SWEDISH DEFENCE
RESEARCH AGENCY

FOI-R--1413--SE
December 2004
ISSN 1650-1942

**User report**

Simon Ahlberg, Pontus Hörling, Karsten Jöred, Christian Mårtenson, Göran Neider, Johan Schubert, Hedvig Sidenbladh, Pontus Svenson, Per Svensson, Katarina Undén, Johan Walter

# The IFD03 Information Fusion Demonstrator -- requirements, methodology, design, and experiences

Command and Control Systems
SE-581 11 Linköping

SWEDISH DEFENCE RESEARCH AGENCY

Command and Control Systems
P.O. Box 1165
SE-581 11 Linköping

FOI-R--1413--SE

December 2004

ISSN 1650-1942

**User report**

Simon Ahlberg, Pontus Hörling, Karsten Jöred, Christian Mårtenson, Göran Neider, Johan Schubert, Hedvig Sidenbladh, Pontus Svenson, Per Svensson, Katarina Undén, Johan Walter

# The IFD03 Information Fusion Demonstrator -- requirements, methodology, design, and experiences

**Issuing organization**
FOI – Swedish Defence Research Agency
Command and Control Systems
P.O. Box 1165
SE-581 11 Linköping

**Report number, ISRN**
FOI-R--1413--SE

**Report type**
User report

**Research area code**
41 C4I

**Month year**
December 2004

**Project no.**
E 7097

**Sub area code**
41 C4I

**Sub area code 2**

**Author/s (editor/s)**
Simon Ahlberg
Pontus Hörling
Karsten Jöred
Christian Mårtenson
Göran Neider
Johan Schubert
Hedvig Sidenbladh
Pontus Svenson
Per Svensson
Katarina Undén
Johan Walter

**Project manager**
Per Svensson

**Approved by**
Martin Rantzer

**Sponsoring agency**
FM

**Scientifically and technically responsible**
Per Svensson

**Report title**

The IFD03 Information Fusion Demonstrator -- requirements, methodology, design, and experiences

**Abstract (not more than 200 words)**

The Swedish Defence Research Agency (FOI) has developed a concept demonstrator called the Information Fusion Demonstrator 2003 (IFD03) for demonstrating information fusion methodology for a future Network Based Defense (NBF) C4ISR system. The focus of the demonstrator is on real-time tactical intelligence processing at the division level in a ground warfare scenario. The demonstrator integrates force aggregation, particle filtering, and sensor allocation methods to create, dynamically update, and maintain components of a tactical situation picture. This represents an important step towards the goal of creating in real time a dynamic, high fidelity representation of a moving battalion-sized organization, based on sensor data as well as *à priori* intelligence and terrain information.

The motives behind this project, the fusion methods developed for the system, its scenario model and simulator architecture, as well as key aspects of its development process, are described. The main services of the demonstrator are discussed, and essential experience from the use and development of the system is shared. Further development of the techniques used in IFD03 may eventually permit concurrent tracking of solid objects and group objects, as well as more powerful sensor resource management methods. Also, studies are being carried out which are likely to lead to capability to automatically recognize certain kinds of tactical plans and intentions.

**Keywords**

Scenario simulation, demonstrator, force aggregation, terrain tracking, sensor resource management, sensor allocation, sensor modelling, terrain modelling

**Further bibliographic information**

**Language** English

**ISSN** 1650-1942

**Pages** 43 p.

**Price acc. to pricelist**

**Utgivare**
Totalförsvarets Forskningsinstitut - FOI
Ledningssystem
Box 1165
581 11 Linköping

**Rapportnummer, ISRN**
FOI-R--1413--SE

**Klassificering**
Användarrapport

**Forskningsområde**
4. Ledning, informationsteknink och sensorer

**Månad, år**
December 2004

**Projektnummer**
E 7097

**Delområde**
41 Ledning med samband och telekom och IT-system

**Delområde 2**

**Författare/redaktör**
Simon Ahlberg
Pontus Hörling
Karsten Jöred
Christian Mårtenson
Göran Neider
Johan Schubert
Hedvig Sidenbladh
Pontus Svenson
Per Svensson
Katarina Undén
Johan Walter

**Projektledare**
Per Svensson

**Godkänd av**
Martin Rantzer

**Uppdragsgivare/kundbeteckning**
FM

**Tekniskt och/eller vetenskapligt ansvarig**
Per Svensson

**Rapportens titel (i översättning)**
Informationsfusionsdemonstratorn IFD03 -- krav, metodik, konstruktion och erfarenheter

**Sammanfattning (högst 200 ord)**

Totalförsvarets forskningsinstitut (FOI) har utvecklat en konceptdemonstrator, Information Fusion Demonstrator 2003 (IFD03), för att kunna demonstrera informationsfusionsmetodik för ett framtida nätverksbaserat lednings- och underrättelsesystem. Demonstratorns fokus är taktisk underrättelsebearbetning på divisionsnivå i realtid av ett markstridsscenario. Demonstratorn integrerar metoder för styrkeaggregering, partikelfiltrering och sensorallokering för att åstadkomma, dynamiskt uppdatera och underhålla komponenter i en taktisk lägesbild. Detta innebär ett viktigt steg mot målet att skapa en dynamisk, verklighetstrogen representation av ett rörligt förband av bataljonsstorlek, baserad på sensordata i kombination med i förväg känd underrättelse- och terränginformation.

Motiven bakom detta projekt, de fusionsmetoder som utvecklats för demonstratorn, dess scenariomodell och simulatorarkitektur, liksom viktiga aspekter av utvecklingsprocessen beskrivs. Demonstratorns huvudsakliga tjänster diskuteras, och väsentliga erfarenheter från användning och utveckling av systemet presenteras.

Vidareutveckling av de metoder som används i IFD03 kommer troligen att leda till förmåga att samtidigt följa solida mål och gruppmål, liksom till mer effektiva metoder för styrning av sensorresurser. Vidare pågår studier som troligen kommer att leda till förmåga att automatiskt känna igen vissa typer av taktiska planer och avsikter.

**Nyckelord**
Scenariosimulering, demonstrator, styrkeaggregering, terrängföljning, sensorresurshantering, sensorallokering, sensormodellering, terrängmodellering

**Övriga bibliografiska uppgifter**

**Språk** Engelska

**ISSN** 1650-1942

**Antal sidor:** 43 s.

**Distribution enligt missiv**

**Pris: Enligt prislista**

# The IFD03 Information Fusion Demonstrator - requirements, methodology, architecture, and experiences

Simon Ahlberg, Pontus Hörling, Karsten Jöred,
Christian Mårtenson, Göran Neider, Johan Schubert, Hedvig Sidenbladh,
Pontus Svenson, Per Svensson, Katarina Undén, and Johan Walter

Swedish Defence Research Agency
SE 172 90 Stockholm, Sweden

`firstname.familyname@foi.se`
`www.foi.se/fusion`

**Abstract**

The Swedish Defence Research Agency (FOI) has developed a concept demonstrator called the Information Fusion Demonstrator 2003 (IFD03) for demonstrating information fusion methodology for a future Network Based Defense (NBF) C4ISR system. The focus of the demonstrator is on real-time tactical intelligence processing at the division level in a ground warfare scenario.

The demonstrator integrates force aggregation, particle filtering, and sensor allocation methods to create, dynamically update, and maintain components of a tactical situation picture. This represents an important step towards the goal of creating in real time a dynamic, high fidelity representation of a moving battalion-sized organization, based on sensor data as well as *a priori* intelligence and terrain information.

The motives behind this project, the fusion methods developed for the system, its scenario model and simulator architecture, as well as key aspects of its development process are described. The main services of the demonstrator are discussed, and essential experience from the use and development of the system is shared.

**Keywords:** Scenario simulation, demonstrator concept, force aggregation, terrain tracking, sensor resource management, sensor allocation, sensor modelling, terrain modelling.

# 1 Introduction

Information fusion requires methods for interpreting observational data in the context of complex models of reality, describing possible alternative future developments and evaluating their likelihood. In defense applications, fusion processes exploit a dynamic target situation picture produced by multisensor fusion, combining its information with relevant *a priori* information, in order to refine and interpret a battlespace situation picture. Ultimately, this semi-automatic intelligence interpretation process aims at delivering a comprehensive picture of the opponents' options and, based on an evaluation of these options, suggest their likely intentions.

In tactical ground-based scenarios, the *a priori* information will typically consist of terrain data, other important information about the environment, intelligence about the opponent's tactics, equipment and organization, known facts about the opponent's logistics situation, as well as other kinds of tactical knowledge (Steinberg and Bowman, 2001). Detailed geographical information will be needed, in particular to support calculation of sensor-to-target detection, classification, and tracking parameters, spatial reasoning about target behaviour based on tactical doctrine, and real-time terrain-dependent management of sensor resources.

Such data have previously been fused to an operational picture through time-consuming manual analyses and discussions. As the availability of sensor data explodes as a result of technological advances and fast decision-making emerges as a key requirement in command and control, this manual fusion process becomes a serious bottleneck.

Automating the tactical information fusion process remains a research issue. In particular, it is not yet clear how the basic methodology of computerized information fusion should be structured. Clearly, this structuring is not a purely technical task, but an issue which must eventually closely involve users of information fusion systems. In the Swedish defense research and development environment, there are few opportunities to achieve the required user involvement until some credible information fusion demonstration platforms have been introduced to prospective users.

Such platforms need to be based on scenario simulation, the only known methodology likely to offer the required versatility, dynamics, traceability, and repeatability of situations to be analyzed and techniques to be applied. Thus, simulation-based systems allowing prospective users first to learn about, later to try out and put strain on proposed information fusion methods and their user interfaces will be a prerequisite for the evolution and gradual user acceptance of these emerging methodologies. No less important is the requirement of being able to apply a sequence of fusion methods to various analysis problems, and then objectively evaluate their combined effectiveness and performance.

To begin addressing these user requirements, the Swedish Defence Research Agency (FOI) has developed a concept demonstrator called Information Fusion Demonstrator 2003 (IFD03) for demonstrating information fusion methodology for a future Network-Based Defense (NBF) C4ISR system. The theme of IFD03 is intelligence processing at the division level in a ground warfare scenario. In the demonstrator, information is transmitted from sensors to a Command and Control, C2, site. At the C2 site information is fused and interpreted.

Drawing upon progress reports presented in several conference papers (Hörling et al., 2002; Svensson and Hörling, 2003; Ahlberg et al., 2004; Schubert et al., 2004), this report presents principles, methods, system architecture, experiences and conclusions from the development of this system, which integrates research results in the areas of force aggregation, ground tracking, and sensor resource management within a state-of-the-art scenario simulation environment.

In Section 2 arguments are presented why scenario simulators are needed to provide early experience in integration, test, and demonstration of the many cooperating analysis methods and algorithms that will be needed in future high-level information fusion systems. Overall research goals of the IFD03 project are also discussed here.

Section 3 reviews the main concepts of the demonstrator, including its major use cases and component models, as well as the scenario used in the first demonstration in December 2003.

Section 4 surveys the fusion methods used in the demonstrator. These are Dempster-Shafer clustering and Dempster-Shafer template matching for force aggregation (Section 4.1), probability hypothesis density (PHD) particle filtering for ground vehicle tracking (Section 4.2), and random set simulation for sensor allocation (Section 4.3).

In Section 5 the software architecture of the IFD03 system is presented. The structure of the fusion node model is briefly described in Section 5.1. Design requirements and principles of the main doctrine and object models are surveyed in Section 5.2. Section 5.3 introduces the main object categories and their roles in the simulation, and briefly describes the data flow between the objects. In Section 5.4, a viewer's perspective of the demonstration is first introduced, then in Section 5.4.1, the organization of the visualization module in IFD03 is described. Modeling techniques used in creating the environment model of the demonstrator are discussed in Section 5.5.

Section 6 discusses the development environment and process created for IFD03. In Section 6.1, experience from using and extending a generic simulation development framework for simulation development is shared. Section 6.2 discusses the pros and cons of using MATLAB and

FLAMES together while developing the IFD03 simulator. Section 6.3 notes the need for version management systems.

Section 7 concludes the paper.

# 2 Project rationale

In December 2003, our project completed the development of IFD03 and performed a demonstration for an invited audience of tactical intelligence, C2 methodology, and information technology specialists.

IFD03 integrates methods related to different fusion “levels” (Steinberg and Bowman, 2001), specifically *multisensor-multitarget tracking*, *force aggregation*, and *reactive multisensor management*. It exchanges data in simulated real time in both directions between the scenario simulator and the fusion system. It has three closely associated main capabilities: to provide a test bed for methodology in information fusion, to provide a supporting scenario simulator for the generation of adequately realistic sensor and intelligence reports used as input to the fusion processes, and to offer software tools, terrain models, and other prerequisites for visualization both of the development of the scenario over time and of selected characteristics and effects of the fusion processes.

## 2.1 Purpose of the demonstrator

The main purpose of the demonstrator project was to provide a research platform for experimentation with specific research issues, in particular the interplay between different modeling techniques used to address subtopics in this research area, as well as to create a means of spreading knowledge to interested parties about the current state of research in information fusion.

Most scientific approaches to understanding specific aspects of reality have to be based on “reductionistic” abstraction and isolation of each aspect considered. On the other hand, in scenario-based forecasting models based on understanding already obtained by reductionist approaches, many more, if not all, significant complexities of the real system may be represented. Thus, *e. g.*, during the last half-century, weather forecasting has gradually developed, not primarily by discoveries of new, meteorologically significant physical phenomena, but by a combination of better mathematical models of the underlying physics, improved algorithms for their evaluation, improved data collection and exploitation in models, and last but not least, a gradually increased complexity and sophistication of integrative, synthetic environment forecasting simulators, made possible by the exponential growth in computer capacity.

Even though information fusion adds the serious complication of hidden, antagonistic human decision-making to the physical processes of weather forecasting models we believe that the success of such modeling could provide inspiration for information fusion research, although this research has a long way to go before it can claim any comparable success (Hall and Steinberg, 2001). So when will information fusion methodology have progressed sufficiently to make meaningful use of synthetic environment scenario simulators? Out of conviction that all necessary ingredients of complex forecasting models need to evolve together, we argue here that this is already the case.

Scenario-based simulation is often the only methodology available for systematic characterization and analysis of the kind of systems that are of interest in information fusion research. This methodology permits experimentation with various methods, configurations, and parameter values, evaluation of the effectiveness and efficiency of algorithms and modeling methods in relation to a reasonably realistic approximation of the final application environment, as well as verification that all problem-relevant components have been included and modelled on an adequate level of resolution. Also, it supports the establishing of a balanced system design, by allowing discovery and early elimination of vague concepts and unsolved or inadequately treated subproblems, as well as system performance bottlenecks. Design proposals which do not work even in a simplified synthetic environment can be identified and quickly eliminated, while methods which look promising can be selected for deeper analysis.

Additional potential advantages from using a simulation-based R&D process in information fusion include:

- shorter turn-around time and lower cost for the modelling activity; this can be exploited to create a close dialog with prospective users and customers,
- higher quality through improved opportunities to pre-test a proposed system in synthetic but increasingly realistic and probably ultimately dangerous scenarios,
- improved basis for the estimation of total system construction costs.

The above-mentioned concept of reactive multisensor management requires that sensor control messages based on fusion results can be fed back to the sensors in (simulated) real time. This suggests an architecture where the entire multisensor data acquisition and fusion process is an integrated part of the scenario, in the guise of an acquisition management and information fusion function of a simulated C2 centre. Note that as long as requirements for man-in-the-loop capabilities do not exist visualization can be done off-line, eliminating real time constraints on simulation processing. Such an architecture is employed in IFD03.

## 2.2 Research goals and issues

We view Level 2 information fusion (Hall and Llinas, 2001) as the interpretation of a flow of observations in terms of a model of a physical process in space and time. This process describes the stochastic interaction between an observation system, a complex target system (such as a hierarchically organized enemy unit), and a complex environment. According to this view, what distinguishes Level 2 from Level 1 fusion is mainly the much higher complexity of the target and environment models, *e. g.*, involving imperfectly known relationships more or less influenced by rule systems such as military tactical doctrine, which affect the behavior of the target system in a way that needs to be stochastically modeled.

The information fusion research at FOI rests on a few basic methodology principles, *i. e.*, cooperation between methods on fusion levels 1, 2, and 4, a tight coupling between a qualified synthetic environment and models of sensor behavior, target force behavior, and communication.

The methodology uses finite set statistics (Mahler, 2000), Dempster-Shafer clustering and template matching (Schubert, 2003b; Schubert, 2003a; Schubert, 2004), and particle filtering (Gordon et al., 1993; Arulampalam et al., 2002). The IFD03 project focuses on analysis, evaluation, and presentation of new methodologies for a collection of important subproblems in automatic information fusion: ground target tracking, force aggregation, multisensor management, and short term situation prediction.

We expect that a combination of the above-mentioned techniques may eventually permit concurrent tracking and short-term prediction of both solid objects (*e. g.*, vehicles) and group objects (*e. g.*, ground force units), logically connected via uncertain and vague information in the shape of doctrinal rules and communication capability.

The new demonstrator system is an extensible research and demonstration platform, where new methodological ideas can be realized, evaluated and demonstrated, and where various aspects of increasingly complex network-based information fusion systems can be tested in reasonably realistic scenarios. Whereas our previous information fusion projects have focused on method and algorithm development for various specific problems, in particular clustering, aggregation, and classification of force units (Cantwell et al., 2001) and sensor management (Xiong and Svensson, 2002; Johansson et al., 2003), the development tools associated with the new platform are intended to support substantial reuse, including evolutionary extension and rewriting, of both software and simulation scenario descriptions.

# 3 Conceptual overview

The IFD03 system was used to perform a demonstration in mid-December 2003, based on a simple battalion-level ground force attack scenario. The demonstration event consisted of a 30 minute replay session, corresponding to 75 minutes of real time. The scenario development was prerecorded during several hours of simulator runtime. Surveillance information was generated during the simulation by a set of sensor models, dynamically instantiated as a collection of distributed actors interacting with their scenario environment. The sensors delivered reports more or less continuously to a fusion node, symbolizing a future division-level intelligence staff.

The demonstrator implementation is based on a combination of three large development environments (see Section 6), the *problem solving environment* (for an in-depth study of this concept, see (Walker et al., 2000)) MATLAB$^{TM}$ (*MATLAB*, 2004), the *simulation framework* FLAMES$^{TM}$ (*FLAMES*, 2004), and the *terrain modelling system* TerraVista$^{TM}$ Pro Builder (*TerraVista*, 2004). In the project, FLAMES and MATLAB were tightly integrated, and FLAMES' new handling of advanced terrain models, generated by TerraVista, was specified and at least partly financed. Finally, the FLAMES software for visualization of simulation results using the new terrain modelling feature was restructured and both functionally and computationally substantially improved.

IFD03 can be characterized as an executable model of a two-party game between a multi-target and a fusion node. Technically, services are implemented as "cognitive models", *i. e.*, behavioral submodels of simulated actor models.

## 3.1 Use cases and actors

The major use cases (Jacobson, 1994) we had in mind when creating the system were:

- performing a demonstration addressing a possibly "infofusion-naive" audience. This is communication, not research, but could be developed into a methodology to present, visualize, and later analyze in detail properties of new components and subsystems,

- performing studies and experiments with sensor models, terrain and other environment models, fusion methods, doctrine models, scenario assumptions, etc., in various combinations, to test different hypotheses about possibilities and limitations related to Network-Based Defence (NBD) and information fusion,

- developing methodology and models for information fusion, *i. e.*, specification, development, and testing of new methods and fusion concepts. The size and complexity of a full-scale scenario-based simulator can be a severe drawback early in the research and development process, prompting the question: how should detailed studies in various separate test environments be structured to eventually support system demonstrations involving the complete demonstrator platform?

The primary types of objects to be involved in our December 2003 demonstration use-case were specified as (Svensson and Hörling, 2003):

- "red" (adversary) forces of battalion strength, consisting of several mechanized and armoured subunits,

- "blue" (own) division-level intelligence staff (fusion node), which can automatically and very rapidly communicate digital information with reconnaissance resources,

- blue home guard soldiers who observe the adversary advance using binoculars and possibly other devices,

- blue surveillance UAVs controlled by radio from the fusion node, carrying video or IR cameras or laser radar, or some combination of such sensors,

- blue communications intelligence (COMINT) surveillance units which can measure bearings to radio transmitters and analyze radio signals (but not decode their messages). They communicate measured directions and signaling timings to the fusion node,

- blue ground target sensor networks capable of detecting moving ground targets. Under favorable environmental conditions, target type and sometimes identity may be concluded from these detections.

Red and blue ground units move largely according to doctrinal rules on or near roads. Their speed and movement pattern is influenced also by road and terrain trafficability, varying between vehicle and unit types. Blue UAVs fly according to a simple dynamic model, while immediately obeying control messages from the fusion node. The control messages are generated by the fusion service sensor-allocation. The fusion node uses the sensor information as input to aggregation, tracking, and sensor management processes (see Section 4) to achieve the best possible situation characterization, given the modelling constraints inherent in the demonstrator system.

### 3.2 Scenario

The scenario takes place in May 2015. Tension in the Baltic Sea area has grown gradually over several years and the state of alert of the Swedish defence has been raised. At the outbreak of the war a number of concurrent events occur. Of these, a “trojan horse” landing at the ferry harbour at Kapellskär is judged to constitute the greatest threat. If the adversary is allowed to move further inland towards the town of Norrtälje and occupy the lake passes to the south of it, they will be difficult to defeat with available own resources.

When the defending battalion commander has received his action orders he wants to obtain as fast as possible a detailed picture of the adversary’s size, composition, and activity in order to be able to judge their action options and decide his own. The only intelligence sources available at the time of the landing are four Home Guard patrols deployed at strategic points along the enemy advance routes, Figure 1. The battalion’s UAV group is ordered to immediately direct two UAVs for reconnaissance above Rådmansö, to obtain a more detailed picture of the situation.

Figure 2 shows the situation at 17.45. The two UAVs directed to Rådmansö have contributed to the rather detailed situation picture. The chief intelligence officer is able to state that the adversary force consists of a mechanized battalion reinforced by antiaircraft and artillery units, advancing along two roads towards Norrtälje. However, as the bridge across Åkeröfjärden is demolished by the Home Guard at 17.30, the advance along the main road is delayed.

The final phase of the scenario involves the continued but delayed adversary march towards the lake passes. As the sensor platforms of the defense become fewer and eventually only a single UAV remains, it becomes critical to utilize that resource in an effective manner, in order to estimate in advance which routes the adversary is likely to use during the final phase of their march, as well as when they will reach the lake passes. With this objective, the automatic sensor resource manager of the fusion node is tasked to find the best route for the UAV and to decide where to drop its deployable ground sensor network.

## 4 Fusion methods

The analysis module has three main tasks and uses four different methods. The tasks are force aggregation, ground vehicle tracking and sensor allocation. They are performed using Dempster-Shafer clustering and template matching for force aggregation, probability hypothesis density (PHD) particle filtering for ground vehicle tracking, and random set simulation for sensor allocation.

Figure 1: Information collection situation at Rådmansö 17.00. Four Home Guard (HV) patrols are located at critical points along the adversary's approach route. A bridge is located at O.

## 4.1 Force aggregation

In force aggregation, sensor reports with given position, time, and type information are used. Here, force aggregation is defined as a sequence of two processes: (1) association of intelligence reports, objects or units (depending on hierarchical level) by a clustering process; (2) classification of cluster content through comparison with templates.

Initially, all pairs of intelligence reports are evaluated, to find whatever is against an association of these two reports to the same object: Wrong type of vehicle? (note that type assignments are allowed to be more or less specific) Is distance too long or too short? Wrong direction? Wrong relative positions? *etc*. This yields a conflict matrix which is supplied to the clustering algorithm. We use the Dempster-Shafer clustering algorithm (Schubert, 1993; Bengtsson and Schubert, 2001; Schubert, 2003a; Schubert, 2004) to partition the set of reports into subsets corresponding to objects, and classify the objects by fusing all intelligence using Dempster's rule. This method continues upwards level by level. At the vehicle to platoon level, vehicles are clustered and groups of vehicles are classified using Dempster-Shafer matching against templates (Schubert, 2003b). At all levels in clustering and template matching several alternative hypotheses are carried. Each alternative hypothesis is matched and evaluated against all templates and a weighted average of fitness is calculated for each potential template.

A screen picture from the demonstrator showing the result of automated force aggregation at the platoon level is shown in Figure 3. This method is currently developed up to the battalion level.

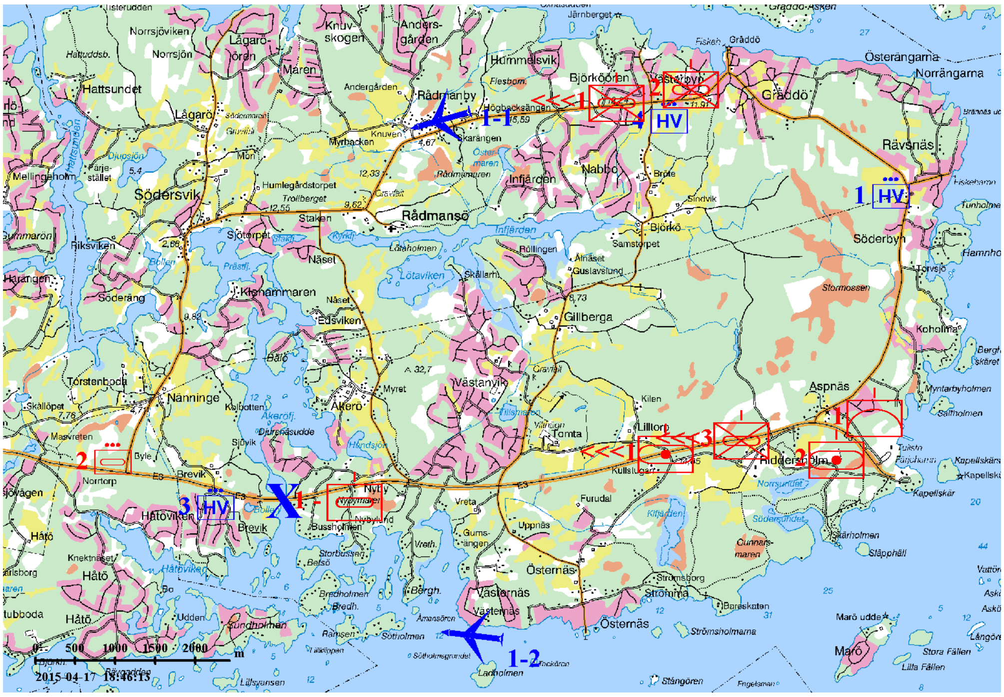


Figure 2: Situation picture at 17.45. The bridge is destroyed, denoted by X.

A few other approaches to force aggregation are (Biermann, 1998; Lorenz and Biermann, 2002; Johnson and Chaney, 1999).

### 4.1.1 Conflict matrix

There is one conflict matrix for each aggregation level. The conflict matrix element $\mathbf{C}_{\mathbf{ij}}$ contains the conflict between the entities $i$ and $j$. The matrix is symmetric and contains zeros on the diagonal.

When computing the conflict matrix for the reports, the conflict between two reports is based on their vehicle type, on how fast a vehicle must travel in order to cause the two reports and on how much their directions differ. When computing the conflict matrix for vehicles and units, the conflicts are based on doctrine data that specify how far apart the objects appear within their unit.

All entities – reports, vehicles and units – contain a classification of types, $\mathbf{T}$. However, the classification is uncertain, so we can only give probabilities for sets of types, representing the varying specificity of type assignments in the reports. The basic belief mass supporting that an entity is of type $\mathbf{A} \in \mathbf{T}$ is denoted $m(\mathbf{A})$. All basic belief functions in IFD03 are consonant, *i. e.*, the focal elements of the belief functions can be ordered by set inclusion, ensuring that their type conflicts are well-defined.

**Conflict matrix for reports.** The value $\mathbf{C}$ of an element in the conflict matrix is computed from the type conflict $\mathbf{C}^{\mathbf{t}}$, the speed conflict $\mathbf{C}^{\mathbf{s}}$ and the direction conflict $\mathbf{C}^{\mathbf{d}}$.

$$\mathbf{C} = 1 - (1 - \mathbf{C}^{\mathbf{t}})(1 - \mathbf{C}^{\mathbf{s}})(1 - \mathbf{C}^{\mathbf{d}}) \qquad (1)$$

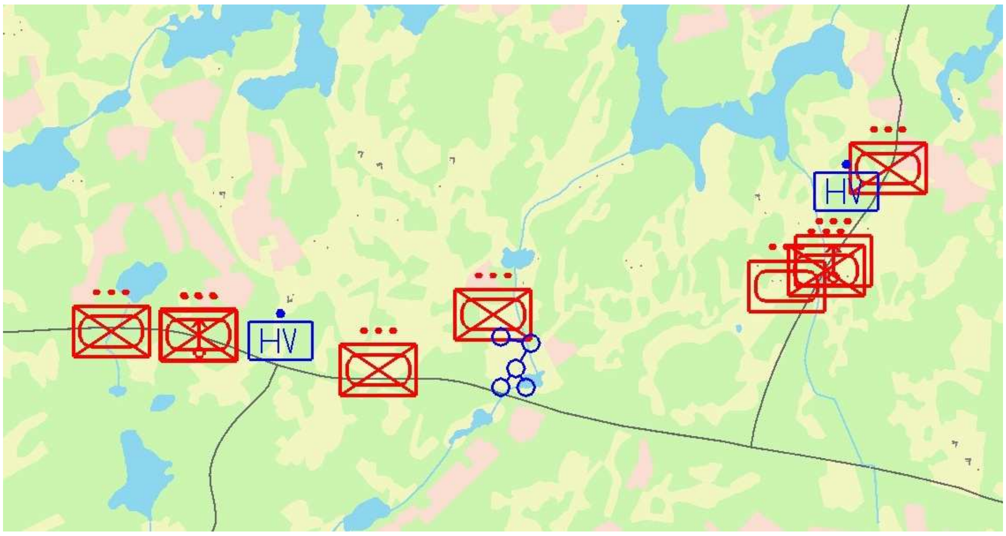


Figure 3: Force aggregation of vehicles into platoons.

The type conflict between the entities $e_i$ and $e_j$ is given by Dempster's rule of combination (Dempster, 1968; Shafer, 1976):

$$\mathbf{C^t}_{e_ie_j} = \sum_{\substack{A \in e_i, B \in e_j \\ A \cap B = \emptyset}} m(A) \cdot m(B) \tag{2}$$

The speed conflict, $\mathbf{C^s}$, is obtained by calculating the speed at which a vehicle must travel, in order to have caused both reports, see Figure 4.

The direction conflict, $\mathbf{C^d}$, is calculated in an analogous way by computing the difference between the directions of movement in the two reports. For details see (Cantwell et al., 2001).

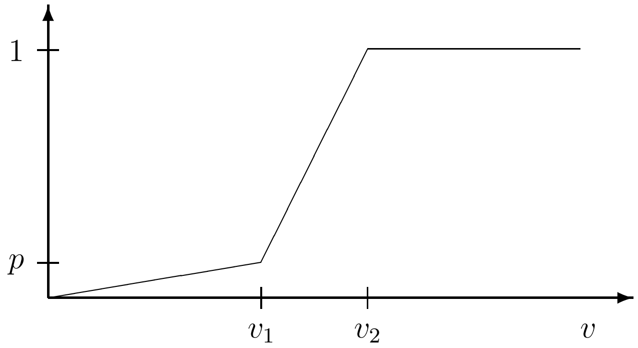


Figure 4: The relationship between speed and conflict.

**Conflict matrix for vehicles and units.** If there is no enemy unit that contains the estimated entity types of both reports, the conflict should be large. If there is such a unit, the conflict is determined by the current estimate of distance between the units and the maximum allowed distance according to doctrine.

For each level – vehicles, platoons, companies,… – a distance matrix, **DM**, is defined so that the allowed distance between $\mathbf{T}_a$ and $\mathbf{T}_b$ is $\mathbf{DM}_{ab}$.

The conflict between entity $e_i$ and $e_j$ is given by:

$$\mathbf{C}_{e_i e_j} = \sum_{A \in e_i, B \in e_j} C_{AB} \cdot m(A) \cdot m(B) \tag{3}$$

where

$$C_{AB} = \min_{a \in A, b \in B} c_{ab} \tag{4}$$

and

$$c_{ab} = \begin{cases} 1 & d > \mathbf{DM}_{ab} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $d$ is the distance between entities $e_i$ and $e_j$.

### 4.1.2 Clustering

In (Bengtsson and Schubert, 2001) a method for clustering intelligence reports based on their pairwise conflict was developed. This method was extended into a method capable of handling also pairwise attractions (Schubert, 2003a). Such evidence is not generated intrinsically in the same way as conflicts. Instead, it is provided by communication intelligence, indicating that two objects probably belong to the same unit (cluster) as they are in communication. Such information is made available from studying communication patterns obtained through COMINT, *e. g.*, if two objects are transmitting in sequence one may calculate a probability that they are in communication and thus belong to the same unit structure.

As conflicts push reports apart (into different clusters) and attractions pull them together (into the same cluster), using both can lead to an improved clustering result and faster computation. The conflict, as calculated in Section 4.1.1, and the attraction together form the basis for separating intelligence reports into clusters. A high conflict between two intelligence reports is an indication of repellency that they do not belong to the same cluster. The higher the conflict, the less credible it is that they belong to the same cluster.

External attracting metalevel evidence is represented as a pairwise piece of evidence, where $p_{ij}$ is a degree of attraction.

The best partitioning of all intelligence reports is found by a clustering process (Schubert, 2004) which minimizes a function $m_{\{\chi_a\} \oplus \chi}(\neg AdP)$ with a proposition that this is not an “adequate partition” AdP. This function was derived by combining the conflicting information $m_{\{\chi_a\}}$ with the attracting metalevel evidence $m_{\chi}$.

Approximately this function can be written as

$$m_{\{\chi_a\}\bigoplus\chi}(\neg AdP) \approx [1 - \prod_{(ij)|\forall a.e_i \wedge e_j \notin \chi_a} (1-p_{ij}) \times \prod_a \prod_{(ij)|e_i \wedge e_j \in \chi_a} (1-c_{ij})] \tag{6}$$

Clustering of the intelligence reports is done by neural clustering using Potts spin theory (Potts, 1952; Chaikin and Lubensky, 1995). The Potts spin problem consists of minimizing an energy function

$$E = \frac{1}{2} \sum_{i,j=1}^{N} \sum_{a=1}^{q} (J_{ij}^{-} - J_{ij}^{+}) S_{ia} S_{ja} \tag{7}$$

by changing the states of the spins $S_{ia}$'s, where $S_{ia} \in \{0,1\}$ and $S_{ia} = 1$ means that report $i$ is in cluster $a$. $N$ is the number of intelligence reports and $q$ the number of clusters. This model serves as a clustering method if $J_{ij}^{-}$ is used as a penalty factor when reports $i$ and $j$ are in the same cluster, and $J_{ij}^{+}$ when they are in different clusters.

The minimization is carried out by deterministic annealing (Peterson and Söderberg, 1989). For computational reasons a mean field model is used, with $V_{ia} =< S_{ia} >$, $V_{ia} \in [0,1]$, in order to find the minimum of the energy function. The Potts mean field equations are

$$V_{ia} = \frac{e^{-H_{ia}[V]/T}}{\sum_{b=1}^{q} e^{-H_{ib}[V]/T}} \tag{8}$$

where

$$H_{ia}[V] = \sum_{j=1}^{N} J_{ij} V_{ja} - \gamma V_{ia}, \tag{9}$$

and $V$, $T$, $H_{ib}$ and $\gamma$ are parameters of the annealing process.

In order to map the function $m_{\{\chi_a\}\bigoplus\chi}(\neg AdP)$ onto a Potts spin neural network it must be rewritten as a sum of terms.

Minimizing the right member of Equation (6) is equivalent to minimizing the expression:

$$\sum_{(ij)|\forall a.e_i \wedge e_j \notin \chi_a} -log(1-p_{ij}) + \sum_a \sum_{(ij)|e_i \wedge e_j \in \chi_a} -log(1-c_{ij}) \tag{10}$$

To apply the Potts model to Dempster-Shafer clustering, interactions $J_{ij}^{-} = -log(1-c_{ij})\delta_{|A_i \cap A_j|}$ and $J_{ij}^{+} = -log(1-p_{ij})(1-\delta_{|A_i \cap A_j|})$ are used in the energy function (Equation (7)), where $A_i$ is the focal element of the simple support function $e_i$ and $\delta$ is a Kronecker function

$$\delta_{|A_i \cap A_j|} \equiv \begin{cases} 1 & A_i \cap A_j = \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

and (in Figure 5)

$$\delta_{ij} \equiv \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{12}$$

In Figure 5 an algorithm for minimizing the energy function through iteration of Equations (8) and (9) is shown.

```
INITIALIZE
    $K$ (number of clusters); $N$ (number of intelligence reports)
    $J^-_{ij} = -log(1 - c_{ij})\delta_{|A_i \cap A_j|} \forall i, j;$
    $J^+_{ij} = -log(1 - p_{ij})(1 - \delta_{|A_i \cap A_j|}) \forall i, j;$
    $s = 0; t = 0; \epsilon = 0.001; \tau = 0.9; \gamma = 0.5;$
    $T^0 = T_c$ (a critical temperature) $= \frac{1}{K} \cdot max(-\lambda_{min}, \lambda_{max})$,
        where $\lambda_{min}$ and $\lambda_{max}$ are the extreme eigenvalues of $M$,
        where $M_{ij} = J^-_{ij} - J^+_{ij} - \gamma\delta_{ij};$
    $V^0_{ia} = \frac{1}{K} + \epsilon \cdot rand[0, 1] \forall i, a;$
REPEAT
    REPEAT-2
        $\forall i$ Do:
            $H^s_{ia} = \sum_{j=1}^{N} (J^-_{ij} - J^+_{ij}) V_{ja}^{\left\{ \begin{smallmatrix} s+1 & j<i \\ s & j \geq i \end{smallmatrix} \right.} - \gamma V^s_{ia} \forall a;$
            $F^s_i = \sum_{a=1}^{K} e^{-H^s_{ia}/T^t};$
            $V^{s+1}_{ia} = \frac{e^{-H^s_{ia}/T^t}}{F^s_i} + \epsilon \cdot rand[0, 1] \forall a;$
            $s = s + 1;$
    UNTIL-2
        $\frac{1}{N} \sum_{i,a} |V^s_{ia} - V^{s-1}_{ia}| \leq 0.01;$
    $T^{t+1} = \tau \cdot T^t;$
    $t = t + 1;$
UNTIL
    $\frac{1}{N} \sum_{i,a} (V^s_{ia})^2 \geq 0.99;$
RETURN
    $\{\chi_a | \forall S_i \in \chi_a, \forall b \neq a V^s_{ia} > V^s_{ib}\};$
```

Figure 5: Pseudo code for clustering algorithm.

### 4.1.3 Number of clusters

In order to estimate the correct number of clusters, the conflict function which results from clustering with different numbers of clusters is calculated and analyzed. Of course, where the number of clusters is too small the conflict will be high. Where the number of clusters is too large a very small conflict, emanating from measurement errors, will remain.

It was found experimentally that there is a change of behavior in the conflict function near the correct number of clusters, which was determined as follows. First, the logarithm of the total weight of conflict as function of the number of clusters was computed from empirical data, see Figure 6. Second, the concave lower envelope of this function was determined using a convex hull algorithm. Third, at an arbitrary abscissa, the envelope function was bisected in a left and a right part, each of which were then fitted by least squares to a straight line. Fourth, the acute angle between the two lines was maximized over all bisection abscissas and the maximizing abscissa

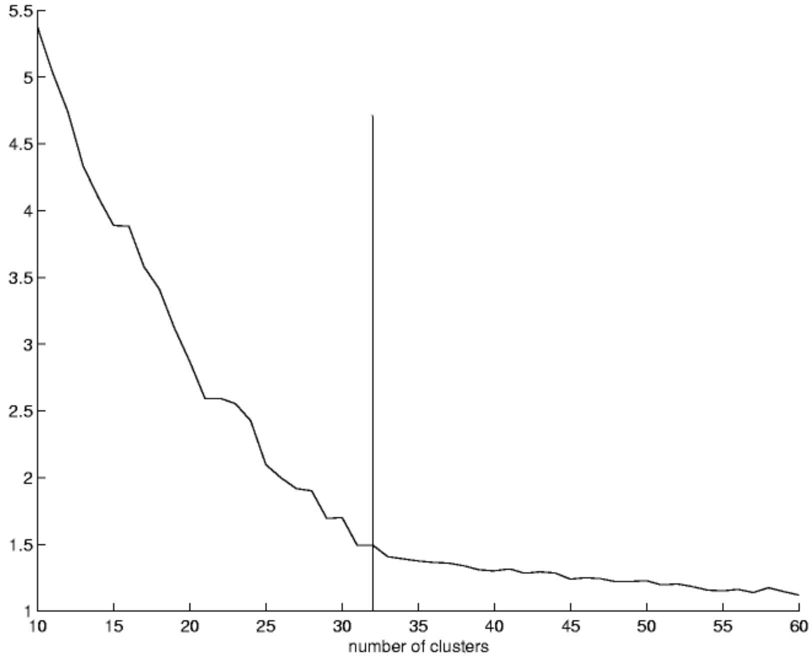


Figure 6: Logarithm of the total weight of conflict, showing a qualitative change in behavior near the correct number of clusters. The result in this figure is based on clustering all intelligence obtained from sensors observing the advance of one mechanized battalion.

was chosen as number of clusters.

Experimental tests of this algorithm using sensor reports and vehicles from the scenario showed good correspondence with the total number of observed vehicles.

### 4.1.4 Classification

The classification process deals with intelligence reports on a cluster-by-cluster basis. Looking at intelligence in one of the clusters, the classification from intelligence using templates takes place in two phases. First, all intelligence reports within the cluster are combined, then the combined intelligence is compared with all available templates.

In the combination of intelligence a special concern is the representation used. As the reports in general are not reports about the same object or group of objects, one can not use a simple representation dealing only with object type. Instead, a more complex representation has to be used, that allows keeping track of different objects and their possible types. Intelligence reports that are judged to be referring to the same object or group of objects are precombined and henceforth viewed as one intelligence report. In this way, all intelligence reports in the cluster under investigation can be combined, providing the opportunity to investigate different resulting hypotheses regarding force composition.

When selecting a template for the current cluster, a best match between template and fused intelligence is sought. Since intelligence consists of multiple alternative hypotheses with an accompanying uncertainty, every hypothesis, to its degree of uncertainty, must be taken into account. As these hypotheses are also nonspecific with regard to object type, *i. e.*, they refer to a subset of all possible types instead of to a single type, one cannot expect a perfect match for each type of object in the template. Instead, one should look for the possibility of a match between intelligence and template, *i. e.*, the absence of conflicts in number of items between what the intelligence proposes and what each available template requests for all subsets of types. With this measure a template can be selected for intelligence with nonspecific propositions.

Let us now focus on one subset $\chi_a$ and the aggregation of the intelligence in this subset. Let $TY$ be a set of all possible types of objects $\{TY_x\}$ where $TY_x$ is a type of vehicle or a type of unit, depending on which hierarchical level is analyzed.

The frame of discernment when fusing reports regarding different sets of objects that should be combined as fragments of a larger unit structure becomes

$$\Theta_{I_a} = \{\langle x_1, x_2, \ldots, x_{|I_a|}\rangle\} \tag{13}$$

where $x_i = (x_{i\bullet}n, x_{i\bullet}pt)$ is information regarding the $i^{th}$ set of objects with $x_{i\bullet}n \subseteq \{1, \ldots, N_{C_a}\}$ and $x_{i\bullet}pt \subseteq TY$. Here, $N_{C_a}$ is the maximum number of objects according to the intelligence in cluster *a*.

Comparing templates having specific propositions that are certain in what they are requesting with intelligence propositions that are not only uncertain but may also be nonspecific in what they are supporting can be a difficult task. The idea being used to handle this problem is to compare a candidate template with intelligence from the perspective of each and every subset of all possible types of objects $TY$.

To do this, one may investigate how much support a subset of $TY$ receives both directly and indirectly from intelligence and template, respectively. The support for a subset of $TY$ is added up from all propositions that are equal to, or itself a subset of this subset of $TY$. This is similar to the calculation of belief from basic probability numbers in Dempster-Shafer theory, except that one does not add basic probability numbers but natural numbers representing the number of objects of the proposed types.

Let $T$ be a set of all available templates $\{T_i\}$. Each template is represented by any number of slots $S_i^j$ where $S_{i\bullet}^j pt \in TY$ is a possible type from the set $TY$ and $S_{i\bullet}^j n$ is the number of that type in $T_i$.

Since there are several different alternative propositions in the intelligence regarding the type of objects and their corresponding number of objects, one needs to compare each potential template with these alternatives and let each proposition influence the evaluation. For each template a measure of fitness is found between the template and each proposition in the intelligence, separately.

A linear combination is then made, where each measure of fitness is weighted by the basic probability number of that proposition,

$$m_{\bigoplus J_a}(\langle x_1, x_2, \ldots, x_{|I_a|}\rangle), \tag{14}$$

giving

$$\pi_{\bigoplus J_a}(T_i) = \frac{1}{2} \sum_{\langle x_1, x_2, \ldots, x_{|I_a|} \rangle} m_{\bigoplus J_a}(\langle x_1, x_2, \ldots, x_{|I_a|} \rangle)$$

$$\left[ \max_{n \in SC_a(TY)} \left\{ \min \left[ \frac{n}{ST_i(TY)}, \frac{ST_i(TY)}{n} \right] \right\} \right.$$

$$\left. + \min_j \left[ \begin{cases} \max_{n \in SC_a(S^j_{a \bullet} pt)} \left\{ \min \left[ \frac{n}{ST_i(S^j_{a \bullet} pt)}, \frac{ST_i(S^j_{a \bullet} pt)}{n} \right] \right\}, & ST_i(S^j_{a \bullet} pt) > 0 \\ 1, & ST_i(S^j_{a \bullet} pt) = 0 \end{cases} \right] \right] \qquad (15)$$

where $S^j_{a \bullet} pt \subseteq TY$.

For each potential template $T_i$, the number of objects requested by the template from the perspective of subset $X \subseteq TY$ in Equation (15) is calculated as

$$ST_i(X) = \sum_{j | S^j_{i \bullet} pt \subseteq X_{\bullet} pt} S^j_{i \bullet} n, \forall X \subseteq TY \qquad (16)$$

and the number of objects supported by proposition $\langle x_1, x_2, \ldots, x_{|I_a|} \rangle$ of the intelligence from the perspective of subset $X \subseteq TY$ as

$$SC_a(X | \langle x_1, x_2, \ldots, x_{|I_a|} \rangle) = \sum_{i \,\left|\, \substack{x_i \in \langle x_1, x_2, \ldots, x_{|I_a|} \rangle \\ x_{i \bullet} pt \subseteq X_{\bullet} pt} \right.} x_{i \bullet} n, \forall X \subseteq TY \qquad (17)$$

The best template is selected if its matching value is above some threshold, Figure 7. While the fitness measure $\pi_{\bigoplus J_a}(\cdot)$ is used for aggregation from the current hierarchical level, one also needs the basic probability of the highest ranked template for any further aggregation from the next hierarchical level. Through a fitness weighted transformation, these templates will share this support in relation to their fitness towards the corresponding focal element in the intelligence.

The basic probability number of a template $T_i$ is found as

$$m_{\bigoplus J_a}(T_i) = \sum_{\langle x_1, x_2, \ldots, x_{|I_a|} \rangle \supseteq T_i} \left[ m_{\bigoplus J_a}(\langle x_1, x_2, \ldots, x_{|I_a|} \rangle) \frac{\pi_{\langle} x_1, x_2, \ldots, x_{|I_a|} \rangle (T_i)}{\sum_{\langle x_1, x_2, \ldots, x_{|I_a|} \rangle \supseteq T_j} \pi_{\langle} x_1, x_2, \ldots, x_{|I_a|} \rangle (T_j)} \right] \qquad (18)$$

The evidential force aggregation method makes it possible to aggregate uncertain intelligence reports with multiple uncertain and nonspecific propositions into recognized forces using templates.

## 4.2 Tracking

In the tracking module, the states of an unknown number of ground vehicles moving in terrain are maintained. The tracking is based on observations in the form of intelligence reports of ground position $\mathbf{y}$, ground speed $v$, and direction of motion $\theta$.

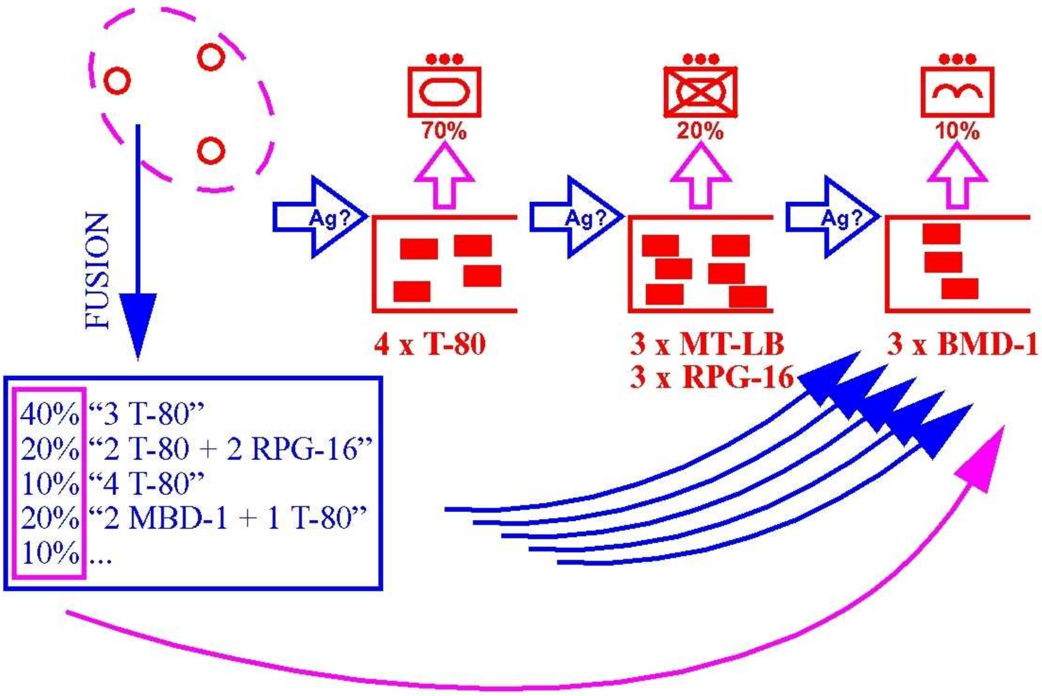


Figure 7: The intelligence is fused into several alternative hypotheses. Each hypothesis is evaluated against all templates to give an overall fitness for each template.

When tracking multiple targets in general, the size of the state-space for the joint distribution over target states grows exponentially with the number of targets. When the number of targets is large, this makes it impossible in practice to maintain the joint distribution over target states.

A mathematically principled approach to avoid the combinatorial explosion is to propagate only the first moment of the joint distribution, the *probability hypothesis density (PHD)* (Mahler and Zajic, 2001). This entity is briefly described in Section 4.2.1. It has the property that its integral over each sub-area $S$ in the state-space is the expected number of targets within this area. Peaks in the PHD can thus be regarded as estimated target states. Since the identities of objects are not maintained, there is no model-data association problem. However, the method has the drawback that no knowledge about dependencies in motion between objects can be represented. Also in Section 4.2.1, a particle filter (Gordon et al., 1993; Isard and Blake, 1998) implementation of PHD tracking, the *PHD particle filter*, is briefly described. For a thorough description, see (Sidenbladh, 2003). Particle filtering is suited for tracking with non-linear and non-Gaussian motion models, and is thus suitable for ground target tracking. The non-linear terrain-dependent motion model is described in Section 4.2.2.

In (Sidenbladh, 2003), the sensor visibility is assumed constant with respect to position and time. Here, we incorporate knowledge of sensor quality and field of view into the filter. This is described in Section 4.2.3.

### 4.2.1 PHD filtering

The number of vehicles (called targets below) to track is unknown and varies over time. This means that the targets at time $t$ is a *random set* (Goodman et al., 1997; Mahler, 2000) $\Gamma_t = \{\mathbf{X}_t^1, \ldots, \mathbf{X}_t^{N_t}\}$, where $\mathbf{X}_t^i$ is the state vector of target $i$ and $N_t$ is the number of targets in the set. A certain outcome of the random set $\Gamma_t$ is denoted $X_t = \{\mathbf{x}_t^1, \ldots, \mathbf{x}_t^{n_t}\}$. Similarly, the set of observations received at time $t$ is a random set $\Sigma_t = \{\mathbf{Z}_t^1, \ldots, \mathbf{Z}_t^{M_t}\}$, where $M_t$ can be larger than, the same as, or smaller than $N_t$. A certain outcome of the random set $\Sigma_t$ is denoted $Z_t = \{\mathbf{z}_t^1, \ldots, \mathbf{z}_t^{m_t}\}$.

For large numbers of targets, it is computationally intractable to keep track of every single target. A more tractable approach is then to represent the first moment of the full joint distribution, the probability hypothesis density (PHD) $D_{\mathbf{X}_t \,|\, \Sigma_{1:t}}(\mathbf{x}_t \,|\, Z_{1:t})$ (Mahler and Zajic, 2001; Sidenbladh, 2003), which is defined over the state-space $\Theta$ of one target instead of the much larger joint target space $\Theta^{N_t}$. The computational cost of propagating the PHD over time is much lower than propagating the full distribution.

The PHD has the properties that, for any subset $S \subseteq \Theta$, the integral of the PHD over $S$ is the expected number of targets in $S$ at time $t$:

$$E[|\Gamma_t \cap S|] = \int_S D_{\mathbf{X}_t \,|\, \Sigma_{1:t}}(\mathbf{x}_t \,|\, Z_{1:t}) \, d\mathbf{x}_t \, . \tag{19}$$

In other words, it will have local maxima approximately at the target locations. The integral of the PHD over $\Theta$ is the expected number of targets, $n_t$.

We now describe one time-step in the PHD filter, which is propagated using Bayes' rule (Mahler and Zajic, 2001; Sidenbladh, 2003). First, a *prior* PHD is predicted from the PHD and observations at the previous time-step. Then, new observations are used to compute the *likelihood* of this prior PHD. This results in a new *posterior* PHD. The steps are described below.

**Prediction.** The temporal model of the targets include birth (appearance of a target in the field of view), death (disappearance of a target from the field of view) and temporal propagation. Probability of target death is $p_D$ and of target birth $p_B$.

Target hypotheses are propagated from earlier hypotheses according to the dynamical model

$$\mathbf{X}_t = \phi(\mathbf{X}_{t-1}, \mathbf{W}_t) \tag{20}$$

where $\mathbf{W}_t$ is a noise term independent of $\mathbf{X}_{t-1}$ (Section 4.2.2). This gives

$$f_{\mathbf{X}_t \,|\, \mathbf{X}_{t-1}, \mathbf{Z}_{1:t-1}}(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}, \mathbf{z}_{1:t-1}) \equiv f_{\mathbf{X}_t \,|\, \mathbf{X}_{t-1}}(\mathbf{x}_t \,|\, \mathbf{x}_{t-1})$$

with no dependence on the history of observations $\mathbf{z}_{1:t-1}$.

Other target hypotheses are born from observations at the previous time instant (Sidenbladh, 2003) according to the model

$$\mathbf{X}_t = \phi(h_{\mathbf{X}_t}^{-1}(\mathbf{Z}_{t-1}, \mathbf{V}_{t-1}), \mathbf{W}_t) \tag{21}$$

where $\mathbf{V}_t$ is a noise term (Section 4.2.2). This model defines the birth pdf $f_{\mathbf{X}_t \,|\, \mathbf{Z}_{t-1}}(\mathbf{x}_t \,|\, \mathbf{z}_{t-1})$. To take all observations $\Sigma_t = \{\mathbf{Z}^1_t, \ldots, \mathbf{Z}^{M_t}_t\}$ into account for target birth, a birth PHD is defined from the set of birth pdfs as

$$D_{\mathbf{X}_t \,|\, \Sigma_{t-1}}(\mathbf{x}_t \,|\, Z_{t-1}) = \sum_{\mathbf{z}^i_{t-1} \in Z_{t-1}} f_{\mathbf{X}_t \,|\, \mathbf{Z}_{t-1}}(\mathbf{x}_t \,|\, \mathbf{z}^i_{t-1}) \,. \tag{22}$$

Given the models of motion, death and birth, the prior PHD (Mahler and Zajic, 2001) is estimated from the posterior PHD at the previous time instant as

$$\begin{aligned} &D_{\mathbf{X}_t \,|\, \Sigma_{1:t-1}}(\mathbf{x}_t \,|\, Z_{1:t-1}) = \\ &p_B D_{\mathbf{X}_t \,|\, \Sigma_{t-1}}(\mathbf{x}_t \,|\, Z_{t-1}) + \int (1 - p_D) f_{\mathbf{X}_t \,|\, \mathbf{X}_{t-1}}(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}) D_{\mathbf{X}_{t-1} \,|\, \Sigma_{1:t-1}}(\mathbf{x}_{t-1} \,|\, Z_{1:t-1}) \, d\mathbf{x}_{t-1} \,. \end{aligned} \tag{23}$$

**Observation.** We define $p_{FN}$ as the probability that a target is *not* observed at a given time step (the probability of false negative). This entity is further discussed in Section 4.2.3. Assuming that there are no spurious observations, the posterior PHD distribution is computed (Mahler and Zajic, 2001) from the prior as

$$D_{\mathbf{X}_t \,|\, \Sigma_{1:t}}(\mathbf{x}_t \,|\, Z_{1:t}) = \sum_{\mathbf{z}^i_t \in Z_t} f_{\mathbf{X}_t \,|\, \mathbf{Z}_t, \Sigma_{1:t-1}}(\mathbf{x}_t \,|\, \mathbf{z}^i_t, Z_{1:t-1}) + p_{FN} D_{\mathbf{X}_t \,|\, \Sigma_{1:t-1}}(\mathbf{x}_t \,|\, Z_{1:t-1}) \tag{24}$$

where

$$f_{\mathbf{X}_t \,|\, \mathbf{Z}_t, \Sigma_{1:t-1}}(\mathbf{x}_t \,|\, \mathbf{z}^i_t, Z_{1:t-1}) \propto f_{\mathbf{Z}_t \,|\, \mathbf{X}_t}(\mathbf{z}^i_t \,|\, \mathbf{x}_t) \; D_{\mathbf{X}_t \,|\, \Sigma_{1:t-1}}(\mathbf{x}_t \,|\, Z_{1:t-1}) \,, \tag{25}$$

which is a pdf (with the integral 1 over the state-space).

Using Equations (22), (23) and (24), the PHD is propagated in time. The result of the tracking is the estimated number of targets and the location of the detected maxima in the posterior PHD in each time step. An example of a posterior PHD is shown in Figure 9.

### 4.2.2 Terrain dependent motion and birth model

The state of a vehicle hypothesis at time $t$ depends (Equation (20)) on the state of the hypothesis at the previous time-step $t - 1$, and on the terrain at the vehicle position $\mathbf{y}_t$. Likewise, the state of a newly born particle (Equation (21)) depends on the observation from which it was born, and on the terrain at its position.

While the dependence on previous time can be modeled using linear dynamics, the terrain dependence is highly non-linear. For each position, the terrain can be retrieved from the database (see Section 5.5). The terrain influence on the vehicle position is represented as probability ratios $\pi_{\text{water}} = p_{\text{water}}/p_{\text{road}} = 0$, $\pi_{\text{forest}} = p_{\text{forest}}/p_{\text{road}} = 0.04$, $\pi_{\text{field}} = p_{\text{field}}/p_{\text{road}} = 0.2$, and $\pi_{\text{road}} = 1$ of the vehicle being positioned in different type of terrain.

The sampling from the conditional pdf $f_{\mathbf{X}_t \,|\, \mathbf{X}_{t-1}}(\mathbf{x}_t \,|\, \mathbf{x}_{t-1})$ is performed in two steps:

```
% prediction:
for $j \leftarrow 1 : n_{t-1}\mathcal{N}$
    % from previous time-step:  posterior $\boldsymbol{\xi}_{t-1}^j$.
    sample prior $\tilde{\boldsymbol{\xi}}_t^j$ from $f_{\mathbf{X}_t \,|\, \mathbf{X}_{t-1}}(\mathbf{x}_t \,|\, \boldsymbol{\xi}_{t-1}^j)$.
    compute prior weight $\varpi_t^j \leftarrow (1 - p_D)/\mathcal{N}$.
end for
for $i \leftarrow 1 : m_{t-1}$
    % from previous time-step:  observation $\mathbf{z}_{t-1}^i$.
    $J \leftarrow j$.
    for $j \leftarrow (J+1) : (J+\mathcal{N})$
        sample prior $\tilde{\boldsymbol{\xi}}_t^j$ from $f_{\mathbf{X}_t \,|\, \mathbf{Z}_{t-1}}(\mathbf{x}_t \,|\, \mathbf{z}_{t-1}^i)$.
        compute prior weight $\varpi_t^j \leftarrow p_B/\mathcal{N}$.
    end for
end for

% observation:
for $j \leftarrow 1 : (m_{t-1} + n_{t-1})\mathcal{N}$
    compute likelihood $\pi_t^{0,j} \leftarrow p_{FN}\varpi_t^j$.
end for
for $i \leftarrow 1 : m_t$
    for $j \leftarrow 1 : (m_{t-1} + n_{t-1})\mathcal{N}$
        compute likelihood $\tilde{\pi}_t^{i,j} \leftarrow \varpi_t^j f_{\mathbf{Z}_t \,|\, \mathbf{X}_t}(\mathbf{z}_t^i \,|\, \tilde{\boldsymbol{\xi}}_t^j)$.
    end for
    for $j \leftarrow 1 : (m_{t-1} + n_{t-1})\mathcal{N}$
        normalize likelihood $\pi_t^{i,j} \leftarrow \frac{\tilde{\pi}_t^{i,j}}{\sum_k \tilde{\pi}_t^{i,k}}$.
    end for
end for

% resampling:
expected number of targets $n_t = \sum_{i=0}^{m_t} \sum_{j=1}^{(m_{t-1}+n_{t-1})\mathcal{N}} \pi_t^{i,j}$.
for $j \leftarrow 1 : n_t\mathcal{N}$
    monte carlo sample posterior $\boldsymbol{\xi}_t^j$ from weighted set
    $\bigcup_{i=0}^{m_t} \{(\tilde{\boldsymbol{\xi}}_t^1, \pi_t^{i,1}), \ldots, (\tilde{\boldsymbol{\xi}}_t^{(m_{t-1}+n_{t-1})\mathcal{N}}, \pi_t^{i,(m_{t-1}+n_{t-1})\mathcal{N}})\}$.
end for
```

Figure 8: Pseudo code for time-step $t$ in a PHD particle filter.

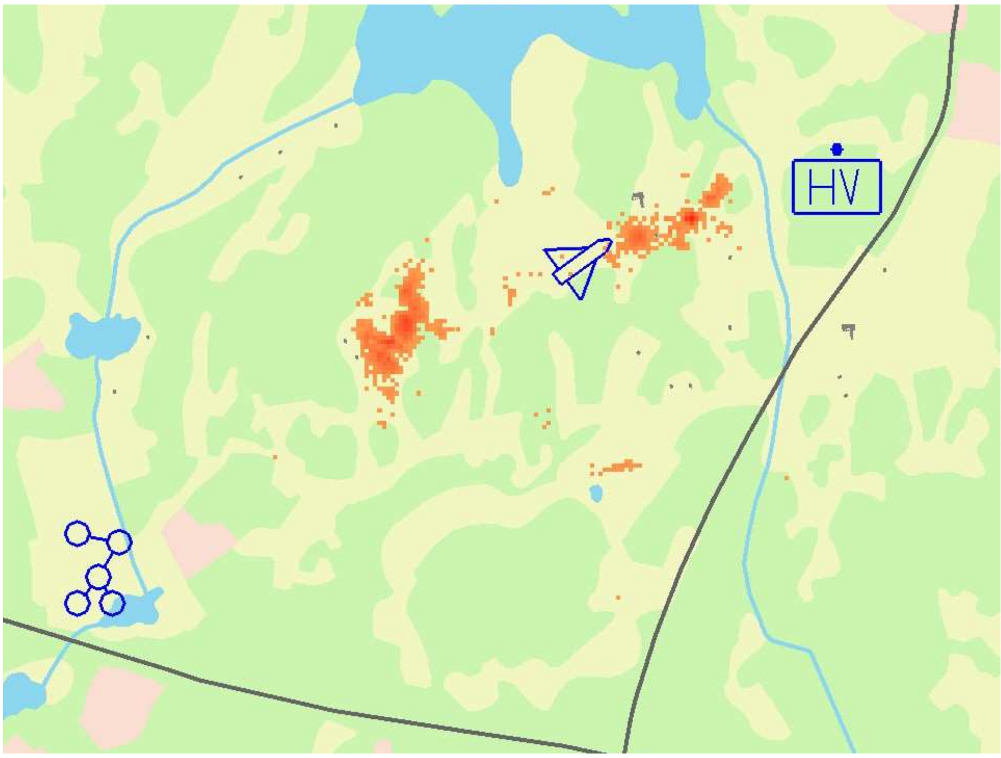


Figure 9: The posterior PHD represented as a set of particles. For greater visibility, the histogram over particle position is shown; the saturation of red in a certain sub-area (*i. e.*, histogram bin) represents the particle concentration in this area. The three blue symbols denote, from lower left to upper right, a ground sensor network, a UAV, and a home guard patrol.

1. Each particle in the old posterior cloud $\{\boldsymbol{\xi}_{t-1}^{1}, \ldots, \boldsymbol{\xi}_{t-1}^{n_{t-1}\mathcal{N}}\}$ is propagated using a first order linear dynamical motion model.
2. Each new particle is given a weight $\pi_{\text{terrain type}}$ depending on the terrain type at its position. A new particle cloud is then Monte Carlo sampled from the weighted particles.

Likewise, the sampling from the conditional pdf $f_{\mathbf{x}_t \,|\, \mathbf{z}_{t-1}}(\mathbf{x}_t \,|\, \mathbf{z}_{t-1}^{i})$ for each old observation $\mathbf{z}_{t-1}^{i}$ is performed as

1. $\mathcal{N}$ particles are sampled from observation $\mathbf{z}_{t-1}^{i}$ using a linear Gaussian model. The cloud is propagated using a first order linear dynamical motion model.
2. Identical to step 2. above.

### 4.2.3 Sensor position dependent detection rate

The probability of missed detection $p_{FN}$ varies over space and time, due to the type and fields of view of the different sensors.

To achieve a correct PHD estimate it is important to model this variance.

For each sensor $i$ in the system, the target detection probability $p_t^i$ and the present field of view $A_t^i$ is known at a given time-step $t$ (see also Section 5.3). The probability of missed detection in a certain position $\mathbf{y}$ can then be derived as

$$p_{FN}(\mathbf{y}) = \prod_{\mathbf{y} \in A_t^i} (1 - p_t^i) \ . \tag{26}$$

This varying $p_{FN}$ is used for propagation of the PHD over time as described in Equation (24).

This is intuitively obvious: if there are no sensors nearby, the prior particle distribution is accepted as posterior distribution as is. However, prior particles that come inside the field of view of a sensor are suppressed if there are no observations to support them. Accurate sensors with high $p_t^i$ suppress particles to a higher degree than sensors with a low $p_t^i$.

## 4.3 Sensor allocation

The aggregation module in IFD03 implements a simple version of sensor allocation based on *random set simulation*. As in the tracking module (Section 4.2), random sets (Goodman et al., 1997; Mahler, 2000) are used to formally describe the operation of our algorithm, and the probability hypothesis density is used to render the method computationally feasible.

The purpose of the sensor allocation method implemented in IFD03 is to determine which of several sensor allocation schemes should be used in a given tactical situation. Input to the module are a list of such allocation schemes or plans, a road network that describes the geography of the situation of interest, and positions of enemy units.

Pseudo code for our sensor allocation algorithm is shown in Figure 10. The algorithm, which will be described in more detail in future papers, works as follows. A density vector $x_0$ is given, which describes the positions of the units of interest at time $t = 0$. A set $S$ is defined, consisting of sensor allocation schemes and information about the road network on which the enemy is assumed to move.

Three different random sets are introduced:

1. $\mathbf{X}(t)$ denotes the positions of the enemy units at time $t$, conditioned on them being at $\mathbf{x}_0$ at time 0. It can be seen as representing a simulation of ground truth: the instance $x(t)$ of $\mathbf{X}(t)$ occurs with probability $P[\mathbf{X}(t) = x(t) | \mathbf{X}(0) = x_0]$. For simplicity of notation, the conditioning on $x_0$ is not explicitly shown in the following.

2. For each sensor allocation scheme $s \in S$ and instance $x(t)$ of the future ground truth, a set of possible observations $\mathbf{Z}(x(t), s, t)$ is calculated at time $t$. $\mathbf{Z}$ is also a random set; note that it depends on ground truth as well as allocation scheme.

3. Finally, we determine what our view of ground truth would be, given the set of observations $\mathbf{Z}$. This gives rise to the final random set, $\mathbf{Y}(t)$. $\mathbf{Y}(t)$ is our fusion system's approximation of the (simulated) ground truth $\mathbf{X}(t)$ using the observations $\mathbf{Z}$ obtained by deploying sensors according to sensor allocation scheme $s_i$.

Note that all of the random sets introduced are explicitly time-dependent. Here, an expression like $P[\mathbf{X}(t)]$ denotes the probability of the entire time-evolution of $\mathbf{X}(t)$, not just the probability at a specified time. $P[]$ can thus be seen as a probability density functional in the space of all explicitly time-dependent random sets. Further mathematical details on this will be presented in future work.

Determining which sensor allocation scheme to use is now done simply by comparing the assumed ground truth $x(t)$ to the fusion system's simulated view $y(t)$. For each instance $x(t)$ of $\mathbf{X}(t)$, the best $s$ can easily be determined by averaging over the ensembles of observations $\mathbf{Z}$ and simulated filter output $\mathbf{Y}$ entailed by that simulated ground truth. An allocation scheme is good if the simulated filter gives a good approximation of the simulated ground truth. The fit of a specific allocation scheme $s$ for a certain simulated ground truth $x(t)$ can be written as

$$\begin{aligned} &H(x(t), s) = \\ &\int P[\mathbf{Z}(t) = z(t)|\mathbf{X}(t) = x(t), s] \times P[\mathbf{Y}(t) = y(t)|\mathbf{Z}(t) = z(t)] \times h(x(t), y(t)) dz(t) dy(t) \end{aligned} \tag{27}$$

where $h$ is a functional that compares $x(t)$ and $y(t)$ and the integrals are functional integrals over all random sets $y(t)$ and $z(t)$. In IFD03, four different $h$-functionals are used: two which compute the entropy of $y$ at either a user-specified target-time or averaged over all time, and two which calculate the $L_2$ distance between $x$ and $y$, again either at a specific time or averaged over all times. The difference between the entropy-like measure and the distance measure is that the entropy measure rewards allocation schemes that give rise to peaked distributions, but might miss some of the enemy units. A measure that uses a specific time is termed a local measure, while global $h$-measures average over all times.

The overall best sensor allocation scheme is then determined by averaging also over the random set $\mathbf{X}(t)$, as

$$s^{\text{best}} = \arg\min_{s \in S} \int P[\mathbf{X}(t) = x(t)] H(x(t), s) dx(t) \tag{28}$$

Implementing Equations (27) and (28) would thus entail averaging over three different random sets, which is clearly computationally infeasible. There are several possible ways of approximating these equations. One way is to use approximations of the probabilities $P$ appearing in them, perhaps employing some kind of Monte Carlo sampling instead of the ensemble averages. In the implementation used in IFD03, we use a number of approximations:

1. As stated above, all motion of adversary units is constrained to a road network. Also, discretised time is used instead of continuous.

2. Instead of full random sets for simulated ground truth, observations, and simulated filter, PHD's are used for these. This means that, for instance, $x(t)$ only gives the expected number of units at different positions in the road network.

3. A very simple model is used for determining $P[\mathbf{X}(t) = x(t)]$ and averaging over all $x(t)$: it is assumed that the adversary's movement can be described by a motion model $\mathbf{T}$. This model is used to determine paths for all adversary units present at time $t = 0$. Instead of averaging over all possible futures, a certain number $N_f$ of such paths are generated and assumed to have equal probabilities of occurring.

```
% Pseudo code for the four major steps of the allocation algorithm.
% The module simulates $N_t$ ground truths and does
% $N_o$ realizations of the observation process for each
% ground truth, averaging the fitnesses.  This process is
% repeated for each sensor allocation scheme $s$, and the best $s$
% is selected.
% Note that $x_t$, $y_t$ and $z_t$ here are vectors;
% we have discretised space to only include nodes that
% are present on the road network.  $\mathbf{T}$ and $\delta$ below
% represent the motion model constrained to this network.
% $T$ is the end time of simulation, while $p_d$ is the
% assumed detection probability of a sensor.

% Simulating ground truth:
$x_1 = x_0$
for $t = 1 : T - 1$
    $x_{t+1} = x_t + \delta$ where $\delta$ is randomly selected
end for

% Generate fictitious observations:
for $t = 1 : T$
    if $s$ has sensor with view of $x_t$ at time $t$
       generate observation $z_t = x_t$ with probability $p_d$
    end if
end for

% Simulate filter:
$y_1 = x_1$
for $t = 1 : T - 1$
    $y_{t+1} = \mathbf{T} y_t + z_{t+1}$
    where $\mathbf{T}$ is a transfer matrix corresponding to the road network
end for

% Compare simulated ground truth and filter:
$h_1(s) = \|y_T - x_T\|_2$
$h_2(s) = \sum_t \|y_t - x_t\|_2$
$h_3(s) = H(y_T)$
$h_4(s) = \sum_t H(y_t)$
where $H(x)$ is the entropy of the vector $x$
```

Figure 10: Pseudo code for sensor allocation.

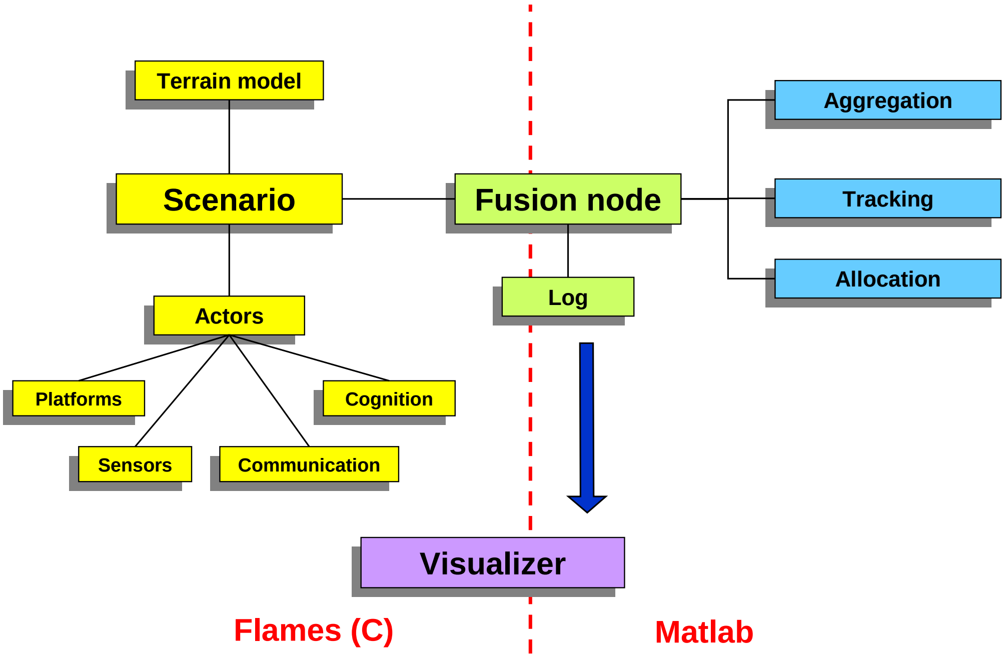


Figure 11: Connections between different parts of IFD03. Lines between modules mean that the modules exchange data. Note that the Visualizer is a separate program, while all the other modules are linked into the `Fire` program.

4. A similar motion model in the form of a transition matrix $\mathbf{T}$ is used to simulate the filter determining $\mathbf{Y}$, and we average only over a number $N_o$ of possible observations (*i. e.*, realizations of $\mathbf{Z}$).

The sensor allocation module returns the best found sensor allocation scheme $s$ as well as a quality measure that simply gives the fractions of the number of simulated ground truths and observations in which $s$ dominated all other allocation schemes.

# 5 System architecture

In this section, the overall design of the IFD03 system is described. The demonstrator utilizes all of the methods described in Section 4, and also makes use of an advanced terrain database (Section 5.5) that has been integrated into the simulation framework FLAMES (Section 6). An overview of how the different components fit together is shown in Figure 11.

The simulation in IFD03 describes the stochastic interaction between an observation system, a complex target system, in this case a hierarchically organized adversary unit, and a complex environment. Information is transmitted from simulated sensors to a simulated Command and Control, C2, site. At the C2 site information is fused and interpreted, using automatic information fusion processes. Some of these interpretations are then used by the C2 site as basis for issuing control messages intended to improve sensor utilization in relation to a predefined surveillance objective. Models and methods are not allowed to require operator interaction during the simulation. A key component of the demonstrator is the off-line visualizer, which provides a movie-like, interactively controlled multi-screen playback display of a set of parallel views of the prerecorded simulation.

IFD03 provides a standard FLAMES procedure for scenario definition, which can be used to combine the various object models in different ways to form specific scenarios.

## 5.1 Cognitive model of the fusion node

The fusion node has access to a priori information in the form of a terrain model and a doctrine and materiel database, generically describing the adversary's military organization. Also, it has the capability to perform dynamic remote control of a set of sensors which can observe portions of this force. On the highest level of abstraction the fusion node provides services for report clustering, aggregation, classification, and tracking of force units, and allocation and control of information collection resources, see Section 4.

When a sensor model has made an observation, it communicates this to the fusion node component of IFD03. Upon receiving an observation, the fusion node stores its data in an internal format. Currently, detailed analyses are not performed for each observation that is logged; instead different analysis modules are called at different pre-specified times. The three fusion modules implemented in IFD03 have somewhat different requirements on the data supplied to them. The particle filter implemented in the `Tracking` module needs to get observations fairly regularly, while the `Aggregation` module can be called at greater time intervals.

If the fusion node determines that a certain fusion module should be called, it collects the appropriate input data for the module and then takes care of its output. The output is logged to data files for later playback in the visualizer, see Section 5.4.1. For the `Tracking` (Section 4.2) module, the appropriate input is a list of observations that have occurred since the last time `Tracking` was called. Its output consists of histograms of the particles representing the PHD of the hostile units. Input to `Allocation` (Section 4.3) consists of a list of positions of adversary units, as well as a set of possible sensor schemes and the road network used in the random set simulation. The `Aggregation` module utilizes all observations collected so far, and attempts to build a hierarchical situation picture of the units. Its first step consists of calculating a conflict matrix (Section 4.1.1) and then clustering the observations (Section 4.1.2). The number of clusters to use is unknown, so an empirically derived procedure is used to determine this, see Section 4.1.3. Once the proper clusters are found, a classification procedure that compares each cluster to a pre-specified set of templates is performed, Section 4.1.4.

The output of the `Aggregation` module consists of a list of clusters and their classification in terms of templates. This output forms the input for the next level of aggregation: classified clusters of sensor observations are termed vehicles, and further analyzed to get platoons, clusters of which in turn give rise to companies. Each iteration of the `Aggregation` module thus produces lists of vehicles, platoons, and companies.

## 5.2 Modeling doctrine, organization, and equipment

IFD03 requires models describing the behavior and motion of adversary ground forces according to their doctrine, *i. e.*, the set of tactical rules that is expected to guide the behavior of the opponent's army. This includes telecommunication and transportation along a road network of mechanized forces in hostile territory. As source for military doctrines, unclassified Swedish Armed Forces publications were used. The application of these doctrines to our scenario was developed in dialog with military experts.

The adversary battalion model consists of approximately 60 vehicles: battle tanks, armed personnel carriers, antiaircraft missile launch vehicles, grenade launcher vehicles, and two types of truck. To create models for these target objects, a table of normalized detection, classification, and identification probabilities were needed for each object type and each type of sensor. In these tables, objects are assumed to be viewed at a fixed distance and against a clutter-free image background, noise-free seismic or acoustic environment, etc. Attenuating properties of the environment will reduce these probabilities as they occur in observing situations. Five battalion options were included:

1. mechanized infantry battalion
2. mechanized infantry battalion extended by a tank company
3. mechanized infantry battalion extended by a howitzer company
4. tank battalion with 3 companies
5. tank battalion with 5 companies

The descriptions include unit hierarchy down to vehicles of optional types. From these resource descriptions the application “march under low threat” was developed, which includes the sequence of and distance between vehicles and units, from vehicles via platoons to companies.

The information used in modelling the radio communication needed to stimulate COMINT interceptors describes the commanding hierarchy and simple communication rules. Simulated radio messages are exchanged within platoons in the scenario. This exchange follows two patterns: (1) commander gives an order, subordinates answer one after another, and (2) two-party dialogues. Orders are more frequent when platoons pass certain geographical regions in the scenario where it is likely that they must communicate because of commanders’ change of route *etc*.

### 5.2.1 Sensor modeling principles

How a sensor can be modelled depends strongly on its type. In general what is needed is some kind of detection or recognition time for each sensor, *e.g.*, for an image sensor, a shortest time during which an object must be continuously visible to be detected, classified, or identified, each step in this sequence requiring additional time. These times depend on sensor type, obstacles in the line of sight, and target object type, in combination with target attitude in relation to the sensor.

The resolution of an image-generating sensor is vital for the sensor’s ability to detect, classify, and identify a target. It depends on optics, zoom factor etc. Additionally, the contrast between light and dark parts of the image has to be strong enough (Klein, 1997). The object’s aspect angles in relation to the observing sensor are also of relevance. Finally, the surrounding environment generates clutter which reduces the sensor’s ability to distinguish objects. It is here not sufficient to build a sensor model at a certain level of detail. The subject is threefold:

1. A sensor detects energy (light, vibrations, radio signals, *etc*.),
2. The energy has been emitted from somewhere, as well as propagated and attenuated, reflected *etc*. on its way to the sensor.
3. For detection to take place, the detected signal-to-noise ratio (SNR) from a target must be high enough for a signature-extracting mechanism to find the features it is trained to discern.

The higher SNR, the more detailed estimates can be produced. The main issue is how the SNR should be realistically modelled for a chosen sensor in a typical terrain where a typical target is located.

The fusion methods tested in IFD03 expect input such as observation times, target positions and velocities with their uncertainty estimates, as well as target types with uncertainty within a given target classification hierarchy. Recognition type hierarchies relevant for the different sensors' energy classes were constructed. According to these, an image sensor can, *e. g.*, discern (with decreasing discernibility) <T80> / <tank> / <tracked vehicle with 6 track rollers> / <vehicle>, and a seismic sensor can discern <heavy tracked vehicle> / <vehicle>. Each entry in these hierarchies can be expanded into all vehicle types belonging to it, limited by the ontology of known vehicle types. This allows comparisons based on evidential reasoning to be made between type information from different sensor categories.

Sensors used to detect ground targets will likely show greater rates of false detection the more difficult prevailing surveillance conditions are, *i. e.*, the more hilly and diversified the terrain is, and the more complex sound and seismic wave transmission conditions are.

### 5.2.2 Image sensors

In the scenario used, there is a diversified terrain with many forested areas of different sizes, open croplands, roads, lakes, and littoral regions. For real observations from UAV's carrying video cameras working in the visible region, this type of terrain would give sparse bursts of reports when targets happen to pass by sufficiently transparent areas of the tree canopy, open areas, or are travelling on roads. Other routes through the terrain provide better camouflage. In future work, it is desirable to reproduce this behavior in greater detail, in order to study how the non-smooth flow of reports will affect the performance of fusion methods.

Image sensor detection probability is based on the empirical "Johnson criterion" (Johnson and Lawson, 1974). It gives a relation between the number of resolved pairs of light/dark bars in a bar pattern of the same size as the target minor extension projected towards the observer, and the probability of detection, classification, or identification. The number of resolved bars is related to the contrast between light and dark regions in the image, also interpretable as SNR. For a sensor in the visible region, this is the contrast of reflected light within and at the bounds of a potential target. This contrast is dependent on target surface reflectance variations, and the strength and direction of the ambient light. For an IR sensor in the thermal region, contrast depends instead on the target and background temperature variations. Outside the target itself, a contrast-rich (clutter-rich) background makes detection more difficult.

The attenuation of light is modelled for an image sensor observing from a UAV. The attenuation factor is dependent on terrain cover type (forest/open land) and, in the forest case, on the angle between the line-of-sight (LOS) and the vertical direction.

### 5.2.3 Ground sensor networks

The ground sensor network model was implemented in the simplest possible manner. An integrated tracker was assumed, to motivate the removal of terrain effects as well as the influence of the individual positions of the network nodes. This led to a statistically homogeneous detection quality inside the range of the network. The purpose of such a sensor is to contribute intelligence carrying high quality position and speed measurements, but poor classifications.

### 5.2.4 Human observers

The model of human observers (home guard patrols equipped with advanced measuring binoculars) is less detailed than the image sensor discussed above. This is mainly due to difficulties of modelling the complex fusion performed by the human brain. A basic relationship of detection quality proportional to distance was assumed and model parameters were then adjusted so as to produce reasonably realistic output.

### 5.2.5 COMINT interceptors

Radio messages can be intercepted by blue COMINT interceptors deployed in the terrain. The interceptors give rather coarse information about bearings to emitters. Bearing crossings are computed to get an indication of the position of an emitter. In order to resolve the platoon structure, information about position and communication pattern is transmitted to the fusion node, which tries to find out who is communicating with whom, see Section 4.1.2.

### 5.2.6 Sensor carriers (platforms)

Sensors will usually be carried by some kind of platform, ranging from aircraft or UAVs to APCs, soldiers and civilians. Sensor platforms can be characterized by their ability to elevate their sensors to different heights, as well as their speed, their ability to move to various positions after longer or shorter alerting time, *etc*. On the ground, stationary platforms may exist which are either completely immobile, or are able to move only after a certain redeployment time. Vehicle-bound sensor systems may also be present, whose carriers are either restricted to move on roads of some minimum quality, or are able to move in terrain of some minimum trafficability.

## 5.3 Scenario simulator

All data originate in the scenario simulator. The primary objects of the simulation fall into three categories: actors, terrain model and fusion node.

- Actors. The actors in a scenario simulation consist of red and blue units. Each unit is equipped with a platform model and a radio model for communication. Blue units are also equipped with various sensors for target detection and classification. Sensor types modeled are an autonomous video and IR camera, a “soldier” sensor (visual), a ground target multi-sensor system (acoustic/seismic) and a communications intelligence surveillance system. The video/IR camera can be attached to an unmanned aerial vehicle (UAV), which can also carry and drop a ground target multi-sensor system. To enable target detection and classification, visual and acoustic/seismic signatures are attached to all red platforms. When a detection is made, a report is sent to the fusion node with target information.

- Terrain. As detailed in Section 5.5, the terrain model is structured as a triangulated terrain skin with additional vector data, representinging the features of the environment, such as vegetation foliage, by polyhedra. The platforms and sensors in a scenario use this information for mobility and visibility calculations. The same information is available for the fusion node, which currently uses it only in the tracking algorithm. In future versions, the terrain model may also be used for path planning in the sensor allocation method.

- Fusion node. In a formal sense, the fusion node (Section 5.1) is also an actor in the scenario. It is implemented as a FLAMES cognitive model attached to a blue unit. The other blue units constantly feed the fusion node with target reports and upon request, sensor status reports. By sending sensor status reports a unit can update the fusion node on the current coverage of its sensors. This information is used by the tracking algorithm, see Section 4.2.3. The fusion node acts in the scenario by making sensor status requests on suitable units and by directing blue sensor resources.

The parts of IFD03 that handle scenario simulation are written in C and directly linked into the FLAMES suite of programs. All of our fusion modules are implemented in compiled MATLAB code, which is linked into the FLAMES program `Fire` to produce an executable.

## 5.4 Scenario display

During demonstrations, three adjacent projection screens show, respectively:

- reports and ground truth data displayed on a synthetic map background,

- results from the different information fusion methods displayed on map backgrounds, and

- dynamic plots of various statistics and other information about the current state of the fusion processes.

These views are intended to support a tactical intelligence staff in building a situation picture.

At the beginning of the scenario only a few reports have arrived. These are indicated on the first screen (Figure 12) and then appear as clustered objects on the second screen. This is the first chain of fusion events shown during the demonstration. At the same time the process can be followed on the third screen where plots of the number of received reports and the estimated number of objects are displayed (Figure 14).

As the scenario progresses, more surveillance resources are allocated and therefore many more reports are delivered. On the second screen, views showing clustered vehicles and clustered platoons are displayed. Here, vehicles and platoons are automatically classified into more or less specific categories, when possible into specific types. The categories or types are displayed using standardized army symbols (Figure 3). Comparing the two levels reveals a good correspondence. The operator can switch instantly between different aggregation levels such as reports, vehicles, platoons and companies, showing how the different information fusion methods work at different levels. The display can also be paused, to show how correspondences vary in different situations and between different levels of the scenario. By zooming in on any desired display area, detailed situations can be visualized and discussed.

To indicate to the audience how the fusion methods are performing, various results can be compared with ground truth as it is displayed during the demonstration. Having access to all scenario information, the ground truth view shows the location of all vehicles and all sensors in the displayed area over time. Some of the information fusion methods used in the scenario require a specific context to best show their capability. One example is the particle filter tracking which is more effective in terrain than on roads and should therefore be displayed during a terrain passage.

### 5.4.1 Visualizer

The IFD03 Visualizer is a substantially modified version of the original FLAMES visualizer `Flash`, aimed at illustrating the full functionality of the IFD03 fusion node. The simulation results can be visualized in multiple parallel visualizers, making it possible to use several computers and screens simultaneously. New views can easily be created and customized.

Two categories of views are used in IFD03:

- Analyst views. These views could help an analyst build a situation picture. An Analyst view is a window with a 2D projection of the terrain model, with zooming and panning possibilities. On top of the map is a layer of Open GL graphics and FLAMES icons, representing sensor reports, unit positions or particle filter histograms. For IFD03 the following views were designed and demonstrated: ground truth, sensor reports, vehicle, platoon and company aggregates and particle filter histograms on vehicle level. Snapshots of the ground truth and sensor report views are shown in Figure 12, the vehicle and platoon aggregate views in Figure 13, and the particle filter histogram view in Figure 9.

  Additional information for the analyst can be obtained by clicking on the symbols in a view. An information box then appears with details specific for each type of symbol. As an example, clicking on a platoon symbol gives information on the age of the aggregate, its classification and certainty, the number of vehicles in the cluster forming the platoon and the best alternative classification.

- Status views. These views display technical data from the analysis methods. Two Status views were designed for the IFD03, using basic MATLAB plotting tools. The first view

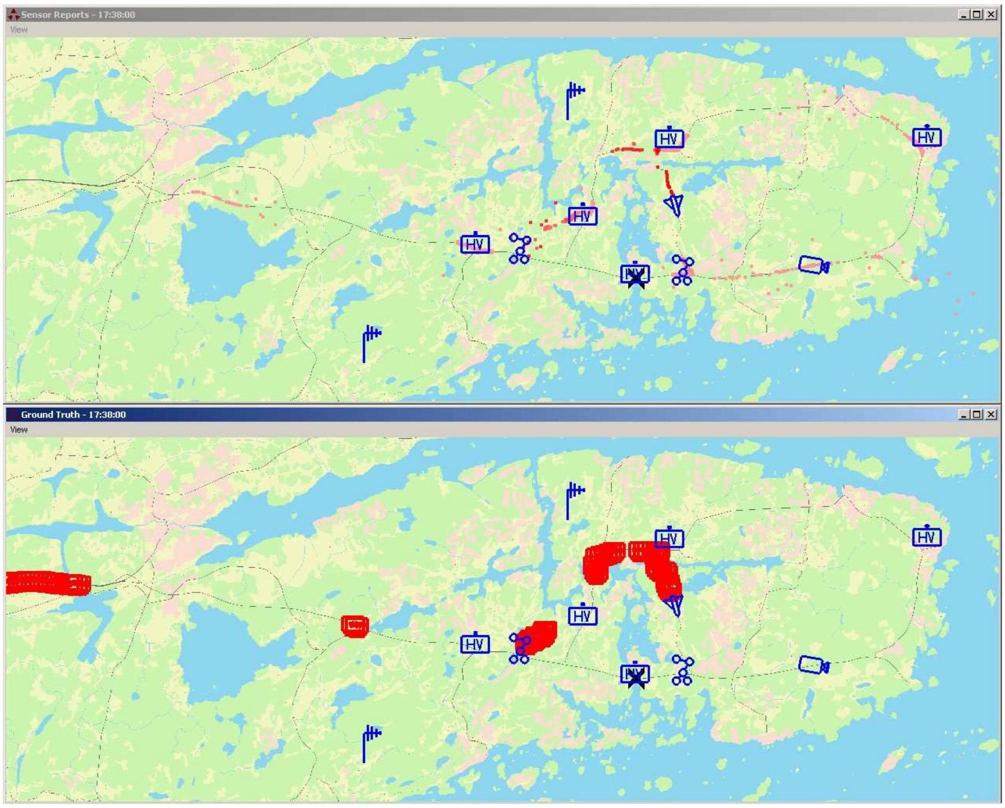


Figure 12: Snapshot of the sensor report (top) and ground truth (bottom) views in IFD03.

(Figure 14) shows the stream of incoming reports, the estimated number of vehicles, platoons and companies over time. In the case of estimated number of vehicles, two graphs are shown, representing the estimate given by the `Aggregation` module and the `Tracking` module respectively.

The second status view is used for demonstrating sensor allocation. The alternative paths available for the sensor platform are shown and the choice of the sensor allocation method is highlighted.

From a programming perspective the IFD03 Visualizer consists of four entities, three of which are executable applications. The database, handled by a MySQL database manager (*MySQL*, 2004), stores simulation result data to be visualized. The postprocessor application is responsible for creating tables in the database and for converting and transferring simulation result data into the database. The playback control application is responsible for synchronizing the playback of the scenario across the different connected visualizers. The user controls the playback of the scenario from this application, which works as a server to which the visualizers can connect as clients. The user can move freely in scenario time and the clients will be updated accordingly. The modified `Flash` application is responsible for the actual visualization of the data.

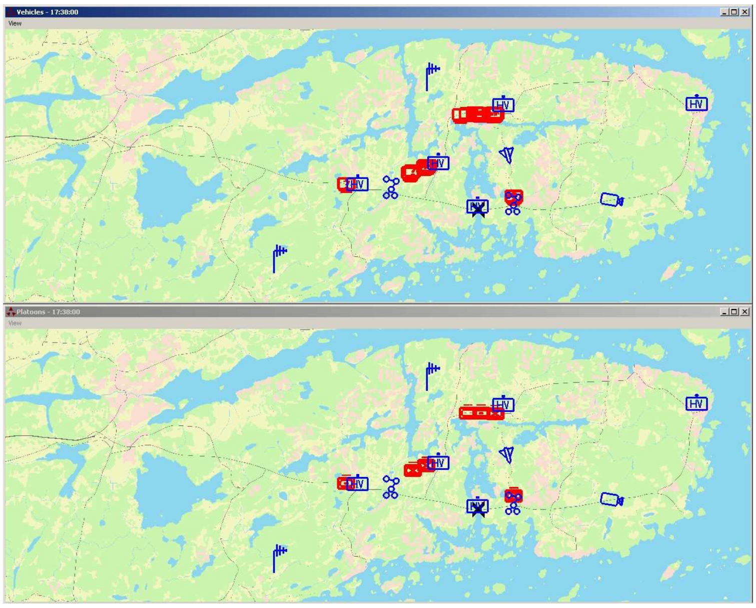
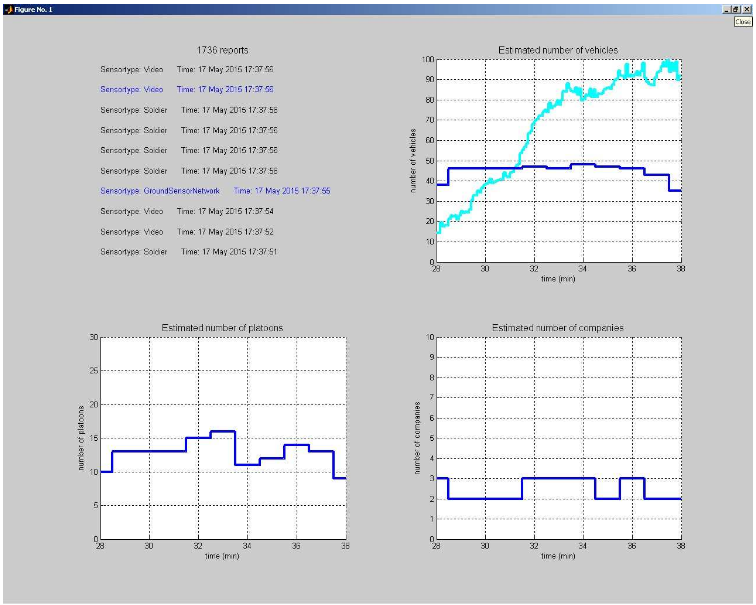


Figure 13: Snapshot of the vehicle and platoon aggregate views in IFD03.



```
1736 reports
Sensortype: Video      Time: 17 May 2015 17:37:56
Sensortype: Video      Time: 17 May 2015 17:37:56
Sensortype: Soldier    Time: 17 May 2015 17:37:56
Sensortype: Soldier    Time: 17 May 2015 17:37:56
Sensortype: Soldier    Time: 17 May 2015 17:37:56
Sensortype: Soldier    Time: 17 May 2015 17:37:56
Sensortype: GroundSensorNetwork    Time: 17 May 2015 17:37:55
Sensortype: Video      Time: 17 May 2015 17:37:54
Sensortype: Video      Time: 17 May 2015 17:37:52
Sensortype: Soldier    Time: 17 May 2015 17:37:51
```

Figure 14: Snapshot of the standard status view in IFD03. The sub-plots show incoming reports and the estimated number of vehicles, platoons and companies over time.

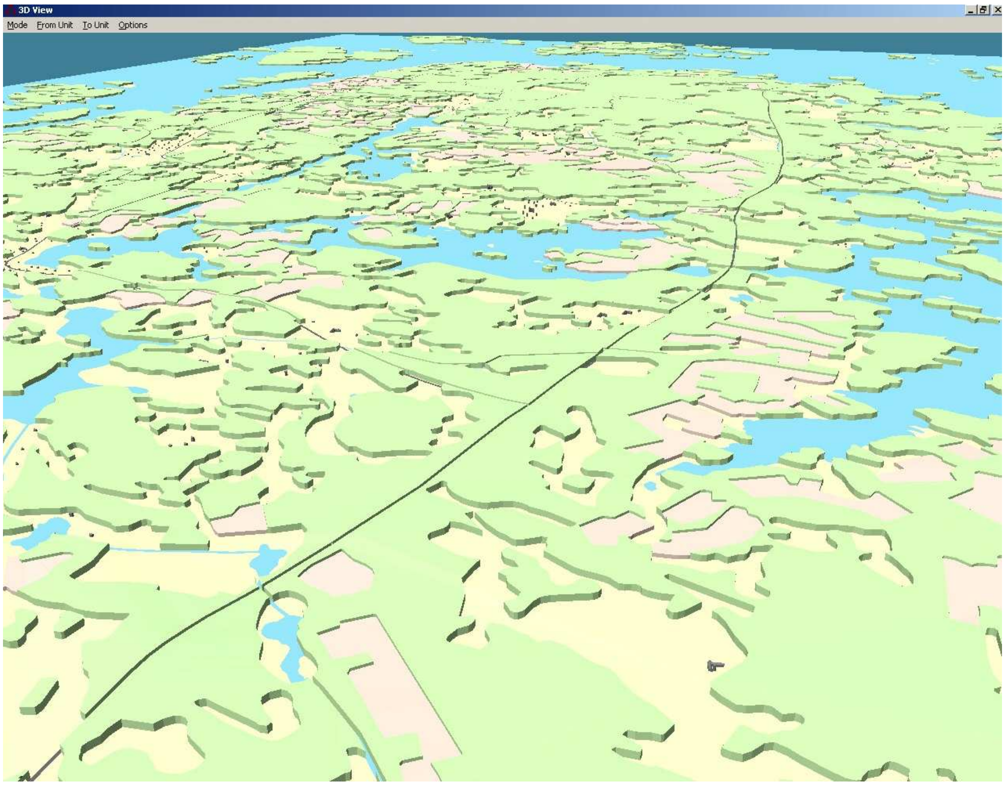


Figure 15: Visualizer perspective eastward, covering a part of the scenario area of interest. The terrain is color-coded according to (forest: green, open land: yellow, water: blue, road: grey, built-up areas: pink). Note the different heights of forest canopies and built-up areas.

## 5.5 Environment model

The terrain model for IFD03 is created by using a third party terrain database generation tool, TerraVista Pro Builder (*TerraVista*, 2004) developed by Terrex, that can import data from different sources and export a single correlated terrain description. TerraVista also has the ability to write the correlated data in a variety of formats. The terrain model is structured as a TIN DEM (Triangulated Irregular Network Digital Elevation Model) representing the terrain skin, and additional vector data describing terrain features, such as roads, rivers, lakes and houses. TerraVista is primarily a tool for creating terrain databases for visual simulation, however, it also has options for creating vector files or raster images for, *e. g.*, GIS applications. All output is correlated, *i. e.*, geometrically and topologically consistent.

### 5.5.1 Terrain data

The source data for this terrain model consisted of conventional off-the-shelf geographic data from the Swedish Land Survey. Source data used in the project come from GSD (*Geographical Data of Sweden*). For the scenario used in IFD03, data describe a 45 x 20 km$^2$ area including the

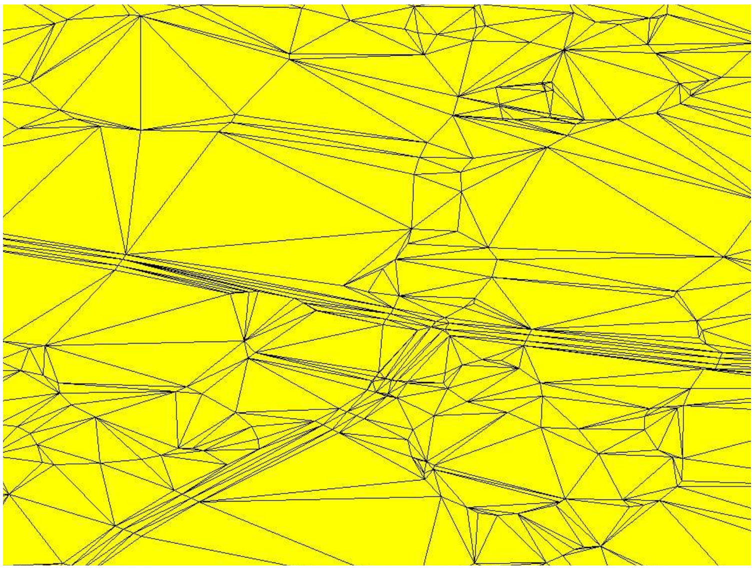


Figure 16: A small portion of the TIN DEM model. Note the road junction near the middle of the figure.

peninsula of Rådmansö, north-west of Stockholm. The terrain features were grouped into seven classes defined by FLAMES: bridge, building, canopy, land region, lake, river, and road.

The 50 x 50 $m^2$ ground elevation database from which our triangulated 3D model was built is probably detailed enough for our needs. However, for realistic modelling of the strongly varying tree canopy transparency, we would need, say, 25 x 25 $m^2$ raster information on tree population density and type, as well as typical tree height and mixing ratio of coniferous and deciduous trees. The CORINE Land Cover project (*CORINE Land Cover Project*, 2004) aims at obtaining higher specificity databases of land cover information for the EU countries. Work on this map product for the scenario area of interest, although ongoing, was not completed in time to be used in our demonstration.

# 6 Evolutionary development and its environment

In general terms, evolutionary system design and development (Coplien, 1999) may be described as a methodology where large development projects are partitioned into an organized collection of separately agreed subprojects or phases. Each phase is developed according to a predefined design contract, which can and must be operationally verified by a set of “users” representing the “customer” organization. Originally, the main rationale of evolutionary design and development is to facilitate close customer and end-user involvement in the development process. But a similar iterative design and development process is also well suited to the needs of a research group which develops comprehensive software while striving to retain much individual responsibility for design and work planning (Fredriksson et al., 2001).

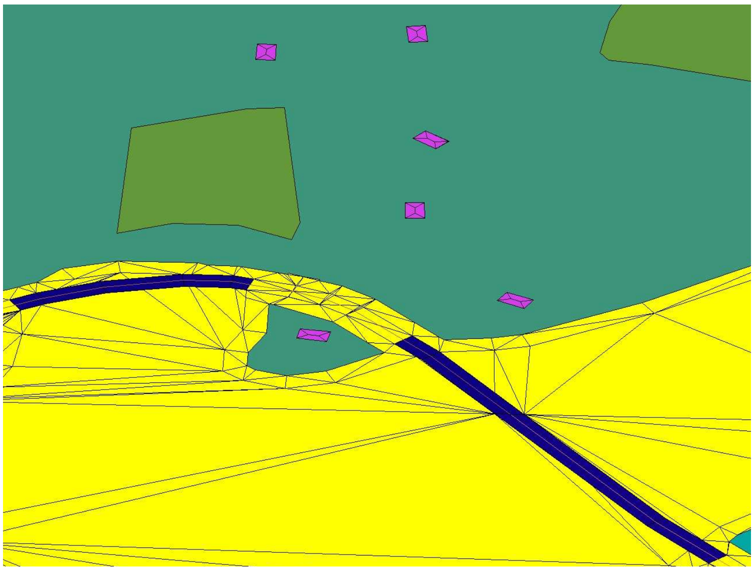


Figure 17: Terrain feature data after adding roofline data for six buildings (purple) and areal data for two bridges (dark blue).

To keep complexity, development time, and cost within acceptable limits, the development of simulations for information fusion research need to be based on extensive reuse of both software and scenarios. Models need to be adapted to the scale of command and control scenarios, typically involving hundreds or even thousands of actors in a mainly ground-based environment encompassing, say, between 10 and 1000 km$^2$. A development environment suitable for evolutionary development of information fusion test and demonstration systems should allow information fusion algorithms and ideas to be tested in a software plug-in type of open simulation development platform (Hörling et al., 2002).

## 6.1 Simulation development environment

A simulation framework is a product which provides a generic development environment, or toolset, for modelling and simulation. Over the past six years FOI has acquired a simulation development platform, based on the commercial simulation framework FLAMES (*FLAMES*, 2004), suitable for experimental evaluation, evolutionary development, and demonstration of many kinds of event-driven scenario-based simulations. FLAMES offers an infrastructure that includes common facilities for models, such as object, time, memory, and data base management, as well as execution control and more. It also provides a set of standard applications to support scenario definition and execution.

To adapt the FLAMES framework to the needs of information fusion research, advanced terrain modeling facilities were included (Hörling et al., 2002), allowing fully integrated topographical and thematical models of geography to be used in the simulations. In the IFD03 project, the resulting development environment was further extended by allowing simulation program modules to be developed using the problem-solving environment MATLAB (*MATLAB*, 2004). According to

FLAMES conventions, models should be written in C or C++. Therefore, our development process includes automatic translation of MATLAB modules into C using the MATLAB C Compiler.

The simulation framework approach has by now proved its ability to offer substantially increased productivity in a number of complex cases. This is due to faster development of scenarios and models, extensive reuse of scenarios, models, and program modules, as well as cost-sharing and cooperation with other projects working with related tasks.

Obviously, these advantages come at a price:

- license costs for the simulation framework can become substantial, and may need to cover development costs irrelevant to one's own organization,
- increased productivity can often be achieved only in the long run, and sometimes as higher-quality research results rather than as straightforward cost savings,
- in each usage category a simulation framework supports, there is a need for in-house know-how which needs to be accessible to several competence groups and projects.

## 6.2 Using MATLAB for simulation software development

The decision to use MATLAB instead of C or some other language (CommonLisp was a seriously advocated alternative) for developing the fusion node was taken because we wanted to spend as little time as possible developing and debugging the implementation of our algorithms, and focus our work instead on algorithm design. The fact that most of our group has significantly more experience in the use of MATLAB than of C influenced this choice.

Using MATLAB for software development has had both positive and negative consequences. On the pro side, new ideas may be quickly implemented using MATLAB's rich variety of built-in functions. MATLAB algorithms could often be conveniently debugged by loading input data, previously generated and then saved during execution of the compiled system, into an interactive MATLAB session. Also, test code could very easily be added, such as plotting the input or output of a function. All this contributed to a very significant, sometimes tenfold, improvement in development productivity.

On the con side, MATLAB does not provide fully automatic garbage collection. Instead, the MATLAB system handles allocation and deallocation of memory for objects by use of a heap mechanism. Memory space is automatically reallocated when an object grows. Initially, this caused severe memory fragmentation problems. To diagnose and fix such problems, MSWindows diagnostic functions had to be used to obtain information on memory availability. Ultimately, the cause of these problems may be found in a programming style not adjusted to the development requirements of large systems. MATLAB allows preallocation of matrices that will contain a large number of data. This is the style to be preferred when developing large systems in MATLAB (*MATLAB*, 2004). It is, however, not necessarily easy to apply this style consistently and our efforts to remove all memory fragmentation problems from the system have not yet been successful.

To illustrate the possibly large performance difference between compiled code written in MATLAB and C, respectively, consider the time-critical aggregation computation in IFD03 (Section 4.1).

The average measured computing time per completed reports-to-vehicle aggregation in the demonstration scenario was initially about 30 min. on a 2.6 GHz single-processor PC. A few hundred reports were clustered each time, usually yielding an optimum number of 10-30 clusters (vehicles). In this problem ten clustering trials were made for each of 21 different cluster sizes in each force aggregation. Through a systematic, profiler-guided code optimization effort involving, *i. a.*, hand-translation of the most time-consuming loops into C, total clustering execution speed was later improved by more than a factor of ten.

MATLAB, designed as an interactive environment, will not catch many errors when the MATLAB Compiler is used. Even simple things like misspelling a function or variable will cause run-time errors. However, MATLAB Compiler does issue compilation warnings for many errors like these. Thus, MATLAB Compiler messages should be closely watched.

The large size of the terrain database makes the simulator run close to the Windows upper limit of 2 GB per process memory size. Using a larger terrain database size would therefore not be possible under this operating system. Conceivable solutions of this problem include switching to a computer system with 64 bit address space, or changing the terrain database part of FLAMES to use a disk-stored database. In a short term perspective, both approaches seem unrealistic. We recently noticed, however, that applications running under the Windows XP and 2003 Server operating systems may use up to 3 GB memory per process, which promises to resolve our short term problem, although this has not yet been confirmed.

### 6.3 Code versioning and documentation

The CVS (Concurrent Version System) configuration manager (*CVS configuration manager*, 2004) played an essential part in our system development process. While the use of CVS requires considerable discipline from developers (*e. g.*, not committing untested code, writing proper change logs), we would probably not have been able to interface the different parts of the system without using it, or some similar system.

## 7 Summary and future work

The Swedish Defence Research Agency (FOI) has developed a demonstrator, called IFD03, for demonstrating information fusion methodology focused on intelligence processing at the division level. Using this demonstrator FOI is able to demonstrate possible information fusion methods for a future Network Based Defence (NBF) C4ISR system. The demonstrator also functions as an internal development tool for testing newly developed methods in an established environment together with previously developed methods.

The primary purpose of the project has been to investigate how information fusion methods can be combined into a system and work together in the context of that system. We also wanted to create and exercise an effective mechanism whereby information fusion concepts can be communicated to our customers and other interested parties. Finally, we wanted to create a basis for discussions with customers and prospective users about how research in the information fusion area should be prioritized. On the other hand, IFD03 is not a design and certainly not a prototype of a deployable system. To create a prototype, a significant additional R&D effort would be required.

Further development of the particle filtering tracking techniques used in IFD03 may eventually permit concurrent tracking of both solid objects (*e. g.*, vehicles) and group objects (*e. g.*, ground force units), logically connected via uncertain and vague information in the shape of doctrinal rules and communication capability. Work is also ongoing to develop new, more powerful methods in the general area of sensor resource management. Finally, studies are being made which may lead to some capability to automatically recognize tactical plans and intentions (Suzić, 2003; Johansson and Suzić, 2004).

Successively for various scenarios, in the future we also expect to create a capability to address effectiveness issues, such as:

- what improvement in effectiveness of various aspects of situation modeling can be expected from different information fusion methods?
- what improvement in effectiveness can be expected from a network-based defence technology, with and without information fusion methods?
- how do delays and “inertia” of various kinds, arising from, *e. g.*, information transmission or information processing, influence expected improvements in effectiveness?

The demonstration of IFD03 in December 2003 for the Swedish Armed Forces, as well as subsequent ones during the Spring of 2004, were very well received.

# References

Ahlberg, S., Hörling, P., Jöred, K., Mårtenson, C., Neider, G., Schubert, J., Sidenbladh, H., Svenson, P., Svensson, P., Undén, K. and Walter, J. (2004). The IFD03 information fusion demonstrator, *Proceedings of the Seventh International Conference on Information Fusion, Stockholm, Sweden*, International Society of Information Fusion, Mountain View, CA, USA, pp. 936–943.

Arulampalam, M. S., Maskell, S., Gordon, N. and Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking, *IEEE Trans. Signal Processing* **50**(2): 174–188.

Bengtsson, M. and Schubert, J. (2001). Dempster-Shafer clustering using Potts spin mean field theory, *Soft Computing* **5**(3): 215–228.

Biermann, J. (1998). HADES – A knowledge-based system for message interpretation and situation determination, *in* A. P. del Pobil, J. Mira and M. Ali (eds), *Tasks and methods in Applied Artificial Intelligence, Proceedings of the Eleventh International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, LNCS 1416, Springer-Verlag, Berlin, Germany, pp. 707–716.

Cantwell, J., Schubert, J. and Walter, J. (2001). Conflict-based force aggregation, *CD Proceedings of the Sixth International Command and Control Research and Technology Symposium, Track 7*, US Dept. of Defense CCRP, Washington, DC, USA, pp. 1–15.

Chaikin, P. M. and Lubensky, T. C. (1995). *Principles of condensed matter physics*, Cambridge University Press, Cambridge, UK.

Coplien, J. O. (1999). Reevaluating the architectural metaphor: Toward piecemeal growth, *IEEE Software* **16**(5): 40–44.

*CORINE Land Cover Project* (2004). www.gsf.de/UNEP/corine.html. Online.

*CVS configuration manager* (2004). www.cvshome.org. Online.

Dempster, A. P. (1968). A generalization of Bayesian inference, *Journal of the Royal Statistical Society B* **30**(2): 205–247.

*FLAMES* (2004). www.ternion.com/index.htm#products/need.htm. Online.

Fredriksson, J., Svensson, P. and Risch, T. (2001). Mediator-based evolutionary design and development of image meta-analysis environments, *Journal of Intelligent Information Systems* **17**(2-3): 301–322.

Goodman, I. R., Mahler, R. P. S. and Nguyen, H. T. (1997). *Mathematics of Data Fusion*, Kluwer Academic Publishers, Dordrecht, Netherlands.

Gordon, N., Salmond, D. and Smith, A. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation, *IEE Proceedings F (Radar and Signal Processing)* **140**(2): 107–113.

Hall, D. L. and Llinas, J. (eds) (2001). *Handbook of Multisensor Data Fusion*, CRC Press, Boca Raton, FL, USA.

Hall, D. L. and Steinberg, A. (2001). *Handbook of Multisensor Data Fusion*, CRC Press, Boca Raton, FL, USA, chapter 21: Dirty Secrets of Data Fusion, pp. 21:1–12.

Hörling, P., Mojtahed, V., Svensson, P. and Spearing, B. (2002). Adapting a commercial simulation framework to the needs of information fusion research, *Proceedings of the Fifth International Conference on Information Fusion, Annapolis, MD, USA*, International Society of Information Fusion, Mountain View, CA, USA, pp. 220–227.

Isard, M. and Blake, A. (1998). Condensation – conditional density propagation for visual tracking, *International Journal of Computer Vision* **29**(1): 5–28.

Jacobson, I. (1994). *Object-Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley.

Johansson, L. R. M. and Suzić, R. (2004). Bridging the gap between information need and information acquisition, *Proceedings of the Seventh International Conference on Information Fusion, Stockholm, Sweden*, International Society of Information Fusion, Mountain View, CA, USA, pp. 1202–1209.

Johansson, L. R. M., Xiong, N. and Christensen, H. I. (2003). A game theoretic model for allocation of mobile sensors, *Proceedings of the Sixth International Conference on Information Fusion, Cairns, Australia*, International Society of Information Fusion, Sunnyvale, CA, USA, pp. 583–590.

Johnson, J. K. and Chaney, R. D. (1999). Recursive composition inference for force aggregation, *Proceedings of the Second International Conference on Information Fusion, Sunnyvale, CA, USA*, International Society of Information Fusion, Mountain View, CA, USA, pp. 1187–1195.

Johnson, J. and Lawson, W. (1974). Performance modeling methods and problems, *Proceedings of the IRIS Imaging Systems Group*.

Klein, L. A. (1997). *Millimeter-wave and infrared multisensor design and signal processing*, Artech House.

Lorenz, F. P. and Biermann, J. (2002). Knowledge-based fusion of formets: discussion of an example, *Proceedings of the Fifth International Conference on Information Fusion, Annapolis, MD, USA*, International Society of Information Fusion, Mountain View, CA, USA, pp. 374–379.

Mahler, R. (2000). *An Introduction to Multisource-Multitarget Statistics and its Applications*, Lockheed Martin Technical Monograph.

Mahler, R. and Zajic, T. (2001). Multitarget filtering using a multitarget first-order moment statistic, SPIE Vol. 4380 *Signal Processing, Sensor Fusion and Target Recognition X*, SPIE, Bellingham, WA, USA, pp. 184–195.

*MATLAB* (2004). www.mathworks.com. Online.

*MySQL* (2004). www.mysql.com. Online.

Peterson, C. and Söderberg, B. (1989). A new method for mapping optimization problems onto neural networks, *International Journal of Neural Systems* **1**(1): 3–22.

Potts, R. B. (1952). Some generalized order-disorder transformations, *Proceedings of the Cambridge Philosophical Society* **48**: 106–109.

Schubert, J. (1993). On nonspecific evidence, *International Journal of Intelligent System* **8**(6): 711–725.

Schubert, J. (2003a). Clustering belief functions based on attracting and conflicting metalevel evidence, *in* B. Bouchon-Meunier, L. Foulloy and R. R. Yager (eds), *Intelligent Systems for Information Processing: From Representation to Applications*, Elsevier Science, Amsterdam, Netherlands, pp. 349–360.

Schubert, J. (2003b). Evidential force aggregation, *Proceedings of the Sixth International Conference on Information Fusion, Cairns, Australia*, International Society of Information Fusion, Sunnyvale, CA, USA, pp. 1223–1229.

Schubert, J. (2004). Clustering belief functions based on attracting and conflicting metalevel evidence using Potts spin mean field theory, *Information Fusion* **5**(4): 309–319.

Schubert, J., Mårtenson, C., Sidenbladh, H., Svenson, P. and Walter, J. (2004). Methods and system design of the IFD03 information fusion demonstrator, *CD Proceedings of the Ninth International Command and Control Research and Technology Symposium, Copenhagen, Denmark*, US Dept. of Defense CCRP, Washington, DC, USA, Track 7.2, Paper 061, pp. 1–29.

Shafer, G. (1976). *A Mathematical Theory of Evidence*, Princeton University Press, Princeton, NJ, USA.

Sidenbladh, H. (2003). Multi-target particle filtering for the probability hypothesis density, *Proceedings of the Sixth International Conference on Information Fusion, Cairns, Australia*, International Society of Information Fusion, Sunnyvale, CA, USA, pp. 800–806.

Steinberg, A. and Bowman, C. (2001). *Handbook of Multisensor Data Fusion*, CRC Press, Boca Raton, FL, USA, chapter 2: Revisions to the JDL data fusion model, pp. 2:1–19.

Suzić, R. (2003). Representation and recognition of uncertain enemy policies using statistical models, *Proceedings of the NATO/RTO Conference on Military Data and Information Fusion, Prague, Czech Republic*, in press.

Svensson, P. and Hörling, P. (2003). Building an information fusion demonstrator, *Proceedings of the Sixth International Conference on Information Fusion, Cairns, Australia*, International Society of Information Fusion, Sunnyvale, CA, USA, pp. 1316–1323.

*TerraVista* (2004). www.terrex.com. Online.

Walker, D. W., Rana, O. F., Li, M., Shields, M. S. and Huang, Y. (2000). The software architecture of a distributed problem-solving environment, *Concurrency: Practice and Experience* **12**(15): 1455–1480.

Xiong, N. and Svensson, P. (2002). Sensor management for information fusion — issues and approaches, *Information Fusion* **3**(2): 163–186.