? How deep are the hypothesis trees? How are hypotheses scored: Bayesian, Likelihood, Heuristic, other? What search algorithm is employed to develop track hypotheses? Is an assignment algorithm used: Munkres, Brogan-Lemay, other? Does track spawning occur? For satellite-to-satellite association, how is the resolution difference between the satellites accounted for in performing association? How are stars and false alarms [stationary clutter] handled?)

6.2 TRACK MAINTENANCE - STATE ESTIMATION

(Specify the approach used to estimate the state vector for the track. What is the coordinate system? What is the state vector? What type of filter is used? What assumptions are made in the filter, e.g., what is the dynamical model, what dynamics are unmodeled and treated as noise, are noise measurements assumed uncorrelated, is the measurement noise assumed to be constant? Is the filter nonlinear, iterated, batch, sequential? Are weighted sums of Gaussians used to generate the state vector estimate as in PDAF or JPDAF? How does the filter account for biases due to targets becoming resolved [spawning] or become unresolved [crossing]?)

6.3 TRACK MAINTENANCE - TRACK PROMOTION/DEMOTION

(Specify the criteria for promoting, demoting and terminating tracks. Specify the method for maintaining the track status. [The track status indicates the degree of confidence that the track represents a valid target.] What is the criteria for pruning or promoting a track? What is the scoring method, e.g., Bayesian, Likelihood, heuristic measures such as m out of n?)

7. TRACK FILE MAINTENANCE

(Specify the data that is maintained in the track file. Are extended objects, clusters, maneuvering objects, complete trajectories, threat corridors, etc. detected and maintained? What special data structure for datafile management have been used?)

8. OUTPUT TO BM/C3 AND USERS

(Specify the interface to users of the tracking data, esp. BM/C3. What is the data in the interface? How often is the data sent? What is the reporting criteria? How is the data computed?)

9. COMPUTATIONAL REQUIREMENTS

(Specify the throughput and memory requirements of the tracking algorithm. What scenario is used? How are throughput and memory measured? What machine/language is used? Are the results scaleable? Are the results empirical or theoretical? Are there performance bounds? What is the target machine? Is the target machine special or general purpose? Describe the degree of parallellism [e.g., 10 processors, 100 processors or more?] and the processor architecture. What are the sequential and parallel throughput requirements?)

10. CURRENT STATUS

(Describe the current status of the algorithm: conceptual development, design, coding/debugging, implementation, testing? Has the algorithm undergone performance optimization? Has hardware been optimized to execute the algorithm? What are the future plans? Has real data from current sensors been used?)

11. PERFORMANCE MEASURES AND RESULTS

(Specify the performance measures used to characterize the tracking algorithm. How are the performance measures defined? What scenario(s) were used to generate the performance values? What are the results? Are there theoretical performance bounds, e.g., Cramer-Rao bounds? What are the performance limits? Under what conditions does the algorithm perform poorly?)

12. REPORTS

(List report(s) that are relevant to the description and performance of the algorithm. Additional data such as equation development should be referenced.)

GENERIC ALGORITHM PROCESSING CHAINS

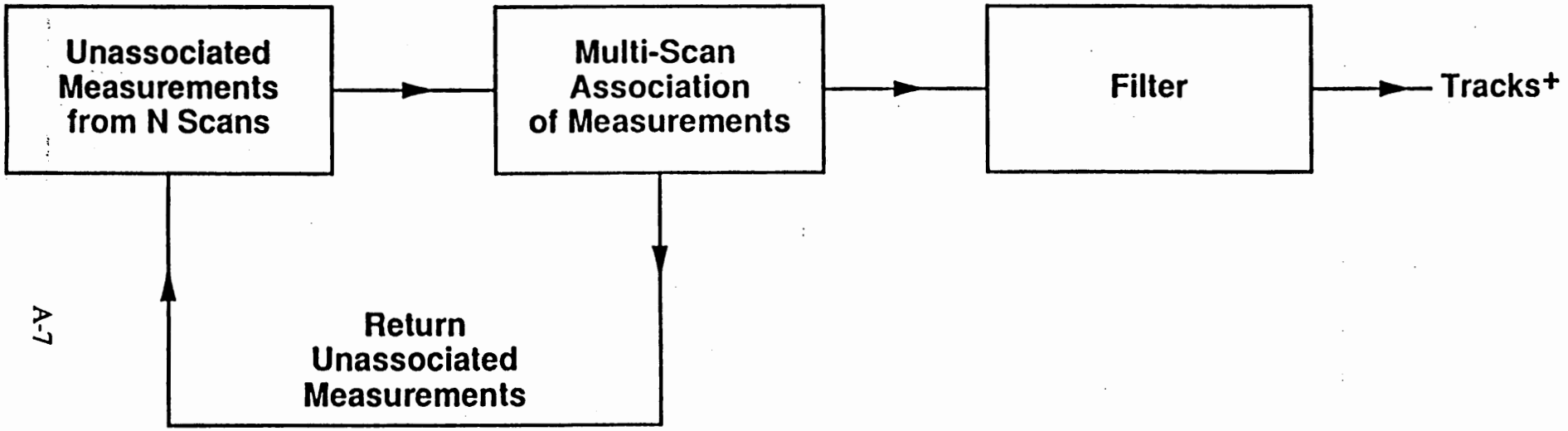
The panel on Critical Issues in Tracking has determined that there are four basic processing chains for track initiation and four basic processing chains for track maintenance. These chains are pictured below.

The depictions of processing chains presented here are meant to be high level descriptions of the logical flow of information in algorithms. It is expected that any algorithm can be defined in terms of these chains.

In order to remain at a sufficiently high level of generality so that these chains are applicable to all algorithms, it is important to use terminology that is indicative of but not specific to techniques in particular processes. Hence, the terms association, filter and track are used in their most general sense.

There are two major areas of detail lacking in these depictions. First, specific definitions and descriptions of techniques for each part of the process. Second, the distribution and location of computer resources.

(Cold Start) Single Sensor Track Initiation [Type I]



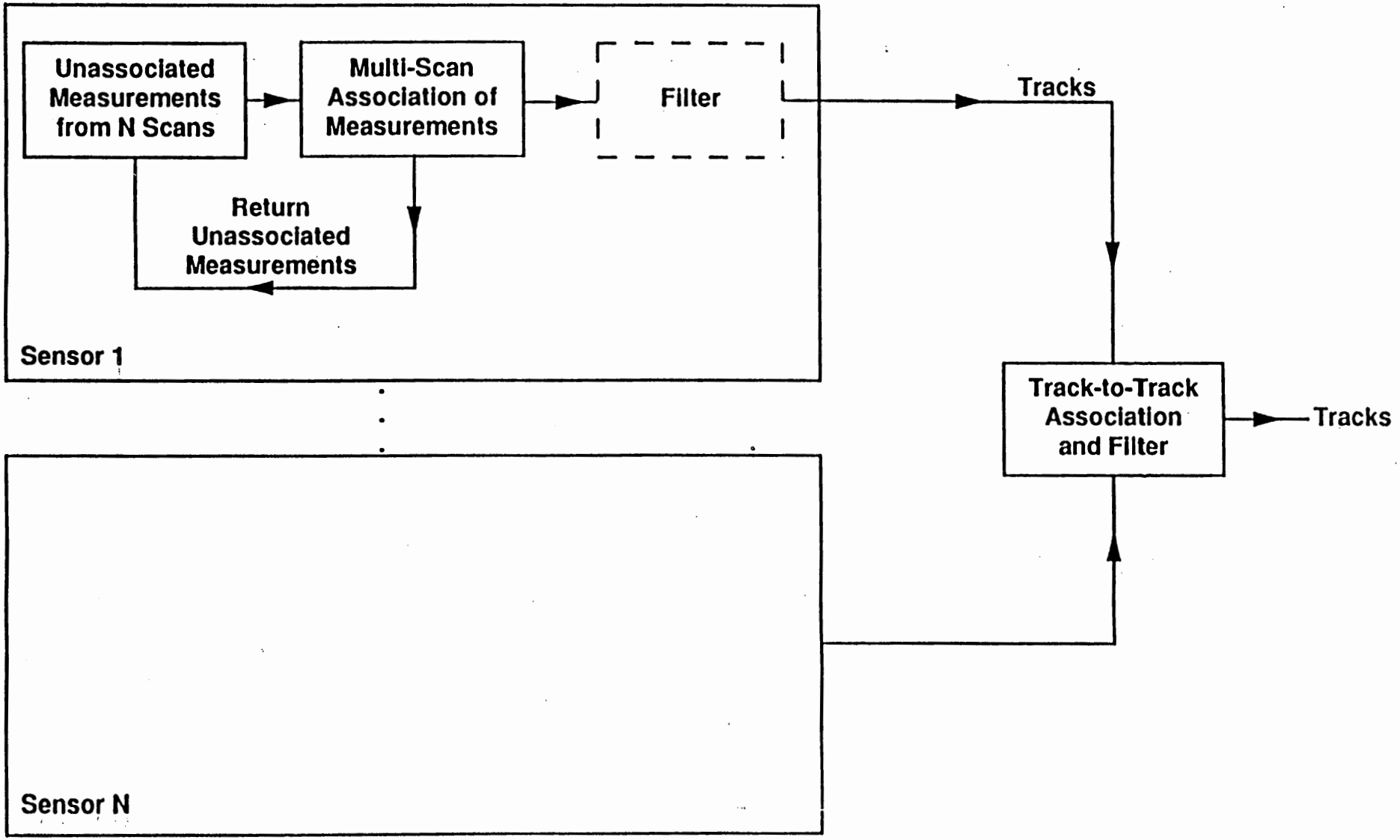
Track $\triangleq$ sequence of measurements

<u>or</u>

filter

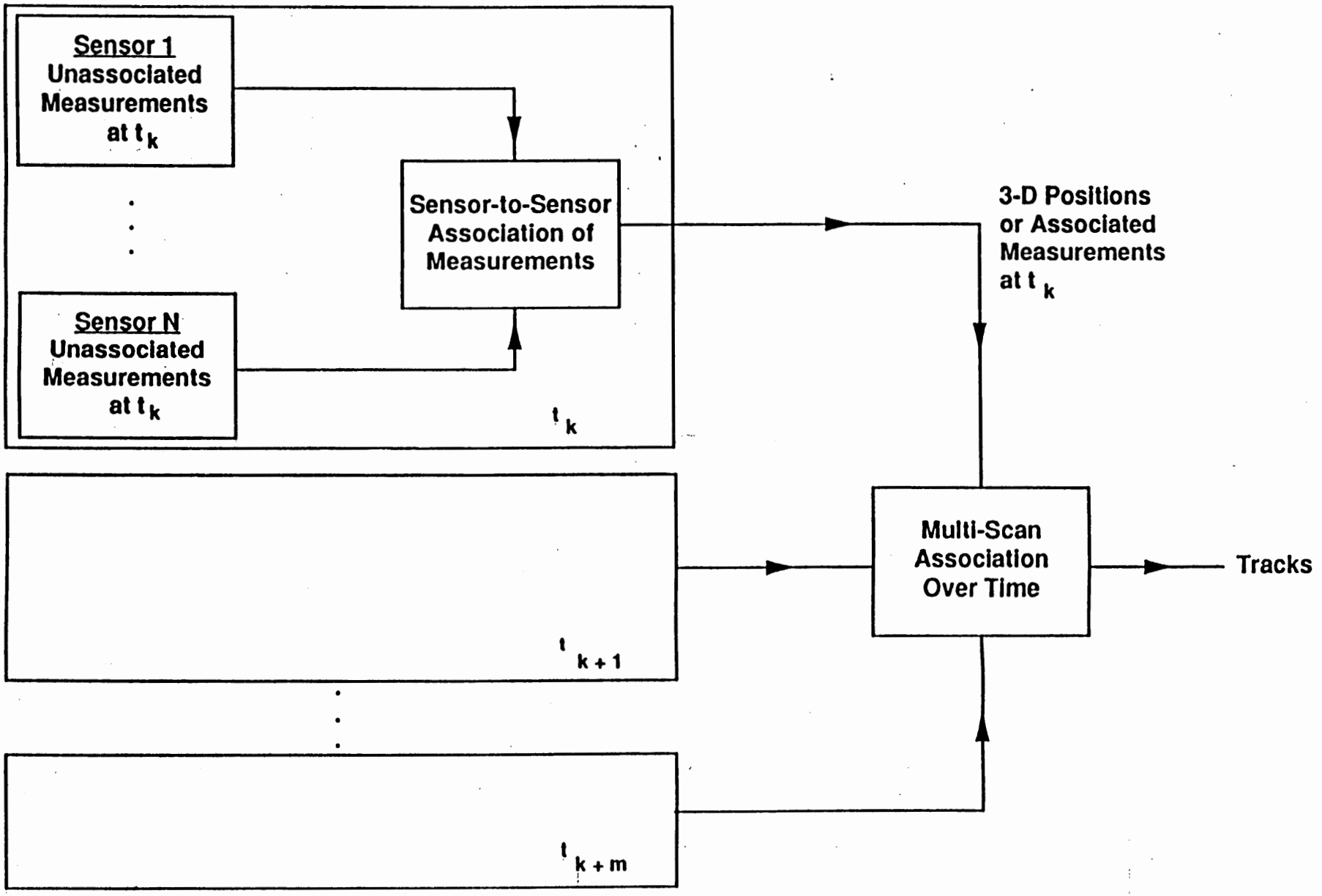
BF:kg-1
1/23/89

(Cold Start) Scan-to-Scan then Sensor-to-Sensor Track Initiation [Type II]



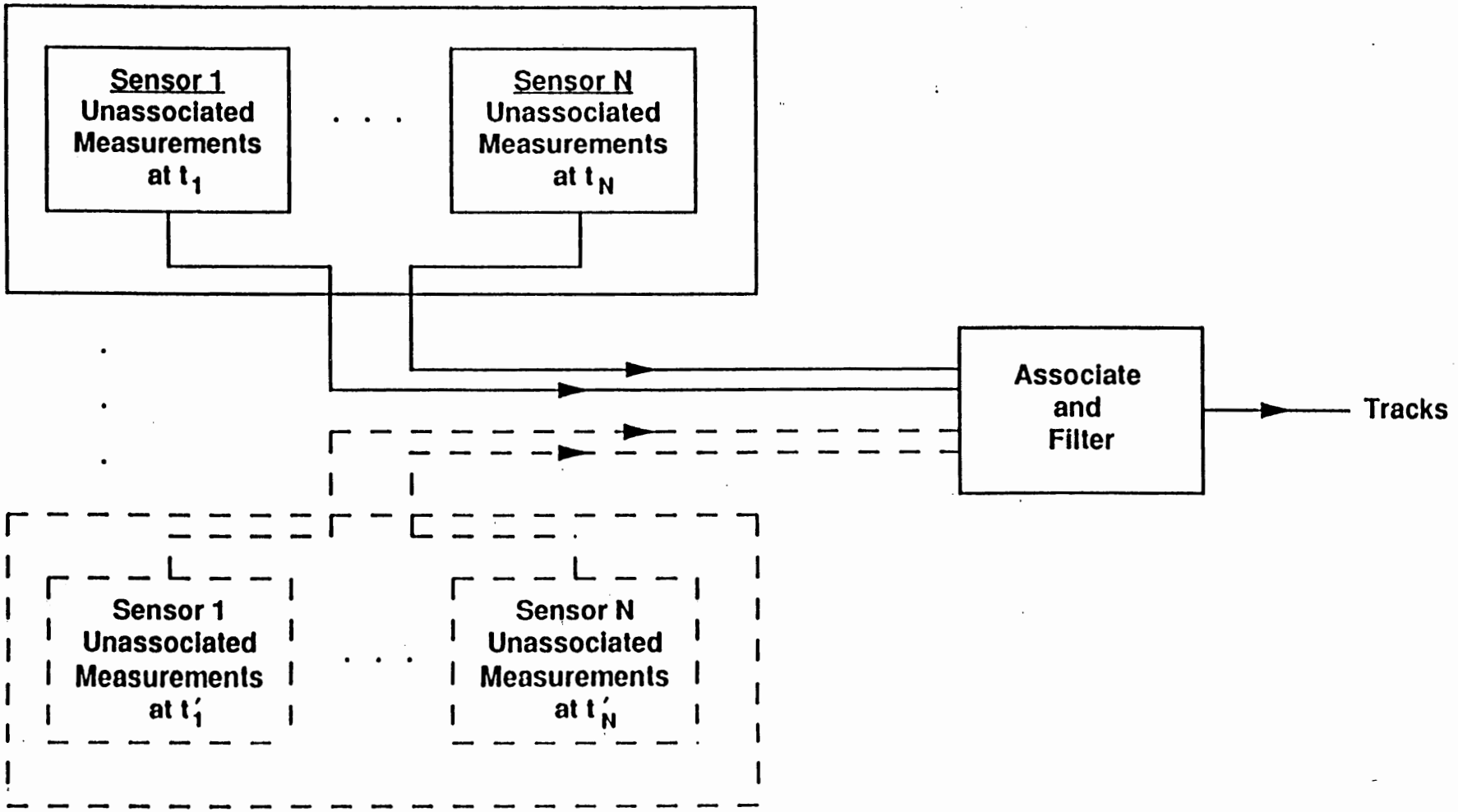
BF:kg-2
1/23/89

(Cold Start) Sensor-to-Sensor then Scan-to-Scan Track Initiation [Type III]



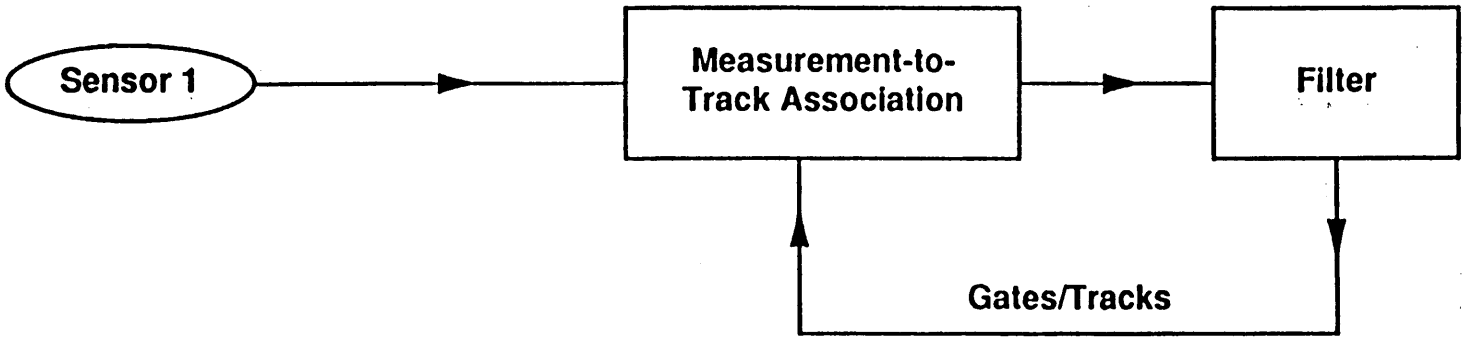
BF:kg-3
1/23/89

(Cold Start) Centralized Track Initiation [Type IV]



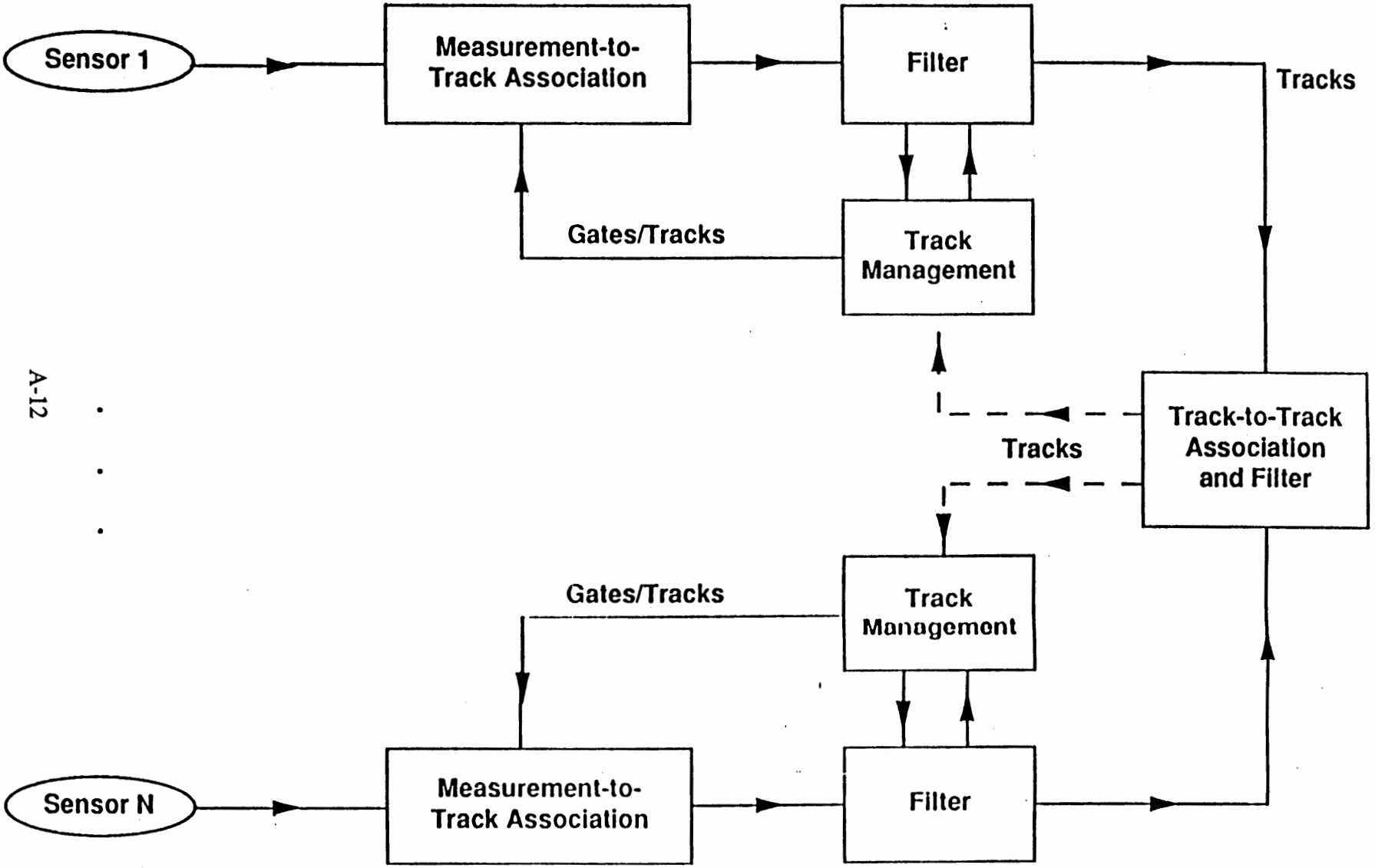
BF:kg-4
1/23/89

Single Sensor Track Maintenance (Type I)



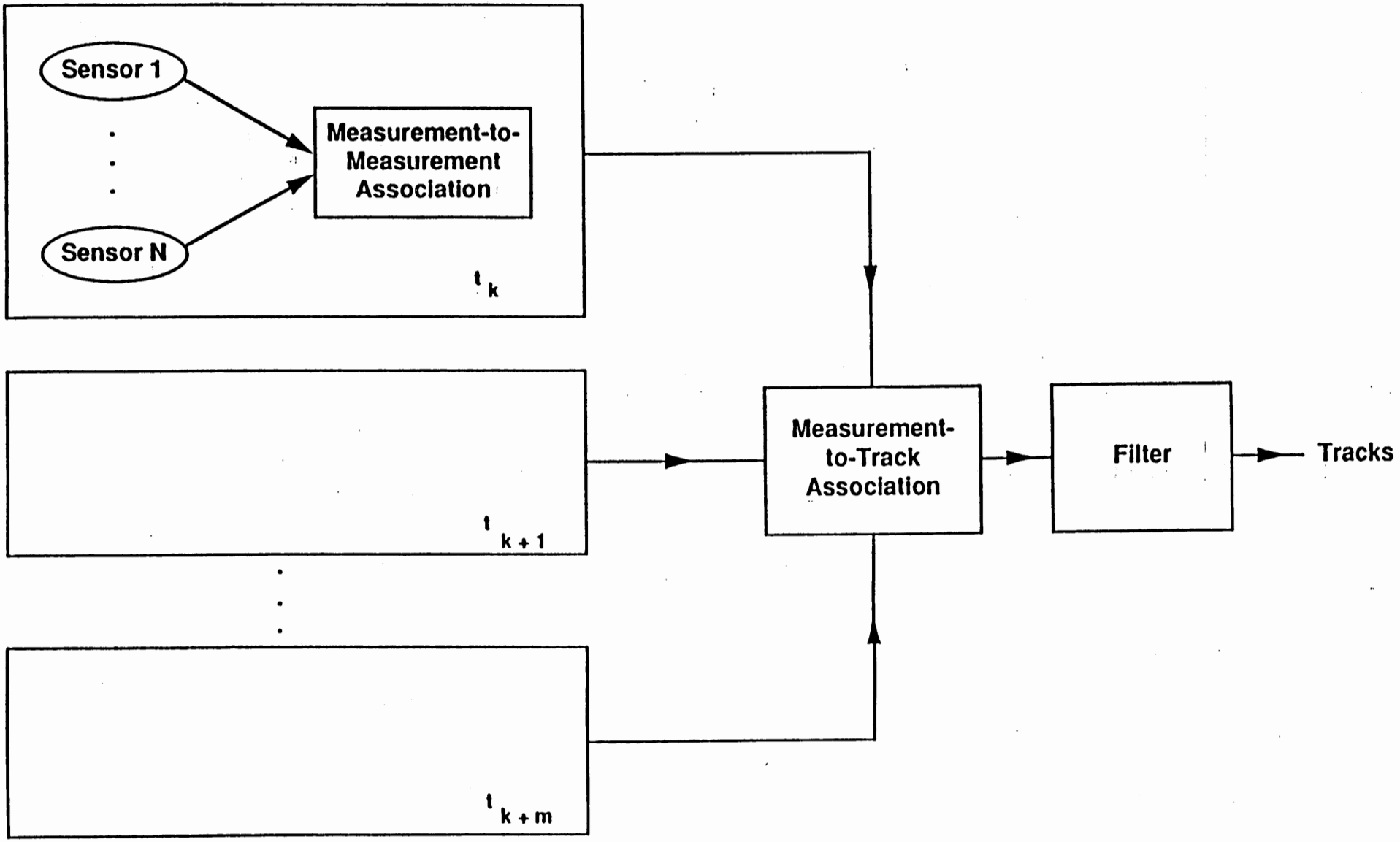
BF:kg-5
1/23/89

Single Sensor Then Fuse Track Maintenance (Type II)



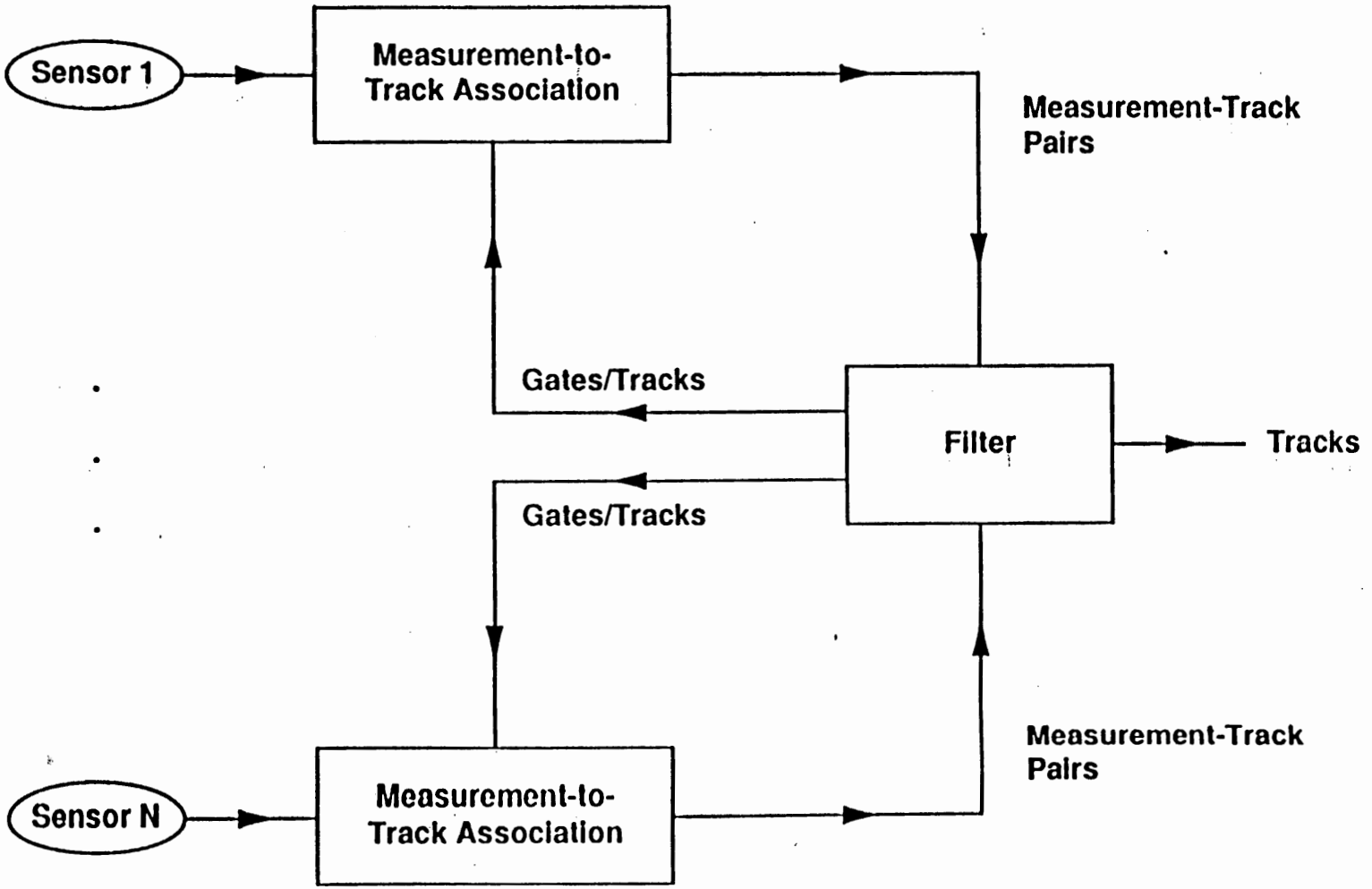
BF:kg-6
1/23/89

Sensor-to-Sensor then Scan-to-Scan Track Maintenance (Type III)



BF:kg-6
1/23/89

Centralized Sensor-By-Sensor Track Maintenance (Type IV)



BF:kg-8
1/23/89

APPENDIX B

ALGORITHM SURVEY RESPONSES

TRACKING ALGORITHM SUMMARY

SUBMITTED BY: Edward J. G. Goodchild 7th February 1989
Advanced System Architectures Ltd.
Johnson House,
73 - 79 Park Street,
Camberley,
Surrey,
England, GU15 3PE.

Telephone 011 44 276 682756

AN OBJECT-ORIENTED ARCHITECTURE FOR SENSOR DATA FUSION/TRACKING IN DENSE TARGET ENVIRONMENTS

SPONSORS: ADVANCED SYSTEM ARCHITECTURES,
SDIO BM/C³, Captain Johnson

DEVELOPER: Advanced System Architectures, Data Fusion Department

AUTHOR: Edward J. G. Goodchild

SUMMARY

The "algorithm" is a target-oriented system designed to perform multi-sensor, multi-object data fusion and three-dimensional tracking in a computationally practical manner in dense target environments. It has been designed to operate with data from any mix of bearing-only and 3-D sensors, and tracks both individual targets and target clusters of arbitrary size.

The algorithm is based upon an object-oriented system architecture, embedding the tracking filters and data association gating functions within replicable logical objects. One such object, or process, is assigned to track each perceived target, thus exploiting to the maximum the inherent parallelism in the data fusion task. A tracking process has the sole task of increasing its knowledge about the body it is modelling by examination of sensor data, and using any such reports that are relevant to that body.

1. CONTEXT

1.1 PROCESSING CHAINS

The processing chain characterising the data fusion algorithm differs depending on the type of data provided by the sensor. The system has been designed to accept data in the form of hits or scan-to-scan correlated tracks in either two or three dimensions.

Track initiation from uncorrelated 2-D hits follows the processing chain shown in figure 1, which indicates how new track hypotheses are formed from hits from all available sensors without an intermediate stage of track formation by individual sensors. Track initiation from sensor tracks or with three-dimensional hits follow variants of the "Centralised Sensor by Sensor" processing chain shown in figure 2[1]. Track maintenance follows variants of the "Centralised Sensor by Sensor" processing chain for all types of input. The principal variations from figure 2 concern the type of gating applied to the sensor inputs; this is dependant on the sensor data type.

The algorithm covers the 3-D tracking functions and the gating and association of incoming data with the 3-D track projections, but excludes any single-sensor, scan-to-scan correlation and track forming functions.

1.2 PHASES OF THE SDI BATTLE

The algorithm has been designed for birth-to-death tracking of ballistic missiles, from launch all the way through to the terminal phase. Effort has been concentrated, thus far, on the transition from boost to mid-course phases, including the period of target proliferation during the post-boost phase.

1.3 INPUTS

The algorithm has been designed to accept any available sensor data with no prior assumptions on the number or configuration of sensors. Sensor data may be clustered or otherwise, and may be scan-to-scan correlated or otherwise.

1.4 OUTPUTS

The algorithm produces full three-dimensional tracks of individual targets, of clusters of targets, and of individual members of clusters where sufficient data has been received to resolve them. Extensions are planned, but not currently supported, to include target discrimination functions, and to provide extrapolation of the tracks both forward and backward to yield aim-points and launch parameters.

2 NOTABLE FEATURES

- the ability to handle any mix of sensor data, as described above.

[1] Figure 1 in attachment 4 of IDA Memorandum dated December 27th from Gabriel Frenkel.

- the ability to initiate full 3-D tracks from three uncorrelated 2-D observations on a target from any sensors (even a single sensor provided that its movement relative to the target is sufficient), and to prune swiftly the rapid proliferation of false track hypotheses.

- the scalability of the architecture to any conceivable threat size, with predictable growth with growth of the threat environment.

3 SENSOR ARCHITECTURE & THREAT SCENARIO

The algorithm, although originally concieved as a generic data fusion architecture, has been developed in the context of the SDI. The scenario driving the design has been that of a mass ballistic missile attack including many launchers producing vast numbers of mid-course bodies (>10^6). These have been assumed to travel in clusters of up to a few hundred objects each.

The system has been designed to operate with any likely configuration and number of sensors. The algorithm has been designed to take account of measurement errors and biasses (including sensor own-position reporting, pointing accuracy, and bearing resolution) but performance measurements have not been made.

Three kinematic models have been assumed for targets, a pure ballistic model for the mid-course phase, a ballistic model with acceleration along the trajectory for the boost phase, and a ballistic model with variable acceleration, predominantly along the trajectory for the post-boost vehicle. This variable acceleration is included as a noise term in the kinematic model.

At present, tracking is on target position and motion only. Versions of the design exist which include photometric parameters in the target state vector, and it is planned to make use of these for target discrimination purposes in the future.

No specific assumptions have been made, to date, as to the nature, quantity, or pre-filtering of background clutter.

4 SENSOR MODEL/PROCESSING

In principal, the algorithm is not limited to operation with particular sensor types. The evolution of the design in the SDI context has directed concentration towards operation with passive, focal-plane, IR sensors. It will operate with both scanning and staring sensors, producing either geometric data only or both geometric and photometric data. Scan rates and frame rates may be fixed or variable.

The algorithm requires data to be supplied on at least: observation time, sensor position, bearing of observed object in two orthogonal planes. No other limits have been assumed on sensor signal processing capabilities.

5 TRACK INITIATION

5.1 TARGET TRACKS

Tracks are initiated directly in three-dimensional form even from un-correlated 2-D hits. They are initialised by association of any sensor reports not

associated with existing tracks to form new-track hypotheses. These are tested for realism and likelihood at every stage, with hypotheses failing either test being deleted. Association of three hits in a new-track hypothesis is sufficient, in general, to allow its promotion to a full track.

There is no necessity to initialise the system with tracks handed over from elsewhere following a cold-start. Provision for such hand-over could be added to the algorithm without difficulty, and could reduce the settling-time of the system under cold-start conditions.

5.2 CLUSTER TRACKS

All launch vehicles are assumed to produce clusters of mid-course bodies, thus cluster tracks are generated for all boost-phase tracks. At this stage they have a membership of one target, i.e. the booster itself. New tracks formed by splitting during the post-boost phase are assigned as members of the cluster.

Cluster tracks are also formed later by association of a number of non-cluster-member tracks all having near identical trajectory behaviour.

6 TRACK MAINTENANCE

6.1 DATA ASSOCIATION

A multiple gating approach is used for data association. Firstly, new data is gated with the cluster tracks, using a simple rectangular gating strategy. A two-level gating strategy is then applied to the data by target tracks belonging to clusters with which the data is successfully associated. The first level comprises a rectangular gate; successful association at this level leads to a more stringent, ellipsoidal gate. Only data passing all stages succesfully is used for track updating.

6.2 STATE ESTIMATION

A Kalman filter technique is employed for target track state estimation. Currently, this comprises a six-degree-of-freedom filter, tracking position and velocity. Plans exist to extend the state vector to include aim-point, launch point, and photometric parameters with appropriate filters.

A similar six-axis Kalman filter is used to maintain the cluster track trajectory, and cluster extent. The cluster track also maintains its target-track membership list.

6.3 TRACK PROMOTION/DEMOTION

Tracks are continuously monitored for validity. Pruning tests implemented and planned include: consistent and sensible track behaviour, behaviour of the error co-variance with time, and continued updating with new sensor data. Consistent failure of any of these tests would cause deletion of the track.

7 TRACK FILE MAINTENANCE

Currently not implemented.

8 OUTPUT TO BM/C³ USERS

Currently, the output of the algorithm is all the track state vectors, as they are updated.

9 COMPUTATIONAL REQUIREMENTS

The algorithm has been designed to operate on a large, concurrent, message-passing processing system, with the ability to distribute the system over a number of platforms. The algorithm has been designed explicitly to maximise processing parallelism. The number of processors required in such a machine would probably exceed 1000, in a loosely-coupled architecture.

The algorithm has been developed with the Auto-G CASE tool, using the G notation, with automatic code generation of Ada and the Ada-based SADMT language developed by the IDA. Simulations have been run of part of the design coded in SADMT on SUN 3 workstations.

10 CURRENT STATUS

The algorithm is in a concept proving stage, with the core part of the design having been completed in Auto-G. A portion of it, the target tracking process has been converted to SADMT code.

Only limited performance optimisation has only been carried out.

11 PERFORMANCE MEASURES AND RESULTS

The target tracking process coded in SADMT has been run successfully in simulations using the IDA SADMT Simulation Framework, demonstrating track initiation and maintenance. Performance metrics have not been produced.

12 REPORTS

ORIGINAL PRESENTATION OF CONCEPT:- "An Object Oriented Approach To Data Fusion"; Chapter 3, Section 3.5 of "Application Of Artificial Intelligence To Command And Control Systems"; C. J. Harris (Ed); Peregrinus Press for the Institution of Electrical Engineers.

INITIAL DESIGN OF ALGORITHM:- ASA Technical Report T88/001, dated 15th March 1988, produced for the SDIO as the final report of the original DoD SDIO contract SDIO84-87-C-0040, entitled "Demonstration Of Specification Methodology For SDI Data Fusion".

DETAILED DESCRIPTION OF ALGORITHM:- ASA Technical Report T88/009, dated 15th June 1988, produced for the SDIO as the First Phase report under the extension to DoD SDIO contract SDIO84-87-C-0040, entitled "Data Fusion Architecture Refinement And Simulation".

DEVELOPMENT OF SIMULATION AND PRESENTATION OF SIMULATION RESULTS:- ASA Technical Report T88/014, dated 15th December 1988, produced for the SDIO as the Final Report of the extension to DoD SDIO contract SDIO84-87-C-0040.

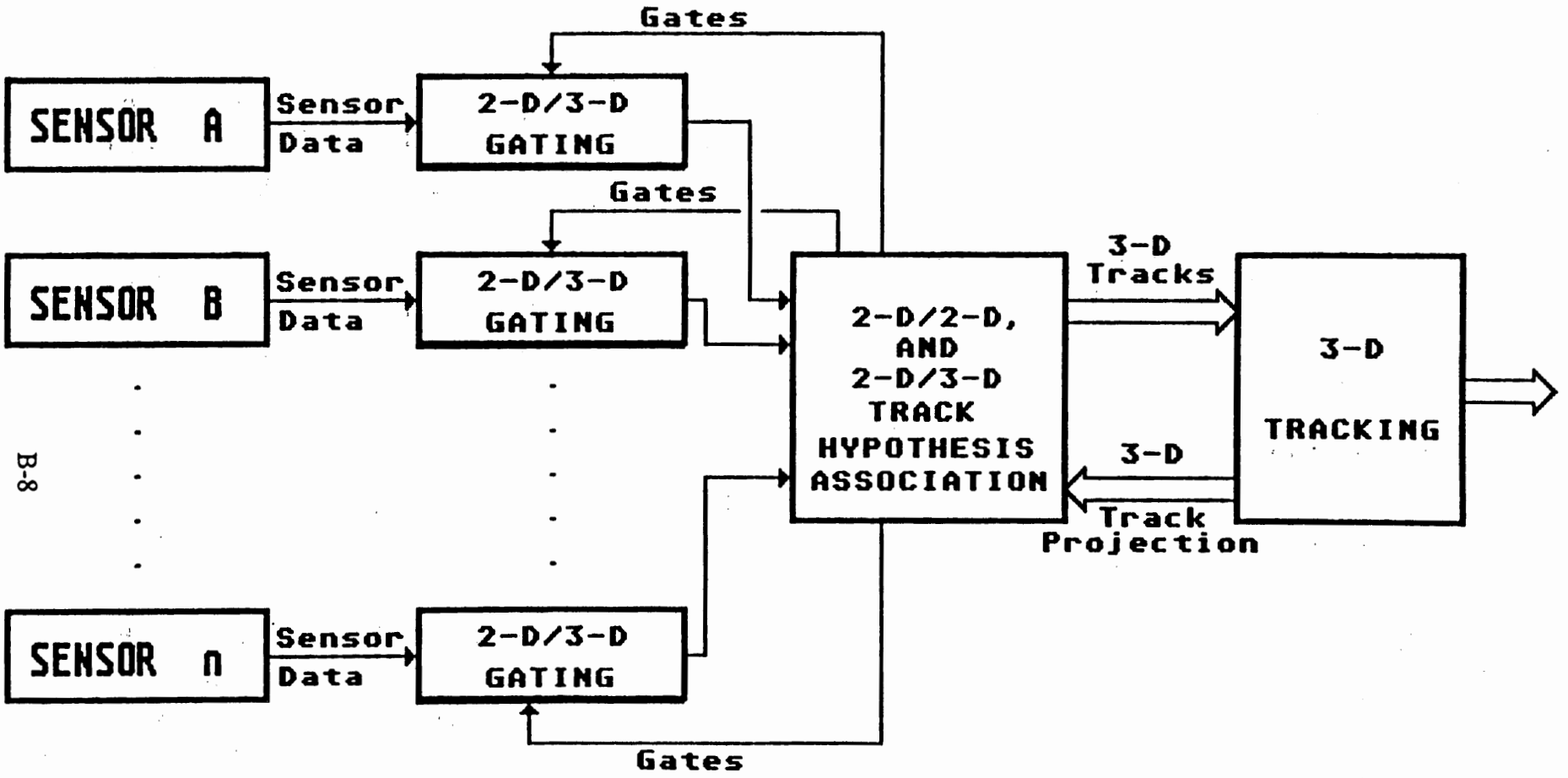


FIGURE 1; PROCESSING CHAIN FOR TRACK INITIATION FROM 2-D HITS

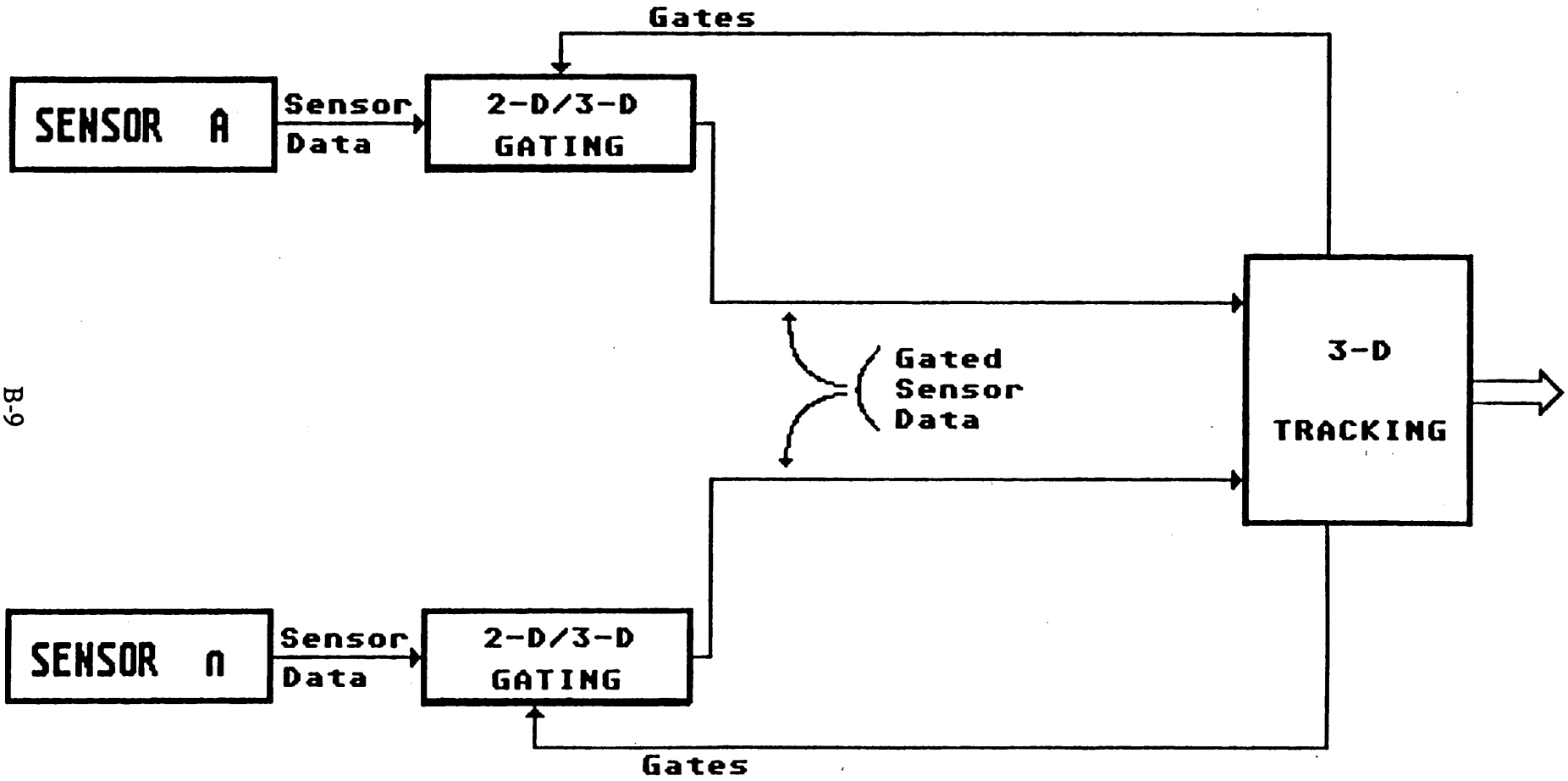


FIGURE 2; PROCESSING CHAIN FOR TRACK MAINTENANCE

TRACKING ALGORITHM SUMMARY

SUBMITTER'S NAME: Robert B. Washburn, Jr. **DATE:** March 27, 1989
SUBMITTER'S COMPANY/ORGANIZATION: ALPHATECH, Inc.
SUBMITTER'S PHONE: (617) 273-3388
ADDRESS OF SUBMITTER: 111 Middlesex Turnpike, Burlington, MA 01803
AUTHOR'S COMPANY/ORGANIZATION: ALPHATECH, Inc.
AUTHOR'S PHONE: (617) 273-3388
ADDRESS OF AUTHOR: 111 Middlesex Turnpike, Burlington, MA 01803

TITLE OF ALGORITHM: Multiple Information Set Tracking Correlator (MISTC)

SPONSOR: U.S. Army Strategic Defense Command CSSD-H-SBY

DEVELOPER: ALPHATECH, Inc.

1. DESIGN SCENARIO

The algorithm design was driven by midcourse scenarios after RV deployment and before re-entry. Total threat size before data partitioning was assumed of the order of 10^5 objects, group size after data partitioning was assumed of the order of 100 objects. A range of target densities were used for simulated scenarios in evaluation of group tracking and correlation. The threat SDC-I-1 was used for evaluation of data partitioning. All objects were assumed to fly Keplerian trajectories without maneuvering. Models for space-based, airborne, and ground-based sensors were used to allow representation of SSTS, GSTS, AOS, and MGBR. Handover track data could also be included. A variety of sensor architectures were used. Registration, calibration, etc. was assumed to be performed prior to tracking; only sensor noise errors were modeled.

2. SENSOR MODEL/PROCESSING

The sensor models included effects for detection (constant probability of detection within the sensor FOV, Poisson uniform false alarms in the FOV), measurement resolution (measurements within specified sensor resolution were combined — however, no algorithms

were developed to handle this during the MISTC project), measurement accuracy (additive Gaussian noise), and single sensor data processing (tracking in the single sensor's measurement coordinates — e.g., angle scan-scan correlation and tracking). Attitude, navigation, etc. errors and biases were not modeled. The sensor output to the tracking algorithm included time, CSO indicator (however, not used in algorithms), measurement track, measurement track error covariance, measurement track ID, measurement track probability distribution of target class, measurement track time. The measurement track format varied depending on the type of sensor (e.g., four dimensional angle and angle rate state for angle-only sensors).

3. TRACK INITIATION

Tracks are initialized as 2-D tracks until multiple sensor correlations are made (then a 3-D track is initialized). Handover data can be used if available, but is not necessary. Tracks are processed individually and sequentially.

4. DATA ASSOCIATION

Data association was accomplished in two stages. The first stage partitioned one scan of LOS data from multiple sensors (two or three) into spatially separated groups. Then each group of data was processed by the tracking/correlation algorithm. Within the group eight different data association algorithms were investigated, including three zero-scan, pairwise approaches (row-column, row-column with backtracking, and optimal RELAX assignment) and five N-scan, multiple hypothesis approaches (multidimensional, maximum, marginal return, branch and bound, and three new algorithms by Tsaknakis). These algorithms are described in [2]. The same correlation algorithm performed scan-scan and sensor-sensor correlation. Hypotheses were scored by likelihood ratios. Ten branches were allowed per target (five appeared adequate). The initial data partitioning was able to keep the group sizes fairly small so that gating and pruning were sufficient to keep the number of hypotheses from growing.

5. STATE ESTIMATION

State estimation was accomplished using extended Kalman filters assuming Keplerian motion between measurement times. The algorithms developed did not handle track spawning.

6. TRACK FILE MAINTENANCE

The track file stored track time, track state type (e.g., 2-D or 3-D), mean, error covariance, ID, probability distribution of target class (e.g., RV, balloon, light replica), list of sensors correlated for that track and corresponding sensor track IDs.

7. OUTPUT TO BM/C3 AND USERS

Not applicable. Entire track file was assumed to be available to user systems.

8. PERFORMANCE MEASURES

Performance measures included number of true tracks generated, number of false tracks generated, true track accuracy, track time to initialize, and life-time. These are detailed in the report [1]. Performance bounds were not computed.

9. COMPUTATIONAL REQUIREMENTS

Computational requirements were obtained by partitioning realistic threat tapes to obtain a distribution for group size and then running the tracking/correlation algorithm on groups of different size to obtain empirical estimates of run-time. Run-times on a VAX 11/750 (using VMS FORTRAN) were translated to MEAS (million equivalent floating point additions per second) and scaled to larger threat sizes. The details and results of this computational evaluation are contained in [3]. The rough requirements were 90-120 MEAS per 10000 objects for average throughput with about 80 MBytes for total memory.

10. NOTABLE FEATURES

The data partitioning algorithm proved effective in decomposing the sensor data into groups small enough for uniprocessor computation. The Tsaknakis algorithms exhibited near-

optimal performance with processing not much greater than the simplest algorithm. Description of these algorithms and their tracking performance are contained in [2].

11. CURRENT STATUS

Algorithms have been implemented for sequential, off-line processing in FORTRAN (VMS). Code was not optimized. Plans are to incorporate CSO tracking and resolution into algorithms and to implement on different parallel processors.

12. REPORTS

All reports are unclassified with the usual SDI limitation on distribution and are available from the U.S. Army Strategic Defense Command.

1. Catlin, R.A., and R.B. Washburn, *Evaluation Methodology for Multiple Information Set Tracking Correlator (MISTC), CDRL Item A009*, ALPHATECH Technical Report TR-385, Burlington, Massachusetts, March 1988.

2. Allen, T.G., L.B. Feinberg, R.O. LaMaire, K.R. Pattipati, H. Tsaknakis, R.B. Washburn, W. Wren, P. Patterson, and T. Dobbins, *Multiple Information Set Tracking Correlator (MISTC) Final Report CDRL Item A007*, ALPHATECH Technical Report TR-406, Burlington, Massachusetts, September 1988.

3. Washburn, R.B., *Multiple Information Set Tracking Correlator (MISTC) Processing Requirements, CDRL Item A006*, ALPHATECH Technical Report TR-348-2, Burlington, Massachusetts, September 1988.

TRACKING ALGORITHM SUMMARY

SUBMITTER'S NAME: Mr. Larry Filippelli DATE: 15 November 1988
SUBMITTER'S COMPANY/ORGANIZATION: Ball Systems Engineering Division
SUBMITTER'S PHONE: (703) 528-3337
ADDRESS OF SUBMITTER: 1100 Wilson Blvd. Suite 1710, Arlington, Va. 22209
AUTHOR'S COMPANY/ORGANIZATION: Same as above

TITLE OF ALGORITHM: SDI Midcourse Tracker/Correlator Algorithm (TRC)

SPONSOR: Naval Research Laboratory (Dr. Kurt Askin, Mr. Steve McBurnett)

DEVELOPER: Ball Systems Engineering Division and Daniel H. Wagner, Associates

---

The following responses are offered in response to questions posed in the summary related to the NRL Tracker/Correlator Algorithm under development by Ball Systems Engineering Division (VERAC, Incorporated) and Daniel H. Wagner, Associates. Since some responses could result in lengthy algorithm descriptions that exist in our Algorithm Design Document, a reference to a pertinent section or sections of the document is provided in lieu of a more definitive answer. A copy of the document is enclosed for your convenience.

1. DESIGN SCENARIO

A. The algorithm design is essentially scenario independent except that it takes advantage of the fact that objects tend to cluster into spatially inseparable groups in order to reduce the combinatorial explosion that results from the high target density and potentially high sensor observation rate.

B. No assumptions about the threat size and density were made in the design, however the algorithm's implementation on the SUN computer provides upper practical limits due to processor speed and memory availability. A future implementation, currently in progress, on a parallel processor will extend the utility of the algorithm to greater numbers of objects.

C. The algorithm assumes some preprocessing by the sensors to remove clutter. It does however treat the case of false reports in its scoring formulae.

D. The algorithm design is entirely independent of the sensor configurations.

E. No fixed values are assumed for sensor measurement errors and biases. Algorithm parameters may be adjusted to permit use of a full range of postulated sensor errors.

F. Post Boost Vehicles (PBVs) are assumed to undergo random accelerations during MIRVing. All Other objects are assumed to be in ballistic Keplerian orbits through midcourse [See sections 3.4 and 3.5.2 and Appendix A].

ENCLOSURE 1

G. We have selected attributes which are independent of aspect for association purposes.

2. SENSOR MODEL/PROCESSING

A. Sensor configurations and parameters are established by the NRL systems engineer. The algorithm makes no assumptions except those described in section 1.2.

B. The TRC receives the following information from the sensors:

time, sensor position and velocity, azimuth and elevation and errors
IR only: irradiances at three wavelengths, S/N
Radar only: range, radar cross section

3. TRACK INITIATION

A. Currently tracks are initiated by a boost-phase handover report assumed to provide the 6-state estimate of an object (presumably a PBV) plus an 18-element covariance matrix. Enhancements to the algorithm, presently under design, will initiate tracks from 2-D and 3-D sensor reports.

B. Sensor reports are batch processed as indicated in Section 3.2.2.

4. DATA ASSOCIATION

A. Reports from each sensor scan are associated against current target space, i.e. the algorithm is a report-to-track correlator.

B. The algorithm takes a multi-hypothesis approach.

C. Combinatorial explosion is controlled in several ways:
1. local processors and clusters [Section 3.3.1]
2. within a cluster, report-to-map associations must pass a retention test to be considered for inclusion in a scene (hypothesis) [Section 3.5.5]
3. only hypotheses which pass a retention test are passed on for future use. Others are pruned. [Section 3.8.4.4]

D&E. We do not have a fixed depth hypothesis algorithm. Hypotheses are deleted if they are below (in score) an input percentage of the score of the best hypothesis. There is also a maximum number of hypotheses which can be saved.

F. Hypotheses are scored using a Bayesian approach and taking into account a probability of detection, probability of false alarm, false and new target densities and report-to-map association scores. [Section 3.8.4.2 and 3.8.5]

G. A heuristic search algorithm is used to formulate the hypotheses. The algorithm employs knowledge about target space and a key assumption that no two reports in a sensor scan were generated by the same target. [Section 3.8.4.3]

H. Tracks are spawned as described in Section 3.8.4.3 figure 3.8-2. Target maps are merged as described in Section 3.11.

5. STATE ESTIMATION

A. The approach for state vector estimation for a track is described in Section 3.10 and Appendix D.

B. The state vector is a 6-dimensional position/velocity vector plus covariance matrix plus the target temperature estimate and its associated error.

C. The algorithm uses a Kahlman filter with equations of motion based on the modified Euler method. The dynamical model is described in Appendix A.

D. For information on spawning, track scoring and pruning see Section 3.14 as well as previously referenced sections on report-to-map association and scene scoring.

6. TRACK FILE MAINTENANCE

As above, see Section 3.14. At each periodic snapshot, constructed track complex data are output to the track file to update the state estimates for all tracks. CTCs which are not output have been deleted from the track file. In the NRL TSS, the track file data base will be maintained by a separate CSCI. Output CTCs will include a flag which indicates if the object is a PBV.

7. OUTPUT TO BM/C3 AND USERS

A. Snapshots of the current state estimate of objects which exceed a confidence (CTC weight) threshold are output to the track data base. [Section 3.15.3]

B. Data are output periodically, (nominally every 12 seconds). The snapshot interval can be varied by changing an input parameter.

8. PERFORMANCE MEASURES

A. The algorithm is measured in four areas: tracking, target count, correlation and execution time. Please refer to the Test Bed Design Document (Enclosed) for a detailed description of all measures of performance.

B. We are not using maximum likelihood estimation and do not have a theoretical performance measure.

9. COMPUTATIONAL REQUIREMENTS

A. Throughput and memory requirements - none have been established

B. Throughput is measured as number of reports processed per second, where a report is a single observation of a target or unresolvable group of targets by a sensor.

C. The algorithm is implemented on a SUN computer in FORTRAN.

D. Results can be considered scalable on a global, but not a local basis. That is, objects from each missile are considered and processed independently of objects from other missiles. Within the component clusters of a local processor, the problem scales non-linearly with the density of the targets. Problem should scale linearly with the number of missiles provided each missile produces similar target densities.

E. The results to date are empirical based on simple target scenarios.

F. No computational performance bounds have been imposed at this time.

10. NOTABLE FEATURES

A. The combinatorial explosion resulting from high target densities and report rates is limited by clustering and use of retention thresholds.

B. Constructed Track Complexes give a useful output to other battle manager functions. These are a set of maps judged to be alternate representations (under different report-to-track correlation hypotheses) of the same target.

C. Hierarchical algorithm architecture is designed for eventual parallel processing.

D. Target non-kinematic attributes are processed and associated. Currently the algorithm uses temperature derived from irradiance measurements.

11. CURRENT STATUS

A. A second generation algorithm has been completed and is in the testing phase. It has been installed at NRL and LANL.

B. Performance optimization is currently being investigated.

C. Hardware has not been optimized for this algorithm.

D. Future plans include:

- o exhaustive testing and behavior analysis under stressing scenarios
- o continued enhancements underway or proposed:
  - track initiation without handover
  - improved PBV motion models
  - use of color indices as association attributes
  - dynamic thresholding
  - dynamic new and false target density values for scene scoring
  - improved report-to-map assignment scheme for accelerated processing (monotonic logical grid)
  - improved scene processor

- improved scene processor
- parallel implementation on Butterfly

12. REPORTS

Relevant algorithm description documents are enclosed. Performance testing reports are pending completion of testing and will be available from Dr. Kurt Askin at NRL or Mr. Larry Filippelli at Ball Systems Engineering Division.

TRACKING ALGORITHM SUMMARY

SUBMITTER'S NAME: James Ortolf DATE: 13 Jan '89
SUBMITTER'S COMPANY/ORGANIZATION: Applied Research and Engineering
SUBMITTER'S PHONE: 617/271-0258
ADDRESS OF SUBMITTER: 3 Preston Ct, Bedford MA 01730
AUTHOR'S COMPANY/ORGANIZATIION: Caltech Concurrent Computation Program
AUTHOR'S PHONE: 818/356-6671
ADDRESS OF AUTHOR: Thomas D. Gottschalk
California Institute of Technology
Pasadena, California 91125

TITLE OF ALGORITHM: CALTRAX: The Tracking Program for Simulation-88

SPONSOR: Electronic Systems Division (AFSC), Hanscom AFB, MA 01731

DEVELOPER: T. Gottschalk (caltech) and R. Yeung (JPL)
at Jet Propulsion Laboratory, Pasadena, CA

(Complete as appropriate.)

NOTES & INSTRUCTIONS
(The Summary/Abstract of the algorithm should be limited to a half of a page. The basic answers to the questions below should be limited to 3 pages. Additional information, referenced to the section numbers, whould be included in a separate appendix that should be limited to four pages. Classified information should be included in a separate supplement. The total information including Abstract/Summary, basic answeres, appendix and classified supplement shoulld not exceed eight pages. If some of the algorithms or details are proprietary, indicate what is proprietary and discuss only the non-proprietary aspects.)

ABSTRACT/SUMMARY
(Describe, in general terms, how the algorithm works and if applicable provide a high level flow diagram.)

1. CONTEXT
(What processing chain characterize your algorithm? (See Figs. 1-4.) What functions within the chain are covered by your algorithm, e.g., Track Initiation, Track Maintainence, etc? To what phase(s) of SDI is your algorithm applicable - Boost, Post-Boost, Midcourse, Terminal? What are your inputs, e.g. single sensor or multiple sensors, clustered data or not clustered data, etc? What are your outputs, e.g. 2-d tracks or 3-d tracks, launch parameters?)

2. NOTABLE FEATURES
(List any features that distinguish the algorithm.)

3. SENSOR ARCHITECTURE & THREAT SCENARIO
(Characterize the scenario that drives the algorithm design. What is the threat size and density? Whar are the assumptions about background clutter? Is all the clutter removed prior to tracking? What is the constellation size and orbit? What are the measurement

Simulation88 Tracking Model

Thomas D. Gottschalk

California Institute Of Technology
356-48
Pasadena, CA 91125, 818-356-6671

ABSTRACT

This note describes the tracking module for Simulation88 - a boost/post-boost simulation of SDI functions for an SS18/ASAT threat scenario performed on the MarkIII hypercube. The simulation involves a number of separate tasks (environment generation, sensing, tracking and battle planning) each running on separate subcubes of the MarkIII hypercube. Within a given subcube, the particular SDI task is done in a concurrent fashion. Communications among subcubes are done asynchronously. The Sim88 tracking task involves separate subcubes for sensing and tracking, with the tracking task of an individual subcube/tracker in turn divided into 2D mono tracking (using only data from a single sensor) and 3D precision tracking (using 2D mono tracks from itself and an additional tracker/subcube). While the details of the present tracker are rather tightly tied to specifics of the Sim88 threat model, the overall concurrent tracking prescription can be generalized to deal with arbitrary threat scenarios, and modifications along these lines have been begun.

Context

The tracking model for Simulation88 involves eight separate subcubes of the MarkIII hypercube, performing MEO and GEO sensing and tracking tasks for boost and post-boost phases on multi-target scenarios. In barest terms, the task of the tracker is to process pixels from the sensors and ultimately produce trajectory parameter estimates for individual targets which are then passed on to the battle-planning phase of the integrated Simulation88 demonstration. The overall processing chain is illustrated in Fig.(1). At the GEO level, each sensor develops pixel sets at individual scans and passes them to an associated tracking function. The trackers develop 2D kinematic mono tracks from these data. Mono track files are exchanged between stereo partners and associated to form 3D kinematic tracks (ECI position and velocity estimates) which are then used to solve for launch parameters according to a powered flight model. Tracking at the MEO level is conceptually identical to that of the GEO system, with the addition of direct initiation of precision tracks from the GEO tracker.

Figure (2) shows the general processing strategy used for both the GEO and MEO trackers during a single scan. The 2D tracking is done using simple, 2D kinematic Kalman filters for each of the two projections measured by the sensor. Tracks are initiated using a 3-scan batch processor and extended using a track-splitting formalism. Focal plane tracks of sufficient age (typically 4 or 5 total scans) are propagated to a common reference time and collected into a "Focal Plane Report" which is exchanged with the stereo partner. The association of 2D reports into 3D tracks is done in terms of projected positions and velocities along a common reference line (the 'intersection' of the two focal planes). The precision tracking task actually consists of two distinct parts. For precision track initiation, the ECI position and velocity vectors from the stereo association are used to determine initial values for the launch parameters. Once initialized, parameters are updated on subsequent scans by means of extended Kalman filters.

The focus of the Simulation88 effort was the establishment of a flexible, integrated framework for boost/post-boost SDI functions, and the demonstration that the computations for these tasks could be done concurrently in an efficient manner. In the interest of meeting somewhat ambitious demonstration deadlines for the entire simulation, a number of 'shortcuts' in the tracking formalism have been taken - most notably, the decision to design a tracker which is intimately tied to a specific powered flight model and the use of extremely simplistic sensor models. Present work is aimed at removing these defects, with the intent of producing a generalization of the Simulation88 tracker which can process threat tapes.

Architecture And Threat Scenario

The generic sensor architecture for Simulation88 consists of a ring of LWIR sensors in geostationary orbits and two rings of SWIR sensors in circular, polar orbits. The number and nature of sensor rings as well as the number of sensors in an individual ring are specified at run time through an architecture definition file.

While the environment generation phase of Sim88 actually 'flies' the entire sensing constellation, only subsets (typically 2 GEO sensor/tracker pairs and 2 MEO pairs) are simulated in high fidelity as separate subcubes within the hypercube. The choice as to which particular sensor/trackers are actually simulated is determined at run time within the overall simulation framework.

The threat scenario for Sim88 consists of a distributed launch of 'generic' SS18's together with some number of ASAT's. Trajectories for individual SS18's and ASAT's are described using a simple, analytic powered flight model ('Power88') which was developed specifically for the Sim88 task. An individual launch is described by the longitude and latitude of the launch site, the launch time, the initial launch azimuth and a final thrust angle for each stage. The SS18 model has two stages (6 parameters) while the ASAT model is single stage (5 parameters). The SS18 threat typically has about 40 boosters from each of six launch sites launched within a two minute wave. The exact nature of the threat (including targeting to specific sites) is run time reconfigurable. The ASAT threat is generated dynamically at the start of the simulation, with the actual ASAT launches determined by the nature and kinematic properties of the CV architecture.

Sensor Modeling

Sensors for the current program are embarrassingly simplistic : 3D target positions are projected onto sensor focal planes giving a 2D data point. For the GEO sensors, the focal plane is divided into a number of rectangular pixels, and the reported observation is simply the center of an active pixel. The GEO sensor model does not have multi-hit discrimination within individual pixels. For the MEO sensing model, the truth position on the focal plane is simply smeared using Gaussian distributions. At present, $P_D = 1$ and there is no noise. In typical runs, the scan time for the GEO system is 5 seconds and that for the MEO sensor is 10 seconds.

In the integrated Sim88 framework of Fig.(1), an entire subcube (typically 1 node) is used for each sensor. This is sufficient computing power to do a more realistic description of the sensing process, and this task will be included at some level in the next generation of the simulation. The actual fidelity of sensor simulation remains to be determined.

Track Initiation

Two types of track initiation occur within Sim88: 2D focal plane and 3D precision tracks. The initiation algorithms are quite different.

For focal plane tracking, initiation is done using a 3-scan batch processor. At any time, the full sensor pixel lists for the present and two previous scans are maintained in memory, and the batch initiator constantly searches for provisional new tracks subject to essentially only three constraints:

1) The three data used in the track must not coincide with the last three data in any existing track.

2) The last datum in the track must not coincide with the last datum in any established track (defined by $N_{SCANS} \geq N_0$, with typical cutoff $N_0 = 7$.

3) The 3-scan segment must be sufficiently straight, as defined by a number of simple, heuristic cuts on the apparent acceleration.

For the 3D tracker, a precision track initiation is attempted whenever the ECI position vector determined by the kinematic stereo algorithm cannot be associated with any existing precision track. In such cases, the ECI position and velocity vectors of the target (together with covariance estimates) are first determined by a straightforward geometric algorithm, and these values are then used to determine initial launch parameter and covariance values (within the Power88 framework) by a Newton-Raphson inversion of the equations of motion.

The correlation of focal plane reports to form 3D ECI position and velocity vectors is a central part of the precision initiation process. As noted previously, the focal plane tracks from the two mono trackers are first propagated to common reference times, and values of the focal plane positions and velocities projected onto a common reference axis are computed. The two focal plane reports are then associated by forming associations between these projections according to a modified nearest-neighbor algorithm.

Track Maintenance

For the focal plane trackers, track extensions are done using a track-splitting formalism. 2D tracks are extended to predicted positions using the system/measurement models and association regions based on the position covariance are formed. Each observation within the association gate results in a separate extension of the initial track. Tracks are presently deleted on a single missed detection (this can/will be generalized to accommodate $P_D \neq 1$). Track pruning is done by a simple 'common history' algorithm: tracks are deemed equivalent if they use the same sensor observations over the past N_{EQUIV} scans, with typical value $N_{EQUIV} = 4$. Since the kinematic filter is high gain, the choice of which track is kept of an equivalent pair is inconsequential.

For precision tracks, the association of tracks with data is done using a Nearest-Neighbor global association scheme. Specifically, predicted data positions are evaluated for each sensor, and a global association of predicted and actual data sets is done using a modified nearest-neighbor scheme. The precision track is then updated using an Extended Kalman Filter, with each projection processed separately.

Present Status and Future Generalizations

The tracking model based on the prevision descriptions has been completed and integrated within the entire Simulation88 framework, and was part of the successful full Sim88 demonstration for ESD in January 1989. While the achievement of this milestone was a non-trivial accomplishment in itself, there are a number of shortcomings in the present tracker. Specifically

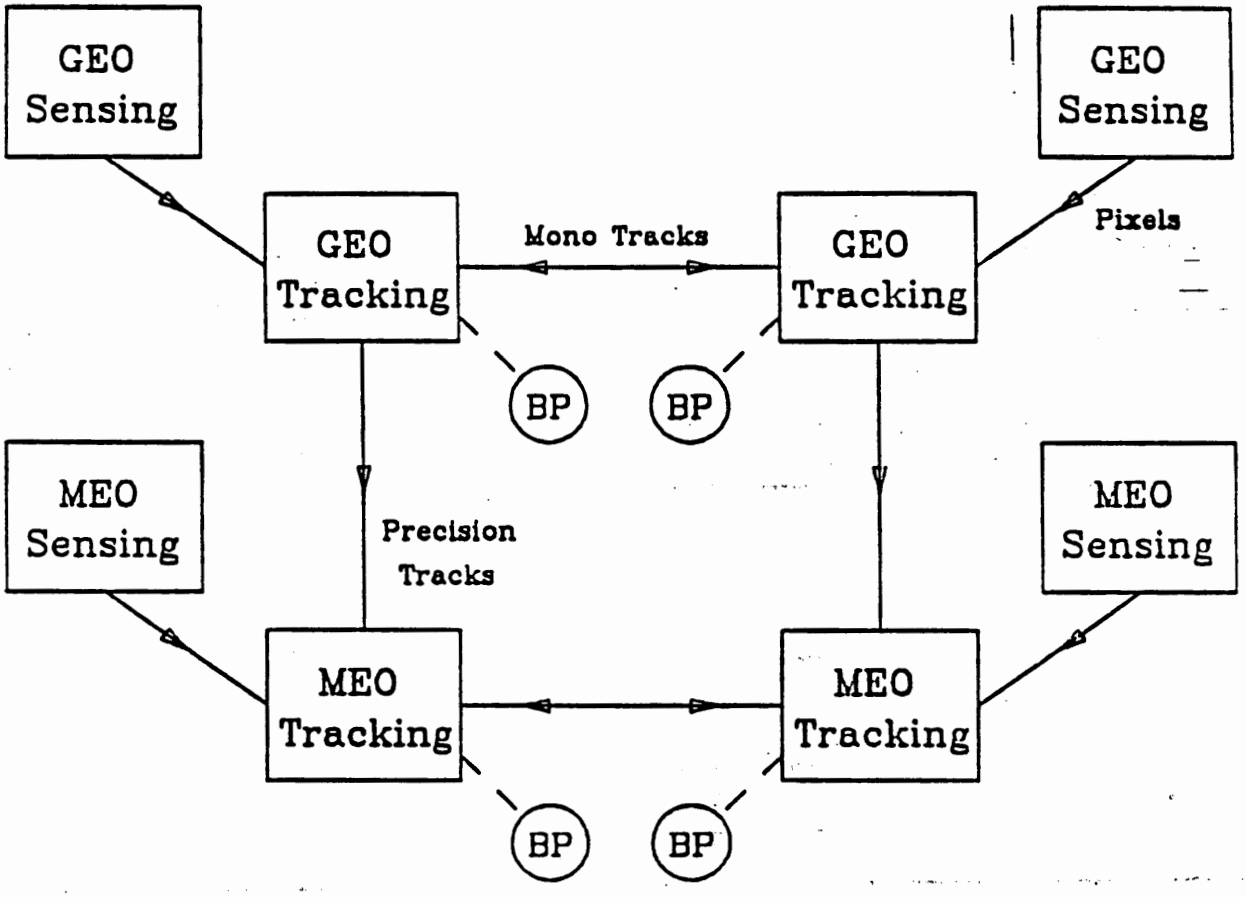
1) The present tracker is tied too intimately to the assumed Power88 threat model.

2) The tracker does not do as good a job as it should in estimating target velocities during post-boost maneuvers.

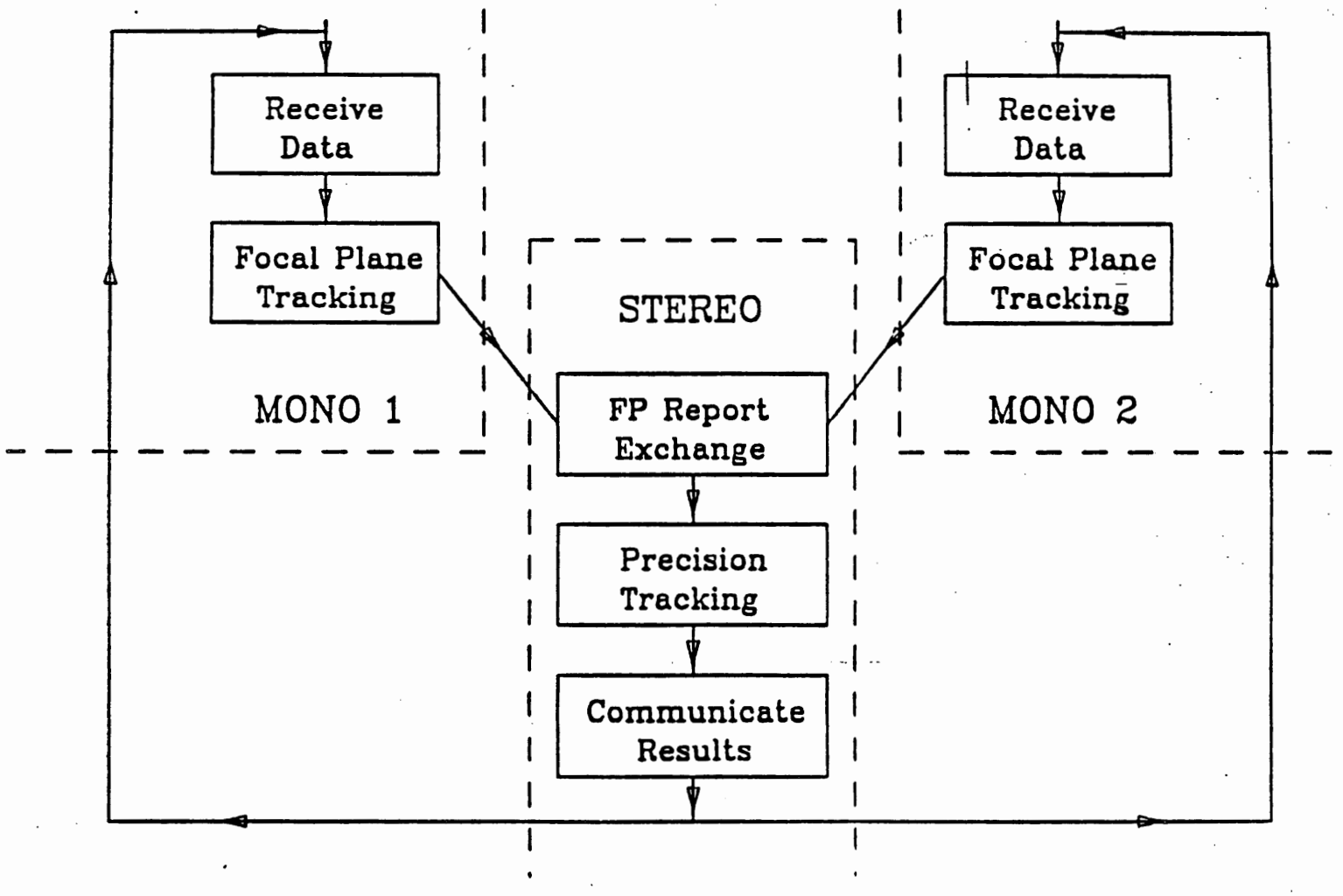
Both problems have a common origin : the design of the precision tracker is based entirely on parameterized trajectories, with all updates of existing precision tracks done using Extended Kalman Filters for assumed trajectory models. For boost phase, there is no reason to believe that the Power88 model might describe a generic threat. During post-boost, the 'noisy-Keplerian' system model has been found to be inadequate.

The next generation tracker will attempt to 'solve' both these problems by using a 3D ECI kinematic state vector formalism as the basic part of the precision track file (for purposes of track-hit associations), with parameter estimations according to arbitrary flight models entering in a manner which is well-removed from the multi-target tracking logic. The essential open question for the new tracking scheme concerns the output of the tracker and the interpretation of this output by the Battle Planning module of Sim88. The present tracker produces trajectory parameters which are then used by the Battle Planner to predict positions and select engagements. The assumed 'complete knowledge' predict ahead used in Sim88 will need to be replaced by something more fuzzy/realistic.

Sim88 Tracking Formalism



SIMULATION88 TRACKING FORMALISM



TRACKING ALGORITHM FOR PROJECT SWAT

Status: August 1988

Submitted by Jack Liu

Project SWAT is sponsored by DARPA with funding from SDIO. Phase 1 of this program is an 18 month effort to evaluate the applicability of applying DARPA's MOSAIC architecture developed by ESL in supporting the development and implementation of algorithms required for tracking and battle management for SDI. To demonstrate feasibility, the phase 1 project scope is limited to the problem of midcourse track maintenance with the possibility of evolving CSO resolvability and low velocity object deployment. One potential mechanization was developed based on the classical concepts of gating, track splitting, multiple hypothesis track spawning and pruning, and extended Kalman filter prediction and correction. To date (August 1988), scenarios consisting of hundreds of tracks have been successfully run with the algorithm. Single platform tracking has been demonstrated; multiple platform scenarios will be addressed in the near future. No claim is made regarding the optimality of the algorithm or its computational efficiency. This algorithm has been selected on the basis of being representative of the class of algorithms that would need to be considered for development.

The key feature of the MOSAIC architecture is its ability to link heterogeneous (different) processors to address a common problem. MOSAIC provides the capability for extremely high data rates (80 Mbytes/sec in, 80 Mbytes/sec out) between each processor. This is an important feature because of the large number of objects that must be tracked. Sixteen processors can be interconnected in a tightly coupled configuration using MOSAIC. The processors can be parallel machines, algorithmically specialized processors, or general purpose computers and workstations. MOSAIC provides the capability to utilize the power of parallel machines and algorithmically specialized processors more effectively. MOSAIC has a programmable scheduler to control the data flows between processor nodes. The schedules are precompiled.

The SWAT algorithm for midcourse tracking selected for the project utilizes classical concepts. There is a **data source** that generates the scenarios consisting of the objects to be tracked, the sensor platforms, uncompensated stars, and system error sources. The **data/track associator** allocates the sensed data to existing tracks and spawns and prunes tracks when appropriate. The **extended Kalman filter** updates tracks and generates gates for use by the data/track associator. The **track file** maintains the databases of true object tracks and algorithm generated tracks, emulates the process of field of view extraction, and produces indicators of algorithm performance. These components are being allocated to separate computers connected to MOSAIC. The data source is being implemented on a Sun 4; the data/track associator, on a Convex; the trajectory estimator, on a Warp; and the track file, on an Encore. These computers were GFE for the project. This configuration was chosen for convenient attachment to MOSAIC; it is not necessarily an optimal configuration for an algorithm development test bed.

The data source consists of a scenario generator, a propagation model, and a sensor model. The scenario generator specifies the objects and their flight parameters and the sensor platform configuration.. The propagation model provides selectable dynamics propagation in a non-spherical earth gravity field of up to J_6 . The sensor model has the capability to implement an "all-seeing" sensor, a staring sensor, and a coolie hat sensor. The data can be provided to the tracking algorithm in terms of either sensor focal plane azimuth/elevation coordinates or platform centered inertial right ascension/declination coordinates.

The data/track association algorithm selected utilizes a multiple hypothesis track spawning and pruning approach for midcourse track maintenance. Therefore, it is a multiple scan approach,

the scan level being variable and affected by a redundancy parameter and the density of objects. Each track gate is determined by the predicted track uncertainty projected on the sensor focal plane. The closest detection within the gate is processed for updating the track. Other detections within the gate are used to spawn new tracks. Detections within overlapping gates are utilized by all affected tracks. Rectangular gates are used. Tracks are scored based on the log likelihood function of the detection relative to the prediction plus a maintenance bias. The scores are cumulative. Tracks that have no detections are degraded. When a set track redundancy level is reached, the low score tracks are deleted. Tracks that are very close in terms of estimated object state and uncertainty are merged probabilistically based on their track scores.

The track estimator is a standard extended Kalman filter predictor/corrector. The state dimension is 6 (3 positions and 3 velocities). Predicted state estimates for determing the gate position are generated by Runge-Kutta solution of the differential equations of motion. The earth gravity model is selectable up to J_6. The gate size is determined by propagating the state estimation errors using the filter dynamics partials, projecting this on the sensor plane, and adding filter assumed sensor jitter.

The track file emulates two functions. The first is to perform trackfile database management for the tracking algorithm, and the second is to evaluate tracking performance by relating the algorithm generated tracks to the objects produced by the scenario generator. Algorithm generated tracks are kept in a geometrically organized database to enhance access of tracks in the sensor's field of view. Various tracking algorithm evaluation functions are being developed with a preliminary set currently in place.

Tracking Algorithm Survey

Submitter's Name: Oliver E. Drummond Date: 12 June 1989
Submitter's Company: Hughes Aircraft Co.
Submitter's Phone: (213) 616-2624
Submitter's Address: 2141 E. Rosecrans Ave., MS: E52/D223
El Segundo, CA 90245
Author's Names: Oliver E. Drummond & Samuel S. Blackman
Author's Company, Phone & Address: See above.

Algorithm Title: **Multiple Sensor Cluster Tracking**

Sponsor: Hughes Aircraft Co. IR&D
Developer: See authors above.

Abstract/Summary

These algorithms provide a means to track multiple clusters of objects and take advantage of data from multiple sensors. Multiple sensor data is especially useful with passive sensors because clusters can appear to cross or overlap in the field of view of a single sensor.

The cluster tracks can be used to initiate tracking of the individual objects when the closely spaced objects are resolved. These algorithms use only a fraction of the computing resources normally required to initiate and maintain tracks on individual objects after deployment.

These cluster tracking algorithms were conceived to overcome the difficulties and meet the unique needs posed by the SDI target threat. Cluster tracking is challenging because a cluster, as seen from different platforms will differ significantly. The apparent size and shape of a cluster varies from sensor to sensor. The apparent number of objects in a cluster can also be different for each sensor.

These algorithms are based on new concepts that permit the estimation of not only the cluster centroid position and velocity in inertial space but also the cluster "extent." The extent establishes the cluster size and shape in inertial space, not just relative to a particular sensor.

A major feature of these algorithms is that data from multiple sensors is combined in a natural way. As a consequence, the approach is quite general. It is applicable to passive or active sensors or a combination of both. This approach is also applicable to tracking extended objects, such as plumes, and may be useful in other applications such as tracking a salvo launch.

1. CONTEXT

These algorithms were designed to provide a means to track clusters during the early midcourse stage of SDI, starting with deployment of clusters. In this context, a cluster might consist of a reentry vehicle and attendant objects, including decoys. Immediately after deployment of a cluster, many objects will not be resolved because of sensor resolution limitations. Thus initially, the cluster might appear from the sensor data to be a single "clump" (unresolved closely spaced objects) or a small group, namely, a collection of clumps and individual targets.

It would be difficult and too demanding of a processor to track each individual target or clump during the early midcourse phase of SDI after deployment of a cluster. After a cluster is deployed, many new objects will continue to appear as more and more objects become resolved. As a consequence, the number, type and size of resolved objects of a cluster can change rapidly during this early phase. To compound the problem, the number and relative location of objects in a cluster would vary from sensor to sensor. In addition, the targets are initially so close that the high density can make individual target tracking impractical. See reference 1 or 2 for a discussion of these difficulties.

These cluster tracking algorithms permit efficient tracking of cluster position, size and shape in spite of crossing and overlapping clusters and the other difficulties outlined. This permits a smooth transition from post boost vehicle (PBV) to individual target tracking without overloading the processors. The algorithms are intended to be compatible with birth-to-death tracking, which has been conceived as a practical solution for the computationally intensive SDI midcourse tracking task (see reference 1 or 2). However, these algorithms should be useful with minor modification for other applications.

While this approach could be used for independent sensor tracking as in the Type I processing chain, the best use is for multiple sensor tracking. This approach can be used effectively in any of the multiple sensor processing chains, namely, Types II, III and IV. The four types of processing chains are defined in the appendix.

This approach should be most useful in the early midcourse phase of SDI, before almost all the targets are resolved. In that SDI phase this approach would be useful in three ways: 1) to provide cluster handoff information from one sensor system to another, 2) to combine cluster data from different systems, and 3) to combine data from multiple sensors within a specific sensor system. Thus SSTS, GSTS, GBR and AOS are applicable sensor systems.

The inputs to the tracking algorithms are assumed to be the observations as provided by the sensor signal processor. Either or both active and passive sensor signal processing data can be used effectively. For a passive sensor, the tracker could use simply angle location data (azimuth and elevation) for each detected signal (observation) without any indication of whether it is a resolved object, extended object or unresolved closely space object.

On the other hand, additional information such as object intensity, multiple wavelength intensities, and object type and size could be used effectively and should improve tracking performance. The outputs (during most of the midcourse cluster tracking phase) include the estimated position (centroid location), velocity and extent of the cluster (group) in inertial coordinates.

Certain aspects, phases and outputs of this approach are Hughes proprietary so some details have been omitted from this Tracking Algorithm Survey. Also, this is not the only approach being developed at Hughes. Because of the importance of cluster tracking to SDI, other alternative cluster tracking methods are also being explored.

2. **NOTABLE FEATURES**

The critical issue that this approach addresses is the tracking of multiple clusters with multiple sensors. The use of multiple sensor data greatly improves the tracking accuracy and the information that describes or characterizes a cluster.

This cluster tracking is designed to provide an effective and efficient transition from cluster deployment to individual target tracking. Through cluster tracking, the onboard processor size, weight and power can be substantially reduced.

These algorithms accommodate the differences in the data obtained for a cluster by sensors at different locations. This approach is quite general and can be adapted to various types of applications and sensors. It is designed to process multiple clusters that overlap or cross in the field of view of passive sensors. It has been modified and tested for tracking extended objects, i.e. small targets larger than a point source.

The way the extent is characterized, parameterized and processed in this approach is believed to be unique and novel. The extent estimate as formulated greatly facilitates both the processing and the usefulness of the output.

3. SENSOR ARCHITECTURE AND THREAT SCENARIO

The initial design of the algorithms was for SDI midcourse tracking. For initial feasibility testing, the simulation modeled an above the horizon infrared sensor system akin to SSTS with two platforms. The platform altitude was in the range of 1000 to 5000 km, the sensor resolution was in the range of .001 to .005 degrees and the angle measurement error standard deviation was from one-fourth to one-half the resolution.

A typical cluster in the initial simulations was 40 targets (RV's and decoys) with a maximum dispersion rate of 2 meters/sec, nominally. The targets trajectories were generated using a ballistic model including the J2 term. See reference 4 for further details. Further testing with a range of simulation parameters in underway with limited funding.

4. SENSOR MODEL/SIGNAL PROCESSING

This approach can accommodate false and missing observations. For simplicity in initial feasibility testing, clutter and false signals were not included. A simplified signal processing simulation was used that included missing observations due to unresolved closely spaced objects caused by finite sensor resolution. Gaussian errors were first added to the true position for each target in the field of view for a sensor. The resulting clumps and resolved objects were passed to the track processor. The worst case was modeled by not providing observation intensity information to the track processor, which received only azimuth and elevation angles and time for an observation. The range of frame times was from five to 100 seconds.

5. TRACK INITIATION

For initial testing the cluster tracks were initiated by track spawning based on the estimated PBV track. The centroid covariance matrix was larger than that of the booster state and the time of cluster deployment was assumed known. The covariance matrix for the extent was set to the identity matrix multiplied by a large scalar value. These initial conditions were then used for track maintenance processing. A more advanced track initiation approach has been designed but requires further development.

6. TRACK MAINTENANCE

In this approach, the cluster is modeled in inertial space as an ellipsoid. This is modeled in two parts, the centroid and the extent. The elements of the centroid state vector are the inertial position and velocity. In this approach the extent is the second central moment in inertial space of the objects in the group.

The extent provides information on the size and shape of the cluster relative to the centroid. The extent not only describes the characteristics of the cluster, but it has a very important function in the track processing. The extent can also play a vital role when making a transition from cluster to individual target tracking.

6.1 TRACK MAINTENANCE - Data Association

The extent is used to determine which observation (signal/object detected by a sensor) belongs to which group. This is a vital function for tracking multiple clusters, especially for crossing or overlapping clusters. Based on the projection of the predicted extent on to the field of view of a sensor, a gate (validation window) is computed for a cluster.

For an isolated cluster, the first and second central moments in 2-dimensions of the observations in the gate are computed. For a Type I, II or IV processing chain, these moments are passed to the filter described in Section 6.2 to update the established tracks. The four types of processing chains are defined in the appendix. For a Type III processing chain, the moments computed from multiple sensors must first be combined and then the resulting estimated moments in inertial space are passed to the filter.

For clusters that are overlapping or crossing in the field of view of a sensor, association processing is needed to resolve the ambiguities. Association algorithms have been designed but require further development.

6.2 TRACK MAINTENANCE - State Estimation (Filters)

This approach employs two filters, one for the state of the centroid and another for the extent. The filter for the 6-D centroid state is a simplified extended Kalman filter. The extent state has been simplified to 6 elements, which greatly reduces the processing load. The filter for the extent is a pseudo-linear filter. The details and unique approaches used in this filter are provided in reference 4.

7. TRACK FILE MAINTENANCE

The track files in the initial tests contained the two states and the corresponding covariance matrices. The advanced versions of this approach retain some additional information for purposes of association.

8. OUTPUT TO USERS

During early cluster tracking, the available data is the number of clusters, the number of objects in each cluster, the state estimates (centroid and extent) and corresponding covariance matrices. After many of the objects in a cluster are resolved, it is practical to transition from cluster to individual target tracking. At that time the available information includes the state vector and corresponding covariance matrix for each target and additional information as needed, such as predictions and a target object map.

9. COMPUTATIONAL REQUIREMENTS

With isolated clusters the reduction in throughput and memory is approximately a factor of 2/N, where N is the average number of targets per cluster. For example, with an average of only 20 targets per cluster the savings in track processing would be roughly a factor of 10. With overlapping clusters the savings is not as much and would depend on the processing accuracy required. Further detailed analysis is required to determine the processing load for a specific application.

10. CURRENT STATUS

Feasibility tests have been conducted successfully for cluster tracking under realistic conditions using practical values for sensor resolution and measurement errors. Earlier testing results with finite sensor resolution but without measurement errors are reported in reference 4. Further limited development and feasibility testing is underway on Hughes IR&D.

While single sensor cluster tracking is relatively mature (see reference 3), multiple sensor cluster tracking has only recently been addressed. Accordingly, new ground is being broken, even in how to evaluate performance as well as how to track clusters. Substantial algorithm development and testing remains to evaluate the various phases and aspects of the approach and then adapt it to a specific sensor application.

11. PERFORMANCE MEASURES

Measures of performance for the cluster centroid can be similar to the usual measures for an individual target. However as formulated, the extent is new and therefore new meaningful measures of performance must be established. One identified measure of extent estimation performance is the ratio of the estimated to actual volume of a cluster. Another measure is the eigenvalues of the estimated relative to the actual cluster second central moment. These measures and some early test results are discussed in reference 4.

12. REFERENCES

1. Drummond, O.E. and S.S. Blackman, "Multiple Sensor, Multiple Target Tracking Challenges of the Strategic Defense Initiative," *Proceedings of the 1st National Symposium on Sensor Fusion*, Orlando, FL, April 1988.

2. Drummond, O.E. and S.S. Blackman, "Challenges of Developing Algorithms for Multiple Sensor, Multiple Target Tracking," *Signal and Data Processing of Small Targets 1989*, Proc. SPIE Vol. 1096, Orlando, Fl, March 1989.

3. Blackman, S.S., *Multiple Target Tracking with Radar Applications*, Artech House, Dedham, MA (1986).

4. Drummond, O.E., S.S. Blackman, and K.C. Hell, "Multiple Sensor Tracking of Clusters and Extended Objects," *Technical Proceedings 1988 Tri-Service Data Fusion Symposium*, Laurel, Maryland, May 1988.

13. APPENDIX - Types of Processing Chains

Four types of processing chains have been adopted by the IDA SDI Tracking Panels for ease of reference. Reference 2 discusses these generic processing chains and summarizes the four types as follows.

Type I: Independent Sensor Processing - Tracks are processed for each sensor independently of the data from the other sensors.

Type II: Hierarchical Processing - Sensor level processing is followed by track fusion. Frame-to-frame association and filtering are followed by sensor-to-sensor processing.

Type III: Observation Fusion - Multiple sensor, observation association and target position estimation is followed by frame-to-frame association and filtering. Sensor-to-sensor processing precedes frame-to-frame processing.

Type IV: Centralized Processing - Observation-to-track association is followed by multiple sensor filtering. Association and filtering is performed on each frame of data from each sensor as it becomes available.

SUBMITTER'S NAME: MICHAEL KOVACICH DATE: MAR 1, 1989
SUBMITTER'S COMPANY: LOCKHEED MISSILES AND SPACE COMPANY
SUBMITTER'S PHONE: (408) 756-8509
ADDRESS OF SUBMITTER: DR. MICHAEL KOVACICH
ORG. 51-90 BLDG 554
LOCKHEED SPACE & MISSILES COMPANY
1111 LOCKHEED WAY
SUNNYVALE, CA 94089-3504

TITLE OF ALGORITHM: SSTS TRACKING AND ASSOCIATION ALGORITHM

SPONSOR: AIR FORCE SPACE DIVISION (AFSD)

DEVELOPER: LMSC, SUNNYVALE

ABSTRACT/SUMMARY

The algorithm processes measurements from a constellation of passive LWIR sensors to generate precision cartesian state vectors on midcourse objects, clumps or groups. The algorithm performs birth to death tracking on RVs, penaids and PBVs. The algorithm also performs group to object tracking.

1. CONTEXT

Track Initiation is performed using the Mono-then Stereo architecture. After Stereo tracks are initiated, track maintainence is carried out using the Centralized architecture. The algorithm is applicable to Midcourse. The inputs are line of sight measurements from multiple LWIR sensors. The line of sights may reference distinct objects or clumps of objects (CSOs).

2. NOTABLE FEATURES

The algorithm uses a Bayesian network architecture to carry out state estimation and association. The algorithm decomposes the association problem into scenes. Each scene is a distinct association problem, either a contact-to-track or track-to-track problem. A variant of the A* search algorithm is used to generate feasible scene hypotheses. The algorithm performs birth to death and group to object tracking.

3. SENSOR ARCHITECTURE & THREAT SCENARIO

The threat scenario driving the algorithm design is the TSCB1 threat. Background clutter is assumed to consist of stars, RSOs, structured background and nuclear redout. All clutter is not assumed to be removed prior to tracking. The constellation consists of 18 satellites inclined to 90 degrees. The RVs and penaids are assumed to be in free fall. The PBVs execute typical laydown trajectories.

4. SENSOR MODE/PROCESSING

The sensor is a scanning, 3 color LWIR sensor. The frame time varies from 5 sec to 30 sec. The scan speed can also vary. CSOs are modelled using a functional model. The interface to the tracker consists of [time, az, el, snr, covariance, extended object indicator, extent parameters].

5. TRACK INITIATION

Tracks are initiated as 2-d tracks. After 4 to 6 updates a passive ranging algorithm is executed to initiate a coarse cartesian track. Cold start initiation and warm start intitiation are performed. Tracks are processed both individually and in a batch. False alarm and new track densities as well as chi-square scores are used to score new tracks. Tracks satisfying the firm track criteria are promoted; otherwise they are deleted.

From a data association standpoint, the track initiation and track maintainence phases are not clearly separable since a variant of a multiple hypothese algorithm is used.

6.1 TRACK MAINTAINENCE - DATA ASSOCIATION

A track is considered to have completed initiation when a precision cartesian state vector is created. This requires that individual mono tracks be formed. Passive ranging creates coarse cartesian tracks and track-to-track association forms precision cartesian tracks completing initiation.

Pruning, merging and clustering are used to control the combinatoric explosion. In addition, scenes are managed to assure that they do not get too large. The hypotheses are scored using a Bayesian approach. A variant of the A* algorithm is used to search the hypothesis tree. Track spawning is managed. Resolution is accounted for in track-to- track association by allowing a contact to be shared by multiple tracks. Stars are eliminated using a star catalog.

6.2 TRACK MAINTAINENCE - STATE ESTIMATION

Triangulation is performed to initiate the precision cartesian state vector. ECI coordinates are used. State vectors are combined when a merge action is declared and the covariance matrix is adjusted accordingly in a manner similar to PDAF. The filter is an extended Kalman filter. The passive ranging algorithm is an iterated maximum likelihood algorithm. Biases are accounted for by adjusting the system noise matrix or measurement matrix.

6.3 TRACK MAINTAINENCE - TRACK PROMOTION/DEMOTION

A Bayesian multiple hypothesis approach is used in data association. The hypothesis scores are used to generate track quality scores for each track. The track quality scores account for clutter density, new track density, miss detection rates and cumulative chi square scores. Thresholds are used to promote or prune tracks.

7. TRACK FILE MAINTAINENCE

Essentially all track data is maintained in the Bayesian network. The network maintains singleton object tracks, singleton CSO tracks, singleton group tracks and formations of these tracks. Critical deployment points are also maintained.

8. OUTPUT TO BM/C3 AND USERS

State vectors for threatening targets are output to BM/C3.

9. COMPUTATIONAL REQUIREMENTS

Computational requirements are discussed in Reference 2.

10. CURRENT STATUS

The algorithm is currently being implemented, in ADA, in preparation for an algorithm demo in the first half of 1989. The algorithm to be demonstrated will incorporate frame-to-frame and track-to-track association, birth to death tracking and group to object tracking on RVs and penaids. Future demos will include PBV tracking.

11 PERFORMANCE MEASURE & RESULTS

The primary performance measures to be used in the algorithm demo are track accuracy, track resolution, track continuity, track time to firm, track purity and track handover performance.

12 REPORTS

Pertinent information and performance data can be found in

1. Technical Operating Report. Processor Development Report. CDRL 008A9. Volume II. Processor Preliminary Development. Section III. Critical Algorithm Development. 31 July 1987. LMSC. Contract F04701-84-C-0102. (Secret)

2. Technical Operating Report. Processor Development Report. CDRL 008A9. Volume II. Processor Preliminary Development. Section I. Processor Architecture Development. 31 July 1987. LMSC. Contract F04701-84-C-0102. (Secret)

3. Design Review Data Package. SSTS Demonstration Design Review Data Package. Tracking Algorithms. 12 October 1988. LMSC. Contract F04701-87-C-0093. (Unclassified)

TRACKING ALGORITHM SUMMARY

SUBMITTER'S NAME: Ming J. Tsai DATE: 14 February, 1989
SUBMITTER'S COMPANY/ORGANIZATION: M.I.T. Lincoln Laboratory
SUBMITTER'S PHONE: 617-981-2858
ADDRESS OF SUBMITTER: 244 Wood St., Lincoln Lab., Lexington, MA 02173

TITLE OF ALGORITHM: A MID-COURSE TRACK INITIATION AND MAINTENANCE ALGORITHM

SPONSOR: SDIO

DEVELOPERS: M.J. Tsai, K.P. Dunn, L.C. Youens, and C.B. Chang

---

ABSTRACT

The algorithm presented is intended to perform functions of track initiation and track maintenance in the exoatmospheric ballistic missile defense scenario, assuming LWIR sensors aboard a set of mid-altitude satellites. The target density is effectively reduced by calling upon an edge tracker initially, which forms track files of edges of clusters. Track initiation is then accomplished by referencing to those edge track files, assuming that targets in the same cluster travel in parallel. The iterative-least-square (ILS) filter employed in the track initiation process is itself initialized in a special manner in order to handle conditions of high target density and poor observability. Tracks initiated by two sensors are merged to provide precise state estimates to the extended Kalman filter that is used to carry out the track maintenance task.

1. CONTEXT

The algorithm is applicable to the mid-course tracking problem. It contains two major components: track initiation and track maintenance. The track initiation portion basically follows the functional chain of *mono-then-stereo* and the track maintenance portion follows the *centralized sensor-by-sensor* approach.

Inputs to the algorithm are angle measurements from either a single or multiple passive optical sensors. The outputs are track files and associated state estimates.

2. NOTABLE FEATURES

In the track initiation stage, track files are formed based on the assumption that targets within the same cluster travel in parallel. Reference trajectories are first established for edges of clusters using an edge tracking algorithm. The state associated with a track file is estimated using an iterative-least-square filter which is initialized in a special manner.

In the track maintenance stage, a centralized extended Kalman filter is used to continue tracks based on angular measurements from multiple sensors.

3. SENSOR ARCHITECTURE & THREAT SCENARIO

The algorithm can handle satellite-borne or probe-borne sensors. Sensor parameters such as resolution, sensitivity, measurement precision, bias, and detection threshold, are adjustable.

Structured background clutter and background stars are assumed being removed prior to tracking. The only noise effect considered is the additive Gaussian noise.

CSOs exist in the threat. Targets in the threat are all ballistic and non-maneuverable.

4. SENSOR MODEL/PROCESSING

A sensor functional model is used to generate angular measurements from

a scanning passive optical sensor. Included in the functional model are effects of random noise, bias, and CSOs. The CSO modelling is particularly elaborate. The scanning mechanism is also simulated.

5. TRACK INITIATION

Tracks are initiated from a cold start. An edge tracker is first used to establish track files for edges of clusters. All targets are then initiated in either a closely-coupled or a loosely-coupled parallel search mode, assuming targets within the same cluster travel in parallel. Five frames of data are usually processed in a batch mode. Tracks are split initially and then pruned based on the nearest neighbor rule or the criterion of minimum residual chi-square. A complete trajectory is generated for each track file by calling upon an angle-only tracking algorithm.

6.1 TRACK MAINTENANCE - DATA ASSOCIATION

A simple nearest neighbor rule is used for data association during the stage of track maintenance.

6.2 TRACK MAINTENANCE - STATE ESTIMATION

An EKF is used to maintain tracks. The filter recursively updates the state estimate of each track file using angle measurements from multiple sensors. The ECI coordinate is adopted. Only white Gaussian noise is considered for the filter.

7. TRACK FILE MAINTENANCE

A track file maintains three categories of data: (1) the current state estimate, its covariance matrix, and the chi-square value, (2) status of CSO's, and (3) intensity measurements from three color bands.

8. OUTPUT TO USERS

The current algorithm does not explicitly provide outputs to users.

But, signature data associated with each established track file could be sent to discrimination algorithms, and metric data of all objects could be sent to BM/C^3.

9. COMPUTATIONAL REQUIREMENTS

The algorithm was implemented in an IBM mainframe computer. The computational requirement depends on the number of targets and the number of frames to be processed.

10. CURRENT STATUS

The algorithm has been implemented, tested, and run in a number of simulated threat/sensor scenarios. Currently, it is being integrated with discrimination algorithms and radar tracking functions.

11. PERFORMANCE MEASURES & RESULTS

Two scoring methods, the target oriented measure and the track-file oriented measure, have been used to evaluate the performance of the algorithm. The most critical factors which influence the performance are the target density and the sensor resolution.

12. REPORTS

1. C. B. Chang and L. C. Youens, "An algorithm for multiple target tracking and data association," Technical Report 643, Lincoln Laboratory, M.I.T., June 1983, DTIC AD-A1131313.
2. C. B. Chang, K. P. Dunn and L. C. Youens, "A tracking algorithm for dense target environments," Proceedings of American Control Conference, San Diego, CA, June 1984.
3. M. J. Tsai, L. C. Youens, and K. P. Dunn, "Track Initiation in a dense target environment using multiple sensors," Proceedings of SPIE conference on Digital Signal Processing, Association and Tracking of Point Source, Small and Cluster Targets, Orlando, Florida, March 1989.

TRACKING ALGORITHM SUMMARY

Submitter's Name: Thomas R. Blackburn Date: March 8, 1989

Submitter's Company/Organization: McDonnell Douglas Space Systems Company

Submitter's Phone: 714/896-3626

Address of Submitter: 5301 Bolsa Ave., Huntington Beach, CA. 92647
Mail Station 13-3

Author: Same

Title of Algorithm: Integrated Correlation, Track

Sponsor: IRAD

Developer: T. R. Blackburn

Abstract/Summary

The algorithm is applicable to two passive sensors tracking individual objects in the late midcourse phase. It correlates observations between sensors "up front" before attempting track initialization.

1. Context

   The processing chain is Type Iv (?). The algorithm assumes a cold start. It includes track initialization and maintenance. It is applicable to the late midcourse phase, when objets are resolvable but before any atmospheric influence at reentry. It assumes two passive sensors in exoatmospheric ballistic trajectories are tracking the target objects during the same time-period. The outputs are 3-D target track files.

2. Notable Features

   The algorithm is much faster than alternatives we have pursued or heard about. It initiates track files relatively rapidly, and its correlation mistake ratios are low. It also corrects for bias and navigation errors in the observers.

3. Sensor Architecture and Threat Scenario

The threat consists of thousands of objects in ballistic trajectories. The incidence of unresolved objects is low. The sensors are passive scanning infrared sensors deployed in exoatmospheric ballistic trajectories.

4. Sensor Model/Processing

The Lincoln Laboratory sensor functional model[1] is used. The sensor is a generic scanning IR sensor. The frame time is variable. One wave band is used. The sensor-signal processor supplies line-of-sight, LOS-rate, observation time, and estimated measurement accuracy to the data processing function. Attitude bias errors and navigated position and velocity errors are included. Detection is based on signal to noise ratio.

5. Track Initiation

Cold-Start Track Initiation is performed. The track files initiated are 3-D. A scan-to-scan correlation is done on two frame cycles of data. These paired observations are interpolated to a common time. The interpolated angles are matched between sensors with a "pseudo-elevation" angle[2]. With this correlation four observations are now correlated. These are used to initiate a track file with a square-root information filter. A Householder algorithm is used to solve for the state vector. Multiple hypotheses are not generated. On the first two scan cycles scan-to-scan correlation is performed by estimating line-of-sight rate from the average LOS rate taken from a sector by the signal processing. Once one or more track files have been established the line-of-sight change between two frame cycles is estimated by assuming the object in track corresponding to the observation nearest to the observation in question is at the same range and has the same velocity as the object corresponding to the observation to be correlated. Sensor-to-sensor correlation is performed by matching the pseudo elevation angle generated from the interpolated line-of-sight measurements taken from two frame cycles of data.

[1] C. B Chang and K. P. Dunn, "A Functional Model for the Closely Spaced Object Resolution Process", Lincoln Laboratory Technical Report 611, 20 May 1982.

[2] Blackburn, T. R., "A Practical Correlation Test for Cooperative Passive Optical Sensors", AIAA Journal of Guidance, Control and Dynamics, Vol. 6,, No. 1, Jan-Feb, 1983, p. 62.

6. Track Maintenance

6.1 Track Maintenance

Data association in track maintenance relies on the apriori prediction of the next observation inherent in the structure of recursive (Kalman) filters. New observations "nearest" to the prediction are chosen for updating the track file. The nearest distance measure is weighted with the filter error residual covariance.

6.2 Track Maintenance - State Estimation

A U-D mechanization of the Kalman recursive filter is used for state estimation in the track continuation phase. Coordinates are earth-centered inertial. The filter utilizes estimates of the measurement noise level generated in the signal processing. The object tracking filter utilizes a 6-element state vector, containing object position and velocity. A separate estimator is used to estimate the relative elevation angle bias between the two sensors.

6.3 Track Maintenance - Track Promotion/Demotion

If observations fail outside 3-sigma bounds on the error residual four times in a row the track file is dropped. The dropped file is expunged from the inventory of track files.

7. Track File Maintenance

Individual or assumed-individual objects 7-D state vectors are maintained in track files, along with estimation uncertainty covariance matrices stored in upper triangular square-root format.

8. Output to BM/C3 and Users

These questions haven't been addressed. The data of item 7 above is what's available now.

9. Computational Requirements

WE have done most development work on a CDC-860 machine which is rated at about 2.1 mega-flops. It took $4.28X10^{-3}$ sec per object per scan cycle. The computing load increases linearly with the number of objects and frame cycles. This speed is over 100 times faster than the type -II scheme we developed early

in the decade. Recently we have rehosted it on a VAX computer. It functions well almost entirely in single precision on this 32-bit machine.

10. Current Status

The algorithm is in the late conceptual development stage. There are still areas for refinement. It has been tested and debugged running against threats with slightly over 600 objects in them. It has not been integrated with bulk filtering for star and false alarm rejection, etc.

11. Performance Measures

The performance measure concentrated on has been correct observation assignment. The figure of merit is the difference between the number of observation assignments to a file and the number of times the source object assigned to a file the most was assigned to it. The difference between these two numbers is the number of mistakes made. Our scores are running better than 99% against threats furnished us. The tracking filter estimation accuracies reflect Cramer-Rao bounds when files are free of assignment mistakes.

12. Reports

IRAD Program Description 1-221, Optical Sensor Technologies, Appendix-A, MDAC Report MDCQ0931-1, 1989 Independent Research and Development, McDonnell Douglas Astronautics Co., Huntington Beach Division, March 15, 1989

Tracking Algorithm Summary

Name:	Lawrence M. Beyl	**Date:** February 1, 1989
Company:	MindGate Technologies, Inc.	
Phone:	(615) 937-8004	
Address:	Route 1, Flintville, TN 37335	
Title of Algorithm:	Cluster Map Tracking	
Sponsor:	TRW/USASDC	
Developer:	Lawrence M. Beyl	

Abstract and Summary

The *Cluster Map Tracking* approach is designed to treat the problems of multi-sensor, multi-precision, and multi-coordinate system tracking within a battle management system. This problem is resolved beginning with the solution to the transition of thrusted objects into clusters, and their subsequent transition into single object track files. The process by which this is accomplished is called *cluster mapping* and the details of this concept are presented in the paper *Cluster Map Tracking in a Battle Management System* produced by this author for the battle management program's design notes.

The crux of the idea is that there are patterns within a threat that are naturally formed by the objects dispersed from the same PBV and which are heading toward the same target. This collection of comoving objects is called a *cluster* within this discussion and will serve as the keystone to the understanding of the data reduction and parallelization that can be gained from the cluster mapping methodology.

When angle measurements are made of the objects in a cluster, the patterns formed can be traced from one scan of the associated sensor to another by using the previous scan's two-dimensional map of the cluster as a pattern for the next scan's data associations. This simple technique leads to a battle system that is capable of learning as it evolves from single point source clusters, just birthed from a PBV, to fully developed clusters with all objects resolved. This learning phenonomena makes the battle manager tracking system adaptive by nature and robust under a variety of different threat scenarios.

These patterns of angle data measurements are stored from one scan to the next as *cluster maps*. Each cluster has a list of the associated cluster maps, one for each sensor that views the cluster. Notice here that the explicit problems of multi-sensor correlation are not tackled until the cluster maps have been extracted. This interlocking of the cluster to its source data permits each map to be used as a filter to remove the cluster's new measurements from the field-of-view for that sensor. The extracted data set, if determined to be valid, can then be used to replace the existing set as the new cluster map. This new map continues into the extraction process on the next scan, with each new map defining more of the cluster's detailed structure. Each time a cluster map is updated, the associated cluster centroid state is updated via extended Kalman filtering, where the data used in the update is a calculated pseudo-measurement created from the collection of angle measurements that define the cluster map. These updates provide a mechanism by which the cluster centroid state is improved continuously from the many different sensor views.

By using cluster maps in this manner, the multi-sensor correlation problem is first indirectly solved at the cluster level, and then later at the object level. As each cluster map is updated, and if that cluster is deemed by the battle system to be subject to object tracking, the estimated measurement data is passed on to the object tracking algorithm. Because the cluster mapping process has already divided the threat into clusters, this process can be accomplished in parallel with specialized hardware. If it is the first time that data is being passed to the object tracking system, then an initial state vector and covariance is formed for each object within the cluster. This is done through the

centroid state, the angular separation of the objects within the cluster from the centroid, the cluster spread and spread rate, and the angular measurement accuracy.

Once an initial state has been sent, the object tracking algorithm uses that state as the initial track for that object, updating it as time progresses with the new measurements as they appear. Splitting, merging, and other object track phenomena are handled within the object tracking environment, but are always restricted to the domain of the cluster. On subsequent updates of the cluster map, the associated object angle measurement data is sent to the object tracking algorithm, where it is correlated with the limited list of tracks associated with the cluster in question. This process, as it was conceived, will automatically generate multi-sensor correlated tracks for each object within the cluster without actually performing multi-sensor correlation.

1.0 Context

The cluster mapping approach is designed to cover the entire domain of a battle scenario, from the details of thrusted boost-phase tracking to the specifics of engagement-level tracking and the reporting of the needed in-flight guidance updates to the kill vehicles. Because the concept is applied over the entire scenario of the battle, from the birth of objects in the sensor view to their death from a kill operation, it is necessary to specify the details of each phase to appreciate the context in which the method can be applied.

For the boost-phase, the approach requires that the thrusted objects (boosters and PBVs) be tracked with a minimum of 9-12 state elements being developed maintained the PBV. This is necessary so that the deployment seed points for the lethal objects can be calculated via thrusted flight algorithms. The main requirement to track these thrusted objects is two-fold. First, by knowing the details of the PBV's track, it is possible to determine the approximate location at which a cluster of objects was released. This then gives the cluster a tie-in to the PBV that 'birthed' it. Second, with knowledge of the booster track, the launch point can be determined. This allows intelligence (*a priori*) information to be used in the determination of the vehicle type (SS18, etc.). In addition, with explicit knowledge of the booster track, it is possible to make an association between the PBV and the booster, thus providing a relationship between all deployed objects in a cluster to their parent booster and suspected payload.

The details of the boost-phase operation are more critical to the battle planning, engagement planning, and threat assessment aspects of a battle manager than they are to the tracking system. This is because it is still possible to create an initial track for a cluster without regard to its parent booster. This does imply a much more complex system than is currently envisioned for a battle manager. For this reason, the details of how the *a priori* information should be used in a battle management system are not discussed in this paper.

The contribution to tracking given by the boost-phase is that a cluster's centroid state can be initially estimated based on the PBV state at the time of deployment. This warm start approach greatly simplifies the process of track initialization. If PBV association is not available, then the standard method of track initiation is performed, where several frames of sensor measurement data must be gathered before the track can be initialized. In doing this however, the information specirfying the lethality content of the cluster is unknown.

In the post-boost phase, the boosters, PBVs, and clusters are in a state of transition, with the boosters either completely or nearly burned out, the PBVs in the process of deploying their payloads, and the clusters in the process of being formed. In this phase, the battle management system must begin to deal with the multi-sensor correlation problem. First, all tracking performed on the thrusted objects must either be handled through a thrusted track Kalman filter, where the data input is either the raw angle measurement data or pre-processed six-state estimates. In either case, the thrusted states must be updated to maintain the track on these objects. In addition, the

sensor subsystems in the post-boost phase will begin to see clusters emerging from their associated PBVs as they are seeded along the fly-out path.

These cluster reports arrive in the form of single object angle measurements or CSO measurements. In the later case, the measurements report ensembles of objects that are too closely spaced to be resolved by the sensor signal processing system. No assumption is made here about the resolution of the detectors or the resolving capability of such a system, but a few well-known rules should be met. First, the system should be capable of resolving CSOs at the level of 4-5 objects per non-separable zone in the signal domain. Second, the information report regarding CSOs should include quality of fit, estimated number of objects, and some amplitude estimate or ensemble to represent the intensity of the clump. A system using sensor technology limited to single and dual object resolution, with the remainder regarded as a clump, will certainly lead to system-wide failures (not only in tracking).

The angle measurements arriving from the sensor must also be specified in a useful form. This form is dictated by what the battle management system intends to do with the data in question, and not by the simplest method for any one sensor subsystem. Since the main goal is to merge the multi-sensor views, establish viable tracks, and use the collection of irradiance data to discriminate the lethal objects, the system must be cast into a common frame of reference. This is accomplished by specifying, as a requirement, to each sensor subsystem, that it render all of its angle measurements and all of its state calculations into a selected Earth-Centered-Inertial (ECI) coordinate system (no specific definition required; only that there is unique definition). This rotation to the global inertial frame is necessary for the battle manager, in order for it to eliminate the data's dependence on the sensor's pointing, IMU, and other detailed sensor system operations. In rotating the data to this common frame, the sensor must correct for all of its known variances. This includes changes that would naturally result from a non-inertial coordinate system in those sensors that are not inertially directed.

Now, with the data arriving from each sensor in a common frame of reference, the cluster map tracking system is equipped to deal with this data, whether it be in the form of angle measurements or in the form of six-states. It is more productive and accurate, however, for the system to operate off of the transformed raw data, which has been unaffected by any on-board sensor tracking algorithms. Raw data is important to the overall performance of the multi-sensor correlation problem, as data modified by tracking tends to in-build non-linearities that cannot always be overcome in stereo-viewing problems.

As the system moves into the midcourse-phase, the angle measurement data continues, but from other sensor subsystems. It is these sensors that tend to supply the required information needed to refine the assessments and improve the guidance updates of the interceptors (which are typically in the air at this point in the scenario).

In the cluster map approach to tracking, the raw angle data is honed at each step of the way by the cluster's 2D cluster map (one for each sensor view). This data reduction method is based on the *pattern matching* of this 2D image to the subsequent data set, and then using the selected objects from that data set as a replacement image, perpetuating the operation until its conclusion. In performing this pattern matching, the system naturally subdivides itself, at the cluster level, into parallel operations, greatly increasing the possibility of a true parallel architecture within the battle manager. In addition, as each cluster map set is selected, it is passed on to the object tracking subsystem, where again it can be operated upon, in parallel, to render the desired object track files.

The overview of the cluster mapping approach is shown in Figure 1. Here the diagram shows the inherently parallel nature of the approach, segmenting the sensor data pipeline into parallel paths and the clustered object pipeline into parallel paths. This diagram is designed to demonstrate the possible locations for specialized hardware in the cluster mapping approach.

Overview of the Cluster Map Tracking Method

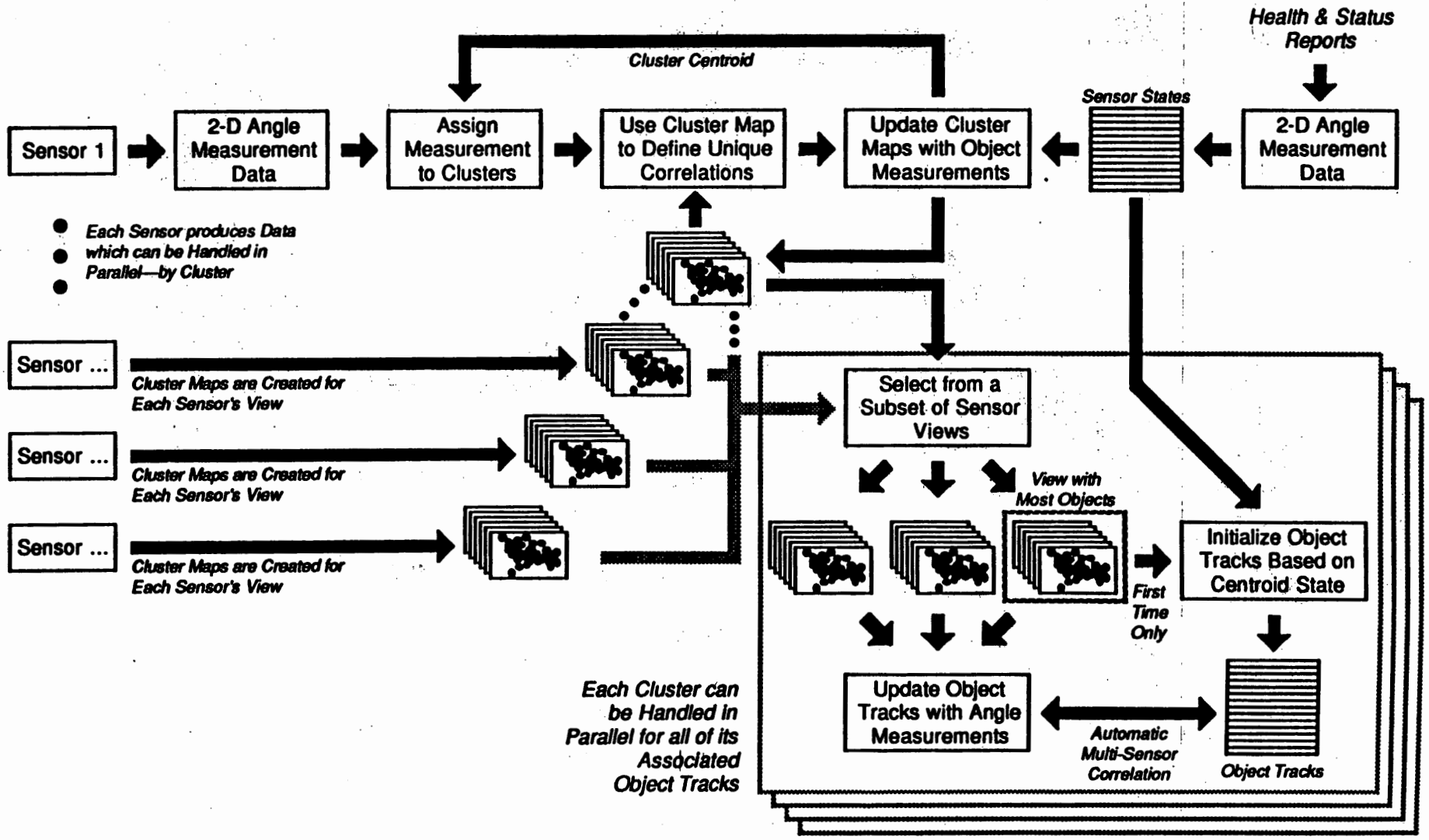


Figure 1

Cluster Tracking Concept Operating in Parallel per Sensor

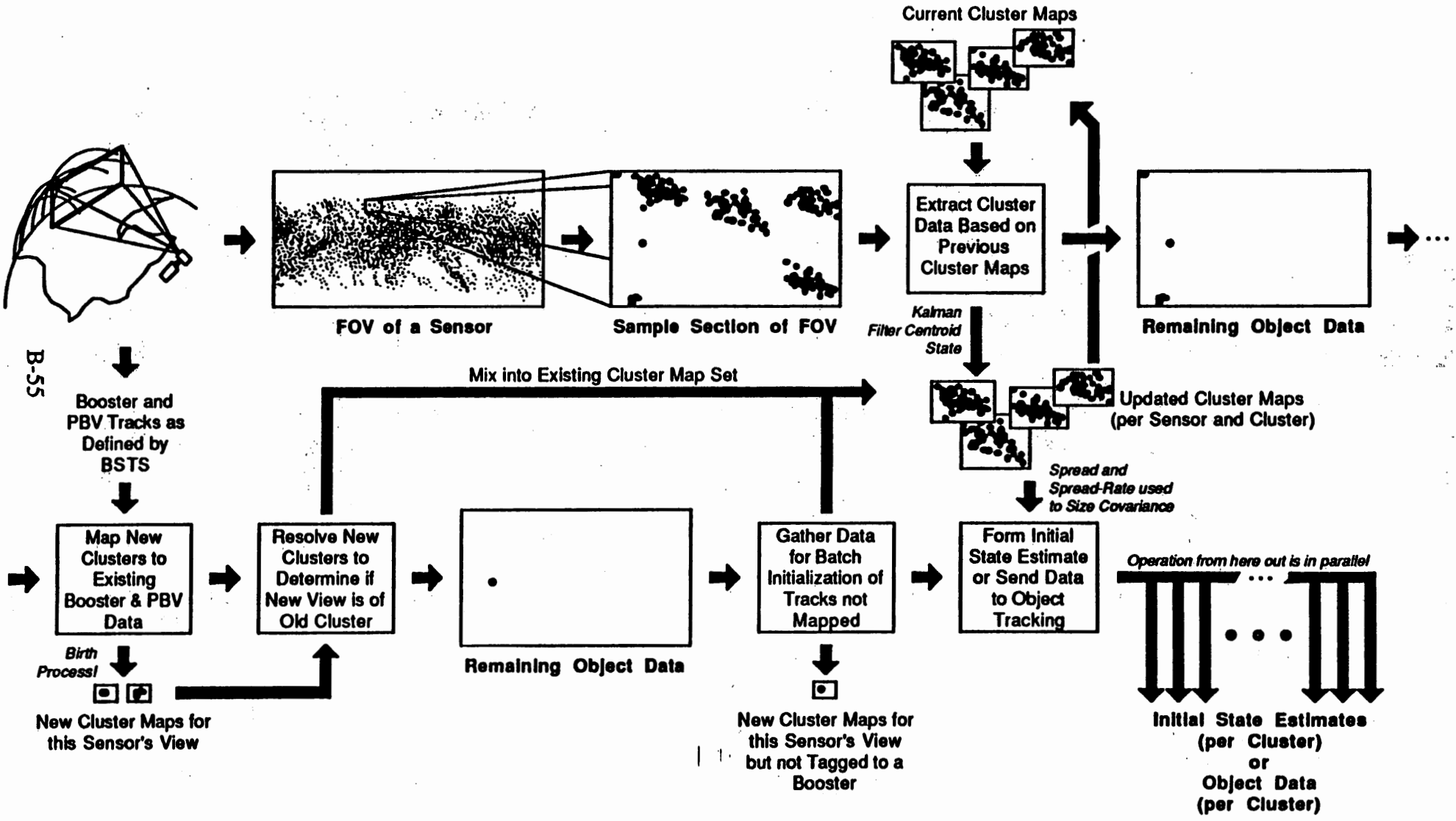


Figure 2

Multi-Sensor Object Tracking Concept Executed in Parallel per Cluster

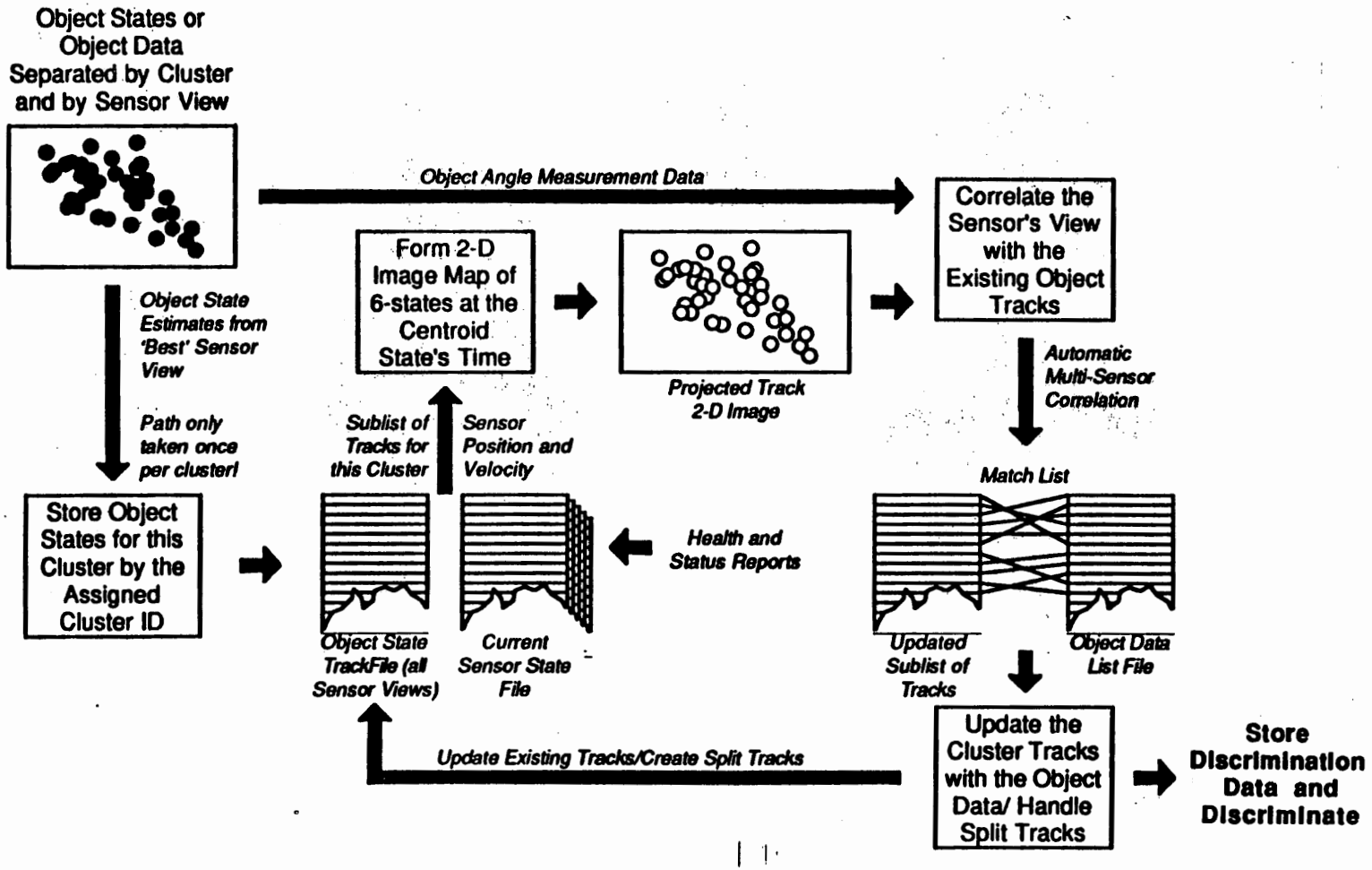


Figure 3

The details of the cluster mapping method as applied to cluster tracking is shown in Figure 2. Here, the diagram shows how the methodology hones the data down from the full sensor's view to the cluster maps, and then to the new clusters. It also demonstrates how the data is parallel piped, by cluster, into the object tracking system.

The details of object tracking under the cluster mapping approach are shown in Figure 3. Here, the process is demonstrated to be restricted to the data associated with all the sensor views of a single cluster. The mechanism for update is simple 2D correlation, by which the algorithm is capable of updating the tracks and simultaneously perform multi-sensor correlation at the clustered object level.

2.0 Notable Features

The cluster mapping approach is capable of

- Six-state tracking of clusters in a threat (over multiple sensors),
- Correlating newly formed clusters to their parent boosters/PBVs,
- Creating initial object track states,
- Updating object track states,
- Reducing front-end correlation by matching object data to clusters,
- Reducing front-end correlation by pattern matching object groupings to existing clusters, and
- Performing multi-sensor correlation as an automatic consequence of object track correlation.

3.0 Sensor Architecture and Threat Scenario

The assumptions made in the cluster mapping approach to a battle management system are that the sensor has done its best to remove clutter, star fields, biases, etc. and that the sensor system is capable of resolving more than just dual CSOs (typically 4-5 object CSOs).

No assumptions have been made about the threat density, configuration, or individual vehicle capabilities. The number of objects has been estimated to be above 100,000.

4.0 Sensor Model/Processing

The basic information expected is:

- The angle values associated with the rotation of the azimuth and elevation into the common ECI coordinate system,
- The 3-element symmetric covariance associated with the errors in that coordinate system,
- The time of measurement,
- The irradiance data for the different colors of the focal plane array,
- A status to estimate the nature of the measurement (single, dual, triple, etc.), and
- The sensor's unique ID number in the battle system.

5.0 Track Initiation

There are two forms of track initiation in the cluster mapping methodology, cluster centroid track initiation and object track initiation. In the case of cluster centroid track initiation, there are also two methods by which initialization occurs, PBV-seeding and cold start.

5.1 Cluster State Initiation

In general, the preferred method of centroid track initiation is to seed the track from a PBV at the time of the cluster's release. It is at this point in time that the PBV state can be directly transferred as the centroid state of the associated cluster. In assigning the PBV to the cluster, other vital (to the battle manager) information can also be transferred for use by other algorithms. The associated state initiation process is accomplished by one of two means. If the PBV can be flown directly to the point (to within some tolerance) at which the cluster is first observed, then the PBV state can be directly used (with only minor time adjustments to the state) as the cluster's centroid state.

On the other hand, if the cluster detection is sufficiently late, and the cluster cannot be associated with any known PBV track during its deployment cycle, then a cone of dispersement must be defined to permit the traceback of the cluster's angle position to an appropriate PBV parent. This cone is defined by projecting the state at the estimated time of the first deployment forward for a fixed length of time, and then projecting the state at the estimated time of the last deployment forward a related fixed length of time. These two projections form a cone-like structure, whose apex angle represents the deviation of the PBV over its seeding cycle, and whose height is related to an estimated maximum time-of-flight before the cluster will be observed. If the cluster lies within this cone and can be uniquely assigned to a PBV, then it is assigned to the associated PBV and its parent booster.

If the cluster cannot be assigned to a cone or if a sensor's first viewing of a cluster is sufficiently late to make associating it with a PBV inappropriate, then cold start initiation is used. In this approach, all object data rejected from the basic pattern matching correlation is collected as potential new clusters. Those groupings which cannot be correlated to an existing PBV track are then compared to the existing cluster centroid tracks. This comparison is made because it is possible for a newly arriving sensor to view, for the first time, a cluster, that has been seen by another sensor and is already in cluster track .

If a correlation is made, then the associated track is updated as a track maintenance operation. The remaining object measurements that could not be properly correlated are then collected over 4-6 scans of data. This is done on a per-sensor basis as there is no stereo correlation applied to raw data. After it is determined that a sufficient number of data sets have been collected, windowing is used (there are only a few objects to consider) to eliminate obvious miscorrelations, a 4-point fit is calculated and an initial state is estimated. Once each state is created, that 6-state is re-checked against possible centroid states to determine if a misclassification has occurred. This re-checking operation is performed periodically to purge the system of remnant tracks that really belong to other clusters.

If there are several objects within grouping distance, then these objects, and their new state estimates, are used to form a new cluster and a new cluster centroid state. Although this later method will create a valid cluster track, it does not associate the data needed from the parent booster to the cluster. This lack of detailed information from a cold started track will affect the performance of the remainder of the battle manager.

5.2 Object State Initiation

The second aspect of track initiation occurs at the object tracking level. Here, the state estimates are initiated based on the centroid state estimates of the encompassing cluster. This is accomplished by first determining which of the cluster maps (or sensor views) associated with this cluster contains the largest number of objects. This map, by definition, defines the view of the cluster where most of the objects are resolved and was chosen because it will render the largest number of initial tracks. This supports the philosophy that it is easier to dispense with old tracks than create new ones from scratch.

It is this view of the cluster that is used as the source definition of the object states within the cluster. The object states are formed by first moving the associated sensor state and cluster centroid state to the time of measurement for the object of interest. At this point, the centroid state has been rendered into the same angle space as the angle measurement for the object. The change in angle required to get from the centroid to the object is then calculated and used as a lever-arm to determine the deviation in position from the centroid to the object. This deviation can only specify the two position elements that are perpendicular to the line-of-sight from the sensor to the centroid. The third position is determined as the range along the line-of-sight and is taken to be the same range as found from the sensor to the centroid state. The velocity estimate is defined to be the same as the centroid's state velocity.

With the six-state of an object in the cluster determined, the error in that estimate must also be calculated. This error will represent the covariance of the object state and can be used in future track maintenance filtering operations. The error in the direction perpendicular to the line-of-sight is determined by the error in the centroid state in that direction and the angular measurement error of the data as defined by the sensor. The error in the range direction is determined by the error in the centroid state in that direction and the estimated spread rate of the cluster (a parameter calculated in the cluster mapping process). The error in velocity is determined based on the covariance of the centroid state for velocity and the estimated spread rate of the cluster.

Once the state for each object within a cluster is estimated, it is not re-calculated by the cluster track system. Also, newly formed object tracks, resulting from splitting, are seeded from the original state prior to the split, thus eliminating the need to perform cold-start initialization on any new object tracks. The remainder of the operations performed on this object are determined by the track maintenance cycle. This method of cluster-induced processing guarantees that the required and resulting tracks used during the operation are related only to the cluster, thus allowing the processing to be implemented in parallel.

6.0 Track Maintenance

The maintenance phase of the track occurs at two levels, the cluster track level and the object track level. In the cluster track maintenance, there are two possible problems that must be dealt with: the problem of simply updating an existing track file, and the problem of adding a new sensor's view to an existing cluster's track file. This later problem is equivalent to multi-sensor correlation at the cluster level and is dealt with explicitly in the cluster mapping approach. This approach also gurantees that the multi-sensor correlation problem never actually arises at the object track level, where it occurs as a natural consequence of the correlation process.

6.1 Data Association

The association of the new incoming data to existing tracks can be broken into the association of raw angle measurement data to cluster tracks and the association of cluster mapped angle measurement data to object tracks.

In the case of clusters, the angle measurement data arrives from the sensor, is queued, and is eventually processed to determine to which cluster(s) the measurement may belong. This is accomplished by rotating the cluster centroid state into the frame of reference of the sensor at the time of the measurement. In doing so, the centroid position is rendered into the same angle coordinate system as the incoming data. A quick comparison is made and a separate 'vertical' linked list is built for each cluster that an object measurement could be assigned to. In doing this, the system is creating the list of possible matches that can occur as a result of trying to correlate the data with a given cluster. This list is limited to the extent of the number of clusters in the angle data's immediate neighborhood (as determined by input).

Once each queued object measurement has been correlated to its list of possible clusters, the queue of data is scanned. As it is scanned, each measurement is 'horizontally' correlated with the other measurements that are also attached to a common cluster. These horizontal threads are woven through the node for the cluster under scrutiny, building a series of horzontontal lists of virtual cluster maps. Once these lists have been completed for each object measurement in the queue, they are arbitrated to determine a unique set of assignments for each object measurement. The method used to arbitrate the comparisons is called *pattern matching*.

This technique begins with a trial measurement in the horizontal list of the associated cluster, and starts by associating the object measurement to the first element in the cluster's cluster map (Note: Both the horizontal list and the cluster map are organized in time order to reduce the mis-match possibilities). If a valid correlation is found, bias is removed and the next element is considered. A new bias is calculated from the nearest neighbor within a predetermined tolerance window. If a correlation can be made, then the pattern of the cluster is adjusted by one-half the calculated bias. Once accomplished, the bias for matching the third element to its nearest neighbor is calculated, except that here the window is now smaller (owing to the concept that the pattern should be getting closer to a correct match). If an object is found within this smaller window, the pattern is shifted again, only this time by one-third of the required bias. Each succeeding entry in the list is considered in this manner, with smaller windows and smaller bias corrections (where the corrections applied are inversely proportional to the number that have been previously correlated). This process continues until either a predetermined number of matches have been reached, the list is exhausted, or there is no match within one of the target windows.

If the process is unable to find a match, the algorithm shifts to the next entry in the list as the start and repeats the above steps. At the end, either a match was found that satisfied the window reduction criterion, no matches were found, or a list of partial matches was found that satisfied the criterion up to some number of matches. In this later case, if a minimum number of matches is satisfied, the algorithm will declare this partial match list as the correct assignment list. In this case, or in the case of a list which did satisfy the criterion, the remaining matches are processed by first removing all unique matches, isolating all object measurements or cluster map entries without a match, and then performing a chi-square optimization on the remaining mutiple matches to optimally resolve their multiple correlations.

If there exists a situation where multiple horizontal lists of object measurements are uniquely trying to match to a single cluster map, then that cluster map is scrutinized for the possibility of cluster splitting. One indication of this might be the lack of any object measurements near the calculated centroid. This can occur if, for example, a bimodal population forms and the centroid is calculated to be in the empty region between the two populations. In this case, the system assumes a split has occured and then uses the centroid state, rotated appropriately to each of the new centroid positions, to generate the new states for the split clusters.

Once a unique assignment is made, the list of assigned object measurements is redirected from the object measurement queue to the cluster map pointer for that sensor and cluster. In doing so, the space allocated to the existing cluster map is freed and the existing pointer is set to point to the new list, which now becomes the new cluster map. Finally, the data associated with the new cluster map is merged into a pseudo-measurement and this is used to update the cluster track with a extended Kalman filter. This process of update and replacement continues for all new object measurements.

When updating an existing object track, the data must first be correlated to the existing object tracks. This is accomplished very quickly and in parallel (for each cluster) in the cluster mapping approach to tracking. First, because the tracks for the objects within a cluster are initiated based on the object data in the cluster (see section 5), it means that the system will never start a cluster with track files that contain data outside the cluster. Thus, each cluster is independent of the other

clusters and can be processed as separate data sets. As a cluster map is updated (replaced) by the new object measurements, its data is collectively processed as a set of possible correlations to object track files.

To accommodate this, the object tracks associated with this cluster are transformed to the associated sensor's view. This renders the object tracks into a 2D image directly comparable to the object measurement data in the cluster map. It is at this point that a simple correlation operation is performed, with single correlations being removed and immediately processed into updates. Next, correlations involving equal lists of object measurements matched to equal length lists of tracks are optimized via any one of a number of optimization algorithms (Munkres, etc.) to render these short lists into a resolvable set. These resolved equal length match lists are then processed into updates.

The remaining possibilities are: more object measurements than tracks, or more tracks than object measurements. In the case of more object measurements than tracks, the system must examine the nature of the measurements to determine if they represent the splitting of tracks. If this is the case, then a new track is developed to handle the split object and the old track is updated. If more than one track is assigned to an object measurement, then the best match should be taken and the remaining track(s) should be left for later update attempts.

Under no circumstance does the object tracking system ever cold start a track from a sequence of stored, or batched, object measurements. This capability may be considered as a last resort, although it introduces similar heritage problems as discussed in the cluster initiation problem.

6.2 State Estimation

In the case of cluster tracks, once the data list has been used to replace the existing cluster map, the object data in the cluster map is averaged (unweighted) to determine a new pseudo-measurement. The pseudo-measurement covariance for this calculated measurement is also calculated based upon the measurement covariances of each object in the cluster map and upon the current spread rate of the cluster since last update. This change represents the possible entry of mis-matched measurements into the data set. With the calculated measurement and measurement covariance in hand, the cluster centroid state is updated according to a extended Kalman filter. Here, the precision is maintained at 7 digits and a numeric error plant noise (Marquardt parameter) is used to weight the diagonal covariance elements. This helps maintain the required convergence of the algorithm at this precision level and improves the solution steering that occurs in the extended filter.

In the case of object tracks, the object measurement and covariance are directly applied according to a extended Kalman filter. Here the precision is maintained at 15 digits to accommodate the requirements of the angle changes and coordinate transformations.

6.3 Track Promotion and Demotion

As the system maintains track, it handles the birth and death of a track differently depending on whether the track is defined for clusters or for objects. In the case of cluster tracks, the only measure of the track's validity is its existence, for a cluster is valid even if it contains no lethals. Cluster tracks are maintained over the entire scenario of the battle to provide for the entry and exit of new sensors that may view the cluster. Because of this, a cluster track is not terminated until it is determined that the centroid has impacted.

However, cluster tracks do undergo different stages of existence. When a cluster is formed in the battle manager, it is initially marked 'active' to indicate that something is happening. However, this does not activate the cluster for the remainder of the battle manager. When a cluster is properly associated with the parent booster and PBV, it is marked as 'available'. This marking is an

indicator to the battle planning system that a cluster with valid heritage is available for processing. Because of this, the cluster is also assessed to determine its potential impact-point and the associated possible lethality of the cluster mass. This assessed value is also stored in the track file for the planning operation. If the planning function determines that this cluster contains objects of interest to the battle manager, it directs the tracking system to pickup object-level tracking on the objects within the cluster.

At this point, the tracking on this cluster switches from its pure 'data-gathering' mode to the 'object tracking' mode and creates the initial object states for the object tracking operation. Upon subsequent updates to this cluster, the cluster tracking system passes the accumulated object measurements (in the cluster map) down to the object tracking system, where they are correlated. The cluster tracking system maintains this mode of operation on the clusters that it is tracking until the clusters impact the Earth.

In the case of object tracking, the initial status of a track as defined by the cluster tracking system is that the track is active. Once a track is activated, it must receive continuous update measurements from the cluster tracking system. If a specified (by input) number of reports are missed, then the object track is considered for termination. An object track can only be terminated from missing data under two circumstances, it is not lethal or it has been killed.

If an object track is determined to be lethal during its discrimination processing, and if engagement planning has deemed the object to be under engagement, then an expected kill should occur. The track is therefore forcibly maintained until such time as the expected kill does occur, or the expected kill time has elapsed. If an expected kill of the lethal object does not occur, then a miss is declared and the old track file is deleted. If a kill did occur, then the track file will show a loss of data at the expected time of kill. If there is a data is loss, the track file is again deleted. The primary reason for not deleting an engaged lethal track no longer receiving measurement data is that this track information must be supplied to support in-flight guidance updates and homing views. This data and the associated interrogation of the track files would not be possible if the suspected object track and its cluster members were deleted. This preventative medicine greatly simplifies the data management aspects of the weapon controller algorithms at the expense of some minor additional file maintenance on the part of object tracking.

If an object was not determined to be lethal, then the loss of data is sufficient grounds for the termination of the track.

If at any time during the operation, a track which has not been updated for a specified length of time, receives new angle measurements that can be used to update the track, then the track is reactivated and the old discrimination data is purged. This is done to eliminate the possibility of old discrimination data polluting future discrimination.

Lethality determination is the primary measure by which objects are deleted. In the battle manager, an object's lethality is first determined through a batch discrimination process, where 10-20 measurements (from any sensor) of irradiance data and state estimates are used to determine the lethality value. Subsequent lethality calculations are performed on a 4-8 measurement cycle basis, depending on the throughput loading, in an iterative manner so that new sensor inputs can be used to weight the lethality of the object as it progresses in the scenario. This lethality estimate is used as a mechanism to determine when tracks can or cannot be dropped.

7.0 Track File Maintenance

The cluster track file contains:

- The current six-state estimate of the centroid,
- The associated 21-element symmetric covariance,
- The time at which the cluster was first seen,
- The time of the last update,
- A status to indicate the cluster's current mode of operation (normal, a member of the object tracking group, being locally tracked, marked for deletion, etc.),
- The assessed value, its assigned cluster ID,
- The associated parent booster's ID, and
- Some battle planning information maintained from one epoch to the next.

In addition, a sister file is used to maintain pointers to the data lists that make up the associated cluster maps for that cluster. This second file is accessed by cluster ID and by sensor ID. The actual cluster maps are maintained as part of the queueing mechanism used to manage the input sensor data to the battle management system.

The object track file contains:

- The current six-state estimate of the centroid,
- The associated 21-element symmetric covariance,
- The time at which the state was initialized,
- The time of the last update,
- A status to indicate the object's current mode of operation (normal, under engagement, marked for deletion, etc.),
- Its lethality estimate,
- Its assessed value,
- Pointers to the discrimination data (which is kept in another file),
- Pointers to the last discrimination state (which is kept in another file),
- The associated cluster's ID,
- The assigned track ID, and
- The associated parent booster's ID.

8.0 Output to BM/C3 and Users

In general, the battle management system makes use of both the cluster and object tracking data. In the case of the clusters, the battle planning function has direct access to the cluster track files of the cluster tracking system. It is in these files that the cluster centroid states are maintained. These tracks are then used at the battle planning level to facilitate the global battle planning operations and platform-to-cluster assignments.

The object tracks are used by the engagement planning and weapon controller processing. In engagement planning, the lethal tracks are extracted and propagated to the estimated kill point, where proper launch parameters are generated for the assigned weapon (details not given here). Once the weapon has been launched and has reached burnout, the associated weapon controller extracts from the object track files the guidance corrections that must be specified to intercept the lethal object.

In addition, because the object tracking system maintains its tracks in a cluster organized manner which is organized by cluster, the associated weapon controller can interrogate the file to determine which other objects are likely to be traveling with the designated object. This list of comoving objects defines the homing view of the seeker as it moves into its terminal phase of operation.

Thus, although no data is explicitly output to BM/C3, the raw data maintained in the track files of the battle manager, itself, are used by the other parts of the system to determine which cluster to shoot, which object in the cluster to shoot, how to correct for in-flight errors, and what the arrangement of the objects will be during the terminal phase of the kill. All of this is made possible by the data reduction technique in the cluster mapping approach.

9.0 Computational Requirements

The shell for this tracking approach is written entirely in Ada and has been implemented on a Vax 8800. Timing and performance curves are unavailable and have yet to implemented and determined on the tracking subsystem. The implicitly parallel nature of this approach must be tested to determine what gains can be achieved in a specialized architecture that will allow the full parallel operation to be invoked.

10.0 Current Status

Currently, a truth version of the cluster mapping methodology, has been implemented. It performs the overall cluster birth, maintenance, filtering, and pattern matching operations. However, track and correlation operations are not currently implemented at the level of an algorithm, but rather are modeled at the truth level. Future implementations will substitute various track filtering and pattern matching schemes to define the best possible combination of these elements.

11.0 Performance Measures and Results

Individual track performance is based on the specific filtering algorithms used. Overall system performance has not been evaluated for this system.

12. Reports.

The equations, methodology, and the bulk of the discussion of details of the cluster map tracking approach are discussed in the paper *Cluster Map Tracking in a Battle Management System* (contact Gary Abercrombie, TRW, (205) 830-3302).

The MITRE Experimental Version (EVP) BALLISTIC TRACKER

POINT OF CONTACT: J. A. Krajewski
(617) 271-4547
The MITRE Corporation
Burlington Rd.
Bedford, MA 01730

ABSTRACT/SUMMARY

The ballistic tracker accepts as input handover data from the boost-phase tracker and observations from any number of SSTS sensors. It then attempts to maintain PBV tracks and initiate RV, decoy, and debris tracks. Three-dimensioal tracks are extrapolated into the future, and projected onto the SSTS focal plane. The "expected observations" are then associated with the actual SSTS observations using the Hungarian algorithm. The six-dimensional object state vector is then updated with the two-dimensional observations using an extended Kalman filter. Objects, including PBVs, are modeled in the filter as ballistic objects. Due to the relatively small magnitude of PBV thrusts, a PBV can be considered to be a ballistic object with a small perturbation. Additional process noise is added to the filter to compensate for this approximation. RV/decoy observations that fail to be associated with existing RV/decoy tracks are used to spawn new RV/decoy tracks using the nearest existing PBV track, if such a track is sufficiently nearby. After each update, the track library is examined to discover any tracks that should be pruned, based on a user-input number of failed updates.

CONTEXT

This algorithm belongs to the Type IV Sensor-by-Sensor Track Maintenance category described by the SDIO Panel on Critical Issues in Tracking. There is no cold start track initiation capability yet. Tracks are initiated by handover from the Boost-Phase Tracker and by spawning when a PBV deploys an RV or decoy. This algorithm performs track maintainence and spawning of RVs from PBV tracks. Inputs include booster handover data from the boost-phase tracker, SSTS observations, and a set of algorithm parameters. The outputs are object track estimates over time.

NOTABLE FEATURES

The ballistic tracker updates a six-state Kalman filter in Keplerian elements with one sensor's azimuth and elevation data at a time. It accomodates data from any number of sensors.

SENSOR ARCHITECTURE AND THREAT SCENARIO

The EVP can examine tracking performance variations with threat characteristics, sensor configurations, and sensor capabilities. The EVP does not create threat data, but rather treats the threat as an input. The EVP does model scanning sensors for both the BSTS and SSTS.

SENSOR MODEL/PROCESSING

The sensor model is a scanning sensor with user-selectable scan rate. The model includes quantization error in the azimuth and elevation values output and also the effects of platform attitude error. The sensor is assumed to be blinded if viewing any object within a planetary limb defined by the Earth's radius plus a user-input value representing the height of the bright atmosphere. The probability of detection is 1.0, provided that the object is in a position whose geometry permits observation (above the limb and within range). The probability of false alarm is 0. The EVP does not attempt to remove clutter. The sensor model assumes that PBV and RV/decoy observations can be distinguished, but does not model or report brightness to the tracker beyond that distinction. Consideration of aspect angle of the viewed object relative to the line-of-sight between viewer and viewed object are ignored. A single observation is generated when two or more objects fall within the same pixel. The SSTS has a user-input range limit for RVs.

TRACK INITIATION

PBV tracks are initiated when the boost-phase tracker produces a handover. The boost-phase tracker estimates a burnout time, a burnout position, and a burnout velocity. The boost-phase tracker also updates the covariance matrix in the boost-phase Kalman filter to reflect the uncertainty in the estimations. The ballistic tracker accepts the burnout time, position, velocity and covariance data, and initiates a track. The position and velocity of the estimated burnout point are transformed into Kepler elements. The covariance matrix in position and velocity is also transformed into a covariance matrix of Kepler elements. The Kalman filter in the ballistic tracker uses Kepler elements as state estimation parameters. Thus, an estimated burnout state and covariance in the boost-phase tracker are transformed into state and covariance estimates in Kepler elements to initialize the filter.

When an RV observation fails to be associated with an existing RV track, the distances between that RV observation and all existing PBV tracks are calculated. The closest PBV track is then found. If the distance between the unassociated RV observation and the nearest PBV track is less than a user-defined threshold, then the PBV track is used to spawn a new RV track. A copy of the PBV track parameters is made and entered into the track file. This copied PBV state is then updated with the RV observation, and the original PBV state is updated with its associated PBV observation. Thus, the PBV track has been split into a continued PBV track and a new RV track.

TRACK MAINTENANCE - DATA ASSOCIATION

SSTS data are associated with ballistic tracks in a two-dimensional sense. The three-dimensional tracks are projected onto the SSTS focal plane and, using a scanning sensor model, a set of expected observations is calculated. These, together with the actual observations, are sorted by azimuth. The azimuth values in the list are examined to check for any gap in azimuth that may exist that is larger than a threshold. If such a gap exists, then the two groups of expected and actual observation on either side of the gap are treated completely separately. This is done to take advantage of any structure within the data, and to reduce on the processing time required for association. In this way, potential associations between an actual observation on one side of the gap and an expected observation on the other side of the gap are not even considered.

Each azimuth group of expected and actual observations is then ordered by elevation angle. Similarly, the elevations are examined to determine if there are any gaps in elevation angle that are greater than a threshold. Groups that are separated by sufficiently large gaps in elevation angle are then treated separately.

A distance matrix for each separate group is calculated. Each row corresponds to an expected observation, and each column corresponds to an actual observation. Each entry in the matrix is the angular distance between the corresponding expected observation and actual observation. The Hungarian algorithm is then performed upon this matrix to determine the best set of associations. If, after an input number of iterations, the Hungarian algorithm fails to determine the complete solution, the subset of the solution that has been unambiguously associated is kept, and the remaining objects are passed to a greedy-type algorithm to complete the associations. After this process is complete, the associations are checked against a threshold. If the angular distance between the expected observation and the actual observation are greater than a threshold, the association is suppressed, and the actual observation and track are considered to be unassociated in this scan.

An option exists to bypass the Hungarian algorithm, and rely exclusively on the greedy algorithm to perform the associations.

PBVs and RVs are treated separately. First, RV observations are associated with existing RV tracks. Then, unassociated RV tracks are compared to existing PBV tracks to determine which tracks should be spawned. Finally, PBV observations are associated with PBV tracks.

TRACK MAINTENANCE - STATE ESTIMATION

The estimation process utilizes an extended Kalman filter using Keplerian elements as the state variables. The six state variables are updated using two-dimensional observational data from one SSTS at a time. Thus, azimuth and elevation data update the six Keplerian elements.

All objects are modeled using ballistic trajectories. To compensate for the fact that PBVs are not truly ballistic objects, more process noise is added to the filter for PBV tracks than for RV tracks. We believe the ballistic formulation for PBVs to be a good one; since PBV maneuvers are small compared to the total velocity of the PBV, the PBV's trajectory is nearly ballistic.

TRACK MAINTENANCE - TRACK PROMOTION/DEMOTION

Tracks are not promoted and demoted per se, but rather are all considered equally valid until pruned. Each time a track fails to be updated, even though it has been calculated that the track should have been visible to the SSTS, a counter in the track file is incremented. If the track fails to be updated for an input number of successive scans, the track is pruned, and no longer is considered in any future associations. If a track is expected not to be visible to the SSTS, it is not included in the association, and its failed update counter is not incremented. Unclassified test cases with the current SSTS architecture have shown periods of time when some RV's are not seen by any sensor.

TRACK FILE MAINTENANCE

The following information is kept in the track file for each track:

Current state estimate: Keplerian elements:

- Semi-major axis
- Inclination angle
- Eccentricity
- Mean Anomaly
- Argument of Perigee
- Argument of the Ascending Node

Six-by-six error-covariance estimate in the state variables

- Time of the most recent update
- Object type (PBV or RV/decoy)
- Number of successive missed updates
- Track status (active or pruned)

OUTPUT TO BM/C3 AND USERS

The output of the Ballistic Tracker to other BM/C3 algorithms and user includes:

- Number of tracks maintained over time
- Object state estimates for each track over time
- Innovation sequences for each track over time

COMPUTATIONAL REQUIREMENTS

See CURRENT STATUS, below.

CURRENT STATUS

All algorithms described above have been coded. Test cases are currently being run and results are being analyzed. The preliminary software for the ballistic tracker was written in FORTRAN and runs under the VMS operating system on a VAX 8700. In addition, this algorithm is being written in Ada as a component of the EVP Release 4.0 for the SDI National Test Bed (NTB). EVP Release 4 will be submitted to the NTB in April 1989.

PERFORMANCE MEASURES AND RESULTS

Performance Measures:

- Track purity over time (percentage of active tracks with m out of n consistent updates
- Track accuracy over time on a track-by-track basis (distance between estimated state and true object state)
- SSTS sensor coverage over time on an object-by-object basis (which SSTS views a particular object during what time period

- Impact point accuracy over time on a track-by-track basis (distance between estimated impact point and true impact point)

Results:

A very small, unclassified test case (20 RVs) has been run using a notional unclassified SSTS architecture, architecture, primarily as a 'maiden voyage'. Results indicate that SSTS coverage of RVs may be very sparse, even to the point where 0 or 1 SSTS is observing a particular RV at times. The test showed that the Ballistic Tracker can function even with such limited data, but at a degraded level.

Larger tests cases involving the SDIO's Phase One Test Specification (POTS) threat are currently being performed.

RELEVANT REPORTS

In progress.

The MITRE Experimental Version Prototype (EVP) BOOST-PHASE TRACKER

POINT OF CONTACT: J. H. Latimer
(617) 271-4553
The MITRE Corporation
Burlington Rd.
Bedford, MA 01730

ABSTRACT/SUMMARY

The Boost-Phase Tracker uses the inputs from two simulated sensors, associates the sensor reports for each scan into three-dimensional position estimates, and supplies those three-dimensional estimates as input to the tracking filter for associating with existing tracks or initiating new tracks. Three-dimensional position estimates are associated with existing tracks by a Greedy-type method based upon the distances between measured and expected points. Tracks are initiated from a single three-dimensional position using *a priori* assumptions about target trajectories. These algorithms have been coded, and have been tested using subsets of the SDIO's Phase One Threat Scenario (TSCB1-A).

CONTEXT

This tracking tool belongs to the Type III (Sensor-to-Sensor then Scan-to-Scan) Track Initiation category and the Type III (Sensor-to-Sensor then Scan-to-Scan) Track Maintenance category as described by the SDIO Panel on Critical Issues in Tracking. The tracker performs track initiation and track maintenance for boost phase. When booster targets are observed to burn out, the tool hands tracks over to MITRE's midcourse tracker (described separately). Inputs to the sensor model are model truth target trajectories. Output consists of a file of sensor reports including sensor identity, two angles (azimuth and elevation), an epoch of observation, and (for analysis only) model truth target identity. These, together with sensor orbits, are input to the tracker, which outputs tracking data consisting of target state estimates, covariances, and auxiliary (status) data plus track estimates to hand over to the midcourse tracker.

NOTABLE FEATURES

This tracker combines sensor data to derive three-dimensional position data for targets before associating those points to tracks. This algorithm initiates tracks very rapidly, using only one triangulated three-dimensional position plus *a priori* assumptions about the target's velocity and acceleration profile to initialize the nine-vector state estimate for the target track.

SENSOR ARCHITECTURE AND THREAT SCENARIO

The algorithm is designed to examine tracking performance a) against various threats, b) using various sensor configurations, and c) assuming various sensor capabilities (resolution, attitude error, and sampling rate). The algorithm uses the sensor measurements from two sensors of similar capability and in distinctly differing positions.

The two sensors are expected to be located so as to view all the targets to be tracked. Although there are provisions to detect and remove singly observed measurements from further processing, the emphasis is on the estimation process for those targets that actually afford the possibility for stereo-optic tracking. The measurement errors and biases are of two categories: a quantization error in the individual measurements in each of the two coordinates -- azimuth and elevation -- and an attitude error in the orientation of the platform which is constant over one frame, but which varies randomly with a given variance from one frame (scan) to the next. It is assumed that all background clutter is removed prior to tracking.

The boost-phase tracker was intended to use externally supplied threat data. Therefore, there are few if any assumptions about target kinematics embedded in the tracker. The tool has been tested with roughly 50 targets from one silo complex and also with approximately 100 targets from a handful of launch fields.

SENSOR MODEL/PROCESSING

The model used for the sensor is a scanning sensor that reports the two focal plane coordinates (azimuth and altitude) and the times of observations. Sensor resolution and altitude error are assumed to be variable. Measurements reported by the sensor are quantized based on the input resolution. Probability of detection is assumed to be 1.0, provided the object is in a position whose geometry permits observation. The probability of false alarm is assumed to be zero. There is no provision to handle clutter. The sensor emulator will report only on the first occupant of a pixel, even if there are two or more targets present. There is no representation of brightness other than the gross distinction between the brightness of a burning booster, and the relatively lower brightness of anything else. (See the description of the Ballistic Tracker sensor for more discussion of the brightness distinctions between post-boost vehicles and RVs.) Since there is no brightness calculation, no relationship is assumed between the target signature (brightness) and the aspect angle. Frame time is variable; tests have been run with frame periods from one to ten seconds. There is no treatment of wavelength or closely-spaced objects (CSOs).

TRACK INITIATION

Track initiation begins when an object is first observed by two sensors. The preliminary data association (sensor-to-sensor) is identical to the method described below in TRACK MAINTENANCE - DATA ASSOCIATION. Because the sensor observations are asynchronous, it is necessary to allow for the target motion in the time interval between the two observations. The velocity of the target is estimated by identifying the target as one of the existing tracks, and using the track velocity estimate. When an object is first observed, there is no corresponding track, and therefore an assumption is made that the object is close to the surface of the Earth. The velocity of the object is taken to be that of the Earth's spin at the point of intersection of the observation line-of-sight with the surface. Once the velocity is estimated, a reduced three-dimensional position estimate and covariance are used to start the track initiation process.

A reference trajectory is used, together with the reduced three-dimensional position estimate and covariance described above, to generate the initial state estimate for track initiation. This reference trajectory is derived from a three-degree-of-freedom rocket trajectory model in use at MITRE for about four years. The trajectory model allows for a variable pitch angle profile, as well as acceleration magnitude profile. A Newton's method

iterative procedure is used to fit the reference trajectory to the three-dimensional target position. Flight azimuth is assumed. The value of the reference trajectory velocity and acceleration at the point of fitting is used to initialize the track state estimate. In this way, a three-dimensional trajectory estimate is begun. This procedure is performed serially for each three-dimensional position that has not been associated with a track, using the data association algorithm described below. This style of rapid track initiation has the advantage that only one scan is needed to generate a track. The technique works well for those targets that are still relatively near the ground, and with degraded performance for targets initialized at some altitude.

There is no formal method of track promotion or demotion. Instead, the covariance associated with the track state estimate, plus the count of associated three-dimensional measurement points, indicates the level of confidence to be ascribed to each track state estimate. There is no specific provision for cluster track initiation, as in boost phase the closely-spaced-object problem is thought to be less severe than in midcourse.

TRACK MAINTENANCE - DATA ASSOCIATION

Two data association processes are performed as part of track maintenance: association of data from two similar, but asynchronously scanning, sensors to generate three-dimensional object position and covariance estimates; and association of those three-dimensional positions with existing tracks.

In this tracker, the association of images from two sensors takes advantage of the fact that a line-of-sight from a sensor at a known position can be mapped onto the focal plane of another sensor and intersected with images on the second sensor. This is because the positions of the two sensors and the position of the target at the (assume, for the moment, unique) epoch of observation define a plane. Image data from each sensor, plus the direction to the other sensor define a set of planes. The dihedral angles between these planes and some reference plane such as the plane containing the sensors and parallel to the Earth's equator, form scalar keys which are ordered and used to associate the images from each sensor (see Figure 1). When several images lie on, or nearly on, the same plane, it is necessary to invoke further, more involved logic, which we term disambiguation.

Disambiguation is selectively performed on those groups of images not clearly associated by the plane-mapping technique discussed above. This technique relies on the ability of the track library to associate lines-of-sight from sensors at known positions and epochs with tracks in the track file. When a line-of-sight is associated with a particular track, the height of the identified track is intersected with the sight line to give a temporary position estimate, which can then be used to form a pseudo-image at the companion sensor. At the companion sensor, real and pseudo- images are then associated with each other using the angular positions in the plane defined by the sensors confidently and the targets. When these in-plane angles are too close to associate real and pseudo- images, then a third level of discrimination is invoked for those ambiguous associations remaining. This method uses both coordinates of the real and pseudo-images and a Greedy-type method of associations.

The two-sensor data correlation algorithm requires lists of target observations. The assumption is that each list contains observations of the same targets. If the two lists are not of equal length, then the longer list must be examined to determine which observations in the longer list are without corresponding observations in the shorter list. We denote the

algorithm that does this examination the "Discard" algorithm. The algorithm is very simple, and is based on the principle that the two lists of hinge angles are very similar. It looks for a difference in hinge angle exceeding a threshold, and ascribes that difference to an unequal population.

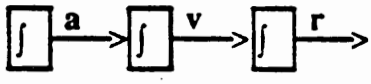
Because of the asynchronous observations resulting from the scanning sensor model, it is necessary to reduce the triangulated positions obtained from the correlated sensor lines-of-sight for the effect of the target velocity. An estimate of the target velocity is obtained by associating a sensor line-of-sight with one of the estimated track positions at the epoch of observation. The relative figure of merit used is the statistical distance (or chi-square) between the observation and its associated covariance and the track and its covariance. With the estimated target displacement in the interval between observations, an adjustment (reduction) to the triangulated position is performed to yield an unbiased estimate of the target position.

For one scan's worth of paired sensor data, a batch of three-dimensional target positions is finally obtained, and the process of associating these points with existing tracks is begun. A Greedy-type algorithm is used to assosiate points with tracks, with upper bounds for associations observed. Thus there are no multiple hypothesis techniques employed and therefore no problem with any combinatoric explosion.

We are not handling stars, false alarms or other forms of stationary clutter.

TRACK MAINTENANCE - STATE ESTIMATION

The Kalman filter uses a nine-state model representing the target position, velocity and acceleration in Cartesian earth-centered inertial (ECI) coordinates. The model is represented as three ganged integrators, and the state is the current output of the integrators. There is no input to the integrator producing the acceleration, hence the model tries to produce constant acceleration. As the acceleration of a booster is certainly not constant, we allow the injection of process noise into the acceleration integrator to permit the variable acceleration of the target to be tracked.

MODEL: 

STATE VECTOR: $\begin{bmatrix} r \\ v \\ a \end{bmatrix}$

STATE TRANSITION MATRIX Φ:

$$\begin{matrix}
1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}\Delta t^2 & 0 & 0 \\
0 & 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}\Delta t^2 & 0 \\
0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 & \frac{1}{2}\Delta t^2 \\
0 & 0 & 0 & 1 & 0 & 0 & \Delta t & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & \Delta t & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \Delta t \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{matrix}$$

Noise measurements are assumed uncorrelated, and the measurement noise level is assumed constant. Crossing targets are not a problem unless one sensor sees the two targets in one resolution cell, in which case there is apt to be a phantom image for that frame.

TRACK MAINTENANCE - TRACK PROMOTION/DEMOTION

Tracks are not promoted or demoted. Instead, the covariance estimate and auxiliary information (such as the accumulated count of associated observations) indicate the confidence to be assigned individual tracks. Tracks are terminated when a user-specified interval of time has passed without the association of observations to a track. Typically, the user might specify three to five scan intervals as the threshold.

TRACK FILE MAINTENANCE

Track files consist of the following:

- Track file serial number (ID)
- Track state estimate (9-vector) representing the three Cartesian components of each of the three quantities: position, velocity and acceleration in Earth-centered inertial coordinates
- Track state estimate error-covariance matrix corresponding to the state estimates (9 by 9 matrix)
- Time of track establishment
- Time of last update
- Accumulated count of associated observations (hits)
- Root sum square (RSS) of last Kalman filter innovation value in position
- RSS of the three current error-covariance matrix diagonal elements in position

COMPUTATIONAL REQUIREMENTS

See CURRENT STATUS, below.

IT0153REFVT0391

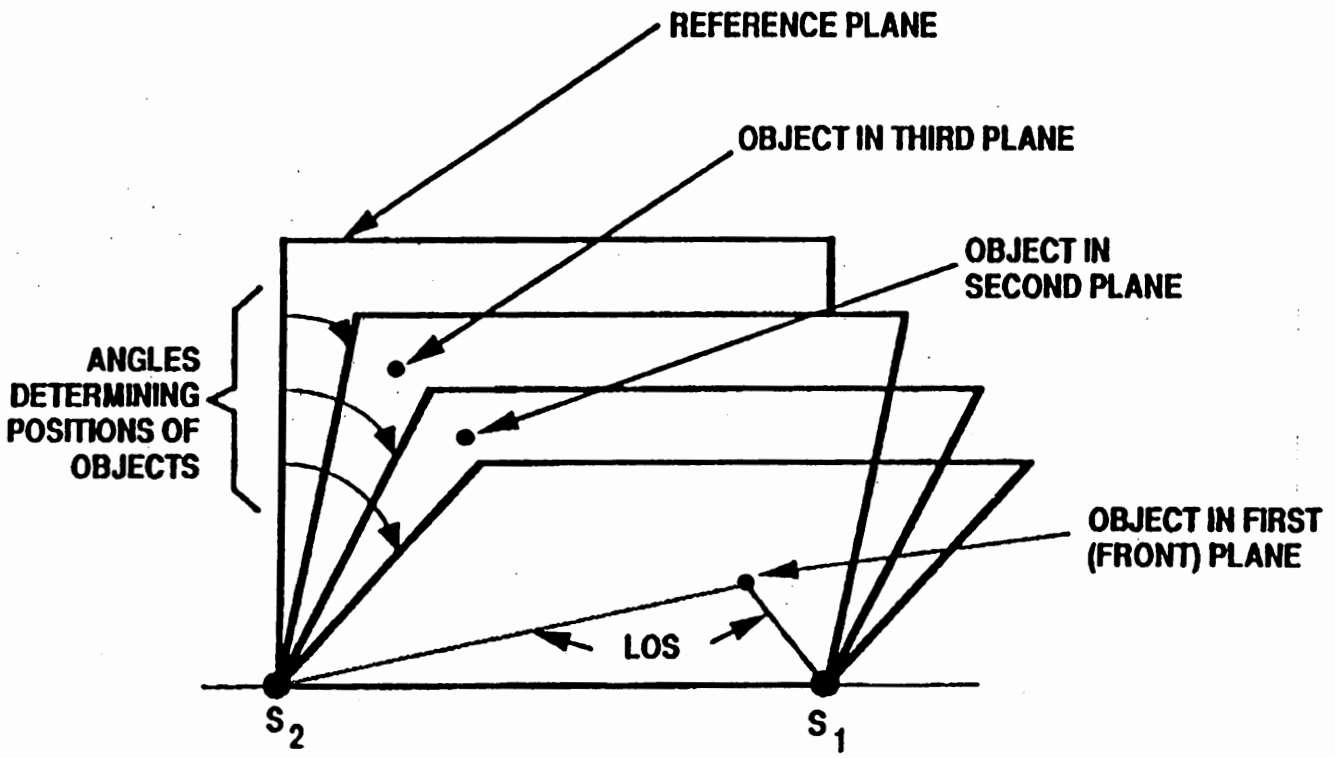


Figure 1. GEOMETRY OF TARGETS AND SENSORS

The MITRE MULTI-SENSOR, MULTIPLE TARGET TRACKER

POINT OF CONTACT: R. Varad (617) 271-4555 J. T. McKernan (617) 271-4546

The MITRE Corporation
Burlington Road
Bedford, MA 01730

ABSTRACT/SUMMARY

MITRE's multiple-sensor algorithm performs sensor-to-sensor and scan-to-scan correlation to initialize and maintain tracks of ballistic missiles in boost phase. Three sensors are used in the current software implementation of the algorithm.

As seen in Figure 1, the two main sections of the algorithm are the multiple-sensor fusion and the scan-to-scan correlation and tracking. The first section fuses data from the stereo association of sensor pairs AB and AC. Range is determined from the common sensor, A, once target lists from each pair are formed and then matched based on (1) hinge angles (the angle between each sensor-target-sensor plane and a previously defined reference plane), (2) in-plane angles (angles at each sensor in the sensor-target-sensor planes), and (3) estimates of the baseline ranges determined from each pair. Figure 2 and Figure 3 show the geometry of the sensor-target-sensor layout.

The second section, correlation and tracking, performs mainly track initialization and track maintenance. The key element to the tracking scheme is calculating rate of change of hinge angles, in-plane angles, and ranges. When these quantities are available for tracks, the rates of change can be propagated ahead to make predictions to be associated with the next set of sensor reports. The algorithm assigns a status to each track based on age and number of associations (see Section 5), and this status is the basis for upkeep/demotion and confirmation/deletion of tracks. Tracks are stored as state vectors.

Since the algorithm's core processing consists of sorting, rather than mathematical optimization, processing requirements are relatively low, on the order of $n \log(n)$, for n targets.

1. CONTEXT

The processing chain in MITRE's multiple-sensor algorithm most closely fits the general description of the "Static then Dynamic" chain as defined by the SDIO Panel on Critical Issues in Tracking. The two main functions of the algorithm are sensor-to-sensor and scan-to-scan correlation. The sensor-to-sensor portion includes the functions of two-sensor stereo association, unequal list matching, and three sensor fusion — fusion of the two sets of stereo associated data. The scan-to-scan portion's functions include track initiation and track maintenance.

The algorithm has been implemented to operate on the SDI boost phase, but the concepts are directly applicable to midcourse tracking with some modifications. The inputs are non-clustered observations from three scanning sensors. The outputs are 3-

IT0089

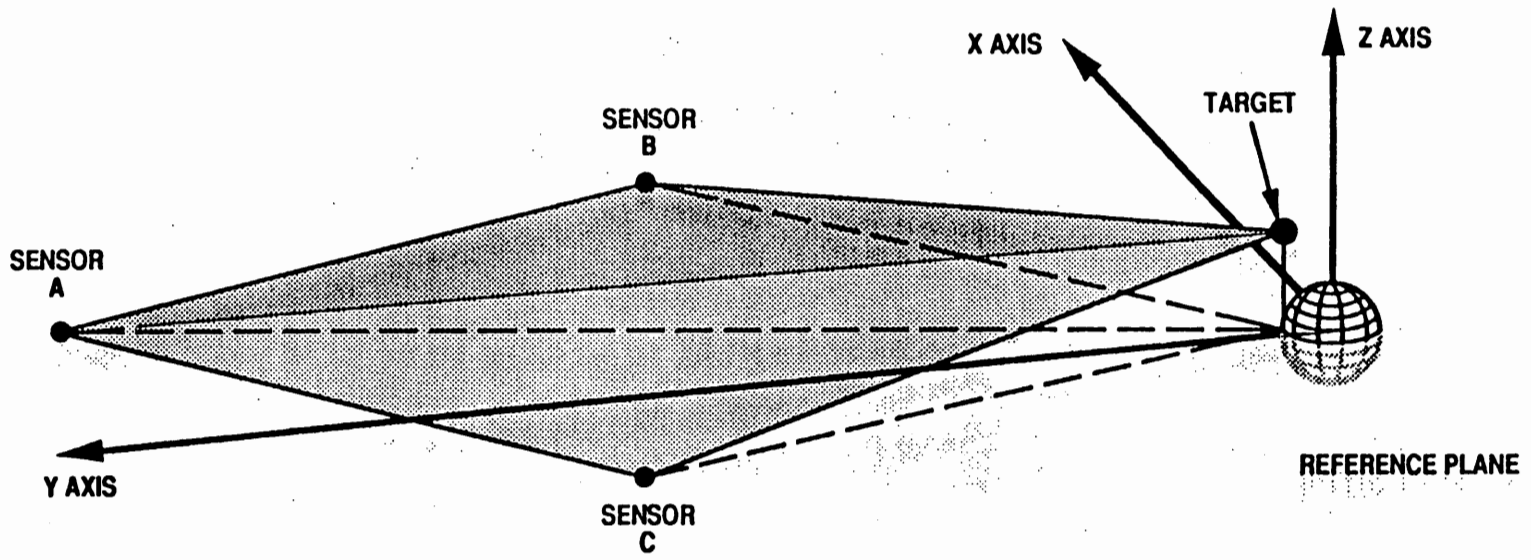


Figure 2. SENSOR TARGET GEOMETRY (BOOST PHASE TRACKING)

dimensional, unsmoothed tracks (state vectors). Provision is made for pre-clustering of targets, if feasible.

2. FEATURES

The multiple-sensor algorithm makes use of a coordinate system that exploits the sensor-target-sensor geometry. Figures 2 and 3 show the three components of the coordinate system: the hinge angle, the in-plane angle, and the baseline range (as measured from a sensor to a target). The algorithm associates using two lists of targets sorted by the hinge and in-plane angles. The association technique, however, does not employ *a priori* information, matrix optimization algorithms, or Kalman filters.

3. SENSOR ARCHITECTURE & THREAT SCENARIO

The software implementation of the multiple-sensor algorithm has been tested using several threats, including a subset of the SDIO's classified TSCB-1 threat, and notional unclassified threats consisting of 105 and 130 ballistic missiles in both spike and ripple launches. (The 105-missile threat is launched from six complexes, while the 130-missile threat is launched from two launch sites.) The sensor model does not treat clutter, sun, stars, or clouds.
The sensor models we use are the notional Boost Phase Tracking System (BSTS) sensors in a geosynchronous orbit. The three sensors are assumed to be centered over the Soviet Union, and separated by a variable amount (sensor separation between 40 and 70 degrees). Tests have been run using various representative estimates of sensor resolution and jitter.

The unclassified ballistic missile threat follows a notional powered flight model from The MITRE Corporation's simulation, while the classified threat follows the TSCB-1 powered flight models.

4. SENSOR MODEL/PROCESSING

The modeling of the sensors is <u>not</u> a function of this particular algorithm.

The program can use either scanning or staring sensors with any user-specified frame time. Data are made available throughout powered flight. Sensors report azimuth, elevation, and time of observation.

5. TRACK INITIATION

Scan-to-scan association consists of two processes, namely, track initialization and track maintenance. A track is initialized for a given target when correlation can be obtained in hinge angle, in-plane angle and range for data from two consecutive scans for the chosen sensor. Gates are established commensurate with sensor resolution and noise characteristics. The correlation is performed independently and sequentially in the three coordinates. This method of initialization is suitable for targets with relatively small velocities. For targets with relatively high velocities, hinge rates obtained from non-simultaneous observations in the sensor-to-sensor correlation process are used to tentatively initialize tracks.

8. OUTPUT TO BM/C3 AND USERS

Results are in the form of output files with tracking statistics, from which track purity, range accuracy compared with model truth, and other quantities can be derived. (Track purity is defined as: for a given set of updates to a track, in this case six scan updates, if five of those updates, in any combination, have identical model truth IDs, then the last scan update to the track is pure.)

9. COMPUTATIONAL REQUIREMENTS

The algorithm is implemented in Pascal on a VAX/VMS system. The memory requirement for 100 targets is about 300 Kbytes, and should scale linearly for larger threats. Throughput resulting from analysis of runs with 100 ballistic missiles is less than one second on the VAX 8700, and though not explicitly tested, the processing should be on the order of $n \log(n)$.

10. CURRENT STATUS

The algorithm is fully designed, developed, and implemented. It has been debugged and tested with notional unclassified threats using as many as 130 ballistic missiles launched from two complexes, and a subset of the SDIO's classified TSCB-1 threat. No performance or hardware optimization has been attempted. The concept used in this boost-phase model is applicable to midcourse as well; no midcourse model has been implemented, however.

11. PERFORMANCE MEASURES AND RESULTS

We characterized the efficiency and accuracy of this algorithm mainly by using track purity tests and by testing measured results against model truth for the calculated ranges, positions, and IDs. Tracking results are highly dependent on the threat characteristics. In general the ripple-launched ballistic missiles were tracked very well; the spike launch performance was poorer. For the case of the 105 boosters spike-launched from six complexes, the track purity (5 out of 6), ranges from 70 to 90 percent throughout.

12. REPORTS

Rajan Varad, The MITRE Corporation M87-73, Scalar Correlation Algorithm : Multi-Target, Multiple Sensor Data Fusion, December 1987.

Joseph T. McKernan, The MITRE Corporation Memorandum D44-M-337, "The Implementation of the Three Sensor Data Fusion Tracking Program in the Scenario Modeling System (SMS)", December 1988.

Project: BMEWS Phased Array Radar Upgrade

Organization: Raytheon Company

Sponsor: U. S. Air Force, Space Command

Contact: Fred Daum (~~617~~ SOS-440-8734)

Status: Operation since June 1987

Reference: B-5 Spec. for CPCI-2 (Software Requirements)

Simulations: Real-time detailed simulation of threat, radar and C^3 available in JOVIAL on CDC 170-865 machine.

Purpose: Process 3-D radar inputs to estimate position and velocity vectors and support discrimination

Data Processing Architecture: General purpose digital computer (CDC 170-865).

Threat: Dense multiple target environment; objects can be very closely spaced; measurements are occassionally unresolved in range; threat density is approximately the same as the standard CSO SDIO threat for LWIR; ballistic or boost phase.

Sensor: Phased array radar which measures range, elevation, azimuth and target amplitude. Data rate varies from 1 pps to 0.25 pps per cluster of targets with single pulses or pulse-pairs. Signal-to-noise ratio is adaptive, but nominally 15 dB. Resolution is nominally 15 m.

A Priori Information: None needed and none used.

Implicit Assumptions: None

Characteristics of Algorithms:

- 6-state Kalman filter
- Multiple hypothesis track initiation
- Pulse-pair track initiation and tracking
- Nearest neighbor chi-square test for return-to-track correlation
- Up-down chirp pulse-pairs are specifically designed to make track initiation and tracking easy
- Recognizes unresolved returns using quadrature monopulse test and prediction of target ranges

Project: Ground Based Radar (GBR)

Organization: Raytheon Company

Sponsor: U. S. Army Strategic Defense Command (USASDC)

Contact: Fred Daum (~~617~~ 508-440-8734)

Status: Under development and testing in simulation

Reference: Software Requirements for Tactical Application Software for GBR

Simulations: various pieces but no overall simulation currently available

Purpose: Process 3-D radar inputs to estimate position, velocity and higher order rotational dynamics for discrimination

Data Processing Architecture: General purpose digital computer. Immediate target machine is von Neumann type with vector pipeline, but can easily be adapted to parallel processing of various types.

Threat: Dense multiple target environment; objects can be very closely spaced; dense chaff; all ballistic trajectories.

Sensor: Phased array radar which measures range, elevation, azimuth, target amplitude and phase. Data rate is adaptive to local environment and immediate requirements. Signal-to-noise ratio is adaptive as well. Resolution is adaptive to local environment.

A Priori Information: Can use handover data (position and velocity with covariance matrix) or can search autonomously.

Implicit Assumptions: None

Characteristics of Algorithm:

- Three Kalman filters (6-state, 7-state, and 16-state)
- Multiple hypothesis track initiation
- Pulse-pair track initiation and tracking
- Nearest neighbor chi-square test for return-to-track correlation
- Weighted average of monopulse angular measurements over multiple returns from a given object
- Pulse-pairs are specifically designed to make track initiation and return-to-track correlation easy

SUMMARY DATA ON VELOCITY FILTER ALGORITHM FOR SDI DETECTION AND TRACKING WITH PASSIVE ELECTRO-OPTICAL SENSORS

Organization

Space Computer Corporation
2800 Olympic Boulevard, Suite 104
Santa Monica, California 90404

Projects and Sponsors

1. Advanced Processing for Infrared Sensors (Army Strategic Defense Command)
2. Integrated Track Acquisition and Discrimination Concepts (MIT Lincoln Lab)
3. Object Acquisition, Tracking and Discrimination via Bulk Processing (SDIO/ONR)
4. Velocity Filter Approach to Boost/Midcourse Tracking (RADC)

Contact

Dr. William J. Jacobi or Dr. William B. Kendall
(213)829-7733

Status of Algorithms

Under development and evaluation by Space Computer Corporation. The original concept is due to Prof. Irving S. Reed at the University of Southern California [1].

Additional Information about Algorithm

1. Purpose of Algorithm. The velocity filter algorithm performs a combination of signal-to-noise ratio enhancement and scan-to-scan correlation functions utilizing a "track-before-detect" approach. It utilizes an input sequence of image frames (or equivalent data list) which have been processed for detector gain/offset correction, TDI, radiation hit suppression (gamma circumvention), etc. The filter output is an image (or equivalent data list) of objects having velocities within the passband of the filter. A bank of filters tuned to different vector velocities provides correlated object positions and velocities for track initiation. The vector velocities (and accelerations) to which the filters are tuned are derived from cross-correlation of successive input image frames.

2. Data Processing Architecture Assumed. The velocity filter algorithm is most efficiently implemented by a special-purpose parallel processor attached to a general-purpose host computer. Studies of alternative processor architectures are being carried out by Space Computer Corporation.

TRACKING ALGORITHM SUMMARY

Submitter's Name: Kenneth Kessler
Submitter's Company: Systems Control Technology
Submitter's Phone: 415 494 2233
Address: 2300 Geng Road, Palo Alto, California 94303
Author: See Above

Type of Algorithm: Streak Detection (Track before Detect)

Title of Algorithm: Dynamic Programming Algorithm DPA)

Sponsor: DARPA/NOSC
Developer: Yair Baarniv

ABSTRACT/SUMMARY

Dynamic Programming can provide an alternative approach to the detection and tracking of dim targets using returns from a wide variety of sensors. The conventional (classical) approach to this problem is to obtain a detection of the target return from a single "frame", then attempt to associate this single return with a "track file" previously established on the object.

Alternatively, the problem of detecting and tracking dim targets in IR mosaic-sensor imagery (or other sensor data) can be thought of as one of detecting and locating whole target trajectories inside the sensor's field of view over some interval of time, selected by the user. A batch processing algorithm is required in this case. Such an algorithm has the potential of obtaining superior performance over that of the conventional approach, because its performance (measured by its probabilities of detection and false alarm) would be based on the integrated energy of a target during its entire stay inside the sensor's field of view rather than on the short single-frame time.

Conceptually, one candidate batch processing algorithm might use exhaustive search techniques. The multi-frame data is recognized as being three-dimensional (3-D), where the third dimension is associated with the frame number, or time of the frame. For the idealized white Gaussian uncorrelated noise case, the optimal detection consists of passing the "batched" data through a bank of matched-filters (MF), where each filter represents a single possible 3-D trajectory.

The Dynamic Programming Algorithm (DPA) is a practical and feasible technique to replace the above exhaustive search technique over all possible trajectories and still get similar performance. The main idea is to define any possible trajectory as a string of straight-line short segments. Each straight-line segment is defined over a small number of frames which constitute a DP "stage". Exhaustive search can now be performed over one stage at a time, and the principles of dynamic programming used to "piece together" the results of one stage with that of the next. The number of the possible short segments, which is user defined, is relatively small so that this single-stage filtering is feasible.

performance is obtained when the target trajectories are separated by at least 1/2 to 1/4 of a pixel and not parallel (in angle) over approximately 6 to 8 frames of data.

No specific clutter statistics are assumed, however the matched filters are designed for white Gaussian noise (other designs can be used). If the standard design is used, the frame data should be differenced, so that the background approaches this statistical assumption.

The target model kinematics is assumed to be nearly straight, constant speed model. User inputs do allow, however, for the target to deviate from this assumption. Some curvature and acceleration is allowed by input specific parameters of the algorithm. Processing time can be minimized as the target characteristics approach those used in the kinematic model, by suitably modifying the input parameters.

4. SENSOR MODEL/PROCESSING

The algorithm was originally designed for a staring mosaic sensor, however in principle, the development can be extended to other mosaics such as step starers and scanners as well as radar. The frame time of the sensor is variable, but some preprocessing is assumed in order to obtain "apparent" target motion of approximately one pixel length per scan. The algorithm processes a single waveband of data per batch; additional wavebands can be processed independently, then merged to obtain a single track, if desired.

Error parameters or characteristics are not explicitly modeled within the tracking algorithm. That is measurement noise, biases etc. are not modeled as in a conventional Kalman filter, which explicitly or implicitly models all process and measurement errors. The Dynamic Programming Algorithm implicitly assumes the following: that each pixel return is associated with the center of each pixel, that the measurement noise is Gaussian and uncorrelated in space and time (or nearly so), and that the assumed target motion is "nearly" straight and of constant speed. The performance of the algorithm is optimal with these assumptions but degrades as these assumptions become less valid.

The precision of the input sensor data has been tested down to 4 or 5 bits of precision with very small loss of performance. The interface between the sensor and the tracker is required to be multiple frames of data, stored for subsequent processing. The numbers of frames used for processing are user defined and each frame contains a single energy return per pixel. Performance is improved if first or second frames differencing is employed, in an attempt to decorrelate the background clutter and noise.

The performance of the algorithm is largely dependent upon the signal strength of the target, background clutter and noise levels, numbers of frames processed, etc. The algorithm has been used to process HiCamp data at NOSC, and has demonstrated the ability to detect and track targets having a signal strength of 2 or so dB, with probabilities of detection of 0.9 or more, and false alarms of 10E-5 to -6.

The algorithm has been tested against both simulated and actual sensor data (HiCamp I and II). The algorithm has not been married to a specific hardware architecture, rather, it has been used to process post-mission data where the specific sensor data has gone through a "pre-processing" stage, typically single frames differencing.

11. PERFORMANCE MEASURES & RESULTS

According to staff at NOSC, the DPA has been tested to detect and track dim targets having a SNR of approximately 1-3 in magnitude. Test cases have also demonstrated simultaneous track of 72 very dim, crossing targets, all within the field of view (simulated data) and actual data from the HiCamp I and II sensors. This latter set is with a variety of geometries and SNR's and targets.

There have been theoretical performance bound computed during the past several years and contained in several reports and papers. The algorithm has not been sufficiently tested to determine under what conditions the algorithm performs poorly

12. REPORT

The most recent project report is:

Dynamic Programming Algorithm (DPA) Analysis
Systems Control Technology, Inc.
Final Report under contract TLJ-5734-1866-87 for
Titan Systems, Inc.
Authors: Kenneth Kessler, Michael Sutphen, Thomas Holmes, Barbara McQuiston
November 1988

TITAN
SYSTEMS, INC.

3 April 1989

Dr. Gabriel Frenkel, IDA
IDA Tracking Panel
Washington, DC

Subject: Panel Survey

Dr. Frenkel:

This letter is in response to the IDA Tracking Panel Survey. Before getting into the actual survey items, I want to provide you with some background on the Knowledge-Based Sensor Fusion (KBSF) program. It is my desire to establish the proper context within which the survey was completed. I have tried to fit a description of our system into your format, which is really designed for algorithm descriptions in a strict sense, so please bear these thoughts in mind as you review the survey.

As I indicated to you earlier, TITAN Systems, Huntsville is not in the process of designing tracking algorithms perse. The KBSF program was initiated by the SDI to determine whether knowledge / rule-based techniques could be applied to the strategic sensor fusion problem arena. Specifically, the intent is to demonstrate a rule-based approach to multisensor discrimination. As you know, it is not entirely possible to decouple tracking issues from discrimination issues, hence the KBSF program's interest is in hosting and evaluating "competing" algorithms (primarily those designed for platform-to-platform association and track maintenance). While some of these algorithms may have been designed for essentially the same track-related purpose, each may contribute differently to the discrimination process. Our goal is to build a machine that automatically selects the "best" algorithm for fusing multiple / single platform, multiple sensor track data, given the peculiarities of the scenario at hand.

Of particular interest to our program are the methods by which uncertainties in the tracking functions (and hence uncertainties in the radiometric / RF quantities important to discrimination) are characterized, propagated and dealt with at the Battle / Sensor Management level. We are experimenting with fuzzy techniques for determining "degrees of membership" in the set of valid tracks that is maintained by the system, and for uncertainty management in general.

2705 Artie Street, Suite 25 Huntsville, AL 35805

5. Track Initiation

Currently, a fuzzy variation on the Sequential Probability Ratio Test is employed to determine the degree of membership of each sensor report in the system-maintained set of "valid tracks". This degree of membership effectively weights each report. The weight is propagated through the system to the fusion and then to the discrimination process, where it impacts lethality decisions. The same procedure is used for track maintenance, in conjunction with an impact point prediction routine for both single and multiple platform association. RSS scoring is used. All target and sensor states are maintained in Earth-Centered Inertial coordinates.

7. Track File Maintenance

See attachment 1. for a description of the Fused Data Frame which is utilized by the KBSF machine.

8. Output to BM / C^3 and Users

See attachment 2. for an example display screen which shows Platform-to-Platform association results and discrimination results.

9. Computational Requirements

The current system is implemented in a multi-SUN workstation architecture. All KBSF components are written in C. However, algorithms written in other languages supported by SUN Unix may be called by the rule basses (the trajectory model used for impact point prediction is an example written in FORTRAN). This feature provides a powerful testbed capability.

10. Current Status

A multiple computer system has been demonstrated and effort is underway to enhance baseline algorithmic, rule - base and uncertainty management capabilities. Simulated sensor data has been used to exercise the system to date. Real sensor data is desired for system testing.

```
struct frm {
        int             message;
        char            sa_time_rec[7];
        char            cor_time_rec[7];
        char            cor_time_ret[7];
        char            fus_time_rec[7];
        char            fus_time_ret[7];
        char            dis_time_rec[7];
        char            dis_time_ret[7];
        char            sa_time_ret[7];
        struct zulu_time launch_time;
        int             items_recvd;
        int             items_procd;
        int             cluster_num;
        int             object_num;
        int             psn;
        int             obj_cat;
        int             d_target_type;
        int             obj_conf;
        int             context;
        float           track_quality[2];
        int             ppsn;
        float           weight;
        struct zulu_time obs_time;
        double          time;
        double          x_eci, y_eci, z_eci;
        double          x_dot, y_dot, z_dot;
        struct zulu_time imt;
        double          s_imt;
        double          imp_x, imp_y, imp_z;
        double          taralt, tarlat, tarlon;
        double          range;
        int             rep_sensor;
        int             sid;
        double          senalt, senlat, senlon;
        double          elevation, azimuth, aspect_angle;
        double          range_rate;
        double          beta;
        float           cor_meas_error[16];
        int             ode_sensor;
        int             bode_sensor;
        float           irrad[4];
        double          radint[4];
        float           temp, dif_temp, roc_temp, avg_temp, avg_dif_temp;
        float           ea,  roc_ea, avg_ea;
        float           gb_ea, dif_gb_ea, roc_gb_ea;
        float           gb_temp, dif_gb_temp, roc_gb_temp;
        float           bode_a[4], bode_b[4], bode_c[4];
        float           sigma, omega, roe, amplitude[4];
        float           signal_noise, vis_mag;
        float           lrcs, rcs, roc_rcs;
        float           spin_rate;

};


struct zulu_time current_time;
struct zulu_time launch_time;
```

TRACKING ALGORITHM SUMMARY

NAME: J. T. Lawson **DATE:** 14 February 1989

COMPANY: TRW System Development Division, Huntsville Operations

PHONE: (205) 830-3213

ADDRESS: 213 Wynn Drive, Huntsville AL 35805 M/S HSV8-1414

ALGORITHM TITLE: ADOP Scan-to-Scan and Track Algorithm Set

SPONSOR: U. S. Army Strategic Defense Command (USASDC)

DEVELOPER: TRW Huntsville Operations

message is sent. Unassociated observations (those not assigned to a track) are then submitted to Track Association, which attempts to form strings of observations to generate new tracks over a period of several scan cycles. Successful strings of associations are reported to the Track process via New Track messages. Only unique assignments are reported.

The Track process is responsible for initializing track filters, updating track filters and making predictions for the next sighting of the tracked object. Two types of track filters are supported: angle-only and six-state. The angle-only filter is a five or six point cubic fit least squares filter in azimuth and elevation. The six-state filter is a Kalman filter in Earth Centered Boresight Inertial (ECBI) coordinates. The ECBI system axes are parallel to the fixed boresight coordinate system of the sensor and are a fixed transformation from Earth Centered Inertial (ECI) coordinates.

1. CONTEXT

The ADOP algorithm set addressed the problem of single sensor onboard acquisition and track for a ballistic probe (e.g. GSTS) during the midcourse engagement phase. The sensor is assumed to hold a fixed orientation along an inertial boresight for the duration of a mission and executes a racetrack scan pattern with a slight overlap zone. Hence, each sensor "frame" consists of a top scan and a bottom scan within the sensor frame time. Detections from sensor signal processing in three LWIR bands are processed by ADOP Measurement Processing algorithms (not discussed here) to perform color correlation, velocity estimation, irradiance calibration, bulk filtering of stars and scan smoothing, among other functions. The input to the Scan-to-Scan process from Measurement Processing consists of a set of object observations from the latest scan. Each observation record includes sighting time, azimuth, elevation, estimated azimuth and elevation rates, standard deviations for the angles and rates, and irradiances measured in the three color bands. Each record is marked as an isolated single or a CSO. For CSOs, azimuth and elevation extent data are also included. Outputs from the Track process to a ground-based battle manager consist of threat reports that include object six-state and covariance, time of applicability, and estimated lethality. A threat report is provided every two to three frames for each object determined to be lethal by discrimination.

2. NOTABLE FEATURES

The ADOP algorithm set was the first, to our knowledge, to incorporate a pattern matching approach to the midcourse track association problem. Moreover, both pattern matching algorithms and classical gating algorithms were used in a complementary fashion

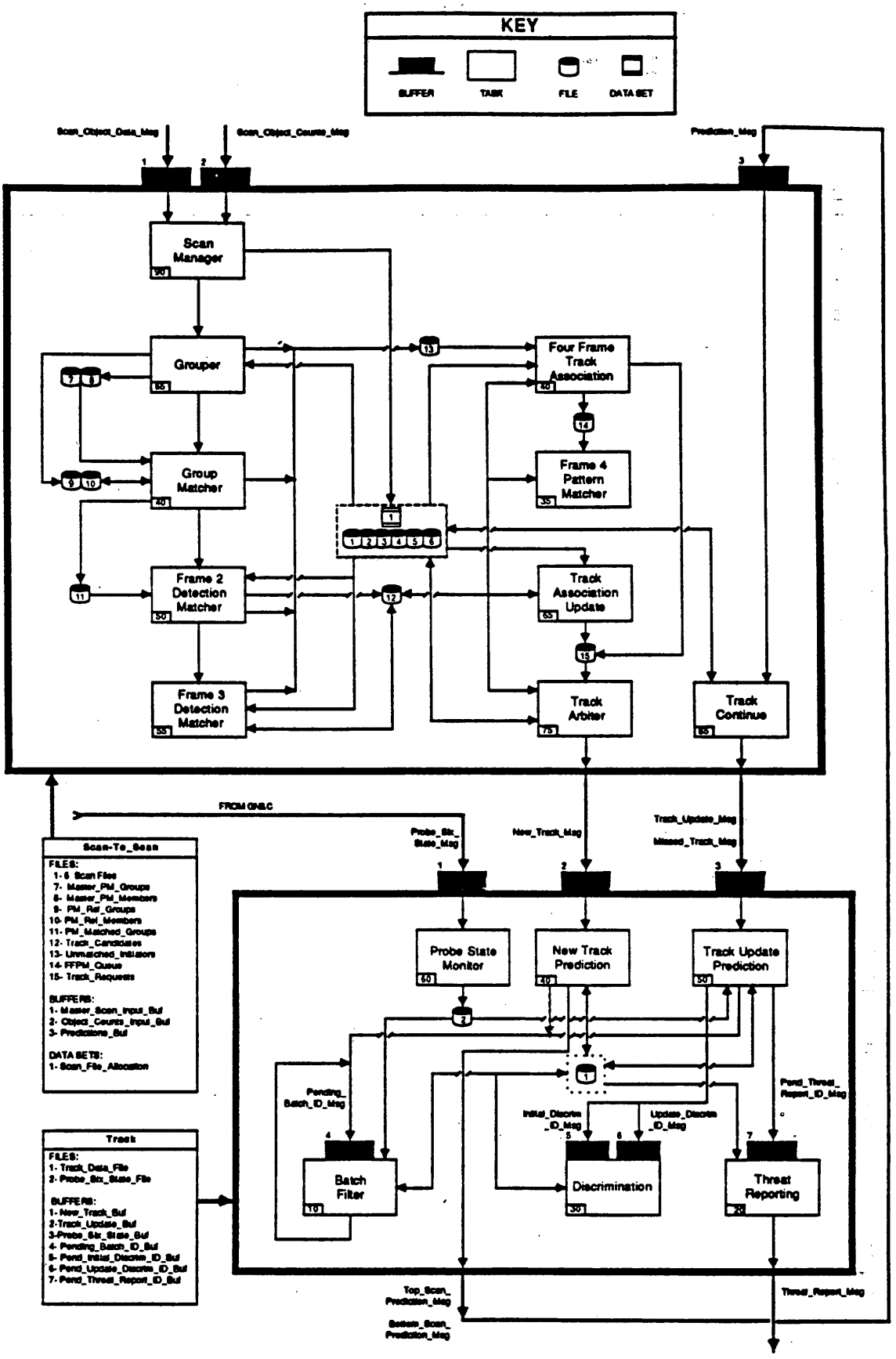


Figure 2: Scan-to-Scan and Track Tasks and Flows

of group members are written as a list to the PM_Ref_Members file. Degenerate groups of one member (isolated singles) are discarded and their scan file record numbers are written to the Unmatched Initiators file. (Alternatively they can be retained for group matching and continue through pattern matching as an option.) The Grouper also operates on the second scan (Frame 2) of the scan-to-scan cycle when it arrives. For this scan the outputs are written to the Master_PM_Groups and Master_PM_Members files. The Group Matcher task is then activated.

Group Matcher: This task attempts to match Frame 1 groups to Frame 2 groups. The primary criteria are that the Frame 2 centroid lies in the predicted centroid gate and the extents of the groups are reasonably close. The comparative numbers of members in both groups could also be used. Where a unique match is found, information for both groups from the ...Groups and ...Members files is written to the PM_Matched_Groups file for use in detection matching. Where there are either no matches or more than one match to a Frame 1 group, the scan file record numbers of the members of the Frame 1 group are written to the Unmatched_Initiators file. When all Frame 1 / Frame 2 groups have been processed, the Frame 2 Detection Matcher is enabled.

Frame 2 Detection Matcher: This task operates in Frame 2 after the Group Matcher, hence its name. Its purpose is to match individual observations within the Frame 2 groups with those in the matched Frame 1 groups. An empirical multipass algorithm is currently used to do the matching. This algorithm seeks to establish an initial match between one Frame 1 group member and one Frame 2 group member, and use that match to narrow acceptance gates for matching the remaining members. Other techniques may give better performance with less effort. Exactly which detection matching technique is best under what conditions is still an open research issue. The ratio of successful matches to group size is used as a criteria to accept or reject the match data. Engineering evaluations during the ADOP project showed that when the ratio slipped below 0.34, the matches were usually incorrect. When the matches are rejected, the scan record numbers for all members of the Frame 1 group are written to the Unmatched Initiators file. When the matches are accepted for the group, the individual matches are written to the Track Candidates file for use by the Frame 3 Detection Matcher.

Frame 3 Detection Matcher: This task operates during Frame 3 and attempts to match Frame 3 observations to the two-point candidate tracks established by the Frame 2 Detection Matcher. Candidate tracks are processed by Frame 2 group. The group centroid and velocity are calculated and a Frame 3 extent gate is established. All Frame 3 observations within this gate are processed in a manner similar to that used by the Frame 2 Detection Matcher. For candidate tracks which pass the match criteria, a predicted position

Track Arbiter: This task is the last stop in the track association chain. Its job is to ensure that only one track claims each of the constituent observations in the chain of observations that result in a new track. The arbitration scheme is simple, but effective -- first come, first served. The Track Arbiter operates first when the scan observations from Frame 5 are available, again immediately after the Frame 6 observations are available, and once again when all proposed tracks from the scan-to-scan cycle have been extracted and the system is ready to begin a new cycle. The Track Arbiter takes each instance in sequence from the Track Requests file, and propagates the candidate track to the most recent frame. If any ambiguities arise (more than one observation per acceptance gate) the nearest to the predicted sighting is chosen. At this point the Track Arbiter checks all observations from the previous scan files that belong to the candidate track being processed. If any of the observations have been previously been taken the candidate is discarded. If none of the observations have been taken, the Track Arbiter removes (or marks as taken) the observations from the previous scan files and sends a New Track message to the appropriate Track process. This message contains all needed data from the string of sightings that is needed to initiate angle-only track.

At the end of the scan-to-scan cycle (six frames) the Scan Manager task decides which frame will be Frame 1 of a new cycle. If pattern matching is to be employed, the new cycle must begin with the first frame following the cycle just completed. However, if use of the gating algorithm alone is acceptable, the Scan Manager can designate either Frame 3 or Frame 4 of the old cycle as Frame 1 of the new cycle.

6. TRACK MAINTENANCE

Most of the "track maintenance" functions are performed in the Track process shown in Figure 2. The exception is the Track Continue task in the Scan-to-Scan process.

6.1 TRACK MAINTENANCE -- DATA ASSOCIATION

The Track Continue task in the Scan-to-Scan process associates scan observations with predicted object positions forwarded by the Track process(es). A simple gating technique was used, with gates computed by the Track process. If no observations lay in the gate, a Missed Track message is returned to the Track process. When one or more observations lay in the gate a Track Update message is returned to track with the angular measurement data for the chosen observation and the track id with which it has been associated. When more than one observation is in the gate, the one closest to the predicted position is chosen. All predictions are processed sequentially and independently. Optimization across multiple

7. TRACK FILE MAINTENANCE

The data maintained in the track file is dependent on the track mode. During angle-only tracking an observation history of up to fifteen sightings is maintained. For each sighting, the time, azimuth, elevation, az-sigma, el-sigma, class (single, CSO), and irradiances are maintained. In the precision track mode, the time, six-state, covariance, lethality, next predicted sighting, last measured sighting, and a radiant intensity history are maintained. In both modes additional flags and counters are used to indicate track status. All track records are considered independent and there was no attempt to organize them by group, cluster, etc. Several improvements could be made to support more sophisticated track continuation techniques, but none have been implemented.

8. OUTPUT TO BM/C^3 AND USERS

Outputs from the Track process to a ground-based battle manager consist of threat reports that include object six-state and covariance, time of applicability, and estimated lethality. A threat report is provided every two to three frames (i.e every twenty to thirty seconds) for each object determined to be lethal by discrimination. Each threat report is formed by extracting information from the object's track file when the object id is passed to the Threat Reporting task by the Track Update task.

9. COMPUTATIONAL REQUIREMENTS

The ADOP software was implemented in Pascal to run on a distributed network of Honeywell ADOP processors. Each node consisted of three 1-MIP MIL-STD-1750A CPUs sharing up to one megabyte of memory. A development test version, also in Pascal executes on DEC VAX hardware. The design goal was to support 300 to 400 objects per MIP with one of the ADOP nodes whether it be Scan-to-Scan or Track. A 1000 object test case was executed in real time on a three-node ADOP network, but detailed internal data were not logged as it was primarily a hardware test. A 65 object test case has executed successfully in real time on a single VAX 11/780.

10. CURRENT STATUS

Versions of the ADOP software are available at the USASDC Advanced Research Center (ARC) for both the Honeywell and DEC VAX configurations. No significant improvements to the implementation have been made since 1985. A number of performance and capability improvements were identified under the APIRS program, but could not be implemented due to funding constraints. Some of these upgrades may be made under the GSTS program if an option is elected to convert the software to Ada for use

APPENDIX C

RESPONDENTS AND CONTRIBUTORS

LIST OF SURVEY CONTRIBUTORS AND RESPONDENTS

Frank Albini, Institute for Defense Analyses
Bohdan Balko, Institute for Defense Analyses
Lawrence Beyl, MindGate Technologies, Inc.
Thomas Blackburn, McDonnell Douglas
Donald Brand, Geodynamics Corporation
Timothy Brockwell, Titan Systems, Inc.
Charles Capps, U.S. Army Strategic Defense Command
David A. Castanon, Alphatech
Chee-Yee Chong, Advanced Decision Systems
Fred Daum, Raytheon Corporation
Oliver E. Drummond, Hughes Aircraft
Keh-Ping Dunn, MIT Lincoln Laboratory
Richard Edsinger, Boeing Aerospace Corporation
Larry Filippelli, Ball Systems Engineering Division
Jack Fleming, McDonnel Douglas
Gabriel Frenkel, Institute for Defense Analyses
Barry Fridling, Institute for Defense Analyses
Andrew Goldfinger, Johns Hopkins University Applied Physics Laboratory
Edward Goodchild, Advanced System Architectures Ltd., U.K.
Thomas Gottschalk, Caltech Jet Propulsion Laboratory
Christopher Hawkins, ESL
Daniel Holtzman, Vanguard Research, Inc.
Tom Janssens, The Aerospace Corporation
Larry Johnson, Teledyne Brown Engineering
Kenneth Kessler, Systems Control Technology
Michael Kovacich, Lockheed Missiles and Space Company
Joel Krajewski, MITRE Corporation
James Latimer, MITRE Corporation
Jim Lawson, TRW Huntsville
Jack Liu, ESL, Inc.
Keith Maples, Teledyne Brown Engineering
Joseph McKernan, MITRE Corporation
Arloe Mayne, TRW
Barry Metzger, MITRE Corporation
Shozo Mori, Advanced Decision Systems
Neil Oberholtzer, G.E. Company
Howard Onishi, Hughes Aircraft
James Ortolf, Applied Research and Engineering
Kris Roberts, G.E. Aerospace
Howard Shao, Aerospace Corporation
Ivin Tarnove, TRW Huntsville
Shawn Toumodge, The Aerospace Corporation
Ming-Jer Tsai, MIT Lincoln Labs
Shirley Tucker, U.S. Army Strategic Defense Command
James R. VanZandt, MITRE Corporation
Rajan Varad, MITRE Corporation
Robert Washburn, Alphatech

APPLICATION OF BAYESIAN NETWORKS TO MIDCOURSE MULTI-TARGET TRACKING

MICHAEL KOVACICH

PRESENTED AUGUST 3, 1989 TO THE PANEL ON ADVANCED CONCEPTS.
LMSC, SUNNYVALE, CA. BOHDAN BALKO, CHAIRMAN

K9-7246/060

INTRODUCTION

- BAYESIAN NETWORKS (INFLUENCE DIAGRAMS) HAVE EVOLVED OVER THE LAST DECADE INTO A POWERFUL TOOL FOR PROBABILISTIC INFERENCE:
  - HOWARD & MATHESON (1981) (SEMINAL PAPER)
  - SCHACTER (1986) (DISCRETE INFLUENCE DIAGRAMS)
  - KENLEY (1986) (NORMAL INFLUENCE DIAGRAMS)
  - PEARL (1986) (BAYESIAN NETWORKS)
- INFLUENCE DIAGRAMS PROVIDE A FRAMEWORK TO REPRESENT AND MANIPULATE JOINT PROBABILITY DISTRIBUTIONS FOR COMPLEX NETWORKS OF RANDOM VARIABLES
- INFLUENCE DIAGRAMS CAN BE USED TO IMPLEMENT WITHIN THE SAME FRAMEWORK
  - STATE ESTIMATION (LINEAR GAUSSIAN)
  - DATA ASSOCIATION (DISCRETE)
  - TRACK PROMOTION (DISCRETE)

  } TRACKING
- LMSC HAS BUILT A LIBRARY OF INFLUENCE DIAGRAM UTILITIES TO PROTOTYPE MIDCOURSE TRACKING ALGORITHMS
  - ALGORITHM PERFORMANCE
  - THROUGHPUT/MEMORY (NONOPTIMIZED)

INTRODUCTION

This presentation discusses the application of Bayesian Networks or Influence Diagrams to the implementation of midcourse tracking algorithms. The Influence Diagram is used to represent and manipulate probabilistic information in complex networks of random variables. The generic capabilities of the Influence Diagram are used to carry out the major tracking functions, including linear gaussian state estimation, data association hypothesis scoring and track promotion scoring.

LMSC has built a library of Influence Diagram utilities to construct, scan and manipulate an Influence Diagram. These utilities are used in implementing the midcourse tracking algorithm in order to assess algorithm performance and to begin to estimate throughput and memory requirements. The throughput and memory requirements are upper bound estimates at this stage since the algorithms are executing within a generic environment which is not tailored and optimized for a specific hardware environment.

AGENDA

- INFLUENCE DIAGRAMS USED TO REPRESENT UNCERTAIN KNOWLEDGE IN COMPLEX SYSTEMS
  - GENERIC REPRESENTATION
  - APPLICATION TO MIDCOURSE TRACKING
- OPERATIONS ON INFLUENCE DIAGRAMS TO PERFORM INFERENCING
  - GENERIC OPERATIONS
  - APPLICATION TO MIDCOURSE TRACKING

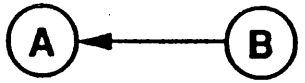
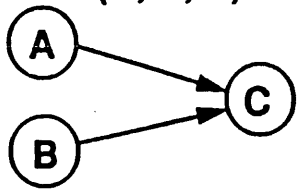
AGENDA

The agenda will cover both the generic aspects of Influence Diagrams and their application to the midcourse tracking problem. Furthermore, the agenda will be partitioned into a discussion of the capability of the Influence Diagram to represent uncertain or probabilistic information in complex systems, and the operations used in manipulating the Influence Diagram.

BAYESIAN NETWORK/INFLUENCE DIAGRAM DEFINED

- ACYCLIC DIRECTED GRAPH REPRESENTING THE JOINT PROBABILITY DISTRIBUTION FOR A SET OF RANDOM VARIABLES
  - NODES = RANDOM VARIABLE
  - ARC = PROBABILISTIC CONDITONING

EXAMPLES

P (A, B)	P (A, B)	P (A, B, C)
		
P (A) P (B/A)	P (B) P (A/B)	P (A) P (B) P (C/A,B)

- OBSERVATIONS
  - A JOINT PROBABILITY DISTRIBUTION CAN BE REPRESENTED BY MANY INFLUENCE DIAGRAMS - ONE FOR EACH DECOMPOSITION.
  - LACK OF AN ARC REPRESENTS CONDITIONAL INDEPENDENCE.

K9-7246/002

BAYESIAN NETWORK/INFLUENCE DIAGRAM DEFINED

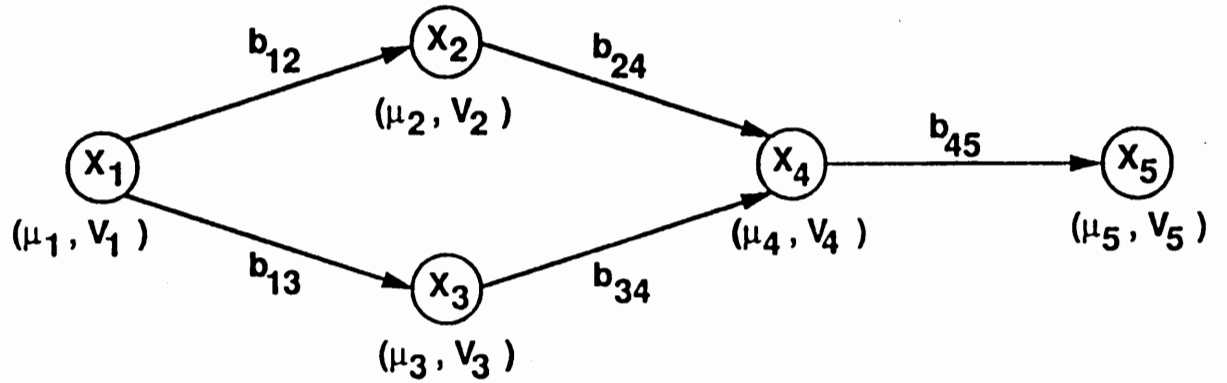
This chart defines the Influence Diagram as a device to represent a joint probability distribution of a set of random variables. Each random variable is represented as a node and conditional dependence between random variables is represented as an arc between the corresponding nodes.

A joint probability distribution can be factored in many ways. In the example, the joint distribution, P(A,B) can be written as P(A)*P(B/A) or as P(B)*P(A/B). Each factorization or decomposition, is represented by a specific Influence Diagram.

The lack of an arc between two nodes indicates that the corresponding random variables are conditionally independent of each other.

NORMAL INFLUENCE DIAGRAM

SCALAR NODE EXAMPLE



- $N = \{1, 2, \ldots, n\}$
- $X_N = (X_1, \ldots, X_n)$ X_i IS A SCALAR NORMAL RANDOM VARIABLE

$$E[X_N] = \mu_N \quad (n \times 1)$$

$$\text{Cov}[X_N] = \Sigma_{NN} \quad (n \times n)$$

- CONDITIONAL DISTRIBUTIONS

$$E[X_j \mid X_{c(j)} = x_{c(j)}] = \mu_j + \sum_{k \in C(j)} b_{kj} (x_k - \mu_k)$$

$$\text{Var}[X_j \mid X_{c(j)} = x_{c(j)}] = v_j$$

C (j) = CONDITIONAL PREDECESSORS OF NODE j

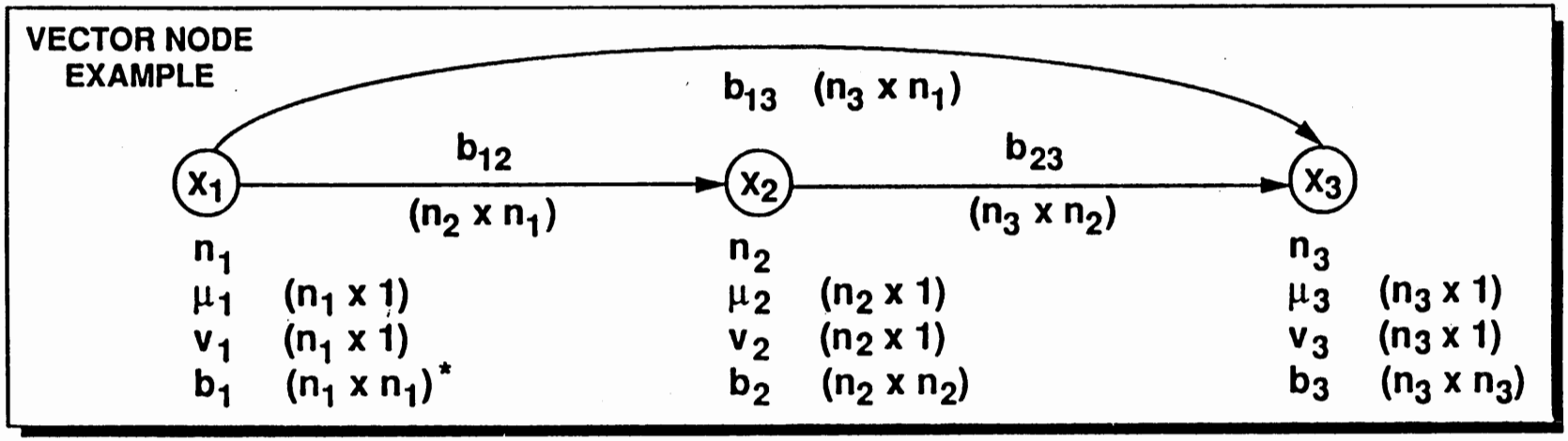
K9-7246/003

NORMAL INFLUENCE DIAGRAM

This chart presents an example of an Influence Diagram for the joint probability density for a set of 5 normal random variables.

Each random variable is a scalar random variable with an unconditional mean, mu, and conditional variance, v. The arc strengths, bij, represents the influence of the ith random variable on the jth random variable and are used in the expression for the conditional mean of jth random variable.

NORMAL INFLUENCE DIAGRAM



- $N = \{ 1, 2, ..., n \}$

- $X_N = (X_1, ... , X_n)$ X_i IS A VECTOR NORMAL RANDOM VARIABLE OF LENGTH N_i

$$E[X_N] = \mu_N \qquad (m \times 1) \qquad m = \sum_i n_i$$

$$COV[X_N] = \Sigma_{NN} \qquad (m \times m)$$

- CONDITIONAL DISTRIBUTIONS

$$E[X_j / X_{c(j)} = x_{c(j)}] = \mu_j + \sum_{K \varepsilon c(j)} b_{kj} (x_k - \mu_k)$$

*NOTE: b_i HAS $n_i \times (n_i - 1)$ 12 INDEPENDENT COMPONENTS

K9-7246/004

NORMAL INFLUENCE DIAGRAM

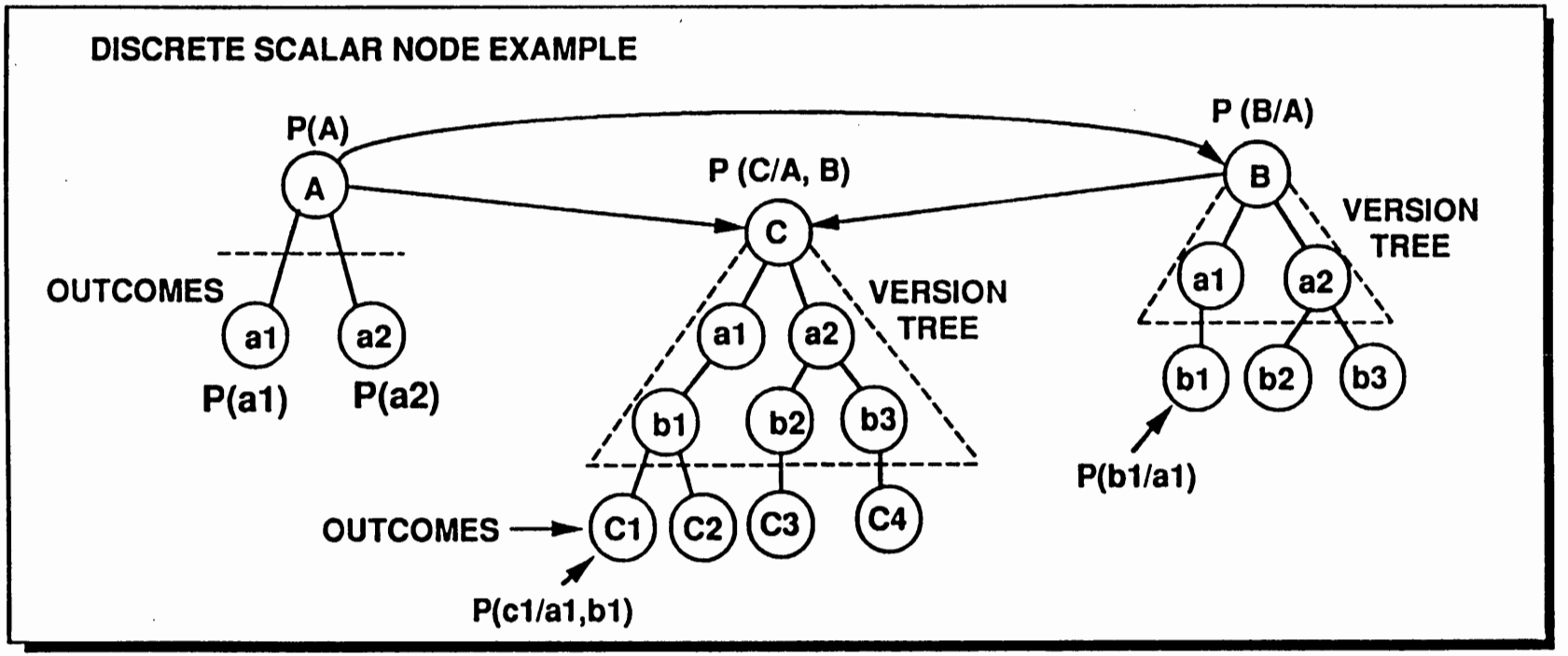
This chart presents the influence diagram for 3 normal random variables in which each normal random variable is a vector.

Each vector variable is represented by the unconditional mean vector (mui), the conditional variance vector (vi) and the internal arc strengths (bi).

The conditional dependence between the vector variables is represented by the external arc strengths (bij).

It should be noted that the internal arc strengths (bi) have ni*(ni-1)/2 components

DISCRETE INFLUENCE DIAGRAM



- JOINT DISCRETE RANDOM VARIABLE S = (A, B, C)
  A, B, C ARE DISCRETE RANDOM VARIABLES
- CONDITIONAL PROBABILITIES
  P(A) P(B/A) P(C/A, B)

K9-7246/015

DISCRETE INFLUENCE DIAGRAM

This chart shows an example of an Influence Diagram for 3 discrete random variable, (A,B,C). The joint probability density, P(A,B,C), is factored into P(A)*P(B/A)*P(C/A,B) which is represented in the diagram.

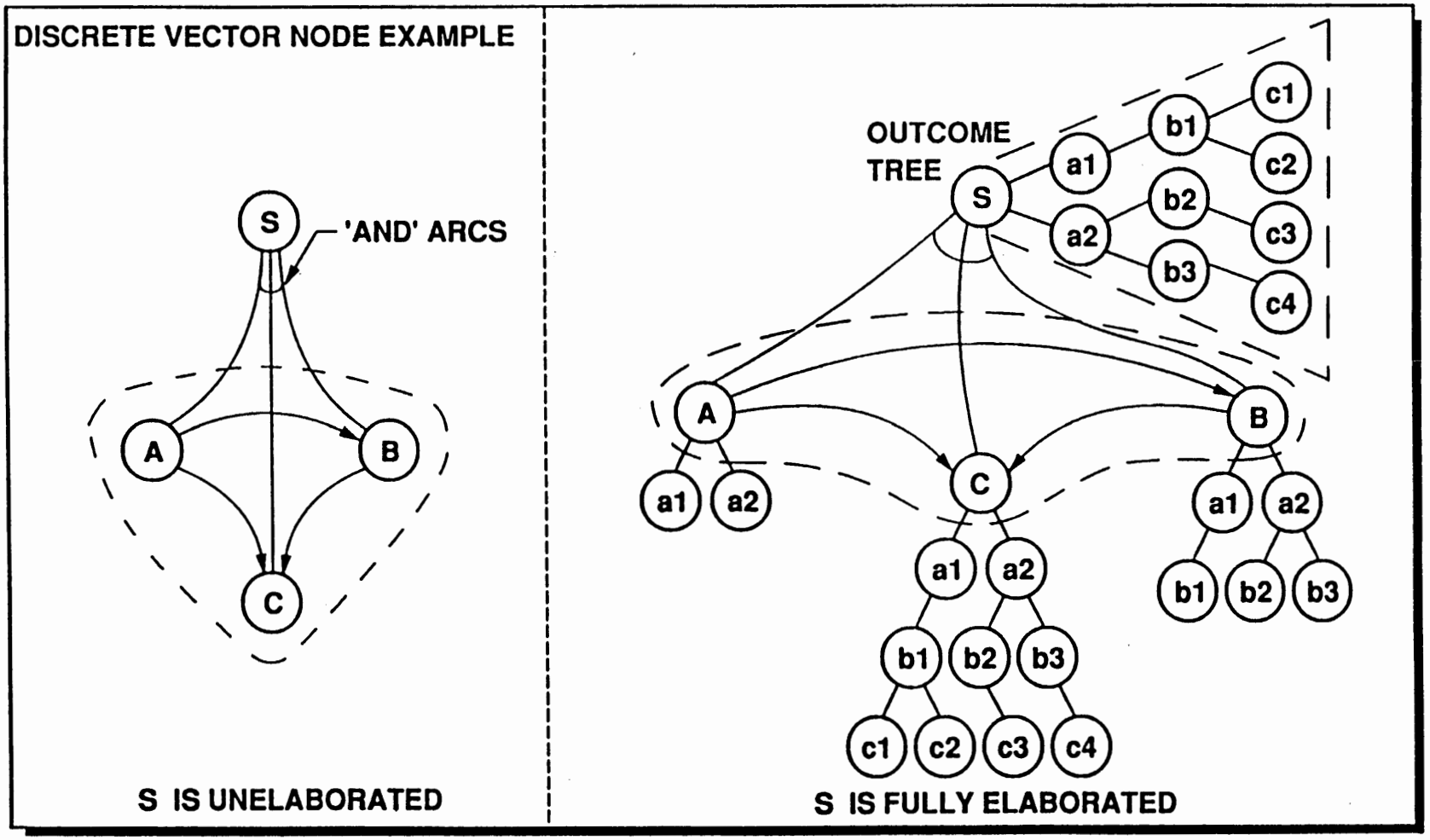
Variable A has two outcomes: (a1,a2) The associated probabilities are P(a1) and P(a2).

These two outcomes for A become versions for variables B and C. Assuming A=a1, then B has the single outcome b1 and conditional probability P(b1/a1) which would equal 1.0. Assuming A=a2, then B has two outcomes: (b2,b3). The associated probabilities are p(b2/a2) and p(b3/a2).

Variable C has a 2 level version tree, that is, an outcome from both A and B must be specified before the outcomes for C can be defined. For example, for A=a1 and B=b1, C has two outcomes: (c1,c2). Their probabilities are P(c1/a1,b1) and P(c2/a1,b1). For A=a2 and B=b2, C has one outcome: c3, and for A = a2 and B=b3, C has one outcome: c4.

It should be noted that the Influence Diagram is represented by the three root nodes (A,B,C) and the directed arcs. The other nodes represent data internal to the root nodes such as outcomes and versions.

DISCRETE INFLUENCE DIAGRAM



- S = {A, B, C} IS A VECTOR DISCRETE NODE

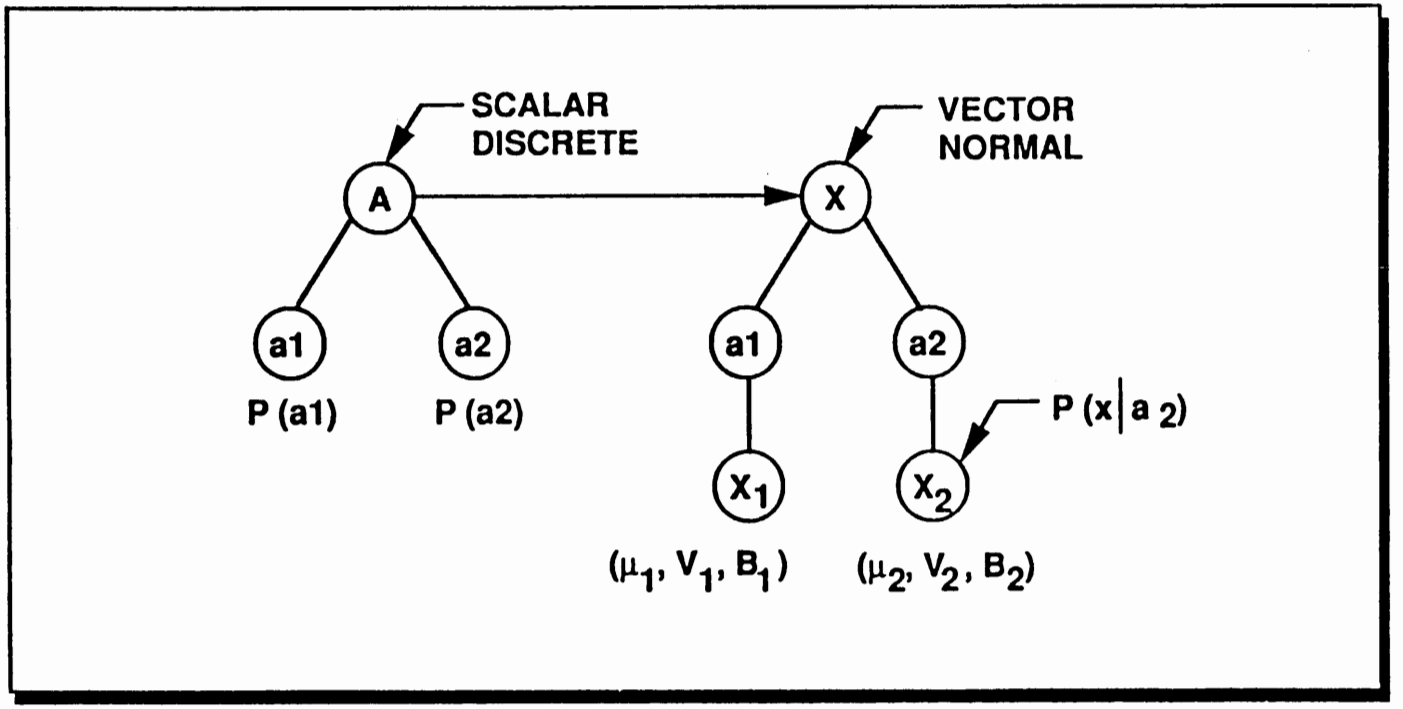
K9-7246/042

DISCRETE INFLUENCE DIAGRAM

This chart shows an example of a single discrete vector node. The vector node represents the joint random variable for a set of three scalar nodes, (A,B,C). The outcomes for S are joint outcomes for the three scalar nodes.

On the left side of the diagram, S is shown connected by 'and' arcs to the three nodes. The 'and' arcs indicate that S consists of A and B and C. The right side of the diagram shows the S node fully elaborated with the joint outcomes froma A, B and C.

MIXED DISCRETE PLUS NORMAL INFLUENCE DIAGRAM



- THE INFLUENCE OF THE DISCRETE RANDOM VARIABLE "SPLITS" THE NORMAL RANDOM VARIABLE INTO VERSIONS ("OR" SPLITS)

K9-7246/041

MIXED DISCRETE PLUS NORMAL INFLUENCE DIAGRAM

This chart shows an example of a scalar discrete node influencing a vector normal random variable. Each outcome of A becomes a version for X. Each version for X has an unconditional mean vector, conditional variance vector and internal arc strengths.

The discrete node can be viewed as splitting the normal node into versions. This splitting is termed an 'or' split to indicate that the X node is X1 (under A=a1) or X2 (under A=a2).

AGENDA

- INFLUENCE DIAGRAMS USED TO REPRESENT UNCERTAIN KNOWLEDGE IN COMPLEX SYSTEMS
  - GENERIC REPRESENTATION
  - APPLICATION TO MIDCOURSE TRACKING
- OPERATIONS ON INFLUENCE DIAGRAMS TO PERFORM INFERENCING
  - GENERIC OPERATIONS
  - APPLICATION TO MIDCOURSE TRACKING

INFLUENCE DIAGRAM NODES USED IN MIDCOURSE TRACKING

NODE	NAME	TYPE	SIZE	OUTCOMES
(Z)	FOCAL PLANE CONTACT	CTNS*	VECTOR (2 x 1)	(AZ, EL, INTENSITY, EXTENT (3))
(F)	FOCAL PLANE TRACK	CTNS	VECTOR (6 x 1)	(AZ, $\dot{AZ}$, $\ddot{AZ}$, EL, $\dot{EL}$, $\ddot{EL}$)
(X)	CARTESIAN TRACK	CTNS	VECTOR (6 x 1)	(X, Y, Z, $\dot{X}$, $\dot{Y}$, $\dot{Z}$)
(E)	EXTENT (ELLIPSE/ELLIPSOID)	CTNS	VECTOR (3 x 1/6 x 1)	(δAZ, δAZEL, δEL)/(δX, δXY, ...,δZ)
(C)	CONTACT ASSIGNMENT	DISCRETE	SCALAR	CONTACT TO TRK ASSIGNMENT ¦ NEW TRACK ¦ FALSE ALARM
(t^+)	TRACK UPDATE ASSIGNMENT	DISCRETE	SCALAR/ VECTOR	TRK TO CONTACT ASSIGNMENT ¦ MISSED DETECTION
(t^-)	TRACK PREDICTION ASSIGNMENT	DISCRETE	SCALAR/ VECTOR	NO SPLIT ¦ SPLIT INTO 2 ¦ SPLIT INTO 3, . . .
(S)	SCENE	DISCRETE	VECTOR	JOINT SET OF CONTACT, UPDATE AND PREDICTION ASSIGNMENTS

*ALL CONTINUOUS NODES ARE ASSUMED NORMAL

INFLUENCE DIAGRAM NODES USED IN MIDCOURSE TRACKING

This chart shows the nodes used in the midcourse tracking algorithm at this stage of development. The char shows the node symbol, the node name, its type (continuous or discrete), its size (scalar, discrete or both) and the outcomes for the node.

The focal plane contact is the random variable representing the estimated line of sight for a target or false alarm. (Note that the outcome column erroneously includes the intensity and extent.)

The focal plane track is the 6 element state vector for a track on the focal plane.

The cartesian track is the 6 element state vector for a track existing in 3-d space.

The extent can be either 3 state or 6 state and represents the parameters of an ellipse or ellipsoid, respectively of a cluster of objects.

The contact assignment variable is a discrete random variable that identifies the possible assignment outcomes for a contact. A contact can be an update to an existing track, the start of a new track or a false alarm.

The track update assignment random variable is a discrete random variable that identifies the possible assignment outcomes for a track. A track can be updated by one or more contacts or not be updated ar all (missed detection).

Thhe track prediction assignment random variable is a discrete random variable that identifies the possible dynamical models for the track. The models entertained for ballistic targets are that the track does not split or splits into two tracks or splits into 3 tracks or etc. These are 'and' splits.

The scene is a vector discrete random variable that represents the joint outcomes of a set of c, t+ and t- nodes.

NORMAL RANDOM VARIABLES USED IN MIDCOURSE TRACKING

CONTACT LINE OF SIGHT	FOCAL PLANE TRACK	CARTESIAN TRACK	FOCAL PLANE EXTENT
(Z) μ = (az, el) V (2x1) b (2x3)/2	(F) μ = (az, $\dot{az}$, $\ddot{az}$, el, $\dot{el}$, $\ddot{el}$) V (6x1) b (6x7)/2	(X) μ = (x, y, z, $\dot{x}$, $\dot{y}$, $\dot{z}$) V (6x1) b (6x7)/2	(E_3) μ = (δaz, δazel, δel) V (3x1) b (3x4)/2

CARTESIAN EXTENT	CONTACT STATE	TRACK STATE
(E_6) μ = (δx, δxy,.... δz) V (6x1) b (6x7)/2	(Z) — (E_3)	a. (F) — (E_3) b. (X) — (E_3) c. (X) — (E_6)

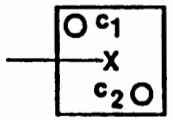
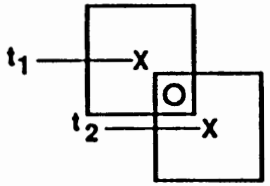
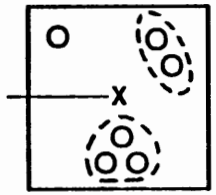
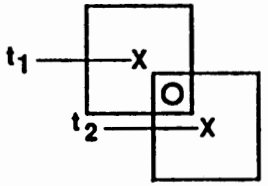
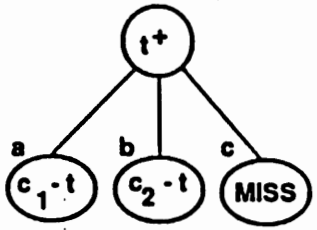
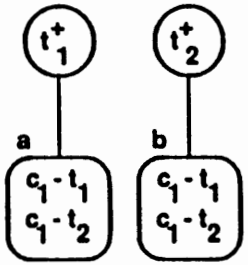
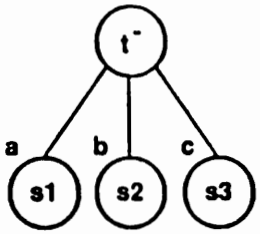
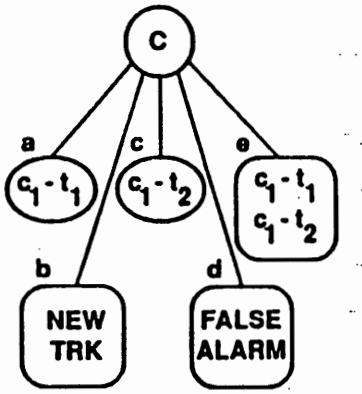
K9-7246/040

NORMAL RANDOM VARIABLES USED IN MIDCOURSE TRACKING

This chart details the normal random variables

The contact state combines the z and E3 node. The track state combines the F and E3, X and E3 or X and E6 nodes.

DISCRETE RANDOM VARIABLES USED IN MIDCOURSE TRACKING (1 OF 2)

TRACK TO CONTACT ASSIGNMENT	SHARED CONTACT ASSIGNMENT	TRACK SPAWN ASSIGNMENT	CONTACT TO TRACK ASSIGNMENT
			
			

a. ASSIGN c_1 TO t
b. ASSIGN c_2 TO t
c. ASSIGN MISS TO t

a. ASSIGN $_1$c TO $_1$t AND$_2$ t
b. ASSIGN $_1$c TO $_1$t AND$_2$ t

a. DO NOT SPAWN TRACK
b. SPAWN TRACK INTO TWO TRACKS
c. SPAWN TRACK INTO THREE TRACKS

a. ASSIGN c_1 TO t_1
b. c_1 IS A NEW TRACK
c. ASSIGN c_1 TO t_2
d. c_1 IS A FALSE ALARM
e. ASSIGN c_1 TO t_1 AND t_2

K9-7246/039

DISCRETE RANDOM VARIABLES USED IN MIDCOURSE TRACKING (1 OF 2)

This chart shows various examples for the discrete random variables.

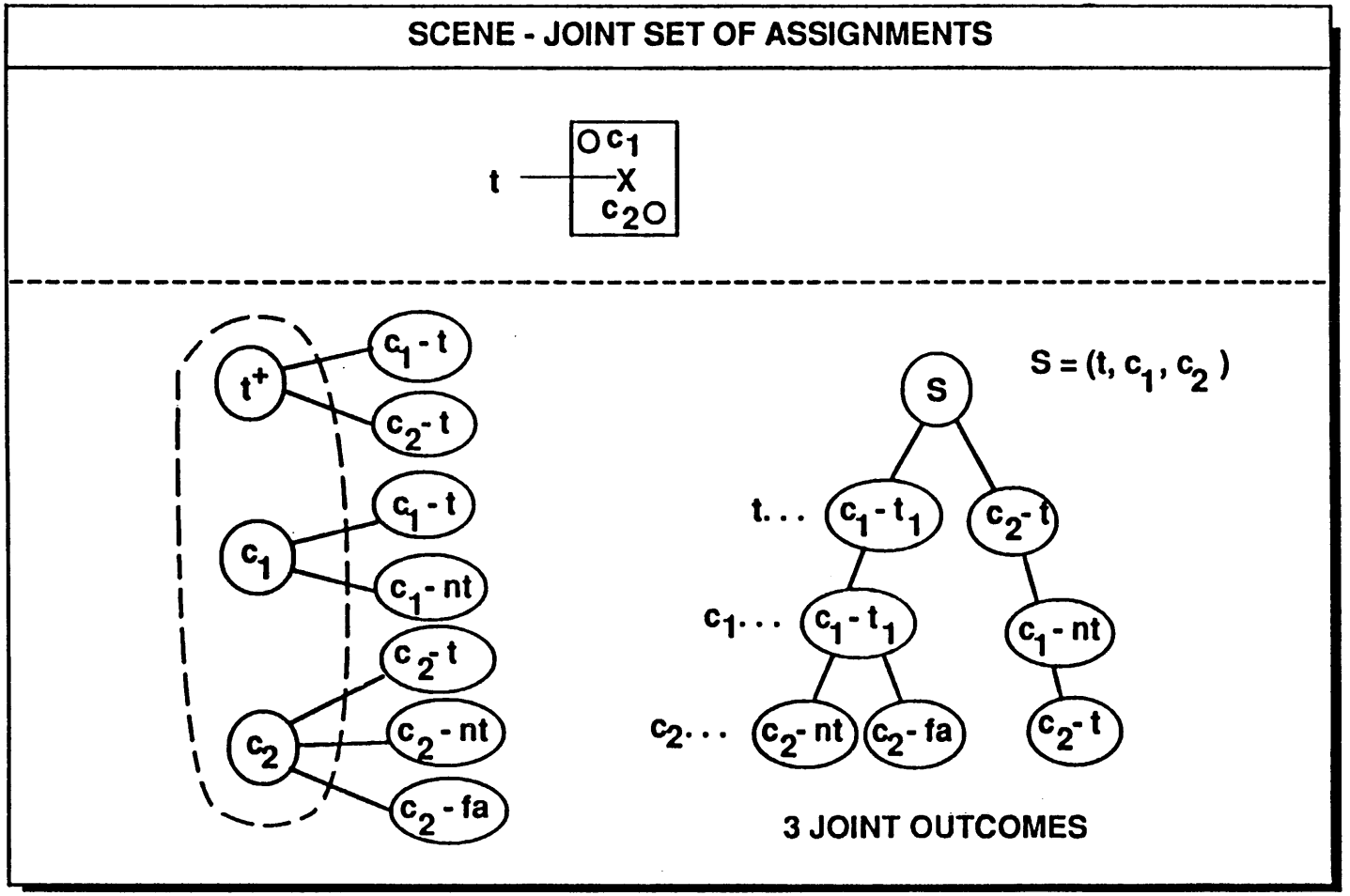
The first column shows the case of two contacts in a track gate. The update assignment node, t+, has three possible outcomes: c1 is assigned to the track (c1-t), c2 is assigned to the track (c2-t) or the track has a miss (miss).

The second column shows the special case of a shared contact. A single contact lies in the overlap region of two tracks and it has been determined that the contact should be assigned to both tracks at the same time. The shared outcome is represented as a set of contact to track assignements: c1-t1 and c1-t2.

The third column shows the case of track spawning ('and' splits). The example shows that the track can split into 3 tracks or into 2 track or not split at all.

The fourth column shows the case of a contact falling into the overlap region of two track gates. The contact assignment node has 5 possible outcomes: assign the contact to t1 only, assign the contact to t2 only, assign the contact to t1 and t2, consider the contact to be the start of a new track or consider the contact to be a false alarm.

DISCRETE RANDOM VARIABLES USED IN MIDCOURSE TRACKING (2 OF 2)



K9-7246/038

DISCRETE RANDOM VARIABLES USED IN MIDCOURSE TRACKING (2 OF 2)

This chart shows the outcomes for the scene node for the case of two contacts in the gate for a single track. The s-node represents the joint probability of the t+ node and the two c-nodes.

In this example, three joint outcomes are feasible. The first one assigns c1 to the track and declares that c2 is a new track. The second outcome assigns c1 to the track and declares that c2 is a false alarm. The third outcome assigns c2 to the track and declares that c1 is a new track.

EXAMPLES OF TRACKS (1 OF 4)

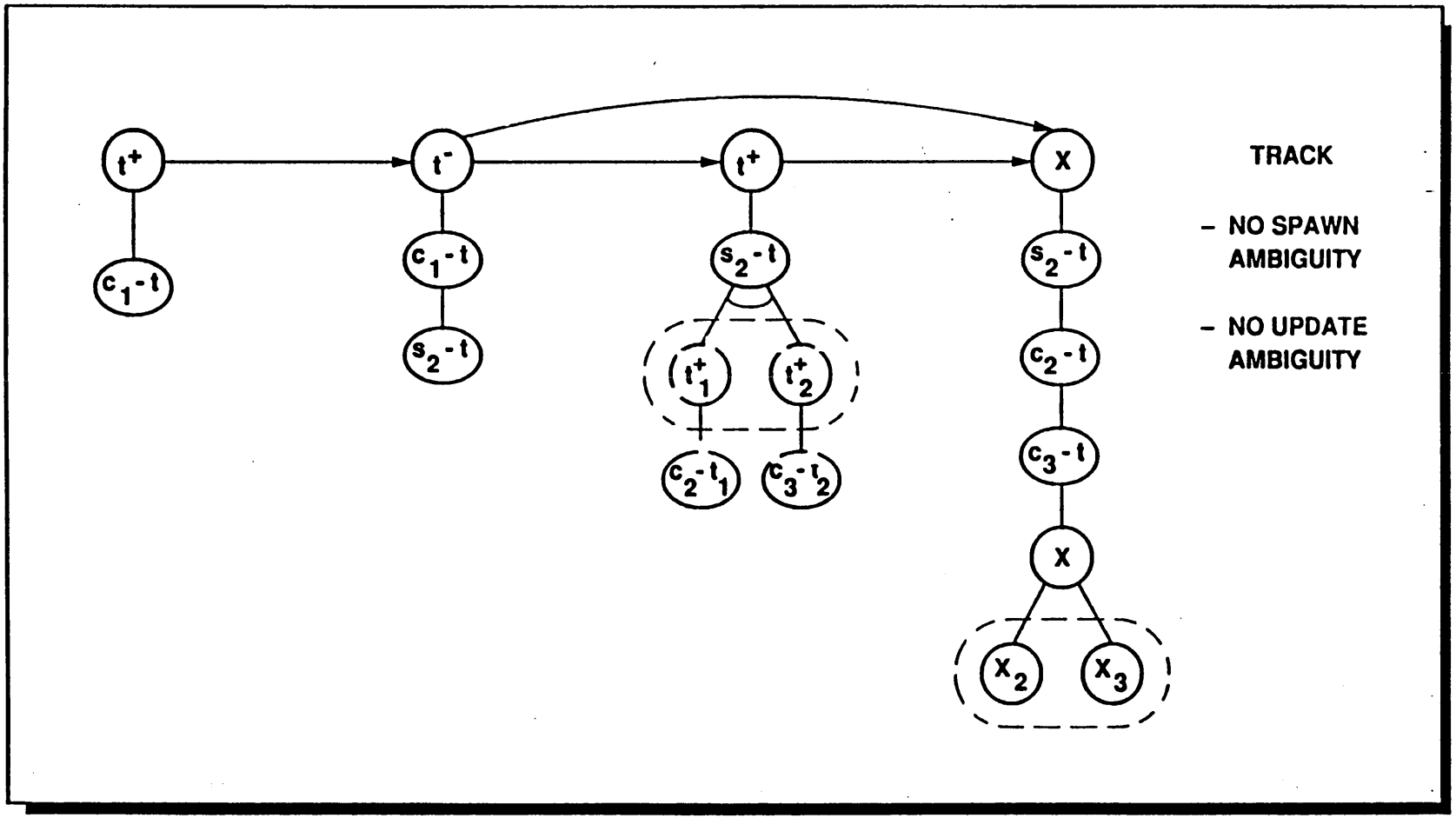
The next set of 4 charts show examples of the nodes making up various tracks in the case of a single satellite.

The first track is a new track. A c-node is given a new track and false alarm outcome which influences the track's first t+ node. The t+ considers the contact assignment outcome for the new track version and false track outcome for the false alarm version. The t+ node influences the F node which is split.

The second track is an urambiguous track with no spawn or update ambiguity. The first t+ node, which is from the previous frame, has a single contact assignment. The t- node has one outcome for the single assignment. The t+ node for the current frame has a single contact update.

The third track has an update ambiguity on the current frame. The track can be updated by one of two contacts. The X-node is split into two versions.

EXAMPLES OF TRACKS (2 OF 4)



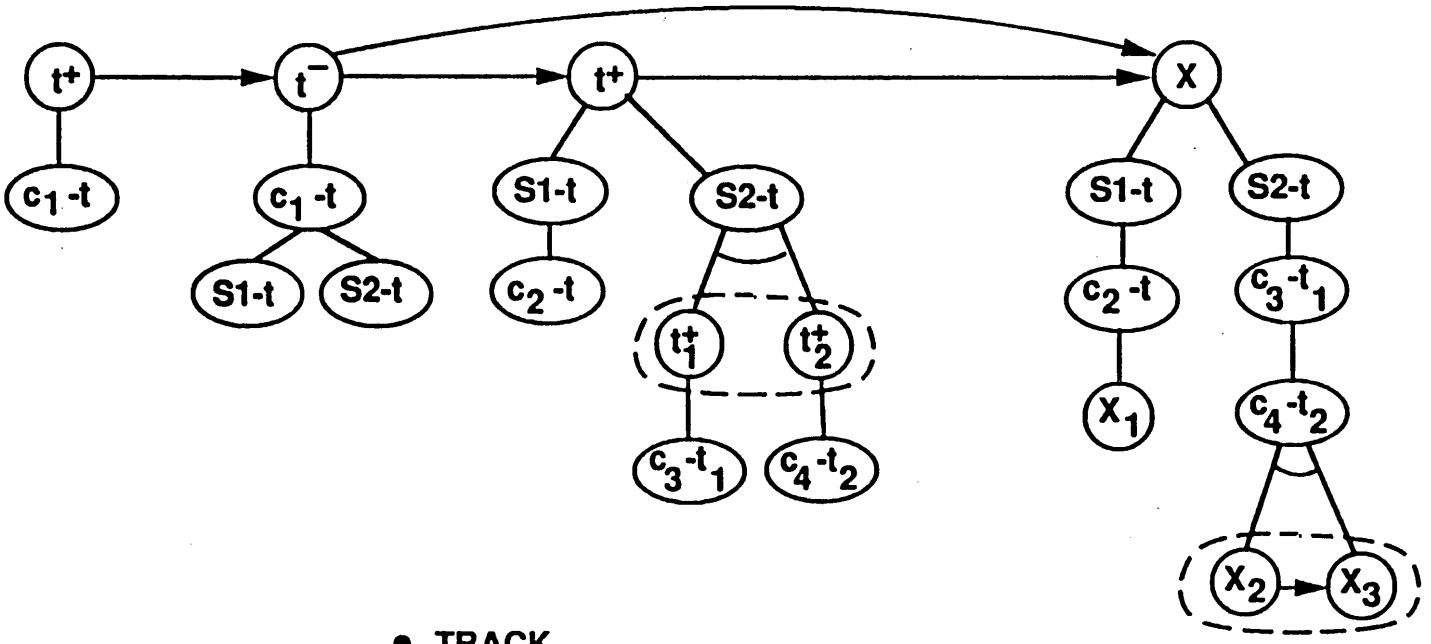
K9-7246/046

EXAMPLES OF TRACKS (2 OF 4)

This chart shows a track which has no ambiguity but shows how a spawn is handled.

The t- node has a single outcome which specifies that the track should be split into two tracks. As a result the t+ node generates a vector version, represented by the 'and' arcs coming out of the version node, s2-t. The vector version consists of two scalar t+ nodes one for each split track. In this case each scalar node has a single update assignment. The X-node is shown with the two track formation shown. (It should be noted that there should be an 'and' arc connecting the lines connecting the X-node with X2 and X3.)

EXAMPLES OF TRACKS (3 OF 4)



- TRACK
  - SPAWN AMBIGUITY
  - NO UPDATE AMBIGUITY

K9-7246/045

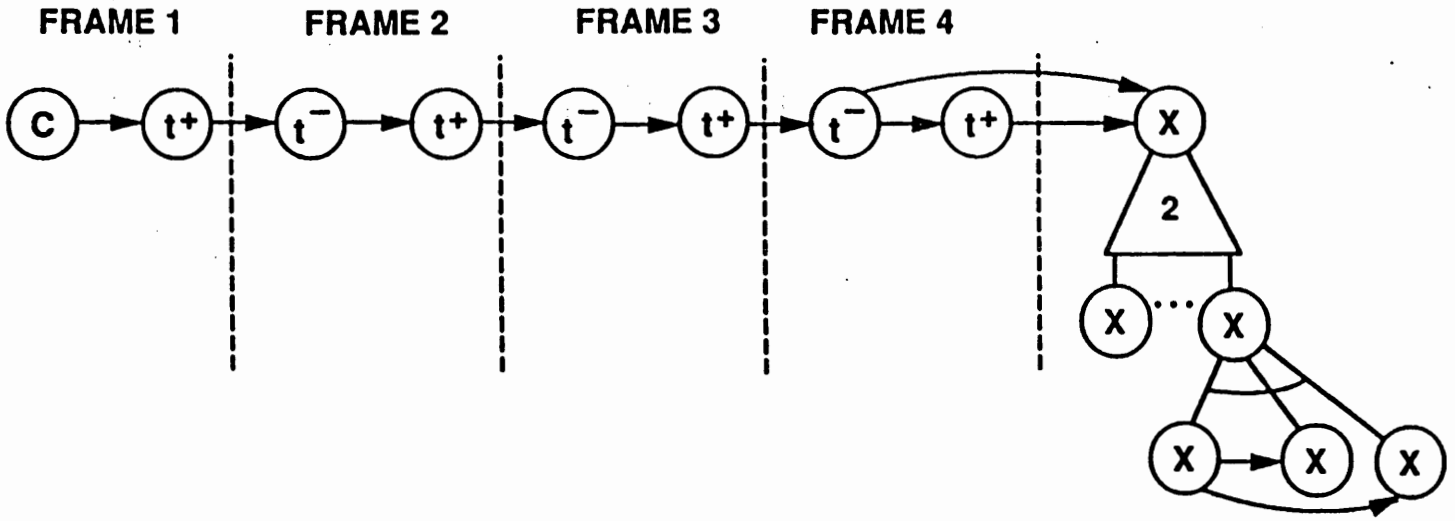
EXAMPLES OF TRACKS (3 OF 4)

This chart shows a track with a spawn ambiguity bu t no update ambiguity.

The t- node has two spawn outcomes: s1 and s2. The s1 outcome specifies that no spawning should take place. The s2 outcome specifies that the track should be split into two tracks thereby creating a two track formation. Each version of the t+ node has unambiguous update assignments leading to the X-node shown.

EXAMPLES OF TRACKS (4 OF 4)

COMPLETE TRACK HISTORY



- A MOVING WINDOW OF THE t NODE CHAIN IS MAINTAINED
  - WHEN THE FIRST NODE IN THE CHAIN HAS A SINGLE OUTCOME, IT IS DELETED AND THE CHAIN IS 'CLEANED UP'

EXAMPLES OF TRACKS (4 OF 4)

This chart shows the general configuration of discrete nodes and the continuous node for a track. If the c-node and t-nodes were not cleaned up, there would exist a chain of nodes beginning with the c-node and initial t+ node and continuing with a t- and t+ node pair on each frame. The t- and t+ nodes on the last frame influence the continuous node. The continuous node has a two level version tree with singleton track or formation track versions.

The chain is not allowed to grow indefinitely. As data is received, the outcome probabilities for the first node in the chain are updated and decisions are made to prune or select an outcome. When a single outcome remains for the first node in the chain, it is deleted and the chain is cleaned up. In this way, a moving window of t nodes is maintained.

AGENDA

- INFLUENCE DIAGRAMS USED TO REPRESENT UNCERTAIN KNOWLEDGE IN COMPLEX SYSTEMS
  - GENERIC REPRESENTATION
  - APPLICATION TO MIDCOURSE TRACKING
- OPERATIONS ON INFLUENCE DIAGRAMS TO PERFORM INFERENCING
  - GENERIC OPERATIONS
  - APPLICATION TO MIDCOURSE TRACKING

INFLUENCE DIAGRAM OPERATIONS

CONSTRUCTORS (1)	SELECTORS (2)		ITERATORS (3)	ERROR CONDITIONS
INITIALIZE	IS-EMPTY-THE-DIAGRAM	GET-OUTCOMES	DEPTH-FIRST-SEARCH	VERTEX-NUMBER-OVERFLOW
ADD-VERTEX	ITEM-OF-VERTEX	VERSIONS-OF	BREADTH-FIRST-SEARCH	VERTEX-IS-NULL
DELETE-VERTEX	IS-DISCRETE	IS-VERSION	LOCATION-OF	VERTEX-IS-NOT-IN-GRAPH
DUPLICATE-VERTEX	IS-CONTINUOUS	VERSION-BOUNDRY	FIND-PATH	VERTEX-HAS-REFERENCES
REPLACE	IS-NULL-THE-VERTEX	VERSION-PART	FIND-THE-ARC	ARC-IS-NULL
COPY-VERTEX	IS-A-MEMBER	TOP-VERSION	VISIT-VERSIONS	ARC-IS-NOT-IN-GRAPH
SET-ITEM-OF-VERTEX	ATTRIBUTE-OF-ARC	BOTTOM-VERSION	VISIT-VERTICES	ARC-IS-RELEVANT
ADD-OUTCOME	NUMBER-ARCS-FROM	DIRECT-PREDECESSORS	VISIT-ARCS	GRAPH-HAS-CIRCUIT
REMOVE-OUTCOME	SOURCE-OF	IS-PREDECESSOR	PARENT-OF	ITEM-NOT-DEALLOCATED
SINGLE-OUTCOME-CLEANUP	DESTINATION-OF	IS-SUCCESSOR	SUBTREE-OF	IS-NOT-REVERSIBLE
ADD-ARC	IS-NULL-THE-ARC	DIRECT-SUCCESSORS	TREE-OF	IS-NOT-A-ROOT-VERTEX
DELETE-NON-RELEVANT-ARC	IS-OUT-ARCS	WEAK-PREDECESSORS	ROOT-OF	IS-NOT-A-DISCRETE-VERTEX
DESTROY-ARC	IS-IN-ARCS	COMMON-PREDECESSORS		IS-NOT-A-CONTINUOUS-VERTEX
REVERSE ARC	IS-REVERSIBLE	IS-ROOT		IS-NOT-A-OUTCOME-VERTEX
REMOVE ARC	IS RELEVANT	IS-LEAF		IS-NOT-VALID-VERTEX-LABEL
SET-ATTRIBUTE-OF-ARC	ARC-EXISTS-BETWEEN	GET-LEAFS		PATH-NOT-FOUND
DESTROY-TREE	OUTCOME-PART	GET-PATH		BAD-SATELLITE-NUMBER
COPY-TREE	TOP-OUTCOME	LEAF-VERSIONS-OF		BAD-VERTEX-NUMBER
PRUNE-PATH	BOTTOM-OUTCOME	CHILDREN-OF		
PROPAGATE	IS-OUTCOME	SIZE-OF		
INSTANTIATE	IS-EQUAL	ROOT-OF		
NORMALIZE	GET-UNIQUE-OUTCOMES	RANDOM-VARIABLE-OF		
SEQUENCE				
PROJECT				
INCORPORATE				
INFER				

(1) CONSTRUCTORS ALTER THE STATE OF THE INFLUENCE DIAGRAM
(2) SELECTORS EVALUATE THE CURRENT STATE OF THE INFLUENCE DIAGRAM
(3) ITERATORS VISIT DIFFERENT PARTS OF THE INFLUENCE DIAGRAM

K9-7246/013

INFLUENCE DIAGRAM OPERATIONS

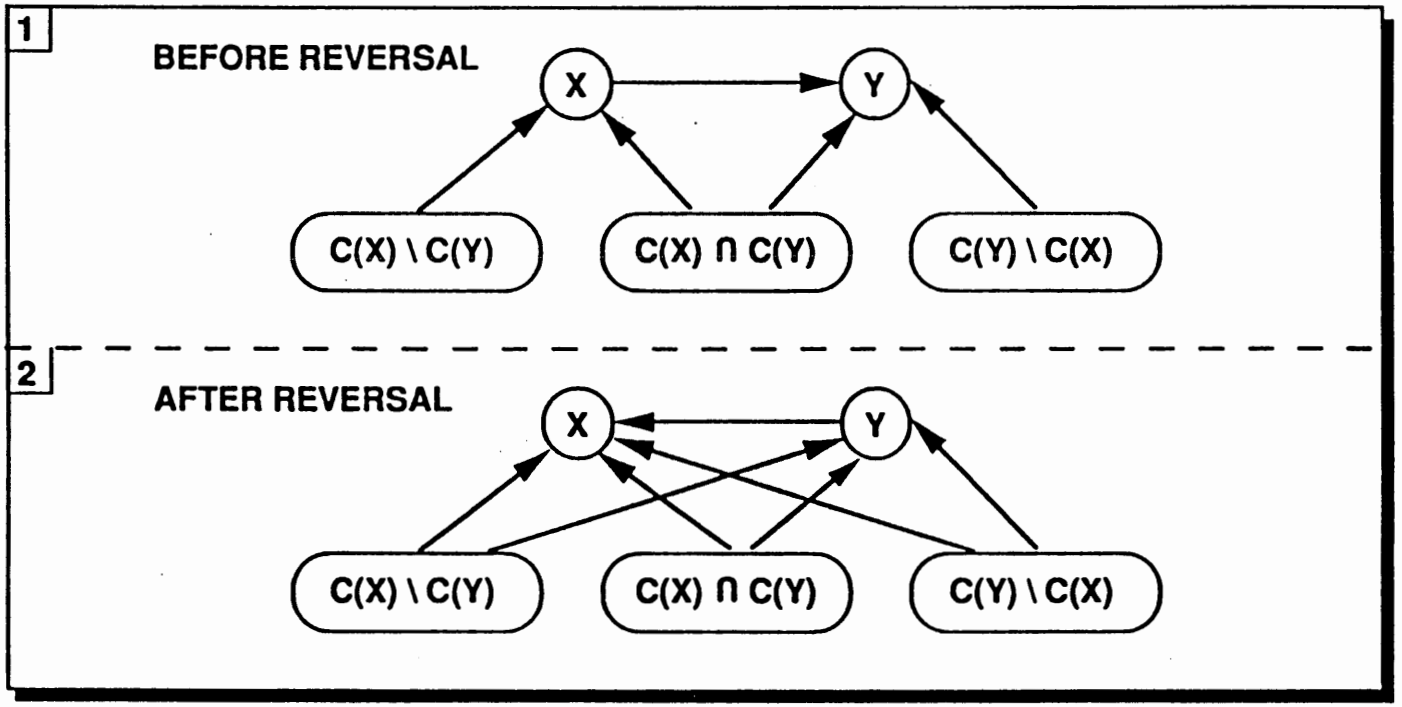
This chart shows the set of utilities created to construct, evaluate and scan the Influence Diagram. These utilities represent about 2 manyears of effort and are written in ADA.

The main routines used in the midcourse tracking algorithm are Reverse_Arc, Instantiate, Infer and Project. These will be discussed in the following charts.

The utility, Is_Relevent, should be mentioned. The routine examines arcs between continuous node to determine if their arc strengths are strong enough to be maintained., Thus, a tradeoff between the processing cost of maintaining arcs and the reduction in performance by deleting arcs can be attained.

REVERSE ARC

- REVERSE ARC IS THE INFLUENCE DIAGRAMS INPLEMENTATION OF BAYES' RULE



- EACH NODE INHERITS THE PREDECESSORS OF THE OTHER NODE

K9-7246/037

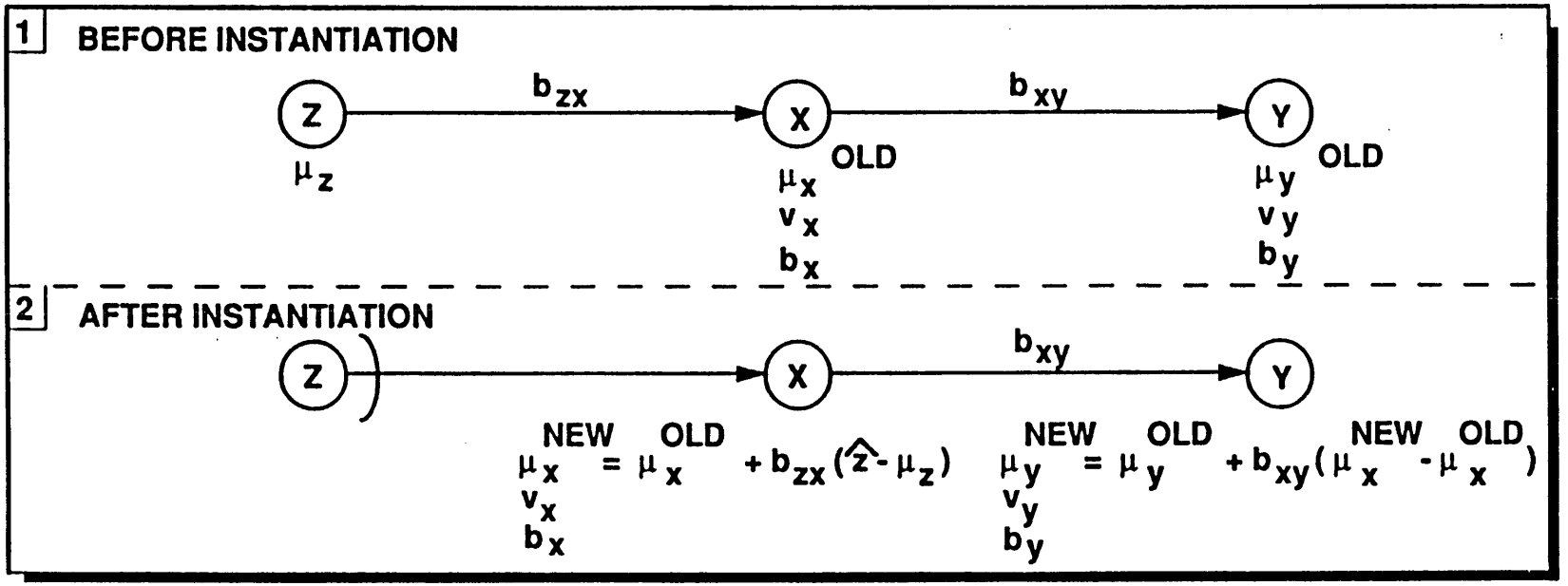
REVERSE ARC

Reverse_Arc is an important utility since it carries out Bayes' rule. In carrying out the arc reversal, each node inherits the predecessors of the other node. Reversing the same arc twice does not get back to the same diagram unless the arcs are tested for relevancy and the irrelevent arcs deleted.

Note that C(X) represents the set of conditional predecessors for the node X.

INSTANTIATE

- INSTANTIATE UPDATES THE DIAGRAM AS A RESULT OF A MEASUREMENT OF A RANDOM VARIABLE



- THE INSTANTIATE ACTION FLOWS THROUGH THE DIAGRAM UPDATING THE CONDITIONAL MEAN OF THE SUCCESSORS, THE SUCCESSORS OF THE SUCCESSORS, ETC. OF THE INSTANTIATED VARIABLE
- AFTER INSTANTIATION, THE INSTANTIATED VARIABLE CAN BE DELETED

INSTANTIATE

Instantiate is the utility which incorporates a measured value for a continuous random variable. In the chart, diagram is shown before and after the variable z is instantiated.

When z is instantiated with the measured value, the unconditional mean of X is updated. Furthermore, because the unconditional mean of X is update then the unconditional mean of Y is also updated. The other parameters in the diagram are unchanged.

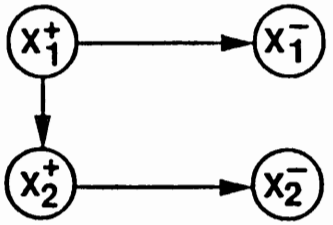
After z is instantiaed, it can be deleted since it is no longer a random variable.

INFER

- INFER IS AN ORDERED SEQUENCE OF ARC REVERSALS TO STRUCTURE THE INFLUENCE DIAGRAM TO REPRESENT A DESIRED CONDITIONAL PROBABILITY

1

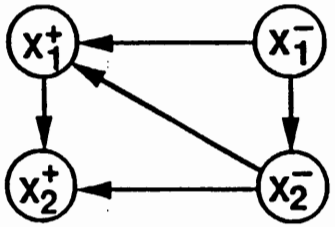
BEFORE INFERENCE OF $\{ X_1^-, X_2^- \}$ ONTO $\{ X_1^+, X_2^+ \}$



DESIRE $P[X_1^+, X_2^+ / X_1^-, X_2^-]$

2

AFTER INFERENCE OF $\{ X_1^-, X_2^- \}$ ONTO $\{ X_1^+, X_2^+ \}$



K9-7246/043

INFER

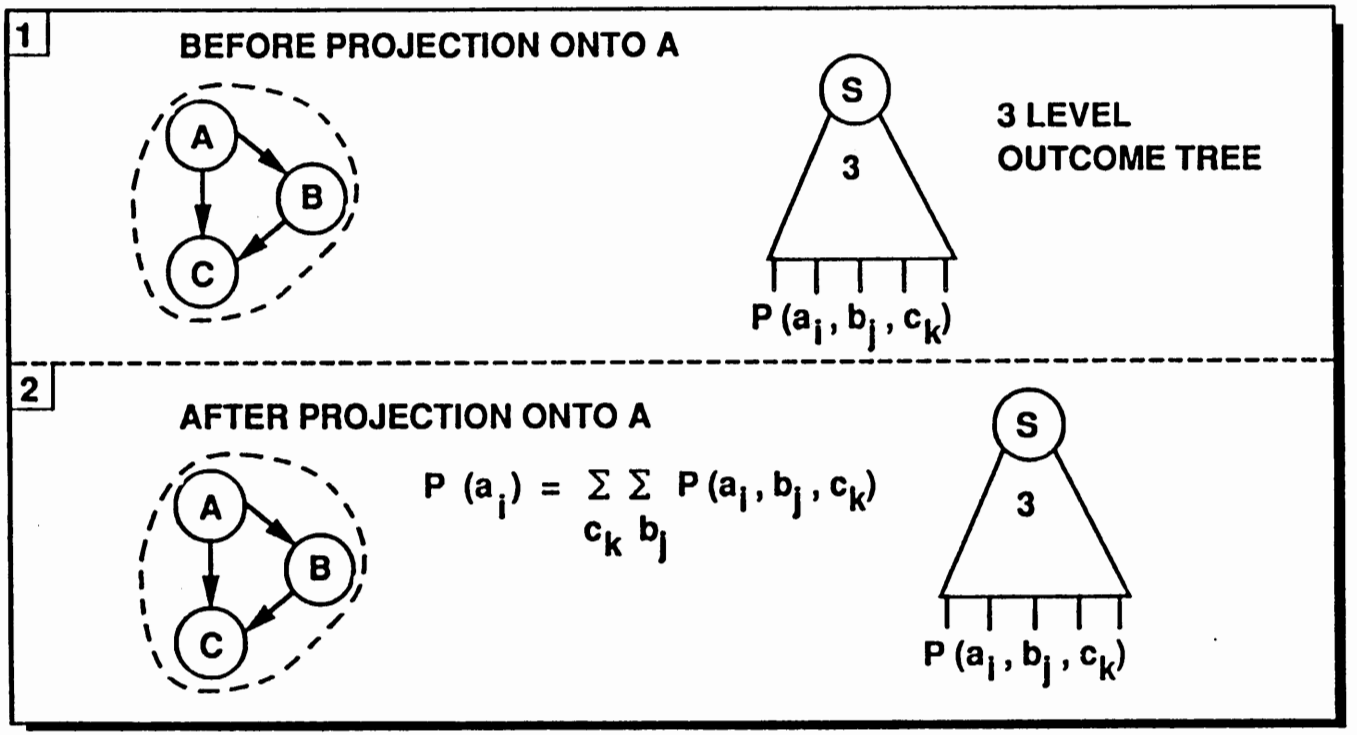
Infer is a major function used in the midcourse tracking algorithm. It carries out an order sequence of arc reversals to structure the Influence Diagram into a desired form.

In the example, it is desired that the set {X1-, X2-} influence {X1+, X2+}. The Infer utility carries out an ordered sequence of reversals. The order to determined to assure that no loops exist in the diagram.

The diagram is shown after Inference. Note that an additional arc from X2- to X1+ is generated.

PROJECT

- PROJECT CALCULATES THE PROBABILITIES OF THE OUTCOMES OF RANDOM VARIABLES IN A SUBDIAGRAM GIVEN THE PROBABILITIES OF THE JOINT OUTCOMES OF THE SUBDIAGRAM



- PROJECT 'SUMS OUT' THE UNWANTED OUTCOMES IN THE JOINT RANDOM VARIABLE

K9-7246/035

PROJECT

Project is the utility that takes the joint outcome probabilities for a set of random variablesand calculates the marginal probabilities for a given random variable in the set.

In the example, the variable S represents the variables {A,B,C}. It is assumed that the joint outcome probabilities exist, P(ai,bj,ck). Next, the marginal probability p(ai) is desired. The Project operator carries out the summation as shown in the chart to calculate P(ai).

AGENDA

- INFLUENCE DIAGRAMS USED TO REPRESENT UNCERTAIN KNOWLEDGE IN COMPLEX SYSTEMS
  - GENERIC REPRESENTATION
  - APPLICATION TO MIDCOURSE TRACKING
- OPERATIONS ON INFLUENCE DIAGRAMS TO PERFORM INFERENCING
  - GENERIC OPERATIONS
  - APPLICATION TO MIDCOURSE TRACKING

INFLUENCE DIAGRAM OPERATIONS APPLIED TO MIDCOURSE TRACKING

- **STATE ESTIMATION**
  - KALMAN FILTER PROCESSING
  - FORMATION TRACK UPDATE
  - TRACK SPAWNING ('AND' SPLITS)
  - SHARED CONTACT UPDATE
- **ASSOCIATION**
  - TRACK UPDATE/MISS PROCESSING (t^+ NODE)
  - TRACK SPAWN PROCESSING (t^- NODE)
  - CONTACT UPDATE/NEW TRACK/FALSE ALARM PROCESSING (C NODE)
  - SCENE PROCESSING (S NODE)

INFLUENCE DIAGRAM OPERATIONS APPLIED TO MIDCOURSE TRACKING

This chart summarizes the agenda for the remaining part of the presentation. It shows four major state estimation and four major association functions to be described in terms of Influence Diagram operations.

MATHEMATICAL MODEL

DYNAMIC PROCESS: $x(k+1) = \Phi(k)\, x(k) + \Gamma(k)\, w(k)$ $\quad k = 0, \ldots, N.$

MEASUREMENT PROCESS: $z(k) = H(k)\, x(k) + v(k)$ $\quad k = 0, \ldots, N.$

PROBABILISTIC STRUCTURE:

$E[x(0)] = \mu_0.$

$COV[x(0)] = P_0.$

$E[w(k)] = 0$ FOR $k = 0, \ldots, N.$

$COV[w(j), w(k)] = \delta_{jk} Q_k$ FOR $j = 0, \ldots, N$ AND $k = 0, \ldots, N.$

Q_k ARE DIAGONAL FOR $k = 0, \ldots, N.$

$COV[x(0), w(0)] = 0.$

$E[v(k)] = 0$ FOR $k = 0, \ldots, N.$

$COV[v(j), v(k)] = \delta_{jk} R_k$ FOR $j = 0, \ldots, N$ AND $k = 0, \ldots N.$

R_k ARE DIAGONAL FOR $k = 0, \ldots, N.$

$COV[w(j), v(k)] = 0$ FOR $j = 1, \ldots, N$ AND $k = 0, \ldots, N.$

$COV[x(0), v(k)] = 0$ FOR $0 = 1, \ldots, N.$

DIMENSIONS OF VECTORS: $x(k) \in R^n$, $w(k) \in R^r$, $z(k) \in R^P$, AND $v(k) \in R^P$.

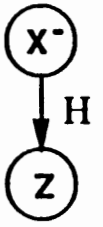
DISCRETE - TIME FILTERING

This chart presents the assumptions made in the discrete-time Kalman filtering model.

KALMAN FILTER PROCESSING CYCLE

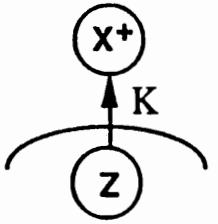
1 PREDICTED TRACK

2 ASSIGN TRACK TO MEASUREMENT



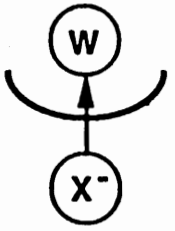
- CONSTRUCT DIAGRAM

3 UPDATE TRACK



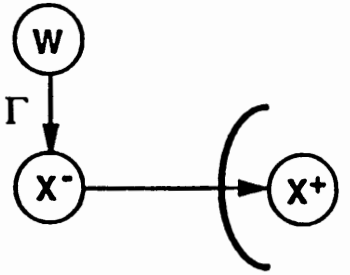
- REVERSE (X^-, Z) ARC
- INSTANTIATE Z
- ELIMINATE Z

6 ADD SYSTEM NOISE



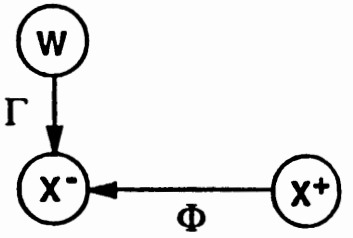
- REVERSE (W, X^-) ARC
- ELIMINATE W

5 PREDICT TRACK STATE



- REVERSE (X^-, X^+) ARC
- ELIMINATE X^+

4 PREDICT TO NEXT FRAME



- CONSTRUCT DIAGRAM

K9-7246/005

KALMAN FILTER PROCESSING CYCLE

This chart shows the cycle of Influence Diagram operations used to carry out the Kalman filter processing cycle.

[1] The state vector is predicted to the next measurement update.

[2] A measurement is assigned to the state vector. An arc from X- to Z in constructed and the measurement matrix, H, is placed on the arc.

[3] The track is updated. First the arc is reversed which updates the covariance matrx for the state vector. Next the Z node is instantiated which updates the mean of the state vector.

[4] The track is set up for prediction to the next frame. First, a construction phase is carried out in which an arc is created between the updated state vector, X+, and the (to be) predicted state, X-. Also, an arc from the system noise vector, W, to the predicted state is also constructed. The propagation matrix is placed on the X+ to X- arc, and the Gamma matrix is placed on the W to X- arc.

[5] The track is predicted. The X+ to X- arc is reversed which calculates the predicted covariance matrx. (Note that the predicted state is calculated outside of the Influence Diagram for the case of the extended Kalman filter.)

[6] The system noise is incorporated. The W to X- arc is reversed which has the effect of adding the Q matrix to the predicted covariance.

DISCRETE - TIME FILTERING

WEIGHTED OPERATION COUNTS FOR PROCESSING A VECTOR OF p MEASUREMENTS

ALGORITHM	WEIGHTED OPERATION COUNTS
INFLUENCE DIAGRAM	$(3.6n^2 + 12.3n)\,p$
CONVENTIONAL KALMAN	$(3.6n^2 + 7.8n + 4.5)\,p$
U-D COVARIANCE	$(3.6n^2 + 15.7n)\,p$
SQUARE ROOT COVARIANCE	$(4.3n^2 + 40.9n)\,p$
POTTER SQUARE ROOT	$(7.2n^2 + 8.6n + 30.4)\,p$
KALMAN STABILIZED	$(10.1n^2 + 16n + 4.5)\,p$
SRIF, R TRIANGULAR	$(2.4n^2 + 6.2n)\,p + 4.3n^2 + 37.1n$
NORMAL EQUATION	$(1.2n^2 + 5.0n)\,p + 0.4n^3 + 3.1n^2 + 30.3n$
SRIF, R GENERAL	$2.4n^2p + 1.6n^3 + 2.6n^2 + 31n$

OPERATION	WEIGHT
+	1
X	1.4
÷	4.5
√	21.4

DISCRETE-TIME FILTERING

This chart shows the weighted operation counts for the update phase of the Kalman filter processing cycle. It can be seen that the Inluence Diagram implementation compares favorably with the conventional Kalman implementation which is the most efficient. The Influence Diagram implementation requires less throughput than the other versions shown.

One other advantage of the Influence Diagram implementation is that it guarantees a positive semidefinite covariance matrix since the variance terms are calculated by summing positive quatities. Subtractions, which can cause numerical instabilities, are not required.

WEIGHTED OPERATION COUNTS FOR TIME UPDATE

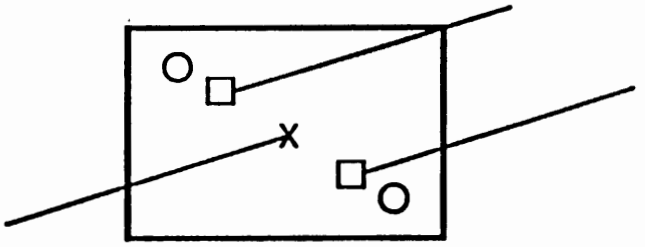
ALGORITHM	WEIGHTED OPERATION COUNTS
INFLUENCE DIAGRAM	$2.8n^3 + 3.95n^2 - 11.55n + 10 + (6n^2 + 2.7n - 4.9)\,r + (2.4n - 2.4)\,r^2$
CONVENTIONAL KALMAN	$3.6n^3 + 4.1n^2 + 0.5n + (1.2n^2 + 2.6n)\,r$
U-D COVARIANCE	$3.6n^3 + 4n^2 + 3.1n - 4.5 + (2.4n^2 + 4.2n - 2.8)\,r$
SQUARE ROOT COVARIANCE	$4n^3 + 4.8n^2 + 26.7n + (2.4n^2 + 2.4n)\,r$

DISCRETE-TIME FILTERING

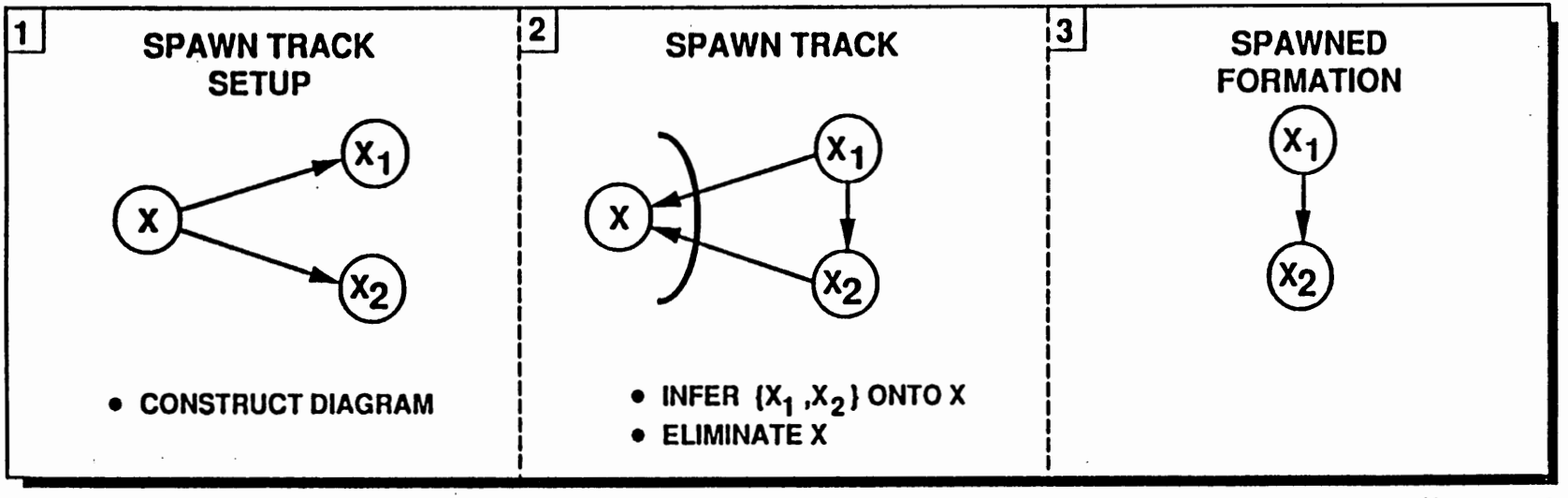
This chart shows the weighted operation counts for the time update portion of the Kalman filter. Again, the Influence Diagram implementation performs well against the implementations shown.

TRACK SPAWNING ('AND' SPLITS)

- SPLIT TRACK X INTO X_1 AND X_2
- X_1 & X_2 BECOME A TWO-TRACK FORMATION



□ ESTIMATED STATE
X PREDICTED STATE
○ CONTACT



K9-7246/006

TRACK SPAWNING ('AND' SPLITS)

This chart shows the Influence Diagram operations involved in performing a splitting of a track into two tracks. Suppose two contacts fall into the correlation gate for the track and it is determined that the track has split into two tracks.

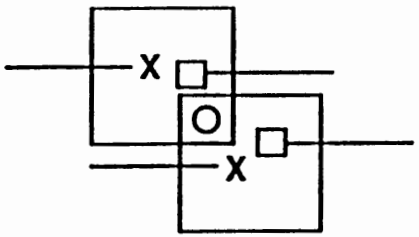
[1] The Influence Diagram for a Track Spawn is constructed. Two new state vectors are created and an arc from the previous state vector, X, to the two new states, X1 and X2, are constructed. Next, the appropriate arc strengths are placed on the arcs. Likewise, the data in the X1 and X2 vertices are set. The data includes the unconditional means, conditional variances and internal arc strengths.

[2] The arcs between X1 and X2 and X are reversed. As a result, and arc from X1 and X2 is created. Thus a two track formation is created. The strength of the arc between the tracks depends upon the conditional variances set in X1 and X2 and the arc strengths from X to X1 and X2. The term, 'formation', is used whenever an influence arc exists between the tracks.

[3] The X node is deleted leaving the spawned formation.

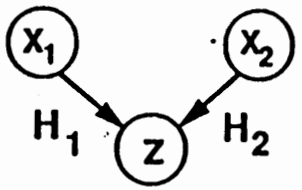
SHARED CONTACT UPDATE

- TWO TRACKS SHARE THE SAME CONTACT. BOTH TRACKS ARE UPDATED WITH SAME CONTACT



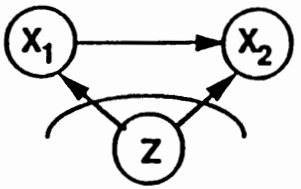
□ ESTIMATED STATE
X PREDICTED STATE
O CONTACT

1 ASSIGN TRACKS TO ONE CONTACT



- CONSTRUCT DIAGRAM

2 UPDATE TRACKS



- INFER Z ONTO $\{X_1, Z_2\}$
- ELIMINATE Z

3 TRACK FORMATION

K9-7246/007

SHARED CONTACT UPDATE

This chart shows the Influence Diagram operations involved in updating two tracks with the same contact. This situation occurs when the sensor has less resolution than the track file or the tracks are crossing from the sensor's perspective.

[1] The contact is assigned to both tracks. The Influence Diagram is constructed with arcs from the tracks, X1 and X2, to the contact, Z. The measurement matrices are placed on the arcs and the appropriate data is set in the Z node.

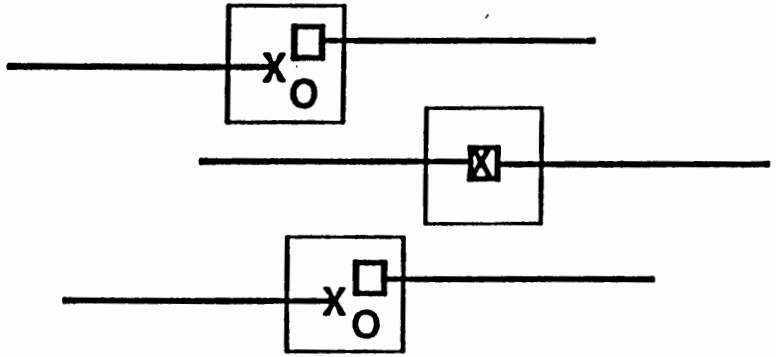
[2] Inference of the Z node onto the X nodes is carried out and the Z node instantiated and then eliminated. This stage updates the covariances and state vectors of the tracks.

[3] A two track formation is created.

It should be noted that one of the advantages of the Influence Diagram implementation is that all relevent probabilistic influences are maintained automatically as part of the Influence Diagram operations.

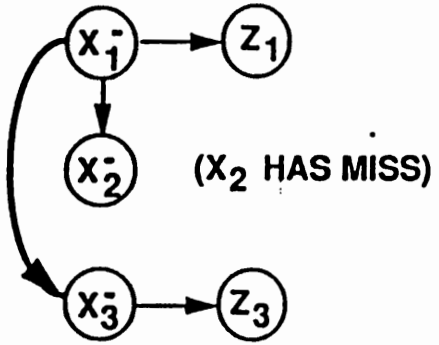
FORMATION TRACK UPDATE

- UPDATE FORMATION TRACK WITH A SET OF CONTACTS



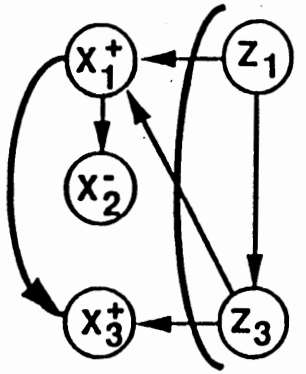
□ ESTIMATED STATE
X PREDICTED STATE
O CONTACT

1 ASSIGN TRACKS TO CONTACTS



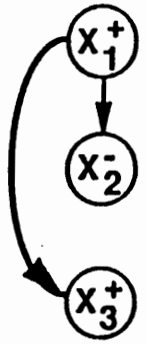
- CONSTRUCT DIAGRAM

2 UPDATE TRACKS



- INFER {Z} ONTO {X}
- INSTANTIATE {Z}
- ELIMINATE {Z}

3 UPDATED FORMATION TRACK



K9-7246/012

FORMATION TRACK UPDATE

This chart shows the Influence Diagram operations involved in updating a formation track.

The example shows a 3 track formation in which two of the tracks have an update and the third track has a miss.

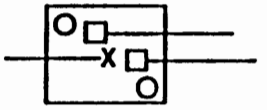
[1] The contacts are assigned to the tracks and the Influence Diagram is constructed.

[2] The tracks are updated. Inference of the measurements onto the state vectors is carried out; the measurement nodes are instantiated and then eliminated.

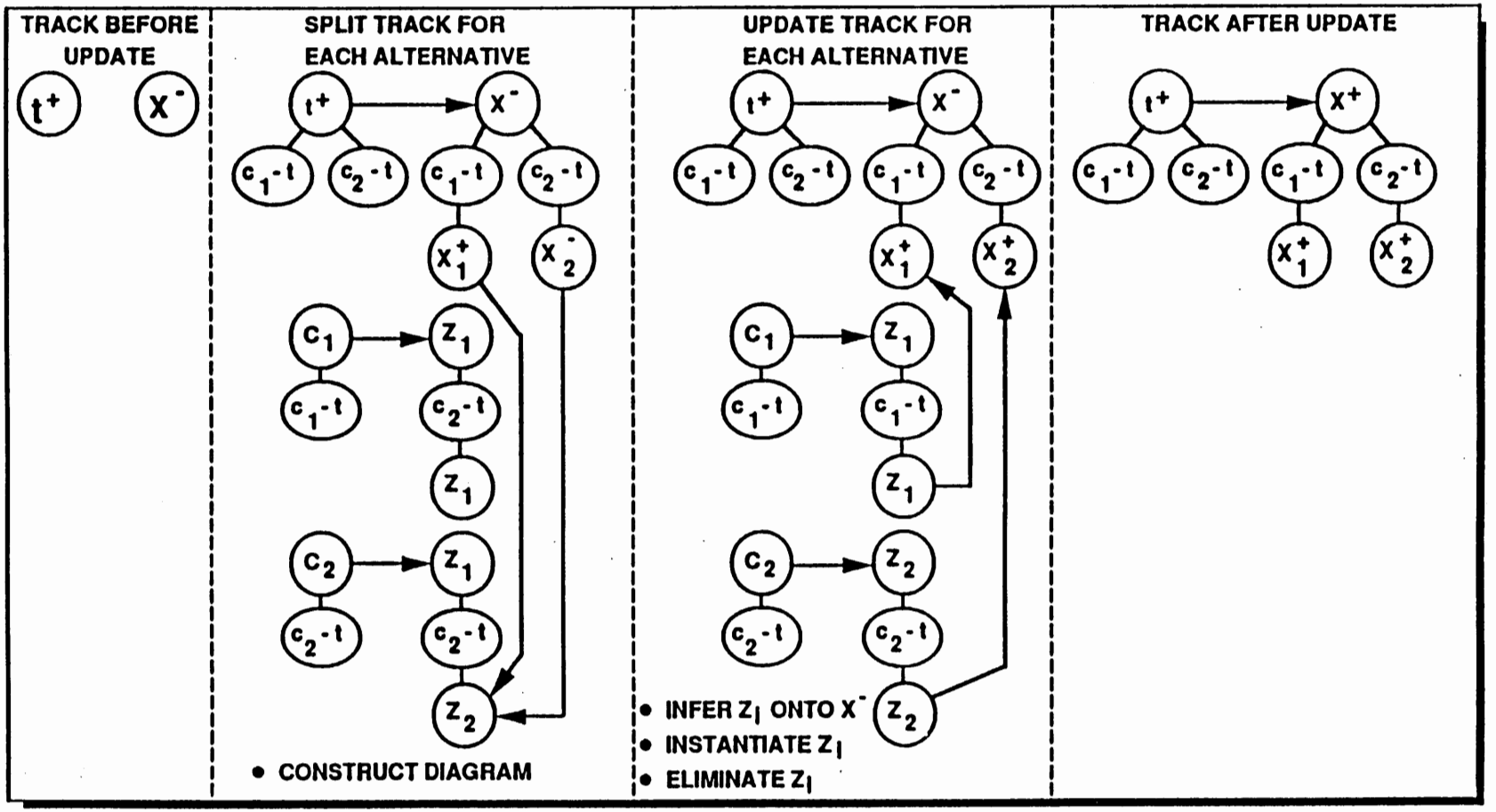
[3] The formation track is updated. Note that the track with the miss maintains the propagated covariance and state vector.

t^+ TRACK SPLITTING (' OR ' SPLITS)

- TRACK HAS MULTIPLE CONTACTS IN THE GATE



X PREDICTED STATE
□ UPDATED STATE
O CONTACT



K9-7246/017

t+ TRACK SPLITTING ('OR' SPLITS)

This chart shows the Influence Diagram operations involved in splitting a track into alternative tracks due to multiple contacts in the gate. In this example, two contacts fall into the gate.

[1] Before the update processing, a single state vector, X-, and an unelaborated update assignment node, t+, exist.

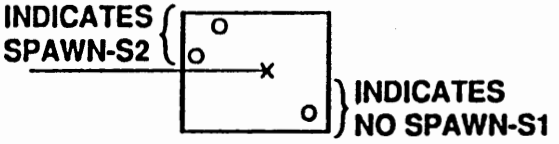
[2] The contact assignments are added as outcomes to the t+ node. An arc is added from the t+ node to the X- node which causes the outcomes to flow to the X- node, thereby creating two x node versions. (An error exists in the chart. X1+ should be X1- and should have an arc to Z1.) The update assignments are also placed under the C nodes and arcs created from the C nodes to the Z nodes. Finally, in order to start the update process, a continuous arc from X1- to Z1, and X2- to Z2 are created and the measurement matrices placed on the arcs.

[3] The arcs are reversed; the Z nodes instantiated and then eliminated.

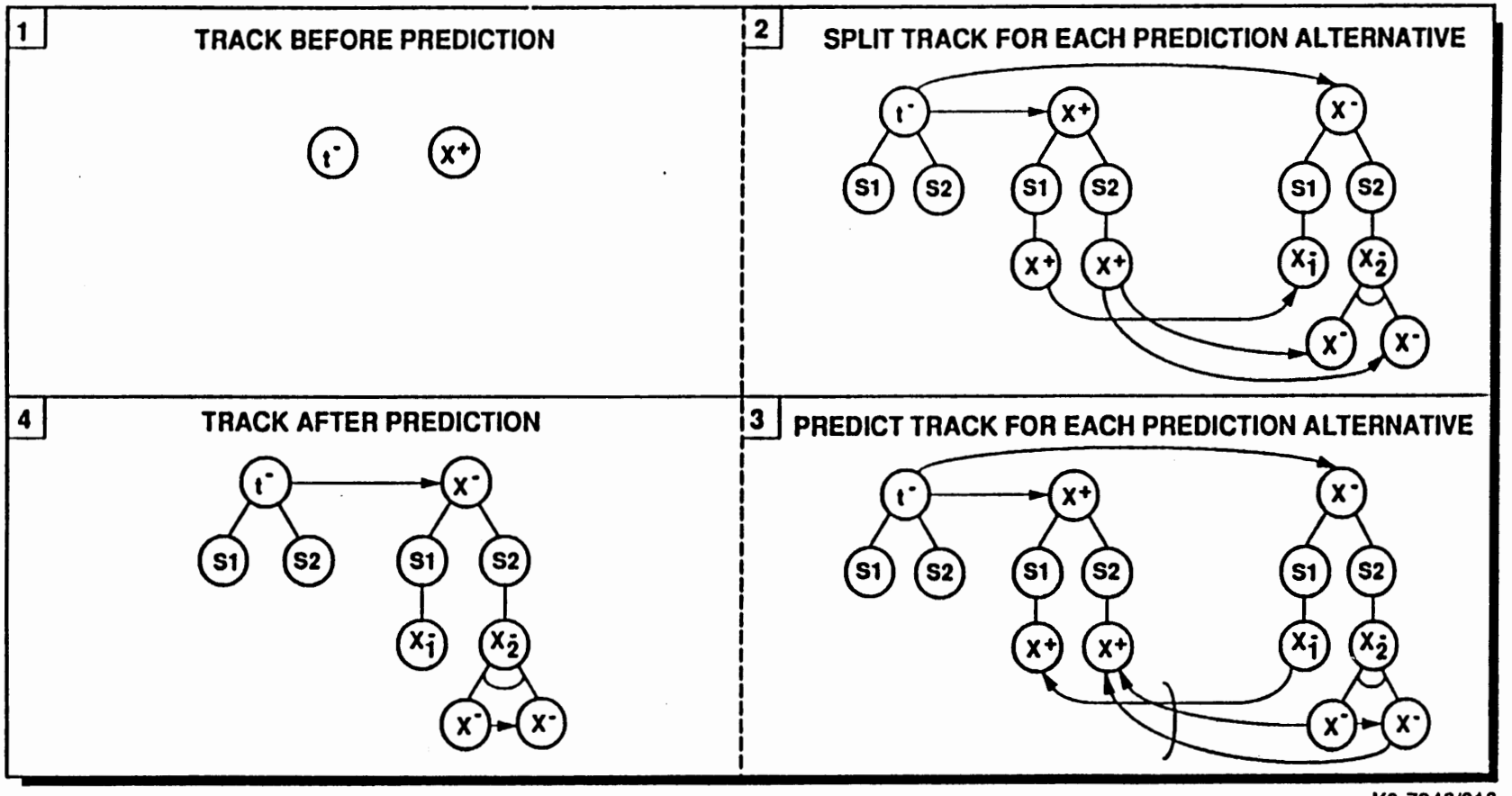
[4] The track after the update shows two updated versions of the track state vector.

t^- NODE PROCESSING

- TRACK HAS MULTIPLE SPAWN ALTERNATIVES



X = PREDICTED STATE
O = CONTACT



K9-7246/018

t- NODE PROCESSING

This chart shows the Influence Diagram operations involved with track prediction processing. This process considers alternative spawn hypotheses in which a track splits into more than one track thereby creating a formation.

[1] Before the prediction processing, an updated state vector exists and an unelaborated t- node.

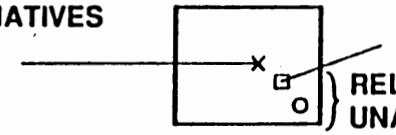
[2] It is determined that the track either spawns a 2 track formation or remains a singleton track. As a result, an S2 assignment and an S1 assignment are added as outcomes to the t- node. An arc is added from t- to the X+ node and the outcomes flow to the X+ node creating two alternatives. Likewise, a predicted state vector, X-, iscreated and an arc from t- to X- is created and the outcomes flow to create two versions for X-. The S2 version creates 2 state vectors. Finally, arcs are added from the X+ versions to the X- versions and the data set on the arcs and in the nodes.

[3] The arcs are reversed and the predicted state vectors are calculated. The X+ node can then be deleted.

[4] After prediction processing, a predicted 2-track version and a singleton track version exist.

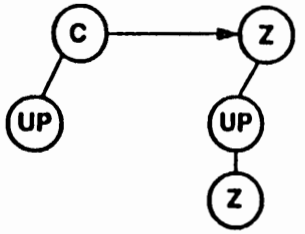
C NODE PROCESSING

- CONTACT FALLS IN GATE BUT IS NOT A GOOD FIT TO TRACK SO NEW TRACK AND FALSE ALARM ARE FEASIBLE ALTERNATIVES

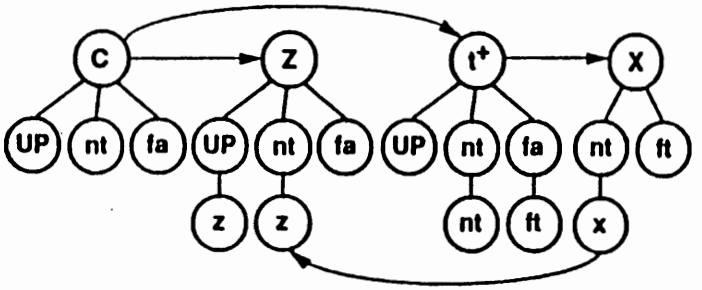


□ = UPDATED STATE
X = PREDICTED STATE
O = CONTACT

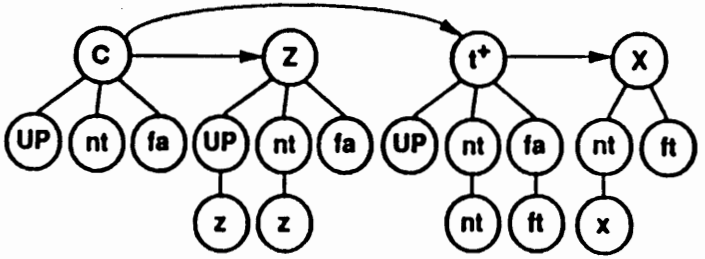
1 CONTACT BEFORE NT/FA PROCESSING



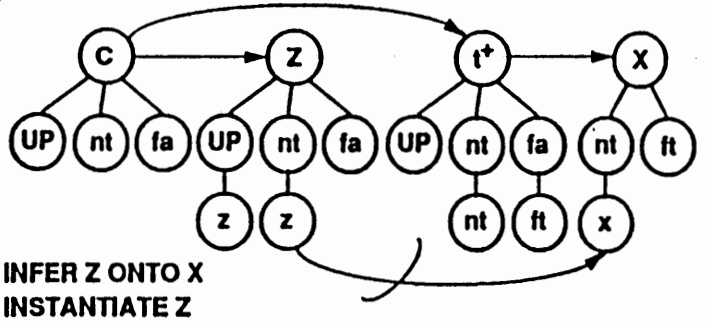
2 ADD NT, FA IF FEASIBLE



4 CONTACT AFTER NT/FA PROCESSING



3 INITIATE TRACK



- INFER Z ONTO X
- INSTANTIATE Z
- ELIMINATE ARC

NOTE: UP = ASSIGNMENT TO A TRACK
NT = NEW TRACK
FA = FALSE ALARM

K9-7246/019

C NODE PROCESSING

This chart shows the Influence Diagram operations involved with processing a C node. In this example, the contact fell into a track gate but fell near the edge of the gate. Therefore, there is a reasonable likelihood that it may be a new track or false alarm.

[1] Before processing the C node, the update assignment was added at the time the t+ node was processed. (See the chart on t+ node processing.)

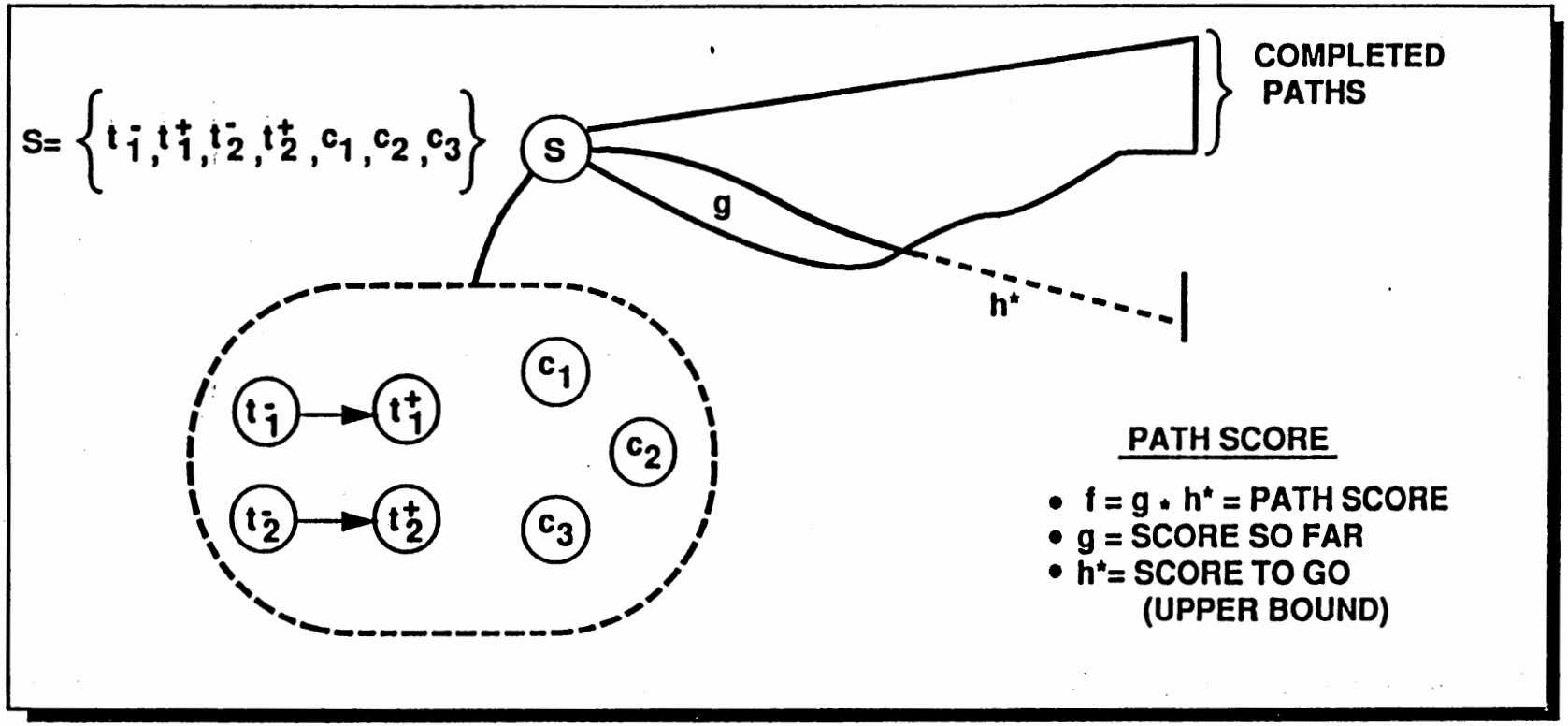
[2] The new track and false alarm outcomes are added and flow to the Z node creating new versions. A Z node state vector is not created for the false alarm version. For the new track version, a t+ node iscreated and an arc from the C node to the t+ node is created. Likewise, an X node is created and an arc from the t+ node to the X is created. Finally, an arc from the new track version for the X node to the new track version for the Z node is created and the data is set.

[3] The arc is reversed; the Z node instantiated and the arc then eliminated.

[4] After C node processing in this example, a new track is created.

S NODE PROCESSING

- S NODE TRAVERSAL USES A* TREE SEARCH TO FIND ALL JOINT OUTCOMES (PATHS) SUCH THAT $f_{path} \geq (1 - \alpha) f_{best}$



NUP NMISS NFA NNT

$g = \pi(\beta_{up_l} PD_l)\ \pi(1 - PD_l)\ (\beta_{FA})\ (\beta_{NT})$

h*= UPPER BOUND ON SCORE FOR REMAINING TRACKS AND CONTACTS

K9-7246/027

S NODE PROCESSING

This chart summarizes the processing associated with the scene node. The scene node, S, is a vector discrete node the represents the joint outcomes of a set of t+, t- and C nodes. The S node processing attempts to generate all feasible joint outcomes by using a search strategy based on the A* tree search algorithm.

The A* algorithm tries to find the best path in a tree by calculating the score so far, g, for a path and an upper bound on score to go, h*. The scores are combined to create a score for each path, f, in the tree. The path with the highest score is then used for continuing the search for the best complete path.

The score so far, g, is a function of the number of updates and their likelihoods, misses, new tracks and new track density, and false alarms and false alarm density on the path so far.

The A* approach is used to find all complete paths, i.e. joint outcomes, that have scores within a certain distance to the best path.

In this example, the S node represents the a sete of seven random variables.

S NODE PROCESSING

SCENE PROCESSING (S)	UPDATE TRACK PROCESSING (t^+)
LOOP 1. SELECT BEST INCOMPLETE PATH ABOVE THRESHOLD 2. IF NULL, EXIT 3. GET NEXT NODE 4. ELABORATE NODE 5. EXTEND PATH END LOOP	1. GET FEASIBLE OUTCOMES 2. PERFORM ACTION FOR EACH OUTCOME: **UPDATE**: • UPDATE STATE • PRUNE ACTION **MISS**: • UPDATE DATA • PRUNE ACTION **FALSE TRACK**: • NO ACTION
	SPAWN TRACK PROCESSING (t^-)
	1. GET FEASIBLE OUTCOMES 2. PERFORM ACTION FOR EACH OUTCOME: **NO SPAWN**: • NO ACTION **SPAWN INTO K**: • SPAWN ACTION **FALSE TRACK**: • NO ACTION
	CONTACT PROCESSING (C)
	1. GET FEASIBLE OUTCOMES 2. PERFORM ACTION FOR EACH OUTCOME: **UPDATE**: • MERGE ACTION **NEW TRACK**: • INITIATE TRACK **FALSE ALARM**: • NO ACTION

K9-7246/026

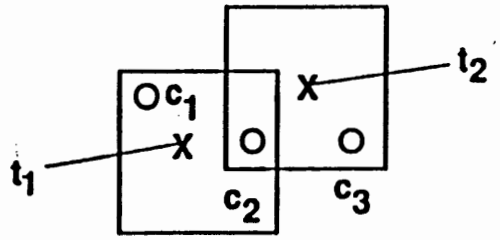
S NODE PROCESSING

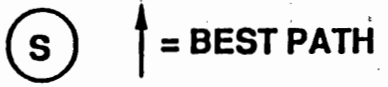
This chart outlines the process flow in elaborating the S node. A search loop is executed which carries out the A* search algorithm. Once a node is selected for elaboration, then the node is elaborated. Three elaboration routines exist: Elaborate t+ node which performs update track processing, elaborate t- node which carries out track spawn processing, and elaborate C node which performs contact processing.

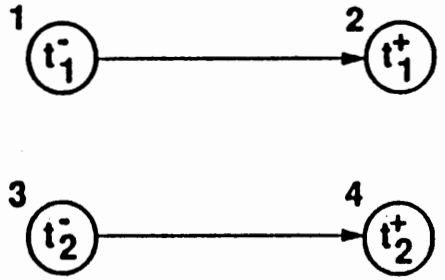
The basic algorithm architecture decomposes into a global search which controls a localized elaboration process.

For each node and for each outcome added to the node an action is performed that is specific to the outcome added.

SCENE PROCESSING EXAMPLE – 0







NOTE: HIGHLIGHTED NODE IS UNDERGOING ELABORATION

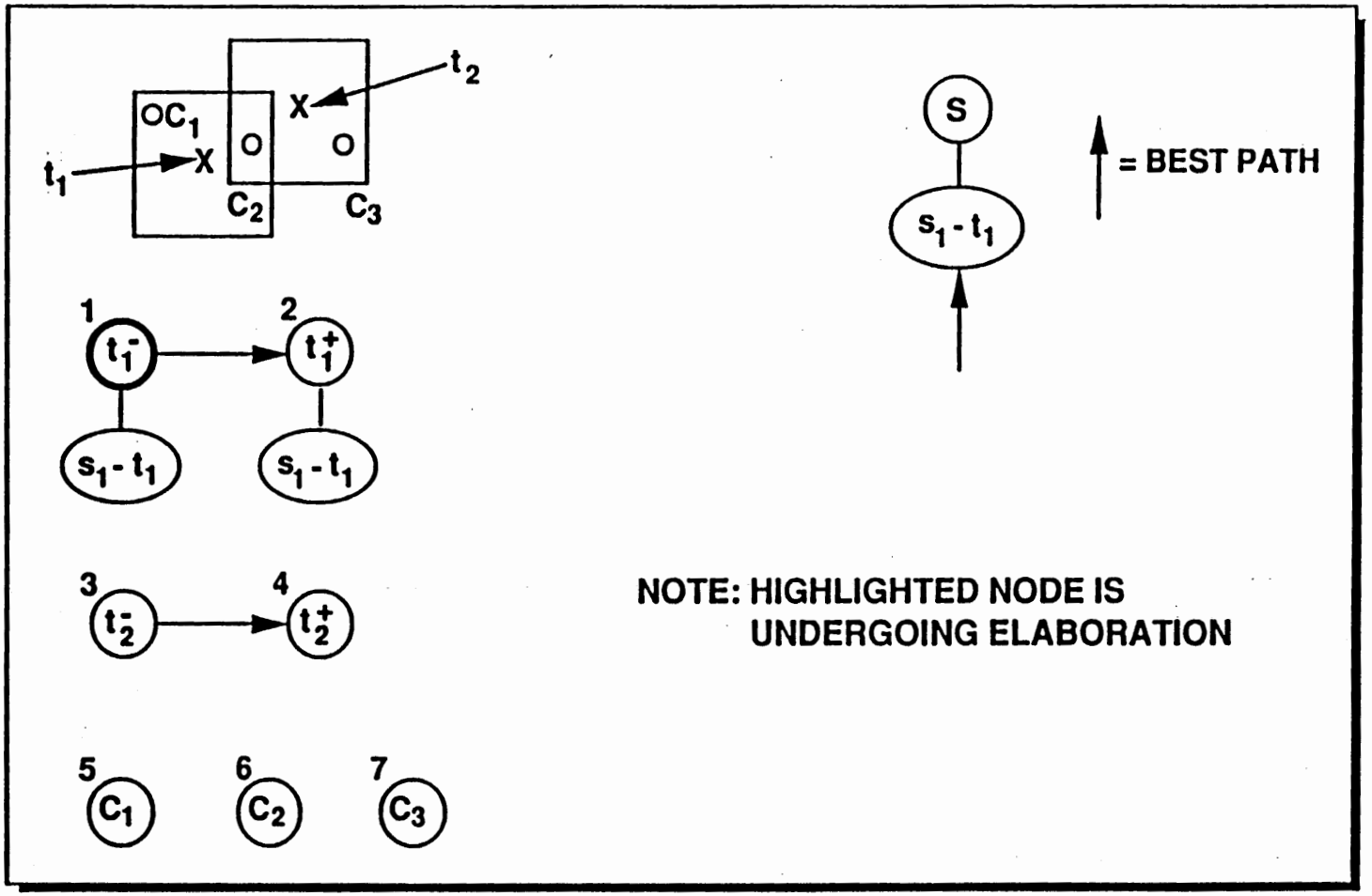
K9-7246/048

SCENE PROCESSING EXAMPLE - 0

In the next 9 charts, the steps in Scene processing are illustrated. The example chosen consists of two tracks with overlapping gates. Three contacts fall in the gates with one contact in the overlap region.

The S node consists of 7 nodes, and the S node and all 7 nodes are shown unelaborated. For the following charts, the best path is illustrated with the up arrow and the node udergoing elaboration is highlighted. The order in which the nodes are visited are numbered.

SCENE PROCESSING EXAMPLE – 1

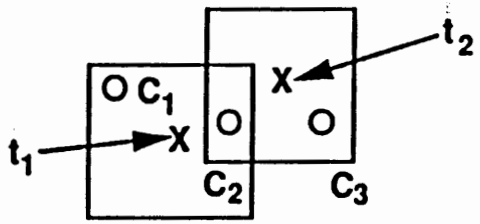


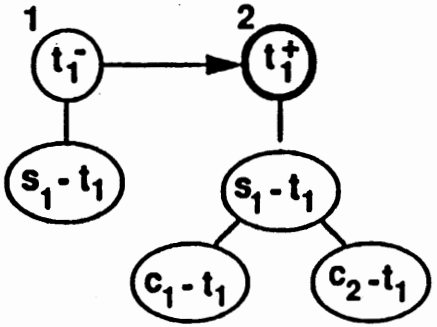
K9-7246/049

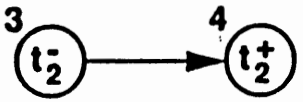
SCENE PROCESSING EXAMPLE - 1 to 9

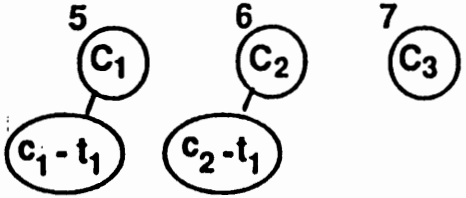
This sequence of charts show the step by step elaboration of the Scene node and the nodes within the scene. At each stage, the best path in the S node joint outcome tree is determined. The next node to be elaborated is selected and the node is elaborated. During the elaboration process, the continuous nodes are managed. When all paths are generated that are within a certain tolerance of the best path are generated the S node elaboration process halts.

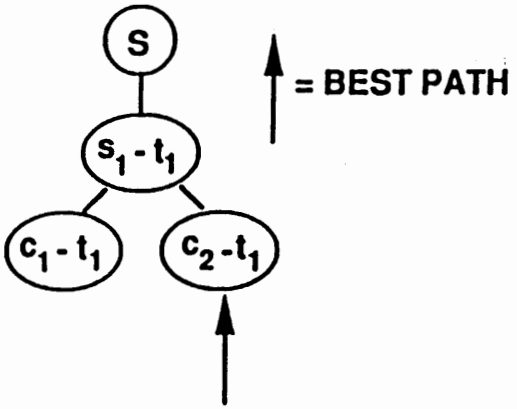
SCENE PROCESSING EXAMPLE – 2







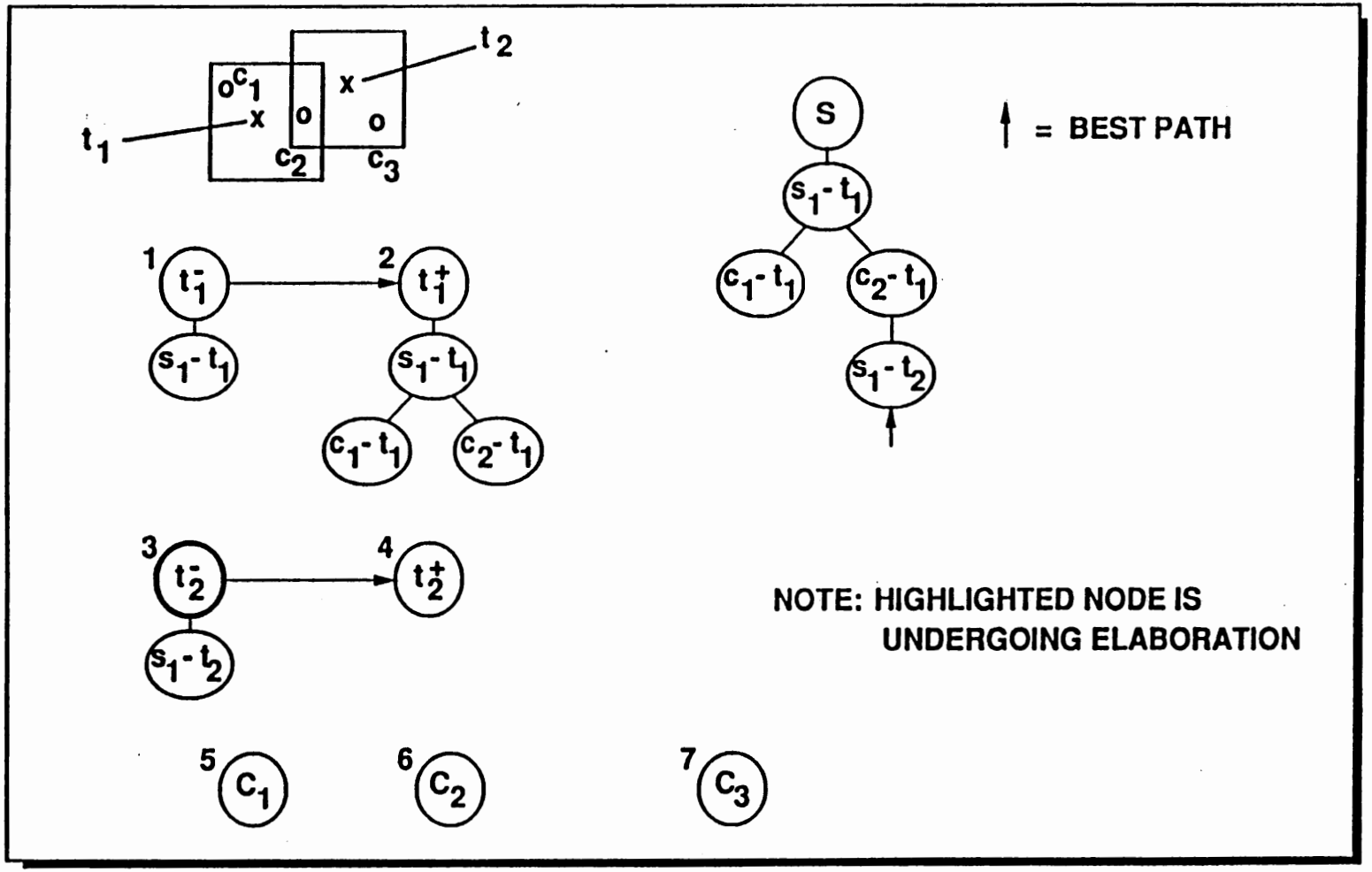




NOTE: HIGHLIGHTED NODE IS UNDERGOING ELABORATION

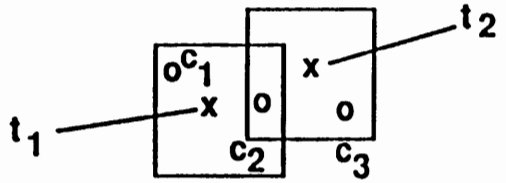
K9-7246/050

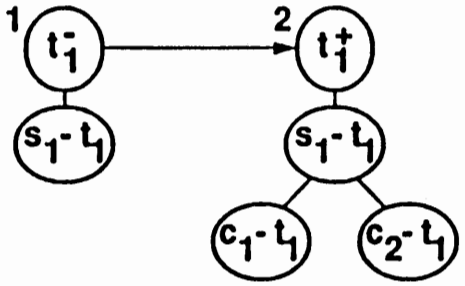
SCENE PROCESSING EXAMPLE – 3

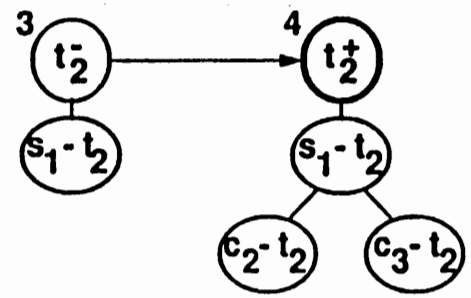


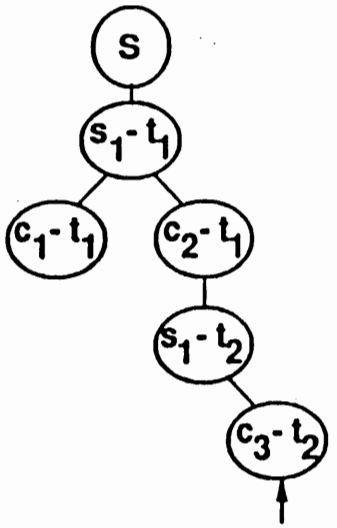
K9-7246/051

SCENE PROCESSING EXAMPLE – 4



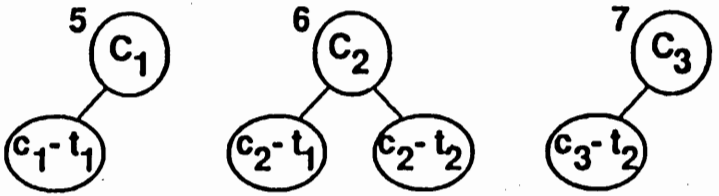






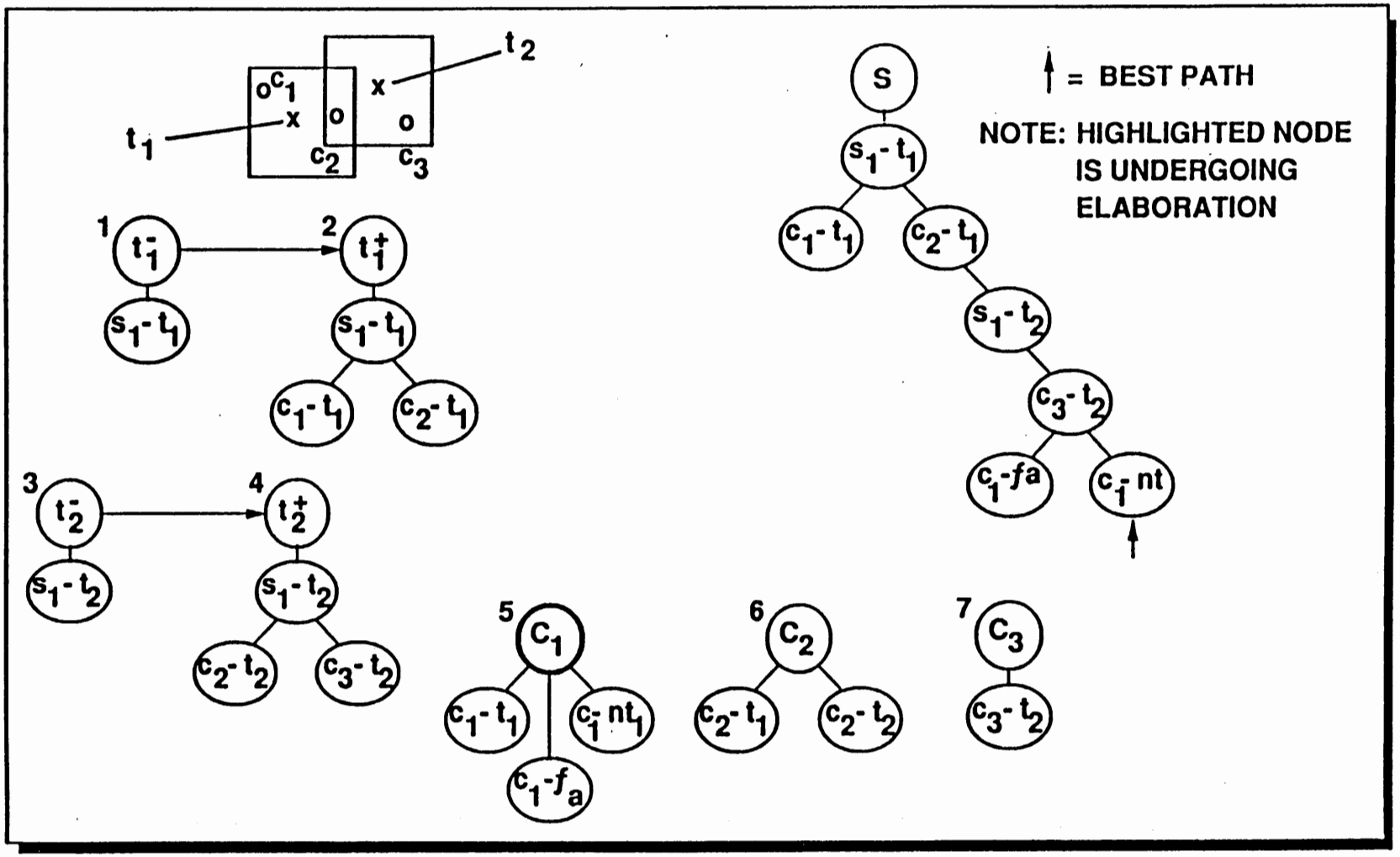
↑ = BEST PATH

NOTE: HIGHLIGHTED NODE IS UNDERGOING ELABORATION



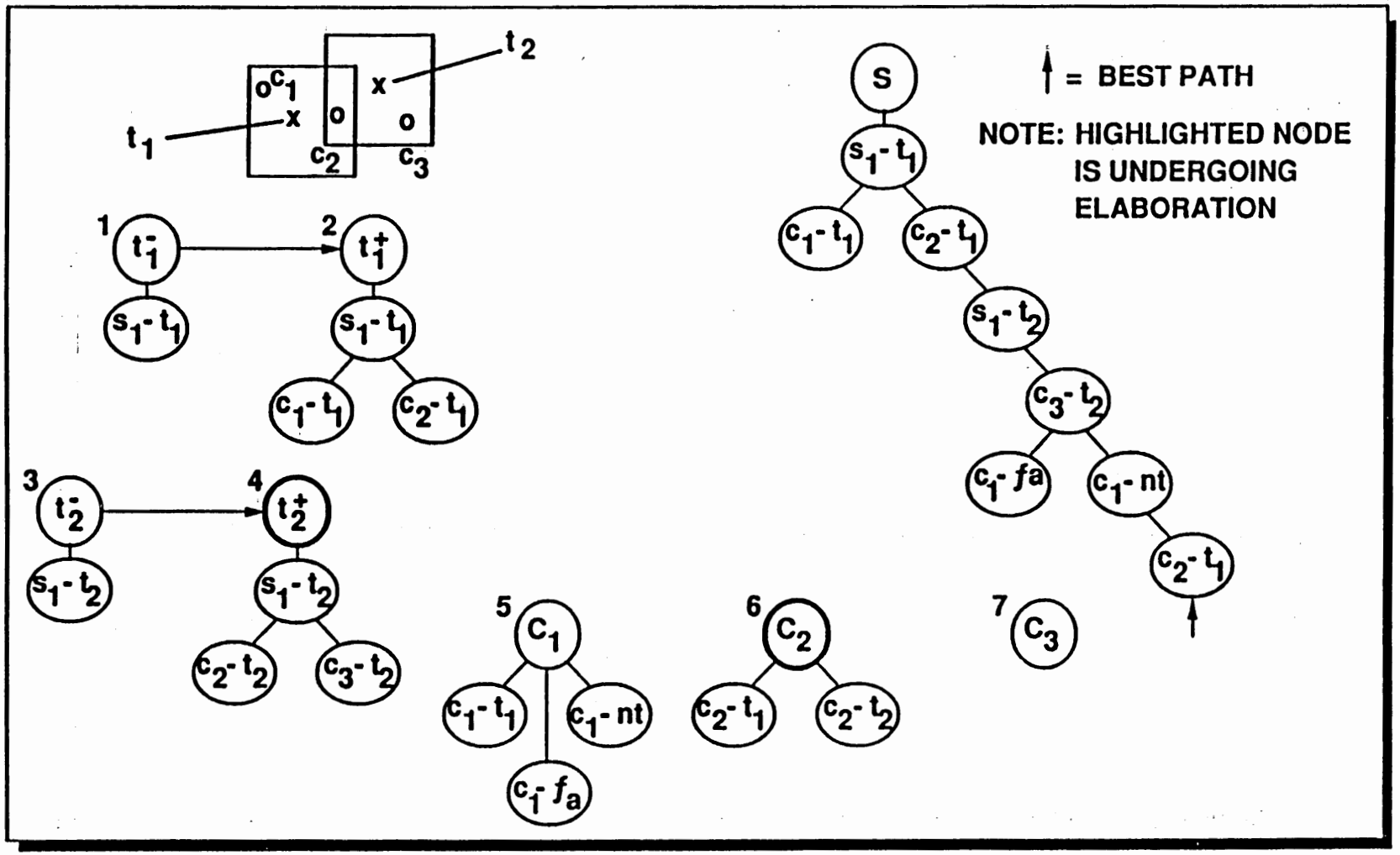
K9-7246/052

SCENE PROCESSING EXAMPLE – 5



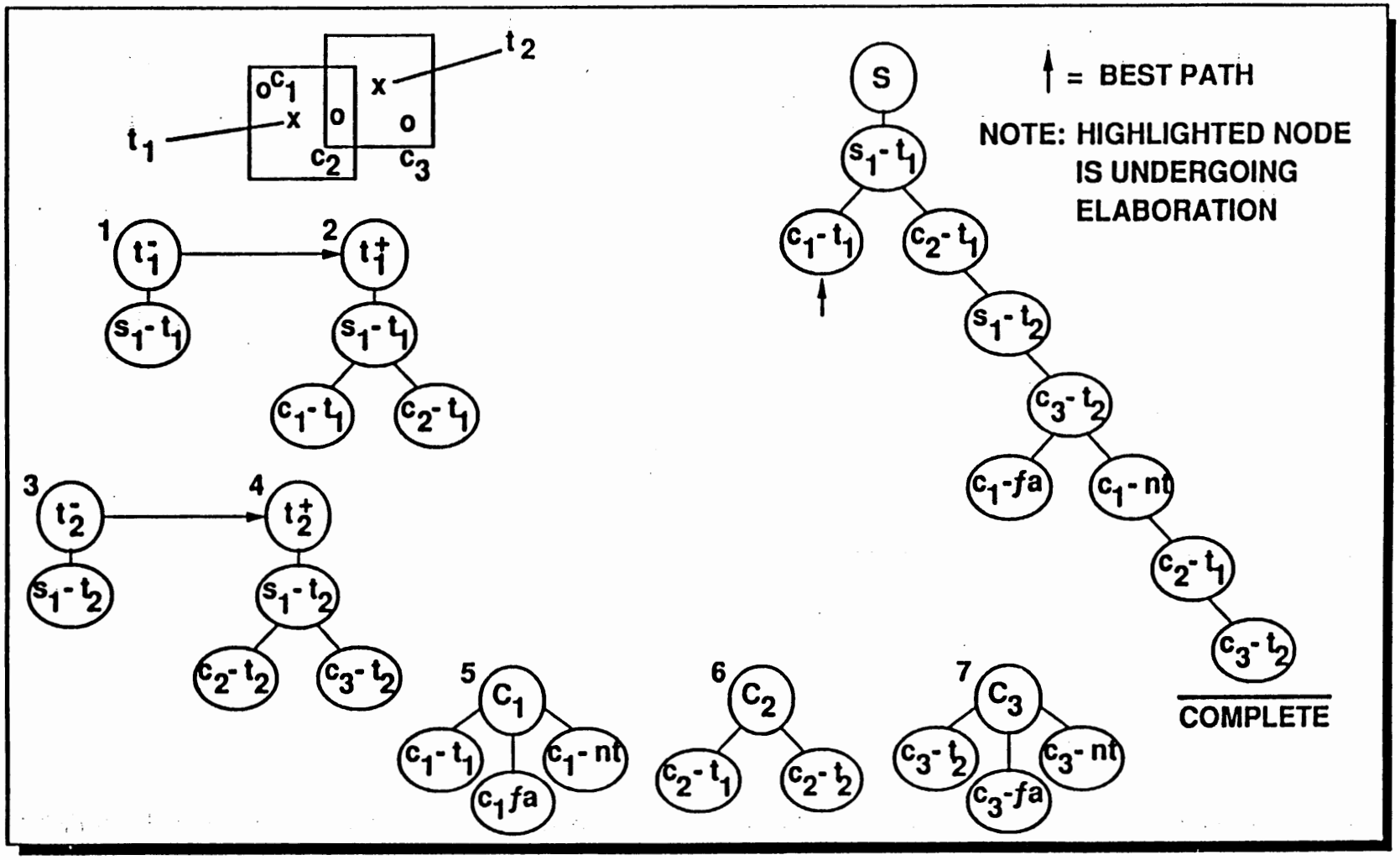
K9-7246/053

SCENE PROCESSING EXAMPLE – 6



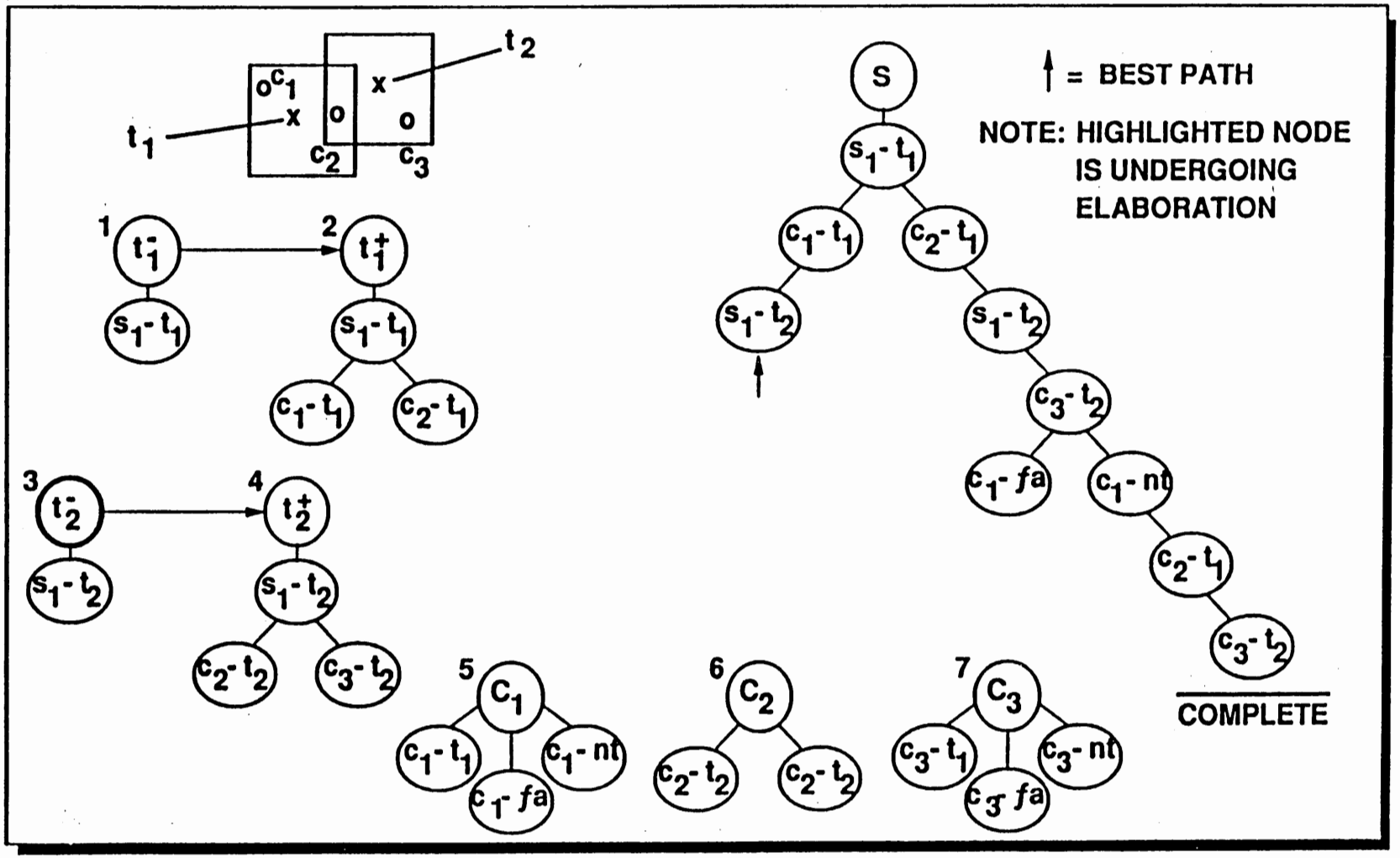
K9-7246/054

SCENE PROCESSING EXAMPLE – 7



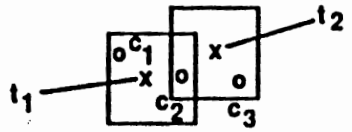
K9-7246/056

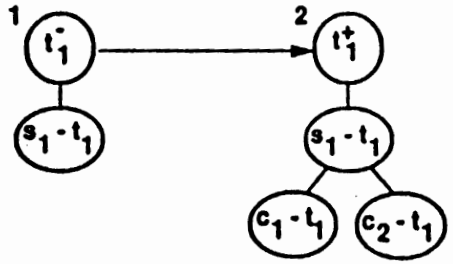
SCENE PROCESSING EXAMPLE – 8

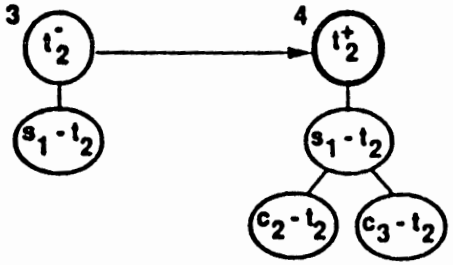


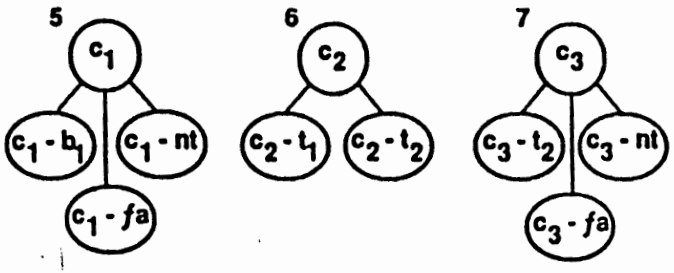
K9-7246/055

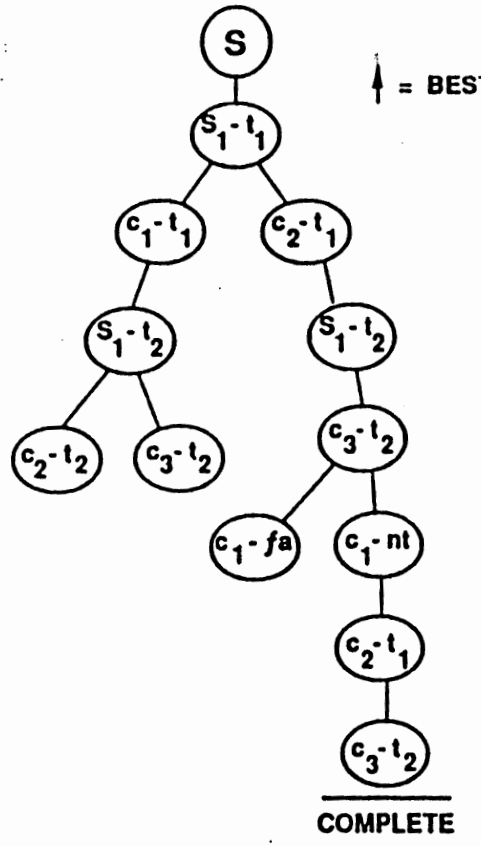
SCENE PROCESSING EXAMPLE – 9







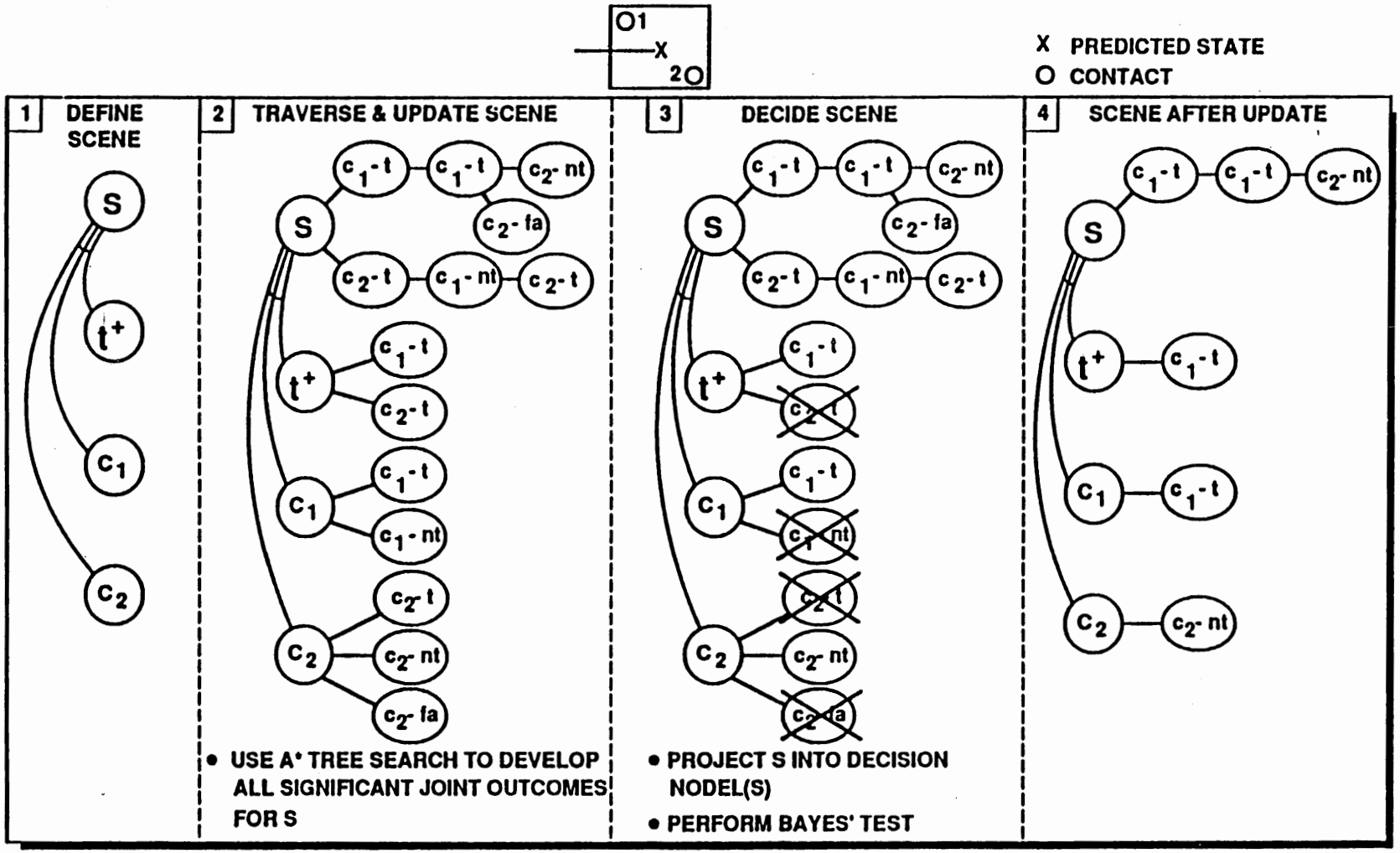




NOTE: HIGHLIGHTED NODE IS UNDERGOING ELABORATION

K9-7246/058

SCENE PROCESSING EXAMPLE



K9-7246/028

SCENE PROCESSING EXAMPLE

This chart shows the complete processing for the Scene node for the example of two contacts in the track gate.

[1] The Scene node is defined as consisting of the update node for the track, t+, and the two contact nodes, C1 and C2.

[2] The traversal algorithm based on A* search is carried out to elaborate the S node and the t+ , C1 and C2 nodes.

[3] n The likelihoods calculated for the joint outcomes in the Scene node are projected onto the t+, C1 and C2 nodes. A Bayes' decision test is performed to select an outcome and prune the other outcomes. Pruning outcomes also removes versions on the continuous side of the Influence Diagram.

[4] After the scene decision process, one outcome remains in this example: C1 updates the track and C2 begins a new track.

SUMMARY

- SHOWED HOW THE INFLUENCE DIAGRAM CAN BE USED TO:
  - REPRESENT PROBABILISTIC INFORMATION GENERATED IN MIDCOURSE TRACKING
  - CARRY OUT:
    - STATE ESTIMATION (UPDATE, PREDICTION, SPAWNING)
    - DATA ASSOCIATION (HYPOTHESIS GENERATION, SCORING AND SELECTION)
    - TRACK PROMOTION
- CURRENTLY, TRACKER IS IMPLEMENTED USING INFLUENCE DIAGRAM UTILITIES, AND IS UNDERGOING EVALUATION

SUMMARY

This presentation summarized the charcteristics of the Influence Diagram and its applicability to implementing midcourse tracking algorithms. The collection of Influence Diagram utilities perform the probabilistic calculations associated with Kalman filter processing, track spawning, shared contact update and formation update processing, as well as association hypothesis scoring.

Other advantages of the Influence Diagram implementation are the following:

** The Influence Diagram implementation automatically maintains all relevent influences as part of the inferencing process.

** The Kalman Filter implementation is efficient and guarantees a positive semidefinite covariance matrix.

** The Influence Digram provides a useful means to organize and manage the track database.