Tasking sensors adaptively using online learning of their achieved performance

By RI Young[+], S Maskell* & S Parsons[++]

[+]QinetiQ
*University of Liverpool
[++]King's College London

Abstract

This paper describes an approach that enables an automated multi-sensor fusion system to coordinate the tasking of sensors while considering the extent to which these sensors do (and do not) behave in the way that their specifications state that they will. The potential for a sensor's performance to depart from its specification currently forces the designer of a fusion system to either use the sensors to less than their full potential or be prone to difficulties when sensors fail to perform as they should. In this paper, a Sequential Monte Carlo (SMC) sampler is used as a state-of-the-art solution to the problem of performing the online learning of how a sensor's diagnostic signals (e.g. a video camera's low contrast flag) can be used to predict when sensor performance is degraded (e.g. due to fog). Simulated scenarios are used to validate the operation of the SMC sampler. It is demonstrated that the proposed algorithm can learn automatically, for example, that sensors only detect targets in their field of view. Discussion highlights that the key benefits would be that the system would only task sensors to do tasks that they can perform and that the system would learn when to trust in the information provided by each of the sensors.

1. Introduction

Sensor networks are now fundamental to gathering data that can be used for the protection of critical national infrastructure or to ensure city centres are safe for the population living there (Rouse *et al.* 2010). Recent development of cheap sensors and networking enabling technology has led to the current position that the protection of land based, high-value assets such as critical national infrastructure is conducted by the deployment of a multitude of sensors around the asset's perimeter. These sensors feed data back to a central hub for inspection by an operator. The sheer volume of data collected can cause the operator to suffer data deluge. Towards preventing this, multisensory data fusion systems have been developed to coordinate sensor management, tasking and automated analysis of the data to provide alerts to feed to the operator (Lane and Copsey 2012).

In hostile environments and over long deployments, factors such as poor weather can cause the sensors to malfunction or even fail. This leads to erroneous or even no data being sent to the data fusion system for analysis, with the resulting consequence of the creation of incorrect situational awareness pictures. Methodologies have been developed to detect and ameliorate the impact of such faults within sensor networks (Zahedi *et al.* 2008) but these require prior knowledge of the type of failings that can occur in order to be useful. The present work is motivated by the need to develop mitigation techniques for use against unforeseen circumstances, resulting in the sensor behaving in a manner other than what its specifications said it would.

This work assumes that the data fusion technology necessary to provide the majority of the functionality of a multi sensor fusion system already exists. Specifically, it is assumed that technology exists to associate and fuse streams of data coming from multiple sensors (of disparate types) and that technology exists to automate the tasking of assets to maximise the chance that an information goal can be met. These technologies have been demonstrated in previous UK MoD funded research, for example, in the Persistent Wide Area Surveillance (PWAS) project (Rouse *et al.* 2010) and the Sensor Information Processing And Management (SIPAM) projects respectively (Strens *et al.* 2007).

The aim here is to augment the state-of-the-art with algorithms that address a problem that plagues fusion systems built on standard technologies that have been demonstrated in projects such as PWAS and SIPAM. The specific problem is that sensors don't always behave in the way that they say they will. This forces the designer of a fusion system to either use the sensors to less than their full potential or be prone to difficulties when sensors fail to perform as they should. Technology developed to address this issue would enable the system to learn how fluctuations in the environment affect sensors' behaviour. Key benefits would then be that the system would only task sensors to do tasks they can perform and that the system would learn when to trust in the information provided by them.

The challenge is to develop technology that exploits an ever-increasing record of historic data to improve an understanding of the current and future performance of the sensors providing data to a fusion system. Specifically, the aim is to understand when, as a function of its environment, a sensor performs according to its published specification, which we anticipate to be optimistic in some environments, and when it is not, meaning we should make use of a pessimistic specification of sensor performance.

Understanding future sensor performance enables the fusion system to task sensors with an improved understanding of their likely performance. Put simply, there is no point in tasking a sensor to detect a target if the current environment is such that the sensor will never be able to detect the target. If the system tasking sensors understands their performance, then it can make best use of this sensor resource and avoid being surprised by sensors' future performance.

Understanding current sensor performance enables the situation awareness picture to be compiled in a way that avoids using an optimistic model when it is inappropriate to do so. Such inappropriate use of an optimistic model can result in pathological behaviour. For example, if the fusion system makes optimistic assumptions about detection performance, track failure can result since the fusion system assumes that a lack of detections actually caused by some environmental factor must be caused by the object not being present. While one can always use a pessimistic model of detection performance to address this issue, in environmental conditions where the optimistic model is applicable the accuracy of a system is reduced.

The relationship between the diagnostic signals provided by a sensor and whether that sensor performs in accordance with its specification could be described with a truth table which considers all combinations of all diagnostic signals. However, the number of entries in that truth table would grow rapidly with the number of diagnostic signals. To avoid the associated computational expense, we propose to consider smaller truth tables that only consider a subset of the available diagnostic signals. We therefore need to search the space of such smaller truth tables as the data arrives and so adaptively learn both the content of the truth table but also the diagnostic signals that are relevant to sensor performance.

The problem of estimating the (smaller) truth table as data arrives is an example of online learning. Online learning has historically proven to be a difficult problem to solve in a way that combines computational efficiency with asymptotic accuracy. However, Sequential Monte Carlo (SMC) samplers offer a candidate solution that can combine efficiency with accuracy. We therefore adopt such SMC samplers here. Note that SMC samplers are a special case of SMC methods. Particle filters are another example of SMC methods and are applicable to situations where we are processing a data stream to estimate a time-varying state. Here, in contrast, the truth table is assumed fixed. Particle filters are therefore not appropriate.

The paper is organised as follows: Section 2 describes the SMC sampler in the context of online learning. Section 3 describes the context in which we consider the SMC sampler to operate. Section 4 then describes two scenarios and presents results. These results are discussed in section 5 ahead of conclusions being drawn in section 6.

2. Sequential Monte Carlo samplers

As already explained, the solution described herein is based on Sequential Monte Carlo (SMC) samplers. This section describes this solution and provides pseudo-code to aid future implementation by others.

2.1. Statistical Model Definition

At the t^{th} time step when a target is being tracked, we receive a binary variable, d_t, stating whether or not a detection event has occurred and a list of N_b binary variables indicative of the system state, $b_t^{1:N_b}$. We assume we have prior knowledge detailing possible different performances for that sensor, perhaps supplied from the manufacture's data sheet. For example $P(d_t = 1|M)$ where $M \in \{M_1;M_2\}$, i.e. there are two models with different probabilities of detection, one more pessimistic than the other.

We assume the model is a function of the binary variables such that $M = M(b_t^{1:N_b}|\Theta)$ where Θ parameterises the mapping. While the approach generalises to multiple sensors, this discussion considers a single sensor for ease of exposition; the approach has utility when multiple sensors are present, but the processing is inherently parallel across the sensors.

We wish to exploit a stream of values for d_t and $b_t^{1:N_b}$ to learn $M(b_t^{1:N_b}|\Theta)$, i.e. to perform online learning of Θ from the stream of $d_{1:t}$. If we knew Θ, we could predict the probability of detection from $b_t^{1:N_b}$ and so make best use of sensors in both the compilation of a situational awareness picture and the tasking of sensors.

We assume that Θ is a look-up table into M, indexed by a subset (the content of which will be estimated) of size N_S, of the binary variables such that there are 2^{N_S} different values that the combination of the subset of variables could take. Since there are two models for M, the look up table can be considered to be a binary vector stating whether or not M_1 is active for each combination of input variables. At time, t, we assume that we have Θ_t as a hypothesis for Θ. To apply an SMC sampler (see below), we need to define the distribution we wish to estimate. We choose[1] to define the marginal target distribution at time, t, as that distribution of interest:

$$\pi_t(\theta_t) = p\left(\theta_t, d_{1:t}\middle|b_{1:t}^{1:N_b}\right) = p(\theta_t)p\left(d_{1:t}\middle|b_{1:t}^{1:N_b}, \theta_t\right) \tag{1}$$

where we can define the prior on Θ as:

$$p(\theta) = Po(N_S;\lambda)\frac{1}{2^{2^{N_S}}} \tag{2}$$

where λ is the expected number of binary variables that are relevant and $Po(N;\lambda)$ is a Poisson distribution on N parameterised by a rate of λ. For clarity, a priori, we assume that the size of the subset, N_s, is Poisson distributed (as a convenient alternative to it being binomially distributed) and each of the look up tables of size N_s is equally likely: each truth table of size N_s has 2^{N_S}entries, each of which relates to a binary variable, such that there are $2^{2^{N_S}}$ different truth tables with size N_s. The likelihood is then:

$$p\left(d_{1:t}\middle|b_{1:t}^{1:N_b}, \theta_t\right) = \prod_{t'=1}^{t} p\left(d_{t'}\middle|M\left(b_{t'}^{1:N_b}\middle|\theta_t^i\right)\right) \tag{3}$$

Note that (3) requires a calculation that involves the entire historic data record and the entire history of binary variables.

2.2. SMC Sampler

We provide a brief summary of the operation of a Sequential Monte-Carlo (SMC) sampler being used for online learning. A more detailed description is available in (Maskell. 2012).

In common with a particle filter, the SMC sampler iteratively draws samples from a proposal distribution such that each sample can be considered to be a trajectory over the iterations of the algorithm. We then use importance sampling to enable the current samples to become a weighted approximation to samples from a distribution of interest, which here is $\pi_t(\theta_t)$. To achieve this, we define a target distribution for the trajectory associated with a sample as the iterations proceed, as:

$$\pi(\theta_{1:t}) = \pi_t(\theta_t)\prod_{t'=2}^{t} L\left(\theta_{t'-1}|, \theta_{t'}\right) \tag{4}$$

where here we make the simplistic assumption that $L(\theta_{t-1}|\theta_t) = \pi_{t-1}(\theta_{t-1})$. The definition of the target distribution for the tractor is the key concept in SMC samplers which gives rise to the property that the current samples can be used to approximate $\pi_t(\theta_t)$ We then assume the existence of a proposal distribution $q(\theta_t|\theta_{t-1})$ such that:

$$q(\theta_{1:t}) = q(\theta_1)\prod_{t'=2}^{t} q\left(\theta_{t'}|, \theta_{t'-1}\right) \tag{5}$$

---

[1] Other choices of target distribution, eg the posterior, would be possible. However, using the posterior specifically would demand the calculation of an intractable integral. We therefore target a scaled version of the posterior (as is, for example, standard (albeit often implicit) practice in particle filtering).

The specific form of $q(\theta_t|\theta_{t-1})$ is that it samples whether to keep the set of binary variables considered by the truth table the same, whether to reduce the number by one (unless there is only one binary variable in the truth table) or whether to increase it by one (unless all the binary variables are in the truth table). If the truth table shrinks, the binary variable to remove is chosen at random from those present in the truth table and each element in the new truth table is sampled uniformly from the (two) associated elements in the old truth table. If the truth table grows, the new binary variable is chosen at random from those not present in the truth table and each element is taken to be the same as the associated element in the old truth table. Whatever the size of the new truth table, there is then a small probability that each element in the truth table is toggled: this acts much like a mutation operation in a genetic algorithm.

The definitions above enable the recursive definition of importance sampling weights:

$$\omega_t = \frac{\pi(\theta_{1:t})}{q(\theta_{1:t})} \tag{6}$$

$$= \omega_{t-1} \frac{\pi_t(\theta_t)}{q(\theta_t|\theta_{t-1})} \tag{7}$$

It would be possible to optimise the choice of proposal distribution and L-kernel though this is not considered here.

2.3. SMC algorithm pseudo-code

So, after initialisation (which just consists of initialising[2] the truth tables with some initial estimates and initialising the particle's weights as uniform), the algorithm proceeds as follows at the t^{th} iteration:

- Add the new datum, d_t, to the historic data record to produce $d_{1:t}$;
- Add the new binary variables, $b_t^{1:N_b}$, to the historic record of binary variables to produce $b_{1:t}^{1:N_b}$;
- For all of the N samples, exemplified by the i^{th} sample:
  - Sample $\theta_t^i \sim q\left(\theta_t \middle| \theta_{t-1}^i\right)$;
  - Evaluate $q\left(\theta_t \middle| \theta_{t-1}^i\right)$;
  - Evaluate $\pi_t\left(\theta_t^i\right)$ using (1);
  - Calculate w_t^i using (7);
- Normalise the weights to produce $\left\{\bar{\omega}_t^i\right\}_{i=1}^N$;
- Given the current values for $b_t^{1:N_b}$, output a weighted estimate of the Probability of detection as $\sum_{i=1}^N \bar{\omega}_t^i p\left(d_t \middle| M\left(b_t^{1:N_b} \middle| \theta_t\right)\right)$;
- If the effective sample size drops below a threshold, resample.

3. Algorithmic Context

We now describe how an SMC sampler could provide the envisaged fault detection functionality. Figure 1 shows the architecture of a possible future multi-sensor fusion system for asset protection. Each component's functionality can be described as follows:

- **Sensors**: The sensors are assumed to accept tasking requests and provide target declarations as well as other information (e.g. diagnostic health information). Note that we assume a system with enough spatial redundancy that an object could be tracked by multiple sensors so that it is apparent whether or not an individual sensor is detecting the object. This is the same level of redundancy that a prudent system designer would employ to ensure that there is not complete reliance on any one sensor.

[2] In the results that follow, all SMC samplers were initialised to think that a single signal (alone) dictates the correct model to use. In fact, the specific signal chosen does not have any such correlation with detection performance.

- **Fusion Engine**: This ingests the sensor data, transforms to information (e.g. detections to tracks) and fuses these to produce a situation awareness picture. It also accepts an input from the **SMC Sampler**, which tells the fusion engine whether each sensor appears to currently be behaving according to its published specification, M_1, or M_2 (and thus whether the optimistic or pessimistic view of the sensor should be adopted). As well as the situational awareness picture, diagnostic information is also output (for example conflicts between different sensor declarations – see section 4 for examples of such outputs). It is assumed that any online calibration (e.g. adjustment for where North and Up are for each sensor) is a subsystem employing mature technology within this fusion engine.
- **Condition Monitoring**: The diagnostic output from the fusion engine is processed and converted into a set of binary variables via suitable (potentially adaptive) thresholding. At the time of each set of track-updates, this component provides a set of binary statements relating to the condition of the sensors and whether each sensor detects any of the tracked targets. Examples of diagnostic signals are given in section 4.
- **SMC Sampler**: The SMC sampler processes the binary variables and the detection data to estimate the truth tables.
- **Sensor Scheduler**: This component converts the user-articulated goals into candidate future schedules for tasking the sensors. It optimises the choice of schedule and constituent tasking based on both the current situational awareness picture and the current perception (provided by the SMC sampler) as to which sensors are behaving according to their published specification. Such scheduling is important if the fusion system is to exploit the heterogeneity of sensing modalities in the sensors: the sensors are assumed to be unaware of which other sensors are present and so, without tasking from the scheduler, are unable to autonomously optimise their behaviour to provide declarations that are have most utility when fused with other sensor declarations.

The SMC sampler is the core of the novel additions to the multi sensor fusion system. This component resolves conflict between the summaries with which it has ever been supplied to infer a consistent mapping of binary variables to whether the sensor is behaving according to its published specification.

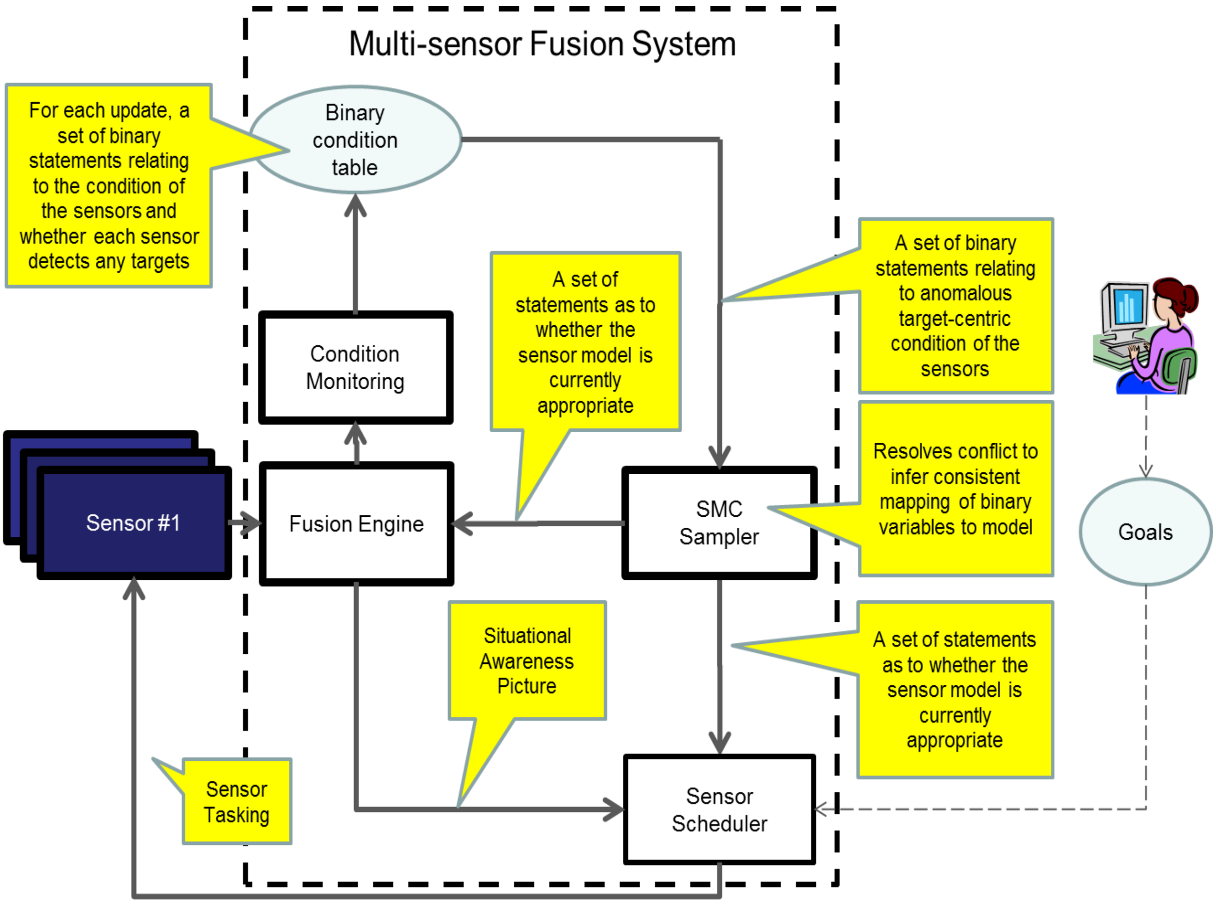

FIGURE 1. Architecture for the multi-sensor fusion system considered in the scenario.

Note that three control loops are present in figure 1, two of which result from the novel inclusion of the SMC Component (with the other being the mature process of optimising sensor tasking in response to the current situational awareness picture[3]):

• As a result of its previous experience, the SMC sampler detects that the environment is currently such that some sensors are not likely to perform according to their published specification. This changes the level of trust that the fusion engine places on data (or lack of it) from the associated sensors. This empowers the system with an ability to learn when to trust in the information provided by sensors.

• As a result of its previous experience gained from Condition Monitoring, the SMC sampler again detects that the environment is currently such that some sensors are not likely to perform according to their published specification. This can also change the confidence that the scheduler has in a sensor's ability to successfully respond to a given tasking. This empowers the wider system with an ability to only task sensors with tasks that they can actually successfully undertake.

4. Simulated System Performance

We consider scenarios based on the geometry shown in figure 2. The sensors have already been tasked and are operating against those tasks. The tasks involve staring at a fixed area of interest and declaring any detections made. Three sensors have been assigned. Sensor_1 and Sensor_3 contain electro-optic (EO) cameras with a field of view as indicated by the yellow areas in figure 2. Sensor_2 contains a radar which is constantly monitoring the region described in blue. Sensor_2 gives wide area coverage with accurate location details. Sensor_1 and Sensor_3 can provide further contextual information if required but are currently tasked to declare all detections. This geometry allows the fusion engine to cross correlate sensor declarations to build a richer understanding of what is happening in the area being monitored. Sensor_1 and Sensor_3 have overlapping fields of view to allow hand off of tracked objects from one field of view to the other if necessary.

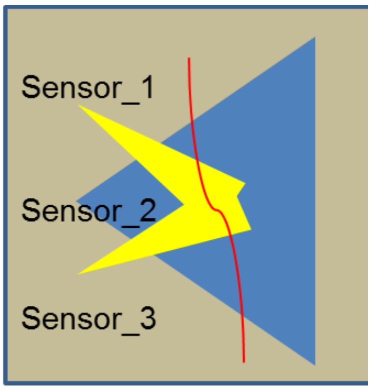

FIGURE 2. Schematic of the deployment of three sensors, showing the overlapping coverage from their respective fields of view and the trajectory of a target moving across the sensor's fields of view.

In each of two scenarios, an object then moves as shown by the red line in figure 2. The diagnostic information that populates the binary variables to be processed is as follows (where we denote data relevant to the radar, camera-1 and camera-2 with suffixes of "r", "c1" and "c2" respectively):

- Detection information (i.e. is an object detected), denoted "detected-r", "detected-c1" and "detected-c2".
- Diagnostic health information provided by sensors, denoted "alive-msgs-c", "alive-msgs-r", "camera-gain" (whether it is within defined limits deemed sensible), "histogram" (whether image saturation is occurring), "velocity-r" (whether the estimated velocity is reasonable for the target type in question) and "error-r (whether an internal error has been detected)

[3] A potentially interesting extension of the mature technology associated with this control loop would be to schedule the use of sensors to detect, track and identify targets (as is already well understood), but also task sensors in a way that enables the system to better understand the performance of its constituent sensors as a function of their diagnostic outputs.

- Diagnostic information derived by the fusion engine:
    - Whether the track is in the sensor's field of view (ie whether the fusion engine perceives that the target is likely to be detected), denoted "fov-r", "fov-c1" and "fov-c2";
    - Whether there is a timing conflict detected (ie whether synchronisation errors have been detected), denoted "t-conflict-r", "t-conflict-c1" and "t-conflict-c2";
    - Whether there is a position conflict detected (ie whether the fusion engine deems a sensor to be providing information about the position of a target that is inconsistent with the position reported by other sensors), denoted "p-conflict".

Timing and Position conflict information reflects whether declarations that should correlate are actually in agreement or conflict with one another. Two sensor models are considered. One in an optimistic model (with an assumed probability of detection of 80%) and the other is a pessimistic model (with an assumed probability of detection of 20%).

Scenario 1
Scenario 1 considers a target moving into the field of view of the radar and then moving through the field of view of the two cameras. For one time-step the target is assumed to move outside the field of view of the radar and is not detected at that time-step. The input data for scenario 1 is shown in Figure 3.

Scenario 2
Scenario 2 is similar to scenario 1, except that the data indicates that the target is never in the field of view for camera 2 (despite it being detected! i.e. it is assumed that either the sensor has given a false detection or the fusion engine has made a mistake) and the radar detects the target at all time steps. This scenario aims to emulate the often encountered situation (eg due to misunderstandings between component suppliers in a complex system or due to faults present in that complex system) where there is a mismatch between what one might expect to observe and the correlations that actually occur. The input data for scenario 2 is shown in Figure 4.

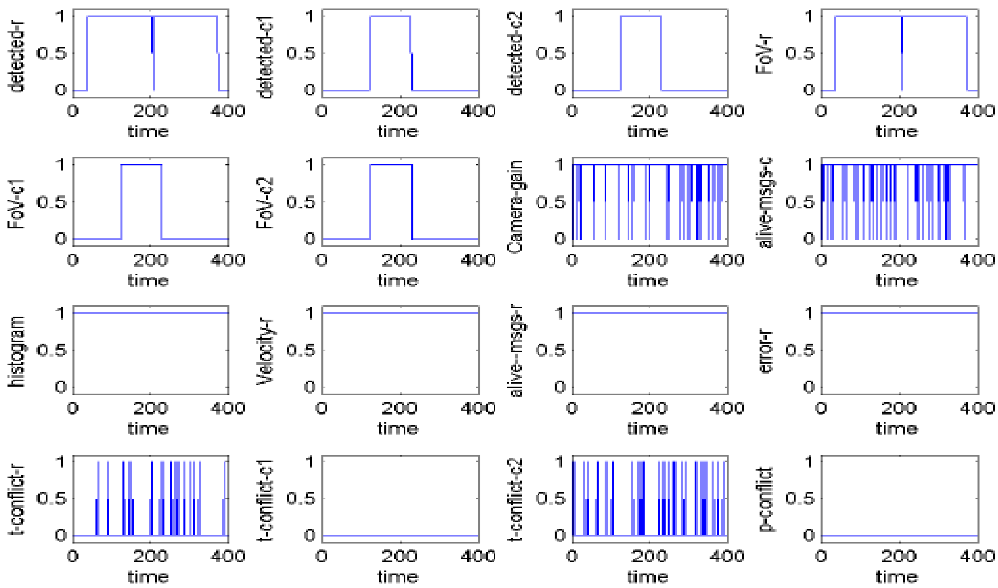

FIGURE 3: Input Data for scenario 1.

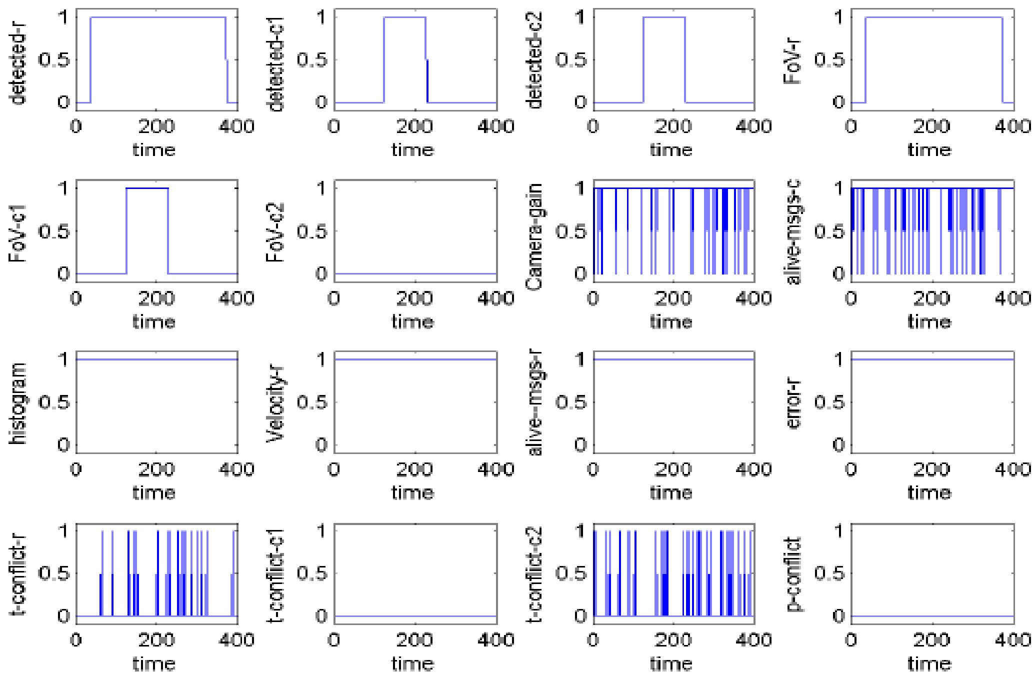

FIGURE 4: Input Data for scenario 2.

SMC Sampler Results

All results obtained using the SMC sampler are based on an SMC sampler with 100 particles. To give some intuition as to computational cost, processing the data associated with scenario 1 (which consists of three sensors and 398 time-steps) took approximately 77 seconds on an Intel i5 2.6 GHz PC running MATLAB 2012b. No optimisation for speed had been undertaken.

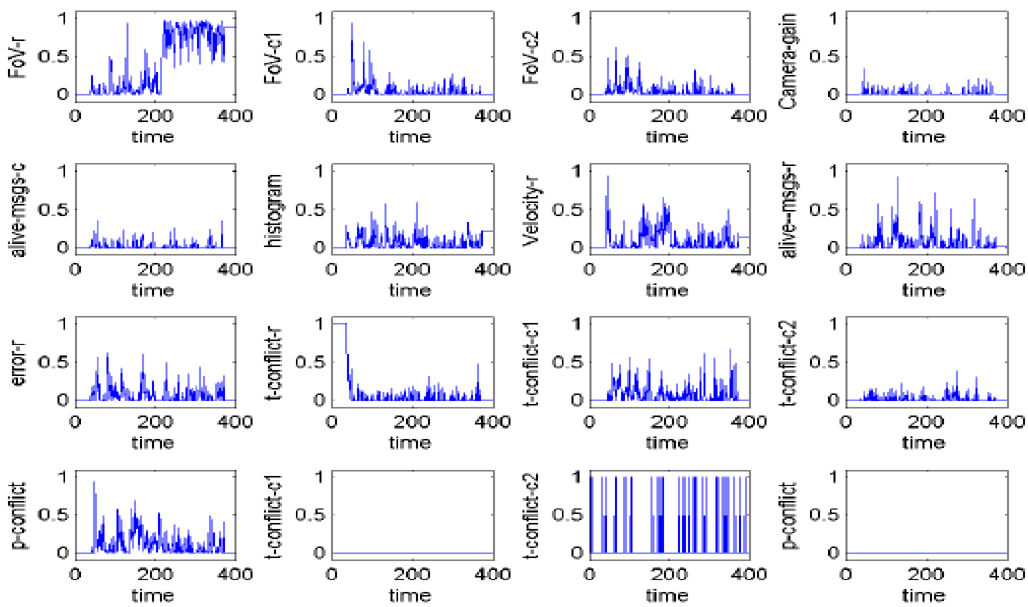

FIGURE 5: Probability that SMC sampler perceives that each binary variable is relevant to determining the radar's performance for scenario 1.

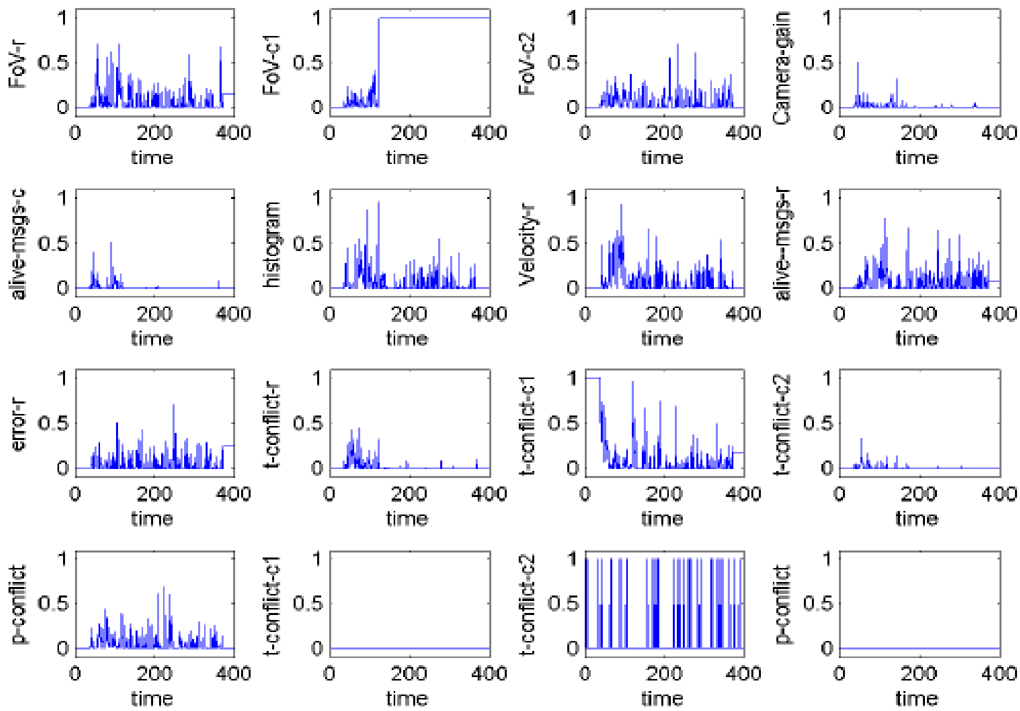

FIGURE 6: Probability that SMC sampler uses each binary variable is relevant to determining camera-1's performance in scenario 2.

To exemplify the results we obtain, figure 5 shows the probability that the SMC sampler uses each of the binary variables for the radar in scenario 1 and figure 6 shows the probability that the SMC sampler uses each of the binary variables for camera 1 in scenario 2.

5. Discussion

As should be apparent from figure 3, in scenario 1, the data indicates that detections from the radar occur when the target is in the radar's field of view (ie when the FoV-r variable is equal to 1). Figure 5 indicates that, up to approximately half way through the simulation, the SMC sampler has identified that all variables are unlikely to be having an influence on detection performance. There is limited (or no) evidence either in favour or against the hypothesis that the sensor's performance is being affected by the variables. This gives rise to the somewhat "noisy" outputs. Once the target exits the radar's field of view and is not detected at that point in time, the SMC sampler is able to detect the correlation between the target being in the field of view and it being detected: the system has learned that a target needs to be in the field of view of the radar for it to be detected.

In scenario 2, figure 4 illustrates that detections from camera 2 occur despite the field of view variable indicating that the target is outside the field of view. Figure 6 indicates that (once camera 2 starts detecting the target) the SMC sampler is able to correctly identify that the field of view for camera 1 is influencing camera 1's ability to detect targets. It is perhaps interesting that further analysis (not detailed here for reasons of brevity) indicates that the geometry of the scenario is such that the fields of view overlap and so, if both fields of view signals are "correct", the SMC sampler infers that both cameras' fields of view are important to determining both cameras' performance.

6. Conclusions

Two scenarios were considered in order to assess the SMC sampler performance. Based on these scenarios, we draw the following conclusions:

• A solution has been developed to the problem of learning how the environment affects the performance of sensors. The solution is based on SMC samplers configured to perform online learning of how binary variables influence detection performance.
• Analysis indicates that the internal signals generated are promising.

Future research will focus on enhancing the SMC sampler described here (eg to enable the SMC sampler's inputs to include continuous variables representing sensors' health, to use sufficient statistics to reduce the data storage requirements and to use truth-trees to succinctly represent truth tables within a redundant structure). It is also of interest to use alternative approaches to anomaly detection, e.g. based on Bayesian surprise (Itti L & Baldi P. 2006) and to embed the SMC sampler within an instantiation of the wider system described in section 3.

ACKNOWLEDGEMENTS

The authors thank Dstl and Innovate UK in general for their financial support to the project.

REFERENCES

ITTI, L. & BALDI, P. 2006. Bayesian surprise attracts human attention. *Advances in neural information processing systems*, **18**, 547.
LANE, R.O. & COPSEY, K.D. 2012, Track Anomaly Detection with Rhythm of Life and Bulk Activity Modelling. *15th International Conference on Information Fusion*.
MASKELL, S. 2012. An application of sequential Monte Carlo samplers: an alternative to particle filters for non-linear non-Gaussian sequential inference with zero process noise. *In Proc IET Data Fusion and Target Tracking Conference*.
ROUSE, S., YOUNG, R., & MCGRATH, B. 2010. "Airborne multisensor demonstrator program for persistent wide-area surveillance", *SPIE 7694*.
STRENS M, BAXTER J, & GARDNER A. A Generic Architecture for Hybrid Autonomous Decision Systems. *2nd SEAS DTC Technical Conference - Edinburgh 2007.*
ZAHEDI, S., SZCZODRAK, M., JI, P., MYLARASWAMY, D., SRIVASTA, M. & YOUNG, R. 2008. Tiered architecture for online detection, isolation and repair of faults in wireless sensor networks. *MILCOM 2008*.