Sensor Resource Management with Hierarchical Target Valuation Models

Joe P. Hill
ALPHATECH, Inc.
3811 N. Fairfax Drive, Suite 500
Arlington, VA 22203 USA
jhill@alphatech.com

K.C. Chang
George Mason University
Department of Systems Engineering
and Operations Research
Fairfax, VA 22030 USA
kchang@gmu.edu

Abstract - *Advanced optimization-based algorithms for sensor resource management have been previously developed and presented. These algorithms offer the potential for automating the sensor control process in response to level 1 sensor data fusion (object or track-level) estimates. In this paper, a hierarchical target valuation model that estimates target value on the basis of not only level 1 fusion information but also on the basis of level 2 fusion information will be presented. These algorithms can benefit the identification of higher level units such as convoys of vehicles. .*

Keywords: Sensor Resource Management, Level 2 fusion, target valuation

1 Introduction

In a previous paper [1], we presented a description of an algorithm that could be used to provide hierarchical target valuation based on not only level 1 fusion (e.g., object or track) information, but also level 2 fusion (e.g., groups of objects) information. This algorithm has the potential to improve the target valuation function used in a sensor resource manager by adding a valuation component related to the ability of an object to be able to identify a group of objects, such as convoys. This algorithm was based on earlier results [2] that only addressed target valuation based on level 1 fusion information. In this paper, we will first present a complete description of the valuation function. We shall then present an evaluation environment which was used to analyze the performance of this valuation algorithm assuming a simple ground moving target scenario. Finally, we will present a sample of some of the results obtained to date.

2 Problem Description

In this section we will first present introductory material, and then briefly describe the sensor resource management problem as well as our proposed solution approach.

2.1 Introduction

Current Intelligence Surveillance and Reconnaissance (ISR) sensors can detect and take measurements on individual entities, such as moving vehicles and installations. These measurements can be used to infer the particular class of these individual entities. However, very few collection assets provide direct measurements on the hierarchical force structure of units that the entities comprise. Consequently, it is desirable to develop the capability to produce inferences on the hierarchical structure of military units based on inferences and measurements of individual entities and sub-units. In this paper, we will reference previous results presented in [3] that describe a technique that can be used to assess the relative merit of force aggregate hypotheses from partial observations of a set of entities. That is, given partial observations of entities that comprise military units, the technique draws inferences about the type of military unit that is present. Furthermore, drawing inference about the type of military unit provides contextual information that enables improved inference about the type of individual vehicles.

These ISR systems typically employ agile, multi-mode sensors which are capable of producing a variable scan pattern within a surveillance region in response to external tasking. A number of platforms are currently available to carry out these missions. Long-range surveillance is accomplished with aircraft capable of high coverage rate, high signal to noise moving target indicator (MTI) and high range resolution (HRR) modes (see Fig. 1) Additionally, these systems can perform long dwell synthetic aperture radar scans for purposes of identification. Closer in, unmanned aerial vehicles (UAVs) provide correspondingly less coverage and shorter dwell SAR imagery; however, their MTI and HRR modes have better signal to noise than the standoff systems and offer higher resolution. Finally, very close-in assets (such as the Joint Strike Fighter (JSF)) provide significant aspect diversity as they maneuver around the target and are able to collect a rich set of feature data useful for ID and track maintenance. Typically, this sensor is a multi-mode, electronically scanned antenna radar capable of tasking individual beams in terms of pointing direction, dwell time, and wave-form. As illustrated in Fig. 1, a typical radar will be capable of not only interleaving various radar beam modes (i.e., wide area search (WAS), sector search (SS), HRR, and synthetic aperture radar (SAR)), but will also be capable of scanning the surveillance region in an asynchronous fashion as the timeline suggests (e.g., irregular revisits could be due to sensor tasking to maintain tracks, search new areas, identify high-value targets, etc.). For such systems the dynamic management of sensor mode control requires an automated process due to timeline for adaptation. The challenge of sensor resource management for such systems is to characterize the exploitation and data production process according to a consistent model that provides for real time adaptive sensor management.

To support solution of the sensor resource management

problem, several algorithms have previously been developed algorithms using different solution approaches

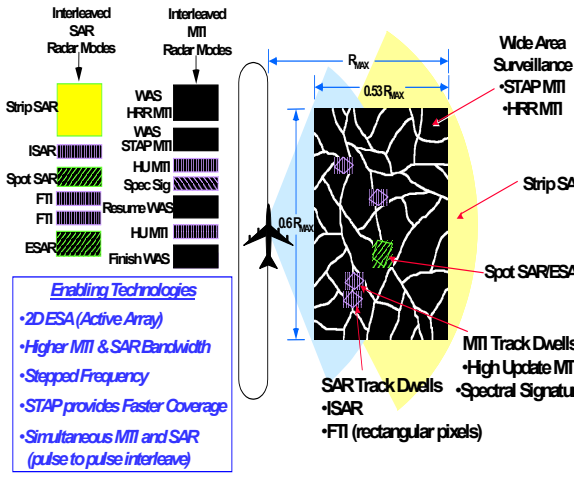

Figure 1. Advanced Airborne Surveillance Radars will include capabilities to operate in multiple modes and interleave modes

based on stochastic dynamic programming [4-6]. These algorithms for the sensor resource management problem were developed to address the problem of determining the optimal time sequencing for the radar's SAR mode (for detecting stationary objects) and for the radar's MTI mode (for tracking moving objects) that maximizes the total information value [7]. The scheduler operates in a feedback manner in real time; for example, as objects are detected by the SAR, they may be eliminated from consideration by the scheduler so that the remaining radar resources can be better focused on only those objects remaining undetected or needing track improvement.

The output of the sensor resource manager is to determine for each instant of time whether the radar should

- operate in the SAR mode to image a single cell (for a stopped target), or
- operate in the MTI mode to detect and update tracks on all moving objects, or
- operate in the HRR mode to image a single cell (for a moving target) in order to obtain identification information (this is indeed the *only* way to obtain classification information), or,
- not be used at all in order to meet some exposure constraint.

As indicated in the previous paragraphs, there are three primary radar modes available on these platforms for collecting data: MTI, HRR, and SAR. The MTI mode typically requires the shortest dwell times (and thus consumes the least amount of radar resources); however, it has low range resolution and typically low signal to noise ratio. The MTI mode is capable of detecting targets moving faster than the so-called minimum detectable velocity (MDV) of the sensor. It is well suited for problems such as tracking moving vehicles, characterizing traffic flow, and lines of communication using low-complexity (highest throughput) algorithms. The HRR mode has slightly longer dwell times (still shorter than SAR) but offers higher range resolution and strong signal to noise performance again for characterizing targets whose velocities exceed the MDV of the sensor. It has been proven to be useful for extracting coarse features (e.g. length and width) and as a tool for low-confidence classification. The HRR mode is eminently well suited for track maintenance problems. Finally, the SAR mode is ideal for two dimensional, high confidence classifications of stationary targets as well as change detection.

In many cases, the optimality of a sensor allocation policy is defined in terms of reduced tracking error and the best policy is determined though the solution of an optimization problem using a dynamic programming approach. While significant progress has been made in this area in the past, there remain open issues in the synthesis and validation of an approach to sensor resource management capable of coordinating a multi-sensor classification system.

2.2 Problem Formulation

In this section we will present a basic description of the hierarchical data fusion functional model and identify how the proposed algorithm relates to this model. The hierarchical data fusion process is well known and is characterized by continuous refinements of its estimates and assessments, and the evaluation of the need for additional sources, or modification of the process itself, to achieve improved results [8].

A more concise definition was later proposed by Steinberg, et. al. [9] as: data fusion is the process of combining data to refine state estimates and predictions. Fig. 2 depicts the data fusion functional model as revised in [9], which further elaborates on the composition of each of the levels as follows:

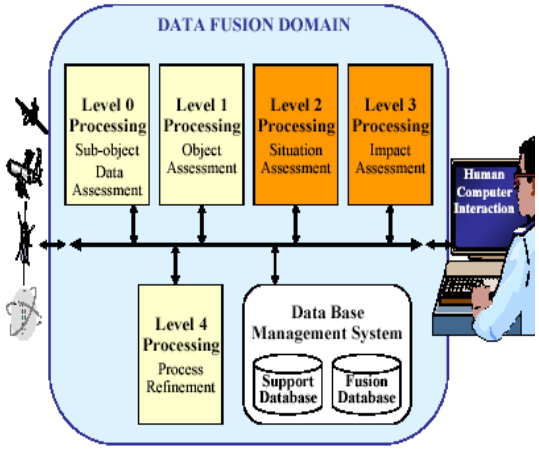

Figure 2. The JDL Data Fusion Functional Model

- *Level 0 - Sub-Object Data Assessment:* estimation and prediction of signal/object observable states on the basis of pixel/signal level data association and characterization

- *Level 1 - Object Assessment*: estimation and prediction of entity states on the basis of observation-to-track association, continuous state estimation (e.g. kinematics) and discrete state estimation (e.g. target type and ID)
- *Level 2 - Situation Assessment:* estimation and prediction of relations among entities, to include force structure and cross force relations, communications and perceptual influences, physical context, etc.
- *Level 3 - Impact Assessment:* estimation and prediction of effects on situations of planned or estimated/predicted actions by the participants; to include interactions between action plans of multiple players (e.g. assessing susceptibilities and vulnerabilities to estimated/predicted threat actions given one's own planned actions)
- *Level 4 - Process Refinement (an element of Resource Management):* adaptive data acquisition and processing to support mission objectives

Typically, the sensor resource management algorithms developed to date have only focused on the ability to maintain track and identification on individual objects (battlefield entities) with little regard to any higher functional level entities (e.g., groups of objects). This has been accomplished by representing the track fusion process as a computational algorithm adapted to address the Sensor Resource Management (SRM) scheduling problem. In the context of the Joint Director of Laboratories (JDL) terminology [8], it is observed that the track fusion models previously considered only address Level 1 fusion (Processing/Object Assessment). We contend that the problem of effective SRM for agile, multi-mode sensors will require improved representations of the process exploitation through Level 2 fusion to adequately address the benefits of agile sensor tasking.

3 Sensor Resource Management Evaluation Environment

In order to be able to test our target valuation algorithms, we implemented a simple test architecture, as shown in Fig. 3. Note that the purpose of this architecture was *not* to provide high fidelity modeling conditions; rather, it was designed to be quickly constructed for the purpose of evaluating the proof-of-concept target valuation algorithm(s) that were developed during this effort.

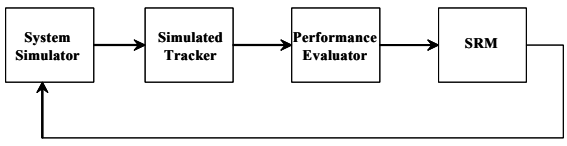

Figure 3. System Architecture

The architecture contains an outer loop where at each instant of time, the system simulator will create and send the current scenario information (related to the targets and sensors) to the simulated tracker. The simulated tracker will then produce the simulated tracks with proper joint (tracking, classification) quality states and classification vectors based the Markov transition model as well as the true target/sensor parameters. The simulated tracker will then send the track results to the evaluator. The evaluator will use the tracking results to determine what is the best sensor mode and pointing direction for next instant of sampling time and send that decision back to the system simulator. In the remainder of this section, we will provide a description of each of the components in Fig. 3.

3.1 System Simulator

The system simulator is the overall driver of the system. It will generate the ground truth scenarios including target trajectories, group/convoy composition, sensor placements, and sensor observations based on sensor mode/characteristics, as well as sensor/target geometry. Note that the two important aspects of the simulator are: (a) the ability to simulate group/convoy behavior, and (b) the ability to switch sensor operating modes based on the information supplied by the Performance Evaluator and SRM components .

The system simulator will send parameters such as the number of targets, their locations and classes, group composition/identity, sensor mode, detection probability, false alarm density, target density, and confusion matrix/classification probability to the Simulated Tracker. The relative target/sensor geometry (which accounts for a target dropping below the sensor's Minimum Detectable Velocity (MDV) for MTI) will be incorporated by the system simulator to produce the required operating parameters.

We will leave the burden of representing the convoys to the system simulator. Namely, the system simulator will send both target and convoy/group information to the simulated tracker. For simplicity, we will assume coverage of all targets in the test scenario area of interest (AOI) when the sensor is in the MTI mode. On the other hand, the HRR mode has a more limited FOV, as indicated in Fig. 4.

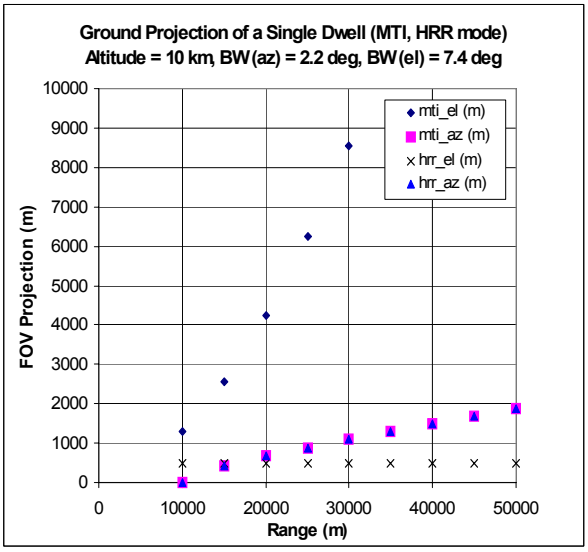

Figure 4. Ground Projection of MRI and HRR FOV

3.2 Simulated Tracker

The purpose of this module is to produce simulated tracking results for the Performance Evaluation Component without implementing a complete tracker. The simulated tracker will receive input such as ground truth, sensor models, etc. from the system simulator and produce a set of tracks each with kinematic and classification joint quality state. Note that for every ground truth target, there will be a "track" which will be in one of the joint quality states at each moment of time. In this effort, we did not attempt to use a complete Multiple Hypothesis Tracker (MHT) for tracking moving ground targets, although we will use such a model in a future effort. Rather, the simulated tracker only estimates the joint kinematic and classification state for each target track.

There are three joint kinematic/classification quality states: untracked/unclassified, tracked/unclassified, and tracked/classified. They behave according to a Finite State Markov transition model based on sensor mode and operating conditions. The model is illustrated in Figs. 4-6 for the three choices available to the SRM: to either use MTI mode (Fig. 5), use the HRR mode (Fig. 6), or not use

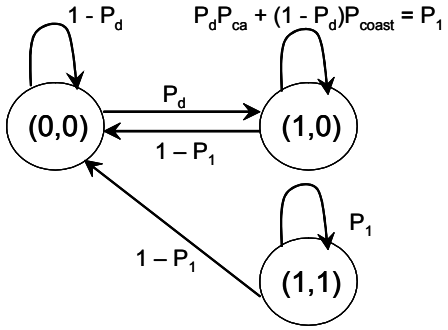

Figure 5. Finite State Markov Diagram Assuming Use of the MTI Sensor

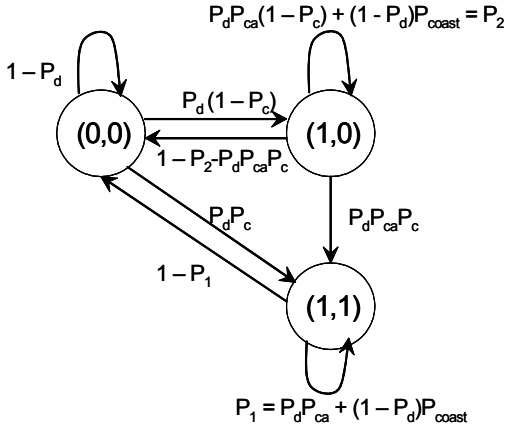

Figure 6. Finite State Markov Diagram Assuming Use of the HRR Sensor

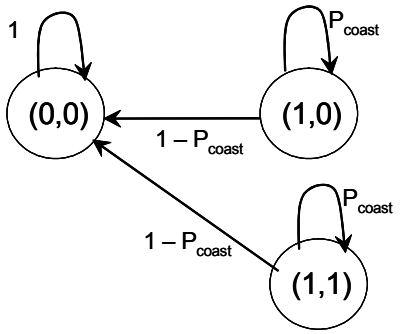

Figure 7. Finite State Markov Diagram Assuming Sensors Not Used

the sensor at all (Fig. 7). Note that this is a simplified form of the stare transitions that were presented in [1], which were chosen for use here on the basis of their ease of implementation. The transition probabilities between these states are specified by the following parameters which depend on the specific sensor mode being used by the radar, the sensor beam geometry for the track position indicating its probability of detection, and tracker-related parameters describing how tracks are nominated, promoted and dropped.

- P_d is the probability of detection, which is a function of the sensor/target geometry
- P_{coast} is the probability that a track is coasted, that is, the probability that a track will continue without a supporting measurement
- P_{ca} is the probability of correct association, that is, the probability that a measurement will be associated with the correct track

In these diagrams, we assume the following notation: the target state is denoted as (tracked, classified); "0" denotes untracked/unclassified while "1" denotes tracked/classified. In addition to the joint quality states, in order to evaluate the track value function, each track needs to have a posterior classification probability distribution. The simulated tracker will produce the classification probability vector based on the true target class, previous track class probability, and the current sensor mode and operating conditions. For example, at the beginning of the simulation, each track will be at untracked/unclassified state with a uniform classification probability. Depending on the sensor mode (of the next sampling time) and operating characteristics, the track joint quality state at the next sampling time will be simulated stochastically based on the Finite State Markov transition model. The track classification probability will also be updated (consistent with the quality state) using the confusion matrix of the particular sensor mode. Note that for kinematic state, the simulated tracker will not represent the tracks in the target state space, but rather only on the quality state space. The simulated tracker will then send the track results to the evaluator.

3.3 Performance Evaluator

The approach used here is based on an open-loop feedback concept, otherwise known as model-predictive control. The basic idea is that at any frame t, we generate the desired sequence of sensor decisions for frames t, t+1, ..., t+H based on the aggregate evolution represented by the information quality Markov chains in Figs. 4-6, where H represents the end of the planning horizon. We then collect the information from frame t, and receive updated tracking information. Given this new information, we repeat the process and select decisions for frames t+1, t+2, ..., t+H+1, receive new information from the tracker and continue the iteration. Thus at each frame t, we compute sensor management decisions for several frames ahead, but use only the next frame's decisions before resolving the SRM problem. An important aspect of the sensor management methodology is that it makes immediate sensor choices (mode, pointing direction) with a view towards how these decisions will affect the information state H frames in the future. The size of H reflects a tradeoff between the desire to account for future actions versus the unpredictable evolution of future target motions. Larger values of H introduce more prediction uncertainty into future target positions, thereby making it harder to predict the effect of future sensor actions. However, for this effort we assumed that H = 1, (i.e., a one-step lookahead) in order to simplify the implementation (this avoids the need for dynamic programming techniques) and to focus on development and implementation of a higher-order fusion valuation algorithm.

The evaluator receives the track results from the simulated tracker. In the results, there will be a set of clusters/groups/convoys where each group contains a set of tracks. Each track has a joint quality state and a classification probability vector. The evaluator will use the track results to determine what is the best sensor mode and pointing direction for the next detection time. In order to do so, in addition to the information from the tracker, truth-related domain knowledge such as unit composition, value function (for each target type, convoy/unit type) are needed as well. The evaluator will first find all the possible detected compositions (based on the possible unit types) and map these to the track compositions of each cluster, which are produced by the simulated tracker. It will then compute the value function based on a hierarchical Bayesian model/inference which takes into account the track classification likelihood of a joint state assignment. The evaluator will then produce a best sensor mode decision (HRR or MTI) and pointing direction based on the resulting information value and send these decisions back to the SRM to be used by the system simulator for the next sampling time.

We next present the detailed models for the hierarchical target valuation algorithms. We first define the SRM objective for sensor decisions selected at a frame t as follows: given a set of SRM *decisions* for frames t, t+1, ..., t+H, where H is the planning horizon, for each SRM track, we want to make the best decisions (u_t, u_{t+1}, ..., u_{t+H}) in order to maximize the value of some cost function, which we can define as

$$J(u_k, u_{k+1}, \ldots, u_{k+N}) = \sum_{SRM\_Clstr} \left[V(SRM\_Clstr) \bullet \sum_{Clstr\_Trk} V(Clstr\_Trk) \sum_{TQS} \sum_{CQS} \left[P(TQS \mid Clstr\_Trk) \bullet P(CQS \mid Clstr\_Trk) \bullet V(TQS, CQS) \right] \right] \quad (1)$$

where

$$V(SRM\_Clstr) = \sum_{Unit\_Types} V(Unit\_Type) P(Unit\_Type \mid Clstr\_Trk) \quad (2)$$

Intuitively, Eq. (1) may be considered to be analogous to the "chain rule", that is, the total value is the sum of values of track(s) to cluster(s) multiplied by the sum of the values of cluster(s) to the SRM. The expression for the cost function in Eq. (1) is analogous to a similar expression in [2] (which considered only the level 1 fusion SRM problem), but in Eq. (1) we have introduced the concept of "clusters" (i.e., groups of tracks) between the (lower) track level and the (higher) SRM level. In Eq. (1), V(*Clstr_Trk*) represents the value (or priority) of each track/object in a particular cluster. Also, the conditional probabilities P(*TQS* | *Clstr_Trk*) and P(*CQS* | *Clstr_Trk*) are computed (predicted) using the Finite State Markovs in Figs. 5-7 for the Tracking Quality States (TQS) and Classification Quality States (CQS), respectively. Clearly, the value of the joint tracking and classification state (V(TQS,CQS)) is higher for the (joint) state of (tracked, classified) and lower for the other states.

In Eq. (2), the unit/cluster value function (V(SRM_Clstr)) represents the value of the cluster (group of tracks) to the SRM. It is dependent upon V(Unit_Type) and P(Unit_Type | Clstr_Trk), where V(Unit_Type) is the value specified by the decision maker (i.e, a commander) based on their priority and P(Unit_Type | Clstr_Trk) is the Unit_Type probability given a cluster of tracks computed by Bayes rule,

$$P(Unit\_Type \mid Clstr\_Trks) = \frac{p(Clstr\_Trks \mid Unit\_Type) P(Unit\_Type)}{p(Clstr\_Trks)} \quad (3)$$

The solution to Eq. (3) represents one of the key contributions of the algorithms described in [3]. First of all, P(*Unit_Type*) represents the prior probabilities, and the denominator p(*Clatr_Trks*) is the likekihood of all measurements which simply normalizes the relative probabilities computed in the numerator. The remaining term, p(*Clstr_Trks* | *Unit_Type*) can be computed as follows:

$$p(Clstr\_Trks \mid Unit\_Type) = \sum_{d \in D(n, r+1)} p(Clstr\_Trks \mid d) P(d \mid Unit\_Type) \quad (4)$$

In Eq. (4), $D(n, r+1)$ is the set of all possible distributions of the n detected vehicles into the r+1

vehicle classes (including the false-alarm class), which is a space with $(r + 1)^n$ elements, $p(Clstr\_Trks \mid d)$ is the likelihood of tracks classification states given the specific detection composition d, and $P(d \mid Unit\_Type)$ is the probability of detection composition given the unit type. Note that in Eq. (4),

$$p(Tks \mid d) = \sum_a p(Clstr\_Trks \mid a) P(a \mid d) = \sum_a \left[\prod_{k=1}^{n} p(\text{Clstr\_Trks}(k) \mid \nu_k) \right] P(a \mid d) \tag{5}$$

and from the detection model,

$$P(d \mid u) = p_o(n(0;d);\lambda_{FA}) \prod_{v=1}^{r} P_B(n(v;d) \mid n(v;u)) \tag{6}$$

where in Eq.(5), $p(\text{Clstr\_Trk}(k) \mid \nu_k)$ is the likelihood of producing a track classification state $\text{Clstr\_Trk}(k)$ given a class ν_k vehicle, and a is the joint assignment of a set of vehicles types to a set of tracks. In Eq. (6), $n(v;d)$ is the number of detected class v vehicles, $n(0;d)$ is the number of false detections, $n(v;u)$ is the number of class v vehicles in a type u unit, $p_o(k;\lambda) = \lambda^k e^{-\lambda}/k!$ is the Poisson distribution for false alarm detection probability, and

$$P_B(n(v;d) \mid n(v;u)) = \sum_{k=0}^{\min[n(v;d),n(v;u)]} B(k;n(v;u),P_D(v)) \times p_o(n(v;d) - k;\lambda_C(v)) \tag{7}$$

is the probability of target detection with $B(k;n,p) = C_k^n p^k (1-p)^{n-k}$, which is a Binomial distribution.

Obviously, Eqs. (4-7) involve intensive computations where the enumeration of an exponentially growing set is needed. In general, this could be very time consuming and may not be feasible. One idea to simply the computation is to approximate Eq. (4) with

$$p(Clstr\_Trk \mid Unit\_Type) \approx P(d(Clstr\_Trk) \mid Unit\_Type) \tag{8}$$

where $d(Clstr\_Trk)$ is defined as the joint detection-classification state by collapsing all the track classification probability distributions into one. Namely, $d(Tks) = \sum_{tk \in Tks} P_C(tk)$, where $P_C(tk)$ is the classification probability distribution of a track *tk*. Then

$$P(d(Tks) \mid u) = \prod_{v=1}^{r} P_B(n(v;d(Tks)) \mid n(v;u)) \tag{9}$$

where $P_B(n(v;d(Tks)) \mid n(v;u))$ is defined similar to Eq. (7). However, since $d(Tks)$ is a vector of positive real numbers (not necessary integers). It may not be possible to perform the calculation in Eq. (7). One idea is to approximate it by

$$P_B(n(v;d(Tks)) \mid n(v;u)) \approx N(n(v;d(Tks);\bar{n},\sigma_{\bar{n}}^2) \tag{10}$$

where $N(n;\bar{n},\sigma_n^2)$ is a Gaussian distribution, $\bar{n} = n(v;u)P_D(v) + \lambda_v$ is the expected number of detected class v targets, and $\sigma_{\bar{n}}^2 = n(v;u)P_D(v)(1-P_D(v)) + \lambda_v$ is the approximate associated variance.

4 Example Results

We implemented the architecture described in Figure 3 and also defined a set of metrics that can be used for evaluation purposes:

- *Sensor allocated resources* – the percentage of time that the sensor is operating in HRR mode (HRR Rate)
- *Average probability of correct unit classification* – the average correct unit classification probability over the simulation time (P_{cc})
- *Average percentage of correctly identified targets* – the average percentage of correct target classification in each unit over the simulation time (Trk Rate)

In order to test these algorithms, we implemented a simple ground moving target scenario containing convoy units, each consisting of a different combination of target types. There are a total of 4 possible types of unit classes: Scud (class 1), C2 (class 2), Tank (class 3), and Unknown, as well as 6 target classes: UAZ-469 (class 1), ZIL-151 (class 2), GAZ-66 (class 3), MAZ-543 (class 4), T-72 (class 5), and Other (class 6). However, this particular scenario only contains two units: unit 1 (containing 2 type UAZ-469 vehicles, as well as one each of types ZIL-151, GAZ-66, MAZ-543) and unit 2 (containing 5 vehicles of type UAZ-469). This means that, for this particular scenario, there are a total of 15 possible combinations, as follows: 2 (unit classes) * 4 (target classes) + 4 (level 1 valuation only) + 2 (level 2 valuation only) + 1 (neither level 1 nor level 2 valuation) = 15. Also, we assume the cost of using the HRR mode is about 20% more expensive than the MTI mode.

We made several test trials with different value functions and strategies. Specifically, in each of the combinations below, the notation "Case xy" refers to x = unit_class and y = target_class. Also, the notation [a b c d] refers to the valuation of each unit (since there are 4 possible units for this particular scenario) and the notation [a b c d e f g] similarly refers to the valuation of individual targets, since there are 6 possible targets. Note that for simplicity, the target valuation is assumed to be a "binary" variable (i.e., valued at either 0 or 1); other combinations of target values are certainly possible, but were not considered here. Here are the possible permutations:

Level 1 value function only, unit class value [1 1 1 1]

- Case 01: Focus on target class 1, track class value: [1 0 0 0 0 0]
- Case 02: Focus on target class 2, track class value: [0 1 0 0 0 0]
- Case 03: Focus on target class 3, track class value: [0 0 1 0 0 0]
- Case 04: Focus on target class 4, track class value: [0 0 0 1 0 0]

Level 2 value function only, target class value [1 1 1 1 1 1]

- Case 10: Focus on unit class 1, unit class value: [1 0 0 0]
- Case 20: Focus on unit class 2, unit class value: [0 1 0 0]

Level 1 and 2 value functions

- Case 11: Focus on unit class 1, [1 0 0 0], and target class 1, [1 0 0 0 0 0]
- Case 12: Focus on unit class 1, [1 0 0 0], and target class 2, [0 1 0 0 0 0]
- Case 13: Focus on unit class 1, [1 0 0 0], and target class 3, [0 0 1 0 0 0]
- Case 14: Focus on unit class 1, [1 0 0 0], and target class 4, [0 0 0 1 0 0]
- Case 21: Focus on unit class 2, [0 1 0 0], and target class 1, [1 0 0 0 0 0]
- Case 22: Focus on unit class 2, [0 1 0 0], and target class 2, [0 1 0 0 0 0]
- Case 23: Focus on unit class 2, [0 1 0 0], and target class 3, [0 0 1 0 0 0]
- Case 24: Focus on unit class 2, [0 1 0 0], and target class 4, [0 0 0 1 0 0]

Neither level 1 nor level 2 value functions

- Case 00: unit class value, [1 1 1 1], and target class value, [1 1 1 1 1 1]

Note that in order to not consider target (or unit) value, all of the entities are equally valued and have a value of 1, e.g., a target class of [1 1 1 1 1 1].

The performance metrics for several different cases are summarized in the following tables. Table 1 shows the performance results using only level 1 value functions. Note that case 00, by definition, contains neither level 1 nor level 2 value functions. The results show that the classification performance of both units are in the range of 40-60%. While the unit 1 classification improves somewhat when adding level 1 valuation, the unit 2 classification performance remains almost unchanged. This is because the individual level 1 value function has little impact on the unit (level 2) discrimination ability.

Table 1: Performance Results Using Only Level 1 Value Function

Case	HRR Rate	U1 Avg P_{cc}	U2 Avg P_{cc}	U1 Trk Rate	U2 Trk Rate
00	0.0864	0.4371	0.5747	0.7746	0.5582
01	0.0988	0.3754	0.5866	0.7606	0.5831
02	0.1111	0.6168	0.5960	0.8316	0.5433
03	0.1111	0.6168	0.5960	0.8316	0.5433
04	0.1111	0.6168	0.5960	0.8316	0.5433

Table 2 shows the corresponding performance results using only level 2 value functions. It can be seen that the classification performance improve significantly as compared to the level 1 performance. For example, for case 10, since the emphasis (i.e., valuation) is on unit 1, the resulting sensor strategy improves the unit 1 classification performance significantly from 44% to 96%.

Table 2: Performance Results Using Only Level 2 Value Functions

Case	HRR Rate	U1 Avg P_{cc}	U2 Avg P_{cc}	U1 Trk Rate	U2 Trk Rate
00	0.0864	0.4371	0.5747	0.7746	0.5582
10	0.1235	0.9671	0.3290	0.9007	0.4016
20	0.1235	0.7191	0.6369	0.8221	0.6015

Table 3 shows the performance results using both level 1 and level 2 value functions. It can be seen that the classification performance values are similar to those obtained using only level 2 value functions. For example, in cases 11 through 14, since the emphasis is on unit 1, the unit 1 classification performance is similar to that of case 10. However, for cases 21 through 24, only case 21 performs similarly to case 20—the others perform significantly worse in being able to classify unit 2. This is because, interestingly enough, unit 2 includes only target class 1. By adding level 1 track value function of the other target class (not including in unit 2) not only does *not* help in classifying unit 2, but it also manages to confuse the sensor manager and subsequently deteriorates the performance significantly.

Table 3: Performance Results Using Both Level 1 and Level 2 Value Functions

Case	HRR Rate	U1 Avg P_{cc}	U2 Avg P_{cc}	U1 Trk Rate	U2 Trk Rate
10	0.1235	0.9671	0.3290	0.9007	0.4016
11	0.1235	0.9669	0.3344	0.9005	0.3994
12	0.1358	0.9676	0.3290	0.9050	0.4016
13	0.1358	0.9639	0.3348	0.8906	0.4012
14	0.1481	0.9638	0.3283	0.8944	0.3960
20	0.1235	0.7191	0.6369	0.8221	0.6015
21	0.1235	0.7039	0.6373	0.8211	0.6032
22	0.1235	0.7472	0.5182	0.8320	0.5231
23	0.1235	0.7472	0.5182	0.8320	0.5231
24	0.1235	0.7472	0.5182	0.8320	0.5231

Another useful comparison is to determine the possible benefit of adding level 2 fusion information to level 1 fusion information. This comparison is particularly relevant because use of level 1 (target-based) valuation can be considered to be the typical baseline or current operating mode for most sensor fusion systems. In order to perform this comparison, we extracted comparable portions of Table 1 and Table 3 and listed these values in Table 4, where the comparison is between case 0n with cases 1n and 2n, where n = 1, …, 4. Note that, in all 4

cases, the unit 1 P_{cc} values improve when adding level 2 fusion information. However, the unit 2 P_{cc} values only improve when adding level 2 information that emphasizes unit 2 valuation, and we can see that cases 22, 23, and 24 actually perform worse (with respect to unit 2) than cases 02, 03, and 04 respectively. One may wonder why adding a level 2 value function for the specific unit can actually deteriorate the performance. Again, since unit 2 consists of only target type 1, the combination of "inconsistent" unit level value and target level value (such as 22, 23, and 24) simply will not improve classification performance. Also, since unit 1 has two targets of type 1, and one of type 2, the P_{cc} values do increase when adding level 2 information.

Table 4: Summary of Performance Results

Case	HRR Rate	U1 Avg P_{cc}	U2 Avg P_{cc}	U1 Trk Rate	U2 Trk Rate
01	0.0988	0.3754	0.5866	0.7606	0.5831
11	0.1235	0.9669	0.3344	0.9005	0.3994
21	0.1235	0.7039	0.6373	0.8211	0.6032
02	0.1111	0.6168	0.5960	0.8316	0.5433
12	0.1358	0.9676	0.3290	0.9050	0.4016
22	0.1235	0.7472	0.5182	0.8320	0.5231
03	0.1111	0.6168	0.5960	0.8316	0.5433
13	0.1358	0.9639	0.3348	0.8906	0.4012
23	0.1235	0.7472	0.5182	0.8320	0.5231
04	0.1111	0.6168	0.5960	0.8316	0.5433
14	0.1481	0.9638	0.3283	0.8944	0.3960
24	0.1235	0.7472	0.5182	0.8320	0.5231

5 Summary

In this paper, we have presented an approach for dynamically choosing both sensor mode and pointing direction based on both level 1 and level 2 fusion information. This approach not only will provide for adequate object identification and tracking performance, but also can provide the ability to be able to identify higher-level entities such as convoys. Thus far, we have analyzed algorithm performance on only a very limited set of ground truth data; in order to completely validate algorithm performance, it will be necessary to implement the algorithm in a higher fidelity modeling environment, including more complex algorithms for the tracker and SRM.

References

[1] J. Hill and K.C. Chang. Improved Representation of Sensor Exploitation for Automatic Sensor Management. In *Proc. Sixth Int. Conf. Information Fusion,* pages 688–694, Cairns, Queensland, Australia, 7 July–10 July 2003. Int. Soc. Information Fusion, 2003.

[2] T. Allen, D. Castañon, and R. Washburn. Stochastic Dynamic Programming for Farsighted Sensor Management, Phase II SBIR Final Report, Technical Report TR-974, ALPHATECH, Inc., Burlington, MA, 2000.

[3] J. Johnson and R. Chaney. Recursive Composition Inference for Force Aggregation. In *Proc. Second Int. Conf. Information Fusion.* Int. Soc. Information Fusion, 1999.

[4] D.P. Bertsekas and J. Tsitsiklis. *Neuro Dynamic Programming*, Athena Scientific, Belmont MA, 1996.

[5] D.P. Bertsekas. A Note on the Robust Calculation of Rollout Policies, No. LIDS-P-2392, Lab. for Information and Decision Systems Report, Massachusetts Institute of Technology, Cambridge MA, 1997.

[6] D.P. Bertsekas and D.A. Castañon. Rollout Algorithms for Stochastic Scheduling", *Heuristics*, Vol. 5, 1999.

[7] R. Washburn and J. Fox. Semi-Annual Technical Report: Continuous Tracking of High-Value Targets Using Multi-Mode Radar, Phase II SBIR Final Report, Technical Report TR-1031, ALPHATECH, Inc., Burlington, MA, Sept. 2001.

[8] F. White. Data Fusion Lexicon, Joint Directors of Laboratories, Technical Panel for C3, Data Fusion Subpanel, Naval Ocean Systems Center, San Diego, 1987.

[9] A. Steinberg, C. Bowman, and F. White. Revisions to the JDL Data Fusion Model, *Proceedings of SPIE, Sensor Fusion: Architectures, Algorithms, and Applications III*, 1999.