Information Acquisition in Data Fusion Systems

Ronnie Johansson

TRITA-NA-0328

Licentiate Thesis

Royal Institute of Technology

Department of Numerical Analysis and Computer Science

Akademisk avhandling som med tillstånd av Kungl Tekniska Högskolan framlägges till offentlig granskning för avläggande av teknologie licentiatexamen torsdagen den 18 december 2003 kl. 14.00 i Kollegiesalen, Administrationsbyggnaden, Kungliga Tekniska Högskolan, Valhallavägen 79, Stockholm..

TRITA-NA-0328
ISSN 0348-2952
ISRN KTH/NA/R--03/28--SE
ISBN 91-7283-655-5
CVAP283

Universitetsservice US AB, Stockholm 2003

Abstract

By purposefully utilising sensors, for instance by a data fusion system, the state of some system-relevant environment might be adequately assessed to support decision-making. The ever increasing access to sensors offers great opportunities, but also incurs grave challenges. As a result of managing multiple sensors one can, e.g., expect to achieve a more comprehensive, resolved, certain and more frequently updated assessment of the environment than would be possible otherwise. Challenges include data association, treatment of conflicting information and strategies for sensor coordination.

We use the term information acquisition to denote the skill of a data fusion system to actively acquire information. The aim of this thesis is to instructively situate that skill in a general context, explore and classify related research, and highlight key issues and possible future work. It is our hope that this thesis will facilitate communication, understanding and future efforts for information acquisition.

The previously mentioned trend towards utilisation of large sets of sensors makes us especially interested in large-scale information acquisition, i.e., acquisition using many and possibly spatially distributed and heterogeneous sensors.

Information acquisition is a general concept that emerges in many different fields of research. In this thesis, we survey literature from, e.g., agent theory, robotics and sensor management. We, furthermore, suggest a taxonomy of the literature that highlights relevant aspects of information acquisition.

We describe a function, perception management (akin to sensor management), which realizes information acquisition in the data fusion process and pertinent properties of its external stimuli, sensing resources, and system environment.

An example of perception management is also presented. The task is that of managing a set of mobile sensors that jointly track some mobile targets. The game theoretic algorithm suggested for distributing the targets among the sensors prove to be more robust to sensor failure than a measurement accuracy optimal reference algorithm.

Keywords: information acquisition, sensor management, resource management, information fusion, data fusion, perception management, game theory, target tracking

TRITA-NA-0328 • ISSN 0348-2952 • ISRN KTH/NA/R--03/28--SE • ISBN 91-7283-655-5 • CVAP283

Sammanfattning

Genom att målmedvetet använda sensorer, exempelvis i ett datafusionssystem, kan tillståndet av en systemrelevant omgivning uppskattas med syfte att stödja beslutsfattande. Den ökande tillgången till sensorer innebär många fördelar, men även stora utmaningar. Genom att använda många sensorer kan man förvänta sig att åstadkomma en mer heltäckande, högupplöst, säker och oftare uppdaterad uppskattning av omgivningen än vad som annars skulle vara möjlig. Utmaningarna innefattar bland annat dataassociering, hantering av motstridig information och strategier för att samordna sensorresurser.

Vi låter termen informationsinhämtning (information acquisition) beteckna den färdighet hos datafusionssystemet som aktivt inhämtar information. Syftet med denna avhandling är att beskriva och förklara den färdigheten i ett generellt sammanhang, att utforska och klassificera relevant litteratur, och att framhäva viktiga egenskaper samt troliga framtida forskningsinriktningar. Det är vår förhoppning att den här avhandlingen kommer att främja förståelsen av, kommunikation om, och vidare arbete inom informationsinhämtning.

Med tanke på nämnda tendens att tillgängligheten till sensorer ökar gör att vi är särskilt intresserade av storskalig (large-scale) informationsinhämtning, det vill säga informationsinhämtning medelst många (och eventuellt utspridda och heterogena) sensorer.

Informationsinhämtning är en generell färdighet som uppträder inom många olika forskningsområden, och i den här avhandlingen återger vi därför resultat från till exempel agentteori, robotik, och sensorhantering (sensor management). Vi föreslår även en taxonomi av tidigare arbeten som framhäver relevanta aspekter av informationsinhämtning.

Vi beskriver en funktion, perceptionshantering (perception management; besläktad med sensorhantering), som verkställer informationsinhämtningen i datafusionsprocessen och relevanta egenskaper av omgivningsens påverkan, sensorresurserna, och systemomgivningen. Ett exempel på perceptionshantering presenteras också. Uppgiften är att hantera en mängd rörliga sensorer som tillsammans skall följa några rörliga mål. Den spelteoretiska algoritmen som vi presenterar för att fördela målen mellan sensorerna visar sig vara mer robust mot bortfall av sensorer än en jämförelsealgoritm som eftersträvar att nå bästa möjliga förväntade mätningsnoggrannhet.

Sökord: information acquisition, sensor management, resource management, perception management, information fusion, data fusion, game theory, target tracking

TRITA-NA-0328 • ISSN 0348-2952 • ISRN KTH/NA/R--03/28--SE • ISBN 91-7283-655-5 • CVAP283

Acknowledgments

First of all, I want to thank my scientific adviser Henrik Christensen for his patience and relentless enthusiasm. Per Svensson and his group at the Swedish Defence Research Agency (FOI) have also been very supportive and inspiring. I would like to send a special thank to Ning Xiong, who was my closest colleague during my first two years as a PhD student, for enduring lengthy discussions. I would also like to thank my fellow co-workers at the computational vision and active perception laboratory (CVAP/NADA) for providing the atmosphere around the office. Also the NADA decision support group (Robert Suzić, Joel Brynielsson and Klas Wallenius being its most active members) must not be forgotten for providing knowledgeable comments from slightly different perspectives.

Preface

The work that resulted in this thesis was financially supported by the Division of Command and Control of the Swedish Defence Research Agency (FOI).

Most of this thesis is the sole work of the author. However, the discernible parts that were the result of cooperative work should be presented. In Chapter 2, the concept of perception management was invented in conjunction with Ning Xiong and first presented in a journal article (Information Fusion, number 3, 2003).

Furthermore, Chapter 4 is based on work by Ning Xiong and Henrik I. Christensen and developed in consultation with them. The chapter was previously published in a technical report (NADA report TRITA-NA-P03/06), and a short version was presented at the 2003 conference on information fusion.

Contents

Chapter 1

Summary and future directions

1.1 Summary

This thesis consists of a sequence of fairly independent chapters that all deal with *information acquisition* in *data fusion systems*. Such a system produces refined information and hypotheses about some observed environment by combining information from different sources. The products of a data fusion process are typically used by some other system for decision-making.

The focus on data fusion systems poses no threat to the generality of the discussion in this thesis. The reason is that data fusion is a comprehensive process that exploits relationships between multiple pieces of information which makes it a natural, perhaps inherent, aspect of any processing of information. Data fusion is further discussed in Chapter 2.

We loosely define information acquisition (IA) for data fusion systems to mean

> *the skill of a data fusion process that allows it to reason about and facilitate acquisition of information about a system-relevant environment.*

The purpose of IA in data fusion systems is to make the data fusion process more efficient. Its task may be to obtain or maintain an adequate description of the environment or both.[Hag90]

Two necessary criteria that justify IA for data fusion systems are that sensing resources are manageable and usage of those resources imposes conflicts that must be solved.[1] For instance, a conflict might arise between sensing tasks when there are more tasks than sensors to satisfy the tasks. Another type of conflict emerges, e.g., when radiated energy must be managed while preserving sufficient target tracking quality.

[1] This is a generalisation of the criteria suggested in [BP99a].

More specifically, with this thesis, we want to situate the process of acquiring information in a general context, explore and classify previous efforts relevant to the subject and highlight key issues and possible future work.

Most efforts related to this subject today concern *small-scale* systems with very specific goals. Furthermore, the general context of such systems, including, e.g., related processes and the stimuli that affects it, is seldom discussed. By small-scale, we mean systems that produce very limited information about some system-relevant environment. A large part of this thesis, on the other hand, is devoted to exploring and paving the way for large-scale systems. Such systems, typically, involve utilising a large number of possibly distributed and heterogeneous sensors to provide diverse information to fastidious decision-makers.

Our point of departure is from the *command and control* influenced world of *information fusion*. Command and control, briefly, refers to the military concept of managing available resources for perceiving the state of a battle field and to plan and take appropriate actions (see, for instance, [Cre85]). Due to its historical connection to the command and control field, the field of information fusion acknowledges a versatile and complex problem domain and environment. Furthermore, since the actors in the environment frequently are hostile and proactive, it is essential not only to estimate their "visible" properties but also their intentions. Hence, given the properties of this field, and the fact that its applications naturally encapsulates the fields of, e.g., robotics and computer vision (where active sensing and IA also have been studied extensively), it should be suitable as a starting point for the journey into the realm of large-scale IA.

A well known model in information fusion is the JDL-model. It is a functional model that describes aspects of a data fusion process. One of the aspects, *process refinement*, involves IA. Process refinement is a meta-function that monitors the data fusion process and tries to improve it. This activity has both an internal and external aspect. The internal aspect involves, e.g., selection of fusion algorithms and sifting through acquired data. The external aspect, concerns management of sensing resources for acquisition of information. In this thesis, information acquisition only refers to the external aspect.[2]

In Chapter 2, we use the JDL-model to create a general agent model that encapsulates both IA and information fusion and refinement to support agent decision-making. Such a model of the context of IA is vital for understanding the potential and limitations of IA.

In the explanation of the model in Chapter 2, we discuss, e.g., the observed environment, the resources used for sensing, and the stimuli that affects the decisions of the IA. It is important to classify the observed environment according to the properties mentioned in Chapter 2. For instance, if the environment is *dynamic*,

[2]Even though there is a conceptual difference between the internal and external aspects of process refinement, in practise, methods for the two types might coincide. Compare, for instance, the internal activity of selecting between various sets of collected data with the external activity of selecting which sensing resource to activate.

a process dealing with IA (in this thesis, we call such a process a *perception manager*) is constrained by the time it can devote to deciding its actions. Instead if the environment is *non-episodic*, the perception manager (PM) has to consider the future consequences of its actions.

Properties of information acquired from the environment are also presented. In Chapter 3, the discussion about environment and information properties is extended with a model of the relationship between the environment itself and the observing agent's description of it. We call the agent's description of the environment its *environment representation* (ER). The structure of the ER is prescribed by the agent's *environment model* (EM). Both may vary over time.

Properties of the resources used for IA, as well as the dependencies between resources, must also be taken into account by a PM. We suggest two useful property categories: *scope* and *value*. Scope properties of a resource affect its sensing capabilities and value properties the utility of using the resource. The stimuli includes, e.g., explicit information requests from users of the system and status reports from sensing devices (e.g., operational condition). We also discuss the advantages of expressing the action space of IA in terms of *services* rather than resources.

Types of *control* and *composition properties* of sensing resources are also presented. Composition properties distinguish whether a set of sensors is *homogeneous* or *heterogeneous*. Control architectures, both for *systems* and *individuals*, are furthermore mentioned. System control architectures are typically *centralised*, *decentralised* or *hierarchical*. Using the classification in [Ark98], we find the individual control architecture somewhere in between reactive and deliberative.

The idea in Chapter 2 is not to prescribe what a data fusion or IA process must include, rather to suggest their potentials, and to provide the reader with a comprehensive context to interpret the literature of the survey in Chapter 3.

As mentioned, this thesis also tentatively suggests the concept of *perception management*.[JX03] PM tries to grasp the complete domain of large-scale IA. As a theoretic concept, it is effectively a superset of *sensor management* (i.e., the management of sensors to improve their performance) and a subset of resource management (which also involves management of resources for other purposes). In practise, however, typical work in PM will probably act as an initiator and controller of various sensor management processes.

Chapter 3 departs from a high-level perspective of IA for data fusion systems, and surveys and classifies previous literature. The classification used here focuses on general aspects of IA and types of perception activities PM performs. The classification of activities is based on their influence on the ER.

We suggest a taxonomy of activities for perception with three type: *incorporation*, *monitoring*, and *discerning*. Incorporation involves integrating information that fits the EM into the ER. In the literature, target detection methods incorporate information (e.g., [CGH96]). Monitoring concerns maintaining the parts of the ER which have already been incorporated. In target tracking applications (e.g., [BP99a]), monitoring is performed. Discerning is an activity that aims at refining information in the ER. A typical example is view planning in computer vision, where camera locations are calculated to find properties of detected objects (e.g., for object recognition).[Roy00]

We envision that perception activities of these kinds operate on different levels of abstraction, possibly in a hierarchy to contribute to the maintenance of the ER and success of the system that uses it. Some activities might operate on a low level very close to the physical sensing devices with a short time horizon obtaining signals of objects and events while others select complex sensing services to obtain high-level information while generally enjoying longer time to consider its actions.

Comprehensive perception activities might exhibit one or two more qualities that we denote *facilitation* and *focus of attention*. The qualities are subtle in some applications and more explicit in others.

Facilitation refers to activities that do not directly affect the ER, but that supports observations, e.g., relocation of mobile sensors, managing energy and other system requirements.

Focus of attention concerns activities that handle conflicts between objectives and decides which information tasks to prioritize. Rather than updating the ER with correct and current information, focus of attention decides the structure of the ER, i.e., alters the EM.

The final part of Chapter 3 deals with focus of attention, which enables large-scale information acquisition. The literature surveyed in that part deals with, e.g., prioritization and task allocation.

Whereas the preceding chapters aim at explaining the domain of IA and control of sensing resources, Chapter 4 provides a concrete example of IA. The problem studied in Chapter 4 is that of managing a set of mobile sensors that jointly track a set of mobile targets. The mobile sensors use a negotiation algorithm to decide where to go and which targets to track. Sensors may make position estimates of more than one target at a time and estimates of the same target by different sensors are fused for better position estimate. Using the concepts and notions from Chapter 2 and 3, we can characterise the problem.

First of all, the environment we study (i.e., an approximation of the real physical world) is inaccessible (the information acquired by the sensors is imprecise), deterministic (we assume sensing actions always yield measurements), episodic (targets are assumed to disregard the sensing actions), static (the targets are assumed not to move far during the short time the sensors require to allocate targets), and semi-continuous (percepts are continuous but the action space is discretized).

The stimuli that affect the target tracking system is limited to a static mission control that requires that all known targets are tracked all the time. Although the sensing resources used in Chapter 4 are described on an abstract level, we can still say a few things about their properties. In terms of scope properties (Chapter 2), we can say that the sensing resources are always available and redeployable. In addition, we find that the set of resources is homogeneous, i.e., sensors have equal capabilities and, hence, can therefore easily cooperate by fusing position estimates.

Although the future aim of the model in Chapter 4 is to express decentralised system control, the model presented here is in practise centralised since perfect information is assumed for the negotiation algorithm. On the individual level, the control of sensors must be considered to be more deliberative than reactive, since the ER of all sensors must contain a representation of the other sensors and the targets. Furthermore, the allocation of targets by all sensors, in every sampling interval, is the compromise that results after a negotiation round.

Even though this application is small-scale in the information it acquires, the intention for the future is to bring its principles to more large-scale applications.

The contents of Chapter 4 were previously published in a technical report,[JXC03a] and a short version was presented at the 2003 conference on information fusion.[JXC03b]

1.2 Contribution

The main contribution of this thesis is its comprehensive conceptual model of information acquisition in data fusion systems. The level of comprehensiveness of the information acquisition model presented here has previously seldom been addressed in the field of data and information fusion. In Chapter 2, the concept of "perception management" is presented to encompass the process of data fusion systems that is responsible for information acquisition. Furthermore, the context of perception management (including external stimuli, sensing resources and environment properties) is also discussed. The main contribution of Chapter 3 is the suggested taxonomy of perception activities, that should be useful when designing large-scale information acquisition systems. We believe that the proposed model will facilitate communication, understanding and future efforts for information acquisition.

Finally, Chapter 4 suggests that game theoretic concepts may be useful for coordinating distributed sensors. A game theory based algorithm for allocating targets to mobile sensors is introduced. In a comparison, the game theory based algorithm proved to be more robust to sensor failure than an algorithm that aims at selecting the allocation that yields the best measurement accuracy. We intend to further explore the benefits of game theoretic concepts for information acquisition in future work.

1.3 Overview

To briefly reiterate the contents of the following chapters; Chapter 2 situates information acquisition (under the name perception management) in the context of data fusion in an agent model. The chapter discusses many aspects of this context such as environment, information, stimuli, and sensing resources. Chapter 3 surveys literature for information acquisition and classifies them according to how they affect the environment representation that is assumed to store acquired information. Finally, Chapter 4 gives an application of information acquisition in a target tracking problem.

1.4 Future research

Large-scale information acquisition involving the management of heterogeneous sensing resources to support decision-making in some complex and dynamic environment is largely underdeveloped.

While some of the fairly “low-level” aspects of information acquisition (IA) have been studied extensively (such as, e.g., target tracking), the comprehensive perspective of multi-sensor multi-task based IA has received little attention.

Some of the opportunities for future work which we discern are the following:

Integration of techniques Integration of available techniques for IA. For instance, connecting detection to tracking. Applications have already been developed, but the potential for future efforts is still huge. On a larger scale, it would be interesting to see more algorithms for focus of attention (e.g., task decomposition) blend with algorithms for perception. Large-scale IA systems also have to deal with many concurrent information requests. To satisfy these requirements, dependencies between perception activities should be explored, as well as dependencies between information requests. Additionally, algorithms for planning and scheduling such activities should be devised.

Long-term view Most efforts possess a greedy approach to information acquisition, selecting actions depending on the current situation. This is often a sound approach, since the uncertainty of the future states of the observed environment is generally considerable and predictions of future states are costly (especially in real-time applications). However, long-term planning is an inherent property of many large-scale information acquisition problems, since the deployment time of sensors might be considerable. Predictions, using models of the environment, and simulations could be used to generate hypotheses about the future environment state.

Pragmatic techniques Many techniques are still immature in the sense that they do not address the worst case environment (or even realistic environments) as characterised in Chapter 2. The observed environment is seldom static and in many applications it is even hostile and able to respond intelligently to the actions of the observing (data fusion) system. Methods should be refined to deal with more (eventually all) of the challenges presented by a “difficult” environment.

Re-planning For “massive” information acquisition (dealing with a profusion of observations), where acquisition priorities, sensing resource availability and environment might evolve non-deterministically while the system is considering its next actions, traditional (“off-line”) planning might produce obsolete plans. Future work directed towards large-scale IA will, hence, have to take the dynamic world into account during deliberation and execution of plans. This issue is discussed in the field of *continual planning* (surveyed in [dDJW99]). Furthermore, for the same reasons, the permissible running time of employed algorithms might not be known in advance. A class of algorithms, called *anytime algorithms*, address this issue and might be interrupted at any time and produce a solution.[Zil96] Typically, the quality of the solution increases the longer the algorithm is allowed to run. A related concept is *anytime actions*,[RW91] i.e., actions whose quality increases the longer the action is allowed to be performed. Clearly, large-scale IA will benefit from anytime algorithms and actions.

Chapter 2

A model of perception management in the data fusion process

2.1 Introduction

The ability of biological beings, such as mammals, to efficiently combine data (stimuli such as vision, scent, touch) from various, disparate sources of information (e.g., eyes, nose, fingers), supported by prior knowledge (e.g., instinct and experience), to interpret their environment is a strong incentive for the emerging *data fusion* research. The general aim of the research in data fusion is to bring this ability into use in autonomous or semi-autonomous artificial systems for enhanced performance.

A *data fusion process* is characterised by its ability to combine, possibly uncertain, incomplete, and contradictory, data. The result is data or information of "better quality", in some sense. As a consequence of the fusion (or merger), the resulting information is often abstract, generalised or summarised, and, hence, the amount of data is reduced.

It should be stressed that an implemented data fusion process does not exist as an isolated system, rather as an integral part of some enclosing system. The purpose of the data fusion process is then, typically, to improve the decision-making of the enclosing system. Whereas the enclosing system supplies the intention and result, the data fusion process improves the system performance. Due to its broad applicability, applications using data fusion have arisen in various, disparate fields,[Hal92a] for instance, military (e.g., avionics,[MM94, Adr93] and command and control [GR98]), remote sensing (e.g., localisation of mineral sources,[MGV+02] and identification of weather patterns), industrial (e.g., control and monitoring of complex machinery and assembly robots). Furthermore, due to the generic nature of data fusion, applications have also been suggested in, for instance, financial analysis.[Low98]

Typically, systems, which have to rely on continual, real-time percepts of a partially unknown, possibly interactive and intelligent, environment, will benefit from data fusion. The environment is said to be interactive and intelligent if it responds intelligently to the actions of the system. In the most difficult case, a malevolent agent[1] is responsible for the response.

Currently, much research in the field of data fusion has been done, but due to its immaturity and the different angles of approach of previous research (e.g., command and control, avionics, or mobile robots), there is a need for a unified comprehensive framework, and a generic taxonomy and terminology has yet to be developed. A step in that direction is the recent study [AAB+01] where different types of fusion processes are classified and fusion techniques for various applications are described.

The main focus of this chapter is put on the optional function of the data fusion process which manages the acquisition of information from the environment to the data fusion process. In the literature, this management of information acquisition (IA) is frequently entitled *sensor management*. However, in this thesis, a wider term, *perception management* (which we introduced in [JX03]), is used instead to emphasise the general standpoint of the thesis and to promote the term *perception resource* (instead of the more restricted term *sensor*) explained in Section 2.4.2. Using a “fresh” term also allows us to reason more freely about the properties of IA in data fusion processes, without the constraints imposed by a worn term such as sensor management. For many readers, however, the distinction we make in this thesis between perception management and sensor management is probably insignificant.

Throughout this thesis, we will use the terms *sensor*, *perception resource*, *sensing resource* and *information source* to denote a source of data or information. The subtle differences between the terms, where such exist, will be made explicit later on in the text.

The development of a perception management process is motivated by the need for, e.g., efficient use of limited resources, automatic resource reconfiguration and degradation in the occurrence of sensor failure, optimised resource usage, and reducing the workload of manual sensor management.[NN00]

The purpose of this chapter is to situate the data fusion process and, in particular, its inherent perception management facet in an application independent context. We believe that our model of perception management, including its properties and context in data fusion systems introduced in this chapter, will constitute a communication aid for reasoning about IA and in the development of applications.

Section 2.2 portrays the data fusion process using the JDL-model, known from the field of information fusion. The control aspect of the JDL-model is process refinement, which is briefly presented in Section 2.3. Perception management, being a subset of the process refinement function, is further discussed in Section 2.4. The input to perception management in terms of stimuli from its environment is identi-

[1] In this chapter, agent is simply defined as something that perceives and acts.[RN95] Hence, it could be either a human or an automated process.

fied and properties of its sensing resources are discussed. General control concepts are also suggested for perception management. Section 2.5 suggests complementary aspects of IA that should be studied in the future, and Section 2.6 offers a brief summary of the chapter contents.

2.2 The data fusion process

A definition of data fusion is provided in [SD98]:

> *"Data fusion is a process that combines data and knowledge from different sources with the aim of maximising the useful information content, for improved reliability or discriminant capability, while minimising the quantity of data ultimately retained."*

Another definition is provided by the Joint Directors of Laboratories (JDL) Data Fusion Subpanel (DFS) which, in its latest revision of its data fusion model, [SB01] settle with the following short definition: "Data fusion is the process of combining data or information to estimate or predict entity states." Due to its generality, the definition of JDL encompasses the one of [SD98].

Here, we, unlike in the definitions above, prefer to use the complete term "data fusion process," instead of just "data fusion." The reason is to separate the general, complex, and versatile (data fusion) process from application specific, and justifiably restricted (data fusion) methods.

One aspect of the data fusion process (`DFP`), which is not included in the first definition and implicit in the second, is *process refinement*, the function of improving the `DFP` and data acquisition. Many authors, as well as we, recognise process refinement and data fusion to be so closely coupled that process refinement should be considered to be a part of the `DFP`.

As implied in the previous section, the `DFP` is not a new technique in itself, rather a framework for incorporating reasoning and learning with perceived information into systems, utilising both traditional and new areas of research. These areas include decision theory, management of uncertainty, digital signal processing, and computer science.[WL90a] The `DFP` comprises techniques for data reduction, data association, resource management, and fusion of uncertain, incomplete, and contradictory information.

As mentioned in Section 2.1, data fusion is successfully utilised by biological systems, among which the human being is one. Some reasons for automating it in artificial systems are:

Replacing manual fusion In some existent systems, fusion processes are performed manually by humans. This might become infeasible if the flow of data to the system exceeds the capabilities of the human resources. In comparison with fusion of data performed by manual labour, (automated) data fusion may be less costly, more reliable and predictable (human beings make mistakes), and, of course, faster. Automated data fusion may also be customised and optimised for a specific task.

Improving performance in automated systems Data fusion may be used to improve the quality of acquired information in an automated system. Here, improved quality may refer to, e.g., data of higher certainty, relevance, precision and resolution in relation to the system objectives. The need for quality improvement arises from the fact that many (not clearly distinguishable) objects may be of interest, conflicting percepts (perhaps due to deception or sensor error), incomplete information, and other ambiguities about the environment and behaviours (mentioned in [PCCB97]). Additionally, entirely new types of information (i.e., properties that are not directly measurable by accessible sensors) may be inferred by combinations of data from disparate sources.

2.2.1 The context of the data fusion process

Because of the nature of the DFP as a support for other systems, it is useful to observe it in a broader context. Figure 2.1(a) shows a coarse sketch of a generic system which performance depends on its interactions with some environment.[2] Note that this model does not make the claim that this system should be implemented on a single physical platform. It is general enough to be implemented in a distributed fashion. This model is basically the common agent model with its perception-action cycle (e.g., in [Woo99, Nil98, RN95]).

The *system control* is responsible for the objectives and result of the system operation. An objective may be either external or internal. *External objectives* concern the goals of the system from a user-perspective, whereas the *internal objectives* concern maintenance and operational goals, and system constraints. The objectives influence the DFP and do not have to be static. On the contrary, it is more likely that the objectives change over time with varying preferences of the system control and new data entering the system. The system control may also be assumed to be able to manage all controllable degrees of freedom of the system (i.e., all controllable resources). It uses its resources mainly to act and perceive. In this simple view, if the system uses a DFP it is contained within the system control. Such a system, may be decomposed as shown in Figure 2.1(b). Here, the system control itself has been decomposed into a *system objective control*,[3] a DFP, and possibly a *knowledge base* (a storage of data, representing the memory of the system, containing data and information about the environment and its inherent activities and the state of the system itself). An arrow, inside of the system box in the figure, denotes influence on the object at its head by the object at its tail,

[2] Here we use the term "environment," others used in the literature are "world" and "workspace".

[3] The system objective control is not equivalent to the system control in Figure 2.1(a) since some control may be performed by the DFP (by its process refinement).

e.g., the system objective control and the DFP may both access the set of system resources and access (either through direct access or with the help of data mining techniques) and alter the knowledge base.[4] The influence between the system objective control and the DFP highlights that the control may access the information of the process, inhibit (e.g., to acquire resources), configure, and control it.

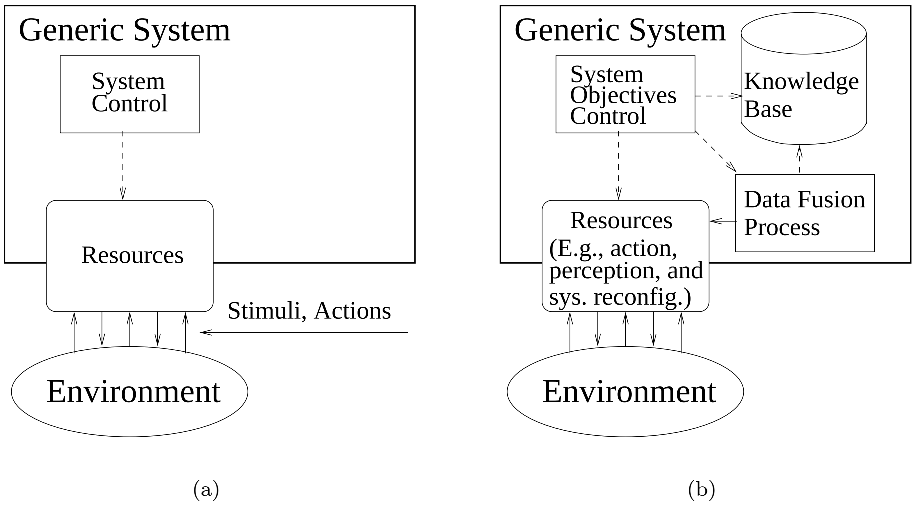


Figure 2.1. (a) A coarse sketch of a generic system which result depends on its interaction with some environment. (b) A generic system including a data fusion process. The system control has been decomposed into a system objective control, a DFP and possibly a knowledge base.

In the following subsections, we further discuss properties of the surrounding environment (Section 2.2.2), the information that can be acquired from it (Section 2.2.3), and of course the DFP itself.

2.2.2 The environment

The degree of difficulty of implementation and management of a specific DFP is heavily dependent on the characteristics of the *relevant* environment[5] it is observing.

[4] The knowledge base may contain both static information (e.g., laws of physics, military doctrine, and building plan drawing) which the DFP may not alter, and dynamic information (e.g., environment object locations and relations) which the DFP may alter if it derives some complementary or contradictory information.

[5] By "relevant" environment, we mean the subset of the environment the DFP has been designed to interpret. If the relevant environment is very restricted in comparison to the "complete" environment, the characteristics of the two may vary greatly.

For the discussion in this section, we call the agent that contains the `DFP`, and observes some environment, the *subjective agent* (SA). The name emphasises that the properties of the environment are highly dependent on the beholding agent (i.e., the SA). The environment which the SA perceives and interprets might, depending on application, be simple and accessible but is typically (environment properties adopted from [RN95]):

Inaccessible The complete state of the relevant environment can not be determined by the SA. The relevant environment is often inherently complex and it is not practical or even possible to design omnipotent sensors that timely determine the exact state of the environment. A chess board, e.g., is completely accessible, whereas a poker game environment is not (at least not to a non-cheating SA).

Nondeterministic The outcome, or value, of actions performed in the environment are not deterministic. The environment is typically nondeterministic if the result of an action in the environment is dependent on some stochastic variable. More frequently, from the perspective of an SA, the environment will *appear* nondeterministic if it is also inaccessible.

Nonepisodic Actions performed by the SA affects the future evolution of the environment. Chess and other multi-player games, e.g., are nonepisodic since there exist opponents who will respond to the moves by the agent. So the performed action in one episode (an episode consisting of a percept and action selection of an agent) may affect the selection in future episodes. Conversely, the process of selecting an action in a nonepisodic environment now should consider the actions influence on future episodes.

Dynamic The configuration of the environment will change with time independent of the SA. In a *static* environment, the SA could consider its choice of actions almost indefinitely. In dynamic environments, however, a lengthy deliberation about actions would lead to decision-making based on obsolete information. The environment is *semi-dynamic* from the perspective of the SA if the state does not change over time, but the performance of the SA does.

Continuous The features of the environment may be continuous, e.g., positions, speed, and temperature, and also the possible actions.

2.2.3 Properties of information

A discussion about the meaning of data and information deserves a paper (or probably even book) of its own, accompanied by an extensive survey. For our study in this chapter, such an effort is outside of our scope, and a vague notion of data being something originating from sensors and information being processed and interpreted data will suffice. However, exact properties of data and information are discussed now and then in data fusion, and are important for the refinement and evaluation of information. We would, hence, like to present a small selection of works that discuss this matter. As we will see, the usage of the terms are not always consistent.

A classification of terms is suggested in [Ste99]. There, the term *data* refers to observations and measurements from information sources, *information* is data, organised and placed in a context (corresponding to Level 1 and Level 2 in the JDL-model), and *knowledge* understood and explained information. Some works emphasise that knowledge should be stored information (e.g., [MMMW01]); information that has been accepted and that constitute a prior belief for further reasoning and decision-making.

The authors of [AAB$^+$01] use information as a general term with subcategories such as *observations* and *knowledge*. According to their classification, observations are, e.g., data from sensors, facts and evidences. More generally, observations all refer to the current state of the environment. Knowledge, on the other hand, is defined as information that describes general properties of the environment, such as characteristics of a class of situations.

Observations and knowledge are, in [AAB$^+$01], called *descriptive knowledge*. Two types of *normative knowledge*, *preferences* and *regulations*, are also suggested. Preferences is information about individuals' desires and regulations are rules governing the environment, e.g., expressing what feasible events there are.

In fusion applications, sensors rarely are capable of accurately conveying (measuring and reporting) the environment features they are observing. To capture this discrepancy between the real world and measurements, some sort of representation of information imperfection is necessary. By imperfect or defective information, we mean information that is in some way insufficient for making efficient decision. [AAB$^+$01] suggests a taxonomy of such defective information, arguing that information used for fusion is imperfect in some sense, otherwise there would be no need for fusion. The aspects of information quality mentioned are: *ambiguity*, *uncertainty*, *imprecision*, *incompleteness*, *vagueness* and *inconsistency*. A brief explanation of these aspects is provided in Table 2.1

In [AABW00], *information awareness* denotes the "understanding of the usefulness of information and the possibilities to achieve better information." In order to deliberately achieve the state of information awareness, i.e., to get a better understanding of the available information and to make better decisions, three properties are attached to information: *precision*, *quality* and *utility*. Precision regards the certainty of a piece of information. A piece of measurement information could, e.g., have a measurement error covariance matrix as the value of its precision property. Quality of a piece of information reflects its ability to support decision-making, i.e., to discriminate between possible actions or decisions. The utility of information is its expected contribution to an action selection situation, comparing it to the situation where the information is not available.

Concepts such as those discussed in this section are important both for the interpretation and usage of information by a `DFP`.

Table 2.1. Aspects of defective information from [AAB+01].

Aspect	Meaning
Ambiguity	It is unclear what the information refers to, and it may be interpreted in several ways
Uncertainty	Lack of information that makes it impossible to say whether a statement is true or false. For instance, the statement "Intruder detected", may be an uncertain piece of information if the information source cannot be completely trusted.
Imprecision	The degree of imprecision in a piece of information is dependent on the *granularity* of the language it is expressed in and the needs of the decision-maker. E.g., the statement "distance to target is 100m" is precise if the required degree of granularity is meters, but imprecise if it is centimetres.
Incompleteness	Information is incomplete if it does not capture all relevant aspects of a phenomenon or entity.
Vagueness	A piece of information containing a vague quantifier, e.g., "young" for age, is vague.
Inconsistency	A set of pieces of information is inconsistent if the pieces contradict each other.

2.2.4 A functional data fusion model

To break down and further analyse the data fusion process, we use a functional model[6] for data fusion; the so called JDL-model, named after the Joint Direct-

[6] A functional model consists of definitions of functions which could comprise any `DFP`.[SB01] Unlike a process model, it does not specify interactions between functions, only functional aspects of a `DFP`.

ors of Laboratories, mentioned above. Although originally developed for military applications, the model presented here is generally applicable. Furthermore, the model does not assume its functions to be automated, they could equally well be maintained by human labour. Hence, the model is both general and flexible.

The revised JDL-model [SB01, Whi98] includes five fusion levels, i.e., a decomposition of the DFP into five different functions. The levels specify logical separations in the DFP and divide information into different levels of abstraction depending on the kind of information they produce (where the lower levels yield more specific, and the higher more general, information).[WL90b]

The purpose of the sketch of the JDL-model in Figure 2.2 is to provide an overview of the functions without suggesting any particular type of implementation or application specific details. The context of the JDL-model, the governing system objective control and the available resources, is also depicted.

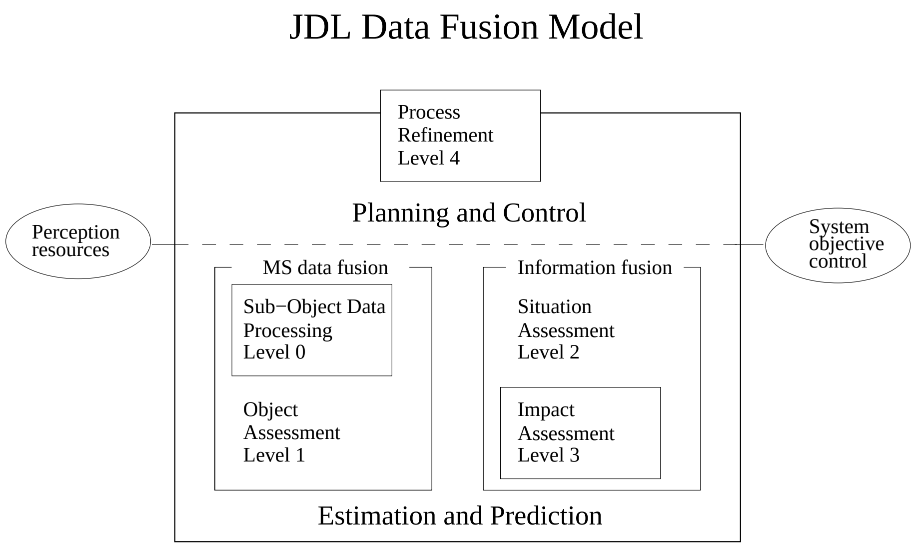


Figure 2.2. The JDL data fusion process model is composed of five different functions (Level 0-4).

Level 0 - Sub-Object Assessment: The purpose of Level 0 is to associate and characterise sensed signals. To associate signals means to assign them to the one and same entity (e.g., tracked target) of the environment. Typical techniques used in this level belong to signal processing and feature extraction. In this level, no semantic meaning is assigned to the assessed data.

Level 1 - Object Assessment: In this level, which is sometimes referred to as *multisensor data fusion* or *multisensor integration*, data is combined to assign dynamic features (e.g., velocity) as well as static (e.g., identity) to objects,[7] hence adding semantic labels to data.This level includes techniques for data association and management of objects (including creation and deletion of hypothesised objects, and state updates of the same). The study of the *grounding problem* [Vog97, Har90] in the artificial intelligence community is related to this level.

Level 2 - Situation Assessment: This level involves aggregation of Level 1 entities into high-level, more abstract entities, and relations between entities. An entity in this level might be a pattern of connected objects of Level 1 entities. Input data are assessed with respect to the environment, relationship among Level 1 entities, and entity patterns in space and time.[PCCB97, Hal92a]

Level 3 - Impact Assessment: The impact assessment, which is sometimes called *significance estimation* or *threat refinement*, estimates and predicts the combined effects of system control plans and the entities of Level 2 (possibly including estimated or predicted plans of other environment agents) on system objectives.

Level 4 - Process Refinement: Process refinement evaluates the performance of the `DFP` during its operation and encompasses everything that refines it, e.g., acquisition of more relevant data, selection of more suitable fusion algorithms,[8] optimisation of resource usage with respect to, for instance, electrical power consumption. Section 2.3 deals with process refinement in more detail. Process refinement is sometimes called process adaption to emphasise that it is dynamic and should be able to evolve with respect both its internal properties and the surrounding environment. The function of this level is in some literature handled by a so called meta-manager or meta-controller.[9] It is also rewarding to compare Level 4 fusion to the concept of *covert attention in* biological vision which involves, e.g., sifting through an abundance of visual information and selecting properties to extract.

A typical logical information flow among the functional levels is depicted in Figure 2.3, where process refinement responds to impact and situation assessment. In the Figure, process refinement, of course capable of interacting with functions of all levels, merges its own plans for action with the expected plans of the observed environment.

[7] It could be debated whether or not "object" is the most appropriate term to use. In some applications, it might not be clear what an object is. More appropriate terms might be "component", "constituent", or "element".

[8] Different algorithms may be the most appropriate for different situations, depending on available data and tasks. E.g., some type of task might require detailed information from the `DFP`, while some other settle for more coarse.

[9] The reason for these names are that it manages the other processes.[PCCB97]

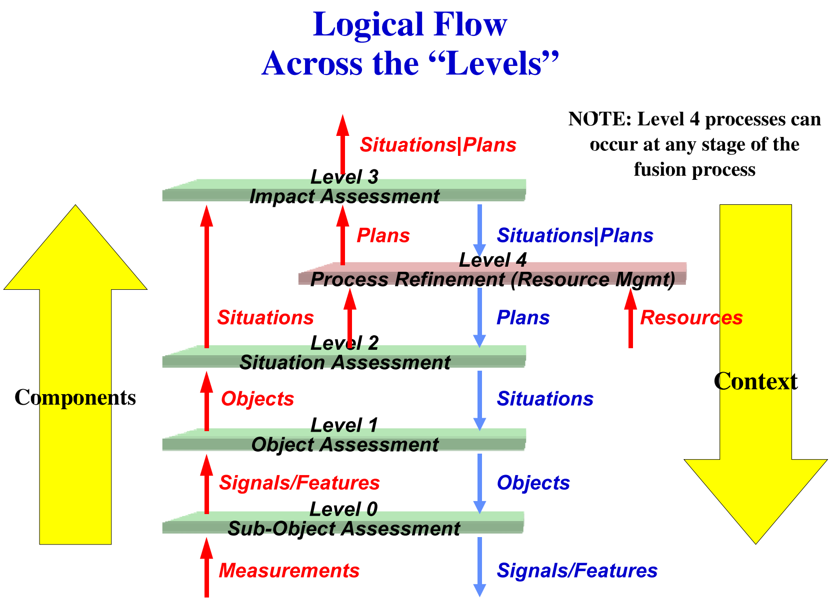


Figure 2.3. Typical logical flow between the JDL model functions (image borrowed from [SB01]). By courtesy of Alan Steinberg.

Remarks on the JDL model

A Level 5, user refinement, has also been proposed.[BH00] While the DFP maintains and refines all available information, a user is only interested in the subset of the information it needs for its own decision-making. Sometimes, some information is restricted to only users with appropriate access privileges. The purpose of Level 5 is to handle the problem of providing users of the DFP with the "right" information, corresponding to the users need and access rights.

A few things should be mentioned about this definition of the JDL-model. First, the term "object", used to denote the entities of Level 0 and Level 1, is a heritage of the military origin of the JDL-model and a bit too restrictive. E.g., the application environment may be represented in such a way that it is not clear what an "object" might be.

Second, although it is useful to emphasise impact assessment, [SB01] identifies Level 3 to actually be a component of Level 2. Likewise, Level 0 is recognised as a special case of Level 1.

Furthermore, in Figure 2.2, Level 4 is separated from the other levels. Level 4 is quite different from the other levels in the sense that it does not produce any new information (i.e., it does not "fuse"), and, also, whereas the functions of the other levels have a direct effect only on the internal operations of a system, it may have a direct effect on the systems external behaviour (through the use of perception resources, explained in Section 2.4.2). Hence, Level 4 is more about control than estimation. In spite of this, Level 4 was incorporated into the data fusion model because of its intimate relationship with the other four levels.As in Figure 2.2, Level 4 is sometimes placed on the border of the data fusion model. Its peculiar position indicates that Level 4, in general, is dependent on outside processes (that might interfere with its usage of perception resources).

It also bears mentioning that the probable reason for the somewhat counterintuitive numbering of Level 0 is due to the fact that it was not included from the first version of the JDL-model, so the name Level 1 was already taken. It was introduced in 1997 with the revised JDL-model.[Whi98]

Whereas Level 0 and Level 1 concern multisensor data fusion, i.e., the combination of data from different sensors, Level 2 and Level 3 are often referred to as *information fusion*.

A comparison between the JDL-model and related models, such as Dasarthy's functional model and OODA process, is provided in [SB01].

2.2.5 Implementation issues

Moving from the functional model in Section 2.2.4 to a working implementation in a real environment involves a number of design considerations: including what information sources to use (e.g., single sensor or multisensor sets), what fusion architecture to employ (centralised/decentralised), communication protocols, etc. In this section, we discuss the properties of single and multisensor systems.

Admittedly, the fusion of data is decoupled from the actual number of information sources and, hence, does not require multiple sensors. The reason is that fusion may be performed on a temporal sequence of data that was generated by a single information source. E.g., a fusion algorithm may be applied to a sequence of images produced by a single camera sensor. However, employing a number of sensors provides many advantages (as mentioned e.g. in [MDW94, LK92, WL90c, Lli88]):

Redundant information When multiple sensors perceive the same feature of the environment, the redundant information can be exploited to reduce uncertainty about the status of the feature and increase the reliability in the case of sensor failure.

Complementary information Multiple sensors may perceive different features of the environment, consequently allowing complex features (which could not be sensed by each sensor independently) to be perceived.

More timely information Due to simultaneous measurements of multiple sensors, information may be acquired at a higher rate.

Extended spatial coverage Measurements can be made over a possibly large area.

Increased robustness Some sensors may be making measurements, while some others fail or are temporarily unable to.

Note that it is sometimes useful to consider multisensor systems as abstract sensors (or logical sensors as in [BG97, GH89, HS84] or virtual sensors [Mui90]). For instance, if the motivation for using multiple sensors, in some situation, is to decrease the time interval between observations, we have constructed a simple abstract (or logical) sensor which is more timely than its "concrete" sensor components. Since the abstract sensor has similar properties to its components', it might be managed in a similar way (rather than being treated as something very different, i.e., a complex system of sensors). This issue is further treated in the discussion about services in Section 2.4.2.

Unsurprisingly, there are also difficulties associated with the use of multiple sensors, as further noted in [LK92]:

Sensor registration Failure to make correct associations between signals or features of different measurements. This problem and the similar *data association* problem are incredibly important and apply also to single sensor fusion.

Conflicting information and noise Assumptions, more or less realistic, are often made to enable the use of some fusion techniques. Noise in input data sometimes yield conflicting observations, a problem that has to be addressed and which does not arise in single sensor fusion.

Administration multiple sensors have to be coordinated and information must be shared between them. Such requirements has to be dealt with by the designer of the multisensor data fusion system.

The JDL-model in Figure 2.2 should not be considered to be an architecture for implementation, rather a classification schema for `DFP` functions. The depicted model is over-expressive for many applications, and not all functions should be implemented for every application. In fact, many systems only implement Level 0 and Level 1.

2.3 Process refinement

The process refinement (i.e., the meta-controller), the fourth level of the JDL-model (described in Section 2.2), monitors the other parts of the data fusion and process and tries to improve its performance. It is important to emphasise the asymmetrical symbiosis between process refinement and the other functions of the JDL-model. Process refinement has no purpose without the other functions, but the other functions may exist in an application without support of process refinement. However, for efficient and flexible data fusion in complex and environments, process refinement is a magnificent support.

Process refinement is decomposed into three functional parts by [PRT98], and presented slightly modified here:

Input refinement: Controlling system resources in order to improve (i.e., produce more useful information) the `DFP`. This includes detecting and avoiding unreliable (possibly faulty) sensors. In order to achieve the refinement, various constraints may have to be considered (e.g., expected information gain, or limited power resources). Perception management (discussed in Section 2.4), including deployment and parameter configuration of perception resources, is also an example of input refinement.

Process control: Modify the parameters of sub-processes (e.g., processes performing multisensor data fusion) of the `DFP` in order to improve its performance. This modification may include fine-tuning or even changing fusion algorithms, choosing or altering connections between components of the `DFP` (depending on, e.g., data traffic and data capacity of network links), selection of information, and tuning of filters.

Inter-level information flow: Controlling the information exchange between levels.[10] The most obvious interaction is perhaps that of lower levels providing information for higher fusion levels, but it would also be common for, e.g., results of Level 2 to infer object hypotheses in Level 1.

Thus, an implementation including the process refinement function respects the uncertainty of the `DFP` and is aware of its limitations in terms of perception resources and internal process properties.

The requirements of process refinement highlights the need for system perception, not only of the state of the environment, but also, of the internal state of the system (in order to improve the internal system performance).

Process refinement is driven and affected by:

- the results of the Level 0 through Level 3 of the `DFP`,
- requests from the system objective control (in Figure 2.2),[11]
- intended actions of the system objective control,[12]

[10]Note that we here deviate from the work in [PRT98], where this part was called "Additional or complementary processing". The difference is that we here expand the original idea of hypothesis reinforcement from higher to lower levels to include all interaction between `DFP` levels.

[11]Two different kinds of requests are identified by [DALH94]: manual and automatic. A manual request in a command and control (C^2) application could, for instance, be an intelligence inquiry by a system operator. An example of an automatic request is the, close to instantaneous, localisation request of a targeted object upon missile launch. Put differently, the automatic requests are deterministic in some sense since they are generated internally, the manual are not.

[12]Since resources are shared, the system objective control may inhibit the process refinement by allocating resources which it could have used.

- its own awareness about the internal system structure and resources (status, characteristics, and limitations), and environment (e.g., topology, terrain, etc).

The next section will describe perception management, the part of process refinement that deals with IA, in more detail.

2.4 Perception management

The need for management of sensors is dependent on the available resources and sensing tasks and must be motivated. In [BP99b], two preconditions are listed: (i) manageable *resources must be agile*, and (ii) *resources are limited with respect to tasks*. If resources are not agile, they cannot switch between sensing tasks and are not manageable in any beneficial way. Hence, if they are not agile, we might as well leave them alone.[13] Some sensors are already optimised for the task they were designed for and no gain is likely to occur by trying to control them in some manor which they were not designed for.

The second precondition, that resources are limited with respect to sensing tasks, suggests that there should be a competition among the sensing tasks for the available sensing resources. If these two preconditions are met, and for highly potent systems working in difficult environments they tend to be, a perception management function for improved IA should be considered. We add that (ii), in a wider regard, could be extended with a need to control sensors in order to respect some mission objective (such as avoiding detection by adversary sensors, interference between own sensors or decrease energy consumption) and to modify the scope of the sensors (e.g., to reposition occluded sensors). Thus, we could replace (ii) by *resources are limited with respect to tasks, objectives or scope*.

Many names have been used for the management of IA. In the context of the general data fusion model described in this work and depicted in in Figure 2.2, most previously suggested names seem more or less delusive:

Sensor management A quite specific term that brings rather uncomplicated information sources, such as, sonar, and infrared sensors, to mind. More complex information sources, e.g., cameras, or other high-level information sources, e.g., news agencies and humans, are not as frequently referred to as sensors.

Asset and Resource management Are too general in this case, since they could encompass all kinds of resources, including money, food, power-supply, cars, sensors, etc.

[13]Some management works only control by turning sensors on and off, hence, we also have to consider this ability as an expression of agility.

Information (re-)source management Is also too general. An information source[14] may for instance be a database, and the purpose of the IA in perception management is not to manage databases (that would possibly be the job of some other component of the process refinement; see Section 2.3), rather to manage perception resources which directly perceive the state of the environment.

Collection management Refers to the collection of data or information, but may be mistaken for the management of some relatively static set of data (e.g., maintenance of a database).

Here, the term perception management is preferred, since, in accordance with the "input refinement" concept in Section 2.3, it is the perception of the environment (and ultimately the comprehension of the environment), also to support the higher fusion levels, which should be improved, not directly the physical sensors as suggested by the term sensor management.

Even though a new concept is used, the ideas of perception management are inspired by the different kinds of management mentioned above. The general relationships between resource, perception, and sensor management,[15] depicted in Figure 2.4, clarify this fact.

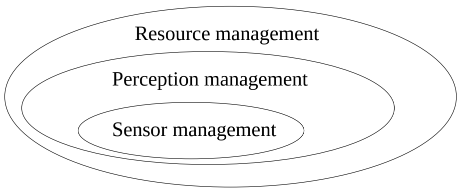


Figure 2.4. A coarse sketch of the relationships between some types of management. Resource management is considered to encompass perception management, which in turn encompasses sensor management.

In [JX03], where we presented the concept of perception management for the first time, we explain our view on perception management. We believe, e.g., that the term and concept of perception management effectively situates the process of IA in an agent theoretic context (which facilitate fluent communication between various fields of research) and naturally encompasses processes that support IA.

As a theoretical concept, perception management is effectively a superset of sensor management and a subset of resource management (which as mentioned also involves management of resources for other purposes). In practise, however, typical work in PM will probably act as initiator and controller of various sensor management processes.

[14] It might also be questioned whether the term "information source" really refers to a perception resource, rather than some signal generating entity of the environment.

[15] In some previous works, sensor management and mission management, i.e., the mechanism which decides which perception tasks to perform, are separated [DALH94, MM94]. Here, they are both considered to be a part of perception management.

The discussion about PM in the rest of this section journey beyond our previous publication.[JX03] It elaborates on the duties of the PM, the stimuli that affect it (Section 2.4.1), properties of the resources it must manage, and defines the concept of *perception service* (Section 2.4.2 and 2.4.3). Properties belonging to the management of sets (or systems) of resources are also discussed (Section 2.4.4), as well as control architectures (Section 2.4.5). Admittedly, most of this work should also be applicable to sensor management, which the reader might feel more comfortable with.

2.4.1 Function and stimuli

The purpose of this section is to give a flavour of typical work a perception manager could be performing, and the description herein is by no means complete. In the following chapter, we will survey previous works in this field show more details about IA.

It is the responsibility of a comprehensive perception manager to perform one or more of the following functions:

- optimise perception resource usage for IA with respect to, e.g., constraints on resources and environment, cost of usage, or risk of detection;
- degrade performance gracefully in the presence of sensor failures, inaccessibility of perception resources (perhaps due to preemption by the system objective control), or when the perception resources are limited and can not serve all information requests;
- prioritise and carry out IA tasks when perception resources are limited and cannot support all tasks simultaneously.

Important issues a comprehensive perception manager has to deal with are, for instance, the conflict between monitoring known entities of the environment, on the one hand, and the need to discover new entities, on the other. Another important issue, just as inherent as the previous, is that of utilising perception resources in such a way that likely and critical events can always be sensed when necessary (e.g., sending a lot of mobile sensors to a remote observation spot might be unwise if this means that critical observations cannot be made in time in some other spot). Additionally, a perception manager typically acts in a dynamic environment and should continuously be prepared to re-plan and reconsider its selected actions and priorities.

Since perception management is a part of the process refinement, it has the same types of stimuli. The stimuli must somehow be transformed to well-specified *information acquisition tasks* that the perception manager can try to satisfy.

Using the stimuli suggested in Section 2.3, we say that *internal* stimuli originate from the results of the other levels of the `DFP`. *External* stimuli originate from sensing resources, requests and plans from the system objective control, and external users (see Figure 2.5):

User Agent A user agent may, here, query the DFP for whatever information it thinks seem interesting and useful (this might be a physical or computational agent). Sometimes this information is already available in the DFP, but otherwise the perception manager will have to consider the request (the DFP here plays the role of a decision support system);

Mission control The mission control informs the DFP about (i) what plans the system intends to execute in the nearest future and (ii) what the *focus of attention* should be. Stimulus (i) helps perception management predict, e.g., what resources will be available to it in the future. Stimulus (ii) directs the DFP towards the aspects of the environment that the DFP should use its resources to estimate;

Resources/services There might be a need for resources or services,[16] which have in turn received a task from the perception management, to report back to the DFP. A report, could, e.g., inform the DFP that a task no longer can be sufficiently performed or that the status of some resource has changed.

There is actually no clear distinction between the notion of user agent and mission control in this context, it is just a matter of roles. I.e., an agent may have the responsibility of making plans for the system, and, conversely, a mission control may query the DFP.

2.4.2 Perception resources and services

Naturally, the performance of perception management is dependent on the (perception) resources that is available to it. The quality of the resources will also decide the quality of the perception management itself. In this section, we discuss the meaning of perception resource and properties relevant for their management.

The resources of a generic system include all resources the system is said to possess, e.g., amount of money or fuel, number of vehicles, competence of labour, or sensors. The resources may also include systems of resources, e.g., complex tracking systems, or buildings. The resources set the limit of the capabilities of the system.

The resources, included in Figure 2.1(a), can be used for actions (to affect the environment), perception (perceiving features of the environment), and system reconfiguration (internal alterations of the system).

To the system control, all resources are accessible (even though they might not always be deployable), but the DFP may only utilise perception resources and resources supporting system reconfiguration. Furthermore, to the perception manager, which is the main focus of this chapter, only the perception resources are directly available (support resources such as money or fuel are only indirectly available through usage of resources).

[16] Here, we think of service as some sort of process (or logical sensor as described in [HS84]) that uses one or more resources to retrieve information.

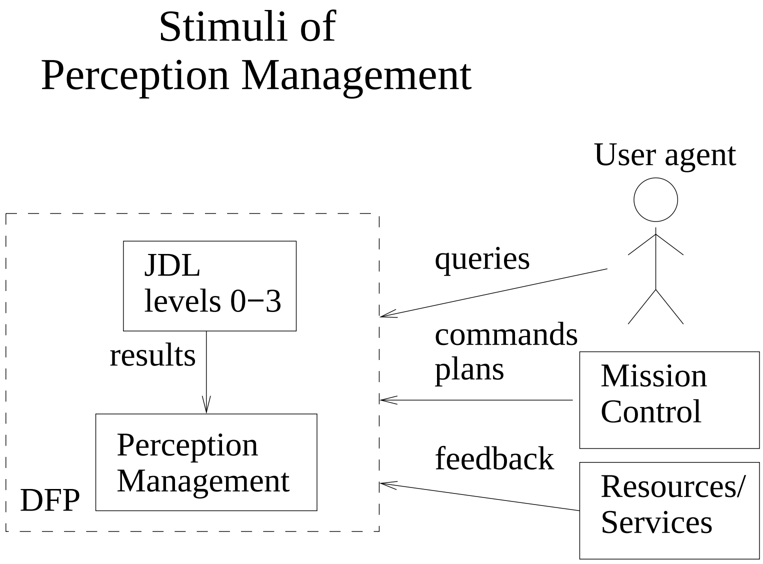


Figure 2.5. The figure shows three sources of external stimuli that affects the perception manager. Internal stimuli comes typically from the lower data fusion levels.

Perception resources can simply be said to include all resources which can be useful for the perception of the surrounding environment, and, analogously, action resources can be said to include all resources that are useful for action in the environment. Naturally, partly because some complex resources have multi-capabilities (e.g., multi-purpose platforms), and partly because some resources (e.g., fuel and money) are applicable to sustain different kinds of processes, some resources are used both for perception and action. Hence, this interdependence creates a inevitable conflict between the DFP and system objective control when requesting the same resources.

A manageable subset of the perception resources, that may perceive the environment and return percepts (meaningful to the DFP, e.g., a sensor) will also, a bit sloppily, be called *perception resource*. In this thesis, that is what we normally mean by a perception resource. Examples of perception resources are diverse entities, such as, tactile sensors (robotics), stock-market analyst (financial applications), and human scouts and signal intelligence services (command and control).

For the sake of perception management, it might be useful to create a hierarchy of *perception services* to control rather than individual resources. The motivation is twofold. First, it is convenient to construct abstract high-level sensors based on actual sensors (this idea is roughly the same as that of logical sensors in [HS84]). Such abstract sensors, or perception services as we prefer to call them, could be tailored to acquire specific information, information that is likely to be requested while using the DFP. Second, instead of moving towards more abstract sensors, we can also see that some sensors have multiple modes. A perception service might as well represent a mode of a physical sensor. Thus, the set of perception services may actually represent a complete universe of sensing action alternatives of perception management. The alternatives (from atomic ones, such as sensor modes, to very abstract logical sensors) might be more or less decoupled from the physical devices that implement them. This is illustrated in Figure 2.6.

However, this convenience comes at a price; while the set of possible sensing actions becomes more straightforward to manage with this approach, increased complexity is introduced in terms of dependencies among the services. The use of one service may inhibit the use of another. Two services that overlap, in the sense that they require access to the same perception resource will inhibit each other. But, note that this problem exist already for perception resources, even though in a smaller scale, since resources themselves might disturb each other, and in that sense inhibit one another. Multiple mode sensors may also only be active in one mode at a time.

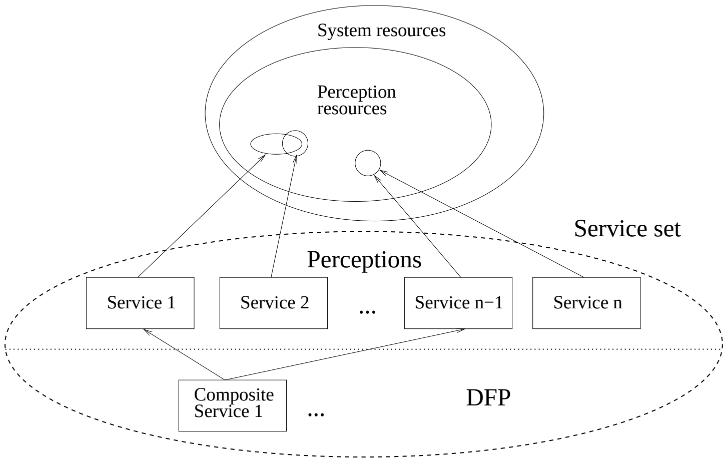


Figure 2.6. The interface between the perception resources and the DFP is the perception services set.

2.4.3 Resource properties

When designing a perception management system, it is useful to ascribe properties to sensing resources to make them amenable to formal mathematical reasoning. These properties primarily refer to perception resources, but may also apply to most services. Here, we present two categories of properties: *scope* and *value*. Scope properties concern the sensing characteristics of the sensing resource and the range of observations it is capable of obtaining. Value properties affect the usefulness of the sensing resource.

Scope properties

The most obvious scope properties of a perception resource are those concerning sensing characteristics including, e.g., resolution, detection performance, frequency range, sensitivity, measurement accuracy, data output format, etc. Some of them are described further in [Hal92a].

Other important scope properties involve the control of the resource. The following sections describe *access type*, *active or passive*, *availability* and *response time*.

Access type

Access of perception resources can be performed at different levels of abstraction. In [BP99b] two views of sensors are discussed: *parameter view* and *mode view*. Parameter control of a resource allows direct access its complete spectrum of expressions. Modes, on the other hand, provide a conceptual view of the resource by encapsulating the parameters and simply presenting pre-specified operations.

Although the parameter view certainly offers the most degrees of freedom, the management of perception resources on the parameter level may be unnecessarily complex. The mode view reflects the notion that it is often more efficient to let the responsibility of the resource parameters be assigned to the resource itself (its modes already optimises its performance). Clearly, it is less scalable to locate responsibilities for a perception resource externally. Furthermore, it is also the case that some perception resources can only be managed by modes, e.g., humans.[17]

Notice that the mode view also applies to whole systems of perception resources, e.g., a target tracking system. In fact, the parameter view may be regarded as a special case of the mode view, the lowest mode view in a hierarchy of mode views.

Active vs passive perception resources

Sensors are often classified as either *active* (e.g., radar) or *passive* (e.g., an IR camera). Active sensors or systems of sensors need to radiate energy of their own to perceive a target source, while passive sensors rely on the target's own radiation.[NN00, BP99b] There are two reasons for this distinction: first, control of an active sensor is generally more versatile and offer the ability to "provoke" the environment for richer IA, e.g., higher resolution due to its control over the radiated energy, and, second, the risk of another agent detecting the energy radiated by the active sensor.

[17] In the case of human being, a mode could be interpreted as an assignment.

Notice that the second reason mentioned in the previous paragraph is quite weak in a theoretical discussion; even non-radiating sensors may reveal themselves to observers (possibly due to poor management). It might still be useful to distinguish between resources which actively expose themselves to the risk of being detected and those which do not. Transferring the classification of active and passive sensors to the general domain addressed in this chapter justifies a wider interpretation. Here, we say that a *passive perception-resource*, just as the passive sensor previously mentioned, does not have to initiate any sequence of actions on the relevant environment in order to perceive (as depicted in Figure 2.7(a)) and, correspondingly, an *active perception-resource*[18] has to affect the state of environment to make the required perception.

The distinction between active and passive perception resources is not essential in all applications, but it reflects that perception might leave "fingerprints" or clues in the environment. That is essential in applications where other alien agents can interpret the footprint and take advantage of it (e.g., in command and control applications).

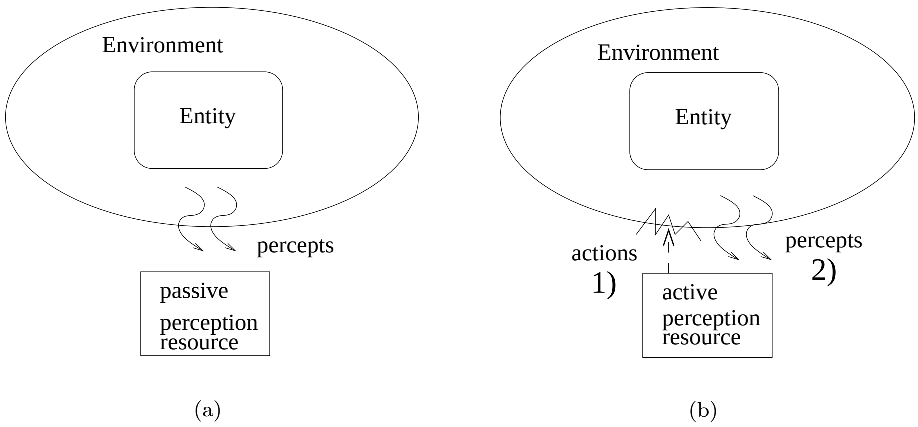


Figure 2.7. (a) Usage of passive perception resource (b) Usage of active perception resource

A perception manager (i.e., a process or agent devoted to perception management) using active perception resources should weigh the advantages of using the resources against the risk of being detected.

[18]Please notice the unfortunate inconsistency with the term "active perception" used in computer vision.

Control space constraints

Whereas the previously mentioned properties are static, i.e., belonging to the structure of a sensor, control space constraints (CSCs) describes objectives that must be considered while managing the resources. E.g., in some works, one wants to minimise energy consumption [PH03] or the number of sensors transmitting observations.[KP98] We can often express CSCs as unsuitable regions of the control space (cf to *configuration space* for motion planning in robotics [Lat93]).

Availability and redeployability

Perception resources are not always available to the perception management. There are various reasons for this. Perception resources could be unavailable, permanently or temporarily, for a specific task due to, for instance:

- characteristics of the resource (e.g., a sonar can not generate data continuously, and a mobile sensor cannot relocate instantaneously)
- preemption by system objective control,
- internal disturbance (e.g., resource break-down or failure),
- external disturbance (e.g., jamming, or unsuitable weather conditions),
- destruction, or capture,
- the operations of another controllable resource (e.g., some resources inhibit each other and can not operate at the same time; consider, for instance, the ordinary multi-meter which can not both measure current and voltage at the same time).
- unsatisfiable task requirements (e.g., the resource is, for some reason, incapable of performing its operation at the time or place which is requested by the task)

A perception management function that is aware of the fact that resources might become unavailable should integrate this knowledge into its operations. It could, for instance, try to estimate, and continuously re-estimate, the risk of sensors being destroyed, or to estimate the risk of them being preempted by studying the plans of the system objective control.

We imagine that some resources are inexpensive and used only once, perhaps, e.g., the sensors of future ground sensor networks. More common in applications are those that might be reconfigured over and over again (i.e., redeployable). If sensing resources are limited and not redeployable, the perception manager should carefully consider the value of deployment.

Value properties

The application of perception resources is not merely dependent on the scope of the resources. It is also dependent on its value, i.e., the expected *utility* of its usage and associated *cost* with respect to the IA task at hand.[19] Utility quantifies the (expected) contribution, of the usage of resources, to the performance of the enclosing system. Cost, conversely, expresses the effort the system needs to undertake in order to execute the complete sensing action.

It is important to make the perception manager aware of the notions of utility and cost, because, certainly, the cost of using a resource that is guaranteed to provide a reliable answer might outweigh its utility and render its usage meaningless. Consider, e.g., the situation of contemplating using a wonderful perception resource, a hypothetical very trustworthy diamond ore detector. You might make a fortune if you mine the rock in question, but if the cost of using the efficient diamond ore detector, powered by an expensive fusion reactor, exceeds your expected profit from the mining, you might just ignore that option.

The usage of a perception resource is normally associated with one or several costs. Costs may be concrete and directly referring to the physical properties of the resource (e.g., measured in fuel, electricity, CPU cycles or money). It could also be a bit more abstract, e.g., referring to the expected time its use will take. It could also be a composite of several factors. Often, cost, just as utility, is simply represented by a single integer, inducing a relative ordering of costs. If the resource has to be reconfigured somehow (e.g., relocated) to be useful, there might be a supplementary cost involved. Some properties that might affect the value of using a resource are *reliability*, *condition* and *response time*.

Reliability

Perception resources may fail, either temporarily or permanently. Furthermore, sometimes a sensor might be sending the misleading signals when it is being subject to deception by an adversary agent. Hence, the perception manager might want to estimate to what degree they may be trusted. A perception manager could monitor the status of the sensors, and use unreliable sensors less frequently or not at all.

Condition

Related to reliability is the property of condition. The perception manager could monitor the health status of its resources. A resource with low health (e.g., low battery power) might be spared from routine work until an emergency occurs.

[19] This does not apply only to sensing actions, but to all kinds of actions (which the sensing actions are a subset of).

Response time

In dynamic environments, information is often perishable. Hence, information that arrives late is normally less valuable than the same information acquired instantaneously.

2.4.4 Systems of perception resources

In the previous section, we discussed properties of a single perception resource which are important to consider when managing it. Those properties are important and useful when designing a perception manager. However, a `DFP` normally has more than a single sensor, and properties of whole sets or systems of resources (or services), that may not be obvious when studying an isolated resource, must also be considered. One of these *set properties* is that of the composition type of the set; whether it is *homogeneous* or *heterogeneous*. Another is if there are *dependencies* between resources or services in the set.

Homogeneous vs heterogeneous resource set

A homogeneous set of perception resources contains resources or services which are indistinguishable to the system. If perception resources produce the same kind of information, e.g., position estimates, we might want to label the resource set as homogeneous. However, we might also require that the control properties of the resources are similar, e.g., that all sensors return a measurement within some time limit. Hence, we might want to distinguish between *information heterogeneous* and *control heterogeneous* sensors.

Distribution of IA tasks in a homogeneous set of sensors is fairly simple. Most work in the sensor management literature today deal with this kind of sensor sets; examples are [PW00, Nas77].

Heterogeneous sets, on the other hand, have members which are distinguishable by the system. Hence, since characteristics may vary a lot among the members, it is essential that the PM makes an intelligent selection as task allocation becomes more tricky.[CFK97] This task allocation problem has mostly been dealt with in the field of distributed artificial intelligence (see, e.g., [Wei99]).

An increase in the number of sensors in a homogeneous set may increase the *scope of congruent observations* of the resource set as a whole (since the system improves its sensing coverage), i.e., more information of the same type (e.g., position estimates) may be acquired. An increase in a heterogeneous set may not only increase the congruent scope, but also the *incongruent scope*, i.e., more types of complementary information may be acquired. However, an increase in the number of resources also means that the efforts to manage the resources efficiently must increase accordingly.

Dependencies

In sets of perception resources, resources may choose to cooperate and aid each other through direct communication in order to improve the performance of the IA (this type of cooperation is called *cueing*). A few situations where such aid is useful are described in [WL90d]:

- When an entity escapes the sensing range of some resource, that resource may be able to direct other resources within sensing range to approximately where and when it will appear within their range;[20]
- The discovery of some information by one resource, s_1, (e.g., the detection of a target) may suggest the use of other resource, s_2, to acquire more detailed information (e.g., a position estimate). The interdependence between s_1 and s_2 can be called a *causal relationship*;

Another type of causal dependency is presented in [DW88]. There sensors are directly dependent on observations of other sensors to form observations of their own.

2.4.5 Control of perception management

As explained in Section 2.2.4, process refinement is inherently more a control than an estimation function (unlike the other levels of the JDL-model) and is a part of the system control in Figure 2.1(a). That is true also for perception management, being a part of process refinement. For the design of the control of the perception management, we here suggest two types of control that should be considered: *system* and *individual architecture*.

To understand the meaning of these architectures, we first need to separate the control from the perception resources or services. Even though, e.g., the control of a sensor is likely to be located close to the sensor itself, this is not required.

System architecture refers to the control and behaviour of the whole set of perception resources or services. This could be *centralised*, *decentralised*, or *hierarchical*. In a centralised system architecture, the actions of all resources are contemplated and issued by one single process node (i.e., the logical home of some control process). It will allow the processing node to make the resources or services act coherently, but it scales poorly with many resources and is also vulnerable (since it is only one node that is responsible for the sensing actions).

In a decentralised architecture, many processing nodes divide the responsibility of control among themselves, and, thus, no single node even has complete overview over the control. One node is, typically, in charge of a set of resources on a specific platform. This type of architecture has better scaling properties than centralised and is more flexible (nodes can be replaced dynamically). However, achieving coherence is a challenge and cooperation must be considered (a survey of cooperating mobile robots is provided in [CFK97]).

[20] Interchange of entities between perception resources is called "hand-off".

A hierarchical architecture is a hybrid of centralised and decentralised. It has a centralised superior control node, but no complete control. Instead responsibilities for subtasks are delegated to inferior nodes in the hierarchy.

For individual architectures, by individual we refer to the architecture of the individual processing node (in the case of a centralised system, there is only one). In agent theory (and elsewhere), a number of control architectures for individual nodes have been proposed (some mentioned in [Woo99]). [Ark98] tries to capture the various types of individual control with a spectrum from *reactive* control on the one hand and *deliberative* on the other.

A deliberative control keeps a detailed representation of the states of the environment and itself, and acts primarily on this representation. A control with a reactive control, on the other hand, acts only on immediate percepts. The characteristics of the extremes, pure reactive and deliberative control, are presented in Table 2.2.

Table 2.2. Individual control architectures [Ark98]

Reactive	Deliberative
Representation-free	Dependent on representation
Real-time response	Slow response
Low-level intelligence	High-level intelligence

There is a natural dependence between the individual control architectures and the functional levels of the JDL-model; reactive control, typically, nourishes from the products of Level 0 and Level 1, while deliberative control take action based on the outcomes of Level 2 and Level 3.

2.5 Future studies

As indicated previously, we do not claim that the model or domain map of IA in data fusion systems presented in this chapter is complete. Future work in this field could consider some of the following directions:

Learning and coevolution In environments where the `DFP` does not have a complete model of the process it is observing, it may be useful for it to learn from its experience with the observed process. Learning examples include the behaviour of the observed process in response to occurring events in the environment (machine learning is treated in, e.g., [Mit97, KLM96]).

Learning is especially important where the environment is inhabited by hostile intelligent adversaries. In such cases, also the adversaries may be learning and adapting their behaviour to their percepts. Their adaption imposes constraints on the decision-making for sensing actions of the `DFP`. Important to learning is representation of other agents. *Perceptual state* in [SB01] and *recursive modelling method* in [GD00] are used for that purpose.

The process of agents learning from each other is called *coevolution*. It would be interesting to study how learning and coevolution enter and affect the DFP.

Perception services Managing services instead of resources appears to have some important advantages (presented in Section 2.4.2). However, a complete description of the relationship between resources and services, has yet to be developed. Service properties, such as dependencies between services should also be investigated.

Prioritisation of information acq. tasks Information acquisition tasks are specifications of desired information that the perception manager will have to deal with. Priorities will have to be assigned to tasks to decide which tasks should be handled first. We can envision at least two types of priorities: external (user or system assigned) and internal (resource dependent). External priorities are assigned to tasks by the user or system who needs the information. The answers to such tasks, typically, contribute to the system objectives. This issue is addressed with the *goal-lattice methodology* in [HM99]. Internal priorities arise from the available resources, e.g., a task with high external priority may get a low internal priorities because the available resources cannot satisfy the task. The notions of external and internal priorities and ways of fusing them require further exploration.

Information acquisition tasks Perception management responds to stimuli of the other levels of the DFP, sensing resources and other agents, as explained in Section 2.4.1. It should be further investigated how this stimuli should be used to create information acquisition tasks. Relevant properties of tasks should also be established, including perhaps dependencies between tasks (e.g., hierarchical), cost and deadlines.

Connecting tasks to services Perception management should somehow exploit its available resources to satisfy information acquisition tasks. It is not obvious how this is best done. Relevant to this issue is [BG97] wherein the process of *explication* is described which denotes the transformation from task to utilisation of sensing resources. It should also be explored how resources can be used for treating several tasks simultaneously.

Data mining For the sake of the DFP, *data mining* techniques should be considered. In [HMS01], data mining is defined as "... the analysis of ... observational data sets to find unsuspected relationships and to summarise the data in novel ways that are both understandable and useful to the data owner." We envision the DFP running data mining algorithms on acquired and stored data to create and refine rules and models, supporting the further work of the DFP. This issue has already been discussed thoroughly in [Ste99], but should explicitly enter the model described in this chapter.

Control of data fusion methods One part of Level 4 fusion, process refinement (see Section 2.3), is controlling and, possibly, exchanging methods for fusion of information. The properties of this activity should be investigated.

Management process models Process models of IA (i.e., sensor management) of other works (e.g., [XS02, NN00]) should be considered, and possibly integrated with this study.

2.6 Summary

In this chapter, we wanted to situate the information acquisition part of data fusion systems and highlight its properties. The focus on data fusion systems is a minor restriction since it coincides well with the perception process of the general agent architecture. A comprehensive model, such as the one sketched in this chapter, might facilitate discussions about information acquisition in data fusion systems and describe its potential to aid development and further studies.

We started out by delineating the data fusion process (`DFP`). A commonly used model to describe the functions of the `DFP` is the JDL-model, which is composed of five functions. Four of them refer to the refinement of data and inference of high-level information. The fifth is a meta-controller function, called process refinement, that controls the `DFP` itself.

By further decomposing the process refinement function, we eventually arrived at the part which deals with information acquisition. Many terms have been used in the literature to name this function, but we prefer to call it perception management.

Perception management, situated inside the `DFP`, is stimulated by the results of the other functions of the process, sensor reports, and requests from external users of the `DFP`. Given the stimuli, tasks for information acquisition are created and actions are issued by the perception management. Actions involve the usage of resources, perception resources in particular (a term we use to denote any resource that can be used by perception management for information acquisition). We further noted that it might be useful to decouple the control space of perception management from the hardware of resources and instead express that space in terms of perception services.

The dependence of perception management on the other functions of the data fusion process should be stressed. It is often inevitable that the degree of usefulness of sensing actions in a system is strongly dependent on the ability of the data fusion modules to take advantage of the acquired information. For instance, a sensing action that acquires information (no matter how interesting) that cannot be used efficiently in the data fusion process has little value.

For management of perception resources and services, we proposed some properties that affect the control. We divide the properties in two groups: scope and value. Scope properties restrict the set of acquirable observations, and value properties refer to utilities and cost of using resources.

Managing systems of resources introduces more properties to consider depending on the composition of the set (homogeneous or heterogeneous) and dependencies (e.g., sensor cueing).

Control architectures, both for systems and individuals, were subsequently discussed. System control architectures are typically centralised, decentralised or hierarchical. Using the classification in [Ark98], we find the individual control architecture somewhere in between reactive and deliberative.

Finally, we mentioned some aspects that was left out of this study and that should be addressed in the future.

Chapter 3

Towards large-scale information acquisition in data fusion systems A survey

3.1 Introduction

Information acquisition (IA) is a fundamental activity of efficient decision-making. "Manual" IA, i.e., the process of acquiring information, both initiated and executed by human beings, has been performed for thousands of years. For instance, before engaging in battle, army leaders (previously as well as now) needed information about their opponents to select a suitable strategy. Another example is the IA relevant for establishing a community in a particular site. It was important to evaluate the transportation properties (e.g., landscape and rivers) and defence properties (considering, e.g., if the site is a hill). The sensing resources used were at first typically human labour and tame and trained animals. Later on, we learned to construct tools to enhance our sensing capabilities (e.g., binoculars).

In the recent history of mankind, we have learned to build sophisticated devices, i.e., sensors, to assist with the acquisition of information. Whereas the control of human labour and animals were indirect and resources assumed to possess a lot of autonomy, contemporary sensing devices require explicit control. With the increase in numbers of sensors, improvement of sensor competence, and the demand for timely information, a need for automatic management of the resources has arisen.

Important concepts and methodologies could possibly be learned from different application fields of manual IA. For instance, it would be rewarding to study methodologies for manual information acquisition in, e.g., command and control and land surveying (one activity being *triangulation* [Sha87]). However, here we restrict our attention to efforts for autonomous and semi-autonomous control of sensing resources. Autonomy will become an increasingly essential property of IA in future intelligent systems. One motivation for this is that it will respect the need for rapid and efficient processing of extensive data and information quantities; requirements that could not be met by manual IA.

A system engaged in IA closely resembles an *agent*, i.e., an entity that is situated in some environment which it perceives and acts upon (e.g., [RN95]). As such, agents appear in many shapes either as artificial or as natural entities. A perception process is fundamental for establishing situation awareness (i.e., an understanding of the status of the environment) of *active* artificial agents. The agent is dependent on environmental stimuli per se and is, here, active in the sense that it is capable of actively looking for the information it needs.

Decision-making and *action selection* are two independent subjects that are important for agent perception and, hence, for IA. IA does not just contribute to better decisions (see, e.g., [RN95]), selecting the right sensing actions is itself a decision-making problem. Action selection (or, sometimes, *behaviour selection*) is about selecting actions to pursue some, perhaps, conflicting system goals/objectives.[Hum97] Although, the interpretation of "action" is normally an action that explicitly makes the system pursue its objectives,[1] it could as well concern sensing actions to perform IA.

Resource and *sensor management* are topics that to a high extent are related to IA. Sensor management, especially, considers the control of resources for sensing and, ultimately, acquisition of information.

Furthermore, agent theory, decision theory and sensor management are firmly intertwined with the independent research fields of computer vision and robotics. In the latter fields, as well as the former, the need to model and realize IA is an inherent issue.

If IA was restricted to enumerating all possible sensing actions, evaluating them and selecting the most rewarding ones, then this would not be a problem to discuss. However, for instance, normally

- there is not enough time to evaluate all possible sensing actions;
- there are not enough resources to perform all the sensing actions one would want to;
- one does not always know what information to aquire.

Problems such as these have been addressed in literature in various fields of research, including the aforementioned ones.

The primary goal of this chapter is twofold; first, to describe some different fields that are related to IA (the important function for decision-making) and, second, to promote *large-scale* IA systems. By large-scale system, we mean

> *a system that includes many heterogeneous and distributed sensing resources and that has conflicting objectives and insufficient resources.*

[1] In this chapter, we call such actions *non-sensing actions*.

We also assume that it is used in a "challenging" environment, that is, e.g., both inaccessible and dynamic.[2]

Section 3.2 explains the boundaries of this chapter. Section 3.3 discusses and exemplifies the relevance of IA in a selected set of research fields. Section 3.4 presents two important aspects of IA, facilitation and focus of attention. Section 3.5 describes a useful model of the relationship between the actual environment and the observers internal representation of the same. Section 3.6 briefly describes a taxonomy of the three types of perception activities for IA which we propose. Section 3.7 through 3.9 give examples of literature that belong to each of the three activities. Section 3.10 surveys literature that deals with facilitation of IA and Section 3.11 surveys literature that deals with focus of attention. Section 3.12 tries to characterise large-scale IA. The final sections, Section 3.13 and 3.14, provides a brief summary and conclusion, respectively.

3.2 Extent of survey

This literature survey covers efforts in various research fields, including agent theory, robotics, computer vision, target tracking, decision theory, sensor and resource management. The amount of literature in some of these fields is enormous, and we could, hence, easily lose focus by getting into too much detail (details such as deadlock resolution and properties of utility function). Instead of delving into details of general subjects, if necessary, we will simply refer to relevant literature. We have tried to concentrate on literature that explicitly deal with acquisition of information or support for it. Since the purpose of the study is to pave the way for automatic large-scale IA in real-life environments, we are especially interested in distributed multi-sensor systems operating under various constraints and uncertainty.

The processing of acquired information for situation assessment (including data fusion) is not within the scope of this survey and is thus not explicitly discussed (see, for instance, [HL01, Hal92b, AG92, WL90e] instead).

3.3 Related fields of research

To explore and develop techniques for automatic IA, it is useful to closely study its context and related fields of research. In Section 3.4, we promote three types of activities and two aspects of IA to decompose the subject into parts that can be studied and considered more or less independently. We use this classification as a rough outline for the rest of the chapter.

The following subsections briefly discuss how the concept of IA arises in a few disparate research fields. The efforts in these fields are by no means mutually exclusive. On the contrary, they are to a very high degree intertwined. This fact

[2] Environment and sensing resource properties are discussed in Section 2.2.2 and 2.4.4, respectively.

reflects the high interdisciplinary importance of IA and related techniques. Section 3.3.1 relates IA to agent theory, and Section 3.3.2 discusses the relevance of decision-making. Section 3.3.3 considers the relation between IA and resource and sensor management, Section 3.3.4 gives examples of IA in computer vision and Section 3.3.5 presents IA in robotics. Section 3.3.6 mentions a few techniques that can be used for realizing IA rather than study it explicitly.

3.3.1 In the agent framework

To situate and motivate IA it is rewarding to consider the *agent* metaphor. A unique definition of agent in computer science does not exist, but many researchers agree that an agent is some entity, situated in some environment, capable of perceiving its environment (using sensors) and acting in it (using various actuators).[RN95] This comprehensive definition applies to biological systems (such as mammals) as well as artificial ones (such as mobile robots or complex decision support systems). Figure 3.1 shows a simple agent architecture.

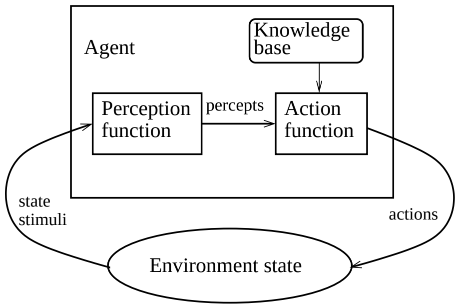


Figure 3.1. A simple agent model

With this agent definition, we are willing to claim that virtually every system with an interest in IA can be embraced by the agent concept. Even decision support systems which have no explicit means of interacting with its environment are embraced by the agent concept since the user of the system controls the agent actuators, and hence, constitutes the lacking action part of the agent.

We here adopt a rather general view of an agent. We do not assume that an agent is a physically delimited entity. It could be physically distributed, but at the same time possess the typical agent properties (i.e., capable of perceiving and acting).

The agent concept has attracted a lot of attention and generated plenty of often interdisciplinary research, spanning, for instance, computer science as well as psychology and ecology. Consequently, knowledge has been generated that is useful in IA (e.g., techniques in decision-making and resource allocation). And, conversely, being an integral part of agent technology, advances in the theory of IA contribute to research in agent theory.

3.3.2 Decision-making and action selection

IA is related to the issue of making optimal decisions in two ways. First, making decisions on sensing actions, e.g., what information to acquire (i.e., what sensing actions to take), may be formalised as decision-theoretical problem. Second, acquisition of information supports decision-making by providing the decision-maker with useful information.[3] The second alternative is probably the most common since it associates the utility of acquirable information with the expected payoff of future non-sensing actions. Thus, it corresponds well to the ordinary decision-theoretical formalism.

An important difference between making sensing actions and other (non-sensing) actions is that rather than making decisions for manipulating the environment in order to achieve objectives, the purpose of IA is (normally) to have as little effect on the environment as possible[4] while acquiring information to support goal-directed decision-making.

In [How66], the ideas of Shannon's *information theory* [Sha48] is extended to a formalisation of the value of acquirable information, i.e., so called *information value theory*. Further, in [RN95], information value theory is used to select a sensing action (if any cost-effective sensing action is conceivable); a step which precedes the step of deciding which non-sensing action to take. A more thorough discussion about decision-theoretic deliberation about sensing actions is provided in [Pea88a]. The computational constraints expressed therein results in a *myopic* policy for sensor control under the assumptions of the viability of a short time horizon for sensor control and that sensor actions are approximately independent.

Decision-making under the name "action selection" has been surveyed in [Pir98, Hum97].

3.3.3 Resource and sensor management

Resource management is the continuous process of allocating, planning, coordinating and scheduling a system's resources (e.g., financial and physical) to meet some objectives, possibly given some constraints on the usage of the resources (this tentative definition is similar to [PCCB97]). Resource management is discussed in, for instance, [Ben83]. The purpose of the before-mentioned reference is business economics, but the ideas are generally applicable. The book describes resource management in three subprocesses: *directional thinking*, *resource allocation* and *resource administration*.

[3]Sometimes the purpose is simply to maintain a sufficiently "correct" state description of the environment.

[4]This is obviously the case since side-effects on the environment, caused by sensing actions, may render the acquired - and thus possibly out-dated information - useless.

Directional thinking is the subprocess of defining and revising the objectives of the system in question. The objectives will typically change with the evolution of the environment and the needs of the user of the system. Directional thinking corresponds to "focus of attention" of IA discussed in Section 3.4.

Given the objectives established by directional thinking, the resource allocation subprocess decides how much system resources to use, and where and when to use them. It seems that there is a symbiosis between directional thinking and resource allocation. Directional thinking directs the resource allocation, but the allocation, in turn, should be able to direct the directional thinking by describing what resources are missing, if any, to satisfy the objectives.

Resource administration deals with planning and control of resources. Resource allocation and administration subprocess both encompass most of IA including the other feature mentioned in Section 3.4, facilitation.

Since resource management is a much wider problem than that of IA, we shall not discuss it any further. However, it is important to bear in mind that IA is a part of resource management, and that resources necessary for IA might be preempted (i.e., made unavailable) by resource management if those resources are needed for system tasks of higher priority.

More directly related to IA is sensor management. Sensor management is a natural subset of resource management[5] and its goal is loosely to "manage, coordinate, and integrate sensor usage to accomplish specific and often dynamic mission objectives."[NN00]

In [BP99a], the authors prescribe two necessary conditions for sensor management to be applicable: (i) sensing resource agility (i.e., that the sensor actually has some degrees of freedom to manage) and (ii) a lack of sensing resources. Furthermore, three important aspects of sensor management implementation are identified. Those are choice of: (i) architecture (i.e., the specification of the location of the management process, e.g., centralised or decentralised); (ii) scheduling technique (e.g., brick-packing); (iii) decision-making technique (deciding which tasks to perform).

There are several instructive surveys in the field of sensor management, including [XS02, NN00, MM94]. Furthermore, [Ben02] includes an overview of sensor management tasks and requirements.

3.3.4 Computer vision

In the field of computer vision, IA is represented by the concepts of *active perception* or *active vision* (when only visual sensors are involved). Active perception is roughly defined in [Baj88] as the active use of sensors for perception (with a focus on the modelling and control strategies for perception). It has been appreciated that some problems in computer vision can be greatly simplified by employment of active perception.[AWB87]

[5] To emphasise this relationship, it is sometimes called sensor resource management.[BP99a]

The ambition in [Rim93] is similar to ours, but restricted to computer vision. In a way similar to this thesis, [Rim93] incorporates the vision (sensing capabilities) into an agent model, to emphasise the importance of the context of the sensing system.

In [BPP99, PPGKB96], the problem of IA for fusion of information is addressed. The term *active fusion* is introduced to describe a system that has a wide range of actions available, including both external actions, such as moving a camera for better views (so called *view planning*), and internal, e.g., activation of image analysis algorithms. We note the resemblance of the extension of active fusion to information fusion with the function of process refinement explained in Chapter 2.

Furthermore, an architecture and control flow of active fusion is presented. It is query-driven and refines a solution to a query iteratively using its active control until it has reached a satisfactory level of confidence (or until no further improvement can be achieved). Applications of active fusion are also implemented based on different techniques for management of uncertainty (probability theory, Dempster-Shafer evidential theory and possibility theory) and compared.

[TAT95] is a survey about *sensor planning* in computer vision. The goal of sensor planning is stated as that of generating appropriate sensor configurations based on a priori information (e.g., knowledge of the current task or query and models of observed objects and available sensors). The survey identifies three distinct problem types of sensor planning for computer vision: *object feature detection*, *model-based object recognition and localisation*, and *scene reconstruction*. The first type corresponds to problems that require a vision sensor to make features of an object (with known identity and pose), e.g., visible, in-focus or magnified, according to the requirements of the task. In contrast to problems belonging to the first type, in problems of the second type the identities and poses of objects are unknown and should be estimated. For a problem of the third type "a model of the scene is incrementally built by successively sensing the unknown world from effective sensor configurations using the information acquired about the world to this point." The focus of the survey in [TAT95] is on the first type.

3.3.5 Robotics

Robotics represents the physical incarnation of agents, and, thus, naturally inherits the need for IA. Actually, the need for perception and related challenges are accentuated in robotics since its relationship with a real physical environment is inherent.[6] Mobile robots use sensing resources, such as cameras, sonars, laser scanners, primarily to avoid obstacles, detect relevant objects and to map its surroundings.

Information gathering is the name used to denote information acquisition in [Hag90]. A theory for information gathering is presented which entails four basic principles: task-direction, uncertainty, computational and representational limitations. Task-direction acknowledges that the origin of information are the tasks the

[6]A real physical environment normally has the most difficult environment properties described in Section 2.2.2.

system has to perform to fulfil its objectives. In our work, we discuss this issue briefly in Section 2.4.1. Uncertainty in information is an inherent feature of IA. We discuss this issue in Section 2.2.3. Computational limitations concerns the fact that the amount of resources available for reasoning about actions may be limited. In this thesis, we do not discuss that issue explicitly. Finally, representational limitations means that it is neither feasible nor desirable to completely represent the environment and observed information. We both support and discuss this claim in Section 3.5 of this chapter.

An interesting distinction is made in [Hag90] between the *environment state space* and the *information state space*. Non-sensing actions operate on the environment space and sensing actions on the information state space. This distinction is introduced to facilitate analogous planning for both types of actions.

The method for IA presented in [Hag90] is based on Bayesian decision theory, meaning that information is evaluated with respect to future system actions (i.e., non-sensing actions). The cost of sensing actions is considered to decide if any sensing actions are feasible. A *batch* solution, which considers sequences of actions, selects the sequence that maximises observation payoff. This approach appears to be most suitable for static environments. Another approach is the *sequential* one, which considers the contribution of a single sensing action.

In [MDW94], a complete data fusion process for decentralised multi-sensor systems is presented. It is applied to the common problem of mobile robot navigation. Environment features are observed in order to localise the robot platform and sensors coordinate using a distributed negotiation algorithm.

3.3.6 Indirect fields of research

There are many techniques available that are not directly related to IA but which are essential for efficient implementation thereof. Such techniques include *scheduling*, *planning*, and various kinds of *protocols for coordination* of distributed sensors. We do not discuss such entirely independent techniques explicitly in this thesis. However, they are discussed extensively in other literature.

3.4 Salient aspects of information acquisition

We say that the system skill of IA is realized through one or, more likely, a number of *perception activities*. A perception activity is, generally speaking, a process that provides the system that needs the information with measurements and observations. A taxonomy of perception activities is presented in Section 3.6.

In the works we survey in this chapter, two salient features of perception activities emerge: *facilitation* and *focus of attention*. Facilitation concerns making observations possible and includes respecting constraints of the perception process, e.g., to minimise energy consumption [PH03] or to make sure that sensors do not interfere with each other. It might also involve altering the observation scope (scope properties of resources are suggested in Section 2.4.3).

Focus of attention involves deciding *what* information is relevant for overall system objectives rather than deciding *how* to acquire information in the best way.

In some works, one or both of these features are very subtle. Contrarily, in works that solely deal with facilitation or focus of attention, the type of perception is secondary or irrelevant. Often perception activities include at least the aspect of facilitation, whereas the focus of attention is often assumed to be more or less fixed.

Section 3.5 provides a model of the relationship between the environment that is observed and the representation of the environment that the system maintains. We find this model to be useful for the further discussion about perception activities. Section 3.6 classifies perception activities depending on the way they contribute to the aforementioned environment representation. Literature amenable to this classification is surveyed in Section 3.7 through 3.9. Literature representative for the IA aspect of facilitation are discussed in Section 3.10 and focus of attention in Section 3.11.

3.5 In the eye of the beholder

For the continuing discussion about perception activities, we need to provide a context for IA. We start with two essential components: the *environment* (sometimes referred to as *workspace* or *world*) and the *observer* (i.e., observing agent). The environment is the source of the information that the observer requires for a successful operation. The observer may be a complex and distributed entity, composed of many coordinated sub-components (i.e., they are coordinated in the sense that they are able to and interested in exchanging information). The observer is capable of perceiving the state and take actions (in this chapter we are mainly interested in those actions that support the IA process).

As depicted in Figure 3.2, we distinguish between the *environment state* as it appears to an observer and the *control processes* that forces it to evolve.

The control processes that affects the evolution of the environment state appears in many shapes. Some, such as those conforming the state to be consistent with the physical laws of nature, are disembodied and permeates the entire world, while others originate from discriminable entities that are part of the environment. Typically, entities that harbour such control processes are biological beings or machines. The latter kind of control processes, typically, has a pretty well defined local effect on the world, but the environment as a whole is more likely to express some emergent behaviour, dependent on both the interactions of the control processes and the evolving state.

In Figure 3.2, the environment states have been given dissimilar cloud-like shapes, to emphasise that the environment evolves over time. What the figure

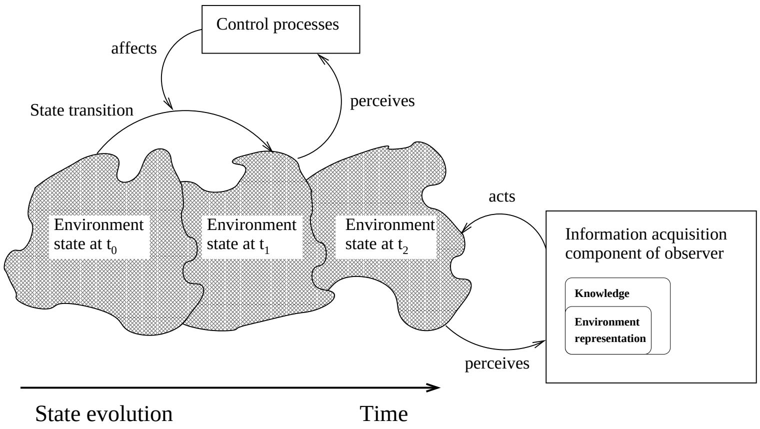


Figure 3.2. The environment evolves due to control process that interact. An information acquisition system has the ability to sense the state of the environment using sensors and may infer properties of the governing control processes.

fails to capture is that the evolution is continuous in general, rather than discrete as it may appear here.

We make a clear distinction between the true environment state (i.e., the ground truth) and the *environment representation* of the observer. We consider the environment state to be a complex entity that can be observed. We do not attempt to parameterise the environment and characterise it with variables. The reason for this is the, in general, continuous nature of the environment (e.g., a physical environment). The information content of the environment could be quantified in variables, but there is generally no unique way to accomplish that. In other words, it is not the responsibility of the environment to interpret itself, it is up to the observer. For instance, aspects of a physical environment may contribute to different variables such as states of molecules or states of aggregates of molecules. A specification of which composites of molecules should be assigned a higher level interpretation of the state should not be included in the environment, it should be up to actors and observers in the environment to make such a distinction.

The *environment representation* (ER), on the other hand, is typically composed of a jumble of discrete and continuous variables and hypotheses, representing the knowledge of the observer. The information in the ER is rarely fully reliable and is contaminated with uncertainties, expressed in probability functions over variables and hypotheses. The ER only contains information that is *relevant* to the observer (i.e., information that it regards relevant for its selection and execution of actions), and, in general, it only expresses a belief about a small part of the environment state. The limited view of the environment of the observer is imposed by its current objectives (or goals). In effect, the objectives (by affecting the focus of attention aspect) decide the structure of the ER, the *environment model* (EM). Apart from limiting the extent of the ER, objectives also greatly affect the selection of actions for IA.

Another salient feature of the ER is that it, in contrast to the environment state, is a composition of information[7] of varying age. Thus, whereas the environment is innately "up-to-date", the ER may be that just partially, or more likely, not at all.

Furthermore, in addition to the ER, the observer may also maintain information about itself, its *internal state representation* (ISR). The ER and ISR jointly constitute the knowledge of the observer. Some information in the ISR is quite reliable (e.g., battery power, if the agent has no reason to disbelieve its internal sensing,[8] and other less reliable (e.g., the agent's exact location in the environment).[9] In this thesis, however, we mainly focus on the observing agent's ER.

Finally, the ER may not only express the observer's belief about environment state, but also the state of the control processes that influence the evolution of the environment state.

3.6 A taxonomy of perception activities

There are basically two types of information that an IA process would like to obtain: properties of the environment state itself, and properties of the control processes that affect the evolution of the environment. Often, environmental properties may be acquired instantaneously, using suitable sensing resources. Properties of the driving control processes, are even more difficult to estimate, and generally have to be inferred by observing a temporal sequence of environment states.

For both types of information, we might want to (1) encompass all relevant information by *incorporating* missing (i.e., yet undetected information) into the ER; (2) *monitor* the subset of the environment that has generated interesting information in the past; (3) *discern* a more detailed or certain understanding about some interesting part of the environment. Note that none of the literature surveyed here addresses explicitly deals with the acquisition of information about control processes. However, the products of the efforts surveyed may be used to infer the state of control processes.

Our classification is model-based rather than technique-based, meaning that we categorise acquisition activities depending on the type of information they provide, rather than the techniques they employ. All types of IA might involve techniques such as management of uncertainty (which motivates use of Dempster-Shafer theory,[Sha76] Bayesian inference [Pea88b] and fuzzy set theory,[Zim91] etc) and optimisation (which motivates use of mathematical programming techniques, evolutionary algorithms, etc).

[7] A "piece" of impression of the environment state.

[8] The process of measuring internal state is sometimes referred to as *proprioception*.[RN95]

[9] Compare to the important issue of *localisation* in mobile robotics.

We propose the following taxonomy of activities for IA, based on the model of the relationship between the environment and internal state explained in Section 3.5:

Incorporation The contents of the ER should change when phenomena, events or properties, of interest of the environment are detected. It could also be that the system loses interest in some part of the environment (perhaps due to altered mission objectives) and decreases the extent of its ER.

Monitoring Phenomena, already incorporated into the ER, might evolve over time and, if so, must be monitored. E.g., target tracking is a monitor activity that seeks to update position estimates of incorporated objects.

Discerning Sometimes it is necessary to identify more details of some entity or phenomenon in the ER perhaps to refute or confirm a hypothesis.

Note that even though aspects of IA are conceptually disparate, in applications the distinction is not so clear. For instance, monitoring is in some sense a continual incorporation activity that performs some administrative work to maintain tracks using a priori information for detection. Some works are composed of both an incorporation part and monitoring part. Furthermore, monitoring may result in the unintentional acquisition of additional information that contributes to discernment of the ER.[10] Conversely, a discernment activity, by identifying the true type of an object, might result in a performance improvement in a monitoring activity, if a more precise dynamic model of the object can be selected.

For comprehensive and large-scale IA systems, *hierarchical layering* is a useful architectural design to manage the normally immense complexity of such systems. In a hierarchically layered control system, a high-level node (nodes encompassing perception activities in our case) typically has a long planning horizon and a broad (possibly global) responsibility, and is capable of giving coarse orders to lower level nodes. Correspondingly, lower-level nodes have short time intervals for selecting actions, local responsibility, and has possibly direct control of sensors. Hierarchical layering is also useful for managing the complexity of information in the ER.

Hierarchical layering for control and environment model representation are used in, e.g., *the RCS system*,[Alb99] *the data fusion and resource management tree architecture*,[BS01] and *the logical sensor/actuator* framework.[BG97]

The following three sections will give examples on literature related to each of the aforementioned perception activities.

3.7 Incorporation

The perception activities surveyed in this section detect and incorporate "new" information into the consciousness of the observing agent, i.e., making the agent

[10] Information acquired through monitoring might, e.g., be related to some available a priori information that infer further information about a tracked object.

aware of the (hypothesised) existence of relevant environment phenomena. Typically, this involves detecting interesting entities and instantiating their estimated properties in the ER. What is interesting is dictated by the observer's objectives and the structure of the new information is given by the EM. A phenomenon is usually an object or an event.

The literature we study in this section mainly deals with applications for object detection. In practise, detection is uncertain and one can rarely say for sure that an interesting object has been detected, rather one must assign some confidence to an alleged detection.

In [Pen98], it is assumed that a hypothesis about the approximate whereabouts of a stationary target is available. A set of sensors is managed to improve their joint probability of detecting the target. Fusion of detection probabilities is performed using the so called OR-rule[11]

$$P_d(D|T = x, r_1, \ldots, r_N) = 1 - \prod_{k=1}^{N} \big(1 - P_d(D|T = x, r_k)\big),$$

where $P_d(D|T = x, r_i)$ is the probability of sensor i, at position r_i, detecting a target at position x.

Since the exact position of the target is unknown, the probability of detection is the expected detection probability over all positions using the a priori hypothesis of the target position.

In the work of [Pen98], the individual detection probabilities, $P_d(D|T = x, r_k)$, are modelled with approximate Gaussian distributions. If the target position hypothesis, also represented by a Gaussian distribution, is peaked, then the sensors tend to position themselves close to the peak. But if the hypothesis is more vague, they tend to spread to get better coverage.

The process of positioning the sensors is proposed as a hill-climbing search where the initial positions is a random sample of the a priori target position distribution and where the sensors simultaneously try to increase the joint detection probability. The off-line search terminates when the increase in detection probabilities is below some threshold or after a predefined maximum number of iterations. The sensor positions that result from the search are the initial sensor positions where the sensors are first deployed. A target is considered to have been detected when one sensor has reported detection a fixed number of times.

Subsequently, sensors start to send observations or report lack of observations. A new set of sensor positions are sampled from the now updated target position distribution and the sensors are redeployed.

The work also discusses how this approach can be used on a mobile target. However, this work does not address time delays and similar issues that are associated with relocation of sensors and which are critical if the target is moving.

[11] According to the OR-rule, the fused probability of detection of a target in position x is one minus the probability of no sensor detecting it, given that the target is actually in position that position.

In [McC98, HM97], the objective is to select sensors with different properties in order to efficiently detect events. The event detection is exemplified by an application of part assembly using a robotic manipulator. The assembly procedure is described as a sequence of events. For a successful assembly, all events have to be detected by the sensing processes available to the robotic system. A detected event marks the termination of the previous assembly motion and the initiation of the next.

Event detection is, in the example, performed by three sensing processes (or sensing services, using the terminology of Section 2.4.2) that utilises position and force sensors. The sensing processes have the same output type, i.e., a tuple including the detected event type and a quantified confidence of the detection, but the running times and confidence levels differ. A stochastic dynamic programming approach is selected to pre-calculate the order which the sensors should be activated for every state of the assembly. During the assembly process, the event detection confidence of the first selected sensing process is insufficient is too low, the next sensing process will be consulted, and so on until either a sufficient confidence level has been reached (by successive fusion of the results of the sensing processes) or the sensing processes have all been exhausted.

Although the example application given is that of robotic assembly, the author [McC98] argues that the discrete event framework can be used recursively in a hierarchy to cover control from the top-level of the factory itself down to individual work stations. In such a hierarchical discrete event control system, a completed assembly on the robot level could be interpreted as an event in a higher level.

Even though the event detection problem here is applied to measuring the state of the system itself, there should be no difficulties transferring it to a context where the discrete events refer to actions of some observed process in the environment.

The aim of [CGH96] is to find observation positions for mobile sensors, where they both are likely to detect interesting objects and where they are unlikely to be observed themselves. The proposed solution, which is based on decision theory, consists of two parts: planning of trajectory and selection of camera parameters (i.e., pan/tilt angles). Sensors cooperate by observing complementing areas, but no fusion of acquired information is performed. A model of the whereabouts (over time) of the interesting objects is assumed.

3.8 Monitoring

A common problem is to monitor some part of a "real" and complex environment that evolves over time and that can only be interpreted through noisy observations. Works that deal with this problem, i.e., that of estimating the state of the interesting subset of the environment (called *system state* in the literature), typically formulates the problem as an optimisation of some objective function (with respect to various constraints on the use of the sensors, i.e., facilitation constraints) that corresponds to the expected quality of the monitoring by controlling sensor parameters accordingly. The objective function used is dependent on the type of the monitoring technique in use.

In principle, the preferred sensing action is the one that optimises the expected quality of the state estimation of the following observation. Quite often an optimal solution is intractable and approximative heuristic techniques are proposed.

While focusing on the accuracies of predicted measurements, works that perform monitoring rarely consider the value of the information they acquire. If sensing resources are shared (being useful also for other purposes than IA), then also the relevance of the obtained tracking accuracies for high-level goals must be considered.

In the *recursive filter* approach to this problem, observations are processed sequentially to produce an up-to-date *probability density function* (pdf) over possible system states at discrete time steps. The procedure of the recursive filter is performed repeatedly. Each step comprises two stages: *prediction* and *update*.[AMGC02]

In the prediction stage, a model of the evolution of the interesting part of the environment, the so called *system model*, is used to predict the pdf at the time of the next observation. The system model is a function of both the current state of the system and a noise component, the *process noise*.

In the update stage, a *measurement model* is used together with the latest observations to update the predicted pdf. The measurement model is a description of how the sensor output is dependent on system state and what uncertainty is attached to it.

The system and measurement models must be supplied somehow by the designer of the monitoring system and reflect the monitoring system's a priori knowledge of the environment it is observing.

Updates can be performed using Bayes' theorem,

$$p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})}, \tag{3.1}$$

where the normalising denominator is

$$p(z_k|z_{1:k-1}) = \int p(z_k|x_k)p(x_k|z_{1:k-1})dx_k \tag{3.2}$$

In the equations above, x_k is the system state at the time of observation k, and z_k the observation measurement itself. In Equation 3.1, $p(x_k|z_{1:k-1})$ expresses the predicted pdf from the previous time-step, $p(x_{k-1}|z_{1:k-1})$, estimated with the system model $p(x_k|x_{k-1})$ by marginalising over x_{k-1},

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1}. \tag{3.3}$$

This formulation of the Bayesian recursive filter is sound but unfortunately unpractical in the general case. However, practical solutions are available under the assumption of various simplifications or restrictions.

The *Kalman filter* is a recursive filter that is the optimal solution under its restrictions. This technique is applicable if the process and measurement noise variables are governed by known Gaussian distributions with zero means, and both the system and measurement models are linear in the state and noise variables.

A frequently referenced work, that has inspired many succeeding works, is [Nas77] in which sensors are allocated to track moving targets. The solution is reached using linear programming minimising the cost (i.e., in this case, the expected measurement errors expressed in properties of the Kalman filter) of possible sensor-to-target allocations.

In [Ben00], the problem is to select a *measurement policy* for some time period for tracking a single target using a Kalman filter. The policy dictates which of the available sensors should be active at what time. The objective function that is constructed is a weighted linear combination of the cost of using sensors and the expected state prediction accuracy.

Also relying on a Kalman filter, [KP98] selects (i.e., activates) the sensors that are expected to achieve at worst some desired maximum state covariance while minimising the computational load on the tracking system (i.e., by selecting as few sensors as possible). The authors call this approach to multisensor management *covariance control*. Actually, three separate objective functions for the covariance control algorithm are proposed and compared to a reference algorithm that always uses all available resources.

To mitigate the problem of linear state evolution required by Kalman filtering, the interacting multiple model Kalman filter (IMMKF) has been developed. Using several Kalman filters (one for each state evolution model), different kinds of evolution can be tracked. In [SK98], e.g., one filter tracks uniform motion and another turning motion. The estimation of every filters is weighted, with a probability value which the system has assigned to the particular model, and combined into an "expected" state estimate, $\hat{\mathbf{X}}(k|k)$.

$$\hat{\mathbf{X}}(k|k) = \sum_j \mu_j \hat{\mathbf{X}}_\mathbf{j}(k|k),$$

where $\hat{\mathbf{X}}(k|k)$ is the updated and combined state estimate at time k, $\hat{\mathbf{X}}_j(k|k)$ is the updated state estimate of filter model j and μ_j is the probability of model j.

To evaluate various senor-to-target allocations, the system evaluates their *expected discrimination gain*. This is the information theoretic Kullback-Leibler (KL) measure[12] of the state estimation density making an observation compared to not making any observation at all (just predicting). The sensor selection is finally solved by formulating it as a linear programming problem with a constraint on the maximum number of targets tracked during a time step.

The work in [SK98] is extended in [DN01] to be performed in a distributed manner to facilitate robustness of the tracking system. Inspired by the game theoretic concept of *coalition formation*, the authors make sensors form groups (the set of

[12] Also called cross-entropy.

groups is a partition, denoted p, of the sensor set), where each group tracks the same targets and fuse their measurements.

The desired result is the assignment of sensor coalitions to targets such that the total measurement utility, $v(p)$, of the sensors is maximised. The optimal solution is formalised as a maximisation over the all partitions where $v(p)$ for every p is the solution of a linear program. The problem is intractable already for small numbers of sensors and targets, and a greedy heuristic, involving sequentially assigning coalitions that are expected to measure targets beneficially, is employed to lighten the computational burden.

This work discusses both a centralised and a distributed algorithm. The centralised is roughly described above. In the decentralised one, each sensor node calculates a preferred *local decision* of which coalitions should track which target based on local information and received estimates from the other sensors. The local decisions of all sensors are shared among the sensors and these decisions are combined in every sensor node to create a final, and coherent, sensor to target assignment.

The *particle filter* approach is not optimal and is computationally more demanding than Kalman filtering, but has the important advantage that it relaxes the linear and Gaussian distribution requirements of Kalman filtering. This advantage and the rapid increase of computational resources has recently made the interest in particle filtering blossom.

The particle filter approximates the $p(x_k|z_{1:k})$ density in Equation 3.1 using a set of particles, i.e., weighted samples of the approximated distribution. The accuracy of the approximation can flexibly be selected by varying the number of particles. An increase in the number of particles brings the approximation increasingly closer to the density function $p(x_k|z_{1:k})$ that is approximated.

In [DVAD02], the problem of selecting one sensor from a set of sensors to observe (i.e., measure) a target is considered. The best sensor to select is the one that gives the best KL measure for the current time step between the expected updated density ("expected" since the measurement obtained from a selected sensor can typically not be known in advance) and the predicted density. This work relies on a particle filter based tracker that provides an approximate description of the updated pdf, $p(x_{t-1}|y_{1:t-1})$. The subsequent process of finding the sensor that is expected to give the best KL measure involves a sequence of Monte Carlo samplings from the particle set.

A particle filter is also used in [KKI03] to maintain the target state pdf. For the sensor management part of the work, which involves selecting a sensing action for the current time step, however, the objective function for sensor control is based on the *Rényi information divergence* measure (also known as *alpha-divergence*), denoted $D_\alpha(f_1\|f_0)$,

$$D_\alpha(f_1\|f_0) = \frac{1}{\alpha-1}\ln\int f_1^\alpha(x)f_0^{1-\alpha}(x)dx$$

Here f_1 and f_0 are two pdf:s to compare, f_0 typically being the predicted density of the target state and f_1 the expected updated density (for some sensor action).

The Rényi information divergence measure is a generalisation of the KL measure, and equals the KL measure when α approaches one. The authors, using this information measure, arrive at fairly simple objective function compared to the one produced in [DVAD02].

Just as [KKI03], [Mah03] addresses the issue of selecting sensing actions to track an unknown number of targets. The information theoretic objective function developed, however, is somewhat different.

In [HKBS02], sensor management for tracking of a single target is described. This work considers the time at which sensor should be deployed not to lose the track, how many and where should the sensors be placed. It also considers which already deployed sensors can be of further use if only a limited number of sensors can be used at the same time. For this work, the activation of sensing actions is primarily driven by the expected development of the *Fisher information matrix* which prescribes optimal performance of the current sensor configurations.

For more on particle filtering see [AMGC02, Ber99]. The basics of Kalman filtering is thoroughly explained in [BSF88].

A common problem to most of the sensor selection algorithms surveyed here is that of time complexity as the number of possible sensor sets is 2^{N_s}, where N_s is the number of sensors.

3.9 Discerning

For applications that senses a real physical environment, the corresponding ER is most likely vague and uncertain, e.g., the information of the current state being represented by a probability function over different hypotheses. Discernment activities, such as those surveyed in this section, address the problem of making the ER more clear, e.g., by discovering values of yet un-sensed object properties or by improving the estimation (i.e., decreasing uncertainty) of variable values.

Classifying a known number of unknown objects within some time interval is the problem in [Cas97]. The classes of objects are further assumed not to vary over time. The sensors used report a classification and the estimated quality of the classification. The classification reports are used to update the belief of the true class of each object (using Bayes' rule) during the time interval. At the end of the time interval, the final classification decides the performance of the classification system.

Sensors have multiple sensing modes (typically of different quality and cost). Facilitation constraints are put on the sensors in that their combined usage cost should not exceed a certain level. A *decision rule* is desired which for every discrete time step prescribes what sensor modes and what objects to observe. A stochastic dynamic programming approach is applied. The usage cost constraint is relaxed to mitigate the resulting computational complexity. It appears that the approach presented in [Cas97] is most suitable when the observed scene is static and that the observation scope of the sensors always include the objects.

[CNR02, CNR01] describes a multi-target classification problem containing a set of targets and a set of sensors. Each target is assumed to belong to one of m classes and each target has n measurable attributes. Every class i has a fuzzy set membership function for every attribute k, $f_i^k(m^k)$, which for every sensor measurement, m^k, states to what degree (from zero to one) a target, that was sensed, belongs to that class. Attributes are rated off-line according to their potential to discriminate between classes. The attribute rating is achieved using a metric called Separation Degree that compares how well of pair of membership functions are separated. The IA idea is to first acquire information about attributes that are more likely to isolate the right class of the target. The expected benefit of this approach is to swiftly identify targets correctly, which is fruitful in real-time recognition problems. Sensing actions, in this case sensor selection and mode selection, are given by the attribute selection. It also considers how contextual information, such as environment properties, target orientation, signal to noise ratio, etc, affects classification. If such contextual information is available, possibly from use of some exogenous sensors, a more sensitive selection of sensor modes can be made.

In [Lee99], sensors (possibly heterogeneous) are modelled to yield as a unified output a tuple including an estimate of an environment feature and a corresponding uncertainty measure. The method for IA proposed performs a search in the parameter space by iteratively updating the controllable parameter vector of the system, p, until an improvement of system performance can no longer be expected. The parameter update is the result of a combination of multiple objectives. The update should respect parameter and system constraints (i.e., to facilitate observations), maintain or obtain an acceptable measurement performance (to the extent this can be estimated from parameter selection), and minimise sensing costs (the cost expressed in, e.g., time or energy of altering p).

In the field of computer vision, *view planning* is a typical discernment activity. The objective is to achieve a classification of some observed object or objects by repeatedly changing camera parameters (typically position and direction) until the probability of correct classification is satisfactorily high. An example is given in [KS00] where an objective function, expressed in camera parameters, is proposed. The objective function takes both opportunities and costs of parameter selections into account.

Another example is [BPP99] which uses the eigenspace object recognition method. The action space contains movements of the camera. For comparison, the uncertainty of the classification results has been modelled by probability theory, possibility theory and Dempster-Shafer theory of evidence. In all approaches, the camera movement $\Delta\Psi$ that reduces an entropy reduction based utility function the most is selected. In the probabilistic view planning, the utility function looks like this:

$$u(\Delta\Psi) = \sum_{o_{i,j}} P(o_{i,j}) \Delta H(\Delta\Psi; o_{i,j}),$$

where $P(o_{i,j})$ is the probability of observing object i with orientation j, and $\Delta H(\Delta\Psi)$ is the expected (information theoretic) entropy reduction by selecting movement $\Delta\Psi$.

The usefulness of the three approaches is justified by comparing their results to the results of an algorithm that selects random camera movements. From their experiments the authors conclude that the probabilistic approach is the most beneficial in their application.

A more elaborate example of view planning along with an extensive survey of object recognition systems is presented in the PhD thesis [Roy00]. The known objects of the system are represented with *aspect graphs.* Each node in the graph represent an aspect of the object, i.e., a set of views of the object which appear identical with respect to a specific set of perceptible features. Edges between nodes represent that the aspects are adjacent from the observer's point of view. Thus, the aspect graph might be thought of as a representation of an object expressed in the capabilities of a particular observer.

In brief, the view planning initially extracts the features from an initial, random, view. The acquired features indicate probabilistically the most likely type of aspect (or class) observed. However, an aspect class is most likely shared by several known objects and the related uncertainty is expressed in a probability function on candidate objects. Based on the probability function, the best reconfiguration (in this case, the best move) of the camera is calculated.

We consider works which involve selecting information heterogeneous sensors (see Section 2.4.4), e.g., sonar, camera, etc. to acquire different types of information to fall into the discernment category. However, we are not aware of any such works in the literature.

3.10 Facilitation

Sensing resources may require many types of techniques to facilitate observations; techniques that are largely independent of what the application is. In fact, some processes are purely concerned with facilitation and do not directly affect the ER.

Admittedly, the facilitation concept is somewhat indistinct. In many circumstances, facilitation is inherent of perception activities. E.g., the control process of relocating sensor platforms is an example of a well integrated facilitation aspect of a perception activity. Another example are the requirements on sensors that often appear as equality or inequality constraints in solutions to sensor management problems. Sometimes, however, as noted facilitation activities can be separated from perception activities and treated independently.

Here, facilitation is considered to be techniques that explicitly support acquisition of information, even though others might also be critical for successful practical IA (e.g., securely encrypted communication in a distributed sensor network).

We suggest a classification of two types of facilitation processes: *resource constraint management* and *scope management.* Resource constraint management deals with constraints that belong to the usage of sensing resources. For instance, consider the resource property *availability* (explained in Section 2.4.3). Resources might become unavailable due to interference with each other. We consider avoiding interference to be an act of constraint management. More commonly treated issues are management of energy consumption and limited transmission rate. Hence, resource constraint management alters the set of possible sensing actions and their utilities, but it does not directly decide the sensing action.

Scope management concerns beneficially altering the conditions for perception. Examples include relocation of sensor platforms,[CGH96] illuminating a scene for image acquisition,[TAT95] dynamical formation of sensor resources (i.e., forming abstract sensors) for fusion of target position estimates,[DN01] etc. Note that scope management, just like resource constraint management changes the utilities of sensing actions.

3.10.1 Resource constraint management

Sometimes, the sensor constraints rather than the type and accuracy of information to acquire is the focus.

In research on wireless sensor networks, respecting battery energy consumption is crucial. [PH03] addresses this facilitation issue (Section 3.4) trying to keep the sensor network "alive" as long as possible while keeping the detection performance of the sensor network above some threshold. At every instance of time, a subset of the sensors in the network are active and send information. The authors express their sensor scheduling problem as a generalised maximum flow problem and solve it using linear programming.

An integration activity for detection through sensor placement is facilitated in [KSRI02]. The problem to be solved is that of minimising the vulnerability of the sensor set with respect to an adversary capable of destroying some of the sensors. In a game theoretic manner, the adversary is assumed to be rational (i.e., acting optimally), and the sensor placement strategy is selected that minimises the loss in case the adversary engage in an (optimal) attack on the sensors.

[KK03] has a strong element of facilitation for a monitoring activity. The authors propose an algorithm to find a policy which switches between an active (and expensive) sensor and a passive (and cheap) sensor to minimise the joint cost of measurement errors and usage of active sensors.

3.10.2 Scope management

The problem of planning paths for a set of UAVs (unmanned airborne vehicles) to make observations, i.e., to actively alter the observation scope, at some pre-specified locations is addressed in [Sol99]. The problem is similar to the well known and NP-Complete TSP (Travelling-salesman problem), but involves path planning for several salesmen, i.e., UAVs. To mitigate the complexity issue, this work formulates the problem as a search using a genetic algorithm.

[BVA03] is another work for UAV-based sensors (or, more generally, mobile) that focuses on the coordination of the sensor platform rather than on the type of information to acquire. Sensing actions for UAVs are two-dimensional, one component being selecting direction of motion and the other selecting which subset of targets to track. The solution proposed to this problem is called *coevolutionary perception based reinforcement learning*. The solution consists basically of a modified Q-learning algorithm.[13] This particular Q-learning is called coevolutionary since two Q-functions learn at the same time, i.e., they coevolve during the unsupervised training. The state space, considered in the learning algorithm, is here discretised by expressing the input variables in terms of fuzzy labels. Furthermore, the Q-functions are approximated by fuzzy rules.

In a survey of sensor planning in computer vision,[TAT95] an example of scope management is that of illuminating a scene before acquiring an image.

3.11 Focus of attention

While the aim of the perception activities is an updated ER with correct and current information, the purpose of focus of attention is to decide which part of the ER (or which activities) to prioritise and to restructure the EM when necessary (i.e., decide what kind of information the ER can be filled with).

To see an example where focus of attention is lacking, consider a target tracking application where a number of sensors track some targets. Frequently, it is merely the expected accuracy of the sensor measurements that is decisive, the usefulness of the acquired information is rarely considered. For large-scale IA, where many information needs compete for resources it is essential to evaluate candidate sensing actions with respect to the usefulness of their expected outcomes (see [How66] for a discussion about this matter).

In [HM99, McI98], the idea of *goal lattices* is introduced to consider the usefulness of information. Mission goals (i.e., system objectives) and subgoals are hierarchically ordered and are members of a lattice, i.e., a partially ordered set $P = (X, \geq)$, where X is a set of goals (or corresponding tasks) and $\geq$ a partial order relation. To fulfil the requirements of a lattice, for every pair of members of the set exists a least upper bound and greatest lower bound. Here the relation reflects whether a pair of members are goal and subgoal, respectively. If for any two members of the lattice, x_i and x_j, $x_j \geq x_i$, x_i is "included" as a subgoal in x_j. Conversely, if $x_i \geq x_j$, x_i is "including" x_j. In other words, if $x_j \geq x_i$, performing task x_i contributes to the completion of task x_j. In this case, x_i is considered to be a more concrete task and x_j more abstract.

The goal lattice construct enforces a prioritisation of sensing actions (the most concrete goals). In a lattice, there exists a unique top element, i.e., if that element is member x_i of the lattice, there exists no x_j such that $x_j \geq x_i$. If the value of the top goal (i.e., relevance or priority if you prefer) is one, then the values of its included

[13] Q-learning and other types of reinforcement learning are surveyed in [KLM96].

subgoal can be determined by apportioning the unit value of the top goal to all its subgoals. For any goal in the lattice, its value is calculated as a the weighted sum of its including goals (i.e., the more abstract goals that it supports).

A lattice can be visualised in a *Hasse diagram* as in Figure 3.3 (figure redrawn from [McI98]). The apportioning of value from the top node down to the bottom nodes, yields a prioritisation of sensing goals. In the figure, using sensing resources for identification get the highest priority.

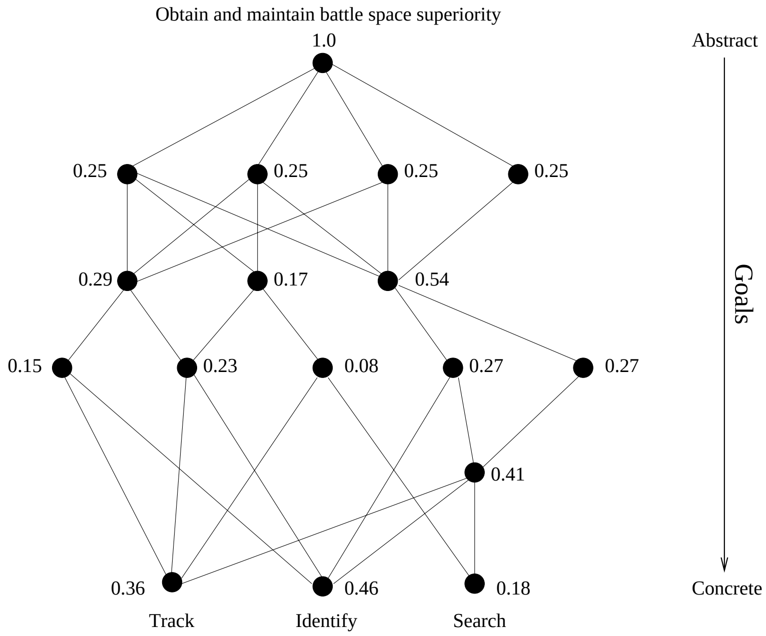


Figure 3.3. An example of a goal-lattice from [McI98]. Towards the top of the nodes in the Hasse diagram are abstract goal, whereas the bottom nodes represent goals which can be treated with resources directly.

The focus of attention issue is also addressed in [LRC95]. There, the management of sensor resources are divided in two steps: prioritisation of tasks, and assignment of sensors to tasks. The prioritisation of sensing tasks (track, search, identify) is realized using fuzzy decision trees (crafted from expert knowledge in surveillance systems design). Using information about the expected sensor performance, sensors are subsequently assigned to the prioritised tasks.

3.12 Large-scale information acquisition

With the apparently everlasting increase in the number of available sensing resources, also the demands on sensor systems will increase. Likely initial application fields for IA are command and control (for battle field situation assessment), production and power plants (monitoring and fault detection), property surveillance (intruder detection). For the future, the proliferation of intelligent and networked mobile devices (such as mobile phones, PDAs, and "wearable" computers) and stationary counterparts (e.g., networked components of household machines) suggests strengthened interest in large-scale IA. However, to successfully manage the resources and enjoy the anticipated advantages, new advanced techniques must be developed.

To realize large-scale IA, e.g., to develop support for a comprehensive decision-making system, we need to be able to manage perception activities and be aware of and refine their inherent aspects (Section 3.4). To initiate and maintain the environment representation (ER) (Section 3.5) it is likely that perception activities of all three mentioned types (Section 3.6), i.e., incorporation, monitoring, and discerning, must be available. Furthermore, strong requirements are also put on the perception activities. They must be aware of the imperfection of acquired information (Section 2.2.3) and be adapt to environment properties (Section 2.2.2).

In the context of large-scale IA, sensing resources are plenty and heterogeneous, i.e., they differ in the their control properties and type of information they yield. However, they are at the same time unable to satisfy a multitude of relevant objectives (i.e., information needs and requests). Sensing resources might, furthermore, have a number of different properties that should be acknowledged and treated. For instance, resources might not be available all the time and the time period between a sensing action has been selected until the time a measurement is returned could be considerable.[14]

Constraints of heterogeneous resources (e.g., interference, mutually exclusive modes) and sensing opportunities (e.g., relocating sensors) makes it important to facilitate observations (Section 3.10). Finally, focus of attention is essential to decide what kind of information and what activities are beneficial to the system objectives (Section 3.11).

The author is not aware of any effort that addresses a larger subset of the aforementioned challenges related to IA. However, there are a few recent DARPA sponsored projects that appear to be moving in that direction. The first is [HVM+01] which uses (potentially) many stationary sensors for target tracking. The most interesting aspect of [HVM+01] for large-scale IA is perhaps its facilitation aspect. In order to enable a large number of sensors to contribute to the target tracking process, the environment is divided into a number of non-overlapping sectors. The sensors are only allowed to communicate with other sensors in the sector it belongs to. Using this convention, communication costs are kept low, and the system becomes scalable.

The second work is [CLOHd+01]. The objective is also in this case target-tracking, but here sensors are mobile which inflates the observation scope. Scaling

[14] [XS02] makes a distinction between short-term and long-term sensor management strategies and includes some references to works that deal with the latter.

is handled similarly as in [HVM+01]; the environment is divided in zones (cf. sectors in the other work). A hierarchy of agents share the responsibility of tracking in the environment. A *zone coalition leader* agent decide how many sensors should pass from one zone to another (if necessary) and a *sampler coalition leader* agent controls the sensors within a specific zone and obeys orders from a superior zone coalition leader. Orders include directing sensors to targets and occasionally sending some to another zone.

From the point of view of large-scale IA, [CLOHd+01] and [HVM+01] make good attempts to facilitate the use of a large number of sensors. However, in other respects they do not contribute as much to large-scale IA. For instance, both works are limited to the perception activity of monitoring and they only consider homogeneous sensors.

3.13 Summary

In this chapter, we wanted to derive important aspects of information acquisition from previous efforts. We mentioned that IA is an interdisciplinary issue that is dealt with, directly or indirectly, in, e.g., sensor management, robotics, and agent theory. A second objective of the chapter was to elaborate on the properties of large-scale IA.

We have emphasised important aspects of the perception activities, i.e., the perception management processes that realize IA. Perception activities may exhibit the aspects of facilitation and focus of attention to varying degrees. We proposed a taxonomy of perception activities by studying in what way they contribute to the environment representation of the observing system.

We saw that, even though, large-scale IA seems to promise to become an important topic for the future, very little attention has been devoted to it to date.

3.14 Discussion and conclusion

With the increasing availability of perception resources such as sensors, large-scale IA appear destined to become a necessity and its associated problems will have to be addressed. The need is perhaps currently most urgent in the defence industry, where decentralisation of resources is an issue being considered. The concept of network centric warfare (NCW), which aims at utilising the information that a military organisation collectively possesses and share it effectively within an information exchange network, has enjoyed a lot of attention over the last decade or so.

The expected increase in intelligent small devices everywhere in society (e.g., in wearable computers and in the intelligent home), is another indication for a growing interest in large-scale IA.

To reiterate, we consider the following features to be characteristic of generic large-scale IA

- distributed resources (for the greatest sensing scope);
- heterogeneous sensing resources (to acquire different types of information);
- decentralised control (to make control manageable also for large sensor sets).

In order to realize large-scale IA, we need to be aware of and master the three types of perception activities and the two emerging aspects. We, furthermore, need to be able to integrate selected perception activities using, e.g., scheduling and planning techniques, to accomplish large-scale IA. There is also an inherent need for long-term planning, rather than the currently more popular "one-step ahead" planning. The reason is simply that sensing actions may have long and varying (sensor dependent) time horizons.

Finally, as mentioned, the classification of literature in this chapter, in perception activities and IA aspects, is neither final nor complete (i.e., there are no clear boundaries between activities and aspects). Rather it serves to highlight pertinent facets of information acquisition. Complementing taxonomies could be considered, including centralised/decentralised control, on-line/off-line algorithms, and mathematical techniques (e.g., decision theory, Kalman filtering, etc). A brief survey of the latter type is provided in [MM94].

Chapter 4

A game theoretic model for management of mobile sensors

4.1 Introduction

Mobile sensing resources (or *mobile sensors* for short) provide a flexible aid to decision support systems for decision-making in dynamic, extensive environments. Their sensing capabilities contribute with observations to the decision support system and their mobility allow them to adapt to changing information needs and altered mission requirements.

Sensors are a limited resource and to achieve good performance in a system with mobile sensors, allocation and use of sensor is a key aspect to consider. *Sensor management* is the process that aims at controlling sensors to improve overall system performance.[NN00] Typical factors of concern for a practical sensor management system are probability of target detection, track/identification accuracy, probability of loss-of-track, probability of survival, probability of target kill, etc.[XS02]

One aspect of managing mobile sensors is coordination of their actions. Choosing a *centralised* approach to coordinate the system promises to provide the system with optimal coordination. However, such a system is both vulnerable (e.g., if the centralised control node is destroyed, the whole system will fail) and slow (e.g., sensors have to await orders from the centralised control). These two factors are essential for systems operating in civilian applications, and even more so in a military application since the environment is expected to be hostile and willing to exploit the two drawbacks (e.g., by jamming communication or targeting the centralised control). Decentralised control, on the other hand, assumes that the system is mainly controlled by its components (e.g., mobile sensors), allowing it to "degrade gracefully" if some of its components fails. However, achieving good performance with decentralised control is a, by far, greater problem.

Distributed artificial intelligence (DAI) is a research field that concerns itself with coordinated interaction among distributed entities, known as *agents*.[Wei99] *Game theory*, constituting a toolbox of methods for analysing interactions between decision makers,[OR94] has attracted a lot of attention from the DAI community. Interestingly, game theory concepts seem to apply better to automated agents than to the real-life human decision-makers for which it was originally intended.[ZR96] The reason is simply that the agents are normally both rational and obedient, qualities which rarely apply to their human counterparts.

Game theory offers models for distributed allocation of resources and provides at the same time mechanisms to handle uncertainty. An important subtopic of game theory is *negotiation*. As part of negotiation there are ways to generate multi-objective optimisation results that are at least Pareto optimal. At the same time, these methods allow for robust handling of game/agent configurations which makes it robust to jamming and use of sensors with limited availability.

Works in DAI seldom consider uncertainties [CFK97] such as those imposed by the physical world (e.g., estimation errors) which are inherent to target tracking applications. Noteworthy recent exceptions concerning target tracking include [DN01] and [CLOHd+01]. In [DN01], stationary sensors form coalitions (groups) where each coalition track a certain target. The members of a coalition fuse their measurements to improve target state estimation. In [CLOHd+01], mobile sensors form coalitions to track targets, each sensor capable of sensing one target at a time. Movements of sensors are decided by a hierarchy of coalition leaders, each responsible for a certain geographical area.

The management, in our approach, is performed using negotiation models from game theory. We utilise an algorithm for agent negotiation which we have previously developed and evaluated.[XCS03] In the previous work, sensor agents negotiated about which targets to track, dividing the set of targets among themselves. Sensors were stationary, but now we apply the same algorithm to the case with mobile sensors and allow sensors to share targets. Our work constitutes a framework for future studies of management of distributed mobile sensors in the face of uncertainties and sensor failures.

In future work, we want to show the advantages of the game theoretic approach when faced with uncertainties and possible sensor failures.

The next section will place this work in the context of prior work. Section 4.3 presents the primary objectives of this work, including management of mobile sensors and sharing of targets. Section 4.4 explains the negotiation procedure and its utility functions. Section 4.5 presents some results of using the negotiation strategy, and Section 4.6 discusses some of the manageable parameters of the model. Finally, Section 4.7 concludes and suggests future research.

4.2 Target allocation using sensor agent negotiation

The work that precedes the work described in this chapter considered a sensor network consisting of geographically dispersed, non-mobile sensing resources.[XCS03]

The sensing resources (sensors) were expected to cooperatively monitor some geographical area to keep track of all targets known to be within that region. A solution to the problem would be a partition of the target set into disjoint subsets, and an assignment of subsets to sensors so that every subset was assigned to exactly one sensor. In other words, a valid solution required that every target be tracked by exactly one sensor. However, sensors were allowed to track more than one target each.

A solution algorithm based on the game theory concept of negotiation was proposed and the utilities of negotiation offers were calculated from the information gain (explained in Section 4.2.2) the corresponding target division was expected to bring.

4.2.1 Agent negotiation

There are two kinds of consequences of an agent negotiation: agreement and disagreement. Disagreement means that no solution acceptable for all agents can be reached. In the other case, an agreement between the agents can be reached.

Every agent has its own preference relation over possible agreements and times of agreements. We assume that a sensor agent $i \in S$, S being the set of sensor agents, has a utility function, U_i, which represents its preference relation. The utility function assign values to all possible outcomes of a negotiation:
$\{\mathcal{O} \times \{0, 1, \ldots, K\} \cup \{Disagreement\}\}$, where $\mathcal{O}$ is the set of possible offers and K refers to the final step of negotiations.

As agents negotiate in order to realize cooperative behaviours among themselves in multi-target tracking, reaching an agreement is in line with the interests of all agents and no one can benefit from disagreement (i.e., non-coordinated behaviour[1]).

A formal description of the negotiation game we study is the 5-tuple $< S, \mathcal{O}, H, P(H), (U_i) >$, where

S is the set of sensor agents (called players in game theory terminology)

$\mathcal{O}$ is the set of *negotiation offers*, i.e., the possible sensor to target allocations. A member of $\mathcal{O}$, o, is an allocation function that maps sensors to subsets of the set of targets, T, i.e., $o : S \rightarrow 2^T$.

H is the set of sequences of offers and responses (called *histories* in game theory) in a negotiation. A non-terminal history is a sequence, for instance (o_0, R, o_1, R), which ends with a rejection, R, preceded by a series of consecutive offers (o_t is the offer a step t in the negotiation). A terminal history, on the other hand, ends with an agreement, e.g., (o_0, R, o_1, R, o_2, A).

$P(H)$ is a function that determines which agent has the turn to make an offer after a non-terminal history h.

[1]Clearly, non-coordinated behaviour (i.e., sensors tracking whatever target they like, disregarding the allocations of the other sensors) would be unlikely to meet the system requirements, such as ensuring that all targets are tracked.

(U_i) are utility functions of sensor agents $i \in S$ over outcomes, which describes how the agents value every allocated group of targets

It is assumed that at a particular step of a negotiation, one of the agents makes an offer and the other agents respond to it by acceptance or rejection. The order in which the agents make their proposals is specified before the negotiation begins. The first action in the game occurs in step zero when one agent makes the first offer and the other agents accept or reject it. Acceptance by all other agents ends the game with agreement while rejection by at least one other agent forces the game to continue with another step. Subsequently, another agent proposes something in the next step which is then accepted or rejected by the others. The game continues in this manner until an agreement has been reached or until the final step K. If no agreement is reached at step K, we say that the game ends with disagreement.

A *negotiation strategy* for an agent is essentially a function that specifies what the agent has to do after every possible history.[2] Concretely, the strategy prescribes what offer to make when it is the turn of the agent to make an offer, and whether to accept or reject an offer in steps when the agent is to respond to a proposal made by another agent. A *strategy profile* is a collection of strategies for all involved agents. We would like to find a strategy profile leading to an outcome that is profitable for all participants and that no agent can benefit from using a strategy not belonging to the profile.

A fundamental concept for analysing behaviours of rational agents is the *Nash Equilibrium*.[Nas53] A strategy profile of a game of alternating offers is a Nash Equilibrium if no agent can profit by deviation given that all other agents use the strategies specified for them in the profile. Unfortunately, simple Nash Equilibrium seems not sufficient in extensive games[3] in the sense that it ensures the equilibrium of its strategies only from the beginning of the negotiation, but may be unstable in the intermediate stages.

A stronger notion for extensive games is that of *subgame perfect equilibrium* (SPE) [OR94] that requires that the strategy profile included in every subgame is a Nash Equilibrium of that subgame. This is a comprehensive concept implying that agents are rational at any stage of the negotiation process: no one can be better off by using another strategy regardless of what happens. It was shown in the preceding paper that if all agents honour SPE strategies, there is an offer made in the first step which is preferred by all parties over all possible future outcomes.

When reasoning about strategies, we start at the final step of negotiations, K. If no agreement has been reached yet, one will certainly be reached in the final step since disagreement (which would be the outcome of a rejection of the final offer) is the worst outcome for all agents and will be avoided. If it is agent i's turn to make an offer at step K, it will choose the offer that is best for its own payoff, and the other agents accept this offer, since disagreement is the only alternative.

[2] Thus, *strategy* is similar to the concept of *policy* in AI-literature

[3] In extensive games, agent actions are performed in sequence as opposed to strategic games, or one-shot games, where actions are performed simultaneously.

At all steps before K, the agent, whose turn it is to make the offer, will consider the agreement that would be reached at the next step and proposes something that is better or at least as good (in terms of utility) for all agents than what they can expect to attain in future steps.

Suppose $o^*(t+1)$ represents the agreement that will be reached at step $t+1$. Given that the agents are rational, then for all parties at time t the *super*(t)-set defines the set of acceptable solutions:

$$Super(t) = \{o \in \mathcal{O} \mid \forall i\ U_i(o,t) \geq U_i(o^*(t+1), t+1)\} \tag{4.1}$$

If an agent selects its offer from the *Super*-set in Equation 4.1, then all other (rational) agents will accept it since no offers with higher utility will be offered in future steps.

Furthermore, it is important to notice that the *Super* set is non-empty for all steps before K. This is induced from the characteristic that the utilities of offers decrease over time. Particularly, the offer $o^*(t+1)$ is included in $Super(t)$ since we have $U_i(o^*(t+1), t) > U_i(o^*(t+1), t+1)$ for all agents i.

The *Super* set of acceptable offers is very useful to establish SPE strategies at steps before K. The non-emptiness of this set ensures that the agent whose turn it is to make an offer has enough choices to make its proposal acceptable to the other agents. The strategy we use is that the agent i whose turn it is to make an offer at step t will propose the offer $o^* \in Super(t)$ that maximises $U_i(o,t)$. If several candidate offers maximise U_i, (this set was denoted $Compet(i,t)$, in the preceding work) we let agent i propose the offer that not only maximises $U_i(o,t)$ but also the sum of the utilities of the other agents, i.e., $o^*(t) = \arg\max_{o \in Compet(i,t)} \sum_{k \in S \setminus i} U_k(o,t)$.

Finally, the fact that $U_i(o(t),t) \geq U_i(o(t+1), t+1)$ for all agents causes the game to end already in the first step with agreement $o^*(0)$.

4.2.2 Sensor performance utilities

We assume that the sensors track targets using a Kalman filter and let the utility functions of the agents, U_i, depend on the decrease in uncertainty that is estimated in the Kalman calculations.

We will not present the entire Kalman filter method here (instead see, e.g., [BSF88]), just simply point out what part of the method was used in the previous work to derive a utility measure for target tracking sensors.

In the Kalman filter method, we let $x_j(k)$ represent the system model of target j at time k, and $y_{ij}(k)$ the corresponding measurement model (for a sensor i). The following familiar equations are used:

$$\begin{aligned} x_j(k) &= F_j x_j(k-1) + w_j(k-1) \\ y_{ij}(k) &= H_{ij} x_j(k) + v_{ij}(k) \end{aligned} \tag{4.2}$$

In Equation 4.2, $w_j(k)$ and $v_{ij}(k)$ are system and measurement noise, respectively.

An expression for the update of the target estimation error covariance reveals the measure of performance [BSL93] which we use:

$$P_{ij}^{-1}(k|k) = P_{ij}^{-1}(k|k-1) + H_{ij}^T R_{ij}^{-1} H_{ij} \tag{4.3}$$

Here, $P_{ij}(k|k)$ is the updated state estimation error covariance and $P_{ij}(k|k-1)$ is the predicted covariance. The matrix $H_{ij}^T R_{ij}^{-1} H_{ij}$ is derived from sensor characteristics, where R_{ij} is the measurement noise covariance. Note that a decrease in the measurement noise covariance also leads to a decrease in the state covariance, i.e., a reduction of uncertainty about the target state. In view of this, we define the norm of the matrix $H_{ij}^T R_{ij}^{-1} H_{ij}$ as *sensor information gain*, $g(i,j)$, contributed by sensor i on target j.

$$g(i,j) \triangleq \begin{cases} \|H_{ij}^T R_{ij}^{-1} H_{ij}\|, & \text{if sensor } i \text{ tracks target } j \\ 0, & \text{if not} \end{cases} \tag{4.4}$$

By means of sensor information gain, we establish the measure of performance of a sensor estimating properties of all targets assigned to it. Suppose sensor i is in charge of a group of targets, D_i, then its contribution to the global picture is accrued by measuring all assigned targets. Hence, the performance of sensor i, P_i, is defined to be the sum of these information gains for state estimates of targets in D_i,

$$P_i(D_i) \triangleq \sum_{j \in D_i} g(i,j) = \sum_{j \in D_i} \|H_{ij}^T R_{ij}^{-1} H_{ij}\| \tag{4.5}$$

We call a value given by the expression $P_i(D_i)$ for the *sensor performance* of sensor i when tracking a group of targets D_i.

We note that an offer is a distribution D of targets among sensors, $D = \bigcup_i D_i$, i.e., each sensor i gets a subset of targets D_i to track. For every sensor, the acquired set of targets corresponds to a value of sensor performance using the definition in Equation 4.5, and we will now explain how we use the sensor performance value, for a sensor and a set of targets, to calculate the corresponding utility value.

An agent is assumed to receive a *reward* not more than unity in terms of its contributed performance. The purpose of doing so is to normalise the sensor performance value for easy handling and to allow for non-zero rewards for sensors that accept no work (i.e., do not track any targets). The reason to allow sensors to be "lazy" is to encourage them not to reveal themselves (by use of active sensors) too often, i.e., to be *quiescent*. The reward r_i of sensor i, appointed target group D_i, is given by

$$r_i(D_i) \triangleq \alpha + (1-\alpha)(1 - e^{-\beta \cdot P_i(D_i)}), \quad 0 \le \alpha < 1 \text{ and } \beta > 0 \tag{4.6}$$

such that $P_i \in R^+$ is converted into a regular interval $[\alpha, 1)$. Here, β is a parameter which decides how eager the sensor is to acquire more information about a target. A high value on β means that the sensor agent is satisfied with less certain state estimates (cf α_j in Figure 4.2). The other parameter, α, controls the agent's willingness to differentiate between the offers. For instance, $\alpha = 1$ means complete indifference, i.e., all offers have the same value ($r_i(D_i) = 1$).

As explained in Section 4.2.1, utility is expected to decrease over time in the negotiation, and we therefore define the time-dependent utility function in this way:

$$U_i(D,t) \triangleq (K - t + 1) r_i(D_i). \tag{4.7}$$

4.3 Primary objectives

In this chapter, we extend the previous work discussed in Section 4.2 considering the following three aspects:

Mobile sensors We allow sensors to move to increase sensor performance. We further allow the characteristics of the terrain to affect the *preferred direction of motion*.

Shared targets We extend the previous work by allowing sensors to track the same targets (previously, the targets were divided between the sensors). Through use of multiple sensors tracking the same target, it is possible to improve the performance on state estimates as typically found in the multi-sensor tracking and multi-sensor fusion literature (e.g., [BSF88]). Here, this problem is studied in the context of target assignment and performance optimisation. We model that the value of tracking a target, which is already being tracked by other sensors, is less than if none tracks the target.

Performance loss when tracking many targets We model that the measurement performance on each target tracked by a sensor decreases with the number of targets tracked by the same sensor. The reason is of course that the sensor has limited time and resources for its measurements and if it has to track more targets and divide its resources among the targets, then also the measurement error covariance will increase for every target (and sensor gain decrease).

In order to allow the mobility of sensors to have any effect, we further assume that sensor platforms have the ability to move at a speed that is comparable to the speed of the targets.

4.4 Utility and negotiation for mobile sensors

There are only small differences between the game considered in the previous work (defined in Section 4.2.1) and the one we consider here. Sensor agents negotiate by making offers that the other agents might accept or reject. As in the previous work, an offer, o, is a specification of allocations, that assigns groups of targets to sensors.

Unlike the previous work, the target groups may overlap, significantly increasing the number of valid offers. The other difference is in the utility functions, which, we shall see in the next section, has a two-dimensional values.

For negotiation about target allocation of mobile sensors, we consider both the reward for each sensor as well as its *directional derivative* in the preferred direction of motion. The reward, as we will see, is calculated somewhat differently than in the previous work and does not immediately yield the negotiation utility. The preferred direction of a sensor platform is the spatial direction in which the sensor would like to travel. When we do not consider terrain characteristics, the preferred direction will simply coincide with the gradient of the reward function.

In the next section, we will first present an approach to consider mobility in the negotiation. In the subsequent sections 4.4.2-4.4.4, we will discuss how to calculate both the reward for the novel considerations of overlapping target groups and decreased tracking performance, and the preferred direction. We also address the resulting multi-objective optimisation problem.

4.4.1 Negotiation

Before we start to discuss the details about reward and directional derivative, we will, for this work, assume that every sensor agent has the required information and is capable of calculating both objectives for all sensors.[4] Hence, given a sensor agent $i \in S$ and an offer of allocations of sensors to targets $o \in \mathcal{O}$, we can calculate reward $r_i \in \mathbb{R}$ and directional derivative $r'_{i,\delta} \in \mathbb{R}$, i.e., a sensor and an offer yields a reward and preferred direction, $S \times \mathcal{O} \rightarrow \mathbb{R} \times \mathbb{R}$. Here $\delta \in \Delta$ is the preferred direction, and Δ the set of unit vectors.

We want to consider both factors, reward and derivative (measured as change in reward per length unit), simultaneously to acquire a combined utility metric. A valid but tentative approach is to assert a utility function $U = U(f(r), g(d))$ that analytically combines the two. However, the factors are incommensurable, and, hence, such a function is sensitive both to the application in question and the choice of measurement unit. E.g., we might propose $U(r, d) = r + d$. While this utility function might yield satisfying results for some applications, it will certainly not do so in general. Rather, the appropriate functions (f and g) have to be found for every specific application or class of applications.

The problem we are facing is that of multi-objective optimisation. Whereas elaborate approaches to this problem has been proposed (such as [FF98]), in this work we prefer to study the results of an approach that does not suggest a preference of one factor over the other. (Hence, it might not work optimally for every application, but is expected to work well for every application.) We order the offers only according to *dominance*.

A sensor agent, i, will prefer an offer o_1 to another offer o_2, $o_1 \succ_i o_2$, if and only if o_1 dominates o_2. An offer o_1 can only dominate another offer o_2 if one of the reward

[4] In a practical application, the complete knowledge is not going to be available to all sensors, but for an initial study it is convenient to make this assumption.

and directional derivative values of o_1 is greater than the corresponding value of o_2 and the other one at least as great as its counterpart, i.e., $r_i(o_1) \geq r_i(o_2)$ and $r'_{i,\delta}(o_1) \geq r'_{i,\delta}(o_2)$ and at least one of the inequalities should be strict. If neither o_1 nor o_2 dominates the other, we write $o_1 \sim_i o_2$.

We elaborate further on the topic of dominance. Figure 4.1 shows twenty offers, here depicted with circles, plotted in a graph according to the reward and derivative in the preferred direction of a certain agent (certainly, the plot would look different for another agent). We find that there are, in this example, five offers that are not dominated by any other offer. We conclude that these are the "best" offers the agent could get. We call the set (or *class*) of these the *offers of the first order*. We iteratively classify the rest of the offers, knowing that an offer o_k, which is dominated by an offer o_l of order l, will be a member of order $l+1$ or greater. Each offer in Figure 4.1 belongs to one of five orders and the members of each order are connected to each other with dashed lines for illustration.

A more formal definition of class of offers for a particular sensor is as follows.

Definition: Class of offers All pairs of offers (o_1, o_2), $o_1, o_2 \in \mathcal{O}$, that fulfil the condition that $o_1 \sim o_2 \wedge \neg \exists o_m \in \mathcal{O}\, [(o_m \sim o_j \wedge o_m \succ o_k)]$ for $j \neq k$ and $j, k \in \{1, 2\}$ are said to belong to the same class of offers.

A class may not be empty, but may contain a single offer o_s iff $\forall o_j \exists o_k\, [o_j \sim o_s \wedge o_j \sim o_k \rightarrow o_k \succ o_s \vee o_k \prec o_s]$. In order to strictly define class order, we first define the notion of *class dominance*.

Definition: Class dominance A class of offers C_a is said to dominate another class C_b, $C_a \succ C_b$, iff $\exists o_a \exists o_b\, [o_a \succ o_b]$, $o_a \in C_a, o_b \in C_b$.

We use the following recursive definition to define order of class.

Definition: Class of first order A class C of offers is said to be of the first order iff none of its offers are dominated by another offer, $\neg \exists o_m \in \mathcal{O} \setminus C\, [o_m \succ o_j]$ for all $o_j \in C$.

Definition: Class of kth order A class of offers C is said to be of order k iff its members are dominated only by members of classes of order k and less.

Now, we will express the utility function for sensors. Using our notion of orders, we can assign a utility value to the offers for all agents. Furthermore, according to the negotiation procedure described in Section 4.2.1, the utility of an offer accepted at time $t+1$ is always less valuable than the same offer accepted at time t. Therefore we need to construct a utility function that is dependent on the time step of the negotiation:

$$U_i(o, t) \triangleq \alpha_U (K - t) - order_i(o), \quad \text{integer } \alpha_U > 0, \tag{4.8}$$

where $order_i(o)$ is a function that maps offers to its order for sensor agent i.

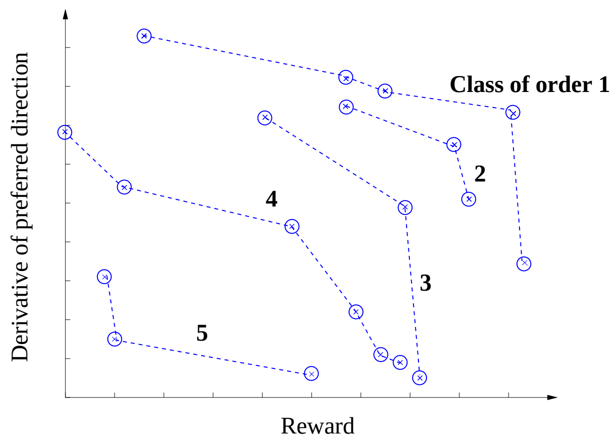


Figure 4.1. Twenty offers plotted according to derivative of preferred direction and reward. Offers which belong to the same class are connected with lines.

The interpretation of the utility function in Equation 4.8 is that the agents will accept less offers the longer the negotiation continues. In fact, for every step the negotiation continues, α_U number of orders of offers will become unacceptable for every sensor agent. Thus, offers of low order (which are desired by the agent) yield a high utility value.

The negotiation procedure in the preceding work described in Section 4.2 is virtually unaffected by the extensions we make in this work. The reason for this is that all the novelties have been encapsulated in the calculations of the utility function. However, the result of the negotiation will of course be quite different.

An agent that have several "best" offers to choose from should select one according to some second criterion. This could for instance entail minimising the sum of orders, i.e., if the set of best offers is $\mathcal{O}'$, then the offer to select should be the one o^* that satisfies $o^* = \arg\min_{o \in \mathcal{O}'} \sum_i order_i(o)$. Another suitable criterion could be to select the offer in $\mathcal{O}'$ that minimises maximum order for any sensor, i.e., $o^* = \arg\min_{o \in \mathcal{O}} \max_i order_i(o)$. If there are still more than one offer that fulfils the criterion, then one offer could be selected randomly.

4.4.2 Workload effect on tracking performance

A sensor is expected to make less certain measurements for each target if it tracks many targets than if it tracks only a few. Let us assume that sensors have some sort of resource (e.g., time, energy, money, samplings) that they can spend utilise to make measurements. The maximum amount of this resource available to the sensor for a time unit is $\rho_{i,max}$ and the amount it chooses to use to track some target j is denoted ρ_{ij}. We model that the measurement noise v_{ij} in Equation

4.2 is dependent on the dedicated resource amount ρ_{ij}. The measurement noise is Gaussian, i.e., $v_{ij}{\sim}N(\mathbf{0}, R)$, with zero mean and the measurement error covariance matrix R which we encountered in the sensor performance formula in Equation 4.5.

Dissecting R, we notice that it is a diagonal matrix (since a prerequisite for the Kalman filtering is that the, say k, measurement noise components of v_{ij} are independent)

$$R = \begin{bmatrix} \sigma_1^2 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \sigma_k^2 \end{bmatrix} \tag{4.9}$$

with the variances σ_l^2, $l{\in}\{1, \ldots, k\}$, as diagonal elements.

The standard deviation functions, $\sigma_l(\rho)$, will take a minimum, $\sigma_{min,l} \geq 0$, for $\rho = \rho_{max}$ and will increase towards infinity when the dedicated resource decreases towards ρ_{min}, $\lim_{\rho\to\rho_{min}(j)} \sigma_l(\rho){\to}\infty$, where $\rho_{min}(j)$ is the minimum resource amount necessary to track target j.

Hence, a varying workload on a sensor will affect the standard deviation and the measurement error covariance matrix, which in turn will have effect on the refined sensor gain expression which we will discuss in the next section.

Note that using this model, we allow sensors to allocate different amounts of resources to different targets.

4.4.3 Target allocation

In the preceding work, the specific task was studied where every target was tracked by exactly one sensor. In this work, we relax that restriction and allow sensors to "share" targets. Thus, we are able to reduce uncertainty by fusing measurements from different sensors and get a higher grade of sensor usage than in the disjoint case.

Our approach here to determine the reward for every sensor, r_i, is to divide the *total reward* on every target, $\sum_j r_j(S_j)$ (S_j being the set of sensors tracking target j), among the sensors in S_j proportionally to their individual contribution.

We define the reward on every target to be

$$r_j(S_j) \triangleq 1 - e^{-\alpha_j g_j(S_j)}, \quad \alpha_j > 0 \tag{4.10}$$

where

$$g_j(S_j) = \sum_{k\in S_j} \|H_{kj}^T R_{kj}^{-1} H_{kj}\|, \tag{4.11}$$

i.e., the total information gain on target j. Here, we might want to replace g_j with some measure from information theory when a Kalman filter is not applicable. Previously, we used sensor information gain (Equation 4.4), but now, as we allow multi-sensor fusion, we define $g_j(S_j)$ as above and notice that whenever S_j contains a single sensor i $g_j(S_j) = g_j(i) = g(i, j)$.

Now, the *net reward* for every sensor (similarly expressed as in the previous work) is

$$r_i^{net}(D_i) \triangleq \alpha_i + (1-\alpha_i) r_i^m(D_i) \quad 0 \le \alpha_i \le 1 \tag{4.12}$$

where D_i is the group of targets tracked by sensor i, α_i reflects the willingness of a sensor agent to compromise about offers, and the measurement reward is

$$r_i^m(D_i) \triangleq \sum_{j \in D_i} \gamma_{ij} r_j(S_j) \tag{4.13}$$

and

$$\gamma_{ij} \triangleq \frac{g_j(i)}{g_j(S_j)} = \frac{\|H_{ij}^T R_{ij}^{-1} H_{ij}\|}{\sum_{i \in S_j} \|H_i^T R_{ij}^{-1} H_i\|}, \tag{4.14}$$

i.e., the relative contribution of sensor i to the state estimate of target j.

This definition of sensor reward, $r_i^m(D_i)$, has the effect that the same gain from a sensor on a target will yield different rewards depending what other sensors track the same target. This makes sense since the target reward does not improve linearly with the information gain (e.g., in a target tracking application, tracking airborne targets at high speeds, to go from metre to centimetre precision in position estimates should not yield much extra reward since the improved precision can not be efficiently utilised).

To prove that $r_i^m(D_i)$ is actually a disbursement of the total reward on targets we need to show that $\sum_i r_i^m(D_i) = \sum_j r_j(S_j)$.

$$\begin{aligned} &\textstyle\sum_i r_i^m(D_i) = \sum_i \sum_{j \in D_i} \gamma_{ij} r_j(S_j) = \big\{\text{Group all } r_j\text{-terms in the sum}\big\} = \\ &\textstyle\sum_j r_j(S_j) \sum_{i \in S_j} \gamma_{ij} = \big\{ \sum_{i \in S_j} \gamma_{ij} \triangleq 1 \big\} = \sum_j r_j(S_j) \end{aligned} \tag{4.15}$$

■

Typical appearances of net reward functions, r_i^{net}, are shown in Figure 4.2. In the figure, we use $\alpha_i = 0.3$ and plot r_i^{net} for $\alpha_j \in \{1, 10, 100\}$. We show the results for a single sensor tracking a single target. The curves have similar shape for other values on α_i. For these curves, we have used the covariance matrix in Section 4.5.1. From the curves, we can see that that an increase in the value of α_j implies that the sensor is satisfied with less certain target state estimates.

4.4.4 Preferred direction

Given the measurement reward function, $r_i^m(D_i)$, for each sensor, the gradient can be calculated in this way:

$$grad\ r_i^m \equiv \nabla r_i^m \equiv \big(\frac{\partial r_i^m}{\partial x_i}, \frac{\partial r_i^m}{\partial y_i}\big), \tag{4.16}$$

where x_i and y_i are the spatial coordinates of sensor i's position.

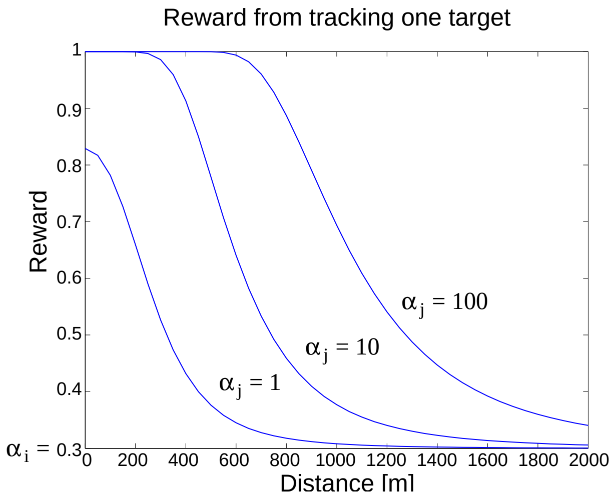


Figure 4.2. The three curves show a net reward function plotted with $\alpha_i = 0.3$ and $\alpha_j = 1, 10$, and 100.

The gradient vector points in the direction in which the reward for sensor i will increase the most.[5] This model makes the subtle (and incorrect) assumption that the targets are stationary. However, it is a fairly good approximation that should be refined in the future; possibly by predicting and exploiting future target states. The gradient would be the preferred direction to move for the sensor if terrain properties were not considered.[6] However, the terrain may make motion in the direction of the gradient difficult or perhaps even impossible, and a more passable path, although less rewarding, might be a better preferred direction.

Now assume we can construct a (possibly rough) terrain dependent function, which discounts the reward change in various directions. Let the *terrain function* be $t(\mathbf{p}, \mathbf{e}_\theta)$, where $\mathbf{p}$ is a two dimensional position in the environment and $\mathbf{e}_\theta$ is a unit vector, $\theta \in [0, 2\pi)$. Furthermore, let the terrain function assume values between 0 and 1, $t \in [0, 1]$. The terrain function $t(\mathbf{p}, \mathbf{e}_\theta)$ takes high values in directions where the sensor platform can easily move (such as in the direction of a good road) and low values in directions where it cannot move very well (zero in the direction of an unpassable obstacle). We assume that the value reflects the passability in the chosen direction in the following time step.

The directional derivative $r'_{\mathbf{e}_\theta}$ in any direction, $\mathbf{e}_\theta$, is simply a projection of the gradient onto $\mathbf{e}_\theta$, i.e., $r'_{\mathbf{e}_\theta} = \mathbf{e}_\theta \bullet \nabla r^m$. The parameter θ is the angle between the gradient and $\mathbf{e}_\theta$, as shown in Figure 4.3.

[5] Note that we are, in this work, only considering the current target states when calculating the gradient. Prediction of future target states to further improve the performance of the mobile sensors is left for future work.

[6] Hence, in the case of airborne sensors, the gradient would suffice as a preferred direction.

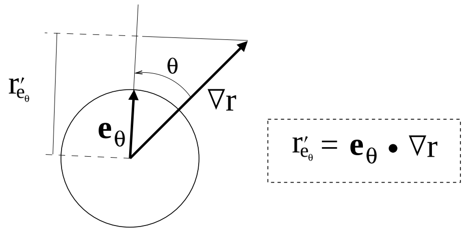


Figure 4.3. The directional derivative, $r'_{\mathbf{e}_\theta}$, in the direction of the unit vector $\mathbf{e}_\theta$ is calculated as the projection of the gradient, ∇r, on $\mathbf{e}_\theta$.

Now, we propose that the preferred direction, δ^*, is the unit vector that corresponds to the largest directional derivative discounted by $t(\mathbf{p}, \mathbf{e}_\theta)$, i.e.,

$$\delta^* = \arg\max_{\mathbf{e}_\theta} \left\{ t(\mathbf{p}, \mathbf{e}_\theta) \cdot r'_{\mathbf{e}_\theta} \right\}. \tag{4.17}$$

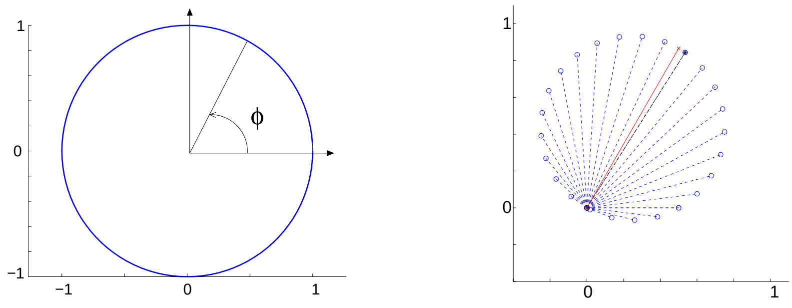
Figure 4.4(a) shows a terrain function in a position $\mathbf{p}$ where terrain has no effect on the mobility, i.e., in all directions, ϕ, $t(\mathbf{p}, \mathbf{e}_\phi) = 1$. Figure 4.4(b) shows the resulting discounted directional derivatives (which in this case were unaffected by the terrain function) where the gradient is depicted as the solid line with a cross on its end point. The length of a line corresponds to the size of its derivative. Directions with derivatives less than zero (those directions which have more than a 90 degree angle to the gradient) are not depicted. Since directional derivatives do not change sign due to the terrain function; they are only discounted with a positive factor so the smallest discounted derivative possible is zero. Hence, the directions with negative derivatives can be ignored.

Figure 4.5(a) shows the heterogeneous terrain function

$$t(\mathbf{p}, \mathbf{e}_\phi) = \begin{cases} 2\phi/\pi & \phi \in [0, \pi/2] \\ 1 & \phi \in (\pi/2, 3\pi/2] \\ 0 & \phi \in (3\pi/2, 2\pi) \end{cases} \tag{4.18}$$

Figure 4.5(b) shows the same gradient as in Figure 4.4(b), but here the directional derivatives have been discounted with the function in Equation 4.18. The direction which has the greatest directional derivative is depicted with a solid line, calculated with Equation 4.17, and would be the preferred direction of an agent. This direction deviates notably from the gradient.

We have now seen how the preferred directions of a sensor platform are calculated for terrain which has no effect on the platform (in Figure 4.4(b)) and terrain which has (Figure 4.5(b)). We now expand our field of view to study the preferred directions in a whole area. Figure 4.6(a) shows the directional derivatives in various positions in the plane when its preferred direction is unaffected by terrain conditions. A target in position (400, 350) (the small "x") attracts a sensor platform. We note that the derivatives are small in the periphery and close to the target, and large in between. This is the same characteristics we saw in the curves in Figure 4.2.



(a) (b)

Figure 4.4. **(a)** shows the terrain function $t(\mathbf{p}, \mathbf{e}_\phi)$. In this case $t = 1$ for all ϕ, so it is the function of the unit circle. **(b)** shows the size of some discounted directional derivatives when the gradient is the solid line pointing upwards and right. In this case, the directional derivatives are discounted with the function in (a) and are thus unaffected.

In Figure 4.6(b), an obstacle (representing almost unpassable terrain) has been positioned to the left in figure. The preferred directions direct the sensor platform away from the obstacle, while trying to preserve a course towards the target. For instance, along the upper and lower edges of the obstacle, the preferred directions are along the edge of the obstacle rather than into the obstacle.

Even though the approach with terrain functions presented here looks nice in this example, it is indeed short-sighted. There is a risk that sensor platforms get stuck behind obstacles. However, this does not necessarily mean that the tracking will fail, rather it means that the current allocation has been given a new value which will possibly affect the outcome in the next round of negotiations (i.e., another allocation, with a better preferred direction, might be a more appealing alternative).

4.5 Experimental results

First, we verify that the negotiation algorithm is still beneficial for stationary sensors with respect to the new features of the problem (Section 4.3). Then we move on to verify its suitability to the case with mobile sensors.[7] In Appendix A we select the values of some of the parameters used in the simulation, and in Appendix B we derive an analytical expression for the gradient.

[7]In this chapter, merely snapshots of simulations are shown. However, full animations are available at this URL: `http://www.nada.kth.se/~rjo/pubs/mobile/anim/`.

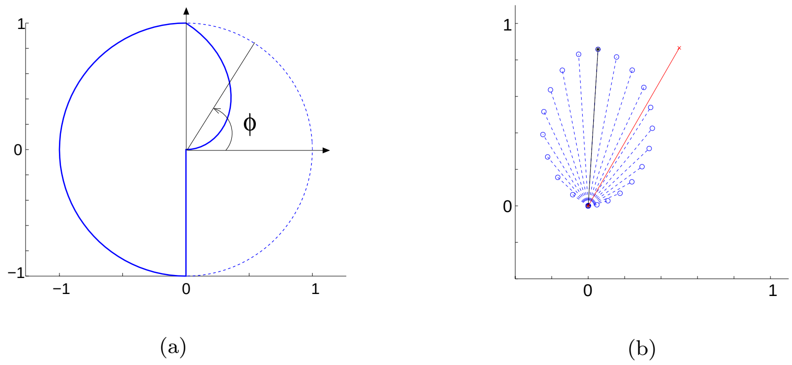


Figure 4.5. **(a)** shows the plot of the terrain function in Equation 4.18. **(b)** shows some of the directional derivatives discounted with the terrain function in Equation 4.18.

4.5.1 Selected parameters and requirements

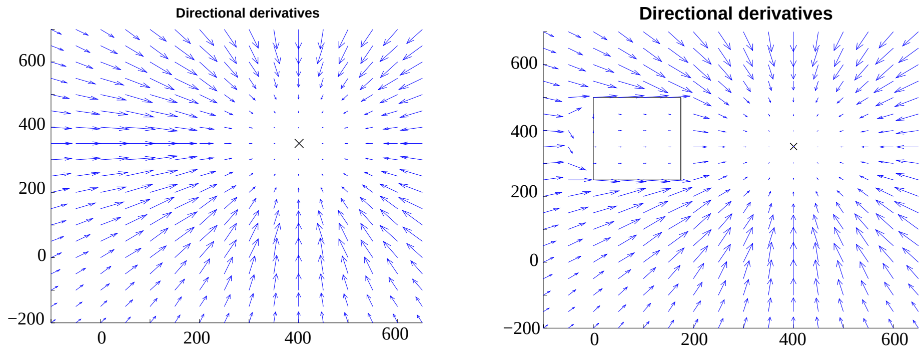
For our simulations, we assume that the standard deviation, σ_{tot}, of the measurement noise covariance R, is equal for every measurement component and tracked target. We, furthermore, assume that it increases inversely linearly with the dedicated relative resource amount, i.e., the standard deviation is scaled by a factor $\left(\frac{\rho}{\rho_{max}}\right)^{-1}$, and quadratically with the Euclidean distance d between target and sensor. If the tracking resource, discussed in Section 4.4.2, is divided evenly between n tracked targets, the resource amount used to track each of the targets is $\rho(n) = \rho_{max}/n$, yielding the scale factor $\left(\frac{\rho_{max}/n}{\rho_{max}}\right)^{-1} = n$ for the standard deviation. From this discussion, we suggest the following standard deviation expression for our experiments

$$\sigma_{tot} = \sigma_{min}\cdot n\cdot(1 + cd^2). \tag{4.19}$$

The first two factors are always greater than zero and $d\geq 0$. The coefficient $c > 0$ controls how greatly the distance from sensor to target affects the measurement error covariance.

Equation 4.19 is plotted in Figure 4.7 for one to four targets. The values on the x-axis denotes the distance to target and the y-axis the relative increase in covariance, with $\sigma_{min} = 1$ as reference. We see, e.g., that the covariance for one target at distance 400 metres, when concurrently tracking four targets, is about ten times the minimum covariance.

We require that the tracking system always tracks all targets (i.e., sensor to target assignments that do not include assignments to all targets will be ignored by all sensor agents).



(a) (b)

Figure 4.6. (a) The derivatives of the preferred directions in various positions. The target, the "x" in position $(400, 350)$, acts as the force that attracts the sensor platform and the terrain does not affect the preferred directions of the platform. **(b)** Here, an obstacle (representing very rough terrain) is situated in the left part of the figure. The generated preferred directions tries to steer the sensor platform away from the obstacle while preserving a course towards the target.

4.5.2 Computational issues

The time complexity of the algorithm to implement is heavily dependent on the number of offers to consider, q, which, in turn, is dependent on the number of sensors, s, and targets, t. In the previous work, offers contained only target distributions where targets were not shared. In that work, the number of offers were $q = O(s^t)$, i.e., it was exponential in the number of targets.

Now, since targets may be shared by sensors, the size of the offer set increases to $O(2^{ts})$ [8] and is, hence, exponential in both the number of targets and sensors. Hence, in our work it is necessary to adopt some heuristic to lower the number of possible offers.

By studying the problem at hand, it is often quite possible to design suitable approximations. In this case, we assume that it is quite unlikely and unwanted that sensor agents change their allocations much from one negotiation to the other. In support of this assumption is the fact that target positions are dependent on kinematic constraints, and the optimal allocation of targets is therefore expected to change slowly over time. Exploiting the expected inertia in the change of the optimal allocation, we construct the set of offers to negotiate about in the following way:

[8] Any sensor may track any number of targets, hence, every sensor may allocate 2^t targets, yielding a total number of $(2^t)^s = 2^{ts}$ possible allocations.

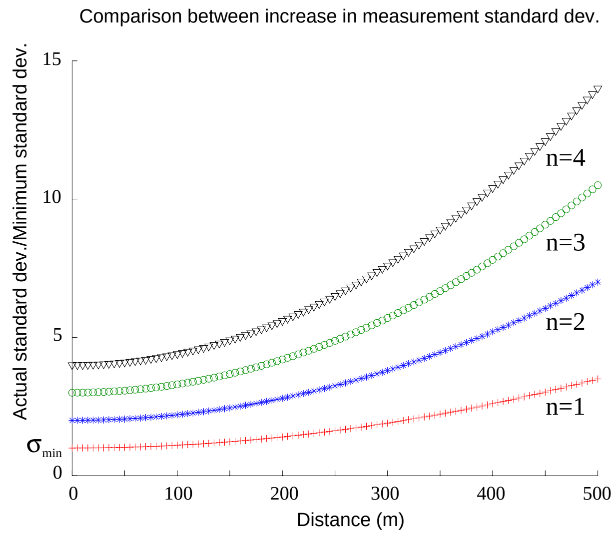


Figure 4.7. The covariance function in Equation 4.19 is here plotted for one to four concurrently tracked targets.

Given the current allocation of targets to sensors, let the set of offers to negotiate about include all combinations for which each sensor, either

- keeps its current allocation,
- drops one target of the current allocation or picks up a new one, or
- exchanges one target for another.

Even though this heuristic reduces the size of the offer set considerably to $O(t^{2s})$, it is still exponential in the number of sensors. For future work, the size of the offer set will have to be decreased even further, but for the experiments in this chapter it suffices.

4.5.3 Target tracking with stationary sensors

The first thing we want to verify is that the negotiation algorithm still, with an implementation of the conditions in Section 4.3, produces satisfying results.

We run a simulation, very much similar to the one used in the preceding work. In the scenario (Figure 4.8), three stationary sensors, spatially separated and placed on a line in east-west direction, track four targets for some time. The targets approach the sensors in pairs, one pair approaching from the east and the other from the west. Targets τ_3 and τ_4 travel slightly faster than τ_1 and τ_2.

In the previous work [XCS03], we compared our negotiation algorithm with an optimal algorithm which optimised the sum of information gain. The results showed

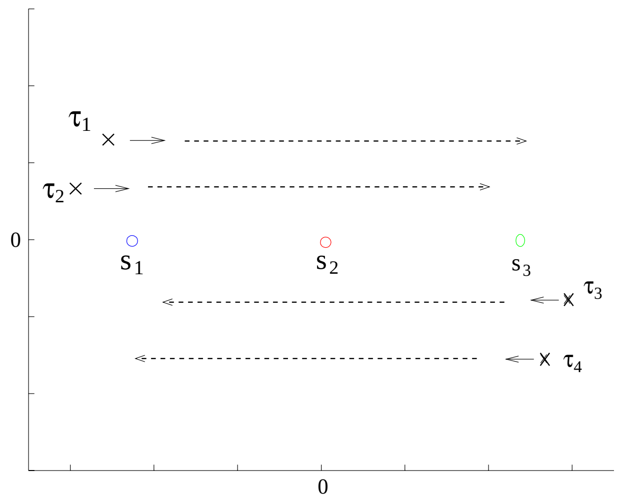


Figure 4.8. In this scenario, three stationary sensors (s_1, s_2 and s_3) track four targets (τ_1 to τ_4).

that the negotiation algorithm reached a result which was very close (99%) to the optimal algorithm in terms of *average total sensor information gain*.

A second criterion to observe was *concentration degree*, i.e., a measure of how well the targets are divided among the sensors. E.g., a target distribution in which all targets are tracked by a single sensor will yield a high concentration degree. This is an awkward situation since if the sensor that tracks all targets fails or is destroyed all targets will be lost. Thus, a low concentration degree, representing that targets are divided evenly among the sensors, is desired. In the previous work, the concentration degree turned out to be about 10% better for the negotiation algorithm compared to the optimal algorithm in a simulation.

In the current work, for every completed negotiation, we compare the result of the negotiation algorithm (i.e., an assignment of sensors to targets) with the result that the optimal algorithm would have yielded in the same situation. In our experiment, we want to compare the following criteria for our negotiation algorithm and an optimal one,

Total reward This is the value of the sum of the target rewards, i.e., $\sum_j r_j$. Of course, a high value of total reward corresponds to a good overall tracking performance and is desirable.

Redundancy This is the number of targets that are being tracked by one or more sensors. For those targets that are being tracked by one or more sensors, we have redundant measurements which can be fused. This is wanted for that reason, but also because if one sensor fails or is destroyed, the other sensor(s) will still receive measurements. If only one sensor tracks a target, if that sensor is lost so is the target. A high value, while preserving high total reward, is desirable.

Lost targets This is a value of the average number of targets lost if one of the sensors fails or is destroyed. A low value is desired.

In the following simulation, the three stationary sensors track the four targets over five hundred rounds of negotiations. The targets are moving fast and the sensors are re-negotiating their target assignments (i.e., starting a new round of negotiations) in every time step (perhaps every second or so).

Figure 4.9 shows the result of the negotiation algorithm (`N-tracker`) and Figure 4.10 the result of the optimal algorithm (`O-tracker`). In each diagram, the x-axis is time and y-axis which targets are being tracked by the sensor corresponding to the diagram. As we can see, the results of the two algorithms appear to be very similar.

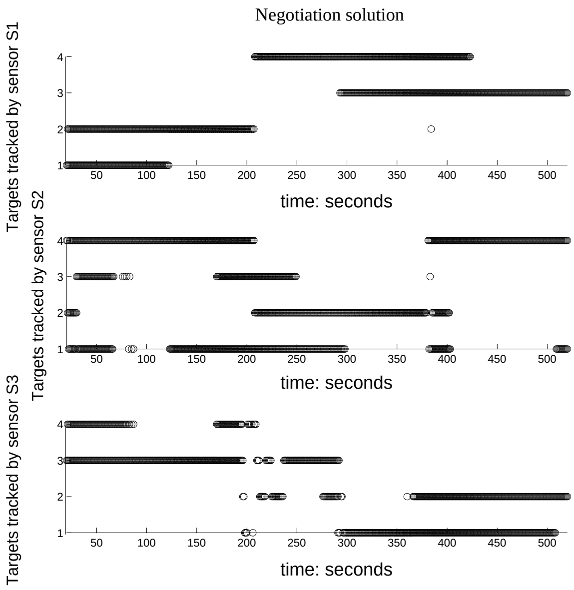


Figure 4.9. The diagrams show the target allocations of all three sensors for every time step in the negotiation.

In Figure 4.11, we see three diagrams. The topmost diagram depicts the relative reward of the `N-tracker` in every time step, i.e., the reward of the `N-tracker` divided by the reward of the `O-tracker`. Of course, the `N-tracker` will never receive as much reward as the `O-tracker`, but its rewards are certainly comparable.

The middle diagram depicts the differences in redundancy between the two algorithms. In every time step, the redundancy of the `O-tracker` is subtracted from the redundancy of the `N-tracker`. As shown, most of the time, the difference

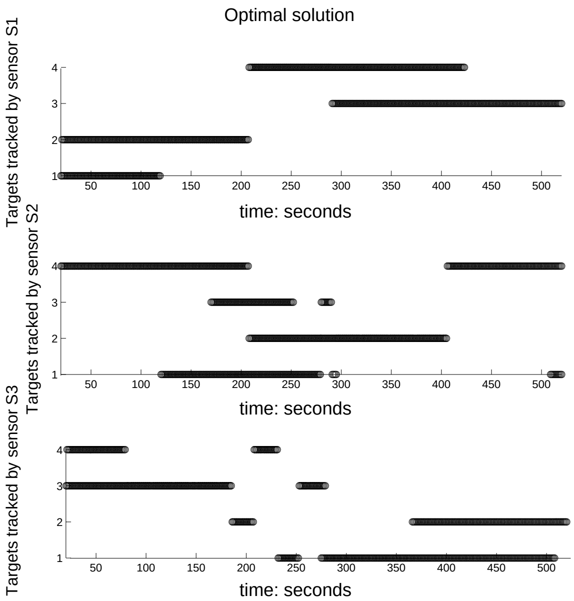


Figure 4.10. The diagrams show the optimal target allocations of all three sensors for every time step in the negotiation.

is zero, i.e., the two algorithms have the same redundancy. However, quite often the `N-tracker` has a greater redundancy and only during a few time steps the `O-tracker` has a greater redundancy.

The bottommost diagram shows the average number of lost targets (if one sensor is destroyed) plotted for both algorithms for every time step. The values of the `N-tracker` is plotted with a dotted line and the values of the `O-tracker` is plotted with a dashed line. We see that the results seem to coincide with the redundancy diagram, i.e., the `O-tracker` outperforms the `N-tracker` only in a few time steps.

Our experiments with stationary sensors show that the negotiation algorithm yields near-optimal tracking quality while improving robustness to sensor failure.

4.5.4 Target tracking with mobile sensors

Now, we introduce mobile negotiating sensors and wish to evaluate their performance. We here use the utility function in Section 4.4.1, but we will for now assume that the terrain has no effect on the negotiation.

For evaluation, we make two types of comparisons:

- For every sensor to target assignment the negotiation algorithm produces, we compare it to an optimal reward one (just like in the stationary case).

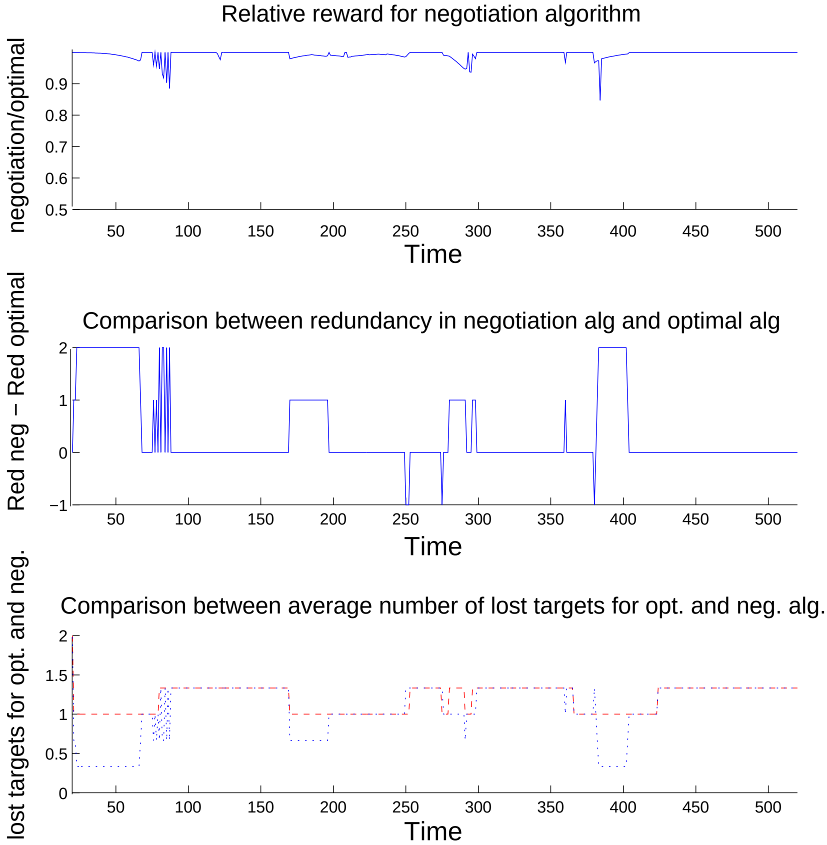


Figure 4.11. In these diagrams, we compare the results of the `O-tracker` and the `N-tracker`. The topmost diagram shows the relative reward of the `N-tracker` compared to the `O-tracker`. The middle diagram shows the difference in redundancy between both algorithms in every time step. The bottommost diagram plots the average number of lost targets (if one sensor fails) for both algorithms (the dotted line corresponds to the result of the `N-tracker`).

- We design and implement a "greedy" tracker (`G-tracker`) which operates independently of the negotiation based tracker (`N-tracker`).

We let the `G-tracker` reconsider the sensor to target assignment as often as the `N-tracker` does. After having selected the most optimal assignment, the sensors travel, at full speed, in the direction of the gradient. Whereas that seems reasonable, we will see in Section 4.5.5 what effects such an approach might have in a scenario where speed is dependent on terrain.

In our first simulation with mobile sensors, we want to know whether our reward function makes sensors try to fixate one target or if they tend to locate themselves where measurement performance on all targets is good. In Figure 4.12, two sensors track four targets. In this and the following figures that depict snapshots of target tracking with mobile sensors, the crosses are targets, the tiny circles are the mobile sensors, and the line that extends from the centre of each sensor indicates the current direction of motion of the sensor (it does not, however, indicate the speed of the sensor). Additionally, in some of the figures, dotted lines are drawn from sensors to targets. These lines clarify which sensors are tracking which targets.

The simulation starts at time $t = t_1$, and at this time the targets are divided between the two sensors in such a way that the upper sensor is willing to track the two upper targets and the lower sensor is willing to track the two lower targets. The upper targets are moving upwards and the lower targets are moving downwards. We see that the sensors, which in this simulation have the ability to catch up with the targets, prefer to situate themselves in between the targets.

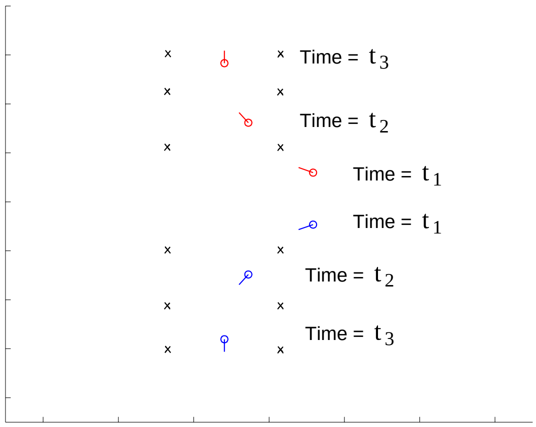


Figure 4.12. The Figure shows three superimposed snapshots, at times t_1, t_2 and t_3 ($t_1 < t_2 < t_3$), of a scenario where two sensors track two targets each.

In our next experiment, we study a scenario where the `G-tracker` runs into problems. In this case, sensor s_1 (in Figure 4.13(a)) wants to track the targets τ_1 and τ_2. However, they move in opposite directions, leaving s_1 with a resulting zero gradient, i.e., s_1 gets stuck while the targets move away (as seen in Figure 4.13(b)).

Sensor s_2 on the right has a similar problem since its targets are also moving in opposite directions. After a while, however, the G-tracker assigns targets τ_1 and τ_3 to sensor s_1 and the others to s_2, allowing sensor s_1 to escape from its deadlock. If we align targets τ_3 and τ_4 with sensor s_2 and rerun the simulation, we can actually make both sensors get stuck forever.

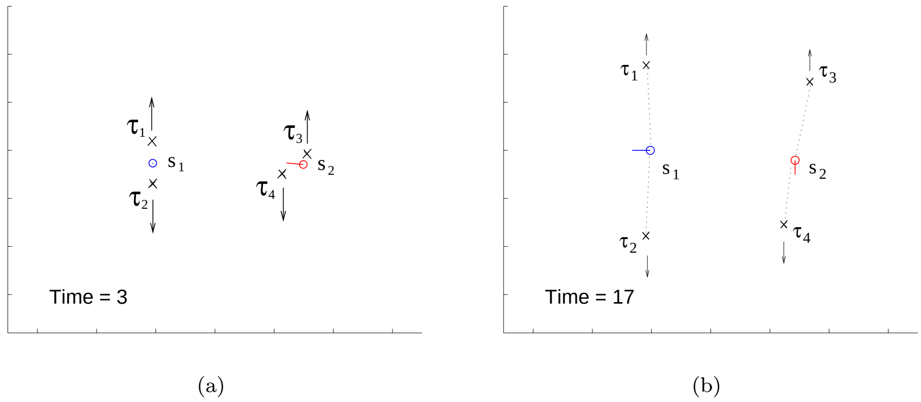
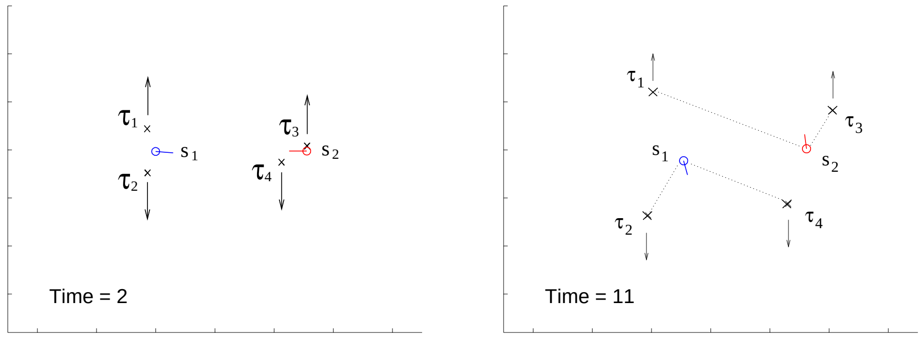


Figure 4.13. **(a)** In this scenario, two sensors s_1 and s_2 track four targets τ_1 to τ_4. Targets τ_1 and τ_3 are moving upwards and τ_2 and τ_4 downwards. Initially, the G-tracker assigns τ_1 and τ_2 to s_1 and τ_3 and τ_4 to s_2. **(b)** After some time, the targets have moved, but due to the "greedy" allocation of targets to sensors, the sensors are stuck between their assigned targets and have hardly moved.

The N-tracker, run on the same scenario, yields a more appealing result. To begin with, we see that the negotiation brings about a somewhat surprising assignment of targets to sensors (Figure 4.14(a)); s_1 tracks τ_3 and τ_4, and s_2 the other two, contrary to the allocation of the G-tracker (see once again Figure 4.13(a)). The reason is of course that the "greedy" allocation yields very low directional derivatives which allows the N-tracker to reach other solutions.

After a short while, sensor s_1 starts to follow the targets τ_2 and τ_4 that are moving downwards, and the other two are followed by sensor s_2 (Figure 4.14(b)).

In Figure 4.15, we compare the results of the N-tracker and G-tracker in terms of reward. At time $t = 10$, the N-tracker decides that sensor s_1 should track targets τ_2 and τ_4 and quickly receives a total reward which is greater than that of the G-tracker. At time $t = 27$, also the G-tracker decides that one sensor should track the targets moving upwards and the other the ones going downwards. However, as we can see from the rewards in the figure, the G-tracker is unable to catch up with the N-tracker. Since the targets in this scenario are allowed to travel at a higher speed than the sensors, the reward drops rapidly and at time $t = 40$ and beyond, both algorithms receive very low rewards.



(a) (b)

Figure 4.14. **(a)** Initially, the negotiation algorithm assigns targets τ_3 and τ_4 to sensor s_1 and the rest to sensor s_2. **(b)** After some time, the negotiation algorithm assigns targets τ_2 and τ_4 to sensor s_1 and the rest to sensor s_2.

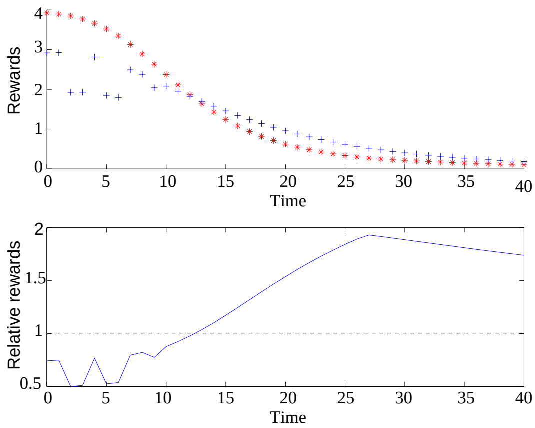


Figure 4.15. This graph compares the total rewards of the `G-tracker` and the `N-tracker` to each other. The absolute reward has been plotted in the top graph ('*' for `G-tracker`and '+' for `N-tracker`. In the lower graph, the relative reward of the `N-tracker` compared to the `G-tracker` has been plotted. For the first time steps, the `G-tracker` outperforms the negotiation one. At time $t = 10$, the negotiation assigns targets τ_2 and τ_4 to sensor s_1 which results in an increase in performance compared to the `G-tracker`. At time $t = 27$ the `G-tracker` comes to the same conclusion, which explains the negative slope of the curve.

In the final experiment of this section, we once again study the scenario in Figure 4.8. However, this time the sensors are mobile and both distances to targets and speed of targets have been decreased so that the sensor can take advantage of their mobility (i.e., it is not beneficial to use mobile sensors if their maximum speed is relatively low compared to the targets).

We run both the `G-tracker` and the `N-tracker` and compare the results in Table 4.1. We see that the `N-tracker` loses in measurement accuracy (its average measurement performance was 90% of that of the `G-tracker`). However, the `N-tracker` instead impresses by its robustness with an average of 1.39 targets being tracked by one or more sensors and average of 0.87 (27% better than the result of the `G-tracker`) of lost targets if one sensor is lost. The reason for this result is that the sensors, through the negotiation, are forced to share targets with each other, and, hence, yield better robustness for the target tracking system as a whole.

Table 4.1. Comparison between `G-tracker` and `N-tracker`

	G-tracker	**N-tracker**	**Relative**
Reward	3.7372	3.3765	0.90
Redundancy	0.4510	1.3922	3.09
Lost targets	1.1830	0.8693	0.73

4.5.5 Mobile tracking with terrain considerations

Until now, we have not considered terrain effects on mobile sensors in our experiments. Since it is highly unlikely that the designer of a mobile sensor system can expect a homogeneous environment, we need to consider varying terrain and its effects. In Section 4.4.4, we discussed how a so-called terrain function can be used to discount the directional derivative generated by a certain assignment.

In the scenario in Figure 4.16, we have put an *obstacle* into the environment. This obstacle has the property that when a mobile sensor tries to cross it, the maximum speed of the sensor reduces drastically. Such an obstacle represents, for instance, rough terrain or a steep hill. In this example, the speed reduces to 30% of the maximum speed it could achieve in an ideal terrain. Close to the obstacle, the terrain function discounts directional derivatives that lead into the obstacle.

We notice that the `G-tracker`, which does not consider terrain, leads the sensors straight into the obstacle, as shown in Figure 4.17(a). As a result of this, the sensors lose touch with the targets. In the case of the negotiation algorithm, the sensors switch targets close to the border of the obstacle, as shown in Figure 4.17(b). One reason for this is that offers that give directions that lead into the obstacle get small derivatives and are suppressed.

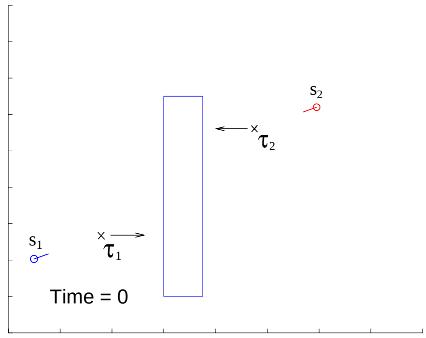
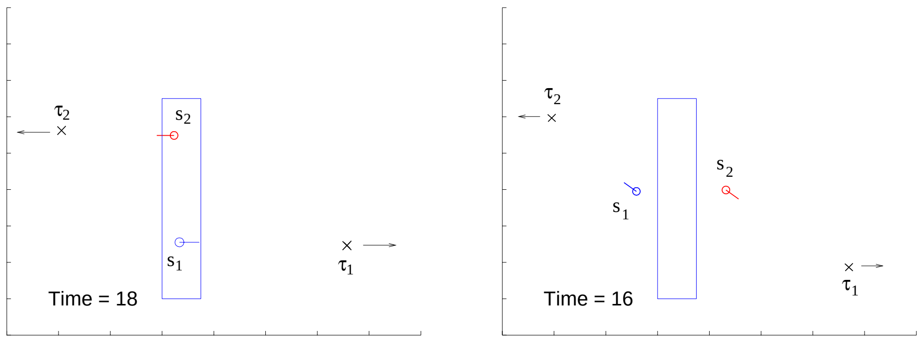


Figure 4.16. Initially, sensor s_1 tracks target τ_1 and sensor s_2 target τ_2. The rectangle represents an area which slows down mobile sensors that enter it.



(a) (b)

Figure 4.17. **(a)** The `G-tracker` does not consider terrain and leads the sensors into the obstacle, where they are slowed down considerably. **(b)** The negotiation algorithm decides to switch targets between the sensors instead.

4.6 Discussion

There are a number of parameters that can be altered that affect the behaviour of the target tracking system. α_j, in Equation 4.10, influences the target reward and reflects the value the system assigns to increased accuracy in measurements. As can be seen in Figure 4.2, by varying the value of α_j, we can customise the value of increase measurement accuracy.

Also included in Figure 4.2 is α_i which is a parameter in the sensor net reward function (Equation 4.12). By varying α_i between 0 and 1, we can modify the ability of sensor agents to differentiate between rewards of offers. For $\alpha_i = 0$ the ability of the agent to differentiate between rewards is at its maximum, but for $\alpha_i = 1$ all offers appear to the sensor agent to have same reward. This can be understood by once again studying Figure 4.2 where α_i is set to 0.3 and imagining the effect of increasing the α_i value.

The negotiation procedure can be adjusted in several ways, e.g., by 1) altering the order of offers, by 2) changing the way the number of valid offers is reduced during a negotiation, and by 3) changing the number of steps in the negotiation.

To reiterate, as we explained in Section 4.2.1, due to the facts that the agents are benign and have complete information about the others, the agent that begins the negotiation can calculate an offer which all other agents will accept. Hence, when we talk about negotiation in the following discussion, we are referring to the search procedure by which the offer, which all agents will accept, is found.

The first way to adjust the negotiation procedure involves deciding on a policy for in which order agents should make their offers (this is the $P(H)$ function of the game definition in Section 4.2.1). Naturally, the agent who makes the first offer has an advantage. There are several ways to do this, one might for instance want the agent that received the best/worst reward in the previous round of negotiations to start and the others to follow in increasing/decreasing order. In most of our experiments, we used another policy which we considered to be more fair. We let all agents have the advantage of commencing the negotiations about the same amount of times each. In a round-robin fashion, one of the agents, say a_1, started a round of negotiations, another agent, a_2, made the second offer, and a third agent, a_3, made the third offer, etc.

Quite often, the number of steps in the negotiation was larger than the number of agents (allowing every agent to participate in the negotiation), and in those cases, when the last agent had made its first offer, the first agent, a_1, continued. In the next round of negotiations, it was a_2's turn to commence the negotiation, followed by agent a_3, and so on.

The second way to adjust the negotiation procedure is to change how the set of valid offers evolves during a round of negotiations. Valid offers are offers, o, that do not violate the requirements of Equation 4.1. In the previous work, the composition of the set of offers was heavily dependent on the utility functions of the agents. If the utility of the offers for all agents, given by Equation 4.7, differed only slightly, then even for many negotiation steps, the set of valid offers would still be considerable. If so, the agent which commences the negotiation will have a great influence over the outcome. It might be debated whether this is a disadvantage or not. On the one hand, the commencing agent will be very powerful, possibly completely ignoring the wishes of the other agents, which might be undesirable. On the other hand, that the offers have similar values to an agent could be interpreted as indifference on the behalf of the agent. Thus, in that case, the agent simply does not care about the outcome of the negotiation.

In this work, we experimented with another approach that systematically reduces the set of valid offers with each step of the negotiation, guaranteeing that the agent that begins the negotiation will have a more restricted set of offers to choose from. Starting from a larger set of possible offers, $\eta = \frac{\#\text{offers}}{\#\text{negotiation steps}}$ number of offers are excluded from the negotiation (i.e., becomes invalid) with every step in the negotiation, ensuring that there are only η offers left to choose from in the end.

To explain by which criterion offers are removed, consider Figure 4.18. The figure illustrates the common situation where offers have different utility for different agents. Assume that an offer, o, has been proposed at some step of the negotiation. Now, every other offer o' has a certain distance in utility to o for every agent i, $dist_i(o, o') = |U_i(o) - U_i(o')|$. Let the maximum distance of all agents for an offer, o', be $dist(o, o') = max_i dist_i(o, o')$. Finally, to decrease the set of offers with every step of the negotiation, the η offers with the largest $dist(o, o')$ are removed. Despite the appealing property of this approach, i.e., the strict monotonic decrease of the size of the offer set, we are not yet convinced by its positive effect on the negotiations.

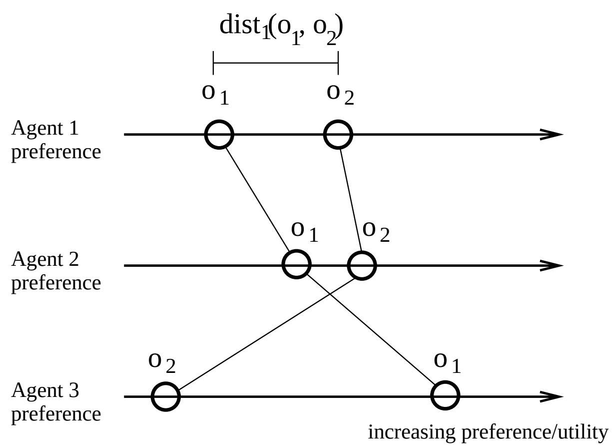


Figure 4.18. Different offers have different values to different agents. For instance, in this example, $o_1 \succ_1 o_2$ but $o_2 \succ_3 o_1$

The third way to affect the negotiation is to change the number of steps of the negotiation. In our experiments, we used quite lengthy negotiations of at least 30 steps, sometimes many more. The outcome of the negotiation, is under some circumstances, very much dependent on the exact number of negotiation steps. This potential problem is to some extent avoided when the systematic approach to decreasing the set of valid offers, presented in the previous paragraph, is applied. Generally, the more negotiation steps used to reach an agreement, the more "democratic" is the outcome of the negotiation.

Finally, we find our solution to the multiple-objective optimisation problem, that arose in Section 4.4.1, intriguing and it encourages to further investigation.

4.7 Conclusion and future work

In this chapter, we have presented a game theoretic model for allocating targets to mobile sensors. Sensor agents negotiate by proposing offers of allocations that involve all sensors. Each agent can evaluate each offer to decide its individual utility.

The utility is composed of two objectives: sensor reward and directional derivative. The first objective, sensor reward, is dependent on the distance between sensor and targets, the number of targets the sensor is concurrently tracking, and whether other sensors track the same target. The other part, directional derivative, is directly calculated from the allocation of the offer, or, when terrain conditions are considered, by discounting derivatives in inconvenient directions.

We showed, in the experiments in Section 4.5, two interesting properties of our negotiation algorithm: first, the negotiation forces sensors to share targets, improving robustness to the target tracking system (e.g., the scenario in Figure 4.8). Secondly, considering directional derivatives allow sensors to pro-actively reconsider target assignments, possibly improving long-term information gain (e.g., as in Figures 4.14(b) and 4.17(b)).

Further studies should investigate under what circumstances these properties imply advantages to the target tracking system. With the support of these early results, we anticipate interesting discoveries in our future exploration of negotiation-based, distributed sensor management.

Some of the most salient, concrete directions for future studies are:

- introduction of uncertainty (e.g., in target or sensor state) into the negotiations,
- prediction of (near) future target and sensor states to improve tracking performance,
- to explore and devise a policy to select negotiation strategy depending on the state of the environment.

A Simulation parameters

The H_{ij} matrices in the Kalman measurement equation (Equation 4.2) used here are all equal:

$$H_{ij} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

The measurement error covariance matrices R_{ij} are also assumed to be identical:

$$R_{ij} = \begin{bmatrix} \sigma_{tot}^2 & 0 \\ 0 & \sigma_{tot}^2 \end{bmatrix},$$

We here assume that the standard deviations for measurements, σ_{tot}, for both measurement components, e.g., x- and y- coordinates, are equal and independent.

B Gradient derivation

Recall that the components of the gradient, depicted in Equation 4.16, are the partial derivatives of the measurement reward function in Equation 4.13. Important to notice is that both factors of the measurement reward function, γ_{ij} and $r_j(S_j)$, are dependent on the derivation variables, x_i and y_i.

The partial derivatives, $\frac{\partial r_i^m}{\partial x_i}$ and $\frac{\partial r_i^m}{\partial y_i}$, are similar, and, thus, we show only the detailed calculation of $\frac{\partial r_i^m}{\partial x_i}$ and claim that the calculation of other derivative is almost identical.

$$\frac{\partial}{\partial x_i}\left(r_i^m\right) = \quad \{\text{since } D(fg) = fg' + f'g\} = \sum_j \gamma_{ij} \frac{\partial}{\partial x_i}\left(r_j\right) + \frac{\partial}{\partial x_i}\left(\gamma_{ij}\right) r_j \quad (4.20)$$

We know r_j and γ_{ij} from Equation 4.14 and Equation 4.10, respectively. However, $\frac{\partial}{\partial x_i}\left(r_j\right)$ and $\frac{\partial}{\partial x_i}\left(\gamma_{ij}\right)$ have yet to be determined.

$$\begin{aligned} \frac{\partial}{\partial x_i}\left(r_j\right) = \quad & \alpha_j \frac{\partial}{\partial x_i}\left(g_j(S_j)\right) e^{\alpha_j g_j(S_j)} = \left\{\text{since } e^{\alpha_j g_j(S_j)} = 1 - r_j\right\} = \\ & \alpha_j \frac{\partial}{\partial x_i}\left(g_j(S_j)\right)\left(1 - r_j\right) \end{aligned} \quad (4.21)$$

$$\begin{aligned} \frac{\partial}{\partial x_i}\left(g_j(S_j)\right) = \quad & \left\{g_j(S_j) = \textstyle\sum_{k \in S_j} \|H_{kj}^T R_{kj}^{-1} H_{kj}\| \text{ from Equation 4.14}\right\} = \\ & \frac{\partial}{\partial x_i}\left(\textstyle\sum_{k \in S_j} \|H_{kj}^T R_{kj}^{-1} H_{kj}\|\right) = \\ & \left\{\frac{\partial}{\partial x_i}\left(\|H_{kj}^T R_{kj}^{-1} H_{kj}\|\right) = 0, \forall k {\neq} i\right\} = \frac{\partial}{\partial x_i}\left(\|H_{ij}^T R_{ij}^{-1} H_{ij}\|\right) \end{aligned} \quad (4.22)$$

The matrices H_{ij} and R_{ij} used are described in Appendix A, but in order to calculate the partial derivative $\frac{\partial}{\partial x_i}\left(\|H_{ij}^T R_{ij}^{-1} H_{ij}\|\right)$ we also have to decide which matrix norm to use. In this work we use the Frobenius norm, $\|\cdot\|_F$, which considers every element of the matrix.[9]

$$\|H_{ij}^T R_{ij}^{-1} H_{ij}\|_F = \left\| \begin{bmatrix} \sigma_{tot}^{-2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{tot}^{-2} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \right\|_F = 2^{1/2}\sigma_{tot}^{-2} \tag{4.23}$$

Thus, the partial derivative then becomes

$$\begin{aligned} \frac{\partial}{\partial x_i}\left(\|H_i^T R_{ij}^{-1} H_i\|_F\right) = \; & -2^{3/2}\sigma_{tot}^{-3}\frac{\partial}{\partial x_i}(\sigma_{tot}) = \{\text{using } \sigma_{tot} \text{ in Eq. 4.19}\} = \\ & -2^{5/2}c(\sigma_{min}n)^{-2}(1+cd^2)^{-3}(x_i - x_j) \end{aligned} \tag{4.24}$$

With the result achieved in Equation 4.24 the derivative in Equation 4.21 can finally be calculated,

$$\begin{aligned} \frac{\partial}{\partial x_i}(r_j) = \; & \alpha_j \frac{\partial}{\partial x_i}(g_j(S_j))(1 - r_j) = \\ & -2^{5/2}\alpha_j c(\sigma_{min}n)^{-2}(1+cd^2)^{-3}(x_i - x_j)(1 - r_j) \end{aligned} \tag{4.25}$$

Now only $\frac{\partial}{\partial x_i}(\gamma_{ij})$ is missing to complete the calculation of Equation 4.20.

$$\begin{aligned} \frac{\partial}{\partial x_i}(\gamma_{ij}) = \; & \frac{\partial}{\partial x_i}\left(\frac{g_j(i)}{g_j(S_j)}\right) = \left\{\text{since } D(\tfrac{f}{g}) = \frac{f'g - fg'}{g^2}\right\} = \\ & \frac{\frac{\partial}{\partial x_i}(g_j(i))g_j(S_j) - g_j(i)\frac{\partial}{\partial x_i}(g_j(S_j))}{g_j(S_j)^2} = \\ & \left\{\text{since } \frac{\partial}{\partial x_i}(g_j(S_j)) \equiv \frac{\partial}{\partial x_i}(g_j(i))\right\} = \\ & \frac{\frac{\partial}{\partial x_i}(g_j(i))}{g_j(S_j)}\left(1 - \frac{g_j(i)}{g_j(S_j)}\right) = \frac{\frac{\partial}{\partial x_i}(g_j(i))}{g_j(S_j)}(1 - \gamma_{ij}) \end{aligned} \tag{4.26}$$

The partial derivative $\frac{\partial}{\partial x_i}(\gamma_{ij})$ is now completely known since γ_{ij} and $g_j(S_j)$ are known from Equation 4.14 and $\frac{\partial}{\partial x_i}(g_j(i))$ from Equation 4.22.

Now, the partial derivative of the measurement reward function, $\frac{\partial}{\partial x_i}(r_i^m)$, is completely determined by inserting Equations 4.21 and 4.26 into Equation 4.20.

The other partial derivative, $\frac{\partial}{\partial y_i}(r_i^m)$, and is equivalent except for only the factor $(x_i - x_j)$ in Equation 4.25 which should be replaced with $(y_i - y_j)$.

[9] The Frobenius norm of a $m \times n$ matrix A with cell elements a_{ij} is $\|A\|_F \triangleq \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2}$.

References

[AAB+01] A. Appriou, A. Ayoun, S. Benferhat, P. Besnard, Isabelle Bloch, L. Cholvy, R. Cooke, F. Cuppens, D. Dubois, H. Fargier, M. Grabisch, Anthony Hunter, R. Kruse, J. Lang, S. Moral, H. Prade, A. Saffiotti, P. Smets, and C. Sossai. Fusion: General concepts and characteristics. *International Journal of Intelligent Systems*, 16(10):1107–1134, 2001.

[AABW00] S. Arnborg, H. Artman, J. Brynielsson, and K. Wallenius. Information awareness in command and control: Precision, quality, utility. In *Proceedings of the 3rd International Conference on Information Fusion* [Int00], pages 25–32.

[Adr93] R.A. Adrian. Sensor management. In *Proceedings, AIAA/IEEE Conference on Digital Avionics System*, pages 32–37. IEEE Computer Society Press, 1993.

[AG92] Mongi A. Abidi and Rafael C. Gonzalez, editors. *Data Fusion In Robotics and Machine Intelligence*. Academic Press, San Diego, 1992.

[Alb99] James S. Albus. The engineering of mind. *Informatino Sciences*, 117:1–18, 1999.

[AMGC02] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2):174–188, February 2002.

[Ark98] Ronald C. Arkin. *Behavior-Based Robotics*. MIT Press, 1998.

[AWB87] Y. Aloimonos, I. Weiss, and A. Bandopadhay. Active vision. In *Proceedings of the first IEEE International Conference of Computer Vision*, pages 35–54, 1987.

[Baj88] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):996–1005, August 1988.

[Ben83] Paul S. Bender. *Resource Management - An alternative view of the management process*. John Wiley and Sons, 1983.

[Ben00] Kaouthar Benameur. On the determination of active and passive measurement strategies. In *Proceedings of the 3rd International Conference on Information Fusion* [Int00].

[Ben02] Abder Benaskeur. Sensor management in command and control. In *Proceedings of the 7th International Command and Control Research and Technology Symposium*, September 2002.

[Ber99] Niclas Bergman. *Recursive Bayesian Estimation Navigation and Tracking Applications*. PhD thesis, Linköping University, 1999.

[BG97] John Budenske and Maria Gini. Logical sensor/actuator: knowledge-based robotic plan execution. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(2-3):361–374, 1997.

[BH00] Erik P. Blasch and Phil Hanselman. Information fusion and information superiority. In *Proceedings of the IEEE National Conference on Aerospace and Electronics*, pages 290–297. IEEE, 2000.

[BO98] M. Bedworth and J. O'Brien, editors. *Proceedings, EuroFusion98 International Conference on Data Fusion*, 1998.

[BP99a] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems*. Artech House, Norwood, MA, 1999.

[BP99b] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems*, chapter 15, pages 967–1068. In [BP99a], 1999.

[BPP99] H. Borotschnig, L. Paletta, and A. Pinz. A comparison of probabilistic, possibilistic and evidence theoretic fusion schemes for active object recognition. *Computing*, 62(4):293–319, 1999.

[BS01] Christopher L. Bowman and Alan N. Steinberg. *Handbook of Data Fusion*, chapter 16, pages 16–1 – 16–39. In Hall and Llinas [HL01], 2001.

[BSF88] Yaakov Bar-Shalom and Thomas E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[BSL93] Yaakov Bar-Shalom and Xiao-Rong Li. *Estimation and Tracking: Principles, Techniques and Software*. Artech House, Boston, 1993.

[BVA03] Hamid R. Berenji, David Vengerov, and Jayesh Ametha. Co-evolutionary perception-based reinforcement learning for sensor allocation in autonomous vehicles. In *Proceedings of the 12th IEEE International Conference on Fuzzy Systems*. IEEE, 2003.

[Cas97] David A. Castañon. Approximate dynamic programming for sensor management. In *Proceedings of the 1997 Conference on Decision and Control*, pages 1202–1207, 1997.

[CFK97] Y. Uny Cao, Alex S. Fukunaga, and Andrew B. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4:7–27, 1997.

[CGH96] Diane J. Cook, Piotr Gmytrasiewicz, and Lawrence B. Holder. Decision-theoretic cooperative sensor planning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10):1013–1023, 1996.

[CLOHd+01] Jr. Charles L. Ortiz, Eric Hsu, Marie desJardins, Timothy Rauenbusch, Barbara Grosz, Osher Yadgar, and Sarit Kraus. Incremental negotiation and coalition formation for resource-bounded agents. Preliminary report, 2001. AAAI Fall Symposium on Negotiation Methods for Autonomous Cooperative Systems.

[CNR01] Marc Contat, Vincent Nimier, and Roger Reynaud. An ordering method to select a sensor and its mode in a multitaget environment. In *Proceedings of the 4th International Conference on Information Fusion* [Int01].

[CNR02] Marc Contat, Vincent Nimier, and Roger Reynaud. Request management using contextual information for classification. In *Proceedings of the 5th International Conference on Information Fusion* [Int02], pages 1147–1153.

[Cre85] Martin L. Van Creveld. *Command in war*. Harvard University Press, Cambridge, MA, USA, 1985.

[DALH94] R.V. Denton, E.I. Alcaraz, J. Llinas, and K.J. Hintz. *Science of Command and Control: Part III - Coping with change*, chapter 13, pages 119–134. AFCEA International Press, Fairfax, Virginia, 1994.

[dDJW99] Marie E. desJardins, Edmund H. Durfee, Charles L. Ortiz Jr., and Michael J. Wolverton. A survey of research in distributed, continual planning. *AI Magazine*, 20(4):13–22, 1999.

[DN01] Pierre Dodin and Vincent Nimier. Distributed resource allocation under communication constraints. In *Proceedings of the 4th International Conference on Information Fusion* [Int01].

[DVAD02] Arnaud Doucet, Ba-Ngu Vo, Christophe Andrieu, and Manuel Davy. Particle filtering for multi-target tracking and sensor management. In *Proceedings of the 5th International Conference on Information Fusion* [Int02], pages 474–481.

[DW88] Hugh F. Durrant-Whyte. *Integration, Coordination and control of Multi-sensor robot systems.* Kluwer Academic Publishers, Norwood, MA, 1988.

[FF98] Carlos M. Fonseca and Peter J. Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms-part I: A unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics*, 28(1):473–509, January 1998.

[GD00] Piotr J. Gmytrasiewicz and Edmund H. Durfee. Rational coordination in multi-agent environments. *Autonomous Agents and Multi-Agents Systems*, 3(4):319–350, 2000.

[GH89] Rod Grupen and Thomas C. Henderson. *Traditional and Non-Traditional Robotic Sensors.* Springer-Verlag, Berlin, September 1989.

[GR98] Paul G. Gonsalves and Gerard J. Rinkus. Intelligent fusion and asset management processor. In *Proceedings of the 1998 IEEE Information Technology Conference*, pages 15–18. IEEE, 1998.

[Hag90] Gregory D. Hager. *Task-Directed Sensor Fusion and Planning - A Computational Approach.* Kluwer Academic Publishers, Norwell, MA, 1990.

[Hal92a] David L. Hall. *Mathematical Techniques in Multisensor Data Fusion*, chapter 1, pages 1–34. In [Hal92b], 1992.

[Hal92b] David L. Hall. *Mathematical Techniques in Multisensor Data Fusion.* Artech House, Norwood, MA, 1992.

[Har90] Stevan Harnad. The symbol grounding problem. *Physica D*, 42:335–346, 1990.

[HKBS02] M. L. Hernandez, T. Kirubarajan, and Y. Bar-Shalom. Efficient multi-sensor resource management using cramér. In *Proceedings of the SPIE International Conference on Signal and Data Processing of Small Targets.* SPIE, 2002.

[HL01] David L. Hall and James Llinas, editors. *Handbook of Data Fusion.* CRC Press, 2001.

[HM97] Geir E. Hovland and Brenan J. McCarragher. Dynamic sensor selection for robotic systems. In *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pages 272–277. IEEE, 1997.

[HM99] K.J. Hintz and G.A. McIntyre. Goal lattices for sensor management. In *Proceedings of the SPIE International Symposium on Aerospace/Defence Sensing & Control*, volume 3720, pages 249–255. SPIE, 1999.

[HMS01] David Hand, Heikki Mannila, and Padhraic Smyth. *Principles of data mining*. MIT Press, 2001.

[How66] Ronald A. Howard. Information value theory. *IEEE Transactions on systems science and cybernetics*, 2:22–26, 1966.

[HS84] Tom Henderson and Esther Shilcrat. Logical sensor systems. *Journal of Robotic Systems*, 1:169–193, 1984.

[Hum97] Mark Humphry. *Action Selection using Reinforcement Learning*. PhD thesis, University of Cambridge, 1997.

[HVM+01] Bryan Horling, Régis Vincent, Roger Mailler, Jiaying Shen, Raphen Becker, Kyle Rawlins, and Victor Lesser. Distributed sensor network for real time tracking. In *Proceedings of the 5th International Conference on Autonomous Agents*, pages 417–424, Montreal, June 2001. ACM Press.

[Int00] International Society of Information Fusion. *Proceedings of the 3rd International Conference on Information Fusion*, 2000.

[Int01] International Society of Information Fusion. *Proceedings of the 4th International Conference on Information Fusion*, 2001.

[Int02] International Society of Information Fusion. *Proceedings of the 5th International Conference on Information Fusion*, 2002.

[Int03] International Society of Information Fusion. *Proceedings of the 6th International Conference on Information Fusion*, 2003.

[JX03] Ronnie Johansson and Ning Xiong. Perception management - an emerging concept for information fusion. *Information Fusion*, 4(3):231–234, 2003.

[JXC03a] Ronnie Johansson, Ning Xiong, and Henrik I. Christensen. A game theoretic model for management of mobile sensors. Report TRITA-NA-P03/06. ISSN 1101-2250. ISRN KTH/NA/P–02/12–SE. CVAP-278, Royal Institute of Technology, Numerical Analysis and Computer Science, Computer Vision and Active Perception Laboratory, 100 44 Stockholm, SWEDEN, 2003. http://www.nada.kth.se/cvap/cvaplop/lop-cvap.html.

[JXC03b] Ronnie Johansson, Ning Xiong, and Henrik I. Christensen. A game theoretic model for management of mobile sensors. In *Proceedings of the 6th International Conference on Information Fusion* [Int03], pages 583–590.

[KK03] Mehmet Karan and Vikram Krishnamurthy. Sensor scheduling with active and image-based sensors for maneuvering targets. In *Proceedings of the 6th International Conference on Information Fusion* [Int03], pages 1018–1023.

[KKI03] Chris Kreucher, Keith Kastella, and Alfred O. Hero III. Multi-target sensor management using alpha-divergence measures. In Feng Zhao and Leonidas J. Guibas, editors, *IPSN*, volume 2634 of *Lecture Notes in Computer Science*, pages 209–222. Springer, 2003.

[KLM96] Leslie Pack Kaelbling, Michael L. Littman, and Andrew P. Moore. Reinforcement learning: A survey. *Journal of AI Research*, 4:237–285, 1996.

[KP98] Michael Kalandros and Lucy Y. Pao. Controlling target estimate covariance in centralized multisensor systems. In *Proceedings of American Control Conference*, pages 2749–2753, 1998.

[KS00] Konrad Klein and Vitor Sequeira. View planning for the 3d modelling of real world scenes. In *Proceedings of the IEEE Conference on Intelligent Robots and Systems*, pages 943–948. IEEE, October-November 2000.

[KSRI02] Rajgopal Kannan, Sudipta Sarangi, Sibabrata Ray, and S. S. Iyengar. Minimal sensor integrity in sensor grids. In *Proceedings of the 5th International Conference on Information Fusion* [Int02], pages 959–964.

[Lat93] Jean-Claude Latombe. *Robot motion planning*. Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[Lee99] Sukhan Lee. *Control in Robotics and Automation - Sensor-based integration*, chapter 12, pages 347–380. Academic Press, San Diego, CA, 1999.

[LK92] R. C. Luo and M. G. Kay. *Data Fusion In Robotics and Machine Intelligence*, chapter 2, pages 7–135. In Abidi and Gonzalez [AG92], 1992.

[Lli88] James Llinas. *Application of artificial intelligence to command and control systems*, chapter 3.1, pages 126–184. Peter Peregrinus Ltd, London, UK, 1988.

[Low98] David Lowe. The relevance and application of information fusion to financial analysis. In Bedworth and O'Brien [BO98], pages 5–12.

[LRC95] J.M. Molina López, F.J. Jiménez Rodríguez, and J.R. Casar Corredera. Fuzzy reasoning for multisensor management. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1398–1403. IEEE, October 1995.

[Mah03] Ronald Mahler. Multisensor-multitarget sensor management: A unified Bayesian approach. In *Proceedings of the 2003 SPIE AeroSense Conference*, volume 5096. SPIE, 2003.

[McC98] Brenan J. McCarragher. *Control Problems in Robotics and Automation*, chapter Discrete Event Theory for the monitoring and control of robotic systems, pages 209–225. Springer Verlag London Limited, Great Britain, 1998.

[McI98] G.A. McIntyre. *A comparative approach to sensor management and scheduling*. PhD thesis, George Mason University, Fairfax, FA, 1998.

[MDW94] James Manyika and Hugh Durrant-Whyte. *Data Fusion and Sensor Management - a decentralized information-theoretic approach*. Ellis Horwood, 1994.

[MGV+02] R. Mavrantza, C. Gountromichou, C. Vamvoukakis, M. Kouli, I. Karfakis, M. Santamouri, K. Seymour-St., D. Argialas, V. Karathanassi, K. Papadimatou, E. Michailidou, and D. Rokos. Multi-sensor image and data fusion for investigating possible epithermal au in the aegean back-arc, hellas. In *Proceedings of the 2002 Conference on Fusion of Earth Data*, January 2002.

[Mit97] Tom Mitchell. *Machine learning*. McGraw-Hill, 1997.

[MM94] Stan Musick and Raj Malhotra. Chasing the elusive sensor manager. In *Proceedings of the National Conference on Aerospace and Electronics*, pages 606–613. IEEE, 1994.

[MMMW01] Bob Miller, Mary Ann Malloy, Ed Masek, and Chris Wild. Towards a framework for managing the information environment. *Information, knowledge, systems management*, 2:359–384, 2001.

[Mui90] Patrick Muir. A virtual sensor approach to robot kinematic identification. In *Proceedings of the 1990 IEEE International Conference on Systems Engineering*, pages 440–445. IEEE, August 1990.

[Nas53] John F. Nash. Two-person cooperative games. *Econometrica*, 21:128–140, 1953.

[Nas77] Jeffrey M. Nash. Optimal allocation of tracking resources. In *Proceedings of the IEEE Conference on Decision and Control*, volume 1, pages 1170–1180. IEEE, 1977.

[Nil98] Nils J. Nilsson. *Artificial Intelligence - A New Synthesis*, chapter 2, pages 21–35. Morgan Kaufmann Publishers Inc., 1998.

[NN00] G. W. Ng and K. H. Ng. Sensor management - what, why and how. *Information Fusion*, 1:67–75, 2000.

[OR94] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory.* MIT Press, 1994.

[PCCB97] Stéphane Paradis, Bruce A. Chalmers, Robert Carling, and Pierre Bergeron. Towards a generic model for situation and threat assessment. In Suresh B. Raja, editor, *Proceedings of SPIE, Digitization of the Battlefield II*, pages 171–182. SPIE, 1997.

[Pea88a] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, chapter 6, pages 289–331. In [Pea88b], 1988.

[Pea88b] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., 1988.

[Pen98] Dawn E. Penny. The automatic management of multi-sensor systems. In *Proceedings of the International Conference on Information Fusion*, pages 748–7559, 1998.

[PH03] Mark A. Perillo and Wendi B. Heinzelman. Optimal sensor management under energy and reliability constraints. In *Proceedings of the IEEE Wireless Communication and Networking Conference*, 2003.

[Pir98] Paolo Pirjanian. *Multi-objective Action Selection and Behavior fusion using voting.* PhD thesis, Aalborg University, 1998.

[PPGKB96] Axel Pinz, Manfred Prantl, Harald Ganster, and Hermann Kopp-Borotschnig. Active fusion - a new method applied to remote sensing image interpretation. *Pattern Recognition Letters*, 17:1349–1359, 1996.

[PRT98] Stéphane Paradis, Jean Roy, and Willem Treurniet. Integration of all data fusion levels using a blackboard architecture. In Bedworth and O'Brien [BO98], pages 195–202.

[PW00] Dawn E. Penny and M. Williams. A sequential approach to multi-sensor resource management using particle filters. In *Proceedings of the SPIE International Conference on Signal and Data Processing of Small Targets*, pages 598–609. SPIE, 2000.

[Rim93] Raymond D. Rimey. Control of selective perception using bayes nets and decision theory. Technical Report TR468, Computer Science Department, The University of Rochester, December 1993. citeseer.nj.nec.com/rimey93control.html.

[RN95] Stuart Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach*. Prentice Hall, Englewood Cliffs, New Jersey, 1995.

[Roy00] Sumantra Dutta Roy. *Active Object Recognition through Next View Planning*. PhD thesis, Indian Institute of Technology, 2000.

[RW91] Stuart Jonathan Russell and Eric Wefald. *Do the right thing : studies in limited rationality*. MIT Press, Cambridge, Mass., 1991.

[SB01] Alan N. Steinberg and Christopher L. Bowman. *Handbook of Data Fusion*, chapter 2. In Hall and Llinas [HL01], 2001.

[SD98] A. Starr and M. Desforges. Strategies in data fusion - sorting through the tool box. In Bedworth and O'Brien [BO98], pages 85–90.

[Sha48] Claude Elwood Shannon. A mathematical theory of communication. Technical report, Bell System Technical Journal, July 1948.

[Sha76] Glenn Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, USA, 1976.

[Sha87] M. Louis Shafer. *The Surveying Handbook*, chapter 10, pages 296–339. Van Nostrand Reinhold, 1987.

[SK98] Wayne Schmaedeke and Keith Kastella. Information based sensor management and IMMKF. In *Proceedings of the SPIE International Conference on Signal and Data Processing of Small Targets*, pages 390–401. SPIE, 1998.

[Sol99] Stephen W. Soliday. A genetic algorithm model for mission planning and dynamic resource allocation of airborne sensors, 1999.

[Ste99] Alan N. Steinberg. New developments: FUSIAC and JDL data fusion model revision. http://www.ic.arc.nasa.gov/ic/data99-workshop/NASAdatafusion/index.htm, 1999. OH-slides from 1999 NASA workshop on data fusion and data mining.

[TAT95] Konstantinos A. Tarabanis, Peter K. Allen, and Roger Y. Tsai. A survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, 11(1):86–104, February 1995.

[Vog97] Paul Vogt. Perceptual grounding in robots. In Andreas Birk and John Demiris, editors, *Proceedings of the 6th European Workshop on Learning Robots*, pages 36–45, 1997.

[Wei99] Gerhard I. Weiss, editor. *Multiagent Systems - A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.

[Whi98] Franklin E. White. Managing data fusion systems in joint and coalition warfare. In Bedworth and O'Brien [BO98], pages 49–52.

[WL90a] Edward Waltz and James Llinas. *Multisensor Data Fusion*, chapter Preface, pages xv–xvii. In [WL90e], 1990.

[WL90b] Edward Waltz and James Llinas. *Multisensor Data Fusion*, chapter 2, pages 15–47. In [WL90e], 1990.

[WL90c] Edward Waltz and James Llinas. *Multisensor Data Fusion*, chapter 1, pages 1–13. In [WL90e], 1990.

[WL90d] Edward Waltz and James Llinas. *Multisensor Data Fusion*, chapter 5, pages 129–158. In [WL90e], 1990.

[WL90e] Edward Waltz and James Llinas. *Multisensor Data Fusion*. Artech House, Norwood, Massachussets, 1990.

[Woo99] Michael Wooldridge. *Multiagent Systems - A Modern Approach to Distributed Artificial Intelligence*, chapter 1, pages 27–78. In Weiss [Wei99], 1999.

[XCS03] Ning Xiong, Henrik I. Christensen, and Per Svensson. Negotiation of target distribution enhancing system survivability. Report CVAP-274, Royal Institute of Technology, Numerical Analysis and Computer Science, Computer Vision and Active Perception Laboratory, 100 44 Stockholm, SWEDEN, 2003. http://www.nada.kth.se/cvap/cvaplop/lop-cvap.html, submitted to IEEE Transactions on Systems, Man, and Cybernetics.

[XS02] Ning Xiong and Per Svensson. Sensor management for information fusion - issues and approaches. *Information Fusion*, 3:163–186, 2002.

[Zil96] Shlomo Zilberstein. Using anytime algorithms in intelligent systems. *AI Magazine*, 17(3):73–83, 1996.

[Zim91] H.-J. Zimmermann, editor. *Fuzzy Set Theory - And Its Applications.* Kluwer Academic Publishers, 1991.

[ZR96] Gilad Zlotkin and Jeffrey S. Rosenschein. Mechanism design for automated negotiation, and its application to task oriented domains. *AI Journal*, 86(2):195–244, October 1996.