TECHNICAL REPORT 1974
November 2008

Knowledge Management for Distributed Tracking: The Final Report

Marion G. Ceruti
Tedd L. Wright
David J. Swanson
Scott C. McGirr
Dwight R. Wilcox

SSC Pacific

TECHNICAL REPORT 1974
November 2008

Knowledge Management for Distributed Tracking: The Final Report

Marion G. Ceruti
Tedd L. Wright
David J. Swanson
Scott C. McGirr
Dwight R. Wilcox

SSC Pacific
San Diego, CA 92152-5001

SSC PACIFIC
San Diego, California 92152-5001

M. T. Kohlheim, CAPT, USN
Commanding Officer

C. A. Keeney
Technical Director

ADMINISTRATIVE INFORMATION

This report was prepared for Office of Naval Research (Code 31) and the SPAWAR Systems Center Pacific (SSC Pacific) Science and Technology Initiative by the Command & Control Technology & Experimentation Division (Code 536).

Released under authority of
T. Tiernan, Head
Command & Control Technology
& Experimentation Division

SB

EXECUTIVE SUMMARY

Modeling and simulation, intelligent software agents, and other technologies can support network-centric distributed tracking. These technologies came together in the Knowledge Management for Distributed Tracking (KMDT) research and development program to improve naval command, control, and decision support. The program's approach is based on the use of simulated data from sensor and motion models, intelligent software agents, integrated sensor ontology, and a line-of-bearing cross-fix algorithm.

Modeling and simulation was used to generate test data that intelligent agents ingested to search for information that could help localize and characterize unknown contacts in the simulated battle space. In these simulations using a hypothetical scenario, intelligent-software agents were deployed over a simulated, secure Web-like network to find additional, possibly disparate, sensor data from other friendly platforms on unknown contacts.

The concept of operations calls for the agents to interact with the integrated sensor ontology to facilitate distributed, heterogeneous sensor-data fusion and to reduce uncertainty. To illustrate these concepts, a simulation is described that generated 400 contact reports consisting of passive lines of bearing that were fused subsequently by intelligent agents to localize the contacts, thus demonstrating the potential of this approach to reduce information overload for the operator. The KMDT approach demonstrates how knowledge management technologies can be employed to improve situation awareness and reduce operator workload.

CONTENTS

Figures

1. INTRODUCTION

The Knowledge Management of Distributed Tracking (KMDT) program supports the concept of network-centric warfare as envisioned in FORCEnet (Clark, 2002; McGirr, 2001; Ceruti et al., 2004), which is the U.S. Navy's operational construct and architectural framework for naval warfare in the information age (Clark, 2002). To accomplish this vision, a distributed tracking problem was simulated for subsequent demonstration and analysis that primarily involves the localization of targets via intersection of lines-of-bearing (LOBs) from distributed passive sensors (Ceruti et al., 2005). Although highly simplified, this problem may reflect current political and operational strategies that restrict usage of active systems for environmental and security reasons. The simplification is intended to focus the demonstration on the advantages of a network-centric concept of operations over those involving legacy stand-alone architectures and systems.

KMDT focuses on technologies that are essential for the design of next-generation tracking systems that use knowledge management techniques and network-based command and control to reduce uncertainty in command decisions. (See, for example, Ceruti and Wright, 2005.) These technologies include distributed sensing, modeling and simulation (M&S) (Ceruti et al., 2005), intelligent agents (Ceruti, 2001; Ceruti & Powers, 2004, Deazeau & Muller, 1989), and ontology (Ceruti, 2004; Ceruti & Wilcox, 2006; Matheus et al. 2005; Swanson et al., 2007). Distributed tracking can revolutionize traditional level-one data fusion (e.g., detection, localization, tracking, classification, and identification) (McGirr, 2001; Ceruti & Wilcox, 2006; Waltz & Llinas, 1990). Today, the Navy does not use the majority of sensor data due to lack of correlation opportunities. Through knowledge management as modeled in the KMDT simulation, some of these unused sensor data that are now unavailable can be quickly fused to localize, track, and classify targets of interest (TOI), which will result in a future reduction in uncertainty in command centers.

The distributed KMDT architecture enables any operator connected to the network to deploy intelligent agents to retrieve sensor data posted to common repositories at ship and shore-based sensor-data collection stations. The network does not contain a single fusion center where all fusion processing takes place, but rather, each ship or shore-based sensor station can perform distributed sensor-data fusion as needed to meet their local requirements. The agents retrieve processed sensor data at the message level and are not connected directly to the remote sensors themselves.

KMDT's M&S approach uses a hypothetical scenario to develop and evaluate knowledge-management techniques (Ceruti et al., 2005). M&S helps to conceptualize the salient features of the battle space and to focus on characteristics of interest, such as how ships move in the water and how distributed sensors detect them in a controlled environment that is not available in the physical world. It also allows the testing of many hypothetical scenarios at low cost. When the concepts are demonstrated using simulated experiments, these results serve collectively as a point of departure for more realistic field tests.

Unlike legacy systems that rely primarily on similar sensor types to detect, localize, and track TOI, new approaches also can use dissimilar sensor types for level-one data-fusion tasks. Our previous research was oriented towards the simulation details (Ceruti et al., 2005) and the integrated sensor ontology (Ceruti, 2004; Ceruti & Wilcox, 2006; Matheus et al., 2005) as separate components. However, the current work also includes agent design, the agent-ontology interaction, and a database to instantiate some of the concepts in the integrated sensor ontology. The initial KMDT integrated sensor ontology was based on ontology components developed in previous projects, as explained in Ceruti (2004), Ceruti & Wilcox (2006), and Matheus et al. (2005).

2. MODELING AND SIMULATION

This section describes the M&S component of the KMDT program. The purpose of the KMDT program is to demonstrate the use of new information technologies such as sensor ontologies and software agents in a secure, distributed network for level-one data fusion (i.e., localization, tracking, identification, and classification of targets) in the maritime battlespace. It is anticipated that by using these information technologies, additional information normally not available at a command center can be quickly fused to help localize, track, and classify TOI. To accomplish this task, a distributed tracking problem was simulated that primarily involves the localization of targets via intersection of LOB from distributed passive sensors. This problem, although highly simplified, demonstrates the advantages of exploiting a net-centric concept of operations over those involving legacy "stovepipe" architectures and systems. The KMDT demonstration concept using this simulation is described in Appendix F.

2.1 MODELING AND SIMULATION DESIGN OVERVIEW

The KMDT simulation represents realistic trial scenarios for testing technologies and concepts without conducting costly field experiments. Additionally, it provides an analysis tool to quantify results. Figure 1 shows a functional diagram of the simulation. The simulation contains two primary components: a scenario specification and a contact generator. The scenario consists of predefined fixed or mobile sensor systems in some theater of operations, with specified performance capabilities. Additionally, the scenario contains multiple targets with specified signal emission signatures, start times, and initial locations. As the simulation evolves in time, the positions of the targets and mobile sensor platforms are updated at discrete time steps using appropriate motion models. Next, the range and LOB from each sensor to each target is computed. A detection model then determines the probability of detection computed as a function of this range. Detection alerts are subsequently declared based upon the probability of detection. False alarms are also generated according to acceptable false alarm rates for each sensor. When the simulation declares a detection or false alarm, the LOB and other information generated by the simulation are posted to detection databases (Swanson et al., 2007).

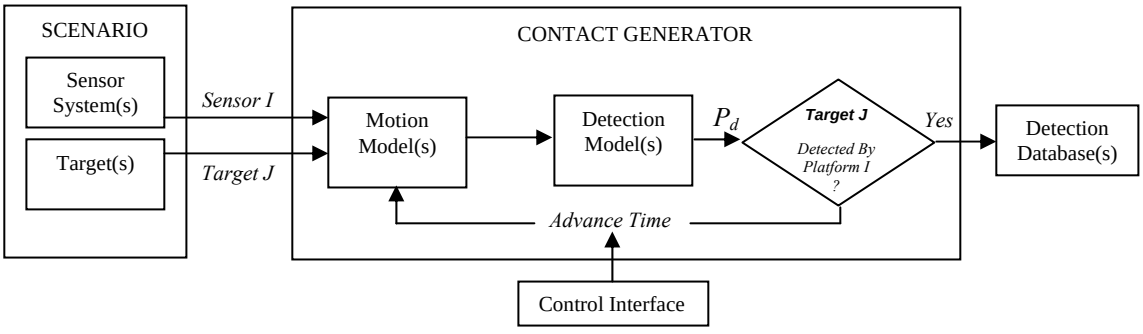


Figure 1. KMDT simulation functional diagram (Ceruti et al., 2005).

2.2 THEATER OF OPERATIONS

The theater of operations chosen for the simulation is the East China Sea and vicinity (Figure 2). Not only is this an area of interest for maritime operations, but also it provides a variety of conditions to test robustly the technologies being developed under KMDT. Both shallow-water littoral and deep-water environments are represented, along with sea access via straits and open ocean. Proximity of both mainland and islands about the perimeter of the East China Sea provide for topographical

constraints as well as locations for land-based sensor sites, air and sea bases, and ports of opportunity for maritime vessels and aircraft (Swanson et al., 2007).

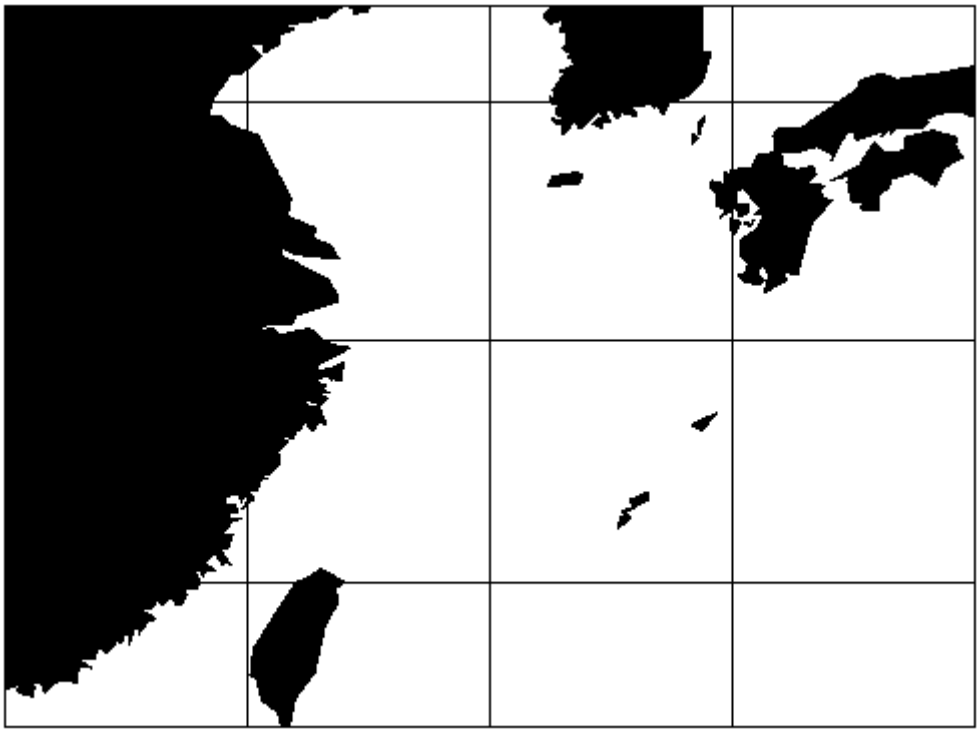


Figure 2. Scenario theater of operations (Swanson et al., 2007).

2.3 SCENARIO

The scenario defines the targets and sensor systems for a simulation. Different scenarios may be designed to provide testing for various types and locations of targets and sensor systems in different maritime environments. Sensor and target attributes and initial conditions are specified via scenario files. Scenario file formats are documented in Appendix B.

2.3.1 Sensor Systems

The simulation can represent both fixed and mobile sensor systems (Swanson et al., 2007). Fixed sensor systems include deep-water arrays, barrier arrays, and land-based systems; mobile sensor platforms are primarily restricted to surface vessels.[1] Mobile sensor platforms utilize a "waypoint" motion model described in Section 4.1.2. The simulation currently models passive acoustic sensor systems (e.g., the SOund SUrveillance System [SOSUS] and the Surveillance Towed Array Surface System [SURTASS]) only.[2] Sensor system and platform attributes/initial conditions

[1] Version 6 of the simulation utilizes a two-dimensional field for target and sensor platform motions, and thus is primarily suited for surface vessels. However, it may be used for submarines and aircraft when the fidelity of the simulation does not require precise knowledge of the depth/altitude of the vessel. For example, with simple transmission loss models that do not require knowledge of the depth of the target or sensor platform, the simulation may be used for acoustic detection of submarines as well as surface vessels. On the other hand, it may not be suitable for airborne sensors whose field of view depends on the altitude of the aircraft. Extensions to three-dimensional fields are planned for future implementations.

[2] Electro-magnetic (e.g., electronic support measures), and electro-optic (e.g., infrared and video) sensor systems, as well as active sensor systems, are planned for future implementations. Note that it is possible to emulate non-acoustic sensors by adjusting source levels, array gains, noise-level specifications, etc., to give appropriate responses for such sensors, as the probability of detection computation is independent of a given source type.

specified via the scenario file include:

- Probability of false alarm deemed acceptable for sensor system
- Initial platform location latitude and longitude
- Shipping noise level offset from moderate traffic density (at initial location)
- Sea-state (at initial location)
- Array gain of sensor system
- Standard deviation for LOB error distribution
- Maximum detection range for sensor system (azimuth-dependent)
- Waypoint motion model track leg destination latitude/longitude and speed

2.3.2 Targets

The simulation focuses on surface targets, although submarine and air targets may be simulated with restrictions.[3] Targets are generally classified as friendly (F), hostile (H), neutral (N), or unknown (U), and contain target-specific signal emission signatures that can be used for more refined identification. The signal signatures provide the source levels for the various targets; in the case of acoustic detection, these signatures consist of sound power levels at discrete frequencies emitted by the target vessel. Targets utilize a "quasi-random" motion model described in Section 4.1.1. This motion model is designed to give realistic but non-deterministic behavior to targets, thus generating different detection statistics for separate invocations of the same scenario. Target attributes/initial conditions specified via the scenario file include:

- Target start time
- Initial target location latitude and longitude
- Initial target speed
- Standard deviation for speed distribution
- Maximum target speed
- Initial target heading
- Standard deviation for heading distribution
- Target classification (Friendly, Hostile, Neutral, Unknown)
- Signal emission signature (frequency and source level of discrete signal components)

2.4 CONTACT GENERATOR

A contact is a detection of a target by a sensor based on a computed probability of detection (P_d). The contact consists primarily of an LOB from the sensor to the target. These computations are performed in the simulation by a Contact Generator. The Contact Generator consists of time-stepped motion models for updating target and mobile sensor platform positions, geodetic (great circle) range, and LOB computations from each sensor to each target, and detection models that compute the signal-to-noise ratio (SNR) at a sensor location given the source level of the target, transmission loss between target and sensor, noise level at the sensor location, and processing gain of the sensor system. Based on the computed SNR at the sensor, the P_d is computed and a detection decision is made. At each time-step, a false alarm may also be generated for each sensor. Any detections or false alarms are then recorded in appropriate detection log databases for subsequent access and fusion by KMDT software agents.

[3] See footnote 1.

2.4.1 Motion Models

Two motion models are provided in the simulation, a "quasi-random" model to represent realistic tracks of unknown targets, and a deterministic "waypoint" model to represent vessels in transit lanes or on patrol.

2.4.1.1 Quasi-random Motion Model

The quasi-random model provides for realistic, non-deterministic target tracks in order to generate detection statistics from multiple invocations of a given scenario that will accommodate Monte Carlo-type analyses. At each time-step of the simulation the model projects new target positions using speeds and headings that are randomly selected from normal distributions whose means are respectively given by the previous time-step speed and heading, and whose standard deviations are given as target attributes in the scenario specification.

Specification of smaller (respectively larger) standard deviations result in smaller (respectively larger) speed and heading changes and consequently more (respectively less) uniform motion. The model also employs a land-avoidance algorithm for waterborne vessels that adjusts the new target position if the randomly selected speed and heading determine a location over land. For such a location, the adjustment is accomplished by alternately varying the heading clockwise and counter-clockwise in increasing increments until a new location is determined that is over the sea.

The determination of whether a location is over land or sea is made by specifying a closed polygon that encloses the sea area. This polygon consists of latitude–longitude vertices, later converted to rectangular coordinates, that define the coastline; islands are excluded from the interior of the polygon by treating them as enclaves, that is, by connecting their boundaries to that of the exterior polygon by a "doubly traversed" line.

Thus, by traversing the exterior polygon in the clockwise sense, islands will be traversed in the counter-clockwise sense, and the sea will always be on the right-hand side of the polygon boundary. Then the number of times the polygon boundary intersects the line parallel to the y-axis (i.e., the north–south axis) extending below (i.e., to the south of) the location in question will be odd (respectively even) if the location is inside (respectively outside) the closed polygon.

2.4.1.2 Waypoint Motion Model

The waypoint motion model provides deterministic tracks to be used by targets confined to transit lanes or sensor platforms on patrol (e.g., racetrack or picket line patrols).[4] Deterministic tracks for a given vessel are defined by track segments or legs specified in the scenario. A track leg consists of a destination location (latitude and longitude) and a leg speed; the departure location is assumed to be the destination location of the preceding leg.

During each time-step of the simulation, the current track-leg speed and heading, as determined from the rhumb (constant heading) line between the departure and destination location, are used to compute a new platform position; if the range to this position exceeds that to the destination location of the current leg, the current course is followed to the waypoint location, the time of arrival at the waypoint is computed, and the new course is followed for the remaining time in the time-step.

[4] In Version 6 of the simulation, the waypoint motion model is used only by sensor platforms, while the quasi-random model is used only by targets.

2.4.2 Detection Models

Targets are detected by sensors via computation of a range-dependent probability of detection, P_d. At each time-step of the simulation, the geodetic (great circle) range and LOB from each sensor to each target in the field is computed. For those targets that are closer to the sensor than the maximum detection range for that sensor along the computed LOB azimuth (as specified in the scenario), the SNR is computed. For passive systems, the SNR is generally a function of the source level of the target, the transmission loss between the target and sensor, the noise level at the sensor, and any processing gain of the sensor system.

Since these SNR components are generally functions of the frequency, a separate computation must be made for each frequency component of the target signal signature. Since the transmission loss is a function of the range from the target to sensor, the SNR is also range-dependent. Subsequently, the P_d is computed from the SNR.[5] A detection alert is declared when a random number selected from a uniform distribution between 0 and 1 is less than the computed P_d.

Note that only some of the frequency components of a target signal signature may be detected at any given time. At each time-step, a false alarm is also generated for each sensor when a random number selected from a uniform distribution between 0 and 1 is less than the sensor's specified probability of false alarm, P_{fa}. Detection alerts and false alarms are reported to individual detection logs for each sensor, as well as a composite log for all sensors.

2.4.2.1 Passive Acoustic Sensors

For passive acoustic sensors, the SNR is modeled by the passive "SONAR equation." In units of decibels, the SONAR equation is

$$DT = SL - TL + AG - AN,$$

where

DT = *Detection Threshold*, the SNR necessary to achieve a desired level of performance (DT is also called the *Recognition Differential*, RD);

SL = *Source Level*, the ratio of the acoustic intensity (proportional to the squared pressure) of the source to that of a plane wave of rms pressure $p_0 = 1$ μPa at a reference distance $r_0 = 1$ m from the source;

TL = *Transmission Loss*, the ratio of the acoustic intensity at a distance $r_0 = 1$ m from the source to that at a distance r from the source;

AG = *Array Gain*, the ratio of the SNR of an array of coherently summed receiver elements to that of a single element;

AN = *Ambient Noise*, the spectrum level (power) of all incoherent sources of noise in a 1 Hz band at the receiver.

Thus, for a given SL, TL, AG, and AN, the passive SONAR equation may be evaluated for the DT that represents the SNR. The P_d is computed as a function of the DT and a specified P_{fa} (see Appendix E). Note that there are various representations for these terms with different levels of fidelity. For example, in a homogeneous acoustic environment, TL may be modeled by simple spreading loss and absorption. However, boundaries and spatial and temporal variability of the sound field can cause refraction and other effects requiring sophisticated computer models and environmental databases for evaluation. For the purposes of the KMDT simulation, only low-fidelity models are generally required. The TL and AN models incorporated in the simulation are described in the

[5] Detection theory, including the computation of the P_d, is reviewed in Appendix E.

following subsections; SL for the targets and AG for the sensors are specified via the scenario specifications.

2.4.2.1.1 Transmission Loss Model

In homogeneous seawater, the TL results from geometric spreading and absorption of acoustic energy, and may be modeled by

$$\begin{aligned} \mathrm{TL} &= 20 \log r + \alpha r, \;\; r < r_t; \\ &= 10 \log r_t + 10 \log r + \alpha r, \;\; r > r_t. \end{aligned}$$

This represents a combination of spherical spreading near the source, and cylindrical spreading away from the source due to the confinement of the propagation to a horizontal layer between the atmosphere and ocean basin. Here, r is the horizontal distance from the source to the receiver, r_t is the horizontal distance from the source to the transition from spherical to cylindrical spreading, and α is an absorption coefficient.

The transition range r_t is computed as follows: Suppose a homogeneous ocean (i.e., where the sound speed $c = c_1$ is constant) is confined to a layer of depth h as shown in Figure 3. For a point source, waves will propagate outward in spherical wavefronts (i.e., surfaces of constant phase), whose rays (i.e., directions of wave propagation perpendicular to the wavefronts) are straight. When the waves reach the ocean boundaries, they will reflect from and/or transmit through the boundaries. Due to the large acoustic impedance difference between the seawater and air, waves will be almost totally reflected at the surface for any grazing angle ϑ between a ray and the horizontal boundary.

At the ocean bottom, however, the acoustic impedance difference is usually less, so that the wave is both reflected and transmitted. For the reflected wave, the angle of reflection equals the angle of incidence ϑ_1; however, for the transmitted wave, the angle of transmission ϑ_2 is related to the angle of incidence ϑ_1 by the ratio of the sound speeds in the adjacent layers. This relationship is expressed by Snell's Law,

$$\frac{\cos \vartheta_1}{c_1} = \frac{\cos \vartheta_2}{c_2},$$

where c_2 is the sound speed in the bottom substrate. If $c_2 = k\, c_1 > c_1$, as is typically the case in ocean environments, then there will be an incident angle ϑ_1 where the angle of transmission $\vartheta_2 = 0$, that is, $\vartheta_1 = \cos^{-1}(c_1/\, c_2) = \cos^{-1}(1/\, k)$. This angle is called the critical angle ϑ_c. For incident angles $\vartheta_1 < \vartheta_c$, there is no transmitted wave, and propagation is trapped in the layer. Now, for a source located at depth $z = z_0$, $\tan\, \vartheta_c = (h - z_0)/r_c$, where r_c the horizontal range to where the ray with critical angle ϑ_c reaches the bottom. Let this range be the transition range r_t between spherical and cylindrical spreading.

Then,

$$r_t = \frac{h - z_0}{\tan\left[\cos^{-1}(1/k)\right]}.$$

Typically, $k \approx 5$ for sound speeds in seawater and bottom substrates consisting of elastic solids.

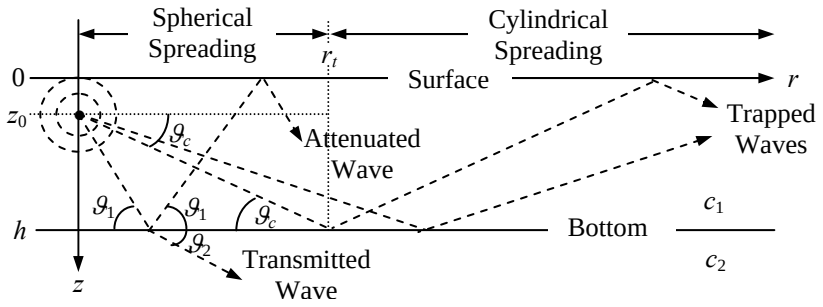


Figure 3. Transmission loss model geometry.

The absorption coefficient α is given by the following expression (Thorp, 1967):

$$\alpha = 0.0010936133\left(\frac{0.1f^2}{1+f^2} + \frac{40f^2}{4100+f^2} + 0.000275f^2\right) \text{ dB/m},$$

where f is the frequency in kHz, which represents the transmission loss per unit length due to viscous and chemical relaxation processes.

2.4.2.1.2 Ambient Noise Model

Acoustic AN is defined as the power level at the receiver of all incoherent sources of noise in a 1-Hz band about a specified frequency; that is, as 10 $\log_{10}(N)$, where N is the noise spectrum level relative to a 1-Hz frequency band. The AN as a function of frequency is modeled by fitting curves to typical deep-water noise spectra. The noise is caused primarily by ocean turbulence, shipping, wind/sea state, and thermal agitation. The family of spectra for different shipping densities and wind/sea states are shown in Figure 4. Shipping offset levels and wind/sea states for individual sensor locations are specified via the scenario file (see Appendix B).

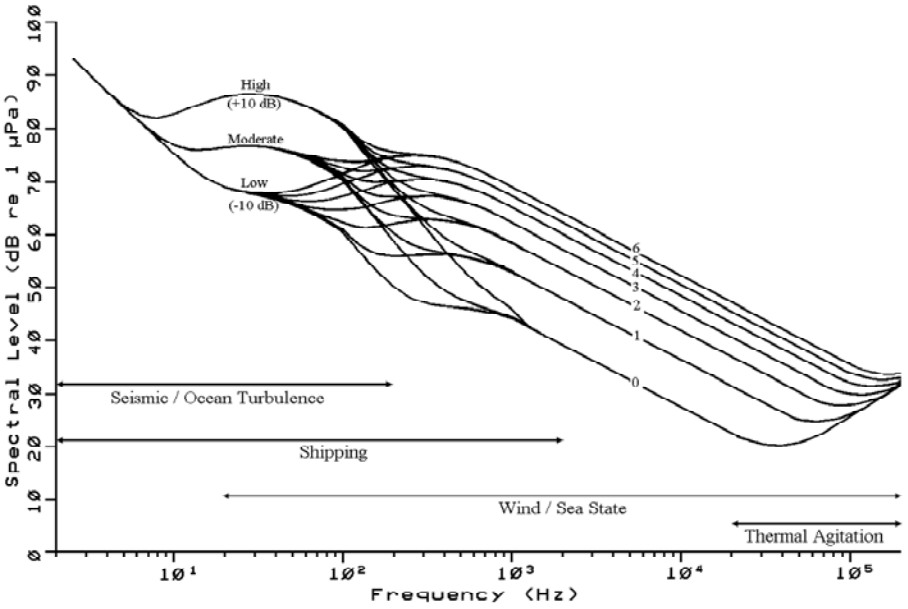


Figure 4. AN level vs. frequency.

2.5 DETECTION DATABASES

The detection databases represent repositories (logs) for all target detections and false alarms generated by the simulation. In the KMDT concept of operations, these databases represent Web services supplied by distributed sensor systems connected to a network that can be searched by agents to discover information for subsequent fusion. The databases consist of individual ASCII files for each sensor, and a composite file containing detections and false alarms from all sensors in the scenario. Each file record consists of a single detection alert (that may be a false alarm), with the following parameters:

- Sensor/platform identification
- Time of the alert/false alarm, including any signal propagation time from the target to the sensor
- Sensor location latitude and longitude
- Approximate LOB from the sensor to the target
- Frequencies of detected signal components
- Maximum detection range along the reported LOB for the sensor (as specified in the scenario)

In addition to the above parameters, the following information extracted from the simulation is appended to the detection alert record for validation purposes. This information, of course, would not be available to an operational system.

- The ground truth location latitude and longitude of the target
- The classification of the target (friendly, hostile, neutral, unknown)

Formats for the detection log records are documented in Appendix C.

3. INTELLIGENT SOFTWARE AGENTS

Current intelligence, surveillance, reconnaissance (ISR) capabilities of naval strike groups have a shortfall in sensor tasking and data exploitation. Tactical commanders must be able to recognize blind spots in their tactical picture so they can obtain specific information to reduce their uncertainty. The Navy does not have the capacity to use all the data it collects from its own sensors. Moreover, tactical commanders and their staffs lack the resources and the tools to examine and consider using all the available information not only from naval sensors, but also from national ISR systems. ISR systems provide large quantities of information from a wide range of national, theater, and organic sensors. However, the Navy does not own all the sensors it wants to access. Various other organizations manage many sensors that the Navy needs.

The insights and potential advantages of this information often are not used at the tactical level because of the difficulties for tactical commanders to task sensors, analyze the information, evaluate its relevance, and then exploit it in time to be effective. Today, this event cycle, including the time required for sensors to respond to a commander's tasking, is usually too long for the results to be used tactically. The information would be overtaken by events by the time it arrived. Even if the information arrived in time, few tools are available to help exploit it.

Naval strike groups are spread more widely over the globe and now rely more frequently on reach back to help cope with the flood of information available from ISR systems. The functionality that the KMDT agents can provide is a natural starting point from which to utilize the new capabilities and better facilitate reach-back.

3.1. AGENT DESIGN AND USE

This section describes the design and use of the intelligent-software agents. All KMDT agents are designed as threaded programs. KMDT agents are constructed from a set of top-level Java® classes. The design is agent-centric in the sense that capabilities are composed of loosely coupled agents that are adapted to a specific purpose. This adaptability means that new capabilities likewise are composed of loosely coupled agents that are added into the general framework as they are developed. KMDT has developed mostly data-integration agents that retrieve, translate, analyze, and combine data from networked data sources with different data representations. Transformation and integration of relevant tracking data, non-spatial data, and spatial data add value. Support agents using data-staging structure like high-speed cache memory facilitate these processes.

The KMDT agent classes were developed and tested on previous projects. Support classes also have been developed and used successfully on past projects. Both the agent and support classes are extensible. Their extensible framework will allow more functionality to be added incrementally over the life of the KMDT project. KMDT agents communicate over TCP/IP. Agent communication code rarely is reused project to project. One reason is that the experimental nature of the code itself usually means that it is more efficient to develop code "from scratch" than to customize code from another project. Another reason is that agent communication in general has become increasingly more complex to implement over time.

We use the Java® event model and sockets for most agent communication. Before the time of firewalls, e-mail worms, Web services, eXtensible Markup Language (XML), and Simple Object Access Protocol (SOAP), agent communication was more straightforward to implement. Socket connections were made between agents and data were exchanged. Communicating via TCP/IP means having to resolve how agents connect to each other, authenticate themselves, encode and decode XML messages, receive messages, and report errors.

Each type of agent is assigned a specific high-level task. KMDT features the following types of agents.

- The **Timer Agent** is like a stopwatch or an alarm clock. It enables the autonomous behavior of an agent through timed method calls. It is installed as part of the KMDT package.
- The **Monitoring Agent** is the base class of the File Agent. The Monitoring Agent can alert the user or another agent whenever a file is updated, whenever the size of a file changes, or whenever a file is deleted. The Monitoring Agent can perform one of four actions when it detects a change. The Monitoring Agent can display an alert, execute a shell command, start an application, or notify another agent about the change.
- The **File Agent** sends recently collected sensor LOB data from the M&S simulation to the **Transfer-Data-to-Database Agent**, which sometimes is called the "Database-Interface Agent."
- The **Data-to-Database Agent** accepts the simulated LOB detection data as comma-separated values from the File Agent over a socket. Then it creates the tables and stores the detection-log data in a detection database using Microsoft Access. The Data-to-Database Agent calculates the end-point latitude and longitude for each LOB entry and stores the information in the database. The Transfer-Data-to-Database Agent is the base class of the Data-to-Database Agent.
- The **Cross-Fix Display Agent** accesses the detection-log data directly from the detection database, downloads the data into local memory, and transforms each LOB from a location into a vector. Then it examines other detections for possible a LOB cross-fix. The Cross-Fix Display Agent attempts to associate each validated LOB with a crossing LOB from the detection-log database, sometimes called the "contact database."
- **The Temporal-Proximity Agent** screens out cross-fixes with time signatures that do not fall within a specified time interval of each other. This agent can display its output in a Microsoft Excel® spreadsheet.
- The user can select an option to trigger an alert-type notification to the user when new data are received. The **Alarm Agent** issues this alert to the user interface.
- The **URL Reader Agent** reads the text of Web pages and forwards the text to any other agent that is listening.
- The **Filter Agent** scores information using a keyword list, a Kohonen neural network, and maintains a user profile.
- The **Agent-Control Interface** enables the user to create and deploy various intelligent agents.

A typical sequence of events for agent use is as follows.

1. As described above, the M&S program generates a detection log every time the program is executed.
2. The operator creates and deploys the File-Monitor Agent with the user-friendly KMDT interface.
3. The File-Monitor Agent detects the presence of updated data in the detection log and issues an alert to the operator.
4. The operator creates and deploys the Data-to-Database Agent with the user-friendly KMDT interface.
5. The Data-to-Database agent ingests the detection-log data and calculates the end point latitude and longitude for each entry.
6. The Data-to-Database agent stores the data in a Microsoft Access® database.

7. The operator creates and deploys the Cross-Fix Display Agent with the user-friendly KMDT interface.
8. The Cross-Fix Display Agent establishes a connection with the database and downloads the data into local memory.
9. The Cross-Fix Display Agent finds all of the LOBs that cross each other, regardless of timing and regardless of LOB validity with respect to false alarms.
10. The operator creates and deploys the Temporal-Feasibility Agent with the user-friendly KMDT interface.
11. The Temporal-Feasibility Agent searches the cross-fixes to determine which ones occurred within the specified time of each other. This window is typically a half an hour.
12. The results of the computations completed by the Cross-Fix Display Agent and the Temporal-Feasibility Agent are displayed on the screen.

3.2 AGENT-CONTROL INTERFACE AND AGENT FUNCTIONS

Figures 5 through 17 show examples of the user-interface screens and results of the intelligent-software agents described in the last section. The example shown in Figures 15 and 16 is like the one in Figures 11 and 12, except that Figures 15 and 16 illustrate the option to display only the cross-fixes as small circles, and not as intersections of LOBs. This option helps reduce screen clutter. In some figures, two concentric circles around each sensor location signify a variable detection range where obstacles, such as islands, limit the range over some angles, but are not an obstruction at other angles. The specific angles are coded accurately in the sensor-specification file of the modeling and simulation program, but the exact shape of the detection area is not shown on the display.

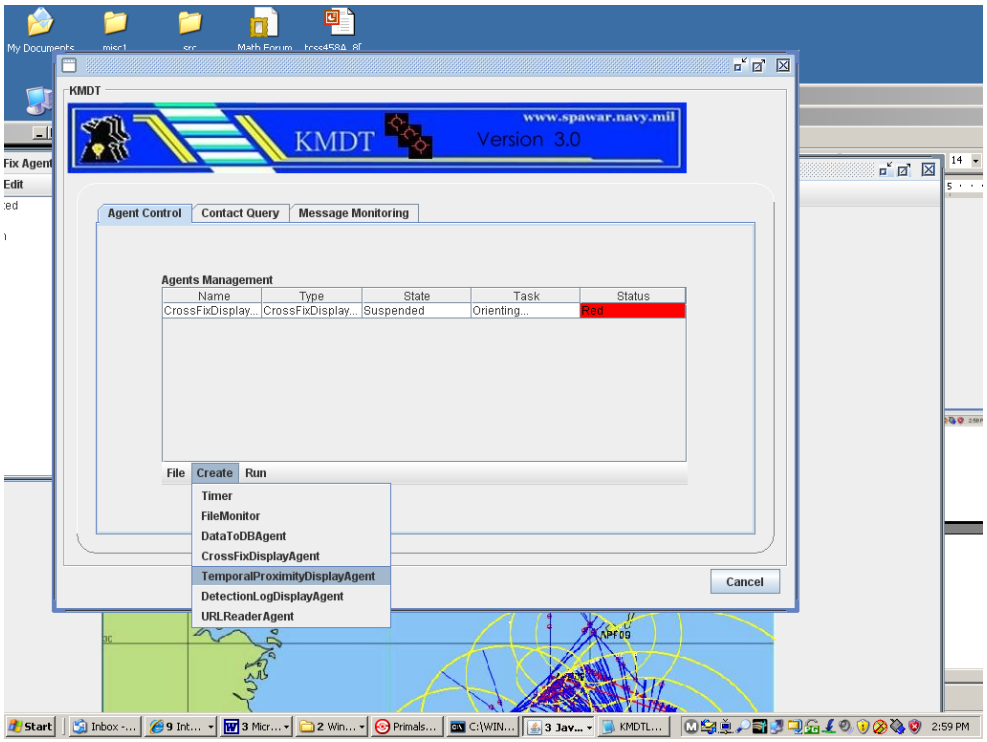


Figure 5. Top-level operator's interface for creating and controlling the agents. This screen shows that the Cross-Fix Display Agent has finished executing.

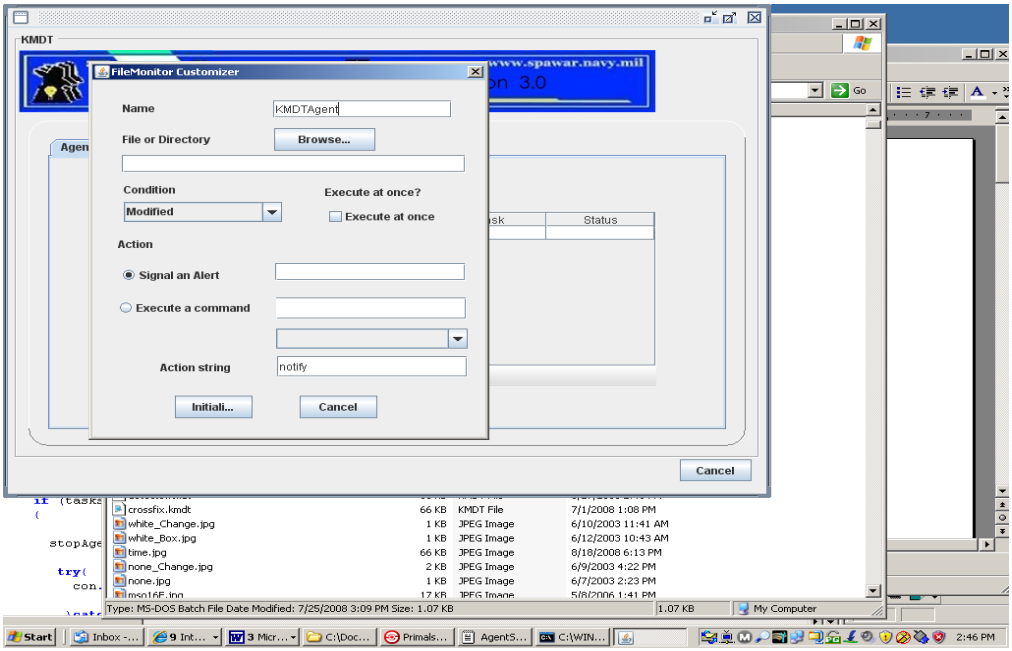


Figure 6. File-Monitor Agent input control screen.

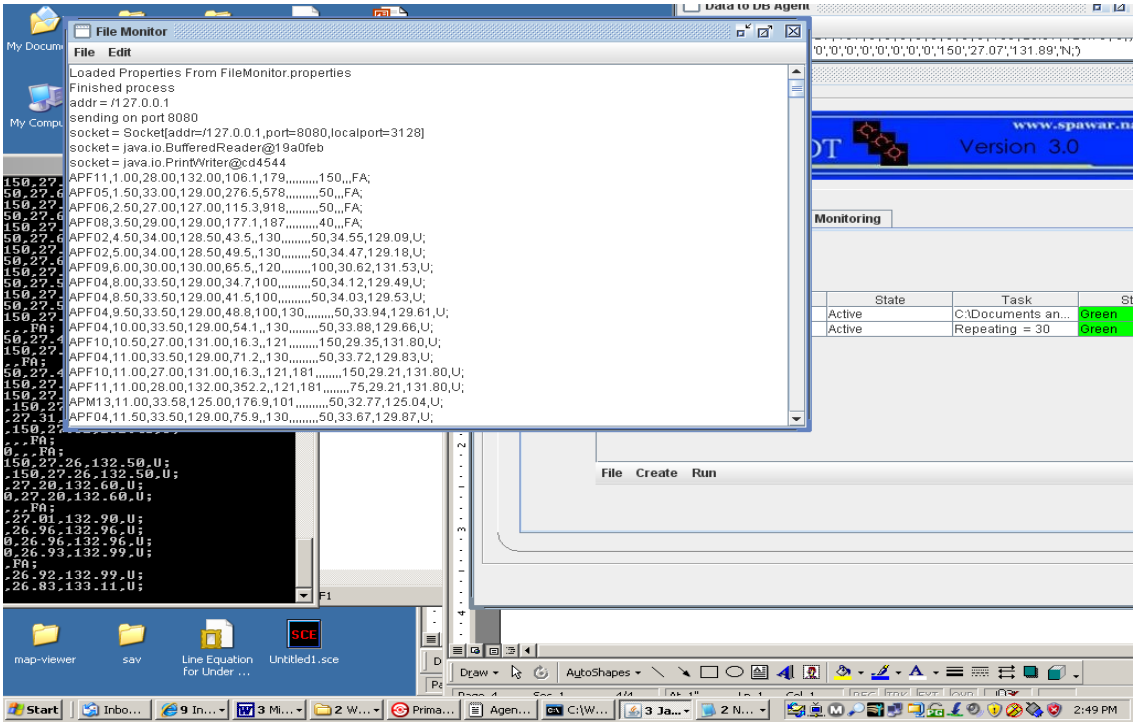


Figure 7. File-Monitor Agent results display.

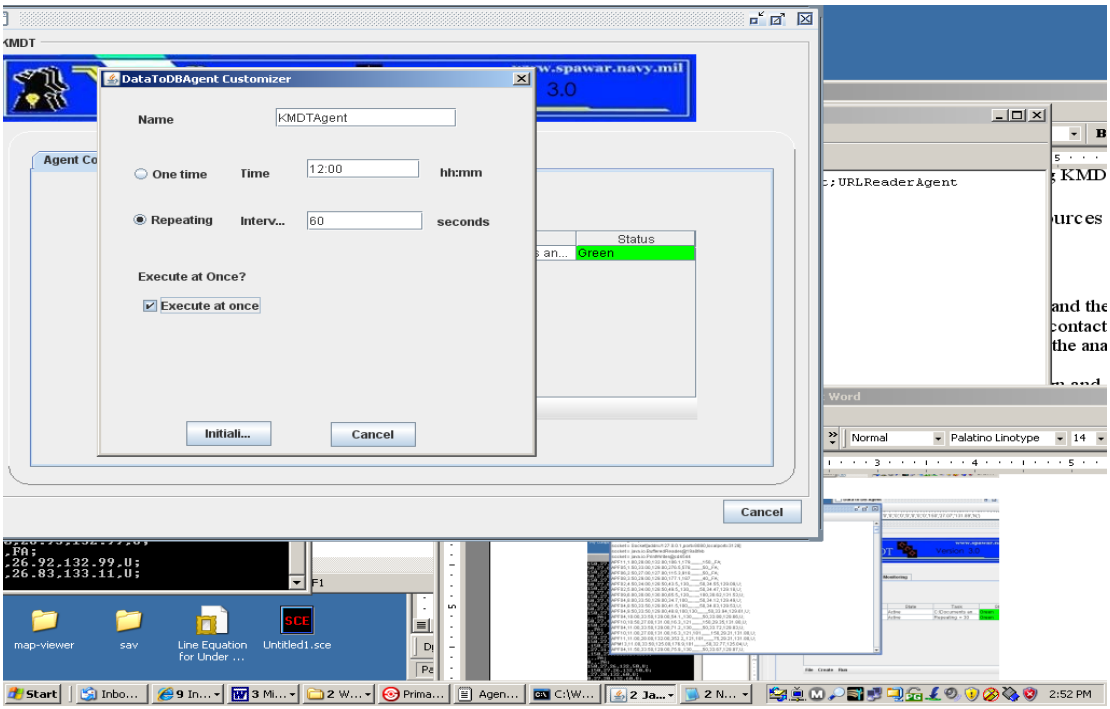


Figure 8. Data-to-Database Agent input control screen.

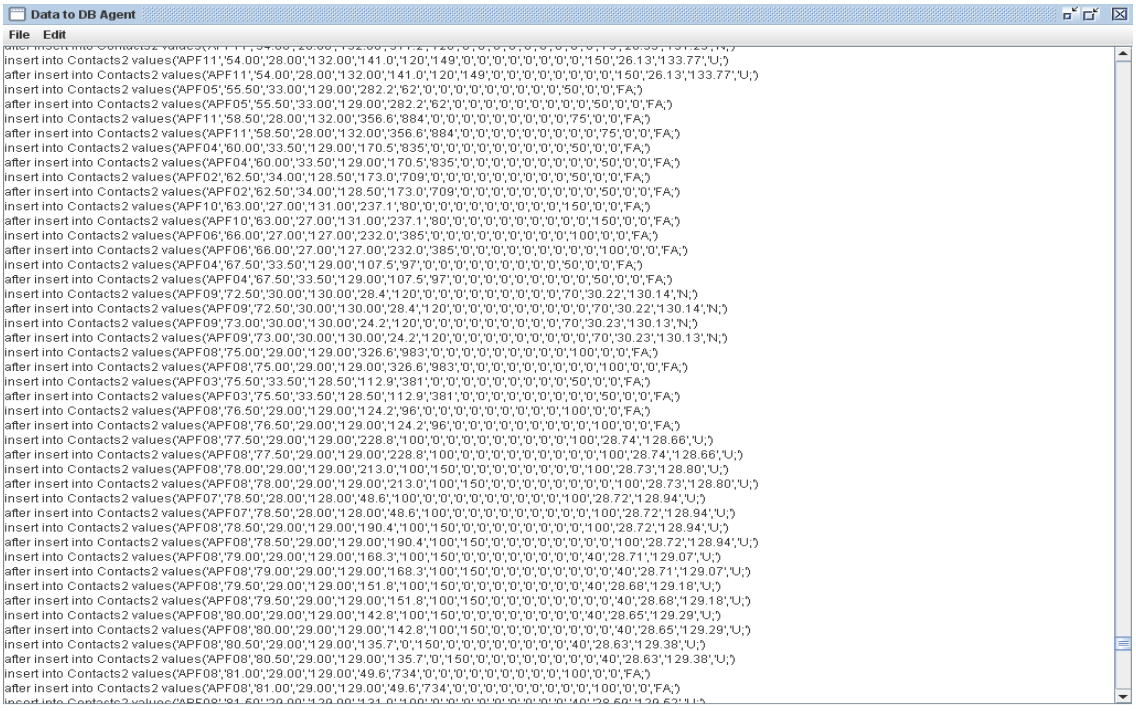


Figure 9. Data-to-Database Agent results display.

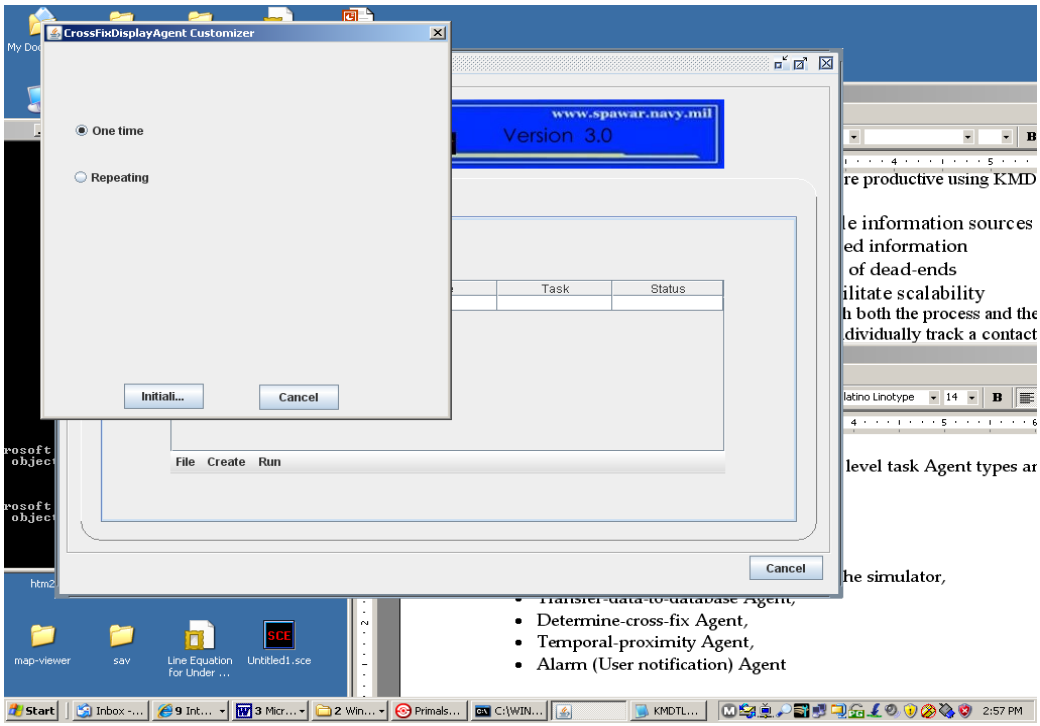


Figure 10. Cross-Fix Display Agent input control screen.

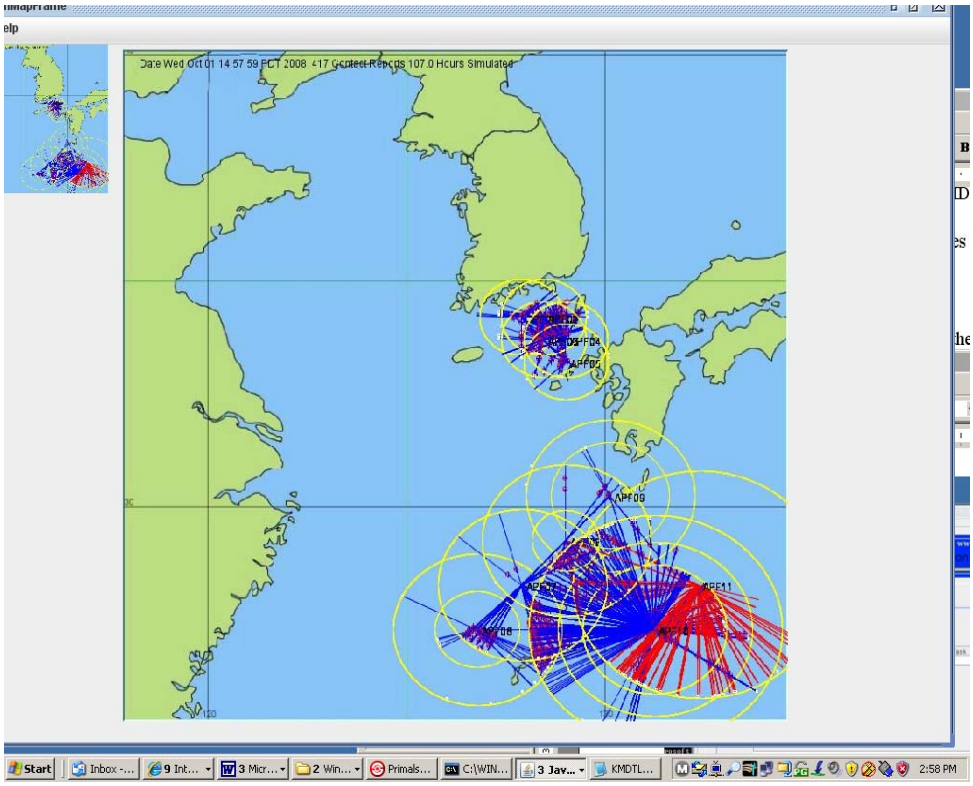


Figure 11. Cross-Fix Display Agent results display.

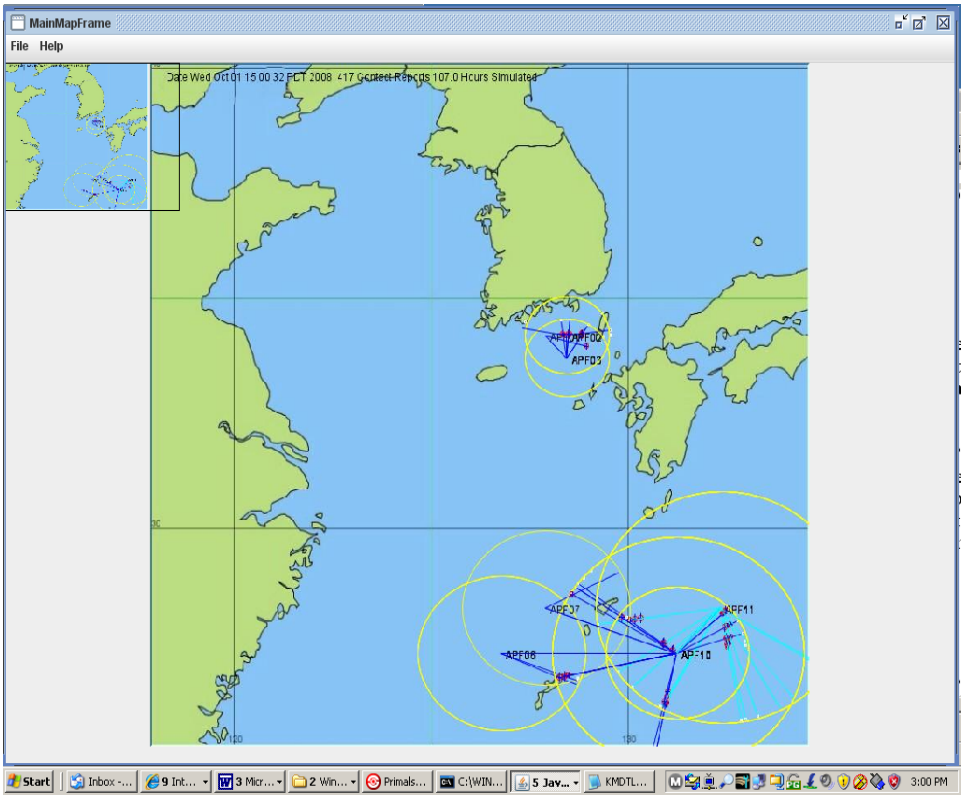


Figure 12. Temporal-Proximity Display Agent results display.

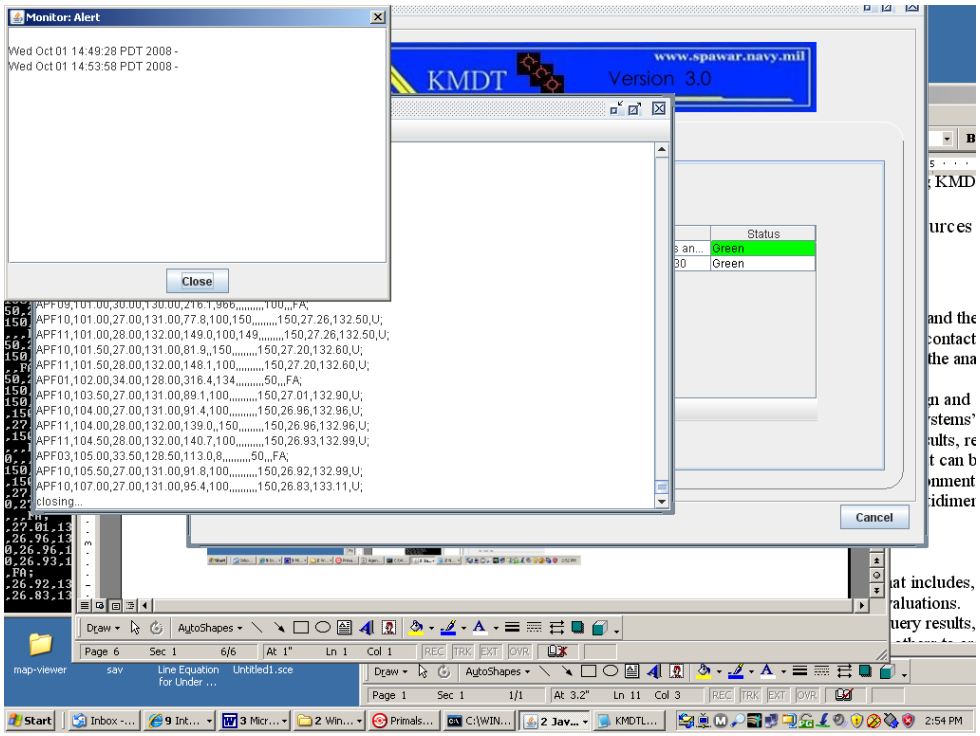


Figure 13. Alarm (User-Notification) Agent results display.

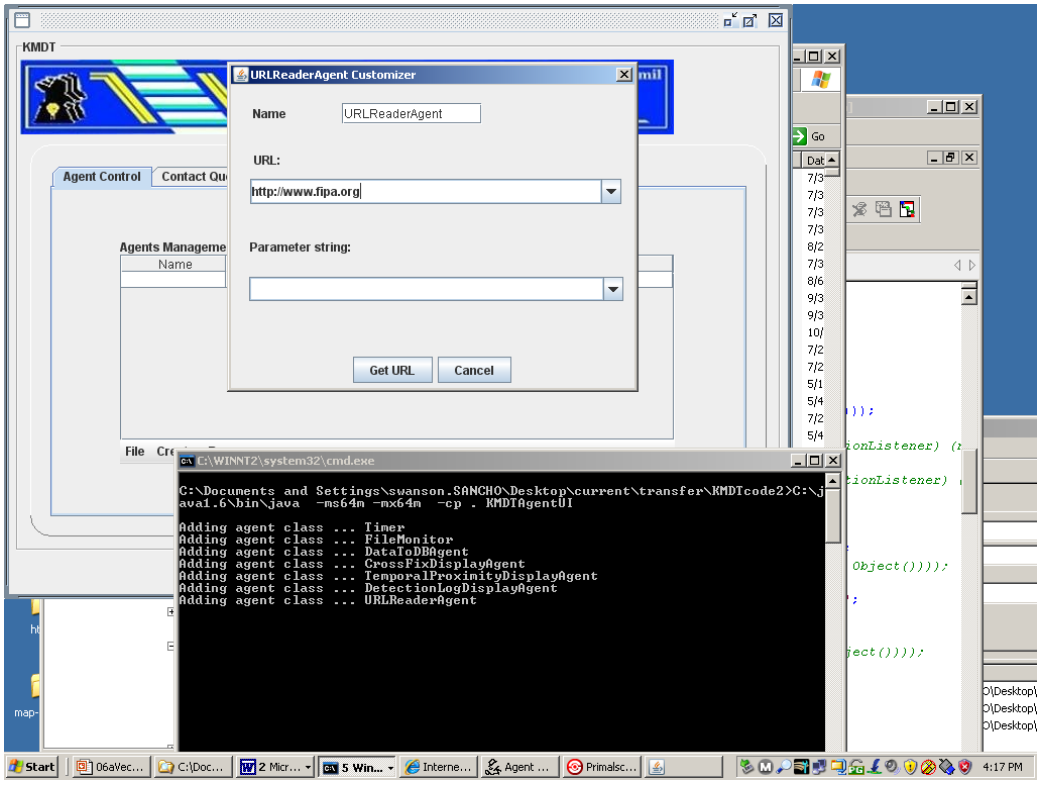


Figure 14. URL-Reader Agent input control screen.

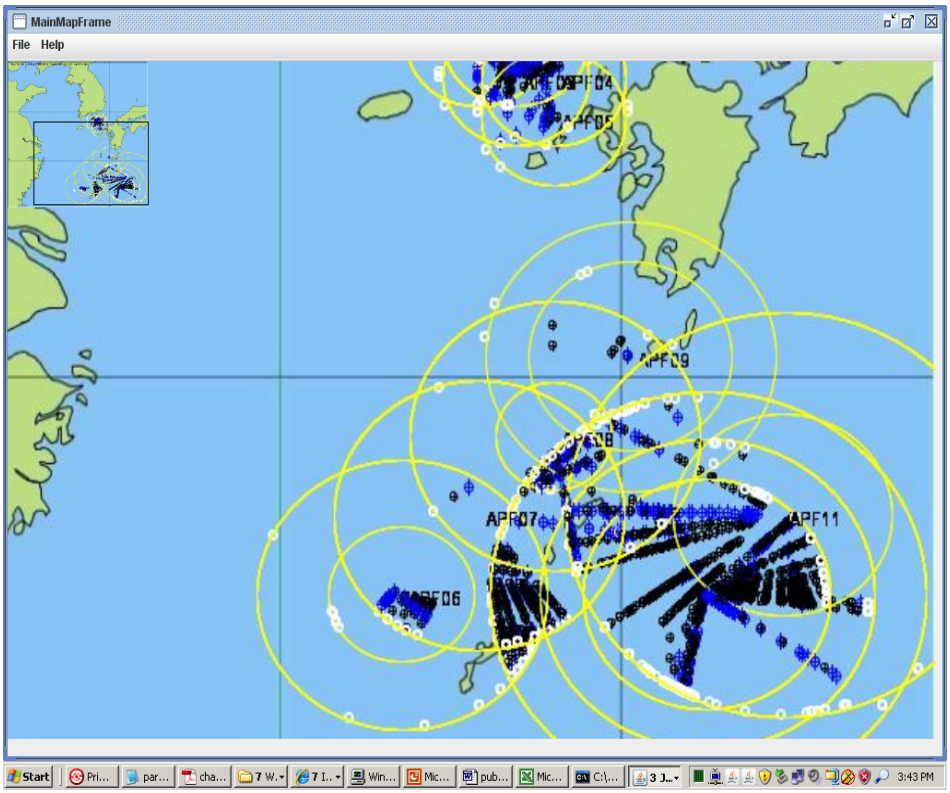


Figure 15. Result of Cross-Fix Display Agent.

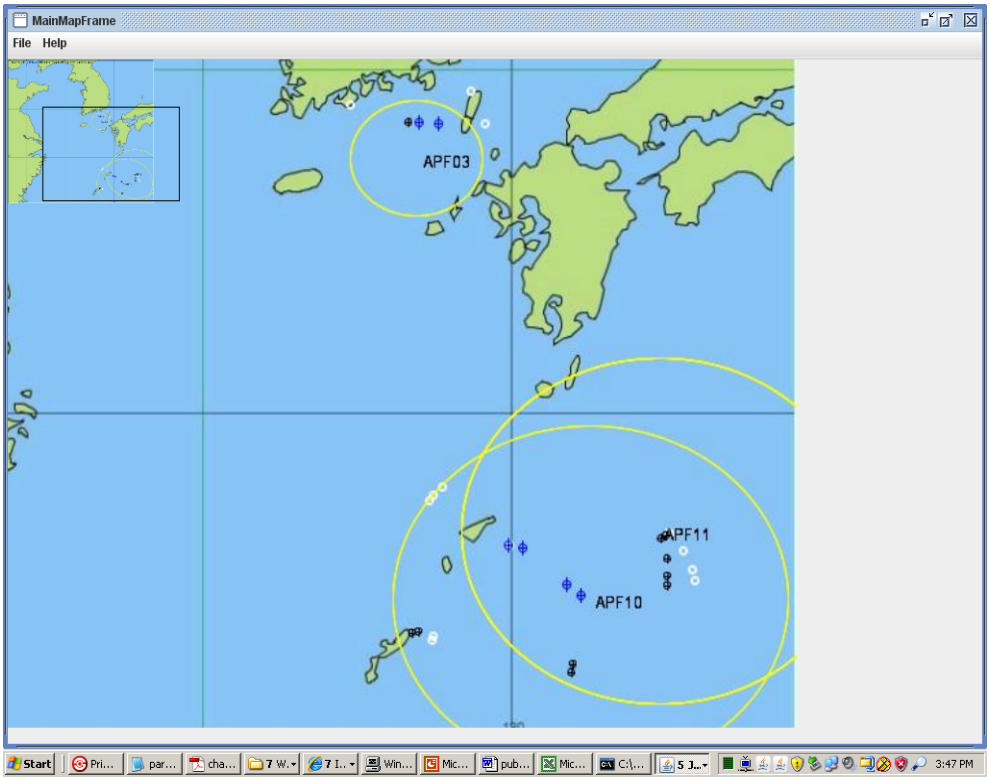


Figure 16. Result of Temporal-Proximity Agent.

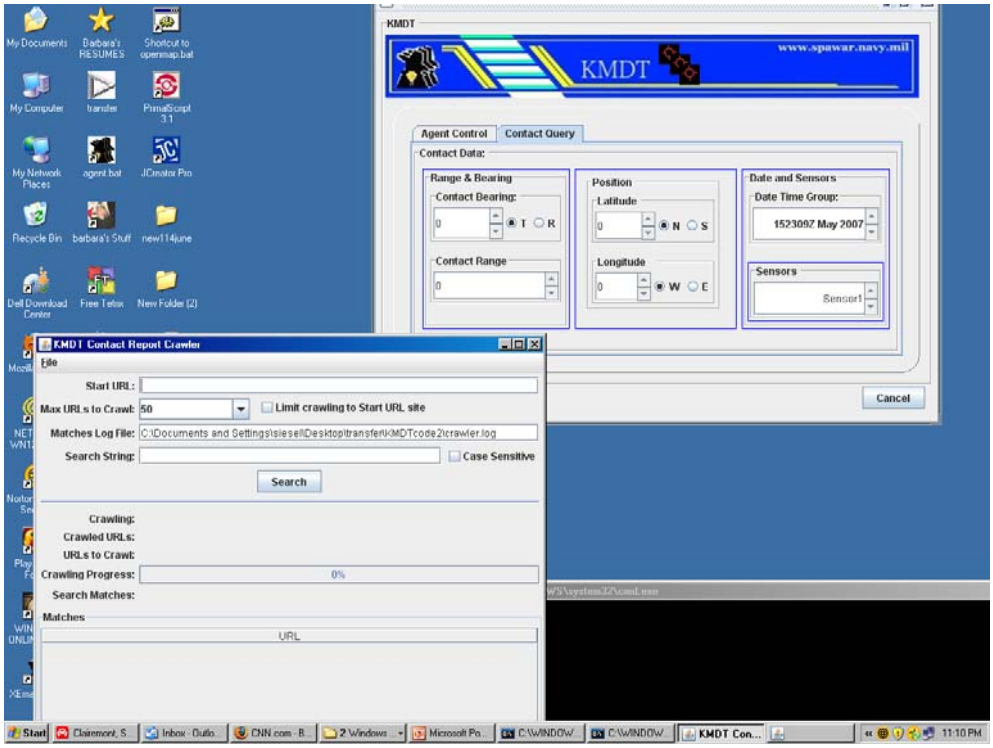


Figure 17. Contact-Query Panel that generates a search string for Contact-Report Crawler.

The results of applying the agents to another simulated data set are shown in Figures 15 and 16. This example is like the one shown in Figures 11 and 12, except that Figures 15 and 16 illustrate the option to display only the cross-fixes as small circles and not the lines of bearing. This option helps reduce screen clutter. Two concentric circles are drawn around sensors that have variable detection ranges due to obstacles, such as islands, in their vicinity.

Figure 17 shows the Contact-Query input screen from which an operator would deploy an agent to search the secure network for LOBs that cross a specified line of bearing for which the operator wanted additional information. When finished executing, the Contact-Report Crawler would list the URLs that have the desired information.

3.3 DETERMINING THE VALIDITY OF CROSS-FIXES AND THEIR COMPONENTS

When the Cross-Fix Display Agent associates validated LOBs with a crossing LOB from the contacts database, the false LOB detections that do not cross any other LOB will be eliminated. The Cross-Fix Display Agent finds all of the LOBs that cross each other, regardless of timing and regardless of LOB validity with respect to false alarms. Because the LOBs are line segments to a first approximation, the maximum detection range will determine the length of these LOBs, not to extend beyond the maximum range, which ensures the geofeasibility of observed LOBs involved in cross-fixes.

As described above, the Temporal-Feasibility Agent searches the cross-fix data to determine which ones were observed within the specified time interval of each other. This window is typically a half an hour. When the Temporal-Proximity Agent screens out cross-fixes with time signatures that do not fall within this specified time interval of each other, the LOBs that were observed outside of these time intervals are excluded from further consideration.

False LOBs are generated according to a stochastic process in the M&S program, whereas valid LOBs that represent correct detections of platforms with valid crossing LOBs represent a systematic process. Thus, the Temporal-Proximity Agent can eliminate many (but not all) false LOBs because few of them will cross other LOBs that occur within in the specified time period of each other.

The Cross-Fix Display Agent and the Temporal-Feasibility Agent indirectly screen out most false LOBs as described above. Direct screening of the remaining data sets to eliminate all other false alarms with total certainty becomes much more difficult at this point because, theoretically, a false alarm can be generated with any spectral characteristics, including those of known, valid platforms. Such a false alarm will be indistinguishable from an actual correct detection except by association with other cross-fixes that occur within the same track.

If all the LOBs observed in a track that consists of cross-fixes screened in the manner described above have the same spectral signature, these cross-fixes correctly are components of the track that belongs to the same valid platform. However, if the spectral signature of an LOB in a cross-fix that appears to be a member of this track is very different from that of the other LOBs comprising the track, one can surmise that the LOB with the different signature is not a valid member of the track. It could belong to a different track or it could be a false alarm.

The design calls for the agents to attempt to classify each contact by comparing spectral (source frequency) signature data in the LOB records against that of known platforms, including targets of interest maintained in appropriate databases. This classification helps determine whether the cross-fixes belong to known platforms. However, it will not be sufficient to identify the cross-fixes that have false alarms, or a track generated by an unknown platform whose data are not stored in the characteristics database.

4. GEOGRAPHIC ALGORITHMS

The three equivalent methods were implemented for finding cross-fixes areas as follows.

1. **The Parametric Method** is described in Section 4.1. This method consists of solving two simultaneous linear parametric equations to find the latitude and longitude of the intersection point.

2. The **Vector Theorem Test** is described in Section 4.2. The result of this algorithm represents the truth or falsehood of a cross-fix but does not determine the point of intersection.

3. A **Vector Cross Product Test** also was developed to find the latitude and longitude of the intersection point.

4.1 THE PARAMETRIC METHOD—LOB INTERSECTION ALGORITHM

The geometry for the LOB intersection algorithm is illustrated in Figure 18. Let S_1 and S_2 be two sensor systems with maximum detection ranges (MDRs) of r_1 and r_2, respectively. Note that since acoustic or electromagnetic energy propagates along geodetic (great circle) paths, the ranges r_1 and r_2 are not simply radii of circles. Similarly, LOBs from sensor to target are not rhumb (constant bearing) lines. The reported LOB of a contact by a sensor is defined as the initial bearing of the geodetic line *at the sensor*, as the bearing generally changes along the path of the geodetic.

Let θ_1 and θ_2 be the reported LOBs for concurrent contacts detected by sensors S_1 and S_2, respectively. If geodetic line segments corresponding to these LOBs intersect *within their respective MDRs*, the location of a potential target responsible for the LOBs may be determined as follows:

1. Given the latitude ϕ and longitude λ of sensor S_1 (the departure), compute the latitude ϕ' and longitude λ' of the location D_1 (the destination) at MDR, r_1, along the geodetic line from S_1 with initial LOB, θ_1:

$$\lambda' = \lambda + \tan^{-1}\left[\sin\theta_1 \sin(r_1/R)/\left(\cos\phi\cos(r_1/R) - \sin\phi\sin(r_1/R)\cos\theta_1\right)\right]$$
$$\phi' = \tan^{-1}\left[\left(\sin\phi\cos(r_1/R) + \cos\phi\sin(r_1/R)\cos\theta_1\right)\sin(\lambda' - \lambda)/\sin\theta_1\sin(r_1/R)\right].$$

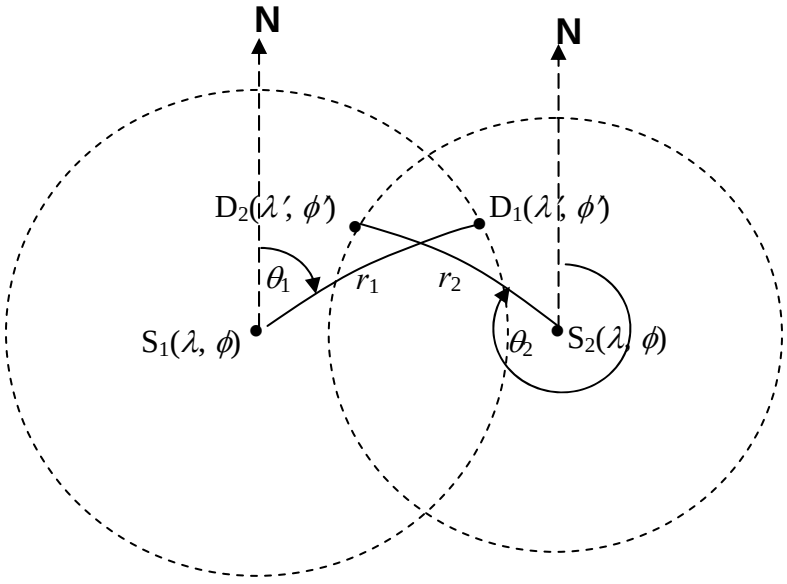


Figure 18. Geometry of two intersecting sensor systems.

Here, all angles are expressed in radians, and r_1/R is the angle at the center of the earth subtending the geodetic segment r_1. R is defined as the radius of the earth. Similarly, compute the destination D_2 along the geodetic line from sensor S_2.

2. Transform the geographic coordinates (λ, ϕ) of the endpoints of each geodetic line segment into rectangular coordinates (x, y) using the Gnomonic projection:

$$x = \cos\phi \sin(\lambda - \lambda_0) / \cos c$$
$$y = [\cos\phi_0 \sin\phi - \sin\phi_0 \cos\phi \cos(\lambda - \lambda_0)] / \cos c,$$

where $\cos c = \sin\phi_0 \sin\phi + \cos\phi_0 \cos\phi \cos(\lambda - \lambda_0)$. Here, the central parallel ϕ_0 and the central meridian λ_0 define the center of the projection (i.e., the point of tangency of the projection plane on the earth's surface), which may be situated anywhere in the same hemisphere as the line segments so that cos $c > 0$. The Gnomonic projection transforms any geodetic line into a straight line on the rectangular (x, y) projection plane. Thus the transformed rectangular endpoints (x_1, y_1) and (x'_1, y'_1) for S_1D_1 and (x_2, y_2) and (x'_2, y'_2) for S_2D_2 define straight-line segments in the projection plane.

3. Represent the straight lines containing the segments S_1D_1 and S_2D_2 by the parametric representations,

$$(1 - p)S_1 + pD_1 \text{ and } (1 - q)S_2 + qD_2,$$

respectively, where $S_1 = (x_1, y_1)$, $D_1 = (x'_1, y'_1)$, and $S_2 = (x_2, y_2)$, $D_2 = (x'_2, y'_2)$. The intersection (if any) of the lines occurs where these parametric representations are equal, i.e., where the parameters p and q are determined from the system of equations,

$$(1-p)x_1 + px'_1 = (1-q)x_2 + qx'_2$$
$$(1-p)y_1 + py'_1 = (1-q)y_2 + qy'_2,$$

or upon rewriting in matrix form,

$$\begin{bmatrix} (x'_1 - x_1) & (x'_2 - x_2) \\ (y'_1 - y_1) & (y'_2 - y_2) \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} (x_2 - x_1) \\ (y_2 - y_1) \end{bmatrix}.$$

If this system of equations fails to yield a solution (i.e., the matrix of coefficients is singular), the lines are parallel or coincide. Otherwise, if $0 \le p \le 1$ and $0 \le q \le 1$, the intersection of the lines will occur within the line segments S_1D_1 and S_2D_2.

4. Determine the rectangular coordinates (x, y) of the intersection by evaluating the parametric representations for either p or q; for example,

$$x = (1-p)x_1 + px'_1$$
$$y = (1-p)y_1 + py'_1.$$

5. Transform the rectangular coordinates (x, y) of the intersection into geographic coordinates (λ, ϕ) using the inverse Gnomonic projection:

$$\phi = \sin^{-1}[\cos c \sin\phi_0 + (y \sin c \cos\phi_0 / \rho)]$$
$$\lambda = \lambda_0 + \tan^{-1}[x \sin c / (\rho \sin\phi_0 \cos c - y \sin\phi_0 \sin c)],$$

where $\rho = (x^2 + y^2)^{1/2}$ and $c = \tan^{-1}(\rho)$. If $\rho = 0$ then $\phi = \phi_0$ and $\lambda = \lambda_0$. Also, if $\phi_0 = \pm 90°$, then $\lambda = \lambda_0 + \tan^{-1}[x/(\mp y)]$.

4.1.1 Algorithm for Computing the Intersection of Two Bearings on the Earth

The following algorithm computes the longitude λ and latitude ϕ of the intersection of two bearings β_1 and β_2 from two known locations $P_1(\lambda_1, \phi_1)$ and $P_2(\lambda_2, \phi_2)$, respectively. By convention, latitudes are specified between $\pm\pi/2$ with north latitudes positive and south latitudes negative; longitudes are specified between $\pm\pi$ with east longitudes positive and west longitudes negative. Bearings are specified between $\pm\pi$ with bearings increasing in the clockwise direction and with 0 coinciding with true north. The geometry of the general case is shown in Figure 19, where all curves are arcs of great circles on the surface of a sphere. LOBs from points P_1 and P_2 are shown as solid curves. Meridians at longitudes λ_1, λ_2, and λ are shown as dashed curves from the north pole N through points P_1, P_2, and P, respectively. Note that spherical triangle sides are specified as the angle at the center of the sphere subtending the side.

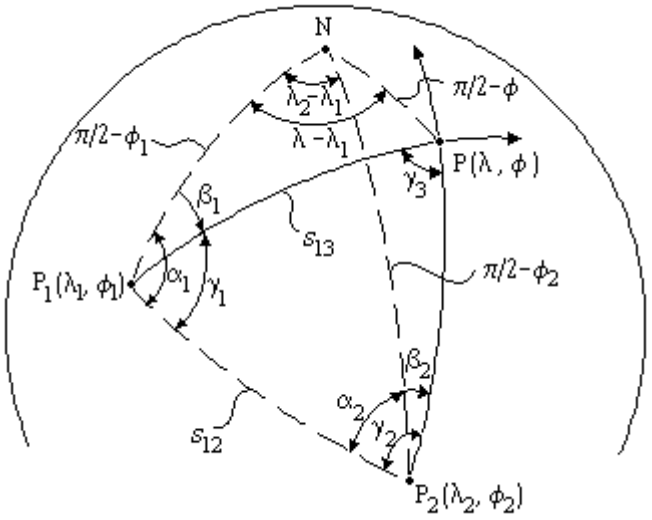


Figure 19. Spherical geometry of bearing intersection computation.

For triangle ΔNP_1P_2:

1. Compute sine and cosine of angle subtending side s_{12}: From the Law of Cosines for sides of spherical triangles,

$$\begin{aligned}\cos s_{12} &= \cos(\tfrac{\pi}{2} - \phi_1)\cos(\tfrac{\pi}{2} - \phi_2) + \sin(\tfrac{\pi}{2} - \phi_1)\sin(\tfrac{\pi}{2} - \phi_2)\cos(\lambda_2 - \lambda_1) \\ &= \sin\phi_1 \sin\phi_2 + \cos\phi_1 \cos\phi_2 \cos(\lambda_2 - \lambda_1)\end{aligned}$$

and

$$\sin s_{12} = (1 - \cos^2 s_{12})^{1/2}.$$

2. Compute angles α_1 and α_2: From the Law of Sines for spherical triangles,

$$\frac{\sin\alpha_1}{\sin(\frac{\pi}{2} - \phi_2)} = \frac{\sin(\lambda_2 - \lambda_1)}{\sin s_{12}},$$

that is,

$$\sin\alpha_1 = \sin(\tfrac{\pi}{2} - \phi_2)\sin(\lambda_2 - \lambda_1)/\sin s_{12}$$
$$= \cos\phi_2 \sin(\lambda_2 - \lambda_1)/\sin s_{12}\,.$$

From the Law of Cosines for sides of spherical triangles,

$$\cos(\tfrac{\pi}{2} - \phi_2) = \cos(\tfrac{\pi}{2} - \phi_1)\cos s_{12} + \sin(\tfrac{\pi}{2} - \phi_1)\sin s_{12}\cos\alpha_1$$
$$\sin\phi_2 = \sin\phi_1\cos s_{12} + \cos\phi_1 \sin s_{12}\cos\alpha_1\,,$$

and after some manipulation,

$$\cos\alpha_1 = \left[\cos\phi_1 \sin\phi_2 - \sin\phi_1\cos\phi_2\cos(\lambda_2 - \lambda_1)\right]/\sin s_{12}\,.$$

Thus,

$$\tan\alpha_1 = \frac{\sin\alpha_1}{\cos\alpha_1} = \frac{\cos\phi_2\sin(\lambda_2 - \lambda_1)}{\cos\phi_1\sin\phi_2 - \sin\phi_1\cos\phi_2\cos(\lambda_2 - \lambda_1)}$$
$$\alpha_1 = \tan^{-1}\left[\frac{\cos\phi_2\sin(\lambda_2 - \lambda_1)}{\cos\phi_1\sin\phi_2 - \sin\phi_1\cos\phi_2\cos(\lambda_2 - \lambda_1)}\right].$$

Similarly,

$$\alpha_2 = \tan^{-1}\left[\frac{\cos\phi_1\sin(\lambda_2 - \lambda_1)}{\sin\phi_1\cos\phi_2 - \cos\phi_1\sin\phi_2\cos(\lambda_2 - \lambda_1)}\right].$$

For triangle ΔP_1P_2P:

3. Compute angles γ_1 and γ_2:

$$\gamma_1 = \alpha_1 - \beta_1$$
$$\gamma_2 = \alpha_2 + \beta_2\,.$$

4. Compute sine and cosine of angle γ_3: From the Law of Cosines for angles of spherical triangles,

$$\cos\gamma_3 = -\cos\gamma_1\cos\gamma_2 + \sin\gamma_1\sin\gamma_2\cos s_{12}$$

and

$$\sin\gamma_3 = (1 - \cos^2\gamma_3)^{1/2}\,.$$

5. Compute sine and cosine of angle subtending side s_{13}: From the Law of Sines for spherical triangles,

$$\frac{\sin s_{13}}{\sin\gamma_2} = \frac{\sin s_{12}}{\sin\gamma_3},$$

that is,

$$\sin s_{13} = \sin\gamma_2 \sin s_{12}/\sin\gamma_3\,.$$

From the Law of Cosines for angles of spherical triangles,

$$\cos\gamma_2 = -\cos\gamma_1\cos\gamma_3 + \sin\gamma_1\sin\gamma_3\cos s_{13},$$

that is,

$$\cos s_{13} = (\cos\gamma_2 + \cos\gamma_1 \cos\gamma_3)/(\sin\gamma_1 \sin\gamma_3)\,.$$

For triangle ΔNP_1P:

6. Compute latitude ϕ: From the Law of Cosines for sides of spherical triangles,

$$\cos(\tfrac{\pi}{2} - \phi) = \cos(\tfrac{\pi}{2} - \phi_1)\cos s_{13} + \sin(\tfrac{\pi}{2} - \phi_1)\sin s_{13}\cos\beta_1$$
$$\sin\phi = \sin\phi_1 \cos s_{13} + \cos\phi_1 \sin s_{13}\cos\beta_1$$

and

$$\cos\phi = (1 - \sin^2\phi)^{1/2}\,.$$

Thus,

$$\tan\phi = \sin\phi/\cos\phi$$
$$\phi = \tan^{-1}[\sin\phi/\cos\phi].$$

7. Compute longitude λ: From the Law of Sines for spherical triangles,

$$\frac{\sin(\lambda - \lambda_1)}{\sin s_{13}} = \frac{\sin\beta_1}{\sin(\tfrac{\pi}{2} - \phi)},$$

that is,

$$\sin(\lambda - \lambda_1) = \sin s_{13}\sin\beta_1/\sin(\tfrac{\pi}{2} - \phi)$$
$$= \sin s_{13}\sin\beta_1/\cos\phi.$$

From the Law of Cosines for sides of spherical triangles,

$$\cos s_{13} = \cos(\tfrac{\pi}{2} - \phi_1)\cos(\tfrac{\pi}{2} - \phi) + \sin(\tfrac{\pi}{2} - \phi_1)\sin(\tfrac{\pi}{2} - \phi)\cos(\lambda - \lambda_1)$$
$$= \sin\phi_1 \sin\phi + \cos\phi_1 \cos\phi\cos(\lambda - \lambda_1)$$
$$\cos(\lambda - \lambda_1) = (\cos s_{13} - \sin\phi_1 \sin\phi)/(\cos\phi_1 \cos\phi).$$

Thus,

$$\tan(\lambda - \lambda_1) = \frac{\sin(\lambda - \lambda_1)}{\cos(\lambda - \lambda_1)}$$
$$\lambda = \lambda_1 + \tan^{-1}\left[\frac{\sin(\lambda - \lambda_1)}{\cos(\lambda - \lambda_1)}\right].$$

4.2 THE VECTOR THEOREM TEST METHOD—LOB INTERSECTION ALGORITHM

Two conditions are necessary and sufficient to demonstrate intersection between line segments.

Condition 1: Two line segments do *not* intersect if and only if one segment lies entirely to one side of the line containing the other segment. Referring to Figure 20, $(p_3 - p_2)$ x $(p_1 - p_2)$ and $(p_4 - p_2)$ x $(p_1 - p_2)$ are both positive.

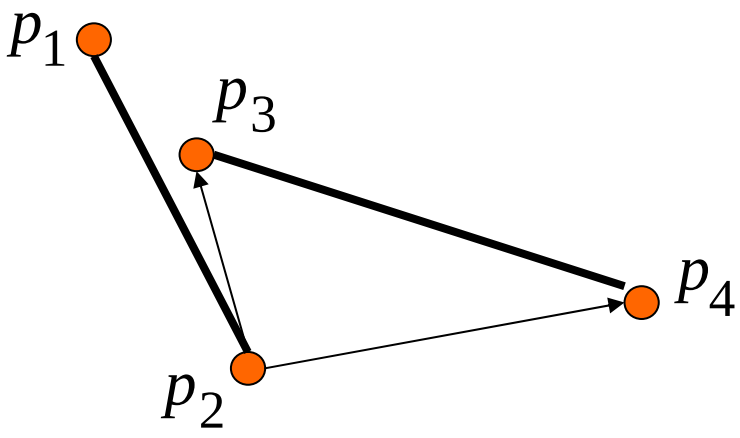


Figure 20. Line-segment geometry for cross-fix test—no intersection.

Condition 2: Two-line segments intersect if and only if **each** of the two pairs of cross products in Figure 21 have different signs (or one cross product in the pair is 0).

$(p_1 - p_4)$ x $(p_3 - p_4)$ and $(p_2 - p_4)$ x $(p_3 - p_4)$ => The line through $(p_3 - p_4)$ intersects the line through $(p_1 - p_2)$.

$(p_3 - p_2)$ x $(p_1 - p_2)$ and $(p_4 - p_2)$ x $(p_1 - p_2)$ => The line through $(p_1 - p_2)$ intersects the line through $(p_3 - p_4)$.

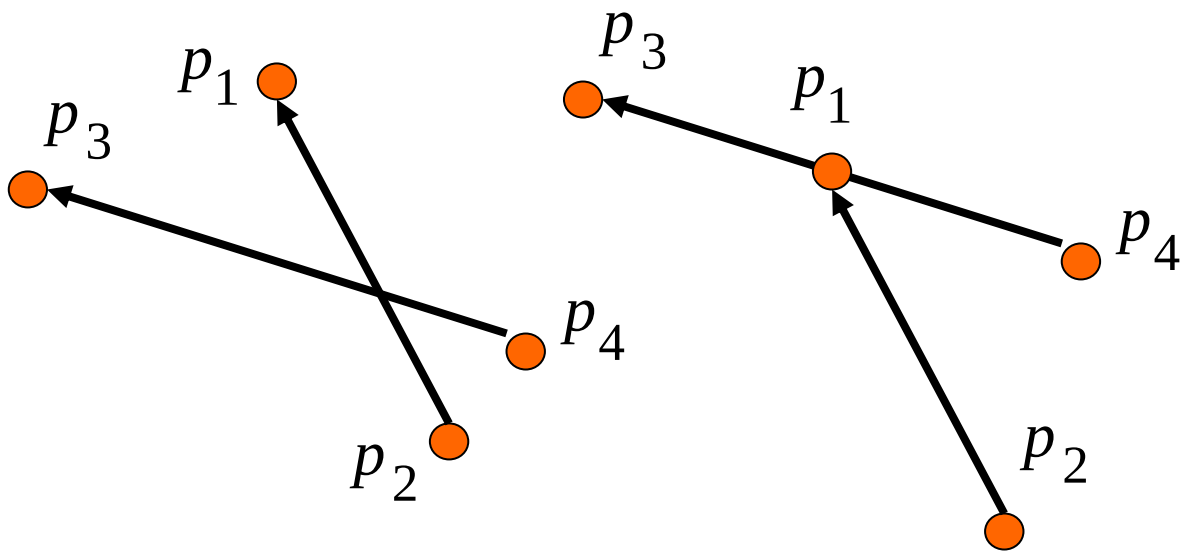


Figure 21. Line-segment geometry for cross-fix test—intersection.

4.3 THE GEO-FEASIBILITY FILTRATION PROCESS

The geo-feasibility filtration process filters out proposed cross-fixes that arise from sensors whose detection-coverage area do not overlap with each other. The remaining cross-fixes are considered geographically feasible. Figure 22 depicts two sensors with overlapping detection coverage, and two without overlap. The sensors are depicted as dots inside of small circles and the maximum detection ranges are depicted as the large circles. First, sensor overlap is a necessary but insufficient condition for valid cross-fixes. For example, the distance between sensor 1 and sensor 2, d, should be less than the sum of the maximum detection ranges, d < R1 + R2, as shown in Figure 22. This guarantees sensor overlap. The best and most definitive detections occur when the LOBs cross at 90 degrees.

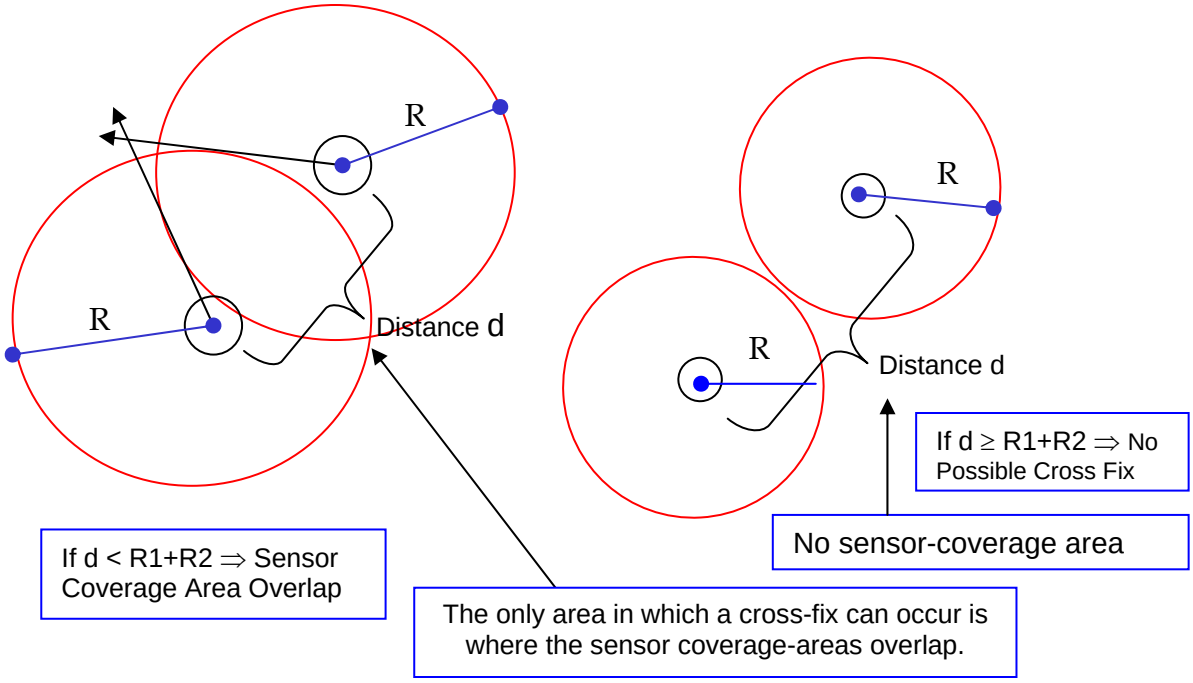


Figure 22. Role of sensor geometry in the geo-feasibility screening process.

In contrast, the worst cases occur when the LOBs are collinear, which can happen when each LOB in the cross-fix points to the other sensor, and the platform detected is between the two sensors, on the line segment that connects them (Figure 23a). It also can happen when the LOBs point in the same direction when the platform is located on a line extrapolated in either direction from this line segment. Even when the distance between the sensors is less than the sum of the maximum detection ranges, a cross-fix will not occur if the LOBs are oriented away from each other, as is the case in Figure 23b. Moreover, a LOB originating from a sensor is not compared against any other LOB originating from that same sensor. True cross-fixes cannot occur in these cases, which are called “degenerate cross-fixes” because the information obtained from them is not specific enough to help locate the platform.

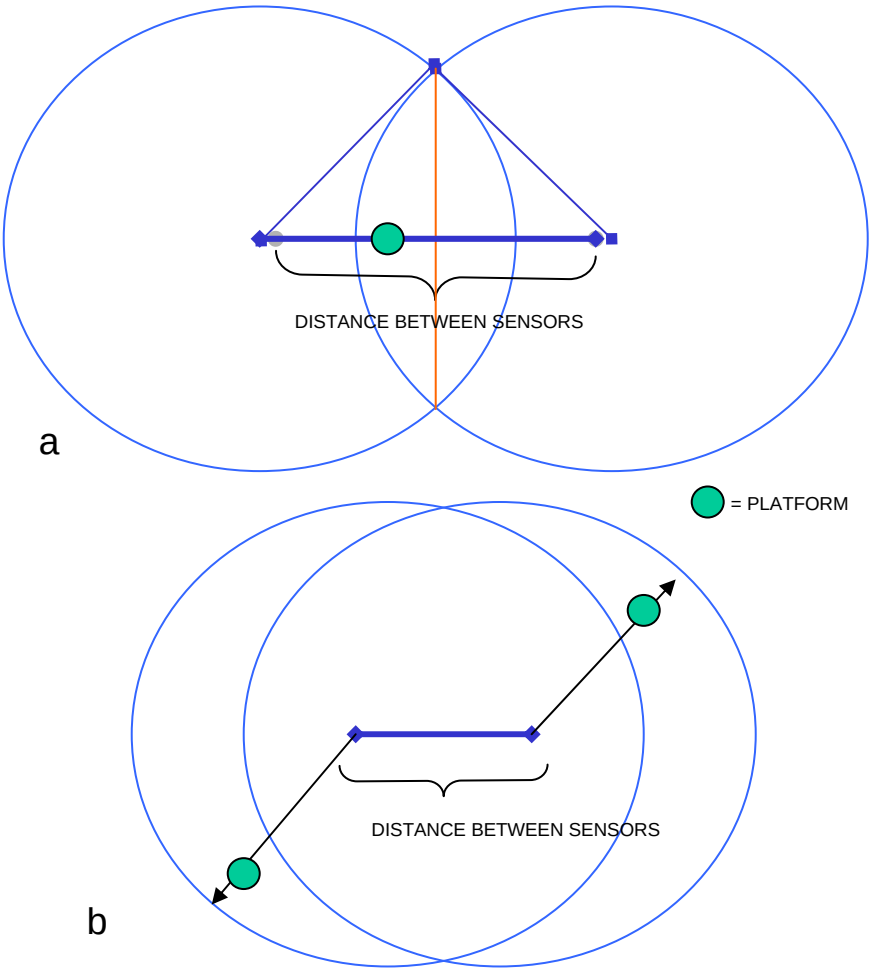


Figure 23. Examples of “degenerate cross-fixes.”

5. SENSOR ONTOLOGY STRUCTURE AND INTEGRATION

Sensors can be classified in various ways, such as by the physical property they detect, by whether they are active or passive, by their accuracy, by their availability, by their owner, etc. In an integrated sensor ontology, these various ways of classifying sensors are represented by multiple inheritance links (Ceruti & Wilcox, 2006). The sensor ontology also is documented in Swanson et al. (2007).

A survey of existing sensor ontologies reported in Ceruti et al. (2005) summarized some of the mappings for the ontologies found in the survey. Several ontologies were found but all were incomplete and no two ontologies were exactly alike. No single ontology included all of the concepts in sensor-data acquisition, fusion, interpretation, and usage, nor did any of the existing ontologies include all of the concepts of any other sensor ontology. Moreover, the ontologies found in the survey addressed primarily noun concepts and did not contain explicit references to verbs, with the possible exception of Cyc (Cycorp, 2002; Reed & Lenat, 2002), which covers some sensor-related verb concepts implicitly in its upper ontology. Some concepts that occur in one ontology also can occur in another ontology but at a different level, as shown for the hypothetical example in Figure 24 (Ceruti & Wilcox, 2006), where a concept at level 2 in ontology A also occurs in ontology B, but at level 3.

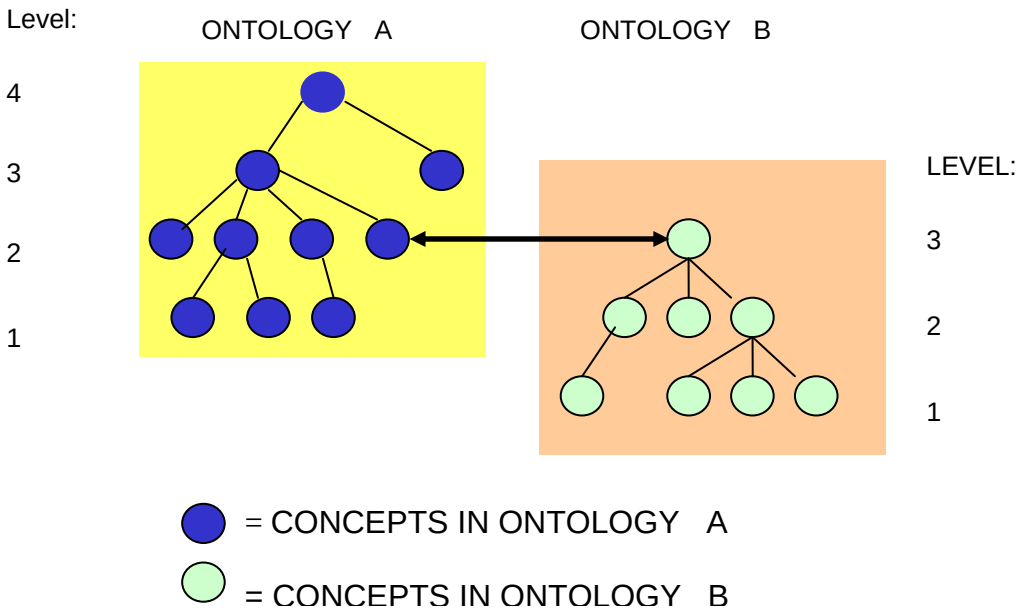


Figure 24. Example of a concept that occurs at different levels of abstraction in different ontologies (Ceruti & Wilcox, 2006).

Some noun concepts that pertain to sensor fusion were not found explicitly in any of the surveyed ontologies. For example, although ‘signal’ was found in three of the ontologies (Ceruti et al., 2005; Gao, 2000; and Reed & Lenat, 2002), characteristics that pertain to signals such as “frequency,” “period,” “wavelength,” “pulse-repetition rate,” “signal strength,” and “spectrum,” were not covered explicitly. Also, the concept of “noise” was not covered in the sensor ontologies, although it could be part of a more general, upper ontology. Concepts associated with

noise, such as “broadband” and “narrowband” were not found. The concept of a “propagation medium,” such as “air,” “water,” and “space” did not occur explicitly in any surveyed ontology, although, here again, it may be present in an upper ontology that does not pertain specifically to sensors. Depending on the placement of these concepts within the ontological structure, the specific data-fusion concepts will inherit characteristics from the higher and more general levels of abstraction (Ceruti & Wilcox, 2006).

In Figure 26 (Ceruti & Wilcox, 2006; Russomanno et al., 2005a and b), the various ontologies are as follows: “VIS Sensor Ontology” (Ceruti et al., 2005), Cycorp’s Cyc ontology for sensor concepts (Reed and Lenat, 2002), the Formal Information Fusion Framework (Gao, 2000), J. Hendler’s sensor ontology in DARPA Agent Markup Language (DAML) (Hendler, 2004; McGuinness et al.; 2002; and Bechhofer et al., 2004), and OntoSensor (Russomanno et al., 2005a and b). Because defense-related standards are based on concepts that pertain to sensors and their utilization, a complete integrated ontology will include concepts from these standards. For example, the ontology implicit in the standard Naval Tactical Display System Symbology (NTDS)[6] (Lead Standardization Activity, 1999; NATO MAS, 1990) has concepts that pertain to sensors and level-one sensor fusion. Other examples shown in Figure 25 include XML registry and the DoD Discovery Metadata Standard (DDMS) (DoD, 2007).

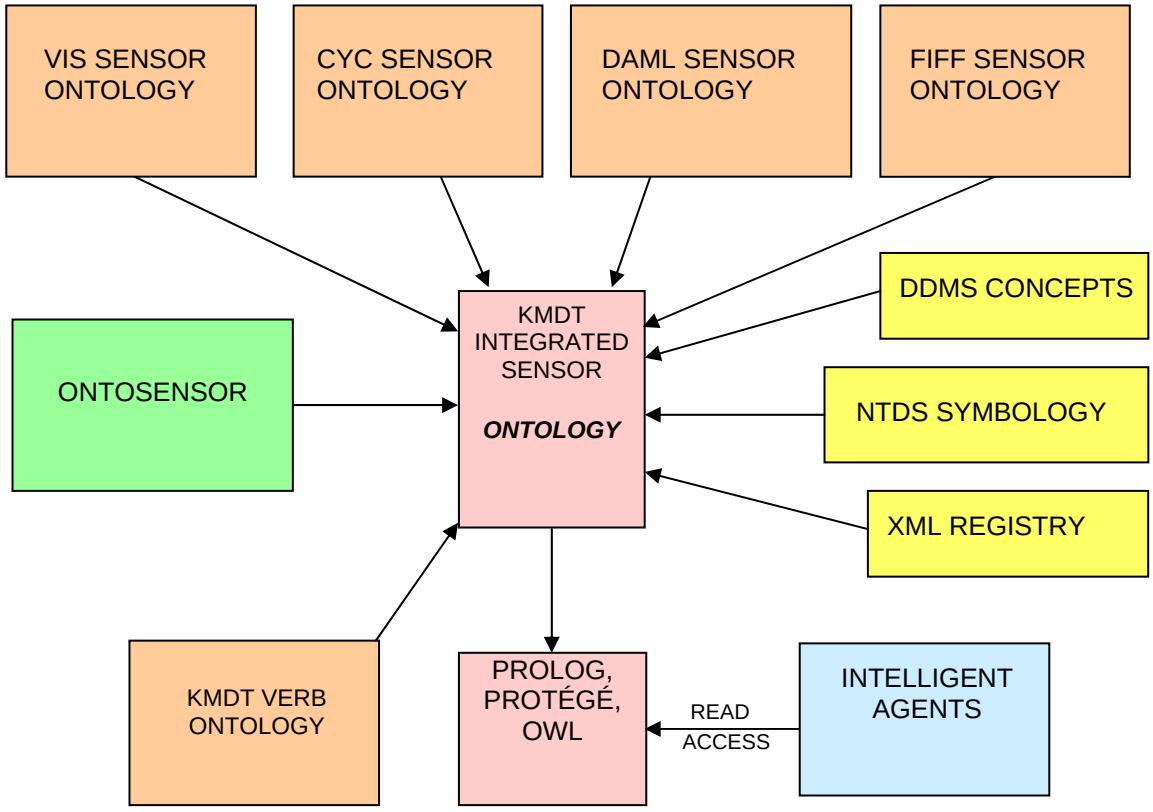


Figure 25. Sensor ontology and platform symbology integration (Ceruti & Wilcox, 2006; DoD, 2007; Russomanno et al., 2005a and b).

[6] Kathleen Fernandes. 2004. “MIL-STD-2525 Symbology Update. Private communication, SSC Pacific. Contact author for availability.

The concept of ontology-based data fusion is discussed in Llinas et al. (2004). Content ontology specifies concepts that include a description of what is believed to be the true nature of objects (Llinas et al., 2004). The content ontology, together with the states of interest associated with a task and the relationships within and between these states, are also are important parts of a sensor ontology. This ontology also includes concepts about the distinction between observations such as signals from sensors and the objects that give rise to these signals. Using a complete sensor ontology and database of sensor-performance characteristics, intelligent agents can obtain data available on the secure network more efficiently (Ceruti & Wilcox, 2006). The ontology also is the basis for the sensor-model design and the C&P database.

5.1 AGENT-ONTOLOGY INTERACTIONS

Top-level Java® classes for constructing KMDT agents were developed knowing that the agents' tasking would involve sensor data processing including, but not limited to, data retrieval, storage, and fusion. Thus, the concept of operations calls for the agents to interact with an integrated sensor ontology (Ceruti, 2004; Ceruti & Wilcox, 2006; Matheus et al., 2005) to coordinate their tasks directly or indirectly through databases designed according to the ontology (see the Section 5).

The ontology and associated databases document the sensor and platform characteristics and capabilities. They inform the agents of the appropriate sources of data for a particular task so the agent will search only for information relevant to the task. Agents might consult the ontology to identify classes of sensors that can detect a certain type of contact, thus limiting the subsequent search space to include only those sensor types. For example, it would be unproductive to search Web sites of space-based sensors for specific information to help identify or classify a contact when the signal in question is known to originate from a submarine.

The sensor ontology and associated databases also supply knowledge to the agents to aid subsequent data fusion. If two sensors of the same type (homogeneous sensors) detect the same signal, their signal signatures can often be compared directly. However, data fusion for dissimilar (heterogeneous) sensors is often more difficult because a direct comparison of the signal signatures is usually impossible. In this case, an ontology is especially useful. If an unknown contact is detected by disparate sensors, agents can use the ontology to identify classes of TOI that can be detected by each sensor type, whose intersection is likely to contain the unknown contact, thus aiding in subsequent classification.

As a simple example, suppose an underwater acoustic sensor detected some *unknown* contact at the same time a space-based electromagnetic sensor also detected an *unknown* contact. Knowledge extracted from the ontology might indicate that the likely acoustic contact was either an undersea or surface vessel, whereas the ontology might suggest that electromagnetic contact was a surface ship or an aircraft. Taken together, the intersection of these two sets of platforms would imply that the contact was a surface target if the respective detections were correlated.

6. RESULTS AND DISCUSSION

To illustrate the operation of the KMDT agents in localizing contacts from intersections of LOB data obtained from separate platform sensors, many simulations were conducted on a scenario in the East China Sea involving 8 target vessels and 11 passive acoustic sensors (Figure 2). During the execution of one simulation representing 72-hours, 400 contact reports (including false alarms) were generated as depicted in Figure 26. The results demonstrate the success of the KMDT M&S software, which generates both correct and false-alarm LOBs in the simulation process. This effect also is visible in Section 3 where the screen shots show considerable reduction in clutter when only the temporally feasible cross-fixes are displayed. The relevant figure pairs that show this effect are Figures 11 and 12 as well as Figures 15 and 16.

Figure 26 also demonstrates that KMDT can reduce the information overload by filtering out irrelevant information. In Figure 26a, all 400 contact LOBs are displayed. However, many LOB intersections do not represent valid target locations because they do not occur simultaneously (i.e., within an acceptable time window). In Figure 26b, many of these cross-fixes are filtered out by selecting only the LOBs observed within an appropriate temporal correlation interval, in this case 30 min. The 30-min time window corresponds to the time between steps in the simulation. The valid cross-fixes would be those that occurred in the same simulation step.

While the agents have filtered out temporally irrelevant data, the simulated data set may still contain false LOBs. The following test was conducted to determine the ability of agents to filter out false alarms indirectly. There were 10 different simulation runs that yielded a total of 5948 detection alerts, 906 of which were false alarms. On average, 3 percent of all the cross-fixes generated were temporally feasible. The number of false alarms in the cross-fixes that were temporally feasible was 42. The false alarms statistics were determined from the ground-truth output of the M&S software. The agents were not programmed to ingest or calculate any data on false-alarms. The elimination of the false alarms is an appropriate topic for future research (see the Section 7).

The agents can find, access, store, process, and display large quantities of information much faster than a human operator can. The results suggest that KMDT can improve the efficiency of detecting ships and submarines to reduce information overload for the operator. Agents can be deployed on the network to obtain additional information to corroborate or refute an hypothesis when the operator is in doubt about the initial results. Experience and research results to date suggest that the evolving KMDT technology can provide the military users improved future capabilities, such as reduced uncertainty in command centers and fewer target-selection errors. The distributed-agent architecture facilitates scalability.

Based on experience with the KMDT prototype, the KMDT approach appears to be able to increase the efficiency of contact-information acquisition and tracking. Moreover, the KMDT approach can facilitate tracking at a higher level of fidelity. Based on the literature review, the current technological gaps, and experimental observations with KMDT agents, distributed-contact tracking can be more productive using the results of the KMDT project. We have found that agents facilitate access to multiple information sources and specialized information.

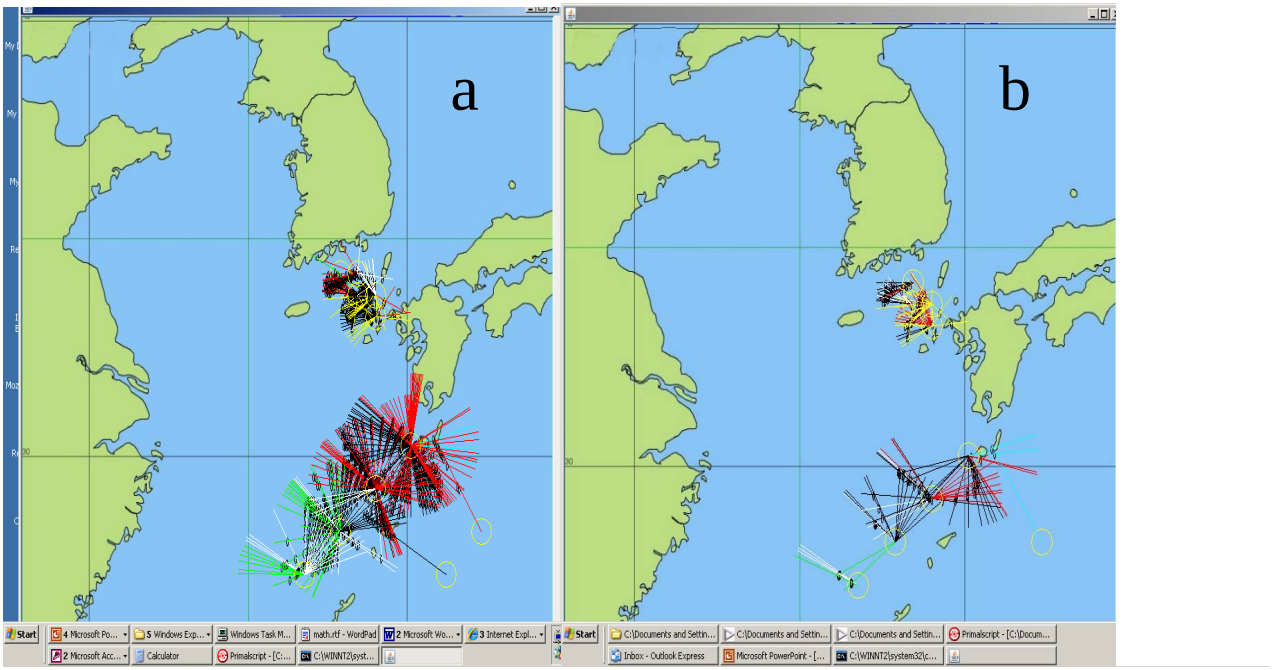


Figure 26. (a) Screen shot showing the results of 400 contact reports (LOBs) and agent computations collected during a simulated 72-hour interval, (b) screen shot showing only LOBs that cross within 30 min of each other as selected by the agents.

7. DIRECTIONS FOR FUTURE RESEARCH, DEVELOPMENT, AND APPLICATIONS

Based on the results of the current KMDT program, more work is needed to demonstrate the full capability level of KMDT and to advance the technology to a level where it can contribute to programs of record. Specific actions are as follows:

- Improve the integrated sensor ontology, incorporating recent (Goodwin and Russomanno, 2006; Niemann, 2007; Preece et al., 2007) and future (Graybeal, 2007) advances in sensor ontology research and development, sensor network prototype environment, sensor domains, and data types.
- Integrate more concepts, relationships, and standards into the KMDT integrated sensor ontology from, for example, OntoSensor (Russomanno et al., 2005a and b), and the Chemical Biological Radiological Nuclear, and Explosive (CBRNE) data model (Neimann, 2007).
- Configure intelligent, autonomous agents and ontologies to identify, acquire, and analyze track information.
- Demonstrate the validity of cross-fixes on the basis of acoustic signatures, explicit Temporal-Feasibility agent computations, and information from additional simulated heterogeneous sensors. Ensure that data sets returned to answer the operator's query contain only the cross-fixes resulting from correct and valid LOBs. Thus, the probability that false alarms will be flagged as valid cross-fix components will be reduced to an acceptably small value.
- Develop a flexible and operationally relevant human–computer interface supporting analytical and intuitive styles of queries. For example, the operator's query results will return in a graphical format like the screen shot depicted in Figure 27.
- Develop testing tools that verify that KMDT can transform and disseminate relevant information to the commander in sufficient time to act.
- Demonstrate reduced uncertainty in command decisions in operational scenarios based on a more efficient use of data from existing sensors that are better correlated and utilized in a net-centric environment.
- Automatically create a query retrieval cache memory that includes lists of the contacts, the queries executed, and any relevance evaluations.
- Develop software to enable multiple users to share the contact query results, thus permitting analysts synergistically to build on the work of others to create queries. This development could lead to more results than a single user could obtain or achieve working alone.
- Eliminate constraints that inhibit analysts working together even if they are distributed physically, temporally, and technologically. Team members can work offline and in parallel to improve their efficiency.

In the current design, agents acquire simulated message-level data from remote Web portals to be fused at the site that deployed the agents. In contrast, a future design could require the agents to process the message-level data from the Web portals remotely at the site from which the data are retrieved. This distributed architecture design reduces network bandwidth requirements. This processing method is designed to relieve overloaded operators tracking multiple unknown contacts and who may have deployed several agents to retrieve data on each one (Ceruti et al., 2005). Future simulations will include heterogeneous sensor models and active systems in addition to passive acoustic or electromagnetic sensors.

The KMDT architecture fits into the larger overall design of net-centric Web services for the warfighter. KMDT LOB correlations can be orchestrated to support individual users and their software agents in the net-centric environment where the next service utilization depends on the outcome of current services. Good orchestration requires good semantic understanding of services through ontologies (Daconta et al., 2003).

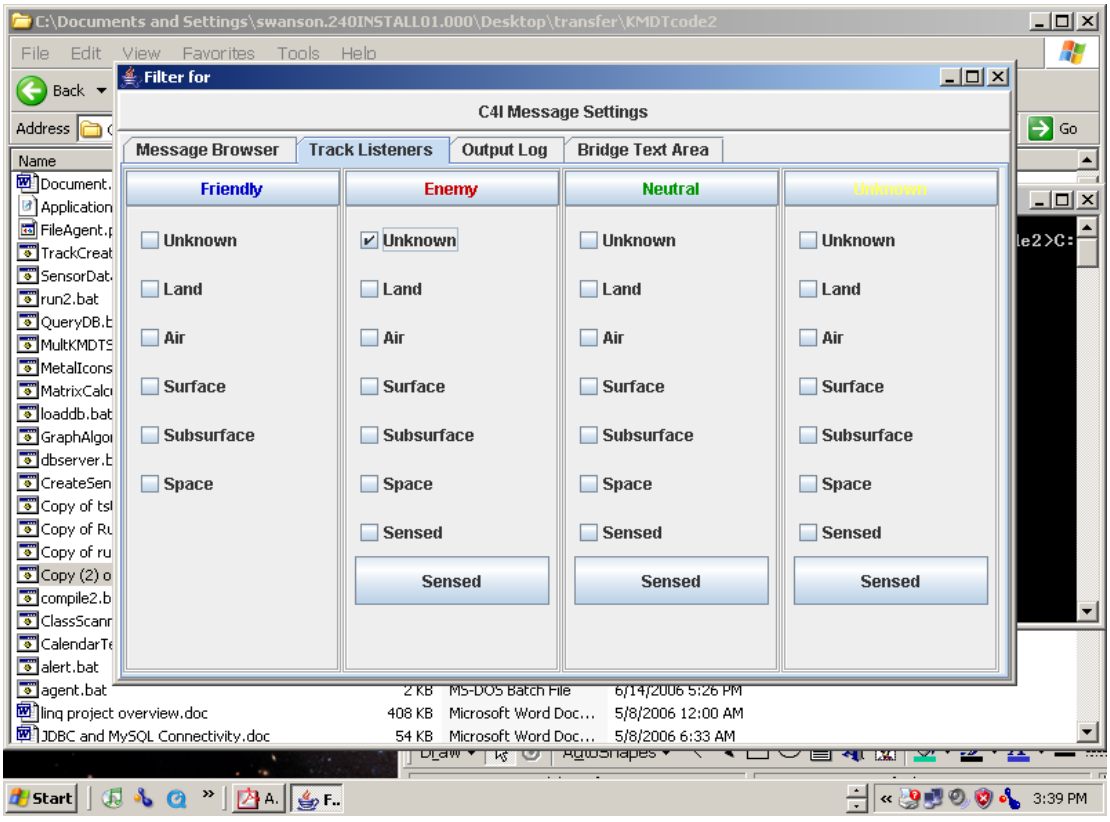


Figure 27. Screen shot depicting a query result as a future enhancement of the agent-user interface.

The authors believe that analysts will be more satisfied with the process and the product than analysts whose members individually track a contact without KMDT and then combine their results after they finish the analysis.

Our future research objectives are to refine and extend the design and implement empirical experiments to test our theories and our systems' architecture, interface, and functionality. Then, based on the results, we will refine the prototype to eventually develop a robust KMDT module that can be effectively integrated into network-centric and shipboard environments. If this proves to be successful, we also have plans to integrate multidimensional tracking and visualization functionality.

8. REFERENCES

Bechhofer, S., F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. 2004. *OWL Web Ontology Language Reference*: 1–42 (February 2004). http://www.w3.org/TR/2004/REC-owl-ref-20040210

Ceruti, M. G. 2001. "Mobile Agents in Network-Centric Warfare," *Institute of Electronics Information and Communication Engineers Transactions on Communications E84-B* (10): 2781–2785.

Ceruti, M. G. 2004. "Ontology for Level-One Sensor Fusion and Knowledge Discovery." *Proceedings of the 2004 International Knowledge Discovery and Ontology Workshop (KDO-2004)* (pp. 20–24) September 2004, Pisa, Italy.

Ceruti, M. G. and B. J. Powers. 2004. "Intelligent Agents for FORCEnet: Greater Dependability in Network-Centric Warfare." *Supplemental Volume of the Proceedings of the IEEE International Conference on Dependable Systems and Networks (DSN 2004)* (pp. 67–75) June 2004, Florence, Italy.

Ceruti, M. G., D. R. Wilcox, and B. J. Powers. 2004. "Knowledge Management for Command and Control." *Proceedings of the Command and Control Research and Technology Symposium, CCRTS'04,* Paper 017, June 2004, Coronado, CA.

Ceruti, M. G. and D. R. Wilcox. 2006. "Sensor Ontology Integration for the Knowledge Management for Distributed Tracking (KMDT) Program." *Proceedings of the 11th Command and Control Research and Technology Symposium (CCRTS 06),* June 2006, San Diego, CA. http://www.dodccrp.org/events/2006_CCRTS/html/papers/072.pdf

Ceruti, M. G. and T. L. Wright. 2005. "Knowledge Management for Distributed Tracking and the Next-Generation Command and Control," IEEE International Software Metrics Symposium (*METRICS-2005*) Industry Track (pp.1–4), September 2005, Como, Italy. http://metrics2005.di.uniba.it/IndustryTrack/Ceruti.KMDTMETRICS2005.pdf

Ceruti, M. G., T. L. Wright, D. R. Wilcox, and S. C. McGirr. 2005. "Modeling and Simulation for the Knowledge Management for Distributed Tracking (KMDT) Program." *Proceedings of the International Workshop on Modeling and Applied Simulation (MAS-2005)* (pp. 67–75), October 2005, Bergeggi, Italy.

Clark, V. ADM, USN. 2002. "Sea Power 21 Series–Part I: Projecting Decisive Joint Capabilities." *Naval Institute Proceedings*, *128* (10): 32–41.

Cycorp. 2002. *Predicates and Denotational Functions*. http://72.14.203.104/search?q=cache:IUnMIbkutwwJ:www.cyc.com/doc/tut/DnLoad/Arity.pdf+arity&hl=it&gl=us&ct=clnk&cd=17

Daconta, M. C, L. J. Obrst, and K. T. Smith. 2003. *The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management,* pp. 72–75. Wiley Publications. Indianapolis, Indiana.

Demazeau, Y. and Jean-Pierre Müller. 1989. "Decentralized Artificial Intelligence." *Proceedings of the 1st European Workshop on Modeling Autonomous Agents in a Multi-Agents World*, pp. 3–13, Elsevier Science Press, Cambridge, England.

Department of Defense. 2007. "Department of Defense Discovery Metadata Specification (DDMS)." 1.4.1, 1 July 2007, http://metadata.dod.mil/mdr/irs/DDMS.

Gao, H. 2000. *Formal Information Fusion Framework (FIFF)*. Information Fusion Group. December 2000. http://www.ece.neu.edu/groups/ifg/

Goodwin, C. and D. J. Russomanno. 2006. *An Ontology-Based Sensor Network Prototype Environment*. www.cs.virginia.edu/~ipsn06/WIP/goodwin_1568983444.pdf

Graybeal, J. 2007. "Sensor Ontology Work Begins," *The Marine Technology Reporter*, September 2007. http://www.mtronline.net/mt/mtStories.aspx?ShowStrory=1013382087

Lead Standardization Activity (LSA). 1999. "Department of Defense Interface Standard Common Warfighting Symbology." MILSTD2525B. Center for Standards (CFS), Reston, VA.

Llinas, J., C. Bowman, G. Rogova, A. Steinberg, E. Waltz, and F. White. 2004. "Revisiting the JDL Data Fusion Model II." *The 7th International Conference on Information Fusion (FUSION 2004)*, June 2004.

Hendler, J. 2004. *DAML Sensor Ontology*. 29 June 2004. http://www.mindswap.org/~evren/services/sensor-jpa.daml

Matheus, C. J., D. Tribble, M. M. Kokar, M. G. Ceruti, and S. C. McGirr. 2005. "Towards a Formal Pedigree Ontology for Level-One Sensor Fusion." *Proceedings of the 10th International Command and Control Research and Technology Symposium,* McLean, VA. http://vistology.com/papers/Pedigree-ICCRTS-05.pdf

McGirr, S. C. 2001. "Resources for the Design of Data Fusion Systems." *Proceedings of the ISIF Fusion 2001 Conference on Information Fusion*, August 2001, Montreal, Canada.

McGuinness, D. L., R. Fikes, J. Hendler, and L. A. Stein. 2002. "DAML+OIL: An Ontology Language for the Semantic Web." *IEEE Intelligent Systems 17*(5): 72–80.

Niemann, B. L. 2007. *Harmonization of Sensor Standards in Semantic Wikis: Sensor Standards Harmonization Working Group Meeting, Sensor Network Pilots For DRM 2.0 11, 12, 34, 35*, 46–49.

North Atlantic Treaty Organization (NATO) Military Agency for Standardization (MAS). 1990. "Draft Standardization Agreement Subject: Display Symbology and Colours for NATO Maritime Units." STANAG 4420 Edition 1 (Feb. 1990). http://colab.cim3.net/file/work/SICoP/2007-04-25/PMirhaji04252007.ppt

Preece, A., M, Gomez, G, de Mel, G, W. Vasconcelos, D. Sleeman, S. Colley, and T. LaPorta. 2007. "An Ontology-Based Approach to Sensor Mission-Assignment." *Proceedings of the 1st Annual Conference of the International Technology Alliance (ACITA 2007)*:19 http://www.csd.abdn.ac.uk/~apreece/publications/download/acita2007a.pdf http://users.cs.cf.ac.uk/A.D.Preece/publications/download/acita2007a.pdf

Reed, S. L. and D. B. Lenat. 2002. "Mapping Ontologies into Cyc." *Proceedings of the AAAI 2002 Conference Workshop on Ontologies for the Semantic Web*, July 2002, Edmonton, Canada.

Russomanno, D. J, C. R. Kothari, and O. A. Thomas. 2005a. "Sensor Ontologies: From Shallow to Deep Models." ieeexplore.ieee.org/iel5/9879/31411/01460887.pdf

Russomanno, D. J, C. R. Kothari, and O. A. Thomas. 2005b. "Building a Sensor Ontology: A Practical Approach Leveraging ISO and OGC Models." *Proceedings of the 2005 International Conference on Artificial Intelligence (ICAI 2005)* pp. 637–643, 2730 June Las Vegas, Nevada, USA, June 27-30, 2005, Volume 2. CSREA Press.

Swanson, D., T. L. Wright, M. G. Ceruti, D. R. Wilcox, and S. C. McGirr. 2007. “Design, Development, and Reuse of Software Agents in the Knowledge Management for Distributed-Tracking (KMDT) Program.” *Proceedings of the 2nd International Workshop on Ontology, Conceptualization, and Epistemology for Software & Systems Engineering (ONTOSE 2007),* Milan, Italy.

Thorp, W. H. 1967. “Analytic Description of the Low Frequency Attenuation Coefficient,” *Journal of the Acoustical Society of America 42*: 270.

Waltz, E. and J. Llinas. 1990. *Multisensor Data Fusion*. Artech House, Boston, MA.

.

APPENDIX A
HISTORY OF KMDT SIMULATION DEVELOPMENT

Version 0

- Map of theater of operations
    - Coastline database
    - Bathymetry database
- Map cursor for extracting geographic information
- Bathymetric contours

Version 1

- Multiple targets
    - Narrowband source-level spectrum acoustic signatures
- Quasi-random target motion model
    - Based on normal distributions of speed and heading
    - Land avoidance algorithm
- Multiple (fixed) sensors
- Passive acoustic sensor model
    - Transmission Loss model (spherical spreading + absorption)
    - Ambient Noise model
- Probability of Detection model (range dependent)
- Keyboard interaction with drop-down menu options:
    - Advance time
    - Show/hide detection log
    - Show/hide sensor coverage
    - Show/hide map cursor, geographic information
    - Quit simulation

Version 2

- Scenarios read from separate script files
- Detection alerts/false alarms written to separate text files
- Menu option to dynamically change simulation time-step
- Additional display and interface modifications

Version 3

- Graphical improvements/enhancements
    - Sensor coverage display
    - Lines-of-bearing alert displays
    - Restart menu option
- Detection log filenames specified through scenario file
- "Fuzzification" of reported lines of bearing

Version 4

- Added interactive control via mouse and cursor

Version 5

- Waypoint motion model for mobile sensors
- Doppler-shifted frequencies due to target/sensor motion
- Added sensor identification labels on display

Version 6

- Target activation delayed until specified elapsed time
- Propagation time delay added to reported elapsed time
- Reported frequency resolution increased to show Doppler effect
- Transmission Loss model extended to include cylindrical spreading loss
- GUI improvements
    - Individual sensor identification and maximum detection range coverage displayed via cursor

Planned Improvements for Transition Programs

- Modify code to accommodate aircraft
    - Modify quasi-random motion model to suppress land avoidance restriction
    - Modify quasi-random motion to specify a minimum speed for fixed-wing aircraft
    - Add altitude profile to motion models
- Add additional sensor models
    - Passive electromagnetic
    - Passive electro-optic
    - Add line-of-sight detection range computation
- Improve acoustic Transmission Loss model
    - Sound speed environmental database
    - Surface Duct model
    - Sound Channel model
    - Convergence Zone model
- Improve Source Level specification
    - Narrowband model
    - Broadband model
- Incorporate bathymetry contour display option
- Incorporate screen refresh algorithm
    - Video memory management
    - Chained code segment overlays
    - Separate video pages (EGA)
- Build capability to specify different theaters of operation
    - Specification via scenario file or separate interactive module
    - Algorithm(s) to automate selection of maneuver polygon (use coarse resolution coastline database?)
- Build non-graphical version of the simulation to facilitate Monte-Carlo analyses
- Port code to Visual Basic
    - More memory/better memory management
    - Better graphics/resolution
    - Microsoft Windows®-like GUI

APPENDIX B
FORMATS FOR SCENARIO FILES

Scenario file records consist of ASCII character strings containing variable-length fields delimited by commas. All records are terminated by ENTER or RETURN keyboard control characters. The records are described below:

Record 1 (1 field) – filename of composite detection log file, enclosed in double quotes
Record 2 (1 field) – number (i) of sensors in scenario
Records 3 thru i+2 – sensor definitions (one record for each sensor):
- Field 1 – filename of detection log file for sensor, enclosed in double quotes
- Field 2 – prob. of false alarm deemed acceptable for sensor (expressed as a %)
- Field 3 – initial latitude of sensor (decimal degrees; positive for North latitudes, negative for South latitudes)
- Field 4 – initial longitude of sensor (decimal degrees; positive for East longitudes, negative for West longitudes)
- Field 5 – shipping noise level offset at sensor location (between ±10 dB re 1 μPa; positive for heavy shipping, negative for light shipping)
- Field 6 – sea state at sensor location (between 0 (calm seas) and 6 (high seas))
- Field 7 – array gain for sensor (dB re 1 μPa)
- Field 8 – standard deviation for distribution of LOB error (degrees)
- Field 9 – number (l) of following pairs of fields defining sectors of maximum detection range coverage for sensor ($1 \le l \le 5$)
- Fields 10 [, 12, 14, …, 10+2(l–1)] – compass bearing for CCW side of sector (degrees CW from North)
- Fields 11 [, 13, 15, …, 11+2(l–1)] – maximum detection range of sector (nautical miles)
- Field k = 10+2l – number (m) of following triplets of fields defining track legs for mobile sensors (m = 0 for fixed sensors; $1 \le m \le 10$ for mobile sensors)
- Fields [k+1, k+4, …, k+1+3(m–1)] – latitude of endpoint of track leg (decimal degrees; positive for North latitudes, negative for South latitudes). Latitude of beginning of track leg coincides with that of previous leg endpoint (or initial position of sensor for first leg)
- Fields [k+2, k+5, …, k+2+3(m–1)] – longitude of endpoint of track leg (decimal degrees; positive for East longitudes, negative for West longitudes). Longitude of beginning of track leg coincides with that of previous leg endpoint (or initial position of sensor for first leg)
- Fields [k+3, k+6, …, k+3+3(m–1)] – speed of sensor platform in track leg (knots)

Record i+3 (1 field) – number (j) of targets in scenario
Records i+4 thru i+j+3 – target definitions (one record for each target):
- Field 1 – target start time (decimal hours; -1 = non-active target)
- Field 2 – initial latitude of target (decimal degrees; positive for North latitudes, negative for South latitudes)
- Field 3 – initial longitude of target (decimal degrees; positive for East longitudes, negative for West longitudes)
- Field 4 – initial speed of target (knots)
- Field 5 – target speed standard deviation (knots)

Field 6 – maximum speed of target (knots)
Field 7 – initial heading of target (degrees CW from North)
Field 8 – target heading standard deviation (degrees)
Field 9 – target classification, enclosed in double quotes (“F”=friendly, “H”=hostile, “N”=neutral, “U”=unknown)
Field 10 – number (n) of following pairs of fields defining narrowband components of signal emitted from target ($1 \le n \le 10$)
Fields 11 [, 13, 15, …, 11+2(n–1)] – frequency of signal component (Hz)
Fields 12 [, 14, 16, …, 12+2(n–1)] – source level of signal component (dB re 1 μPa)

The following example shows the contents of a typical scenario file with 13 sensor platforms and 15 targets. The fields of typical sensor and target records are illustrated in Figures B-1 and B-2, respectively.

```
"KMDTLOG.TXT"
13
"APF01.TXT", 2, 34.0, 128.0,  5, 3, 13, 1, 1, 000,  50, 0
"APF02.TXT", 2, 34.0, 128.5,  0, 4, 13, 1, 1, 000,  50, 0
"APF03.TXT", 2, 33.5, 128.5,  0, 4, 13, 1, 1, 000,  50, 0
"APF04.TXT", 2, 33.5, 129.0,  0, 4, 13, 1, 1, 000,  50, 0
"APF05.TXT", 2, 33.0, 129.0,  5, 3, 13, 1, 1, 000,  50, 0
"APF06.TXT", 2, 27.0, 127.0,  0, 2, 15, 1, 2, 090,  50, 150, 100, 0
"APF07.TXT", 2, 28.0, 128.0,  0, 2, 15, 1, 4, 070,  60, 090, 100, 160,  60, 190, 100, 0
"APF08.TXT", 2, 29.0, 129.0,  0, 2, 15, 1, 2, 130,  40, 180, 100, 0
"APF09.TXT", 2, 30.0, 130.0,  5, 2, 15, 1, 2, 000,  70, 045, 100, 0
"APF10.TXT", 2, 27.0, 131.0,-10, 1, 15, 1, 4, 000, 150, 250, 115, 265,  85, 345, 115, 0
"APF11.TXT", 2, 28.0, 132.0, -5, 1, 15, 1, 4, 000, 150, 250, 125, 265, 100, 345, 125, 0
"APM12.TXT", 2, 35.0, 125.0,  0, 1, 12, 1, 1, 000,  50, 2, 31.5, 125.0, 5, 35.0, 125.0, 5
"APM13.TXT", 2, 34.5, 125.0,  0, 1, 12, 1, 1, 000,  50, 2, 31.0, 125.0, 5, 34.5, 125.0, 5
15
 0, 36.0, 124.0, 15, 2, 30, 160,  5, "H", 3,  60, 135, 120, 135, 300, 125
 0, 26.0, 131.0, 10, 3, 20, 315, 10, "U", 1, 120, 140
 0, 36.0, 132.0, 10, 2, 25, 210,  5, "U", 2, 100, 145, 150, 140
 0, 31.0, 132.0, 10, 2, 20, 225,  5, "U", 3,  60, 150, 120, 155, 180, 145
 0, 32.0, 133.0, 15, 2, 30, 225,  5, "F", 3, 100, 140, 125, 140, 150, 140
 0, 26.0, 123.0, 10, 2, 20, 060,  5, "U", 2, 120, 155, 150, 150
 0, 32.0, 127.0, 10, 2, 15, 045,  5, "U", 3,  60, 140, 120, 150, 180, 145
 6, 31.5, 121.5, 10, 2, 20, 090,  5, "U", 2, 100, 150, 120, 145
 6, 33.0, 135.0, 15, 2, 30, 240,  3, "U", 2,  60, 140, 100, 140
12, 32.0, 130.0,  8, 2, 15, 200, 10, "U", 2,  80, 150, 110, 145
18, 33.0, 135.0, 15, 2, 25, 240,  3, "U", 2,  90, 150, 120, 145
18, 25.5, 121.5, 12, 2, 30, 045,  3, "U", 1, 130, 145
24, 35.0, 129.0, 12, 3, 25, 180, 10, "U", 2, 100, 150, 130, 150
30, 31.5, 121.5, 10, 3, 15, 090, 10, "U", 2, 100, 155, 150, 155
36, 27.0, 128.0, 10, 2, 25, 030,  5, "F", 1, 100, 135
```

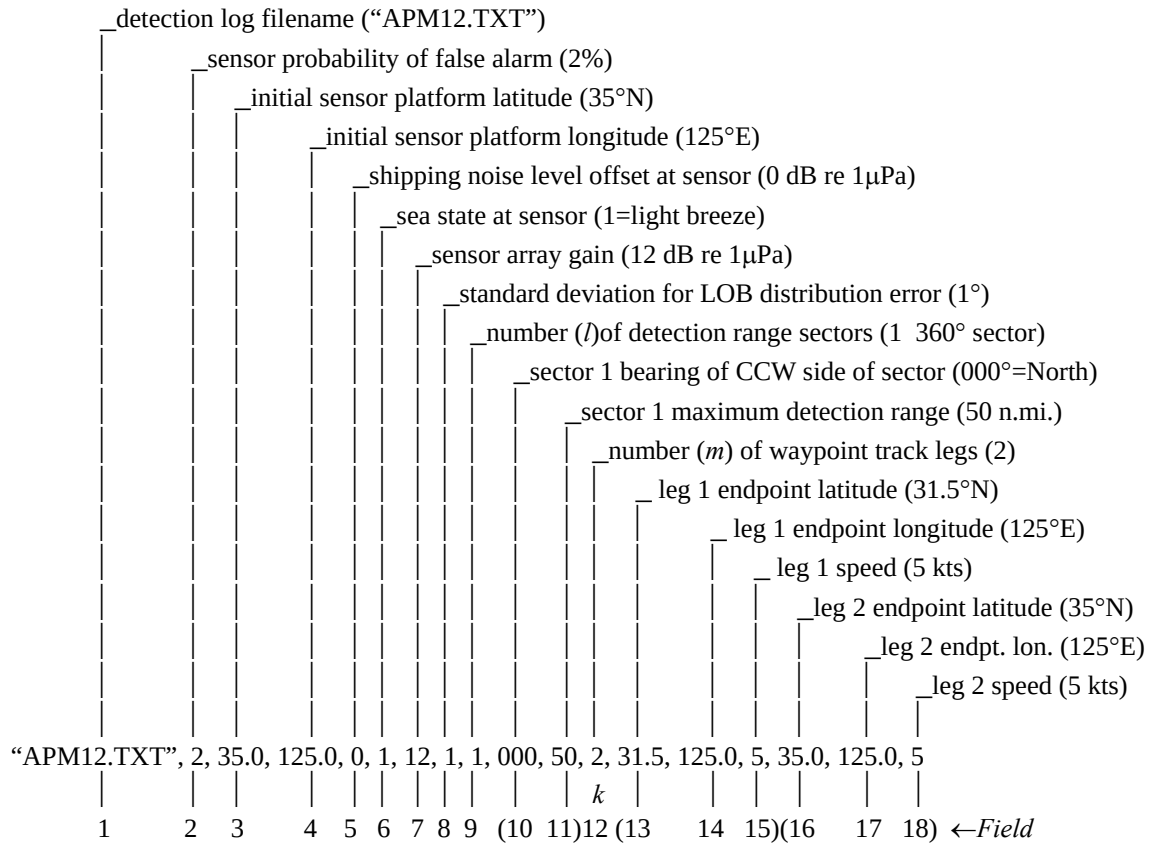


Figure B-1. Typical sensor record specification (with a two-leg “picket line” platform waypoint track).

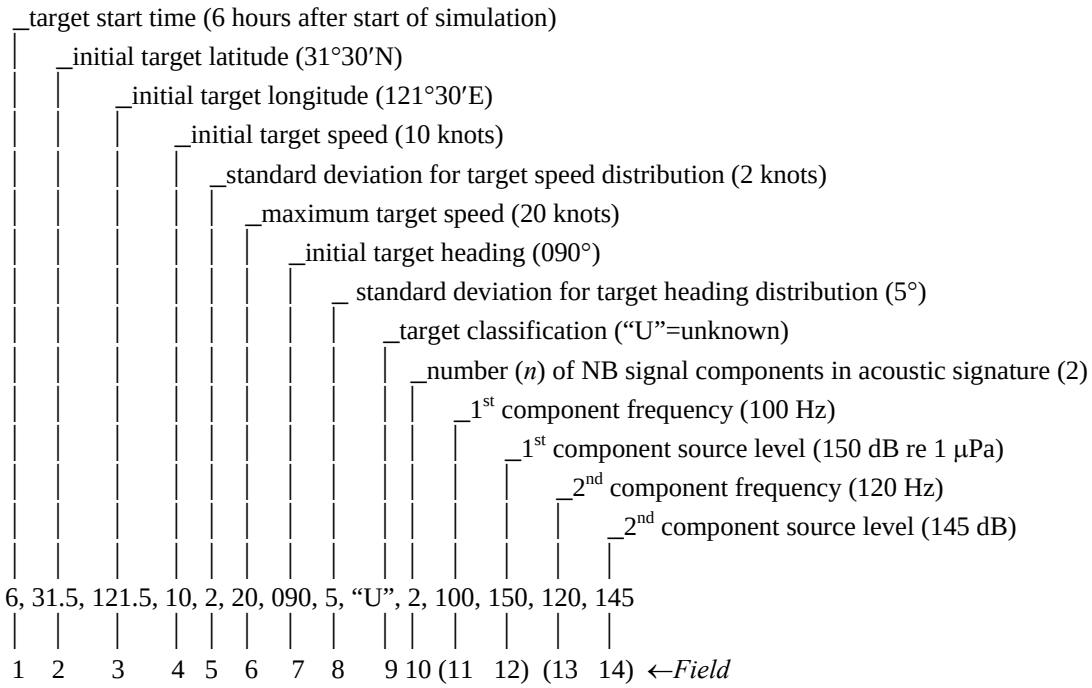


Figure B-2. Typical target record specification (with two narrowband signal components).

APPENDIX C
FORMATS FOR DETECTION LOG FILES

The names of detection log output files are specified as part of the scenario files (Appendix B). Detection log file records consist of ASCII character strings containing variable-length fields delimited by commas. Each record contains detection information for a single detection or false alarm. All records contain the same text fields. A detection record can contain up to 10 frequency component fields; if fewer than 10 frequency components were detected, the remainder of the 10 frequency fields are blank (null). The final three fields of each record contain ground truth information appended for verification purposes. All records are terminated with a semi-colon (;). The field descriptions are as follows:

Field 1 – sensor identifier
Field 2 – time of alert/false alarm (elapsed time in decimal hours from beginning of simulation) Note: alert time includes any signal propagation time to sensor
Field 3 – latitude of sensor (decimal degrees; positive for North latitudes, negative for South latitudes)
Field 4 – longitude of sensor (decimal degrees; positive for East longitudes, negative for West longitudes)
Field 5 – line-of-bearing to target (degrees CW from North) Note: this LOB is not ground truth, but a "fuzzified" approximation derived from an error distribution about the actual LOB
Fields 6-15 – signal frequency components (Hz) Note: these frequencies are those of detected signal components only, and are Doppler shifted to account for any target/sensor platform relative motion
Field 16 – maximum sensor detection range (nautical miles) in the direction of the target. Note: this range is determined from the sensor coverage specified in the scenario file
Field 17 – ground truth latitude of target (decimal degrees; positive for North latitudes, negative for South latitudes) Note: field is empty if alert is a false alarm
Field 18 – ground truth longitude of target (decimal degrees; positive for East longitudes, negative for West longitudes) Note: field is empty if alert is a false alarm
Field 19 – classification of target. Note: field contains "FA" if alert is a false alarm

The following example shows the partial contents of a typical composite detection log file. The fields of a typical detection log record are illustrated in Figure C-1.

```
APF09,4.03,30.00,130.00,69.3,60.1,120.3,180.4,,,,,,,100,30.51,131.60,U;
APF10,4.02,27.00,131.00,207.2,119.9,,,,,,,,,150,26.33,130.66,U;
APF10,4.02,27.00,131.00,101.6,295.9,,,,,,,,,150,,,FA;
APF09,4.53,30.00,130.00,70.1,60.2,120.4,180.6,,,,,,,100,30.46,131.56,U;
APF10,4.52,27.00,131.00,211.2,119.9,,,,,,,,,150,26.36,130.56,U;
APM12,4.52,34.63,125.00,308.0,60.1,120.3,300.6,,,,,,,50,35.10,124.21,H;
APF09,5.03,30.00,130.00,72.7,60.2,120.4,180.6,,,,,,,100,30.37,131.50,U;
APF10,5.02,27.00,131.00,217.2,119.9,,,,,,,,,150,26.37,130.49,U;
APM12,5.02,34.58,125.00,300.5,,120.3,300.9,,,,,,,50,34.98,124.21,H;
APF06,5.52,27.00,127.00,270.5,200.4,,,,,,,,,100,,,FA;
APF09,5.53,30.00,130.00,77.4,60.2,120.5,180.7,,,,,,,100,30.27,131.44,U;
APF10,5.52,27.00,131.00,219.8,119.8,,,,,,,,,150,26.40,130.41,U;
APF10,5.53,27.00,131.00,101.6,261.4,,,,,,,,,150,,,FA;
APM12,5.51,34.54,125.00,295.4,60.2,120.3,300.8,,,,,,,50,34.84,124.24,H;
```

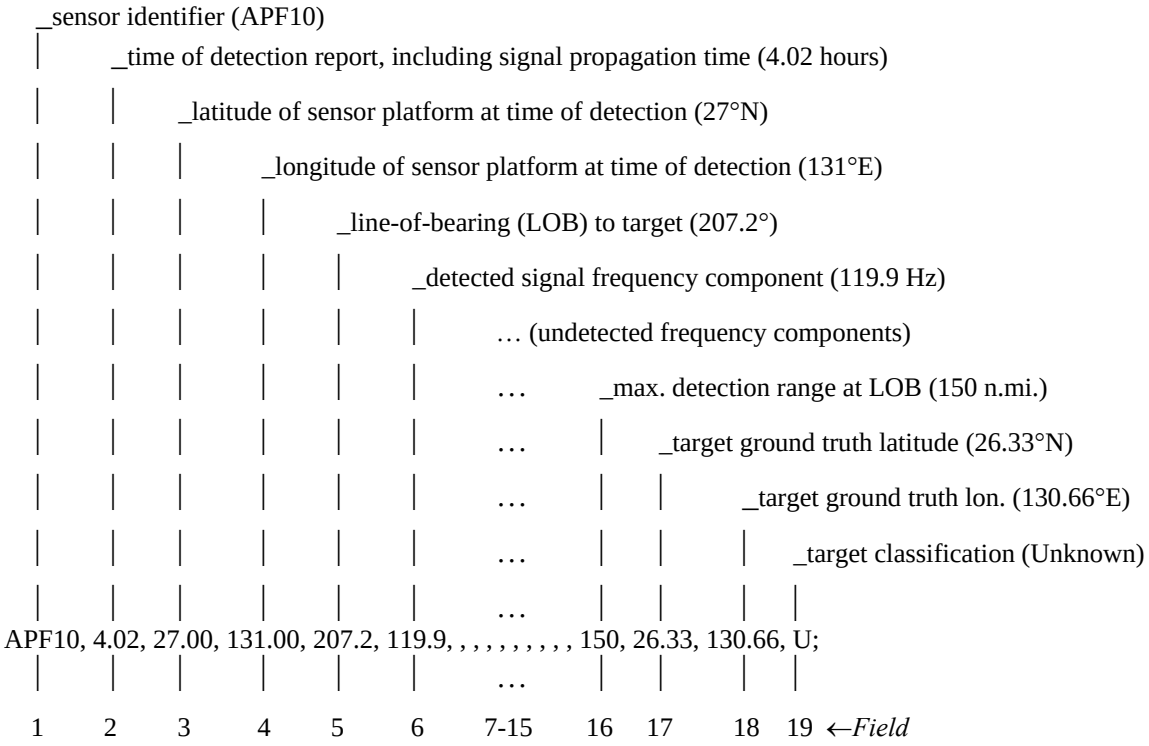


Figure C-1. Typical detection log record (with one detected frequency component).

APPENDIX D
OPERATING INSTRUCTIONS FOR KMDT SIMULATION (VERSION 6)

Introduction. The purpose of the KMDT simulation module is to generate contact information in the form of lines of bearing (LOBs) from passive sensor systems to potential targets distributed in some theater of operations. Sensor systems can be geographically fixed or mobile, as for example, a moored or towed acoustic array of hydrophones. The targets (with pre-designated signal signatures) are propagated through the theater using a quasi-random motion model. At every time-step during the simulation evolution, a determination is made of those targets detected by each sensor system using a range-dependent probability of detection computed for each target-sensor pair. False alarms are also generated, and along with detection alerts, are entered into appropriate "detection log" files for access by external software agents. The attributes and initial conditions of the targets and sensors for a particular scenario are specified through a "scenario" file. See Appendices B and C for more information on the contents of these files. Instructions for running the simulation are presented as follows:

Starting the simulation. Before starting the simulation, make sure all necessary files are in the same folder as the program file, KMDT6.EXE. The required files are as follows:

- KMDT6.EXE -the simulation executable program file
- BRUN45.EXE -a required run-time module
- CIL3.PNT -coastline coordinates database file
- MAPFONT.DAT -map font database file
- TOPODATA.DAT -topographic elevations database file
- Any scenario file(s) to be run

Output detection log files generated during the simulation will be created in the same folder. These include a composite log of detection alerts from all sensors and individual logs of detection alerts for each sensor. Detection log files are assigned names via the scenario file.

To start the simulation, do either of the following:

a. Open the folder containing the KMDT simulation files and double-click on the file KMDT6.EXE. This method will then prompt for the name of a scenario file that must be entered manually. Remember to include any filename extension when entering the scenario filename.
b. Double-click on the desktop icon "Shortcut to KMDT6.EXE", if present. This method does not require typing the name of a scenario file, but only one pre-designated scenario file, specified in the program command line for the shortcut, may be run. The name of the pre-designated scenario file may be changed by right clicking on the icon, opening the Properties dialog box, selecting the Program tab, and editing the entry in the window next to "Cmd line" by replacing the scenario file name following the program name KMDT6.EXE at the end of the entry.

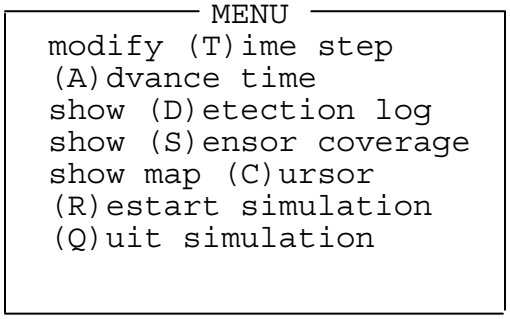
Running the simulation. After program invocation, a geographic map of the theater of operations will appear with a mouse-activated "arrow" cursor. Subsequent interaction with the simulation may be performed either by using the mouse or the keyboard. At the top of the screen is an information/ command line illustrated as follows:

```
TIME =   0.00 hr   (A)dvance time     (M)enu
```

The "`TIME =`" field shows the elapsed time of the simulation (in decimal hours), and is incremented at each time-step of the simulation. The time may be incremented in discrete time steps by clicking the left mouse button while the arrow cursor is over the "`(A)dvance time`" field, or by

depressing and releasing the "A" key on the keyboard. After release of the left mouse button or "A" key the simulation is paused. Alternately, the simulation may be run continuously by holding down the left mouse button or "A" key. Note that when a target detection or false alarm is declared after a time step, a detection alert window appears along with LOBs from any sensors detecting the target. For discrete updating, clicking anywhere on the screen or pressing any keyboard key will clear the alert window and associated LOBs; for continuous updating, the alert window and associated LOBs will momentarily flash and the simulation will continue. While the simulation is paused between discrete updates, the arrow cursor may be moved over the + symbol denoting the location of a sensor to display its identification tag and maximum detection range coverage.

Clicking on the "(M)enu" field (or depressing the "M" key) will open a menu window, as illustrated below:



The menu options are activated by left-clicking the mouse while the arrow cursor is over the desired option field (or alternately by depressing the keyboard key indicated in parentheses). The menu options are described as follows:

modify (T)ime step Activating this option displays a window prompting the entry of a new time-step interval. Enter a numeric value in decimal hours (e.g., 12 min would be entered as 0.2 hours) and press ENTER on the keyboard. Subsequent time-steps in the simulation will then occur using the new time interval. The default time interval is 0.5 hours (30 min).

(A)dvance time Activating this option advances the simulation one time-step as described earlier.

show (D)etection log Activating this option displays a window containing information about the 10 most recent detection alerts. Clicking anywhere or pressing any keyboard key will dismiss the window.

show (S)ensor coverage Activating this option displays the area of coverage of each sensor out to its maximum detection range (the azimuth-dependent detection range for each sensor is specified in the scenario file). Clicking anywhere or pressing any keyboard key will clear the coverage display.

show map (C)ursor Activating this option will display a "crosshair" map cursor on the screen, along with additional geographic information. At the top right of the screen, an "ELEVATION =" field will appear for displaying the land height above sea-level (+ meters) or sea depth below sea-level (– meters) of the cursor position. At the bottom of the screen, "LAT =" and "LON =" fields will appear for displaying the latitude and longitude (in decimal degrees) of the cursor position, along with "RANGE =" and "AZIMUTH =" fields for a geodesic (i.e., great circle) range (in nautical miles) and compass bearing (in degrees clockwise from North) between two positions on the map. Values for these latter two fields appear only when the mouse is dragged while

depressing the left mouse button to move the cursor from a “departure” to a “destination” location; a corresponding great circle line from departure to destination is also plotted. This line can be erased by clicking the right mouse button, *but only if, subsequent to drawing the line, the right mouse button is clicked before any other mouse button or keyboard key*. Note that the displayed azimuth is the initial bearing from departure to destination at the departure location, as this bearing will vary along the geodesic. While utilizing the map cursor mode, the left and right mouse buttons function as follows: Clicking the left mouse button enables discrete updating of the geographic data at the current position of the cursor; subsequent updating occurs only during subsequent mouse clicks. Clicking the right mouse button enables continuous updating of geographic data as the cursor is moved without subsequent mouse clicks. To exit the map cursor mode press any keyboard key while the updating is continuous, or if updating is discrete, press any key followed by a mouse click. The cursor will then revert to an arrow cursor.

`(R)estart simulation` Activating this option will restart the simulation using the same scenario file. Previous target tracks are cleared and targets and sensors parameters are reset to their initial values. All detection log files will be cleared and reinitialized.

`(Q)uit simulation` Activating this option will terminate the simulation and return control to the computer operating system. All detection log files will be saved with their current contents from the most recent simulation run.

Errors. Errors occur very rarely, usually when some mathematical argument is out of range or results in an infinite or undefined value. When such an error is detected, a warning window appears in the middle of the screen with three choices to proceed: “`(A)bort, (H)alt, or (C)ontinue`”. The appropriate choice depends on the error or warning encountered, and is activated by depressing the keyboard key indicated in parentheses (note: clicking the mouse on the appropriate field will *not* activate the choice). Selecting “`(A)bort`” will terminate the simulation, returning control to the computer operating system. In many cases, however, selecting “`(C)ontinue`” will allow the simulation to continue by “stepping over” the error. (“`(H)alt`” is an option primarily intended for use during program development.)

Using the mouse. The simulation is sensitive to the length of time a mouse button is pressed. That is, if the mouse button is not released fast enough, the looping control in the simulation may advance beyond a single time-step, and simulation displays, such as detection alerts, may appear to flash on and off, rather than remain visible on the screen. If such behavior is observed, try to click the mouse faster (that is, unless the mouse but-ton is being held down deliberately to effect a continuous running of the simulation).

APPENDIX E
DETECTION THEORY

Let s be a signal originating from some source in a background of Zero Mean Gaussian (ZMG) noise n. If x represents the signal arriving at some receiver, then either of the following hypotheses may hold:

H_0 **:** $x = n$, if the source signal is absent.
H_1 **:** $x = s + n$, if the source signal is present.

These are called the *null* and *alternative* hypotheses, respectively. If the received signal x exhibits a Normal (Gaussian) distribution $\mathcal{N}(\mu, \sigma^2)$, where μ and σ^2 are the mean and variance of the distribution, then under hypothesis H_0, $x \sim \mathcal{N}(0, \sigma_n^2)$ and under hypothesis H_1, $x \sim \mathcal{N}(\bar{s}, \sigma_{s+n}^2)$, where $\bar{s}$ is the mean signal level. The associated probability density functions for x under H_0 and H_1 are, respectively,

$$p_0(x) = (2\pi\sigma_n^2)^{-1/2} \exp\left[-x^2/(2\sigma_n^2)\right]$$

and

$$\begin{aligned} p_1(x) &= (2\pi\sigma_{s+n}^2)^{-1/2} \exp\left[-(x-\bar{s})^2/(2\sigma_{s+n}^2)\right] \\ &= (2\pi k^2\sigma_n^2)^{-1/2} \exp\left[-(x-\bar{s})^2/(2k^2\sigma_n^2)\right], \end{aligned}$$

where $k^2 \equiv \sigma_{s+n}^2/\sigma_n^2$. The parameter k implicitly incorporates the effect of a fluctuating source signal in the probability density function $p_1(x)$: For a fluctuating signal, $\sigma_{s+n}^2 > \sigma_n^2$ so that $k^2 > 1$, while for a steady signal, $\sigma_{s+n}^2 = \sigma_n^2$ so that $k^2 = 1$.* In this latter case, the effect of adding a constant signal to the ZMG noise effectively shifts the noise density function in the $+x$-direction by the amount $\bar{s}$, as illustrated in Figure E-1.

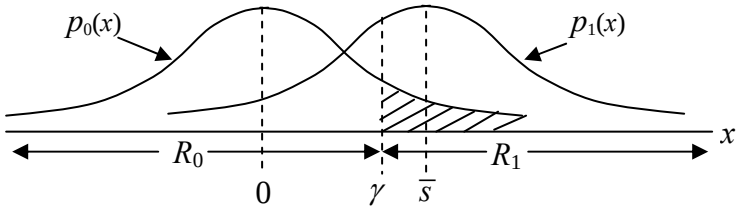


Figure E-1. Overlapping Gaussian distributions.

* Because the use of the parameter k eliminates the explicit dependence of $p_1(x)$ on σ_{s+n}^2, only σ_n^2 explicitly remains in the expressions for $p_0(x)$ and $p_1(x)$. Thus, for notational simplicity, the subscript n will hereafter be dropped with the understanding that σ^2 refers to the variance of the noise distribution.

In detection theory, the objective is to choose between H_0 and H_1, given some sample of the received signal x, called the *test statistic*. In doing so, two types of errors may occur. A Type I error occurs if H_0 is selected when in fact a source signal is present; this error occurs with the following probability:

$$P_{fd} = P(x \in R_0 \mid H_1) = \int_{-\infty}^{\gamma} p_1(x)\, dx\,,$$

called the *probability of false dismissal*. Note that the *probability of detection* of the source signal is $P_d = 1 - P_{fd}$. A Type II error occurs if H_1 is selected when in fact no source signal is present; this error occurs with probability

$$P_{fa} = P(x \in R_1 \mid H_0) = \int_{\gamma}^{\infty} p_0(x)\, dx\,,$$

called the *probability of false alarm*. Note that P_{fa} is the shaded area in Figure E-1. In either case, the limit of integration γ represents a threshold value against which x is compared in order to decide between H_0 and H_1.

The decision between H_0 and H_1 is usually made by defining a *likelihood ratio*, $\lambda(x) = p_1(x)/\, p_0(x)$. Note that for the probability density functions above,

$$\lambda(x) = \tfrac{1}{k}\exp[(k^2x^2 - x^2 + 2\bar{s}x - \bar{s}^2)/(2k^2\sigma^2)]\,.$$

Then for threshold γ, the test becomes

Accept H_1 if $\lambda(x) \geq \lambda(\gamma)$.
Reject H_1 if $\lambda(x) < \lambda(\gamma)$.

The likelihood test depends on finding an appropriate value for the threshold γ. Unfortunately, the source signal s is usually unknown, so the probability density function $p_1(x)$ cannot be specified. In this case, the usual approach is to apply the *Neyman–Pearson criteria*: Maximize P_d subject to some acceptable P_{fa}. For the noise distribution above, the probability of false alarm may be expressed in terms of the Error function, $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \exp(-y^2)\, dy$, as follows:

$$\begin{aligned} P_{fa} &= \int_{\gamma}^{\infty} p_0(x)\, dx = (2\pi\sigma^2)^{-\frac{1}{2}}\int_{\gamma}^{\infty} \exp\left[-x^2/(2\sigma^2)\right] dx \\ &= \tfrac{1}{\sqrt{\pi}}\int_{\Gamma}^{\infty} \exp(-y^2)\, dy = \tfrac{1}{\sqrt{\pi}}\int_0^{\infty} \exp(-y^2)\, dy - \tfrac{1}{\sqrt{\pi}}\int_0^{\Gamma} \exp(-y^2)\, dy \\ &= \tfrac{1}{2} - \tfrac{1}{2}\mathrm{erf}(\Gamma) = \tfrac{1}{2}[1 - \mathrm{erf}(\Gamma)], \end{aligned}$$

where the substitution $y = x/(\sqrt{2}\,\sigma)$ has been made and where $\Gamma = \gamma/(\sqrt{2}\,\sigma)$. That is, $\mathrm{erf}(\Gamma) + 2\, P_{fa} - 1 = 0$. For a specified P_{fa}, this last equation can be solved numerically for Γ (and hence $\gamma = \Gamma\sigma\sqrt{2}$, provided that the variance σ^2 of the noise distribution is known or can be estimated).

With the value of Γ thus determined, the probability of detection for the signal plus noise distribution is

$$
\begin{aligned}
P_d &= \int_{\gamma}^{\infty} p_1(x)\, dx = (2\pi k^2\sigma^2)^{-\frac{1}{2}} \int_{\gamma}^{\infty} \exp\left[-(x-\bar{s})^2 / (2k^2\sigma^2)\right] dx \\
&= \tfrac{1}{\sqrt{\pi}} \int_{(\Gamma-D)/k}^{\infty} \exp(-y^2)\, dy = \tfrac{1}{\sqrt{\pi}} \int_{0}^{\infty} \exp(-y^2)\, dy - \tfrac{1}{\sqrt{\pi}} \int_{0}^{(\Gamma-D)/k} \exp(-y^2)\, dy \\
&= \tfrac{1}{2} - \tfrac{1}{2}\,\mathrm{erf}\left(\frac{\Gamma\text{-}D}{k}\right) = \tfrac{1}{2}\left[1 - \mathrm{erf}\left(\frac{\Gamma\text{-}D}{k}\right)\right]
\end{aligned}
$$

where the substitution $y = (x-\bar{s})/(\sqrt{2}\,k\sigma)$ has been made and where $D = \bar{s}/(\sqrt{2}\,\sigma)$. Since D is a function of the unknown source signal mean $\bar{s}$ (as well as the noise variance σ^2), P_d as yet cannot be determined. However, the quantity $d \equiv \bar{s}^2/\sigma^2 = 2D^2$, called the *detection index*, is a measure of the signal-to-noise ratio (SNR). Extensive work has been performed in acoustics and electromagnetics to develop equations defining the SNR in various environments.

E.1 DETECTION OF ACOUSTIC SIGNALS

For acoustic environments, the appropriate SNR equations are the passive and active SONAR equations. For *passive* SONAR, the source signal originates from the target, whereas for *active* SONAR, the source signal originates from a transducer and reflects from the target. These equations (in units of decibels are

DT = SL – TL + AG – AN

and

DT = SL –2TL + TS + AG - AN,

respectively, where

DT = *Detection Threshold*, the SNR necessary to achieve a desired level of performance (DT is also called the *Recognition Differential*, RD)

SL = *Source Level*, the ratio of the acoustic intensity (proportional to the squared pressure) of the source to that of a plane wave of rms pressure p_0 = 1 μPa at a reference distance r_0 = 1 m from the source.

TL = *Transmission Loss*, the ratio of the acoustic intensity at a distance r_0 = 1 m from the source to that at a distance r from the source.

AG = *Array Gain*, the ratio of the SNR of an array of coherently summed receiver elements to that of a single element.

AN = *Ambient Noise*, the spectrum level (power) of all incoherent sources of noise in a 1-Hz band at the receiver

TS = *Target Strength*, the ratio of the acoustic intensity reflected from a target at a distance r_0 = 1 m from the target to that from a source at distance r from the target.

In the active SONAR equation, the *Reverberation Level*, RL, replaces the terms (AN–AG) for environments where the reverberation level exceeds that of the AN. Note that there are various representations for the terms of the SONAR equation with different levels of fidelity. For example, in a homogeneous seawater environment, TL may be modeled by simple spreading loss and absorption. However, boundaries and spatial and temporal variability of the sound field can cause refraction and other effects requiring sophisticated computer models and environmental databases for evaluation.

Thus, for a given SL, TS, AG, and AN appropriate for the source, target, receiver, and ambient environment, and assuming some model for the TL, the passive or active SONAR equation may be evaluated for the DT. Since DT = 10 log d, the detection index d can be found from

d = antilog[DT/10]

= antilog[(SL – TL + AG – AN)/10], for passive acoustic detection;

= antilog[(SL – 2TL + TS + AG – AN)/10], for active acoustic detection.

Then $D = \sqrt{d/2}$, and the probability of detection may be estimated from

$$P_d = \tfrac{1}{2}\left[1 - \mathrm{erf}\left(\frac{\Gamma - D}{k}\right)\right].$$

Note that since the TL is range-dependent, so is the P_d. Also note that the variance of the noise distribution σ^2 , is implicitly incorporated within the terms Γ and D, and therefore does not need to be known or estimated.

APPENDIX F
KMDT DEMONSTRATION CONCEPT

The KMDT demonstration involves the use of two computers connected via a local-area network (LAN). One computer runs a simulation based on a scenario involving multiple targets and sensor platforms, and a contact generator that provides line-of-bearing (LOB) alerts on detected targets (including false alarms). Detection alerts are posted to databases residing in the simulation computer that may be accessed by the second computer via the LAN. The second computer acts as a command center, where requests for information originate, software agents are tasked to gather appropriate information from the network (i.e., from the detection databases), and subsequent data fusion takes place. The detection databases will serve the KMDT demonstration as "Web portals" providing Web services to the network. The demonstration functional configuration is illustrated in Figure F-1.

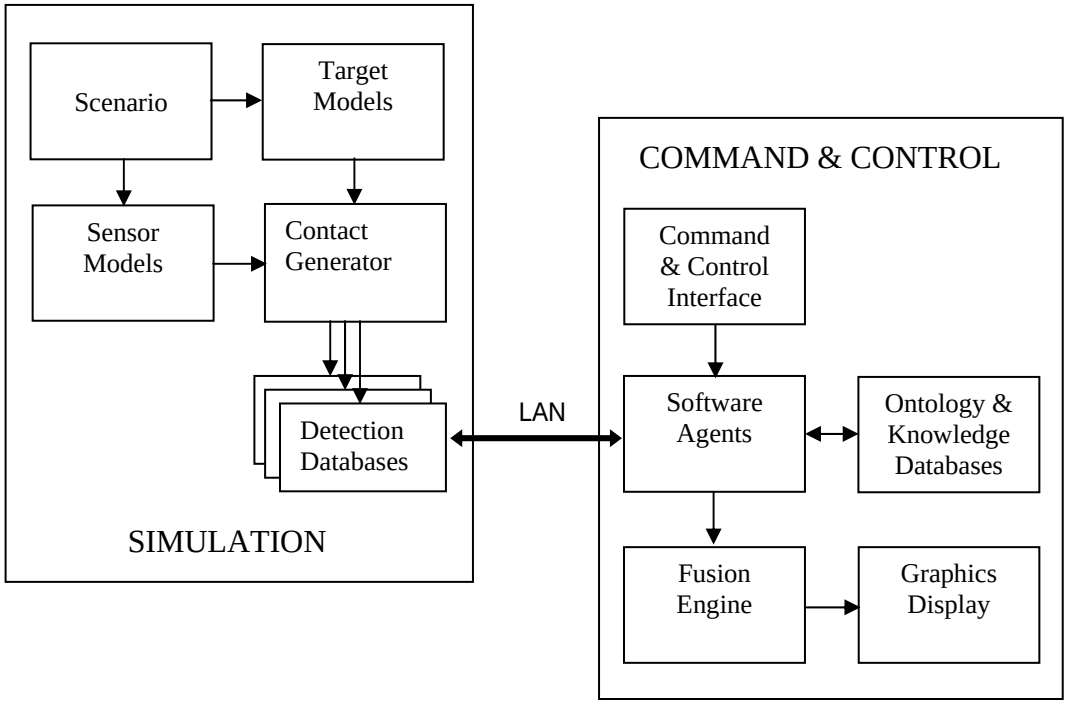


Figure F-1. Demonstration functional configuration.

When a request for information is initiated, a software agent is tasked to acquire records from the detection databases via the LAN. The software agent is basically a program on the command and control computer that establishes connectivity with the simulation computer, reads records from the detection databases, and decides whether or not the records contain information that might be valuable in satisfying the request.

To accomplish this task, the software agent must have, among other things, access to additional ontology and knowledge databases (such as sensor and target information libraries, taxonomies, etc., residing in the command center computer or, perhaps, on a third computer) that contain information on various sensor parameters, classification signatures, data relationships, etc. The software agent must also have algorithms to decide if acquired information is appropriate. In essence, these algorithms, along with the relationships defined in the ontology and knowledge databases, provide the "intelligence" of the agent.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-01-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
November 2008	Final	

4. TITLE AND SUBTITLE

KNOWLEDGE MANAGEMENT FOR DISTRIBUTED TRACKING:
THE FINAL REPORT

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHORS

Marion G. Ceruti
Tedd L. Wright
David J. Swanson
Scott C. McGirr
Dwight R. Wilcox

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

SSC Pacific
San Diego, CA 92152–5001

8. PERFORMING ORGANIZATION REPORT NUMBER

TR 1974

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research, Code 31
875 North Randolph Street, Suite 1425
Arlington, VA 22203-1995

SSC Pacific Science and Technolgy Initiative
53560 Hull Street
San Diego, CA 92152-5001

10. SPONSOR/MONITOR'S ACRONYM(S)

ONR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Modeling and simulation, intelligent software agents, and other technologies can support network-centric distributed tracking. These technologies came together in the Knowledge Management for Distributed Tracking (KMDT) research and development program to improve naval command, control, and decision support. The program's approach is based on the use of simulated data from sensor and motion models, intelligent software agents, integrated sensor ontology, and a line-of-bearing cross-fix algorithm.

Modeling and simulation was used to generate test data that intelligent agents ingested to search for information that could help localize and characterize unknown contacts in the simulated battle space. In these simulations using a hypothetical scenario, intelligent-software agents were deployed over a simulated, secure Web-like network to find additional, possibly disparate, sensor data from other friendly platforms on unknown contacts.

The concept of operations calls for the agents to interact with the integrated sensor ontology to facilitate distributed, heterogeneous sensor-data fusion and to reduce uncertainty. To illustrate these concepts, a simulation is described that generated 400 contact reports consisting of passive lines of bearing that were fused subsequently by intelligent agents to localize the contacts, thus demonstrating the potential of this approach to reduce information overload for the operator. The KMDT approach demonstrates how knowledge management technologies can be employed to improve situation awareness and reduce operator workload.

15. SUBJECT TERMS

Mission Area: Command and Control

modeling and simulation	contact generator	detection databases	temporal-feasibility filtration	intelligent agents	cross-fix
agent design	parametric method	agent-control interfaces	geo-feasibility filtration	agent-ontology interactions	

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			M. G. Ceruti
U	U	U	UU	71	19B. TELEPHONE NUMBER *(Include area code)* (619) 553–4068

INITIAL DISTRIBUTION

84300	Library	(2)
85300	S. Baxley	(1)
85300	Archive/Stock	(1)
530	R. Jaffee	(1)
530	M. Kidwell	(1)
536	T. Tiernan	(1)
5362	N. Campbell	(1)
53624	M. Ceruti	(23)
53624	G. Leonard	(1)
53624	D. Swanson	(1)
53627	K. Adams	(1)
53627	D. Brooner	(1)
53627	D. Hardy	(1)
53627	J. Park	(1)
53627	C. Putnam	(1)
53627	D. Wilcox	(1)
561	G. Shaffer	(1)
5615	P. Sheets	(1)
5615	T. Wright (c/o Paul Sheets)	(1)
5615	S. McGirr	(1)
5615	R.Glen	(1)
72110	R. Boss	(1)

Defense Technical Information Center
Fort Belvoir, VA 22060–6218 (1)

SSC San Diego Liaison Office
C/O PEO-SCS
Arlington, VA 22202–4804 (1)

Center for Naval Analyses
Alexandria, VA 22311–1850 (1)

Government-Industry Data Exchange
Program Operations Center
Corona, CA 91718–8000 (1)

AFRL/RISF
525 Brooks Rd
Rome NY 13441-4505 (1)

Approved for public release; distribution is unlimited.

SSC Pacific
San Diego, CA 92152-5001