Top-Down Abduction for Behavior Detection in GMTI Data

Jacob Crossman, Michael Quist, Richard Frederiksen, and Pat McLaughlin
SoarTech, 3600 Green Ct. Suite 600, Ann Arbor, MI 48105
jcrossman@soartech.com

Abstract – Multi-hypothesis, kinematic trackers are the state-of-the-art in automated GMTI data processing. These systems are not designed to recognize long duration behaviors and under complex conditions these systems often produce track snippets. Our approach, which we call Cognitive Fusion because of it's structural similarity to human analysis methods, reframes the problem from one of tracking all entities all of the time to one of tracking only behaviors the user cares about. CFUS adds value to tracking and fusion under real-world conditions that include moderate contact densities, unpredictable target motion, deception, and unreliable sensor returns. CFUS applies an abductive reasoning approach that combines hypotheses projection, contextual reasoning, and dynamically constructed Hidden Markov Models (HMMs) to find and track instances of hypothesized behavior in GMTI data. We demonstrate, using simulated data, how our algorithms can be used to find complex behaviors in cluttered data sets and can significantly reduce association false positives.

Keywords: level 2/3 fusion, abduction, tracking, GMTI, contextual reasoning, behavior recognition

1 Motivation: Understanding Behavior in Complex Environments

A typical problem faced by analysts in many disciplines is an increasing quantity of sensor data without corresponding improvements in tools to make sense of this data. Often, analysts sift through large amounts of data to recognize threatening or abnormal patterns of behavior in a specific region. A frequent component of this behavior is tracking of vehicular or individual motion amidst clutter. While analysts can often do this well, it can be quite tedious and impractical for very large data sets. Doing this type of detailed analysis over wide regions and long periods is difficult and means that most data goes unprocessed and many events dependent on those behaviors go unnoticed. Tools are needed that help reduce the analyst's workload and that do a better job of supporting their workflow.

Much prior and ongoing research and development has gone into improving sensor data processing. Most of this research has focused on tracking entities through space and time using tools referred to as *trackers*. There are several of these available for tracking geo-spatial sensor data such as radar [1] and video [2]. Trackers tend to share the following characteristics:

1. They are bottom up: they start with sensor data and incrementally piece together a larger pattern, called a "track."
2. They are relatively context free: they typically use sensor data from a single sensor, occasionally including a few physical constraints (e.g. roads).
3. They are batch oriented: they expect data to come in, and without user intervention, they generate output tracks for all entities assumed to exist.

The most advanced of these tracking systems integrate kinematic models with multiple motion hypotheses [3,4,5]. They can achieve very good results under controlled conditions where sensor returns are reliable, motion is consistent, and vehicle densities are low. However, in real-world conditions where these conditions do not hold, they can fail to generate tracks of a useful length.

The root cause of most abbreviated and incorrect tracks is *confusers*. For a variety of reasons, a tracker may be forced to decide between multiple associations for a target. These reasons include missed/false sensor detections, target location error, erratic target movement (e.g. stops and sharp turns), and sensor contacts from other nearby vehicles. Because these trackers are bottom-up and relatively context free, they have very little information on which to make a good decision, and often make the wrong one. They select the most likely among several close choices and errors propagate forward in time, and thus the tracks become more and more incoherent (Figure 1).

In this paper we focus our attention on data derived from Ground Moving Target Indicator (GMTI) sensors. GMTI data presents unique challenges [1]; however, the core issue of making better decisions under confusion is generalizable to any type of spatio-temporal data, be it video, radar, or another modality.

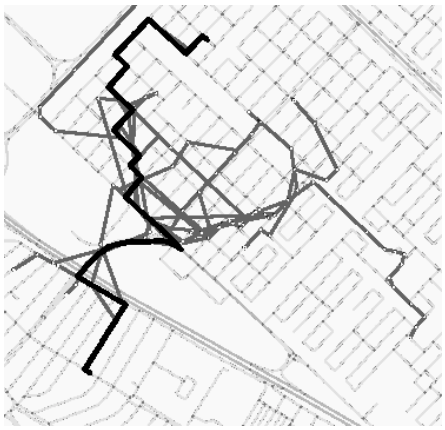

Figure 1: Tracks generated by a tracker (light lines) for a target vehicle (dark line).

2 Approach: The CFUS System

Our approach is inspired by our interactions with analysts and watching how they do their jobs. Analysts often do a fine job of tracking individual targets manually when there are few targets on which to focus. However, automated tracking systems often cannot produce sufficiently long tracks to be useful if the environment is complex. We observed that analysts process GMTI data far differently than automated tracking algorithms. They:

1. Do not try to track everything. They use triggers (e.g. unusual sensor detections, or external report) to indicate where to focus attention,
2. Use kinematic cues (e.g. by looping displays) but also use other contextual sources (e.g. knowledge of which roads are frequented by which groups),
3. Recognize patterns learned from lengthy experience in the same location
4. Start with an idea of the types of behaviors and patterns they are interested in, and look for those.

These differences led us to our approach, which tips the tracking problem on its head. We call this approach *Cognitive Fusion* (or CFUS) because it is based on how humans perform this task.

The CFUS approach starts with a model of one or more behaviors that the user is interested in observing, e.g. diversion around a checkpoint, abnormal routes between a given beginning and end area, or border crossings in highly obscured areas. From this model it seeks to infer instances of that behavior from the data using *abduction* [6]. The difference is subtle, but very important. These *behaviors of interest* (BOIs) provide CFUS with a context within which to interpret data and give it a way to estimate how a target might behave in important situations.

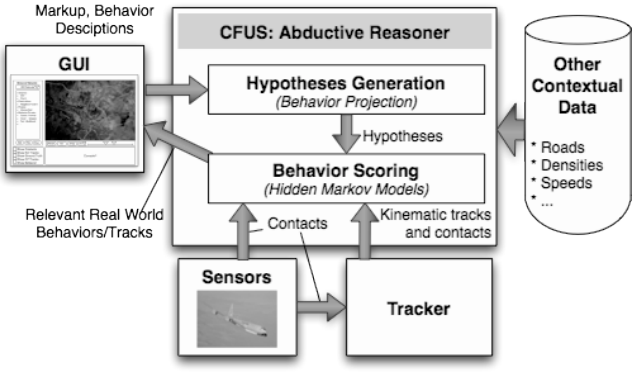

Figure 2: CFUS Architecture

In addition to helping see patterns in clutter, this approach has two other advantages. First, the behavior pattern is a convenient abstraction from which to engage interactively with the user. The user, for example, can easily provide contextual constraints such as the time of day, locations to avoid, and places where sensors are likely to fail. This human-provided knowledge can then be used as context to make the abduction process more accurate. Second, by limiting the problem to detection and tracking of user BOIs, we limit the computational breadth of the system and allow more resources to be used on deeper analysis that improves the quality of the results.

The caveat is that CFUS will not create tracks for every entity or activity, and will make some incorrect inferences. However, our experimental results show that CFUS provides much better, less confusing results in important cases. The result is a higher "signal to noise" ratio for the user.

Figure 2 shows a simplified CFUS architecture. CFUS is an abductive reasoning engine, and as such has two core components – a hypotheses generator and a scoring engine. Additionally, CFUS has an interface from which to gather contextual constraints and other contextual data and presents the user with results.

CFUS is implemented to work in conjunction with a traditional tracker, but this is not necessary in general. It is a multi-INT fusion system, capable of incorporating data from multiple sources. For example the current implementation ingests various kinds of data: sensor contacts, tracks, roads, and statistical information about the accuracy of the tracker under various conditions. The approach is general enough to support a wide variety of data types.

The processing flow is straightforward. The user presents BOIs to the system. CFUS uses a *projection* process (e.g. estimating routes a vehicle might take) to instantiate multiple hypotheses defining how the behavior would execute in the given environment. These hypotheses are scored using Hidden Markov Models (HMM) [7] to estimate the degree to which the incoming data stream reflects the behavior they define.

2.1 Generating Hypotheses and Corresponding HMM

Our goal for the CFUS implementation was to flesh out and evaluate the core concepts outlined above. We developed a basic behavior representation, a simple graphical interface, a basic path planning projection

algorithm, and an HMM generation and scoring algorithm.

Behaviors were specified either as a start and end point or as a route. When given a start and end point, the projection algorithm used an A* route planner to generate alternative routes for the vehicle between these points. Along each route, whether generated or given, the projection system assigned expected speeds to road segments based on the type of road (e.g., main road, highway, or residential).

The projection process has an important constraining effect on the problem. It reduces the space that the CFUS algorithms must address by focusing on the behavior that the user cares about. This process also has the effect of constraining the types of false positives CFUS will produce. Many false positives due to turns, crossing traffic, and noisy sensor data near the route are easily eliminated as not fitting the user's BOI. When CFUS does make a false positive association it typically involves a vehicle behavior similar to the BOI, which we call *masking behavior*.

Given one or more prospective routes, CFUS dynamically generates an HMM to recognize behavior along that route. This HMM is designed to track vehicle position within road segments and was implemented as a discrete-time Markov model over the product space R x (D + {*None*}), modeling both the motion and the tracking of a single vehicle. Here $R=\{R_i\}$ is a set of discrete, non-overlapping regions; R_i consists of all points closest to road segment i. D is the set of GMTI detections, and *None* represents the case where the sensor has so far failed to detect the vehicle.

Therefore, each state of the HMM consists of a variable pair (R_i, d_j). Sensor data arrives at time intervals [t, t+dt) defined by the sensor's *revisit rate*. The legal transitions within the Markov model are of the following three types:

1. Unobserved motion: (R_i, *None*)→(R_j, *None*);
2. Observed motion: (R_i, d_{old})→(R_j, d_{new}), where d_{old} is a detection in a prior time interval (or *None*) and d_{new} is a detection in the current time interval;
3. Previously observed motion: (R_i, d_{old})→(R_j, d_{old}), where d_{old} is an old detection.

A path consisting only of steps of this type constitutes a specific hypothesis as to the motion and tracking of a single vehicle, and the Markovian evolution of the state vector gives time-dependent probabilities for each hypothesis.

Our algorithm is designed to assign state probabilities based on an arbitrary set of observables. In this case we used the road network, the tracker's output (i.e. the detections it associated), the expected maximum speed for each route segment, the conditional probability that the tracker output is correct, and sensor contact data.

Any detection at a particular time constrains the position of the vehicle at that time (strongly, if the sensor error is reasonably low). We therefore assumed that R_i and d_{new} would be compatible for each point (R_i, d_{new}) for each transition of type 2. At subsequent time steps, the vehicle might move without being detected again (via transitions of type 3), and so points (R_j, d_{old}) where R_j and d_{old} were incompatible would become accessible. Our association of old detections with new detections drew heavily from the predictions of the tracker; therefore transitions of type 2 were deemed very likely whenever the tracker placed d_{old} and d_{new} on the same track, and were deemed unlikely whenever the tracker placed them on different tracks.

All vehicle motion was constrained by a global maximum speed and biased toward motion along preferred roads. A transition matrix between road segments was used to represent the vehicle's preferences on encountering each intersection. The number of possible transitions of types 1 and 3 is limited to the number of road segments that can be reached from a source road segment in a full time step, while the number of possible transitions of type 2 is limited to the number of new detections per time step multiplied by the number of regions compatible with each detection. These properties were used to determine the selection of an appropriate time step.

2.2 Generating Transition Probabilities

Transition probabilities for the transitions of types 1 and 3 were generated based on an input transition matrix representing the vehicle's preferences on encountering each intersection. Upon entering an intersection at the end of road segment i, the vehicle would turn around completely with a fixed, small probability e, or leave the intersection at the beginning of road segment j with probability M_{ij}. These rules were used to calculate the probability that a vehicle on road segment i at the beginning of a time step would be on road segment j at the beginning of the next time step. The elements of this large matrix were calculated only when needed; once calculated they were cached for re-use.

A related set of matrix elements were defined for transitions of type 2, in which the exact (up to sensor error) position of the vehicle was observed in the current time step. Here we projected the detection onto the nearest road segment, assumed equal probabilities of traveling in either direction, and applied the same driving preferences to calculate where the vehicle might be at the end of the time step. These calculations were also cached.

Transitions were further weighted based on the predictions of the tracker. For any two detections the tracker's linkage assessment could fall into one of three categories: positive, neutral, and negative. A positive assessment was given when the tracker placed the two detections on the same track. A neutral assessment was given when the tracker placed the two detections on different tracks, but one of those tracks ended before the second track started. (A neutral assessment was also given when the tracker failed to associate either detection with any track.) Finally, a negative assessment was given in the remaining case, in which the tracker placed two

detections on distinct tracks covering overlapping time intervals. In our implementation, we assigned a multiplicative factor to each assessment, with positive > neutral > negative. The positive factor was based on the conditional probability of the tracker being correct in similar situations.

2.3 Scaling and Hypothesis Management

Scalability can be a problem with a naïve implementation of the Markov model described above. Without a cutoff mechanism, many low-weight hypotheses consisting of a partial track followed by many time steps of unobserved motion would build up over time, continuing to fan out through unobserved motion of the vehicle. This is in some sense an artifact. These competing hypotheses are equivalent as far as vehicle detections go, and should be combined into a single hypothesis at the end of the run. However, it is not tractable to maintain them all until that point. Our workaround for this problem was to: first, select high-weight hypotheses at any point in time, rather than waiting until the end of the run, and second, prune the set of "live" hypotheses to a fixed maximum number (say, a few thousand). These two modifications at least partially compensate for one another, and tame the combinatorial explosion.

Increasing the number of vehicles can also cause scaling issues. The set of high-weight hypotheses will tend not to be disjoint, and in fact may well consist of a large number of slight variations on a single track (e.g., hypotheses differing by a single detection in the middle of a track). This means that tracks for additional vehicles, although they should be present in the set of hypotheses, are not easily retrieved and, in fact, are likely to be discarded by the aforementioned pruning. A partial fix for this is to make the pruning process *uniform*: by looking for the most likely hypotheses conditioned on the final detection. This is only a partial fix, however, because it dramatically increases the time and space requirements of the method, and it still considers large numbers of "similar" hypotheses that differ only in endpoints. Given these two methods, our algorithms were able to execute about three times faster than real time on a typical desktop PC in our experiments.

3 Experiments and Results

We prepared an experiment to measure CFUS performance in a wide range of tracking situations. We were interested in testing our claims that: first, the CFUS system can detect accurate and coherent tracks for BOIs and, second, that the CFUS system can reduce errors due to false positives in contact association. We compared our results to those generated by Lockheed Martin's state-of-the-art multi-hypothesis tracker configured to use road network data for improved ground vehicle tracking. The results are not meant to show that the CFUS system is systemically *better* than the tracker, as they solve different problems. Instead we demonstrate that CFUS is a significant value-add for behavior recognition especially for tracking a target from an origin to its destination.

3.1 Test Environment

For a test environment we used a vehicle/GMTI sensor simulation, provided by our partners at Lockheed Martin and the aforementioned multi-hypothesis tracker. We created multiple scenarios across 7 complexity categories varying the following parameters:

- **Density**: the number of vehicles moving in the scenario. Scenarios were over a $2km^2$ region and lasted about 25 minutes.
- **Revisit**: the revisit interval of the sensor in seconds (time between GMTI sensor updates).
- **Probability of Detection**: the probability that a vehicle will be detected during a sensor update.
- **TLE**: the spatial error of the sensor.

The cases increase in difficulty from A, the easiest, to G, the most difficult. F and G are very challenging cases meant to stress test the CFUS system. This range covers various real-world cases of interest and makes these simulated tests useful for estimating real-world performance.

To evaluate the system, we created one "target" behavior per scenario run. This target behavior consisted of a route to be taken between two random points at some random time during the scenario. The CFUS task was to detect and then track the vehicle doing this behavior.

3.2 Example of Good CFUS Performance

Figure 3 shows our target behavior for a case where CFUS performs well. Though we show only one case here, this case represents a common pattern. The route starts in a residential area to the north and then passes through a heavy traffic area near the exit/entrance ramp to the southwestern highway. This exit/entrance ramp area acts as a choke point for the simulated traffic and frequently exhibits congestion. This area, along with sharp turns in the area, makes it a difficult area for the tracker.

Figure 3 also shows a comparison between the "best" track produced by the tracker for the target vehicle (Figure 3b) and the best track produced by CFUS (Figure 3c). The tracker results show a track that appears to be largely correct south of the highway but misses most of the

Table 1: Scenario Settings for Simulated Runs

	Density	Revisit	P. Detect	TLE
A	40	Fast	0.75	Small
B	50		0.70	
C	60		0.65	
D	70	Moderate	0.60	Medium
E	80		0.55	
F	90		0.50	
G	100	Slow	0.45	Large

behavior north of the highway, instead doubling back and passing back through the exit/entrance ramp. An analyst using this track to search for the target behavior would have to spend considerable time removing the incorrect parts of this track and extracting parts of other tracks (see the earlier Figure 1) to piece together the complete behavior.

The results in Figure 3c illustrate the full promise of this approach. CFUS is able to track the vehicle for almost the entire length of the route, even through the area of significant clutter by the entrance/exit ramp area. An analyst looking for this type of behavior would see an obvious, clean track and would be able to focus on it, and assess its validity quickly.

3.3 CFUS Limitations

While this approach is excellent at ignoring pre- and post-track clutter and also does well with crossing clutter, it has some difficulty with *masking behavior*, that is, vehicle behavior similar to the BOI. When CFUS is wrong we see three common failure patterns:

- **Wrong target, correct behavior:** a different target exhibits the same or very similar behavior to the target vehicle and the tracker picks this up. This is not a flaw with the algorithm but rather a limitation of the experimental technique in that we did not constrain the scenario to produce only one target behavior instance.
- **Target loss due to masking behavior**. For especially long and short behaviors the probability of a significant portion of another vehicle's track mimicking the target behavior becomes high. In these cases the other vehicles confuse CFUS algorithms and it produces many disjoint hypotheses, some of which contain the target vehicle.
- **Poor tracking of the early parts of long tracks**. This was the result of sub-optimal (too high) weightings for probability *e*, the probability of a missed detection in the HMM algorithms.

Case 3 can likely be resolved by changing how missed detection probabilities are computed, taking into account the number of nearby confusers. Case 2 is a bit more challenging and requires more investigation. Our current idea is to increase the amount of non-sensor (i.e. contextual) data CFUS uses to distinguish between alternatives. Other sensors (e.g. video) could be useful here as could additional statistical data (e.g. turn probabilities).

3.4 Statistical Results

We also evaluated CFUS performance by looking at contact association metrics across over 1,400 runs, 200+ for each of the cases (rows) shown in table 1. For each run we collected a wide array of metric data, but settled on two key measures: Sensitivity and Positive Predictive Value (PPV). These measures concisely summarize the ability to recognize contacts as part of a track and eliminate false positives.

- **Sensitivity** = TP / (TP + FN), % of correct associations detected
- **PPV** = TP / (TP + FP) , % reported associations that were correct

Here TP, FN and FP refer to the standard classification metrics *true positives*, *false negatives* and *false positives* respectively. These values were computed by measuring the accuracy of pairwise sensor contact associations. These pairwise associations are the fundamental operation of a tracking system as they define which contacts the system believes belong to the same vehicle.

In our results, we compare three cases:

1. The kinematic tracker's best scoring track with results filtered for the target vehicle.
2. CFUS best scoring track.
3. The track that CFUS ranked as best while running. We call this the *top* score

For these cases, we computed *best* using the Matthews Coefficient.

It is useful to think of CFUS results as analogous to a search engine. The user provides a BOI and the system seeks to find it in a large data set. As with all search engines, the top value is not always best; however, analysis showed that the best output, or a nearly identical track, was almost always in the top 10-20 results.

3.5 Reduction of False Positives

Figure 3a,b,c: Original vehicle path (left, 3a) best kinetic tracker result (center, 3b) and CFUS result (right 3c)

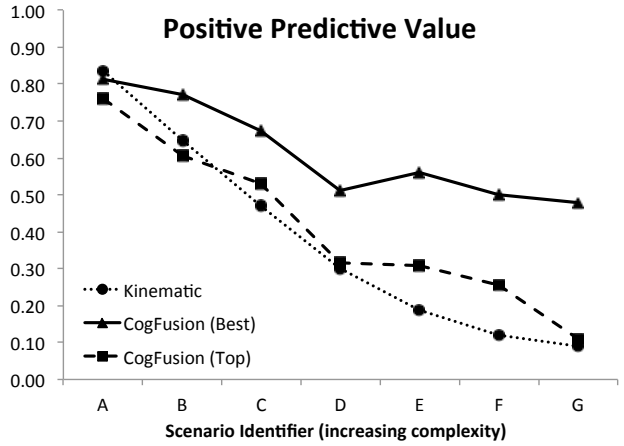

Figure 4: CFUS Positive Predictive Value Results

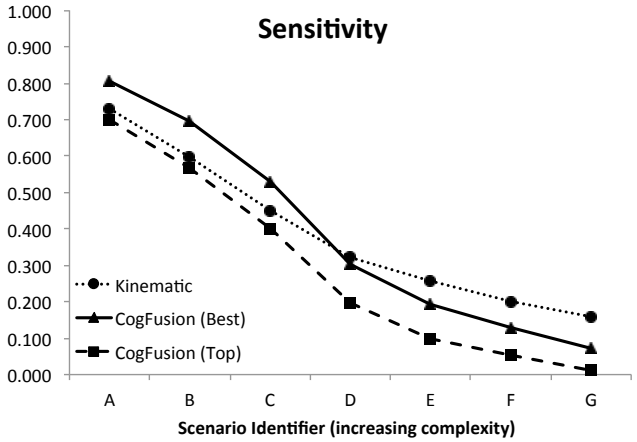

Figure 5: CFUS Sensitivity Results

One common failure pattern in traditional trackers is a tendency for tracks to diverge from a target and for tracks from multiple vehicles to be merged together. This results in connected sequences of valid tracks, but altogether it forms an incoherent behavior pattern that is difficult for a human to interpret (Figure 1). PPV in these cases is quite low, since it includes many incorrect associations between different vehicles.

Figure 4 shows the positive predictive value of the associations made by both the tracker and CFUS. As expected positive predictive value of the tracker decreases as the difficulty of the tracking problem increases. It also shows that the best tracks produced by CFUS score considerably better especially in the most difficult tracking cases (D through G). The top ranked results from CFUS show a more limited improvement.

As we discussed earlier, this is mainly because the CFUS algorithms are successfully eliminating much of the track jumping or switching we see in the kinematic tracker. The extra context given by the expected route, the road connectivity calculations, and the confuser statistics are successfully eliminating many false positives.

The sensitivity results shown in Figure 5 indicate that CFUS is also removing a fair number of true positives. But even in the hardest tracking case the best tracks are still including 30-40% of the detections associated with the behavior of interest. So the tracks, while sparser, are very pure. For use cases where the goal is recognizing specific subsets of behavior in a wider, cluttered region this tradeoff would be worthwhile.

4 Conclusions and Next Steps

CFUS is an example of a new breed of fusion system that incorporates the user as a teammate, focusing the system on high value behaviors of interest. The system's job then, is to find and monitor these behaviors, much like a search engine. CFUS and related systems present a compelling value proposition in domains where it is important to find "needle's in haystacks," such as those facing analysts in asymmetric warfare and homeland defense.

Experimental results suggest that CFUS can find specific patterns of behavior in cluttered environments, making them clear and obvious. Furthermore, false positives are greatly reduced and limited to cases where similar behaviors are occurring – cases that an analyst is likely going to want to examine closely anyway. The principle limitation is that CFUS does not track everything, only specified behaviors, and, in our experiments, loses some sensitivity when compared to kinematic trackers. However, it is not clear whether this sensitivity reduction is a fundamental characteristic of the approach or an implementation issue.

We continue to work to extend this concept and system. Our near term objective is to define an interactive interface and representation to specify more sophisticated behaviors. We are also working on extending the core abduction engine. The projection processes used in CFUS are simplistic and we are interested in incorporated more sophisticated methods like those we developed in [8] to allow abduction of more complex (e.g. multi-part) behaviors. We are also seeking ways to improve the hypothesis management, scoring and pruning mechanisms to resolve issues found with the current CFUS implementation. Finally, we are interested in a hybrid extension integrating learned patterns, eg. from statistical methods, with abducted patterns provided by users.

5 Acknowledgements

This work was funded under a Small Business Innovative Research Contract from AFRL: FA8750-08-C-0201. Findings and opinions are those of the authors and not necessarily those of the sponsor. Special thanks to Ms. Jessica Halpin at AFRL for her help in understanding the real world issues facing analysts. Thanks to Ellen Maclean for her technical support in adapting the Lockheed Martin fusion engine.

References

[1] O'Hern, B. and M. Kozak. 2009. *Baseline Tracker (BRAT)*. The Air Force Research Laboratory. http://www.darpa.mil/i2o/solicit/baa/BAA-10-05_AFRL.pdf

[2] Jones, R., D.M. Booth, and N.J. Redding 2006. *Video Moving Target Indication in the Analysts' Detection Support System*. Australian Defence Science and Technology Organisation: Edinburgh, South Australia.

[3] Benameur, K., B. Pannetier, and V. Nimier, 2005. *A comparative study on the use of road network information in GMTI tracking*, in *Eighth International Conference on Information Fusion*. Philadelphia, PA.

[4] Blackman, S.S. 2004. *Multi Hypothesis Tracking for Multiple Target Tracking*, in *IEEE A&E Systems Magazine*.

[5] Koller, J. and M. Ulmke. 2005. *Multi hypothesis track extraction and maintenance of GMTI sensor data*, in *Eighth International Conference on Information Fusion*. Philadelphia, PA.

[6] Menzies, T. 1996. *Applications of abduction: knowledge-level modeling.* International Journal of Human Computer Studies, **45**: p. 305-335.

[7] Rabiner, L. 1989. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. IEEE Proceedings, 77(2): p. 257-285.

[8] Crossman, J., et al. 2010. *Integrating dynamic social networks and spatio-temporal models for risk assessment, wargaming and planning*, in *The Network Science Workshop*. West Point, NY.