2000 MSS National Symposium on Sensor and Data Fusion
June 20 – 22, 2000

Multiple-Target Tracking and Data Fusion via Probabilistic Mapping

June 2000

K. Mike Tao, Ronald Abileah, and John D. Lowrance
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025

ABSTRACT

A new approach is taken to address the various aspects of the multi-sensor, multi-target tracking (MTT) problem in dense and noisy environments. Instead of fixing the trackers on the potential targets as the conventional tracking algorithms do, this new approach is fundamentally different in that an array of parallel-distributed “trackers” is laid in the search space. The difficult data-track association problem that has challenged the conventional trackers becomes a nonissue with this new approach. By partitioning the search space into “cells,” this new approach, called PMAP (probabilistic mapping), dynamically calculates the spatial probability distribution of targets in the search space via Bayesian updates. The distribution is spread at each time step, following a fairly general Markov-chain target motion model, to become the prior probabilities of the next scan. This framework can effectively handle data from multiple sensors and incorporate contextual information, such as terrain and weather, by performing a form of evidential reasoning.[1, 2] Used as a pre-filtering device, the PMAP is shown to remove noiselike false alarms effectively, while keeping the target dropout rate very low. This gives the downstream track linker a much easier job to perform. A related benefit is that with PMAP it is now possible to lower the detection threshold and to enjoy high probability of detection and low probability of false alarm at the same time, thereby improving overall tracking performance. The feasibility of using PMAP to track specific targets in an end-game scenario is also demonstrated. Both real and simulated data are used to illustrate the PMAP performance. The PMAP algorithm is parallel distributed in nature; for serial computer implementation, fast algorithms have been developed. Some related applications based on the PMAP approach, including a spatial–temporal sensor data fusion application and a gray-scale video sequence stacking application, are also discussed.

1.0 INTRODUCTION

This paper discusses tracking multiple targets in dense, noisy environments and the related contextual and sensory data fusion applications. In the conventional approach, tracking algorithms such as the multiple-hypothesis tracker (MHT),[1] the probabilistic data association filter (PDAF),[1] and the nearest-neighbor tracker[1] are used to (1) eliminate or discount false alarms and (2) develop tracks on targets. These algorithms fix a tracker (or filter) on a potential target or hypothesized track branch, and update the tracker with new measurements in order to stay on track. Since in a dense environment there are many measurements in a given neighborhood (some are false alarms and some may have their origins in other targets), finding the *right* measurements to keep a tracker on track becomes a very challenging and crucial step. In MTT literature, this is known as the data association (DA) problem.[1] The difficulty related to DA increases both with the density of the target environment and with less frequent scan rates.

Instead of fixing a tracker on each potential target as conventional algorithms do, the PMAP differs fundamentally by fixing an array of distributed “trackers” in the target environment space. That is, the environment

[1] Blackman, S.S., *Multiple Target Tracking with Radar Applications*, Artech House, Norwood, MA, 1986.
[2] Lowrance, J.D. et al., “A framework for evidential-reasoning systems,” in *Uncertain Reasoning*, G. Shafer and J. Pearl (ed.), pp. 616 – 618, Morgan Kaufman, San Mateo, CA, 1990.

.

is divided up into cells. In each cell, there is a tracker in the sense that it determines the likelihood of whether a target is present in this location, based on the current measurement and the past history in the neighborhood. This way, the DA problem that has burdened the conventional algorithms is avoided by the PMAP approach. The method of likelihood determination is based on the Bayes' rule; and to account for target motion between scans, a Markov-chain model spreads the likelihood to neighboring cells before the next measurements are taken. Used as a noisy false-alarm pre-filter, the PMAP can identify the most likely target areas, after a few frames, and edit the noisy raw data accordingly. As shown in this paper, the PMAP algorithm also provides predicted target locations in the next frame and the likely links to target hits in the previous frame—the basic elements of tracking. A method of using PMAP as a tracker for a specific target following in an end-game scenario is also discussed. The spatial probabilistic representation adopted by PMAP is amenable to handling multiple-sources of spatial information with asynchronous reporting.

This paper begins by focusing on the PMAP algorithm description, which is followed by some analyses and simulated tracking results, using real as well as simulated data. The real data are GPS ground vehicle position data collected during large-scale vehicle exercises at Ft. Stewart, Georgia, involving multiple vehicles in various modes of travel, battle formations, and battle engagements. For the example data set, the area covered is about 20 km^2 in size. As a result of data dropouts, the effective probability of detection (P_d), in the example data set used, is about 0.7 to 0.8. Uniformly distributed random false alarms are added to simulate false reports; the false alarm rate (P_{fa}) is about 0.0008 (per frame). (For a resolution of 150 × 300 cells, this amounts to about 36 false alarms in each scan, or 1.8 false alarms per km^2 per scan.) The simulated data are based on $P_d = 0.7$ and a noisy $P_{fa} = 0.001$. In the simulated truth data set, there are two convoys, each with 10 targets, and a single target traveling by itself. Very good results are obtained for both real and simulated data cases. Virtually all away-from-target false alarms are eliminated, with little target loss. The analytical results offer theoretical insights to the PMAP algorithm in terms of its strengths, performance limitations, and parameter selections in systems design, and provide guidance for algorithmic variations and refinements. Multiresolution and multiscale PMAP variations are introduced as tools for fast computation, as well as for performance improvement.

2.0 MULTIPLE-TARGET TRACKING AND DATA FUSION—A PMAP PERSPECTIVE

This paper is concerned primarily with post-detection tracking; that is, a detection threshold has been applied to the continuous sensor measurements. The data the algorithms deal with are therefore *binary* in the sense that either there is a detection or there is not. The possibility of using PMAP in "track-while-detect" mode is indicated; in fact, a recent successful PMAP extension to a different but related application is stacking of gray-scale video image sequences.[3]

The PMAP algorithm has its roots in search theory,[4] a branch of military operations research. (It is also related to the nonlinear stochastic filtering approach in the recent literature[5]; the PMAP approach presented here is perhaps more computational in nature.) In the past, various versions of the PMAP algorithm have been applied to search and surveillance problems in different contexts, with a recent application involving fusing data from multiple sensors over space and time.[6] The search space is divided up into cells. Within each cell there is a probability of finding the object of interest. Bayes' rule[7] is used as a fusion mechanism to combine new observations (measurements) with prior probabilistic expectations. Since the object(s) being searched for are subject to motion, motion model(s) are used to predict the object probability distribution between observations. The predicted object probability distribution then becomes the prior probabilities for the next observation period. The motion models are often based on the Markov-chain[8] model as it describes transitions in a probabilistic setting, very compatible with the rest of the formulation.

[3] Tao, K.M., "Bayesian stacking of video image sequences," (in preparation), 1999.

[4] Stone, L.D., *Theory of Optimal Search*, Academic Press, New York, 1975.

[5] Musick, S., K. Kastella, and R. Mahler, " A practical implementation of joint multitarget probabilities," *SPIE AeroSense98 Proceedings*, Vol. 3374, pp. 26 – 37, 1998.

[6] Tao, K.M., *A Probabilistic Inference Approach to Monitor and Search*, Technical Paper, Applied Electromagnetics Laboratory, SRI International, Menlo Park, CA 1998.

[7] Hoel, P.G., S.C. Port and C.J. Stone, *Introduction to Probability Theory*, Houghton Mifflin, Boston, 1971.

[8] Hoel, P.G., S.C. Port and C.J. Stone, *Introduction to Stochastic Processes*, Houghton Mifflin, Boston, 1972.

Before describing the PMAP algorithms used in the MTT context, it is instructive to look first at a simplified problem: track a single target in a noisy environment. Before making any observations, we may have some prior expectations of where the target might be in the defined search space. For example, domain knowledge, such as terrain- or weather-related information, may favor certain areas over others. Otherwise, a uniformly distributed probability map over the search space may be used. With the first set of measurements, assume that three spots have positive indications of the target's presence. These could be the correct indication of the target's position at the time or they could be false alarms. Given the (potentially location-dependent) probabilities of detection and false alarm, applying Bayes' rule will allow the determination of target- presence probability at each location in the search space. Since by the problem definition there is only one target in the space and the target can only be in one place at a time, the target-presence probabilities, corresponding to mutually exclusive events, would have to add up to 1. Assuming the target speed range is known, a Markov motion model amounts to distributing the target presence probability at each location into a neighborhood area whose radius reflects the maximum speed of the target and the time lapse between observations. Certainly, the neighborhoods around the three "hot spots" would have much higher predicted probabilities than the rest of the search space. Then, the next observation is made: for the sake of discussion, assume that only one positive indication is observed and that it falls within one of the three "hot" neighborhoods. Applying Bayes' rule, using the observation over the *entire* search space would raise the target-presence probability at that location (say, location *x*) to a higher value, and probabilities in the rest of the search space would have very low values. Now, if this observation had an additional positive indication elsewhere in one of the three "hot" neighborhoods, then the probability at location *x* would more or less reduce to half of the high value it has enjoyed in the case of a single detection. The point to note here is that under the single-target assumption, these spatial probabilistic alternatives are *competing* for their share of the total probability (= 1.0), and the probability distribution tends to be sharpened by this competition.

However, this spatial probabilistic competition is somewhat lost when considering an MTT scenario because, in a general MTT situation, the number of targets present in the search space is potentially variable and unknown. Hence, the location probabilities in space do not represent mutually exclusive events, nor do they necessarily add up to 1 as they do in the single-target case. At least two problem formulation approaches are possible.

In the first approach, an attempt is made to normalize the total probability over space for each possible target, at each Bayesian update. Each cell is competing with other (neighboring) cells for a target's presence probability. This formulation could be computationally demanding and care must be exercised as the total number of targets is known. In the second approach, a cell does not compete with other cells for target-presence probability; rather, the competing probability event is "*no* target present" in the same cell. (Technically, only the measurement at a cell is used in the Bayes' rule to update the target-presence probability of that cell—similar to decentralized local feedback in large-scale system theory.) This is a computationally less-demanding approximation, and is the setting in which the current MTT-PMAP algorithm is implemented. Since the total number of targets is unknown and no attempt is made to normalize the total probability over the search space, the map is really a relative likelihood or pseudo-probability map. However, it is possible to derive analytical results regarding the achievable steady-state (pseudo-) probability value of a recurring target, as shown later in this paper. This result is very useful in providing a sense of scale for the calculated (pseudo-) probability map and a reference to which a meaningful threshold may be selected to highlight "hot" spots for attention.

The second formulation is simpler to implement and is detailed as follows. Denote $P_0(x = T)$ as the prior probability that a target is present in cell *x*, and $P_0(x \neq T)$ as the prior probability that no target is in cell *x*. Then, following Bayes' rule, the posterior probabilities that a target and that no target is in cell *x*, given a detection in cell *x*, are, respectively,

$$
\begin{aligned}
P(x = T \mid x = +) &= \frac{P(x = + \mid x = T)}{P(x = +)} P_0(x = T) \\
&= \frac{P_d}{P(x = +)} \; P_0(x = T) \\
P(x \neq T \mid x = +) &= \frac{P(x = + \mid x \neq T)}{P(x = +)} \; P(x \neq T) \\
&= \frac{P_{fa}}{P(x = +)} \; P(x \neq T) \quad ,
\end{aligned}
$$

where the notation x = + stands for a (+) detection in cell *x*. (Notice that, with this formulation, only the measurement at x is used to update the probabilities at x.) Also, in these equations, P_d and P_{fa} stand for probability of detection and probability of false alarm, respectively. For the case where there is no detection in cell *x*,

$$
\begin{aligned}
P(x = T \mid x = -) &= \frac{P(x = - \mid x = T)}{P(x = -)} P_0(x = T) \\
&= \frac{1 - P_d}{P(x = -)} P_0(x = T) \\
P(x \neq T \mid x = -) &= \frac{P(x = - \mid x \neq T)}{P(x = -)} P_0(x \neq T) \\
&= \frac{1 - P_{fa}}{P(x = -)} P_0(x \neq T) \quad ,
\end{aligned}
$$

where the notation x = – stands for no detection in cell *x*. Note that these two[9] (target presence and absence) probabilities are complementary probabilities (i.e., they represent disjoint events and add up to 1); the common denominator in each set of Bayes' rules is therefore the sum of the two numerators; that is,

$$
\begin{aligned}
&P(x = +) = P_d \, P_0(x = T) + P_{fa} \, P_0(x \neq T) \\
&P(x = -) = (1 - P_d) \quad P_0(x = T) + (1 - P_{fa}) \quad P_0(x \neq T) \qquad .
\end{aligned}
$$

Hence, there is no need to be concerned about computing the denominator, but simply to carry out the numerator computations and then normalize the probabilities for each cell.

Since the targets may move between observations, based on the updated posterior probability distribution, a motion model is used to predict the prior probability distribution for the next observation period. The most basic motion model is to distribute *uniformly* the posterior probability in each cell to its surrounding neighborhood. The (maximum) travel distance of the target between observations determines the size of the neighborhood. To incorporate the knowledge of minimum speed, a "donut-shaped" neighborhood can be used. Noting that this spreading of probability is a Markov-chain-type of transition model, the target motion model will often be referred to as the Markov motion model.

With more knowledge, the Markov spreading distribution can be made nonuniform, indicating target behavioral preferences. One such example is the use of terrain information and a form of evidential reasoning.[10] Another example is the use of velocity information. For instance, knowing that a target has been traveling from left to right, it probably makes sense to spread the target presence probability only to the right, resulting in more concentrated distribution. This velocity information can be estimated within the PMAP framework. A relatively simple and tractable approach is discussed here. First, the following question is posed: Given that a target is currently in cell *x*, what is the probability that the target was in a specific neighboring cell (say, *z*) during the last scan? The answer to this question is useful

[9] For the aforementioned gray-scale applications, (see note 3 above) more than two states are used for each cell.

[10] See note 2 above.

because, if the target travels in a straight line between frames, then knowing where the target was will help predict the target's location in the next scan. Again, Bayes' rule finds application in answering this question:

$$P(z(t-1)=T \mid x(t)=T) = \frac{P(x(t)=T \mid z(t-1)=T)}{P(x(t)=T)} P(z(t-1)=T) \quad ,$$

where the notation *x(t) = T* represents that, at time *t*, the target is in cell *x*, and, likewise, *z(t – 1) = T* stands for the event that the target is in cell *z* at time t – 1. A moment's reflection suggests that the candidate target locations (in the previous scan) are defined by the same spreading window used in the Markov motion model. Bayes' rule described above therefore applies to cells in that neighborhood within which the normalization is to be performed. Notice that the "prior" probability, *P(z(t – 1) = T)* in this case, is simply the posterior probability of cell *z* at time t – 1, and that *P(x(t) = T | z(t – 1) = T)* is simply the Markov transition probability obtained for time t – 1 and can be initialized as uniform transition. Therefore, the formula above is completely computable for each of the cells in the window neighborhood. By spatially "flipping" these computed probabilities with respect to the center of the window (i.e., the hypothesized current target location), one obtains the probabilities of the projected target locations; that is,

$$P(z^*(t+1)=T \mid x(t)=T) \cong P(z(t-1)=T \mid x(t)=T) \quad ,$$

where cell *z** is the mirror image of cell *z* with respect to the center of the window. Since the target may not travel in a straight line or in constant speed, the projected target location probabilities should be "blurred" (i.e., dispersed to a small neighborhood around each cell) for robustness. This can be done in many different ways. One way is to use a smoothing filter to smear the estimated location probabilities. In summary, using the projected target location probabilities with the Markov motion model will help direct the target search in a more focused manner, thereby sharpening the posterior distribution.

A flowchart of the PMAP algorithm is given in Figure 1. The process begins with a first-time prior probability map (set to 0.2 in the simulations), which can also take into consideration contextual information such as terrain suitability. This prior probability map is then fused with measurements using Bayes' rule to produce posterior probabilities. Using the previous posterior probability map, a Bayesian velocity estimate is made, which in turn influences the kernels of the Markov search model. Also affecting the search models here is the contextual information. The updated Markov motion models then spread and dilute these posterior probabilities to become prior probabilities of the next frame of measurements, and the process continues. After the Bayesian posterior probability update at each frame, the continuous posterior probability map can be thresholded to produce a cleaned-up *binary* target indication map for the downstream track linker algorithm (e.g., the MHT) to use. Simulation experiences have indicated that the results are not very sensitive to the selection of threshold level. Based on the simulated and the real data, a threshold setting at around 0.3 to 0.4 often gives very good results. The steady-state analysis discussed later provides a reference for appropriate threshold settings.

Before ending the discussion on the PMAP basics, a detail on the motion spread function is worth noting. Figure 2 depicts the spreading of the posterior target-presence probability (from each cell) to the corresponding neighborhood in the next frame's prior probability plane. This operation is performed for each cell, and a first inclination suggests that the accumulation of the distributed probabilities be accomplished by simple addition. If the spread function is identical for each cell and is unitary (unity distribution), then it can be shown that the projected prior probabilities are well defined—that is, between 0 and 1. However, a moment's reflection would suggest that, with nonhomogeneous spread functions, such projected probabilities may become greater than 1. For example, if two targets are in cells *x1* and *x2*, respectively, and assuming each has a 0.6 probability of moving to cell *x* and

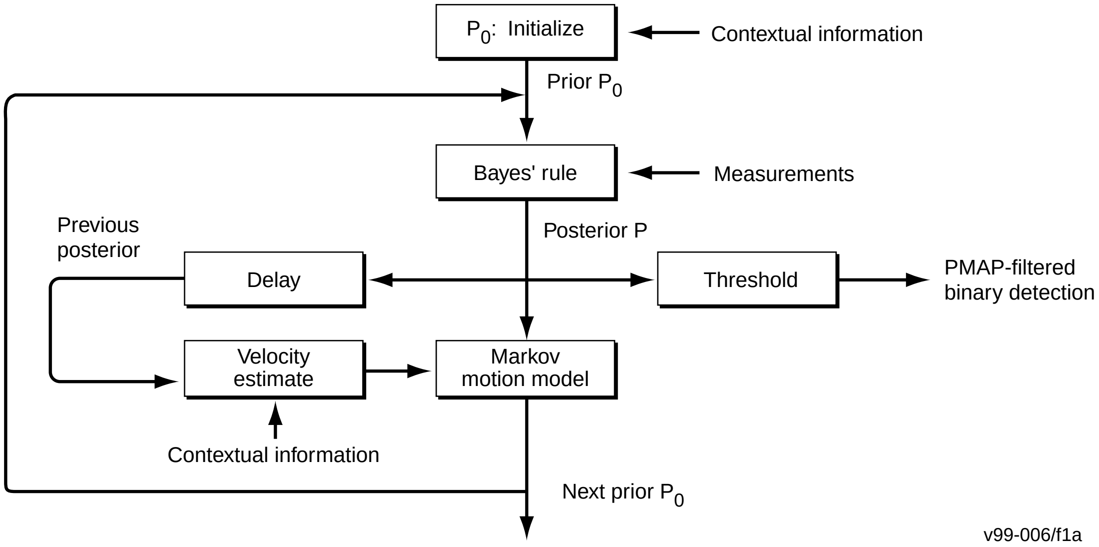

Figure 1. PMAP flowchart.

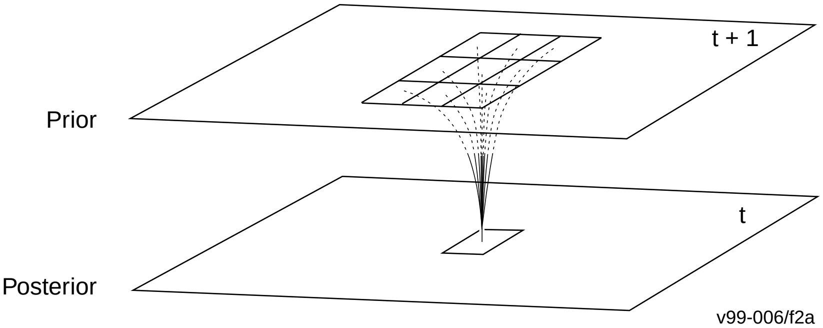

Figure 2. Markov motion spreading.

0.4 probability going elsewhere, then spreading by addition would lead to a “probability” of 1.2 in cell *x*. A way to remedy this situation is to interpret the target-presence probability at cell *x* as the probability that *at least* one target is present. Assuming independence in target motion, the probability that *no* target is found in cell *x* would be 0.16 (= 0.4 × 0.4), therefore making the target presence probability 0.84, a well-defined probability.

3.0 ALTERNATIVE ASPECTS OF PMAP

Viewing the PMAP from different perspectives allows the consideration of various aspects of the algorithm, and leads to greater understanding. First, note that the PMAP can be viewed as a form of *parallel and distributed MHT*. An array of trackers are deployed in the search space and each measurement is associated with a cell—there is no DA problem. Bayesian update reinforces or weakens the hypothesis that a target is in the cell. Markov motion spread poses multiple hypotheses in each cell’s neighborhood to be validated when new observations become available. This is all accomplished without building hypothesis trees and the associated data structure, and is inherently suitable for parallel distributed implementation. Next, it is noted that the PMAP could be viewed as a form of *extended MTI* (moving target indicator). It can separate targets according to their speeds (with 0 speed as a special case). It can also effectively screen out false alarms, even in rather noisy situations. Third, PMAP performs a kind of *dynamic histogramming*. This is actually one of the original motivations of using PMAP for MTT: targets tend to repeat themselves in space over time, whereas (random) false alarms do not. Since the targets are moving, the histogram “bins” must move with the targets to capture them in the next frame. Finally, the PMAP bears some resemblance to *biological systems*. Perhaps the most fundamental similarity is the use of *position encoding*. Instead of fixing trackers on the moving targets, distributed trackers (or cells) are fixed in space. Reported in a recent study,[11] experiments using monkeys confirm the conjecture that the visual coordinate of reference is fixed on the observer, not on the observed external objects. This “self-centeredness” certainly helps avoid the DA problem. Other analogies, such as cell vs. neuron, target-presence probability vs. neuron activation level, and Markov motion spread vs. axon, are straightforward. A less obvious fact is that the Bayes’ rule can be written as a logistic function,[12, 13] which is often used to model the nonlinear sigmoidal activation function in a biological neuron. Collectively, the PMAP algorithm can be viewed as an attempt to capture the self-organizing aspect of parallel-distributed biological visual systems.

4.0 SOME ANALYSES OF THE PMAP ALGORITHM

Some analyses of the PMAP results are in order. First, viewing the PMAP as a nonlinear feedback process, a steady-state analysis is performed. Next, the target acquisition and retention capabilities are analyzed. Problem-specific parameters such as P_d, P_{fa}, target speed range, frame rate, and so on, will play important roles in these analyses, which are very useful in understanding the performance limitations and system design issues.

4.1 THE STEADY-STATE ANALYSIS

Contrary to what one might think, the target-presence (pseudo-) probability, in general, will not approach one even with repeated target presence in the raw data. As a benchmark, the single-target case is the focus of the analysis. It is shown that the steady-state probability will depend on problem-specific parameters such as P_d, P_{fa}, the target speed range, and so on. With some parameter combinations, one might be surprised to see that the (pseudo-) PMAP algorithm may not even be able to hold a nonzero steady-state probability. Therefore, the steady-state analysis is very important in establishing the performance limit and the sensor system requirements. It also helps to provide a sense of scale for the calculated probabilities, and a reference for meaningful threshold setting.

First, notice that Bayes’ rule (with detection) can be viewed as an algebraic feedback loop, as depicted in Figure 3. According to the diagram and the feedback system theory, the higher the P_d and the lower the P_{fa}, the higher will be the posterior probability. This relation between the prior and the posterior probabilities, as dictated by Bayes’ rule, is highly nonlinear, and yet is only a segment of the PMAP processing chain. The projected posterior becomes the prior for the next frame, thereby closing the loop. This latter transformation is accomplished by the Markov motion spread function. Consider the case of a single traveling target, with few false alarms in the neighborhood. The posterior probability is diluted by the Markov motion model among the neighboring cells. If uniform spreading is assumed, then the dilution factor is $1/(n^2)$, where $n \times n$ is the area size of the Markov spread window. Since a single traveling target is assumed and, for simplicity, few neighboring false alarms are considered, the Markov motion contributions from the neighboring cells are negligible. Therefore, the posterior-to-prior transformation can be modeled as a process gain K which is inversely proportional to the area of the Markov motion window—that is, $K \sim 1/(n^2)$. At the output of the gain, K, is the prior probability (of the next scan), which, in turn, feeds the Bayes’ prior-posterior nonlinear transformation. Two curves are of interest here: one is the prior-posterior Bayes’ nonlinear mapping, and the other one is a one-scan posterior-posterior nonlinear mapping; that is, it loops through the posterior-prior-posterior process in one scan cycle. Generally, considerable decrease in probability is associated with the second curve due to motion-related dilution.

[11] Pauls, J., E. Bricolo, and N. Logothetis, “View invariant representations in monkey temporal cortex: Position, scale and rotational invariance,” in *Early Visual Learning,* S.K. Nayar and T. Poggio (ed.), pp. 9 – 41, Oxford University Press, New York, 1996.

[12] Jordan, M.I. *Why the Logistic Function? – A Tutorial Discussion on Probabilities and Neural Networks*, Computational Cognitive Science Technical Report 9503, MIT, Cambridge, MA, 1995.

[13] Ballard, D.H., *An Introduction to Natural Computation*, MIT Press, Cambridge, MA, 1997.

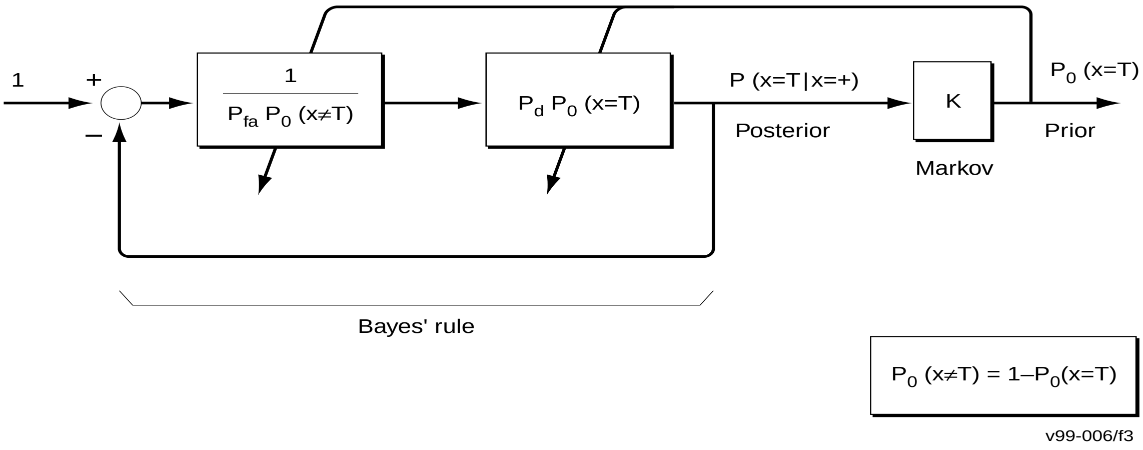

Figure 3. PMAP (with detection) as feedback loop.

To figure out the steady-state probability, one can use a graphical method to work with the posterior-posterior probability curve in an iterative manner. Another solution procedure is to replace the prior probability as *K* times the posterior probability and solve the algebraic equation for steady-state probability *P*:

$$(P_d KP + P_{fa}[1 - KP]) \quad P = P_d KP \quad .$$

The analytic steady-state solutions are 0 and

$$\frac{P_d - P_{fa} / K}{P_d - P_{fa}} \quad .$$

For P_d = 0.7 and P_{fa} = 0.001 and 21 × 21 search windows, the steady-state value is 0.37. (For simplicity, the effect of miss detection is not explicitly accounted for in the analysis, thus making it only applicable to relatively high P_d cases.) Noting that this (nonzero) steady-state probability is constrained between 0 and 1 and that *K* is less than 1, it is concluded that such a nonzero solution does not even exist when $P_{fa} > P_d$. For the more interesting case where $P_d > P_{fa}$, in order to have a valid nonzero solution, the posterior-prior process gain *K* must be greater than (P_{fa} / P_d)—that is,

$$P_d \; > \; P_{fa} \quad \text{and} \quad K \; > \; P_{fa} / P_d \quad .$$

This actually puts a *constraint* on the single target's maximum travel distance between frames, or equivalently, a lower bound on the frame rate. A key factor in this analysis is the posterior-prior gain *K*. If *K* can be made larger, then the steady-state probability would have a better chance to have a higher value. This is certainly the case when velocity estimation is a part of the PMAP algorithm. In such case, one may multiply the (uniform) *K* value by a factor of, say, 2, to reflect the fact that the target is moving in a general direction so that the prior probabilities are concentrated in that general direction. (It can be shown, however, that if *K* were made greater than 1, the process would become "unstable.") With the same set of parameters, P_d = 0.7 and P_{fa} = 0.001 and 21 × 21 search windows, if velocity estimation is assumed and *K* is set to $2/(n^2)$, then the steady-state probability becomes 0.69, considerably higher than 0.37.

Later discussion focuses on a multiple-scale PMAP, which uses a set of donut-shaped motion windows to preserve the probability concentrations for the cases where the targets have very wide ranges of speeds. Another case where the effective *K* is larger is when targets travel together as convoys. Probabilities from neighboring targets contribute toward a higher effective *K* factor, thus leading to higher steady-state probabilities.

4.2 TARGET ACQUISITION AND RETENTION

Target acquisition involves the following question: From the first appearance of a target in the raw data, how many repeated target appearances are required for PMAP to produce a reasonably high probability of target presence? Assuming a prior probability, and then iterating with the prior-posterior and posterior-posterior nonlinear curves (for positive detection), one can obtain a sequence of posterior target-presence probabilities during the target acquisition phase and estimate the number of time steps required to achieve a desired probability level. For example, 63% of the single-target steady-state probability could be used for estimating the acquisition "time constant." (Instead of working with nonlinear curves, looping through algebraic equations will do the same.) Of course, this acquisition time depends on the prior probability assumed, because the PMAP is a *nonlinear* process. A meaningful prior probability is the one-miss target retention probability—that is, the decayed probability after the target has missed one frame.

Consider again the case where $P_d = 0.7$, $P_{fa} = 0.001$, and the search window size is 21×21, and assume that velocity estimation is made. The single-target steady-state probability for this case is 0.69, as before. Using the one-miss target retention probability as the prior probability, the target acquisition analysis reveals that under this specific circumstance, it would take seven repeated target presence for PMAP to "(re)acquire" that single target. However, various circumstances and algorithm refinements can shorten this acquisition time. For instance, a multiresolution version of the PMAP shortens the acquisition time from seven steps to five. Certainly, lower P_{fa}, higher P_d, or smaller search areas would help. Other ways to improve the target acquisition speed include the various vigilance maintenance methods[14] and the multiscale PMAP algorithm; the latter would use smaller search areas and maintain better probability concentration. Use of terrain information in the search may also help keep probability concentration and shorten the acquisition time. Finally, a spatially normalized version of PMAP may be considered for certain applications.

Target retention analysis concerns the question, after reaching the steady state, if the target disappears, what is the decay profile of the PMAP probability? To perform this analysis, a different set of nonlinear curves is required; they are the prior-posterior and posterior-posterior nonlinear curves associated with non-detection. Viewing this process in a way similar to that of the detection case, a similar (but different) nonlinear feedback process diagram can be drawn. It is somewhat more involved in this nondetection case to obtain the posterior-posterior curve, as contributions from neighboring cells need to be accounted for. Space does not allow further discussion of details, except that multiresolution PMAP processing has an edge in target retention.[14]

4.3 PERFORMANCE LIMITATIONS

The above analyses have revealed that the basic PMAP algorithm has its performance limitations. Good results are not always guaranteed; they depend on appropriate combinations of P_d, P_{fa}, target speeds, and frame rate. A common source of problems is that the posterior probability with detection may become too low to be sustained effectively by the subsequent detections. This can happen when $\mathrm{P_d}$ is low, $\mathrm{P_{fa}}$ is high, or target between-frame travel distance may be so long that the search space becomes too large, or two or all three conditions exist. One such limitation discovered under the steady-state analysis is that, for (pseudo-) PMAP to acquire and retain single targets at all, P_d must be greater than P_{fa} and the posterior-prior process gain K must be greater than (P_{fa} / P_d). This certainly places some constraints on the targets' maximum travel distance between frames. Some remedies are discussed in the following sections.

5.0 SOME PMAP ALGORITHM REFINEMENTS

5.1 MULTIRESOLUTION AND FAST PMAP

Multiresolution is an idea that has been used widely in image processing and computer vision.[15] It also seems to have its roots in biological vision systems. The image pyramid is an often used computation structure for

[14] Tao, K.M. et al., *Multiple-Target Tracking in Dense, Noisy Environments*, Technical Report, Applied Electromagnetics Lab., SRI International, Menlo Park, CA, 1999.

[15] Rosenfeld, A., "Some useful properties of pyramids," in *Multiresolution Image Processing and Analysis*, A. Rosenfeld (ed.), pp. 2 – 5, Springer-Verlag, Berlin, 1984.

multiresolution. The image on the bottom of the pyramid has the highest resolution, and the resolution reduces for images residing on the higher levels. The original motivation for a pyramidal PMAP was "action at a distance."[16] That is, when processing at a lowered resolution (LR), targets would travel a fewer number of pixels between frames and therefore could be captured by a smaller Markov search window, thus minimizing probability dilution. It turns out that the gain in probability concentration may be to some degree offset by the increase in the *apparent* LR P_{fa}, *if* the resolution reduction is accomplished by "blurring" an original full-resolution (FR) image, not by a separate set of lower-resolution sensors. However, the computational savings of using a multiresolution PMAP have proven to be of great importance in making the PMAP algorithm practical on a sequential machine. (Another benefit of using multiresolution processing is that it can improve target retention.[17])

Consider the computational savings first. The scheme is to first reduce the FR image frames to LR frames. Certainly, some (but not all) of the FR details are lost at this point. The PMAP algorithm is then applied to these LR images, producing probability maps at the reduced resolution. To recover some of the FR details, each LR probability map is expanded to the FR size. This expansion operation is like a stretching action. Then, by multiplying this expanded LR probability map with the original FR binary raw data, a "recovered" FR probability map is obtained. It should be pointed out that some of the FR details are not recovered. For instance, false alarms that happen to be within the resolution-reduction window around a true target are given the same posterior probabilities as the true target. Experience with this pyramidal PMAP approach has been very positive. The savings in computation time could be orders of magnitude. Although the PMAP algorithm without velocity estimate is reasonably fast on a sequential computer, with velocity estimate, it can be very slow, especially with large search windows. Two fast versions of PMAP have been developed and they perform well. The first one, referred to as the *fast PMAP*, is based on the multiresolution PMAP strategy described above. The savings in computing time could be orders of magnitude.

The other fast algorithm is referred to as the *superfast PMAP*, which is another order of magnitude (or more) faster, depending on the density. Building on the (multiresolution) fast PMAP, the superfast PMAP performs velocity updates only on cells that have a current detection. Since, in general, the majority of the cells do not have a detection in a given frame, the savings are significant. Based on the simulation experience, there is only a very slight degradation in the quality of the final result, definitely worthwhile, at least for sequential implementations. In one example, with 50 frames of 300×300 pixel images and 21×21 search-window size, the fast PMAP based on a 3×3 resolution reduction and expansion cycle reduced the computation time approximately by a factor of 60. The superfast PMAP algorithm reduced processing by an additional factor of 13.

There are other benefits of using multiresolution PMAP. In the above example, better target indication and false alarm rejection have resulted. This is partly because, as mentioned before, an appropriate level of LR processing has an edge in target retention,[17] which, in turn, may benefit target reacquisition. Certainly, for precision tracking one cannot reduce the resolution indefinitely, as severe merging of target tracks would lead to loss of target motion information or loss of filtering capability. (If the whole frame is reduced to one single cell, then after resolution expansion the original noisy raw data will be completely "recovered.") A discussion of optimal level of reduction from a probabilistic viewpoint is found in reference note 14 above. Other beneficial uses of multiresolution PMAP include "coarse-to-fine" tracking where coarse PMAP is used for wide-area surveillance, and fine PMAP for zooming in. The multiresolution pyramid has also been used to demonstrate a form of "clustering" for tracking convoys of targets.

5.2 MULTISCALE PMAP

Finally, to search for *potentially* very fast targets or targets with a wide range of speeds, one may have to consider a set of donut-shaped search windows, one window within another. This is referred to as *multiscale PMAP* in this paper. The analogy is like using a set of magnifiers of different powers to look for objects of different sizes. The PMAP algorithm is run simultaneously with the whole set of windows, an operation similar to that of the filter banks in signal processing theory. Each search window covers a relatively small search area, corresponding to a unique range of travel, while minimizing the dilution of probability. Within each processing step, the complete set of probabilities is combined using a MAX operation, and then processed separately with the different windows.

[16] See note 15 above.

[17] See note 14 above.

Certainly, there is potentially a performance loss in that to search for targets with a wide range of speeds, the number of false alarms retained by these windows collectively would be higher. However, this is simply the result of the problems with wide ranges of speeds. This architecture also provides a means to separate targets by their speeds.

5.3 PMAP FOR SPECIFIC TARGET FOLLOWING

It is not surprising to find that "tracking" may mean different things to different people. Sometimes, it simply means keep track of what is going on in the current situation and perhaps project what is likely to happen in the near future. Therefore, being able to clean up the noise and clearly indicating where the targets are likely found would certainly satisfy that definition. Used as a pre-filter, the basic PMAP algorithm does that very well, and it also provides target projection capabilities for planning purposes. Another widely accepted definition of "tracking" includes the formation of target "tracks." That is, it is also of interest to know where the current targets might have been in the past frames—knowing the past might help predict the future and maintain a historical perspective. If the velocity estimation option is chosen, the basic PMAP algorithm also computes the likelihood of the various neighborhood cells being the previous sites of the given target. This information can be used as an inter-frame target connection indicator. Various algorithms can be readily applied or devised for that purpose.

Taking a slightly different view, the task of pursuing a few specific targets in an end-game scenario is considered here. A variant of the basic PMAP is briefly described as a means to accomplish that task. Consider the case where two specific targets are to be followed. Instead of keeping a generic target-presence probability in each cell, specific target probability states are introduced. Suppose that, in a given frame, two cells with detections are selected as target 1 and target 2 locations. Then, the basic PMAP procedure can be applied to those two cell locations except that it should proceed as in the single-target case. That is, global measurements are used in Bayesian updates, and spatial normalization (to probability of 1) is carried out for each of these two targets. (The probability of false alarms, P_{fa}, should account for the potential presence of other targets in the neighborhood.) The computed probability maps are individually stored and stacked; that is, there will be a target 1 state and a target 2 state in each cell. Note that since each target moves about in a neighborhood between frames, only a relatively local map area needs to be updated for each target at each scan. Consider the classical "track crossing" situation for illustration. After the tracks cross (at least for a while), the location of each target hit will likely have comparable probabilities for being target 1 and target 2, say 0.6 and 0.4. This is a reasonable result. As more data become available, these probabilities may gravitate toward the correct target associations with the help of the (estimated) velocity information, if the data support a clear identification.

6.0 TRACKING RESULTS

Both simulated and real data have been used to test and demonstrate the PMAP algorithm performance. Tracking the simulated data is described first. Fifty frames of (simulated) ground truth are generated. The truth contains two convoys, each with 10 targets, traveling together, and a single target, traveling by itself. These are 300 × 300 pixel binary images; in each frame, locations of nonzero cells indicate target locations. The targets can travel several (up to about 10) pixels between frames. The single target is perhaps the fastest traveler, whereas the convoy at the bottom of the frame picture is the slowest. The probability of detection (P_d) and the probability of false alarm (P_{fa}) were introduced to simulate the effect of the detection process. The P_d is 0.7. False alarms are added using the uniformly distributed Poisson process model,[18] with $P_{fa} = 0.001$—that is, approximately 90 false alarms and 15 targets are present in each 300 × 300 pixel image frame. These 50 noisy post-detection frames are referred to as raw data in this discussion. These noisy raw data are first preprocessed with the PMAP algorithm. The probabilistic PMAP results are thresholded to yield binary results, indicating likely target locations. (Based on the simulated data, a threshold setting at around 0.3 to 0.4 often gives very good results. The steady-state analysis helps threshold selection.) At this point, the majority of the false alarms have been eliminated, while most of the targets are retained. These binary images are then used for display and for a downstream tracker, MHT in this case, to form tracks.

The best PMAP results are obtained with velocity estimation using the (multiresolution) fast or superfast PMAP algorithm and the terrain suitability map (whose values indicate the terrain fitness for a target to dwell or pass through). The resolution reduction factor is 3 in both the *x* and *y* dimensions; that is, the raw data are reduced to 100 × 100 pixel images to be processed by the multiresolution PMAP algorithm. The low-resolution Markov search

[18] See note 7 above.

window size is 7 × 7. Full-resolution details are largely recovered later using the procedure outlined in the discussion of mutiresolution PMAP. (In these simulations, the PMAP algorithm is initialized at a level of 0.2 prior probabilities and then weighted by the terrain suitability map. More sophisticated use of terrain information may include the influence of terrain on target's speed of travel, hence the search window size.) The performance statistics indicate that nearly 98.5% of the raw target hits are captured by the PMAP, and only about 4.2% of the false alarm hits remain after PMAP preprocessing. Note that the target hits in the raw data are *detected* targets; the actual number of targets in the truth is higher with the sensor's probability of detection P_d being 0.7. This means that after PMAP preprocessing, the effective P_d is about 0.69, still very close to that of the sensor's P_d, but the probability of false alarm P_{fa} is substantially reduced from 0.001 to about 0.00004. A closer examination further reveals that most of the false alarms, after the PMAP screening, occurred in the first five frames or so, when the PMAP was still warming up and learning the statistics from data. In the rest of the frame sequence, the few false alarms remaining are the ones that happen to be in the neighborhood of detected targets. These may be termed target-related false alarms, which are sometimes considered target location errors, not as severe as real false alarms. Excluding these near-target false alarms, the false-alarm rate becomes virtually nil. In this case, the performance of the MHT working with PMAP pre-filtered data is considerably better than working with noisy raw data directly. Still, there are some mixed target identities that could be minimized by better setting the MHT parameters, and, also, largely eliminated by tracking clusters instead of individual targets. The PMAP tracking results are displayed in Figure 4 with the terrain suitability map in the background. A cumulative format is used for visual continuity. The white dots are detections or targets or target indicators. The first 10 frames (of the dotted panels)[19] are the PMAP warm-up period and are skipped for cleaner display. This figure clearly illustrates how many false alarms have been rejected and how closely the processed results match the (otherwise unknown) clean data.

The PMAP algorithm has also been tested with real data—in this case, GPS ground vehicle positions collected during exercises at Ft. Stewart, Georgia. In one data set, there are about 50 ground vehicles in various modes of travel, battle formation, and battle engagement conditions. Move-stop-move, on-road/off-road and enter-and-leave-the-scene phenomena are commonplace in this data set. As a result of data dropouts, the effective P_d (in the ground truth) is approximately 0.7 to 0.8. Uniformly distributed random false alarms (with P_{fa} = 0.0008) were added to simulate noisy reports. Subsampled at a 10 second rate, there are 271 (150 × 300) pixel frames. Figure 5 displays, with the area map in the background, a sample frame of the (multiresolution) fast PMAP results. The resolution reduction factor is 2, and the reduced search window size is 7 × 7. Notice that the lower right panel in the figure is a display of PMAP filtered binary output with PMAP prediction superimposed. The shaded region around each binary dot indicates the projected target positions in the next frame. In this case, the velocity estimation option is not used and the projections are uniform. (This is partly because some of these targets are highly maneuverable in their behaviors.) In addition to planning, these projections are also very useful when there is sensor blackout. In the absence of sensor data, the PMAP projections will continue to coast and provide indications where the targets are likely to be found. As evidenced in this sample frame and in the overall statistics, with the exception of a few target-related false alarms, virtually all false alarms have been removed with minimum (less than 3%) loss of targets; that is, if the sensor P_d is 0.8, the filtered results would have an effective P_d of 0.78, but with virtually all false alarms removed.

[19] The MHT panel of display utilizes hindsight and certain plotting criteria to screen out spurious tracks, and has fewer than 10 initial frames skipped.

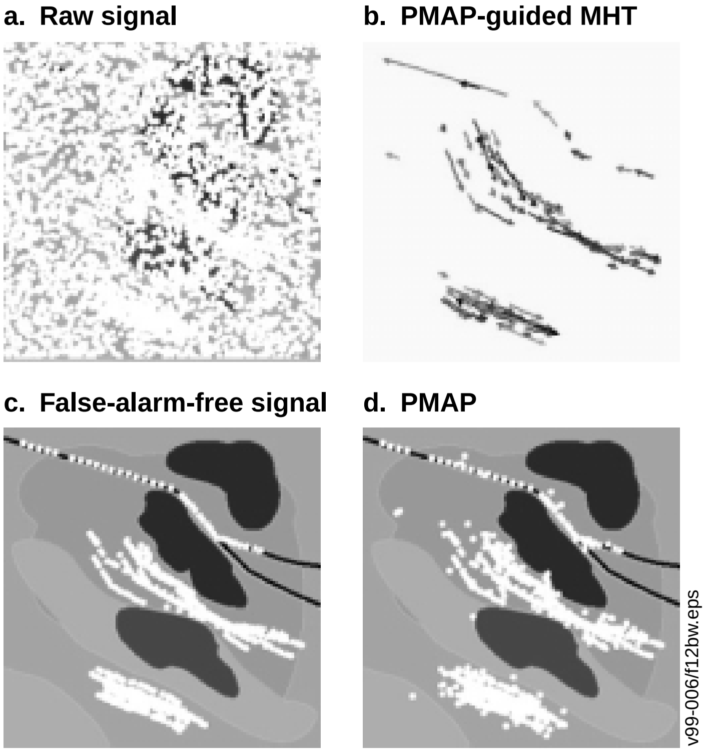

Figure 4. Simulated MTT with PMAP—cumulative display (clockwise from top left): noisy input to PMAP, PMAP-guided MHT result, PMAP result and (clean) "truth."

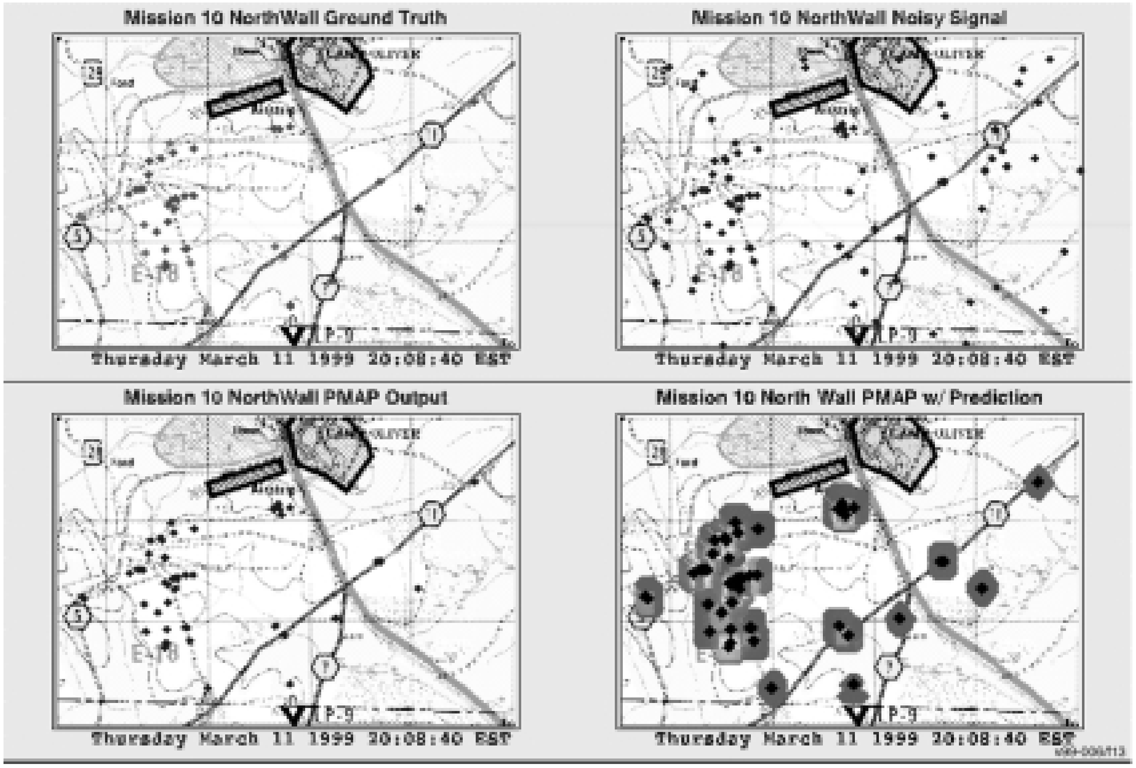
Mission 10 NorthWall Ground Truth
Mission 10 NorthWall Noisy Signal
Mission 10 NorthWall PMAP Output
Mission 10 North Wall PMAP w/ Prediction
LAKE OLIVER
E-10
LP-9
Thursday March 11 1999 20:08:40 EST
Thursday March 11 1999 20:08:40 EST
Thursday March 11 1999 20:08:40 EST
Thursday March 11 1999 20:08:40 EST
v99-006/f13

Figure 5. PMAP result on a sample frame from Ft. Stewart data (clockwise from top left): "truth," noisy input to PMAP, PMAP result (current + prediction), and PMAP result (current).

Finally, Figure 6 displays an example of using PMAP for following specific targets. In this example, the multi-target surveillance version of the PMAP is used to filter out the noisy false alarms first. It is assumed that at time 0 specific targets were picked ("fingerprinted") for close following. Each pinpointed target location, at time 0, was given to a specific-target-following PMAP, which then computes the probability map for that particular target in each of the subsequent frames. Two specific targets are selected: target A and target B. The upper left panel of Figure 6 is such a subsequent sample frame for target A (along with multiple target detections), which serves as input to the target-A-following PMAP algorithm. (Target A is shown with a circle.) The time is 4.5 minutes (or 27 frames) after the initial fingerprinting. The upper right panel shows that out of all the detections present, including one very close by, the target-A-following PMAP algorithm successfully identifies target A. Predicted target-presence probabilities (for the next frame) are super-imposed on the current position of target A (the binary dot). Note that, in this target-flowing mode, *global* Bayesian feedback is used to update the specific-target presence probability for each cell.

Similarly, the results for following target B are shown in the lower left and lower right panels. The time elapse since initial fingerprinting is even longer in this case: 8 minutes and 20 seconds. Also, the environment for target B is more cluttered with many nearby targets, and still the PMAP algorithm, in this case, correctly identifies target B's location. (One may notice that for target B the predicted probabilities occupy a smaller area. This is because target B is off-road; a smaller search window is used – reflecting lower travel speed. Smaller search windows help reduce the chances of getting confused with other nearby targets.)

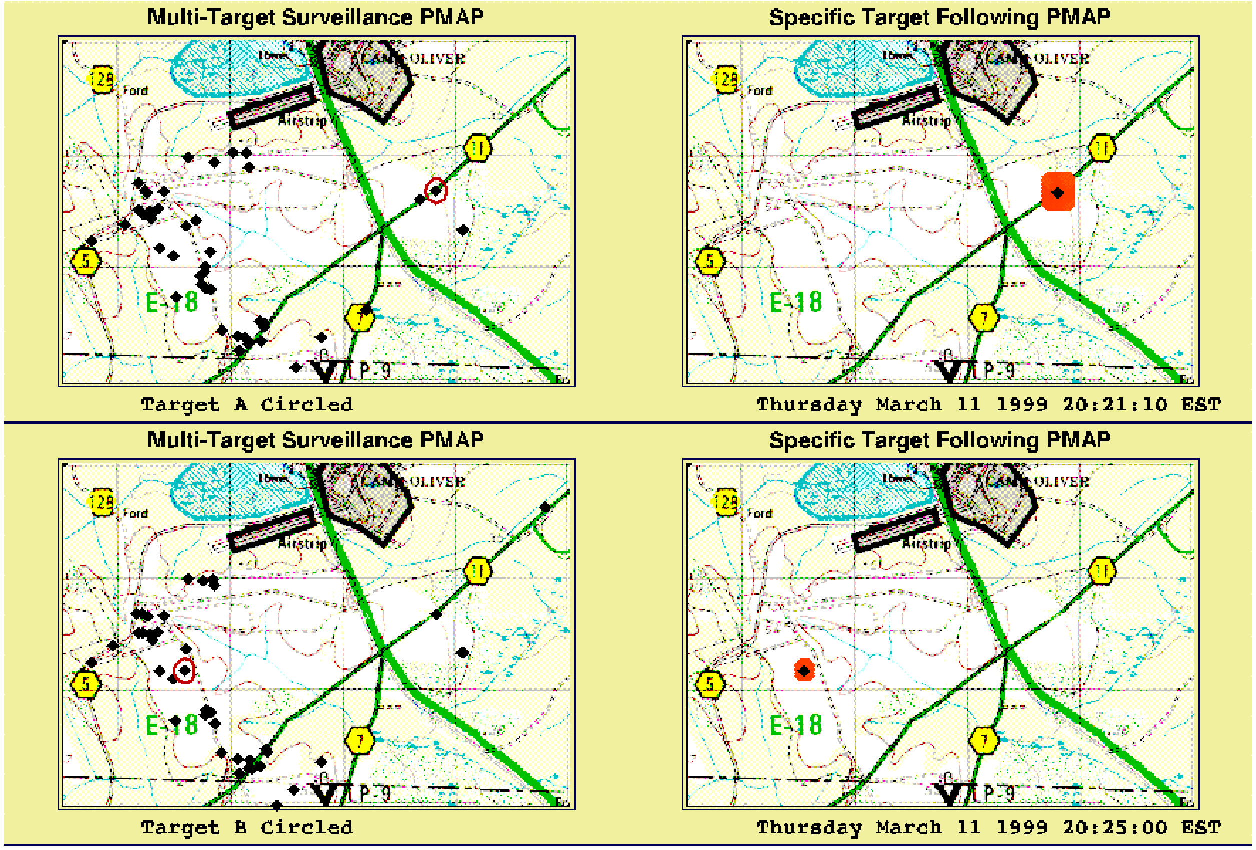
Multi-Target Surveillance PMAP
Specific Target Following PMAP
126
Ford
CAMP OLIVER
Airstrip
11
5
E-18
7
LP-9
Target A Circled
Thursday March 11 1999 20:21:10 EST
Multi-Target Surveillance PMAP
Specific Target Following PMAP
Target B Circled
Thursday March 11 1999 20:25:00 EST

Figure 6. PMAP for specific target following: (Top row - left to right) the multi-target PMAP surveillance result (with the target A circled), and the result of applying the specific target following PMAP algorithm, 4.5 minutes after initial target pinpointing. (Bottom row – left to right) the multi-target PMAP surveillance result (with the target B circled), and the result of applying the specific target following PMAP algorithm, 8 minutes and 20 seconds after initial target pinpointing.

7.0 CONCLUSIONS

A new approach has been taken to address the multi-sensor, multi-target tracking problem (in dense and noisy environments). Using real and simulated data, it is shown that the new algorithm PMAP is able to clean up the false alarms with little target loss. This makes the task of target tracking and linking a much easier one for the conventional tracking algorithms such as the MHT and the PDA. This is accomplished without having to deal with the difficult data association problem, as conventional trackers do. The feasibility of using PMAP to follow specific targets in an end-game scenario is also demonstrated using real data.

Analytical tools are provided that are useful in determining proper threshold setting, the expected performance limitations, and proper combinations of system design parameters. The PMAP computation is parallel and distributed in nature and is a function of the size of the search area and target mobility between frames, and, unlike the conventional trackers, it is independent of the density of the target environment. (Connections with biological visual systems are discussed.) Fast versions of PMAP, using multiresolution principles, are described for sequential computer implementation. Multiresolution PMAP also allows efficient and effective coarse-to-fine surveillance and tracking. An interesting observation is the following. Conventionally, in order to keep the P_{fa} low, it often is necessary to sacrifice P_d by setting the detection threshold level high. With the PMAP preprocessing, however, one may go for a high P_d by setting the detection threshold low and allowing the PMAP to clean up the false alarms effectively. Although only noiselike false alarms have been discussed in this paper, a preliminary simulation study has demonstrated that PMAP can clean up correlated (moving) false alarms, if there is adequate speed separation from the targets. Contextual information such as weather may also be useful in dealing with correlated clutter in certain real-world applications. It should be noted that a similar but independent effort was discovered [20] after the completion of the work described herein. The general ideas are similar but the details are different.

Finally, it is pointed out that the general PMAP approach provides a flexible spatial-temporal fusion environment suitable for various applications involving fusing data reports from different information sources over time. In addition to multi-target tracking, search and monitoring and gray-scale video sequence stacking are two of the applications to which PMAP-like approaches have been successfully applied.

ACKNOWLEDGMENT

The authors would like to thank D.L. Holeman, M.R. Faust, and J.Y. Gilkey for providing the Ft. Stewart data, and D.L. McPherrin (all of SRI International) for computation and display related support.

[20] Stone, L.D., C.A. Barlow and T.L. Corwin, *Bayesian Multiple Target Tracking*, Artech House, Boston, 1999.