Nada är en gemensam institution mellan
Kungliga Tekniska högskolan och Stockholms universitet.

Large-Scale Information Acquisition for Data and Information Fusion

RONNIE JOHANSSON

Avhandling som med tillstånd av Kungliga Tekniska högskolan
framlägges till offentlig granskning för avläggande av teknologie doktorsexamen
fredagen den 7 april 2006 kl 14.00
i sal F3, Lindstedtsvägen 26,
Kungliga Tekniska högskolan, Stockholm.

TRITA-CSC-A 2006:2
ISSN-1653-5723
ISRN-KTH/CSC/A--06/2--SE
ISBN 91-7178-300-8
© Ronnie Johansson, mars 2006

TRITA-CSC-A 2006:2
ISSN-1653-5723
ISRN-KTH/CSC/A--06/2--SE
ISBN 91-7178-300-8

KTH
School of Computer Science and Communication
SE-100 44 Stockholm
SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen i datalogi fredagen den 7 april 2006 klockan 14.00 i konferenssal F3, Kungl Tekniska högskolan, Lindstedtsvägen 26, Stockholm.

© Ronnie Johansson, mars 2006

Tryck: Universitetsservice US AB

Abstract

The purpose of information acquisition for data and information fusion is to provide relevant and timely information. The acquired information is integrated (or fused) to estimate the state of some environment. The success of information acquisition can be measured in the quality of the environment state estimates generated by the data and information fusion process.

In this thesis, we introduce and set out to characterise the concept of large-scale information acquisition. Our interest in this subject is justified both by the identified lack of research on a holistic view on data and information fusion, and the proliferation of networked sensors which promises to enable handy access to a multitude of information sources. We identify a number of properties that could be considered in the context of large-scale information acquisition. The sensors used could be large in number, heterogeneous, complex, and distributed. Also, algorithms for large-scale information acquisition, may have to deal with decentralised control and multiple and varying objectives.

In the literature, a process that realises information acquisition is frequently denoted sensor management. We, however, introduce the term perception management instead, which encourages an agent perspective on information acquisition. Apart from explictly inviting the wealth of agent theory research into the data and information fusion research, it also highlights that the resource usage of perception management is constrained by the overall control of a system that uses data and information fusion.

To address the challenges posed by the concept of large-scale information acquisition, we present a framework which highlights some of its pertinent aspects. We have implemented some important parts of the framework. What becomes evident in our study is the innate complexity of information acquisition for data and information fusion, which suggests approximative solutions.

We, furthermore, study one of the possibly most important properties of large-scale information acquisition, decentralised control, in more detail. We propose a recurrent negotiation protocol for (decentralised) multi-agent coordination. Our approach to the negotiations is from an axiomatic bargaining theory perspective; an economics discipline. We identify shortcomings of the most commonly applied bargaining solution and demonstrate in simulations a problem instance where it is inferior to an alternative solution. However, we can not conclude that one of the solutions dominates the other in general. They are both preferable in different situations. We have also implemented the recurrent negotiation protocol on a group of mobile robots.

We note some subtle difficulties with transferring bargaining solutions from economics to our computational problem. For instance, the characterising axioms of solutions in bargaining theory are useful to qualitatively compare different solutions, but care has to be taken when translating the solution to algorithms in computer science as some properties might be undesirable, unimportant or risk being lost in the translation.

Keywords: sensor management, perception management, data fusion, information fusion, large-scale information acquisition, multi-agent coordination protocol, axiomatic bargaining theory, particle filter tracking, pyro-electric infrared sensor, multi-robot system

Sammanfattning

Syftet med informationsinhämtning för data- och informationsfusion är att erhålla relevant och aktuell information. Den erhållna informationen fusioneras för att slutsatser om en omgivning skall kunna dras. Informationsinhämtningens prestanda kan mätas i kvaliteten av de skattningar av omgivningens tillstånd som data- och informationsprocessen åstadkommer.

I den här avhandlingen inför vi begreppet storskalig sensorstyrning och försöker finna dess utmärkande drag. Vårt intresse för det här ämnet berättigas både av den uppmärksammade bristen på forskning om en helhetssyn på data- och informationsfusion och den ökande tillgången av nätverkande apparater vilka erbjuder tillgång till en mängd informationskällor.

I den befintliga forskningslitteraturen kallas en process som utför informationsinhämtning ofta för sensorstyrning (eng. *sensor management*). Vi inför dock termen perceptionsstyrning istället vilken erbjuder ett agentperspektiv på informationsinhämtningen. Förutom att införliva den rika agentforskningen i data- och informationsfusionsforskningen understryker den också att perceptionshanteringens resursutnyttjande begränsas av den överordnade systemstyrning som nyttjar data- och informationsfusion.

För att gripa an de utmaningar som sammankopplas med informationsinhämtning presenterar vi ett ramverk som lyfter fram några av dess viktiga sidor. Vi har implementerat några intressanta delar av ramverket. Den oundvikliga komplexiteten i informationsinhämtning för data- och informationsfusion blir tydlig i vår studie. Därför är approximativa lösningar av intresse.

Vi studerar en av de eventuellt viktigaste egenskaperna hos storskalig sensorstyrning, decentraliserad styrning, mer detaljerat. Vi presenterar ett protokoll för multiagent koordination baserat på återkommande förhandlingar. Förhandlingarna är baserade på axiomatisk förhandlingsteori (eng. *axiomatic bargaining theory*) vilket är ett ämne inom ekonomin. I vårt arbete upptäcker vi fall där den vanligaste förhandlingslösningen har brister och visar i simuleringar ett fall där en alternativ lösning fungerar bättre. Vi kan dock inte dra slutsatsen att den alternativa lösningen är bättre i allmänhet. Båda lösningarna kan föredras i olika situationer. Vi har även implementerat förhandlingsprotokollet på ett par mobila robotar.

I samband med arbetet med förhandlingsprotokollet upptäckte vi en del subtila problem med att överföra förhandlingsteoretiska lösningar från ekonomin till datalogiska problem. Det är exempelvis nyttigt att använda de axiom som karaktäriserar lösningar i förhandlingsteorin för att jämföra olika lösningar, men man måste vara försiktig när man överför lösningar till datalogiska problem eftersom en del egenskaper inte är önskvärda, oviktiga eller riskerar att gå förlorade i överföringen.

Sökord: sensor management, perception management, data fusion, information fusion, large-scale information acquisition, multi-agent coordination protocol, axiomatic bargaining theory, particle filter tracking, pyro-electric infrared sensor, multi-robot system

Acknowledgements

There are of course a number of people who have supported my work on this thesis. Necessary prerequisites for a research project such as this is funding and guidance. In my case, the research was generously financed by the Division of Command and Control of the Swedish Defence Research Agency (FOI) and the guidance was provided by the Royal Institute of Technology (KTH).

My research was naturally greatly influenced by my supervisor Henrik Christensen. Henrik has been a great support through the years by offering his boundless knowledge and insights and generously sharing his books and enthusiasm. He also explicitly assumed the role of the *devil's advocate* which, apart from being very frustrating at times, encouraged me to make necessary improvements to my work.

My secondary supervisors were Stefan Arnborg (School of Computer Science and Communication/KTH) and Per Svensson (FOI). My supervisors collectively constituted a fascinating blend of experience and a well of knowledge.

During my time in KTH and FOI, I have had the privilege to interact with a great number of both established and aspiring researchers as well as skilled and helpful technical and administrative staff.

My early co-worker and co-author was Ning Xiong who offered a mature scientific perspective on the issues we came across. Our discussions could go on for hours, which colleagues in the lab can testify to. The outcome was two articles of which one is fundamental for this thesis.

I later got to know Robert Suzić who accepted the dual PhD project of mine. I am greatly indebted to Robert in many ways. For a long time he was my closest co-worker and provided many alternative views and ideas. Our cooperation resulted in a series of interdisciplinary articles with the attempt marry our two neighbouring fields of research. His moral support was always appreciated.

I was, furthermore, fortunate to early get involved with the Nada Decision Support Group. The members of the group all brought different views on decision support and we had several interesting discussions and seminars. Klas Wallenius and Joel Brynielsson were two members who provided plenty of comments and suggestions on my work.

Many good friends have passed through the computational vision and active perception laboratory (CVAP) over the years. Ola Ramström was one of them. We had good discussions about the essence of game theory. Furthermore, Andreas

Hedström, Mattias Bratt, and Patric Jensfelt provided generous and invaluable support to a wannabe roboticist in need. Other stimulating CVAP members I have interacted with from time to time include Eric Hayman, Josephine Sullivan, Dani Kragic, Jan-Olof Eklundh, Matthieu Bray, Fredrik Furesjö, Mikael Rosbacke, Christian Smith, Elin Topp, Mårten Björkman, Peter Nillius, Alireza Tavakoli, Barbara Caputo, Johan Thureson, Gareth Loy, Hugo Cornelius, Klas Andersson, Carl Lundberg, . . .

Besides KTH, I was also in close contact with my financiers at FOI, which was not only convenient but also highly enjoyable. Per Svensson should be mentioned again for his kind and sturdy support and his thorough proof-reading. Johan Schubert had many good stories tell. Hedvig Sidenbladh taught me most of the things I know about particle filtering. Göran Neider was my good neighbour and the one who introduced me to CBR; Pontus Hörling, the dedicated project manager; and Karsten Jöred, the friendly mathematician and astronomer. I wish everyone could have an enthusiastic Maths teacher like him. Last but not least, Christian Mårtenson and Pontus Svenson: Where would I be without you guys?

I would also like to thank my family and friends for putting up with me during my intensive studies. Moreover, as PhD students from all times have testified in a section like this, there is often a spouse or a close loved one suffering during the studies of the student. In my case, that special someone has been my beloved Atsuko. How could you endure, my dear?

Preface

The main topic of this thesis is *large-scale information acquisition for data and information fusion.* We hope that this thesis will help to pave the way for holistic information acquisition and an increased interest in some of the frequently neglected issues mentioned herein.

We here provide an Internet reference for material related to this thesis, some notes on the collaborations that directly contributed to this thesis, and, finally, some reading advice for interested readers.

This thesis has been given the research group report number CVAP 300.

World Wide Web Resources

There are a number of resources related to this thesis available through the Internet. Apart from this thesis in digital form, source code, simulations and videos for experiments presented in this thesis are also available.

`http://www.csc.kth.se/~ronniej/project/thesis.html`

Collaboration and Inspiration

Research, in these days, is fortunately strongly catalysed by an intensive and open information exchange.[1] The work in this thesis is no exception, which the extensive bibliography testifies to. Fields that have stimulated the author's work include robotics and agent theory. Open source software packages also greatly accelerated the progress of the work.

Additionally, many people, both directly and indirectly, have stimulated this research. I would like to mention the two researchers who have co-authored some of the articles that this thesis is based upon. Ning Xiong was one of the designers of the concept of *perception management* (Chapter 2) which spawned from a discussion regarding an article of his (Xiong & Svensson, 2002).

I should also acknowledge the great collaboration with Robert Suzić. His work on situation assessment was crucial for the evolution and testing of the large-scale

[1] In contrast, compare this to the secretive and secluded scientific atmosphere that surrounded, e.g., Leonardo Da Vinci and his contemporaries.

information acquisition framework. I owe the implemented plan recognition process (Section 4.4), the design of the application scenario (Section 4.3) and the state transition model for the particle filter (Section 4.5) to him.

Reading Advice

This thesis is a composition of more or less independent chapters originating from previous publications. Although references between chapters have been added to make the thesis more interconnected and unified, most chapters are not a immediate continuation of the preceding chapter. Several chapters, therefore, include their own introduction and literature review.

To assist the reader in grasping the material contained in this thesis, we provide a handy glossary (Appendix E) which lists and explains some of the most used terms. At the end of the thesis is also a subject index to allow the reader to easily locate different parts of the thesis. A brief overview of the thesis chapters is also available in the end of Chapter 1.

As the contents of this thesis span multiple research fields, we expect it to appeal to readers with varying primary interests. To cater for their needs, we provide reading advice for readers interested in *data and information fusion*, *agent theory and decision-making*, *robotics*, and *concrete sensors and sensor networks.*

General Reading Advice

Chapter 2 introduces several of concepts that are used later in the thesis, including *perception management*, *service*, *service configuration space*, etc. Chapter 3 presents the concept of large-scale information acquisition, and some related ideas, e.g., *facilitation*, and *environment representation.*

Data and Information Fusion

The primary audience of this thesis is the research community in the data and information fusion field. Hence, a reader from that community should recognise a lot of the issues discussed throughout the thesis, especially if the reader is also interested in sensor management.

Some ideas presented should be new to the reader. In Chapter 2, e.g., we present the data fusion process in an agent-context to emphasise its relation to an overall process that uses it for decision support. Otherwise, most efforts concern information acquisition. Perception management, is one concept which we introduce to give a broader perspective on information acquisition than we believe sensor management has to offer. In Chapter 3, we survey previous related efforts to extract properties related to perception management. In Chapter 4, we take a comprehensive approach to data and information fusion when we consider information acquisition as a support function for plan recognition. Finally, in the remaining chapters, we study one of the useful skills of large-scale information acquisition, decentralised control, in more detail.

Agent Theory and Decision-Making

This thesis interacts with agent theory in various ways all the way through. Above all, agent theory is used as a source of knowledge to fertilise the research presented. There might, however, also be some issues presented herein that are interesting to the agent theory community.

In Chapter 3, we present a survey and discussion regarding information acquisition. We specifically offer a discussion about *environment representation* and how it is affected by different kinds of perception activities.

The work in Chapter 5 is strongly related to multi-agent systems and discusses multi-agent coordination from a bargaining theory perspective.

Robotics

The robotics aspect of this thesis is most obvious in Chapter 6, where two Evolution Robotics ER1 robotic platforms are used for a coordination experiment. The implemented system combines physical robots with coordination ideas from distributed artificial intelligence.

Concrete Sensors and Sensor Networks

Although, sensor network nodes are used in Chapter 6 and the set of mobile robots presented interestingly constitutes a mobile sensor network, typical sensor network research issues such as packet routing, light-weight software architectures, and battery preservation are not considered. However, apart from the actual sensor nodes themselves, the reader might be interested in the coordination protocol discussed in Chapter 5.

Contents

Chapter 1

Introduction

The increasing availability of low-cost and low-energy sensors (e.g., CCD cameras, microphones, etc) and communication technology (e.g., RFID[1] tags, mobile phones, PDAs, radio chips, etc.) implies the establishment of *intelligent environments*[2] and *pervasive computing*.[3] Devices with these properties may jointly form competent sensing networks to facilitate accurate and precise estimates of some *application relevant* environment. Issues regarding how to practically exploit multiple shared sensing resources was recently addressed by Challa et al. (2005).

Current and future applications based on such devices include situation monitoring in peace-keeping operations, intrusion detection in buildings, and home surveillance for the disabled. The applications all have in common that their performance is dependent on some observed environment state and that the performance will increase if they can acquire a better understanding of the same. A better understanding can manifest itself in terms of, e.g., more accurate or less uncertain state estimates and will result in more beneficial decisions (i.e., actions and analyses).

Data and information fusion is an interdisciplinary research field whose focus is on the integration (i.e., fusion) of information from multiple sources. The field is intended to encompass both fusion of measurements from sensors as well as further processing of data (e.g., through aggregation) and comprises of theory, techniques and tools for the exploitation of (and support for) information from different sources. An introduction to the many facets of data and information fusion is provided by Hall and Llinas (2001).

In this thesis, we sometimes make a distinction between data fusion and information fusion. When we do so, we consider data fusion (or sometimes "sensor data fusion") to involve the integration of immediate sensor data, and information

[1] Radio frequency identification

[2] For instance buildings with integrated networks of nodes with computational, sensing, and communication abilities.

[3] Pervasive (or ubiquitous) computing denotes the vision of a world where all kinds of items (including goods, furniture and even clothes) are potential hosts of microcomputers.

fusion to concern further processing of data.[4]

For historical reasons, the frequently referenced JDL model, which describes the functions of data and information fusion, is simply called a “data fusion model” (Steinberg & Bowman, 2001). It, however, encompasses those functions which we normally consider to be expressed by both data and information fusion. In Chapter 2, where we describe the JDL model in more detail, the concepts of “data fusion process” and “data fusion system” refers to both data and information fusion.

The principal motivation for the data and information fusion field is often represented by the following quote by Naisbitt (1982):

> *We are drowning in information but starving for knowledge. This level of information is clearly impossible to be handled by present means. Uncontrolled and unorganised information is no longer a resource in an information society, instead it becomes the enemy.*

A part of the data and information fusion field concerns *information acquisition*, i.e., the process of acquiring new information to maintain an acceptable level of understanding of the application environment or to improve the understanding. This process is frequently entitled *sensor management*. Its underpinning rationale is that unless the sensing resources of an application system constantly provide it with sufficient information to successfully perform its task (or if the performance could be improved), information acquisition is an important activity of the system (Blackman & Popoli, 1999, pp. 978). Furthermore, automatic information acquisition facilitates timely response to fast changing environments and objectives and offers the opportunity to handle complexities in the control of sensing resources (e.g., the availability and status of resources, and the sharing of resources between multiple system users).

1.1 Holistic Information Acquisition

The principal research problem of this thesis is indicated by a quote by Llinas (2003, pp. 2038):

> *[...] while all of the research going on that is addressing each part of the distributed target tracking problem is of course valuable [...] there appears to be very little research on the holistic design of such [...] networks and the underlying fusion approaches required.*

A holistic design, Llinas argues, would be competent enough to deal with decentralised solutions and especially issues such as efficient information-sharing and reliable

[4] Admittedly, other interpretations of data fusion and information fusion exist. Both data fusion and information fusion are by some authors considered to include the other. These different views are not fundamentally opposing. It merely reflects an inconsistency in the terminology that is in use, and the varying preferences between authors. To avoid confusion and objections (hopefully), we frequently use both terms at the same time in this thesis, i.e., data and information fusion, without assuming that one is contained within the other.

network communication infrastructures. Although Llinas refers to the target tracking part of data and information fusion, a similar concern can be raised towards the information acquisition counterpart. Namely, considerable attention (within the data and information fusion field) has been given to variations of isolated and similar sensor management problems, but much less effort has been devoted to its integration with the overall information fusion system and to achieve a holistic grasp on data and information fusion.

A similar challenge is promoted by Mahler (2004, pp. 535):

> *[Sensor management] for [situation and impact assessment] is an even more daunting challenge than for multi-sensor integration.*

Situation and impact assessment are explained in Chapter 2, but are considered to be functions belonging to information fusion. Moreover, situation and impact assessment are considered to generate "high-level" information that has been inferred or further processed from fused data. Hence, this statement highlights a shortage of research for information acquisition for information fusion.

The state-of-the-art of sensor management in data and information fusion is summarised by, e.g., Xiong and Svensson (2002) and Ng and Ng (2000). However, information acquisition in information fusion is basically an optimisation problem (i.e., selecting sensing actions to achieve the best expected result given the current understanding of the environment, sensing resources, and mission objectives) and, as such, has been studied in various fields of research. Other fields of research will certainly contribute to a holistic information fusion. Decentralised control is one topic which is relevant for information acquisition in information fusion since centralised control becomes unsuitable with an increasing number of resources. Multi-agent theory (e.g., Weiss, 1999), e.g., inherently addresses decentralised solutions, unlike most efforts in information fusion to date which suggest centralised solutions.

In summary, requests have been made for contributions to a holistic view on the field of data and information fusion. The focus of the thesis is then to describe an approach to realise the information acquisition of such a holistic data and information fusion.

1.2 Large-Scale Information Acquisition

In this thesis, we define information acquisition in an information system to be

> *the skill of a system that allows it to reason about and facilitate (active) acquisition of information about a system-relevant environment.*

We define "system-relevant environment" to mean the part of the environment which a system (by design) should be able to model and reason about to perform its tasks successfully. The opposite, i.e., a non "system-relevant environment", is the part of the system's environment which either will have no effect on the system's decision-making or cannot be considered in the system's deliberation (for computational or other reasons). To address the vision of intelligent environments with a large amount of sensors, explained in the preceding discussion, we add the qualifier "large-scale" to information acquisition. Our desire is that the large-scale information acquisition (LSIA) concept will stimulate research in some interesting issues that are relevant for holistic information fusion. Those issues include:

- decentralised control (it is in general not feasible to manage large amounts of sensors using a centralised approach);
- multiple and varying objectives (LSIA may simultaneously serve several, possibly conflicting, information needs).

LSIA should also assume a comprehensive view on sensing resources. The resources may be

- numerous (requires strategies for *which* sensors to use and *when*);
- complex (e.g., multi-modal, mobile, etc);
- distributed (provides a greater sensing scope, but often enforces sub-optimal control);
- heterogeneous (sensors have different qualities).

The contributions of the thesis can be summarised in the following points.

- Our approach to LSIA is from an agent perspective. By integrating data and information fusion, and information acquisition specifically, in a general agent model, the dependencies between fusion techniques, mission goals, and information acquisition become apparent.[5] We, furthermore, introduce the concept of *service* with the intention to express the action potential of a system of sensing resources in terms of a *service configuration space.*
- We also provide an extensive literature review that results in a characterisation of two subtle aspects of information acquisition: *facilitation* (i.e., managing sensing constraints and changing the space of possible observations) and *focus of attention* (i.e., deciding what to focus on). The literature study also elicited the LSIA properties listed above.
- The literature review also forms the basis for a conceptual framework for LSIA. We use the framework to extend a plan recognition application with information acquisition. Plan recognition yields inferred state estimates and our work can therefore be seen as a response to the aforementioned appeal by Mahler (2004).

[5] Note that we use the agent concept for two different purposes throughout the thesis. An agent is defined as some process that can sense and act in an environment. This description fits a whole system (as in Section 2.1) as well as system components (as in Chapter 5).

We implement pertinent parts of the framework such as long-term sensing actions (e.g., when sending an UAV to a remote location to make an observation), multiple objectives, and heterogeneous sensors. An essential component of our implementation is the particle filter tracker, which is used in several ways. The implementation also reveals some insight into the complexity of connecting high-level information need (i.e., in our case, the need for information acquisition based on plan recognition results).

- A final contribution is a detailed discussion about decentralised control from a bargaining theory perspective. To reiterate, we consider decentralised control to be one of the important capabilities of LSIA. We discuss a coordination protocol for the type of *benevolent* agents we expect to have in an information fusion application. We show in simulations that different solutions from bargaining theory can give noticeable different results. The coordination protocol is then brought into a physical context when we implement and test it on a set of mobile robots. A conclusion of this study is that the conventional game theoretic way to resolve agent conflicts is not always preferred.

1.3 Thesis Outline

Here follows a summary of the remaining chapters of the thesis.

Part I: The Domain of Information Acquisition

Chapter 2 This chapter contains an interpretation of the JDL description of the data fusion process from an agent perspective. We argue that the agent perspective situates the data fusion process in a larger context and invites interesting results from agent theory to be considered in the data and information fusion context. Various concepts that are used in subsequent chapters are introduced. The agent-based model encourages us to introduce the concept of *perception management* (which realises information acquisition in a system and extends the sensor management concept). We present *services* as a concept to represent the space of sensing actions for information acquisition.

The discussion of perception management is an extension of a joint publication with Ning Xiong (Johansson & Xiong, 2003). The concept of *perception services* was published in a subsequent article (Johansson & Suzić, 2004).

Part II: Properties of Large-Scale Information Acquisition

Chapter 3 We provide a taxonomy of methods reported in the literature related to information acquisition and give some examples. The identified characteristics of previous efforts in the field along with an envisioned future with efficient, competent and abundant sensors amount to the introduction of the main topic of this thesis: large-scale information acquisition.

Chapter 4 Based on the discussion in Chapter 3, we suggest a comprehensive and general framework for the connection between the information need of a data fusion process and information acquisition. Specifically, we address the issue of realising application-relevant information acquisition for a plan recognition process. The purpose of the plan recognition process is to estimate the activities of some agents that can be observed in the environment. Uncertainty in agent state is represented by particle filters from which agent plans and threats are estimated. Information acquisition, here, manifests itself as task allocation. The details of an implementation based on the framework are given and some results are shown. The chapter also contains a discussion about inherent difficulties related to optimal information acquisition for information fusion.

The chapter is a summary and extension of joint efforts together with Robert Suzić (Johansson & Suzić, 2004; Suzić & Johansson, 2004; Johansson & Suzić, 2005).

Part III: Issues in Large-Scale Information Acquisition

Chapter 5 One of the important properties of LSIA is that of decentralised control. This chapter proposes a decentralised recurrent coordination protocol based on a bargaining mechanism. There are many mechanisms offered for bargaining theory. We study the most common, the Nash bargaining solution, from a computational (rather than an economics) perspective and show that an alternative (the egalitarian solution) is sometimes preferable.

The contents of the chapter have been submitted in the shape of an article to the Journal of artificial intelligence research (Johansson, 2005).

Chapter 6 We use a couple of mobile robots to implement the coordination protocol presented in Chapter 5. The two robots use radio communication to exchange information about their individual states and preferences over negotiation deals. Both robots are equipped with sensors to detect moving infrared emitting sources. In two experiments, the robots negotiate about how to jointly explore a common office environment. In the first experiment they explore the environment without detecting anything. The resulting behaviour is compared to the second experiment where a target is suddenly detected by one of the robots.

Part IV: Summary

Chapter 7 Summarises the most important parts of the thesis, provides some further discussions and highlights issues for future research.

Part V: Appendices

Appendix A This appendix holds some properties for sensing resources that one might want to consider when designing a perception management process.

Appendix B Properties of sets of sensing resources are discussed in this appendix. Sets might, e.g., be heterogeneous and contain resources which yield different kinds of data.

Appendix C In Chapter 4, we briefly compare our framework to other related architectures. This appendix contains details on the comparison that were not included in Chapter 4.

Appendix D We here define the concept of *Pareto frontier* which is used in Chapter 5. We also present an algorithm for how to find it.

Appendix E Contains a list of terms used in this thesis and their explanations.

Part I

The Domain of Information Acquisition

Chapter 2

An Agent-Based Model of the Data Fusion Process and Perception Management

The ability of biological beings, such as mammals, to efficiently combine data from measurements (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 data and information fusion research. The overall aim of the research on data and information fusion is to bring this ability into use in autonomous or semi-autonomous artificial systems for enhanced performance.

Note that we in this chapter, in contrast to the previous, frequently will refer to data and information fusion as simply data fusion. The reason is that the JDL model, which the contents of this chapter revolves around, uses that terminology.

A *data fusion process* (DFP) 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 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 (Hall, 1992, Ch. 1) for instance, military (e.g., avionics, Musick & Malhotra, 1994; Adrian, 1993, and command and control, Gonsalves & Rinkus, 1998), remote sensing (e.g., localisation of mineral sources, Mavrantza et al., 2002 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 (Lowe, 1998).

Typically, systems which have to rely on continual, real-time observations of a dynamic and partially unknown environment, will benefit from data fusion. In the most difficult case, the dynamics of the environment is partially caused by a malevolent agent[1] that in turn observes the other system.

Currently, much research in the field of data fusion have been done, but due to its immaturity and the different angles of approach of previous research (e.g., command and control, avionics, mobile robots, and machine perception), there is a need for a unified comprehensive framework, and a generic taxonomy and terminology has yet to be established. A step in that direction is the study by Appriou et al. (2001) 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 is frequently entitled *sensor management.* However, in this thesis, a wider term, *perception management* (which we previously introduced, Johansson & Xiong, 2003), is used instead to emphasise the general standpoint of the thesis. Using a "fresh" term also allows us to reason more freely about the properties of information acquisition 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 (Ng & Ng, 2000).

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 communications aid for reasoning about information acquisition and in the development of applications.

Section 2.1 portrays the data fusion process using the JDL model (known from the field of data and information fusion). The control aspect of the JDL model is process refinement, which is briefly presented in Section 2.2. Perception man-

[1] In this chapter, *agent* is simply defined as something that perceives and acts (Russell & Norvig, 1995). Hence, it could be either a human or an automated process.

agement, being a subset of the process refinement function, is further discussed in Section 2.3. The input to perception management in terms of stimuli from its environment is identified and properties of its sensing resources are also discussed. In Section 2.4, we introduce a sensor control space of service configurations. Section 2.5 offers a brief summary of the chapter contents. Furthermore, notes on sensing resource properties relevant for perception management are gathered in Appendix A. Management of sets of resources introduces additional control issues that might have to be addressed. Issues of that kind are discussed in Appendix B.

2.1 The Data Fusion Process

A definition of data fusion is offered by Starr and Desforges (1998):

> *Data fusion [is] the 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 which, in its latest revision of its data fusion model (Steinberg & Bowman, 2001), settle with the following concise 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 Starr and Desforges.

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 DFP, which is not included in the first definition and implicit in the second, is *process refinement*, the function of improving the DFP. 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, signal processing, and computer science (Waltz & Llinas, 1990). The DFP comprises techniques for data reduction, data association, resource management, and fusion of uncertain, incomplete, and contradictory information.

As mentioned in the beginning of the chapter, 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 of data In some existing 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 to fusion of data performed by manual labour, (automated) data and information fusion may be less costly, more reliable and predictable (human beings are known to make mistakes; especially under pressure), and, of course, faster. Automated data and information fusion may also be customised and optimised for a specific task.

Improving performance in automated systems Data and information 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 (listed by Paradis et al., 1997). 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.

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] We will refer to this system model as the "overall system", "enclosing system", or sometimes just system, throughout the thesis. Note that this model does not suggest 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 (agent models such as this one can be found everywhere in agent literature, e.g., in Wooldridge, 1999; Nilsson, 1998; Russell & Norvig, 1995).

The *system control* is responsible for the system objectives and results of the system operation. We distinguish between *mission objectives* and *management objectives*. Mission objectives concern the overall goals of the system, its purpose for existence. The management 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

[2] Here we use the term "environment" to refer to the process which DFP has to sense and act upon. Other terms used in the literature are "world" and "workspace".

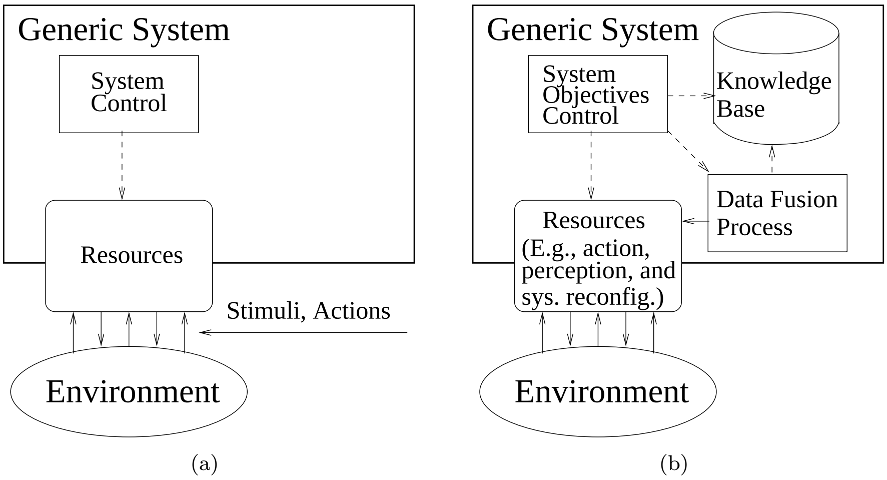


Figure 2.1: (a) A coarse sketch of a generic (agent-like) system which result depends on its interaction with some environment. (b) A generic system including a data fusion process. The system control has been decomposed into a system objectives control, a DFP and possibly a knowledge base.

system (i.e., all controllable resources). It uses its resources mainly to act and perceive. We distinguish between two types of actions: *ordinary actions* (which are intended to change the state of the environment) and *sensing actions* (which affects the perception of the system). We are mainly concerned with the latter type in this thesis. The sensing resources respond to the stimuli of the environment and produce *measurements*. We call measurements that have been compared and related to environment models *observations*. Measurements and observations that are expressed in a format suitable for storing or inference by the system are called *percepts*.

In the simple view of Figure 2.1(a), 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 objectives 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, e.g., the system objectives control and the DFP may both access the set of system resources and access (either through direct access or with

[3] The system objectives control is not equivalent to the system control in Figure 2.1(a) since some control may be performed by the DFP (e.g, information acquisition by its process refinement function), but it is responsible for the objectives and goals of the system.

the help of data mining techniques) and alter the knowledge base.[4] The influence between the system objectives 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.

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

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. For the discussion in this section, we call the agent that contains the DFP, and observes some environment, the *observing agent*. The environment which the observing agent perceives and interprets might, depending on application, be simple and accessible but is typically (environment properties adopted from Russell & Norvig, 1995, pp. 46):

Inaccessible The complete state of the relevant environment can not be determined by the observing agent if it is inaccessible. 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 chessboard, e.g., is completely accessible, whereas a poker game environment is not (at least not to a non-cheating observing agent).

Nondeterministic The outcome, or value, of actions performed in a nondeterministic 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 a observing agent, the environment will *appear* nondeterministic if it is also inaccessible.

Nonepisodic Actions performed by the observing agent affects the future evolution of a nonepisodic environment. Chess and other multi-player games, e.g., are nonepisodic since there exist opponents who will respond to the moves by the agent. In an episodic environment, an agent can act reactively (*reactive* is explained in Section 2.3) and can not gain anything from reasoning about its affect on future events.

[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.

Dynamic The configuration of a dynamic environment will change with time independent of the observing agent. In a *static* environment, the observing agent 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 observing agent if the state does not change over time, but the performance of the observing agent does.

Continuous The features of the environment may be continuous, e.g., positions, speed, and temperature, and also the possible actions. The opposite of continuous is *discrete*. The state of a chessboard, e.g., is discrete, where the game-relevant information is captured in the 64 squares of the chessboard and the discrete set of types of chess pieces.

A further environment property, suitable for domains of conflict, that could be considered is *deceptive*. In a deceptive environment, an adversarial agent might attempt to interfere with the perception of a observing agent. The adversarial agent might, e.g., act to avoid being detected or might purposely provide the observing agent with misleading observations.[6]

Properties of Data and Information

A discussion about the representation of data and information deserves a chapter (or probably even a book) of its own accompanied by an extensive survey. For our study in this thesis, 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, the exact properties of data and information are important for the refinement, fusion and evaluation of information. We would, therefore, like to present a small selection of works that discuss this matter. As we will see, the usage of the terms is not always consistent.

A classification of terms is suggested by Steinberg (1999). 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 described in Section 2.1), and *knowledge* understood and explained information. Some literature stress that knowledge should be stored information (e.g., Miller et al., 2001). Information that has been accepted and that constitute a prior belief for further reasoning and decision-making.

Appriou et al. (2001) 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 relate to the current state of the environment. Knowledge, on the other hand, is

[6] In Chapter 5, we use a more general term: *antagonistic* environment. By that we mean the environment that tries to obstruct the operation of the observing agent, e.g., through deception.

defined as information that describes general properties of the environment, such as characteristics of a class of situations.

Observations and knowledge are called *descriptive knowledge* by Appriou et al. (2001). 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 are rarely capable of accurately conveying (measuring and reporting) the environment features they are observing. To capture this discrepancy between the real world properties and measurements, some sort of representation of information imperfection is necessary.[7] Appriou et al. (2001) suggest a taxonomy of such imperfect or 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

Arnborg et al. (2000) define *information awareness* to denote 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 well-founded decisions, three properties are attached to information: *precision*, *quality* and *utility*. Precision regards, here, 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.

In our thesis, data refers to the contents of a measurement and information to the contents of an observation. Understanding and remembering this distinction, however, is not crucial for comprehending the thesis.

A Functional Data Fusion Model

To break down and further analyse the data fusion process, we use a functional model[8] for data and information fusion; the so called JDL model, named after the Joint Directors of Laboratories, mentioned above and first presented in the mid 1980s. Although originally developed for military applications, the model presented here is generally applicable. Furthermore, the model does not assume its functions

[7] It should be noted that depending on application and decision situation, the necessary degree of information perfection varies.

[8] A functional model consists of definitions of functions which could comprise any DFP (Steinberg & Bowman, 2001). Unlike a process model, it does not specify interactions between functions, only functional aspects of a DFP.

Table 2.1: Aspects of defective information by Appriou et al. (2001).

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 with certainty that 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 100 m" 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.

to be automated, they could equally well be maintained by human labour. Hence, the model is both general and flexible.

The revised JDL model (Steinberg & Bowman, 2001; White, 1998) 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, Waltz & Llinas, 1990).

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 in our interpretation, the governing system objectives control and the available resources, is also depicted.

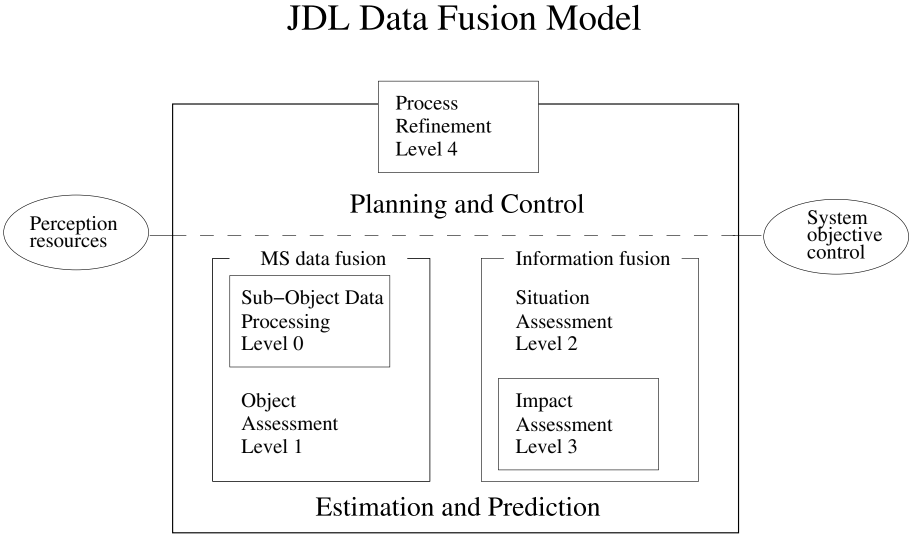

JDL Data Fusion Model
Process Refinement Level 4
Planning and Control
Perception resources
System objective control
MS data fusion
Sub−Object Data Processing Level 0
Object Assessment Level 1
Information fusion
Situation Assessment Level 2
Impact Assessment Level 3
Estimation and Prediction

Figure 2.2: The JDL data fusion 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 *multi-sensor data fusion* or *multi-sensor integration*, data is combined to assign dynamic features (e.g., velocity) as well as static (e.g., identity) to objects,[9] hence adding semantic labels to data. This level includes techniques

[9] It could be debated whether or not “object” is the most appropriate term to use. In some ap-

for data association and management of objects (including creation and deletion of hypothesised objects, and state updates of the same). The *anchoring problem* in robotics (Coradeschi & Saffiotti, 2003), i.e., the problem of creating and maintaining the relationships between internal symbols and sensor data, 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 Level 1 entities. Input Level 1 data are assessed with respect to the environment, relationship among Level 1 entities, and entity patterns in space and time (Paradis et al., 1997; Hall, 1992).

Level 3 - Impact Assessment: The impact assessment, which is sometimes called *significance estimation* or *threat assessment*, 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. A decomposition of threat assessment has been presented by Steinberg (2005).

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,[10] optimisation of resource usage with respect to, for instance, electrical power consumption. Section 2.2 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.[11] 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.

plications, it might not be clear what an object is. More appropriate terms might be "component", "constituent", or "element".

[10] 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.

[11] The reason for these names are that it manages the other processes (Paradis et al., 1997).

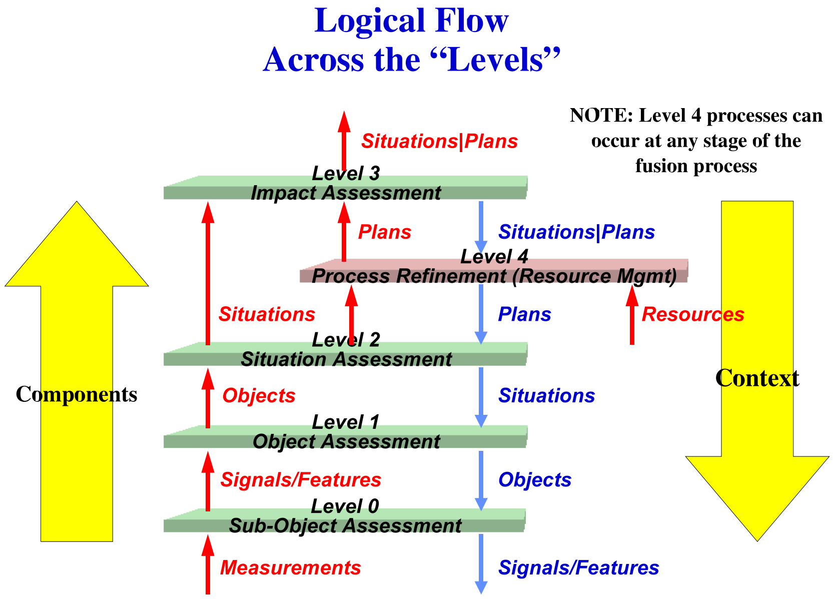


Figure 2.3: Typical logical flow between the JDL model functions (image borrowed from Steinberg & Bowman, 2001). By courtesy of Alan Steinberg.

Remarks on the JDL Model

A Level 5, *user refinement*, has also been proposed by Blasch and Hanselman (2000). 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, hence, a bit restrictive. For instance, 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, Steinberg and Bowman (2001) identify 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, Level 4 may have a direct effect on the system's external behaviour (through the use of perception resources, explained in Section 2.3). 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 in the figure 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 a 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 (White, 1998).

Whereas Level 0 and Level 1 concern multi-sensor 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 by Steinberg and Bowman (2001).

Recently, the JDL model as presented here has been questioned in articles and debates, and a major revision is anticipated. A step in this direction was offered by Llinas et al. (2004) who suggest extensions of the JDL model in the areas of, e.g., quality control, ontologies for a shared understanding of information, and distributed processing. We will return to the fundamental issue of distribution, although from an acquisition perspective, frequently throughout the thesis.

Practical Issues

Moving from the functional model in Section 2.1 to a working implementation in a real environment involves a number of design considerations including what information sources to use (e.g., single or multiple sensors), what fusion architecture to employ (centralised/decentralised), communication protocols, etc. In this section, we discuss the properties of single and multi-sensor 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. For instance, a fusion algorithm may be applied to a sequence of images produced by a single camera sensor. However, employing multiple sensors provides many advantages (as advocated by, e.g., Manyika & Durrant-Whyte, 1994; Luo & Kay, 1992; Waltz & Llinas, 1990; Llinas, 1988):

Redundant information When multiple sensors perceive the same feature of the environment, the redundant information can be exploited to reduce uncertainty and imprecision 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. With complementary information ambiguities could (possibly) solved and more complete information established.

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 (simultaneously) over a possibly large spatial region.

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

Note that it is sometimes useful to consider multi-sensor systems as complex sensors (or logical sensors as denoted by Budenske & Gini, 1997; Grupen & Henderson, 1989; Henderson & Shilcrat, 1984 or virtual sensors by Muir, 1990). For instance, if the motivation for using multiple sensors, in some situation, is to decrease the time interval between observations, we have constructed a complex (or logical) sensor which is more timely than its simple sensor components. Since the complex 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 2.3.

Unsurprisingly, there are also challenges associated with the use of multiple sensors, as further noted by Luo and Kay (1992, pp. 16):

Sensor registration and data alignment: Failure to make correct associations between signals or features of different measurements from different sensors. Sensor registration concerns the estimation of properties of a sensor (e.g., the global position and orientation of the sensor) that affect the information content of its measurements. Given the estimated properties (and quality of the registration) for multiple sensors, measurements can be aligned to a common reference frame (e.g., coordinate system).

Common restricting assumptions: Assumptions, more or less realistic, often have to be made to enable the use of some fusion techniques. Common assumptions include measurement independence and Gaussian distributed noise. One must be aware that the common assumptions might not always hold.

Coordination and conflicts: Multiple sensors might have to be coordinated and information may have to be shared between them. Occasionally, multiple sensors will (possibly due to sensor failures) make conflicting observations, which has to be dealt with.

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.

2.2 Process Refinement

The process refinement (i.e., the meta-controller), the fourth level of the JDL model (described in Section 2.1), monitors the other parts of the data fusion 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 and information fusion in complex and environments, process refinement is a magnificent support.

Process refinement is decomposed into three functional parts by Paradis et al. (1998), and presented slightly modified here:

Input refinement: Controlling system resources to improve the DFP (i.e., produce more useful information). 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 2.3), 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 multi-sensor 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); information filtering and tuning of filters.

Inter-level information flow: Controlling the information exchange between levels.[12] 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.

[12] Note that we here deviate from the work by Paradis et al. (1998) who called this part "Additional or complementary processing." The difference is that we expand the original idea of hypothesis reinforcement from higher to lower levels to include all interaction between DFP levels.

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.

Note that the first part involves system external actions to get new information while the two following parts concern internal actions.

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 objectives control (in Figure 2.2),[13]
- intended actions of the system objectives control,[14]
- 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 information acquisition, in more detail.

2.3 Perception Management

The need for information acquisition is dependent on the available resources and sensing tasks and must be motivated. Blackman and Popoli (1999, pp. 978) list two necessary preconditions: (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.[15] 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 also be a competition among the sensing tasks for the available sensing resources. If both of these two preconditions are met, and for highly competent systems working in difficult environments they tend to be, a perception management function for improved information acquisition should

[13] Two different kinds of requests are identified by Denton et al. (1994): *manual* and *automatic*. A manual request in a command and control application could, e.g., be an intelligence inquiry by a system operator. An example of an automatic request is the (nearly 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.

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

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

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*. We elaborate further on the extension of precondition (ii) in Chapter 3.

Many names have been used for the realisation of information acquisition. In the context of the general data fusion model described in this work and depicted 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 (Kastella & Musick, 1996).

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.

Information (re-)source management Is also too general. An information source[16] may for instance be a database, and the purpose of the information acquisition in perception management is not to manage databases (that would possibly be the job of some other component of the process refinement; see Section 2.2), 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).

Information gathering A term chiefly used in agent theory. It mostly concerns the data gathering of software agents from databases and on the Internet.

Here, the term *perception management* is preferred, since, in accordance with the "input refinement" concept in Section 2.2, it is the perception of the environment (and ultimately the comprehension of the environment), which should be improved, not directly the physical sensors as suggested by the term sensor management.

The perception management term inherits its first part from its affinity with the biological concept of perception used frequently in related fields of research. It is, e.g., often used to describe the process of sensing in robotics, and the term "active perception" (Bajcsy, 1988) is well known in computer vision. The second

[16] 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.

part, “management”, is a flexible term that also reflects its close relation to sensor and resource management. These motivations are further discussed by Johansson and Xiong (2003).

Even though a new concept is used, the ideas of perception management are inspired by the different kinds of management mentioned above. A rough description of the relationships between resource, perception, and sensor management,[17] depicted in Figure 2.4, clarify this fact.

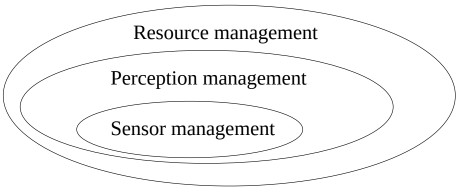


Figure 2.4: A simplified and coarse sketch of the relationships between some types of management. Resource management is considered to encompass perception management, which in turn encompasses sensor management.

In the aforementioned article (Johansson & Xiong, 2003), 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 information acquisition in an agent theoretic context (which facilitates fluent communication between various fields of research) and naturally encompasses processes that support information acquisition. In previous research, we found support for our view where authors tentatively suggest other concepts to grasp activities related to information acquisition. Some of those concepts appear inappropriate to attribute to sensor management, but suitable for perception management. One example is goal management (Hintz & McIntyre, 1999), which we, in Section 3.10, capture with the perception management aspect of *focus of attention.* Another is that of changing the observation space (e.g., by relocating sensors) and enabling sensors (e.g., by optimising the usage of batteries and communication links), which we express with the *facilitation* concept (Section 3.9).

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 perception management will probably act as initiator and controller of various existing sensor management processes.

Admittedly, the term perception management has already been used for a different purpose. Gonsalves et al. (2000) use it to describe deception (i.e., the

[17] In some previous works, sensor management and mission management, i.e., the mechanism which decides which perception tasks to perform, are separated (Denton et al., 1994; Musick & Malhotra, 1994). Here, they are both considered to be a part of perception management.

management of the perception of *others*). To distinguish between the two interpretations, we could add the qualifiers *introversive* and *extroversive* to perception management.

The discussion about perception management in the rest of this chapter journies beyond our previous publication (Johansson & Xiong, 2003). It elaborates on the duties of perception management, the stimuli that affect it, properties of the resources it could manage, and defines the concept of *perception service* (resource properties are discussed in Appendix A). Properties belonging to the management of sets (or systems) of resources are also discussed in Appendix B . Admittedly, most of this work should also be applicable to sensor management, which the reader might feel more comfortable with.

Function and Stimuli

The purpose of this section is to give a flavour for the work a perception manager (i.e., a process or agent devoted to perception management) could be performing, and the description herein is by no means complete. In the following chapter, we will survey previous works in this field and present more details about information acquisition.

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

- optimise perception resource usage for information acquisition 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 objectives control), or when the perception resources are limited and can not serve all information requests;

- prioritise and carry out information acquisition tasks when perception resources are limited and cannot support all tasks simultaneously.

Important issues which 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 one, 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 2.2, 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 objectives 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 (i.e., what information to look for). Stimulus (i) helps a perception manager 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,[18] which have in turn received a task from the perception manager, 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 for information.

Perception Resources and Services

Naturally, the performance of perception management is dependent on the (perception) resources that is available to it. The properties of the resources (e.g., accuracy, reliability, etc) will also affect the quality of the perception management itself. In this section, we discuss the meaning of the *perception resource* concept and properties relevant for its management. We also present the concept of *perception service* in an attempt to uniformly express the sensing potential of the perception resources.

We consider the resources of a generic system to include all resources the system is said to possess, e.g., amount of money or fuel, number of vehicles, competence of

[18] Here, we think of a service as some sort of process (or logical sensor as described by Henderson & Shilcrat, 1984) that uses one or more resources to acquire information.

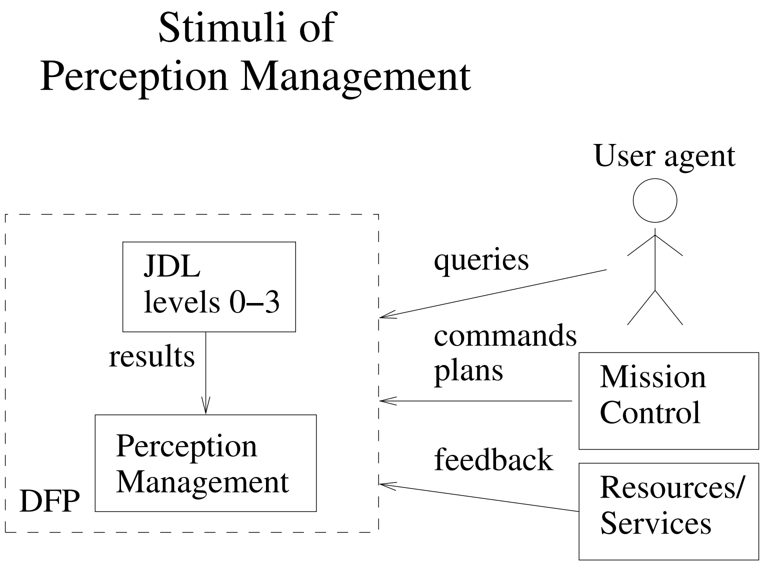


Figure 2.5: The figure shows three sources of external stimuli that affects the perception manager. Internal stimuli comes typically from the different levels of the data fusion process.

labour, and 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 to implement 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 available). The DFP, however, by definition, 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 accessible through usage of resources).

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 objectives control when requesting the same resources.

A manageable subset of the perception resources, that is able to perceive the environment, 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 analysts (financial applications), and human observers and signal intelligence services (command and control).

For an implementation of perception management, it might be useful to create a hierarchy of *perception services*, rather than individual resources, to control. There are several reasons for this approach. First, control of a resource is often abstracted by necessity, i.e., control is achieved through a set of modes which encapsulates underlying design trade-off and optimisation (design choices which could not efficiently be improved by an external sensor manager) as argued by Blackman and Popoli (1999, pp. 990). Hence, control of a resource often involves initiating some process that manages the detailed operations of the resource. The service concept encapsulates both resource and detailed control.

Second, a sensing resource may provide several modes of operation, and rather than considering the resource as a single control object, we can consider the palette of modes, or services, that it provides.

Third, different constellations of sensors (or services for that matter) may form to provide more abstract services. Every such constellation along with some process to combine the sensor measurements could be considered a composite service. There are numerous examples of previous efforts that group resources and treat them abstractly in this way. Muir (1990) calls the encapsulation of sensors and data processing *virtual sensor* and Henderson and Shilcrat (1984) denote a similar concept by *logical sensor*.

The sensing services (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 idea is illustrated in Figure 2.6.

However, the convenience of a unified control space 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. Note that this problem exists already for ordinary perception resources, although 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.

Some useful qualifiers we use for services in Chapter 4 are *accessible*, *available*, and *feasible*. A service is accessible to a perception manager of a DFP if the manager is designed to reason about the service in question and request it (hence, this is a static property). A service is available to a perception manager if the resources necessary to realise the service are currently available (this is a dynamic property). More details about the availability of resources is found in Appendix A. Finally, a service is feasible for a perception manager if certain constraints originating, e.g., from the manager's time and accuracy requirements on observations, and from the

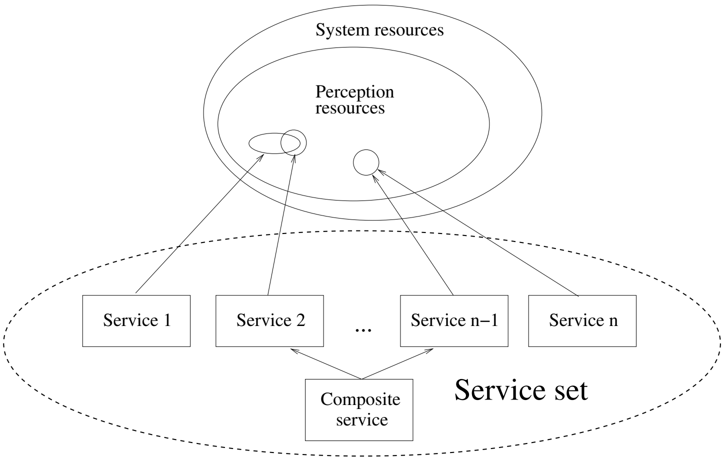


Figure 2.6: The interface between the perception resources and the DFP is the perception service set.

service's ability to serve the manager without causing damage to itself, are met. The feasibility of a service is exemplified in Chapter 4.

Control of Perception Management

As explained in Section 2.1, process refinement is inherently more a control function 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 perception management, we here suggest two types of control that should be considered: *system* and *individual control strategy.*

To understand the meaning of these strategies, we first need to separate the control from the perception resources or services. For instance, even though the control of a sensor is likely to be located close to the sensor itself, this should not be required.

System control strategies refers to the control and behaviour of the whole set of perception resources or services. This could be *centralised*, *decentralised*, or *hierarchical*. With the centralised system strategy, the actions of all resources are contemplated and issued by one single process node. 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).

With the decentralised strategy, 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 strategy has better scaling properties than the centralised one and is more flexible (nodes can be replaced dynamically). However, achieving coherence is a challenge and coordination must be considered (a survey of cooperating mobile robots is provided by Cao et al., 1997). We further discuss the issue of cooperation in Chapter 5.

A hierarchical strategy 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.

By individual control strategy, we refer to the strategy of an individual processing node (in the case of a centralised system, there is only one). In agent theory (and elsewhere), a number of control strategies for individual nodes have been proposed (some mentioned by Wooldridge, 1999). Arkin (1998) tries to capture the various types of individual control with a spectrum from *reactive* control on the one hand and *deliberative* on the other.

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

Table 2.2: Individual control strategy (Arkin, 1998)

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 strategy 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.

2.4 The Service Configuration Space

Based on the discussion in this chapter, we suggest a *service configuration space*[19] (SCS) $\mathcal{C}$ to succinctly express the sensing services of a system and the *degrees of freedom* they provide. The service configuration space concept presented here might not be appropriate for all decision situations, but will serve as a tool to explain the perception management strategy used in Chapter 4. The idea of the SCS is inspired by and vaguely resembles the configuration space concept in robotics (Lozano-Pérez, 1983). However, there is an important difference. In robotics, the configuration space is used to plan a sequence (or trajectory) of actions that brings a

[19] Others, e.g., Ostwald et al. (2005), also refer to sensor states as configurations.

robotic system from an initial configuration to a goal configuration. In our case, the objective is not necessarily to plan a path in the SCS. Instead, each configuration selection $\mathbf{c} = (c_i)_{i=1}^n \in \mathcal{C}$ (i.e., the service specification for n service attributes) results in a change of possible observations, represented by an observation space $\mathcal{O}$. We do not intend to strictly define the structure and contents of $\mathcal{O}$. The idea is simply that the selection of $\mathbf{c}$ should have some affect on the information one can acquire. For instance, in the extreme case that all sensors have been turned off or their data are ignored, $\mathcal{O}$ is clearly empty. Another example is that if we use all our sensor resources to look ahead for targets, we cannot expect the observation space to contain observations relating to targets approach from behind us. A third example is that if we are not using a range measuring sensor, we cannot expect $\mathcal{O}$ to contain observations that concern range.[20] The observation space $\mathcal{O} = \mathcal{O}[\mathbf{c}, \mathbf{x}]$ is dependent on both the service configuration $\mathbf{c} \in \mathcal{C}$ and the actual state of the environment, $\mathbf{x} \in X$.

Each dimension of $\mathcal{C}$ corresponds to a degree of freedom of a certain service. It should of course be possible to only control one service at a time if that is desired. In that case, only the configuration of the service in question is altered while the others are left fixed, e.g., if the service attribute altered is c_i, the configuration selection may change from $\mathbf{c} = (c_1, \ldots, c_i, c_{i+1}, \ldots)$ to $\mathbf{c}' = (c_1, \ldots, c_i', c_{i+1}, \ldots)$.

A simple example with two services is given in Figure 2.7. Here, we have a SCS with two attributes, c_1 and c_2. Attribute c_1 could refer to a microphone that can be either switched on or off. The service for control attribute c_2 could be the continuous orientation of camera mounted on a pole that can rotate a full 360°. The resulting partially continuous SCS $\mathcal{C}$ is the two lines of joint configurations in Figure 2.7.

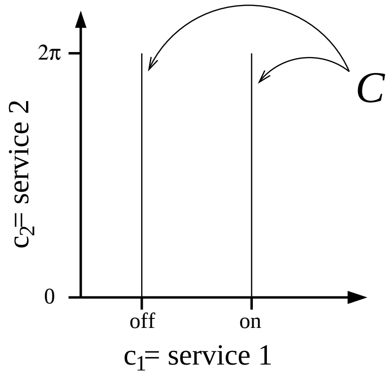


Figure 2.7: A two service configuration space

In the example, both sensors (the camera and the microphone) only had one

[20] Note, however, that such information could, in some cases, be inferred or estimated from a series of observations.

control parameter each. In some cases, services have multiple configuration attributes. A camera might, e.g., have both pan, tilt and zoom attributes. If so it will be useful to group attributes that belong to the same service. If the camera is represented by a service s_c, the configuration subspace controllable through s_c can be denoted by $\mathcal{C}_{s_c} = \mathcal{C}_p \times \mathcal{C}_t \times \mathcal{C}_z$. If then m services are considered, the complete SCS becomes $\mathcal{C} = \times_{i=1}^{m} \mathcal{C}_{s_i}$.

We tentatively envision at least three ways for a decision-maker to express its actions on the service configuration space:

1. a single configuration selection $\mathbf{c} \in \mathcal{C}$;

2. a configuration *plan* or *schedule* (e.g., a time-independent causal sequence of configurations $(c_1, c_2, \ldots)$, a time-dependent configuration function $f : T \to \mathcal{C}$ or perhaps a decision tree);

3. a configuration policy which maps perceived states to service configurations, $\pi : X \to \mathcal{C}$.

Policy generation, planning and scheduling are discussed elsewhere in the literature and we will focus on single action selection for simplicity.

In Section 2.3, we mentioned the two individual control strategies: reactive and deliberative. In this thesis, we focus on deliberative strategies that contemplate service configurations, the estimated state of the environment, observation space characteristics and system objectives.[21] Deliberative perception management for selecting a single configuration from the space involves selecting a "best" (according to some criteria) configuration with respect to some system objectives. Formally, we can express it as establishing a preference relation, $\succeq_{\mathrm{SOC}}$, for the system's objective's control (SOC) over the set of configurations, preferring configurations which are non-dominated according to $\succeq_{\mathrm{SOC}}$.

Admittedly, establishing a complete $\succeq_{\mathrm{SOC}}$ is inefficient as finding the most preferred configuration is sufficient for optimal decisions (while the intra-ordering of inferior configurations is unimportant). Especially for non-ideal problems with large configuration spaces, finding the complete $\succeq_{\mathrm{SOC}}$ is impractical and approximative solutions may have to be sought. Nevertheless, the preference relation $\succeq_{\mathrm{SOC}}$ represents the ability of a perception manager to evaluate and compare configurations to find a most preferred one.

Example We use the work by Cook et al. (1996) to show how to use the service configuration space. Cook et al. consider a military scouting mission with unmanned ground vehicles. In our terminology, the service configuration space of their vehicles has three attributes: the position of the vehicle ($\mathcal{C}_1$), the vehicle sensor ($\mathcal{C}_2$) to use, the geographical area to scan using the sensor ($\mathcal{C}_3$). Hence, $\mathcal{C} = \times_{i=1}^{3} \mathcal{C}_i$. The selected objectives of the application are to maximise

[21] Note that a data fusion process support deliberative strategies per se since it is its purpose to create and maintain an interpretation of its environment.

> the expected value of acquired information while avoiding exposure of the vehicle's presence to a presumptive adversary. A utility function $u(\mathbf{c} \in \mathcal{C})$ is constructed which integrates the conflicting objectives of information value and maintaining stealth. The expected value of information is dependent on the observation space $\mathcal{O}$ achieved from $\mathbf{c}$ and the value accredited to different observations. The utility function realises a preference ordering ($\succeq_{\mathrm{SOC}}$) over the available service configurations.

The discussion about the service configuration space so far concerns centralised control. In Section 2.3, we mentioned decentralised control as an alternative. We express decentralised service configuration space (which we further discuss in Chapter 5) in the following way.

Assume, once again, that we have a system containing various sensing resources. The capabilities of the resources can be expressed in terms of services and the service configuration space is denoted by $\mathcal{C}$. We, furthermore, do not have one but a set of agents $\mathcal{A} = \{a_i\}$. Each agent can control at least one of the configuration attributes. For instance, if a_i is responsible for services $S_i = \{s_{i1}, \ldots\}$ its service configuration space is $\mathcal{C}_{S_i} = \times_j \mathcal{C}_{s_{ij}}$ or expressed as a projection of the complete system service configuration space onto the attributes of S_i, i.e., $\mathcal{C}^{\downarrow S_i}$. In the decentralised case, the individual parts of the complete $\mathcal{C}$, the $\{\mathcal{C}^{\downarrow S_i}\}_i$, are typically controlled asynchronously by the agents. The control of the complete service configuration space is then partitioned and determined by the individual preference relations of the agents, $\{\succeq_{a_i}\}_{i=1}^{|\mathcal{A}|}$. In Chapter 5, we discuss *coordination* of multiple agents. A coordination protocol is presented which allow the agents to affect each other's preference relations.

2.5 Summary

In this chapter, we attempt to situate the information acquisition part of data fusion systems and highlight its properties. The focus on data fusion systems is a negligible 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, which 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. The purpose of this new terms is, e.g., to situate information acquisition in an agent theoretic context and to draw the focus from sensor devices to information.

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 perception management. Actions involve the usage of resources, perception resources in particular (a term which 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. We also presented a control space for perception management called service configuration space.

The dependence of perception management on the other functions of the data fusion process should be stressed. It is often inevitable that the 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.

Part II

Properties of Large-Scale Information Acquisition

Chapter 3

The Emergence of Large-Scale Information Acquisition: A Literature Survey

Information acquisition is fundamental for efficient decision-making. “Manual” information acquisition, 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 information acquisition 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 (scouts and spies) and tame and trained animals. Later on, we learned to construct tools to enhance our sensing capabilities (e.g., binoculars).

In the recent times, 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 information acquisition. 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*). However, in the literature review in this chapter, we restrict our attention to efforts for autonomous and semi-autonomous control of sensing resources. Autonomy will become an increasingly essential competence of information acquisition in future intelligent systems. One motivation for this is that it will cater to the need for rapid and efficient processing of extensive data and information quantities; requirements that could not be met by manual information acquisition.

As argued in Chapter 2, a system engaged in information acquisition closely resembles an *agent*, i.e., an entity that is situated in some environment which it perceives and acts upon. 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 related subjects that are important for agent perception and, hence, for information acquisition. information acquisition does not just contribute to better decisions by selecting the optimal (in some sense) sensing actions is itself a decision-making problem. Action selection[1] is about selecting actions to pursue some, perhaps, conflicting system goals/objectives (Humphry, 1997, pp. 6). Although, the interpretation of "action" is normally an action that explicitly makes the system pursue its objectives,[2] it could as well concern sensing actions to perform information acquisition.

Resource and *sensor management* are topics that to a high extent are related to information acquisition. 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 information acquisition is inherent.

If information acquisition was restricted to simply 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;
- an optimal action can not be described; or
- one does not always know what information to acquire.

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 briefly describe some different fields that are related to information acquisition (the important function for decision-making) and, second, to outline *large-scale* information acquisition (LSIA) systems. By LSIA system, we mean

[1] Sometimes called *behaviour selection*

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

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

We also assume that it is used in a “challenging” environment, that is, e.g., both inaccessible and dynamic.[3]

Section 3.1 explains the boundaries of this chapter. Section 3.2 discusses and exemplifies the relevance of information acquisition in a selected set of research fields. Based on the literature review, Section 3.3 identifies two important aspects of information acquisition: facilitation and focus of attention. Section 3.4 describes a useful model of the relationship between the actual environment and the observers internal representation of the same. Section 3.5 briefly describes a taxonomy of the three types of perception activities for information acquisition which we propose. Section 3.6 through 3.8 give examples of literature that belong to each of the three activities. Section 3.9 surveys literature that deals with facilitation of information acquisition and Section 3.10 surveys literature that deals with focus of attention. In Section 3.11, we attempt to characterise LSIA. The final sections, Section 3.12 and 3.13, provide a brief summary and conclusion, respectively.

3.1 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 only some subfields are covered to ensure focus on the issues of relevance to the present study. Hence, 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 explore candidate properties of automatic LSIA in physical 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, Hall and Llinas (2001), Hall (1992), Abidi and Gonzalez (1992), and Waltz and Llinas (1990) instead.

3.2 Related Fields of Research

To explore and develop techniques for automatic and semi-automatic information acquisition, it is useful to closely study its context and related fields of research. In Section 3.3, we suggest three types of activities and two aspects of information acquisition to decompose the subject into parts that can be studied and considered

[3] Environment and sensing resource properties are discussed in Section 2.1 and Appendix B, respectively.

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 information acquisition 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 reflects the high interdisciplinary importance of information acquisition and related techniques.

In the Agent Framework

To situate and motivate information acquisition 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). 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 basic agent architecture.

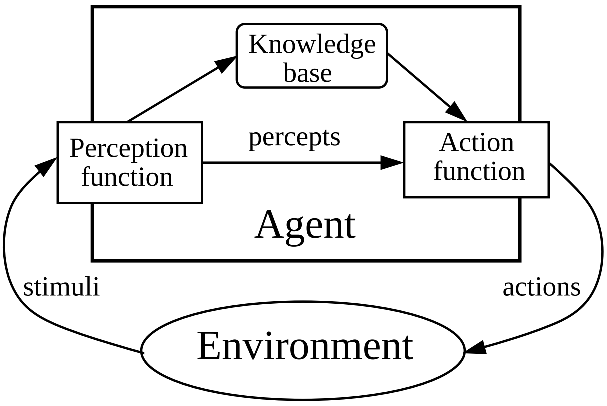


Figure 3.1: A simple agent model

With this agent definition, we are willing to claim that virtually every system with an interest in information acquisition 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).

An agent activity that realises information acquisition is called *information gathering* in agent theory. A model for the related perception process has been offered by Weyns et al. (2003). It appears, however, to mostly deal with the internal aspects of process refinement (presented in Chapter 2) and not the external aspects that we focus on.

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 information acquisition (e.g., techniques in decision-making and resource allocation). Conversely, being an integral part of agent technology, advances in the theory of information acquisition contribute to research in agent theory.

Decision-Making and Action Selection

information acquisition 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 a decision-theoretical problem. Second, acquisition of information supports decision-making by providing the decision-maker with useful information.[4] 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 information acquisition is (normally) to have as little effect on the environment as possible[5] while acquiring information to support goal-directed decision-making.

Howard (1966) extends the ideas of Shannon's *information theory* (Shannon, 1948) to a formalisation of the value of acquirable information, i.e., so called *information value theory.* Furthermore, Russell and Norvig (1995) use information value theory 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 by Pearl (1988, Ch. 6). 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 by Pirjanian (1998) and Humphry (1997).

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

[5] 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.

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 the one of Paradis et al., 1997). Resource management is discussed by, for instance, Bender (1983). The purpose of the aforementioned 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*.

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 information acquisition discussed in Section 3.3.

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 affect the directional thinking by describing what resources are missing, if any, to satisfy the objectives.

Resource administration deals with planning and control of resources. The resource allocation and administration subprocesses both encompass most of information acquisition including the other feature mentioned in Section 3.3, facilitation.

Since resource management is a much wider problem than that of information acquisition, we shall not discuss it any further. However, it is important to bear in mind that information acquisition is a part of resource management, and that resources necessary for information acquisition might be allocated by resource management if those resources are needed for system tasks of higher priority.

More directly related to information acquisition is sensor management. Sensor management is a natural subset of resource management[6] and its goal is loosely to "manage, coordinate, and integrate sensor usage to accomplish specific and often dynamic mission objectives" (Ng & Ng, 2000).

Blackman and Popoli (1999) 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 the ones by Xiong and Svensson (2002), Ng and Ng (2000), and Musick and

[6] To emphasise this relationship, it is sometimes called sensor resource management (Blackman & Popoli, 1999).

Malhotra (1994). Furthermore, Benaskeur (2002) includes an overview of sensor management tasks and requirements.

Computer Vision

In the field of computer vision, information acquisition is represented by the concepts of *active perception* and the more specific *active vision* (when only visual sensors are involved). Active perception is roughly defined by Bajcsy (1988) 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 (Aloimonos et al., 1987).

The ambition of Rimey (1993) is similar to ours, but restricted to computer vision. In a way similar to this thesis, Rimey incorporates the vision (sensing capabilities) into an agent model to emphasise the importance of the context of the sensing system.

Borotschnig et al. (1999) and Pinz et al. (1996) address the problem of information acquisition for fusion of information. 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's evidential theory and possibility theory) and compared.

Tarabanis et al. (1995) survey *sensor planning* for 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[7]), 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

[7] Pose means position and orientation.

configurations using the information acquired about the world to this point." The focus of the survey by Tarabanis et al. (1995) is on the first type.

Robotics

Robotics represents the physical incarnation of agents, and, thus, naturally inherits the need for information acquisition. Actually, the need for perception and related challenges are accentuated in robotics since its relationship with a real physical environment is inherent.[8] Mobile robots use sensing resources, such as cameras, sonars, laser scanners, primarily to avoid obstacles, detect relevant objects and to map its surroundings.

A theory for information gathering in robotics is presented by Hager (1990). It entails four basic principles: task-direction, uncertainty, computational and representational limitations. Task-direction acknowledges that the origin of information need is the tasks the system has to perform to fulfil its objectives. In our work, we discuss this issue briefly in Section 2.3 and more extensively in Chapter 4. Uncertainty in information is an inherent feature of information acquisition. We mention this issue in Section 2.1. Computational limitations concern the fact that the amount of resources available for reasoning about actions may be limited. In this thesis, computational limitation is considered in Chapter 5. Finally, representational limitations concern the issue of representing measurements and knowledge. We both support and discuss this principle in Section 3.4.

Hager, furthermore, suggest two objectives for information gathering: to achieve an "accurate enough" description of the environment (for the mission at hand) or to maintain one.

An interesting distinction also made by Hager (1990) is 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.

The method for information acquisition presented by Hager (1990) 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 a sensing action is considered to determine its feasibility. 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.

Manyika and Durrant-Whyte (1994) present a complete data fusion process for decentralised multi-sensor systems. 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.

[8] A real physical environment normally has the most difficult environment properties described in Section 2.1.

Indirect Fields of Research

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

3.3 Salient Aspects of Information Acquisition

We say that the system skill of information acquisition 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 3.5.

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 (Perillo & Heinzelman, 2003) 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 Appendix A).

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. Conversely, 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 3.4 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 3.5 classifies perception activities depending on the way they contribute to the aforementioned environment representation. Literature amenable to this classification is surveyed in Section 3.6 through 3.8. Literature representative for the information acquisition aspect of facilitation are discussed in Section 3.9 and focus of attention in Section 3.10.

3.4 In the Eye of the Beholder

For the continuing discussion about perception activities, we need to provide a context for information acquisition. 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 information acquisition 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.

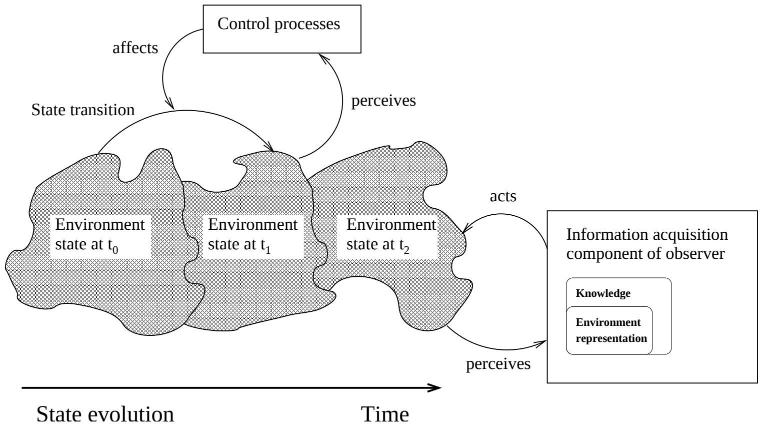

Control processes
affects
perceives
State transition
Environment state at t_0
Environment state at t_1
Environment state at t_2
acts
Information acquisition component of observer
Knowledge
Environment representation
perceives
State evolution
Time

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.

The control processes that affect the evolution of the environment state appear 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 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 and the *environment representation* (ER) of the observer. We consider the environment state to be a complex entity whose qualities may (partially) be observed. We do not attempt to parameterise the environment and characterise it with variables. 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 be perceived and represented by 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, on the other hand, is typically composed of a jumble of discrete and continuous variables and hypotheses on the (true) values of the variables, representing the belief of the observer. The information in the ER is rarely fully reliable and carries meta-information about imprecision and uncertainties (due to the evolution of the environment and imperfect and incomplete observations). 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 system objectives decide (by affecting the focus of attention aspect) 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 information acquisition.

Another salient feature of the ER is that it, in contrast to the environment state, is a composition of information 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.* The ER and internal state representation jointly constitute the belief of the observer. Some information in the internal state representation is quite reliable (e.g., battery power) if the agent has no reason to disbelieve its internal sensing,[9] and other less reliable (e.g., the agent's exact location in the environment).[10] 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 the environment state, but also the state of the control processes that influence the evolution of the environment state.

3.5 A Taxonomy of Perception Activities

There are basically two types of information that an information acquisition 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,

[9] The process of measuring internal state is sometimes referred to as *proprioception* (Russell & Norvig, 1995, pp. 782).

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

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. We deal with the latter type in Chapter 4, where we perform information acquisition for the purpose of revealing plans.

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 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 literature classification is model-based rather than technique-based, meaning that we categorise perception activities depending on the type of information they provide, rather than the techniques they employ. All types of information acquisition might involve techniques such as management of uncertainty (which motivates use of Dempster-Shafer's theory of evidence, Shafer, 1976, Bayesian inference, Pearl, 1988 and fuzzy set theory, Zimmermann, 1991, etc) and optimisation (which motivates use of mathematical programming techniques, evolutionary algorithms, etc).

We propose the following taxonomy of activities for information acquisition, based on the model of the relationship between the environment and internal state explained in Section 3.4:

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. Target tracking, e.g., is a monitoring 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 information acquisition 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.[11] Conversely, a discernment activity, by identifying the true class 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 LSIA 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* (Albus, 1999), *the data fusion and resource management tree architecture* (Bowman & Steinberg, 2001), and *the logical sensor/actuator* framework (Budenske & Gini, 1997).

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

3.6 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 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 deal 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.

Penny (1998) assumes 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[12]

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

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

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

where $P_d(D|T = \mathbf{x}, r_k)$ is the probability of sensor k, at position r_k, detecting a target at position $\mathbf{x}$.

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.

Penny (1998) models the individual detection probabilities approximately with 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.

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. A target is considered to have been detected when one sensor has reported detection a fixed number of times.

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.

For McCarragher (1998) and Hovland and McCarragher (1997) the objective is to activate or utilise 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 2.3) 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 in which the sensors should be activated for every state of the assembly. During the assembly process, if the event detection confidence of the first selected sensing process 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, McCarragher (1998) 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 Cook et al. (1996) is to find observation positions for mobile sensors in an antagonistic environment (see Section 2.1 for an explanation), where they are both likely to detect interesting objects and unlikely to be observed themselves. The proposed solution, which is based on decision theory, consists of two parts: trajectory planning and camera parameter selection (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.

3.7 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 defective 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 formulate 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 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 accuracy 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 information acquisition), 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* (Arulampalam et al., 2002).

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 belief of the environment it is observing.

Updates can be performed using Bayes' theorem,

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})}, \tag{3.1}$$

where the normalising denominator is

$$p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})d\mathbf{x}_k \tag{3.2}$$

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

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{x}_{k-1}. \tag{3.3}$$

This formulation of the Bayesian recursive filter is sound, but unfortunately impractical in general. 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 assumptions. 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 that by Nash (1977) 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.

Benameur (2000) addresses the problem of how 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, Kalandros and Pao (1998) select (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 multi-sensor 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. Schmaedeke and Kastella (1998), e.g., let one filter track uniform motion and another turning motion. The estimation of every filter 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 sensor-to-target allocations, the system evaluates their *expected discrimination gain*. This is the information-theoretic Kullback-Leibler (KL) measure[13] 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 by Schmaedeke and Kastella (1998) is extended by Dodin and Nimier (2001) 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 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 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

[13] Also called cross-entropy.

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 $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ 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 $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ that is approximated.

Doucet et al. (2002) consider the problem of selecting one sensor from a set of sensors to observe a target. 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, $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$. 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 by Kreucher et al. (2003) 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 pdfs 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 approaches the KL measure when α approaches one. The authors, using this information measure, arrive at fairly simple objective function compared to the one by Doucet et al. (2002).

Just as Kreucher et al. (2003), Mahler (2003) addresses the issue of selecting sensing actions to track an unknown number of targets. The information theoretic objective function developed, however, is somewhat different.

Hernandez et al. (2002) describe sensor management for tracking of a single target. This work considers the time at which sensor should be deployed not to lose the track, and how many and where the sensors should 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 works by Arulampalam et al. (2002), Bergman (1999). The basics of Kalman filtering is thoroughly explained by Bar-Shalom and Fortmann (1988).

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.

3.8 Discerning

For applications where the environment to sense is real and physical, 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 of Castañón (1997). The classes of objects are further assumed not to vary over time. Sensors 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 by Castañón is most suitable when the observed scene is static and the observation scope of the sensors always include the objects.

Contat et al. (2002, 2001) describe 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 a pair of membership functions are separated. The information acquisition 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.

Lee and Ro (1999) model (possibly heterogeneous) sensors to yield as a unified output a tuple including an estimate of an environment feature and a corresponding uncertainty measure. The method for information acquisition 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 by Klein and Sequeira (2000) 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 by Borotschnig et al. (1999) who 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's theory of evidence. In all approaches, the camera movement $\Delta\Psi$ that reduces an entropy 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 camera 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 of Roy (2000). The known objects of the system are represented with so called *aspect graphs*. Each node in a graph represents an aspect of an 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 the initial 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 information acquisition activities which involve combining the competences of information heterogeneous sensors (defined in Appendix B), e.g., sonar, camera, etc. to acquire different (i.e., complementary) types of information to fall into the discernment category. However, we are not aware of any such works in the literature.

3.9 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 an inherent part of perception activities. The control process of relocating sensor platforms, for instance, is an example of a well integrated facilitation aspect of a perception activity. The process itself does not yield any observations, but it may facilitate observations by, e.g., a camera by relocating the camera to reduce occlusion. Another example is 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 involve techniques that explicitly support acquisition of information, even though others might also be critical for successful practical information acquisition (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 Appendix A). 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 (Cook et al., 1996), illuminating a scene for image acquisition (Tarabanis et al., 1995), dynamical formation of sensor resources (i.e., forming abstract sensors) for fusion of target position estimates

(Dodin & Nimier, 2001), etc. Note that scope management, just like resource constraint management changes the expected utilities of sensing actions.

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. Perillo and Heinzelman (2003) address this facilitation issue (Section 3.3) 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 by Kannan et al. (2002). 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.

Karan and Krishnamurthy (2003) have 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.

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 prespecified locations is addressed by Soliday (1999). The problem is similar to the well known, and NP-Complete, 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.

Berenji et al. (2003) present 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.[14] This particular Q-learning is

[14] Q-learning and other types of reinforcement learning are surveyed by Kaelbling et al. (1996).

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 (Tarabanis et al., 1995), an example of scope management is that of illuminating a scene before acquiring an image.

3.10 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 should 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 LSIA, where many information needs might compete for resources it is essential to evaluate candidate sensing actions with respect to the usefulness of their expected outcomes (see Howard, 1966 for a discussion on this matter).

Hintz and McIntyre (1999) and McIntyre (1998) introduce the idea of *goal lattices* 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 is 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 (which could be interpreted as relevance or priority) of the top goal is one, then the values of its included 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 McIntyre, 1998). 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 gets the highest priority.

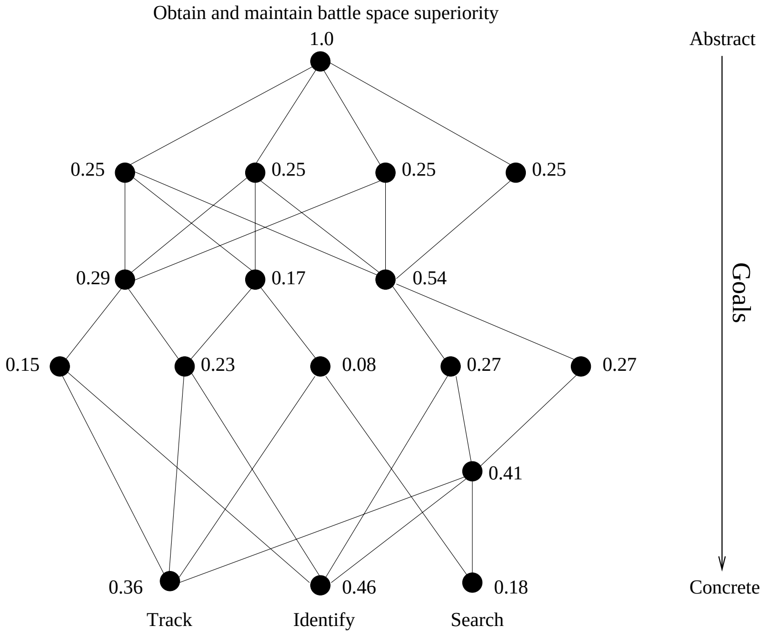


Figure 3.3: An example of a goal-lattice by McIntyre (1998). The top nodes in the Hasse diagram are abstract goal, whereas the bottom nodes represent goals which can be treated with resources directly.

The focus of attention issue is also addressed by López et al. (1995). There, the management of sensor resources is 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.

3.11 Large-Scale Information Acquisition

With the apparently everlasting increase in the number of available sensing resources, the demands on sensor systems will increase. Likely initial application fields for information acquisition 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 LSIA. However, to successfully manage the resources and enjoy the anticipated advantages, new techniques must be developed.

To realize LSIA, 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 3.3). To initiate and maintain the environment representation (Section 3.4) it is likely that perception activities of all three mentioned types (Section 3.5), 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 and be adapted to environment properties (Section 2.1).

In the context of LSIA, sensing resources are plenty and heterogeneous, i.e., they differ in the their control properties and the 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 managed. 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.[15]

Constraints of heterogeneous resources (e.g., interference, mutually exclusive modes) and sensing opportunities (e.g., relocating sensors to alter sensing scope) make it important to facilitate observations (Section 3.9). Finally, focus of attention is essential to decide what kind of information and what activities are beneficial to the system objectives (Section 3.10).

We are not aware of any effort that addresses a larger subset of the aforementioned challenges related to information acquisition. However, there are a few recent DARPA sponsored projects that appear to be moving in that direction. The first is by Horling et al. (2001) who uses (potentially) many stationary sensors for target tracking. The most interesting aspect of their work for LSIA is perhaps its facilitation aspect. 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. By this convention, communication costs are kept low, and the system becomes scalable.

The second work is by Charles L. Ortiz et al. (2001). The objective is also in this case target tracking, but here sensors are mobile which inflates the potential observation scope. Scaling is handled similarly as by Horling et al. (2001); 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 be exchanged from one zone to another (if necessary) and a *sampler coalition leader* agent controls the sensors within a

[15] Xiong and Svensson (2002) make a distinction between short-term and long-term sensor management strategies and include some references to works that deal with the latter.

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 LSIA, Charles L. Ortiz et al. (2001) and Horling et al. (2001) make good attempts to facilitate the use of a large number of sensors. However, in other respects they do not contribute as much to LSIA. For instance, both works are limited to the perception activity of monitoring and they only consider homogeneous sensors.

3.12 Discussion and Conclusion

With the increasing availability of perception resources such as sensors, LSIA appears likely 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 currently being considered. The concept of network centric warfare, 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.

Our research on LSIA is, furthermore, encouraged by the recent introduction of the *opportunistic* information fusion concept (Challa et al., 2005), which aims at discovering and exploiting shared resources.

We consider (at least) the following properties to be important for LSIA:

- decentralised control (it is not feasible to manage large amounts of sensors using a centralised approach);
- multiple and varying objectives (LSIA may serve several, possibly conflicting, information needs);
- flexible acquisition (acquisition for integration of different types of information, such as 'new' information (incorporation), 'updated' information (monitor) and more 'detailed' information (discern));
- flexible control (control for sensing (perception), sensing scope (facilitation) and focus (attention)).
- services (defining services and expressing dependencies between the underlying sensors)

LSIA also takes a comprehensive view on sensing resources which could have some of the following qualities:

- large quantities (requires strategies for which sensors to use and when);
- complex (e.g., multi-modal sensors, mobile, etc)
- distributed (provides a greater sensing scope);

- heterogeneous (sensors have are controlled in different ways or yield different kinds of information).

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

Finally, we summarise the properties for information acquisition, extracted from the literature survey and the discussions in Chapter 2, in Figure 3.4. As mentioned, the classification of literature in this chapter, in perception activities and information acquisition aspects, is neither final nor complete. It rather 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 by Musick and Malhotra (1994).

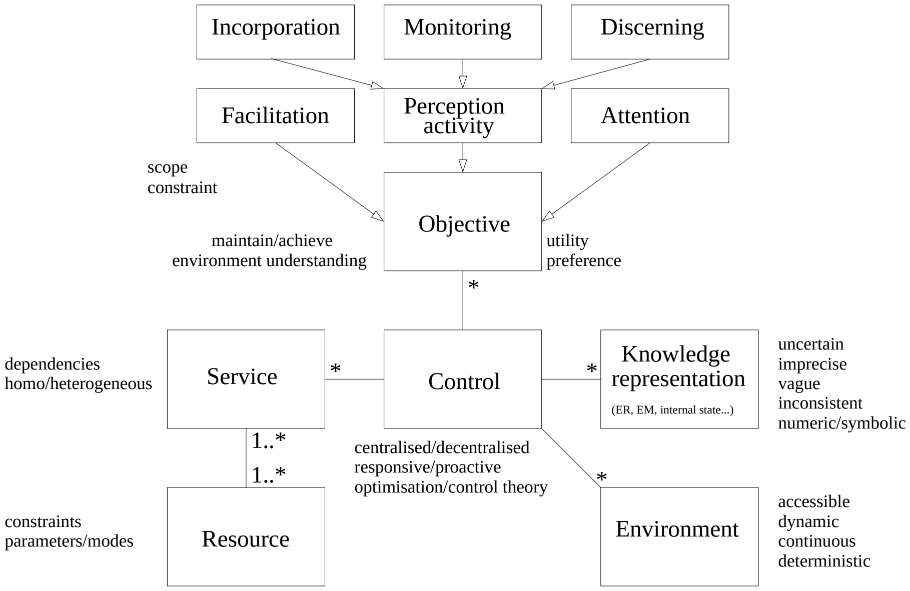


Figure 3.4: UML domain model of research aspects related to information acquisition.

3.13 Summary

In this chapter, we endeavour to derive important aspects of information acquisition from previous efforts. We claim that information acquisition is an interdisciplinary issue that is dealt with, directly or indirectly, in, e.g., sensor management, robotics, and agent theory. Much can be learned from all of these fields. A second objective of the chapter is to elaborate on the specific properties of large-scale information acquisition.

Within the body of work related to information acquisition, many different aspects have been studied and highlighted. The aspects we have discovered are summarised in Figure 3.4.

Since we consider that information acquisition for data and information fusion should focus on the information ultimately attained (rather than sensor device properties), we investigate how different research efforts affect an observer's understanding of its environment. We suggest that the environment understanding is represented by an environment model, i.e., a computational data structure representing, e.g., entities known to exist by the observer. The environment representation, i.e., the current belief of the observer, is then an instantiation of the environment model based on observations.

In our study, considering our focus on the effect of information acquisition on the environment representation, three different classes of efforts emerge: incorporation, monitoring and discerning. Briefly, incorporation is a perception activity that enters new entities into the environment representation; monitoring updates the understanding of previously detected entities; and discerning refines the understanding of a detected entity.

Furthermore, there are two aspects of perception activities that stand out to a varying degree in different efforts. Those are facilitation and focus of attention. Sensing actions that do not directly acquire information, but which support sensing actions, constitute the facilitation aspect of perception activities. Focus of attention involves deciding what kind of information is most important given the objectives of the system.

We conclude that, to fully understand and exploit large-scale information acquisition, we need to be aware of and master the three types of perception activities, the aspects of facilitation and focus of attention, and the properties listed in Section 3.12.

Although large-scale information acquisition seems to be a promising topic for the future, very little attention has been devoted to its characterising properties to date.

Chapter 4

Bridging the Gap Between Information Need and Acquisition

The purpose of the work in this chapter is twofold. One is to introduce a generic framework for expressing the transition from information need to acquisition. The structure of the proposed framework is derived from the characterisation of large-scale information acquisition (LSIA) in Chapter 3. The second purpose is to concretise the framework in an application that combines high-level information need with sensing actions.

With the framework, we seek a holistic description of the process from the recognition of information need to its acquisition. By holistic we mean that the framework should be flexible enough to express various kinds of information acquisition problems, including those which involve multiple (possibly heterogeneous) sensing resources, multiple conflicting objectives and information need on different levels of abstraction. These are properties of large-scale information acquisition, and, hence, we will call the framework the *large-scale information acquisition framework* (LSIAf). The need for holistic descriptions of parts of information fusion has recently been requested and addressed by other researchers, e.g., Llinas (2003) who assumes a target tracking perspective.

By high-level information, we generally mean information artifacts that belong to the higher levels of the JDL model.[1] Examples of high-level information are relations between observed objects or other not directly observable (but inferable) properties. In our application, the high-level information is belief of actions or plans of some observed (enemy) actors. A key property of high-level information, in this discussion, is that it is inferred from observations and that it is computationally costly to evaluate the (expected) impact of sensing actions on the high-level information state.

The beliefs of plans, that we are considering, are represented by estimated distributions, $\pi(h)$, over modelled plan alternatives h_i. An example of a plan alternative,

[1] The JDL model is described in Chapter 2.

in our case, is `attack`. The generation of such distributions is performed by a so called *plan recognition* process. The estimated plan distributions provide a tentative explanation of the observations made of an actor. In our research, we realise plan recognition using a dynamic Bayesian network (DBN) that encodes a priori knowledge about the situation (e.g., actors' organisation, plan alternatives, etc and environmental properties) and integrates current observations with previous estimates of plan distributions.

Predictive situation awareness (Piccerillo & Brumbaugh, 2004) is the projection of a situation into the near future. Plan recognition is one of the methodologies that are aimed to support predictive situation awareness. Plan recognition gives users hints about what the actor is going to do next given sensor information and a priori knowledge about the actor. Due to high complexity and uncertainty, even experienced tacticians are only able to consider two or three possible courses of action for all but the simplest situations (Phister et al., 2003). Moreover, we reuse the high-level information as stimulus for *perception management* that, in our case, takes into account both derived *threat* estimate and its *uncertainty.*

The performance of plan recognition is heavily dependent on the acquired information. If sensor resources are limited and cannot provide *relevant* information in a timely fashion, the results of the plan recognition will be poor. By relevant we mean information that concerns actors whose estimated plan distributions indicate that they are going to be a threat to our own activities. Furthermore, data from observations are only relevant if they are amenable to treatment by the plan recognition. Hence, the information acquisition or perception management aspect (encapsulated within JDL level four) is crucial for plan recognition.

A third contribution of this chapter, less stressed in this thesis, is its integration of plan recognition with particle filters. We represent the state uncertainty of the actors using particle filters (Arulampalam et al., 2002). A particle filter is flexible enough to express non-parametric distributions to an arbitrary degree of precision (controlled by the number of particles used). The particle set representation of uncertainty turns out to be convenient when Monte Carlo sampling from the state uncertainty distribution and we, furthermore, exploit the state transition part of the particle filter to estimate future state uncertainty to direct UAV sensor platforms.

Section 4.1 describes the framework we are proposing. Section 4.2 compares LSIAf to various architectures and control structures. Section 4.3 presents an example scenario and how it is expressed in the terms of LSIAf. Section 4.4 give some details on how the plan recognition is implemented using a hierarchical DBN. Section 4.5 explains how the state uncertainty of actors is represented and how it evolves with particle filter updates. Section 4.6 explains our approach to integrate the particle filter state representation with plan recognition and provide two means to estimate plan distributions. Section 4.7 describes how information acquisition is realised here; mainly in the shape of task prioritisation. Section 4.8 provides an outcome of a simulation and discusses the result. Section 4.9, finally, summarises the chapter.

4.1 A Framework Connecting Information Need to Acquisition

In this section, we present a framework, LSIAf, that emphasises key issues in modelling and implementing the part of the data fusion process (JDL level four) that connects information need to the acquisition of information. One such issue is the recognition of the context of information acquisition (i.e., the possibly conflicting objectives that have to be dealt with in the agent context described in Section 2.1). To facilitate decentralised solutions, LSIAf separates resources from information need.

To approach the problem of resource sharing to support plan recognition processes, we propose a framework that emphasises aspects which we believe are useful to flexibly connect information need to information acquisition (Johansson & Suzić, 2004). Information need is interpreted as a lack of information about the state of the environment, that if it was relieved, is believed to improve the decision-making of the system that uses the plan recognition. Aspects that LSIAf tries to capture include: multiple information needs or objectives, heterogeneous sensors, dependencies among tasks and services, and a separation between the actual sensing resources and the interface of services they provide.

The general structure of LSIAf (depicted in Figure 4.1) involves two types of entities: *space* and *function*. The four space entities: *task origin*, *task*, *service* and *resource* are containers of structured information. The structure of information of each space entity should suit the intersecting function entities: *task creation and management*, *allocation scheme*, and *service management and resource allocation*. The purpose of the function entities is to convey information between its adjacent space entities.

LSIAf prescribes that information need arising in some system (we call the source of this need the *task origin space*) is formulated as *information tasks* with assigned properties (e.g., priority or time horizon depending on what properties the system is designed to handle). Such tasks belong to the *task space* in Figure 4.1.

The materialisation of tasks from information need could be the responsibility of a *task creation and management* function. The *service space* contains services that the sensors in the *resource space* (independently or jointly) can perform. The benefit of utilising these services to satisfy tasks is, subsequently, the (more) relevant data that eventually is returned to the data fusion process by employed sensors. The *allocation scheme* describes how tasks are connected to feasible services.

Note that LSIAf does not suggest that the bridging procedure (from information need to its acquisition) is centralised in any way; tasks and services might be distributed and maintained separately (and are perhaps only made available locally) and the allocation scheme might be decentralised as well.

The tentative ontology, expressed in UML, of the framework shown in Figure 4.2 may further clarify the relations between the framework entities.

The following subsections will describe the different parts of LSIAf in Figure 4.1

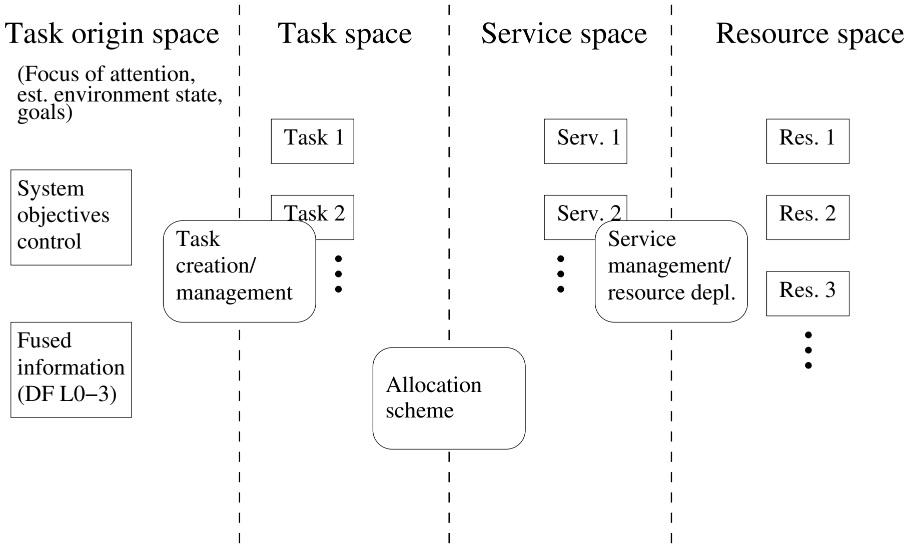

Task origin space
Task space
Service space
Resource space
(Focus of attention, est. environment state, goals)
System objectives control
Fused information (DF L0–3)
Task creation/ management
Task 1
Task 2
Allocation scheme
Serv. 1
Serv. 2
Service management/ resource depl.
Res. 1
Res. 2
Res. 3

Figure 4.1: The large-scale information acquisition framework (LSIAf)

in more detail.

Task Origin Space

The space that contains the sources of all information needs we call the task origin space. Information need is mapped to the *task space*, where tasks represent requests for information useful for a successful operation of the enclosing system. The creation of information tasks is spurred by the members of the task origin space, such as goals and focus of attention of the system objectives control, and by the data fusion process itself. Thus, the character of the information tasks that emerge is influenced by the current information need of the system and demands of ongoing fusion processes.

The space might, e.g., contain one sole static objective that is the only source that affects the information acquisition, or, in the other extreme, it is populated by an ever changing set of appearing and disappearing objectives. Objectives might require information to maintain a certain level of quality of the state estimation, or a specific piece of information required for a sporadic decision.

Task Space

The task space contains the information tasks spawned by the system's desire for more information (explained in the previous section). Describing interdependencies between tasks might facilitate efficient information acquisition. For instance, dif-

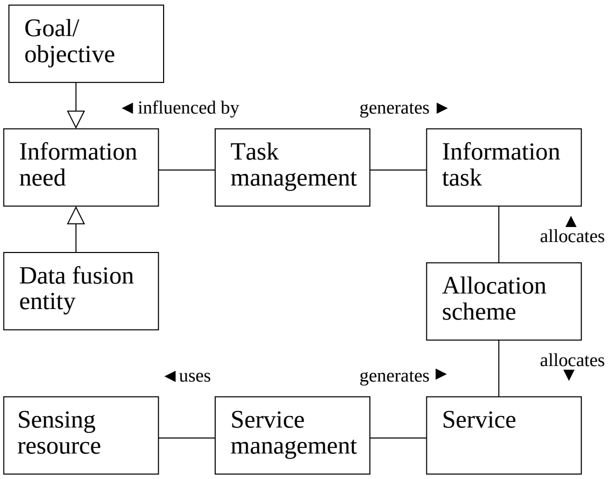


Figure 4.2: The ontology expresses the relations between the framework's space and function entities.

ferent information tasks might overlap, in which case that interdependence should be noted. Tasks, typically, also have various attributes that are discussed in more detail in the next section.

Task Creation and Management Function

Given the contents of the task origin space, the task creation and management function forms information tasks that can be compared both to services and other tasks. The structure and information content of tasks is heavily dependent on the allocation scheme used (described in a later section). The structure is defined by various attributes such as *priority* (e.g., a value reflecting the importance of the tasks), *deadline* (a time point after which the task is irrelevant and should be discarded), *duration* (how long a continuous task should be served), and *origin* (the process which requested the information). Tasks could also express whether they must be solved completely (with a least quality requirement) or if partial solutions are acceptable.

Overlapping tasks, i.e., that have some *commonality*, could be marked as dependent if the system wants to benefit from serving multiple tasks simultaneously. Task decomposition may be performed, e.g., to compare tasks and to find commonalities.

Note that the structure of tasks does not have to be identical for all tasks. For instance, tasks might be treated locally in clusters, where each cluster might have its own task structure.

Service Space

Rather than connecting tasks directly to sensing resources, we propose an intermediate layer of services. The services we think of here are the same as defined in Section 2.3.

The service space (ideally) defines the complete action space of sensor management. The space should only contain *available* services which can be achieved given available resources. It could possibly also contain services that are currently unavailable, but which are known to be available at a given moment or during a predictable time interval.

Similarly to tasks, interdependencies between sensors may be expressed. Particularly, interference dependencies are of the essence. For instance, the operations of two services may interfere with each other by one emitting energy that will preclude the operations of the other. Furthermore, the engagement of a service may disable another candidate service since the resources that made it available are preoccupied with the engaged service.

Allocation Scheme Function

It is the responsibility of the allocation scheme to connect information tasks to feasible services. By feasible we mean that the allocation scheme may refuse the allocation of a service to a task, even if it is available, e.g., if the usage of the service for the particular task jeopardises the survival of the underlying resources. The result is typically also a configuration specification for selected services (e.g., by performing scheduling and selecting control parameters). It is clear that the design of this function for a particular application is heavily dependent on the structure of tasks and services.

The allocation scheme (i.e., the assignment of tasks to services) may appear in different guises. It may, for example, be a centralised scheme that collects all tasks and performs an exhaustive search over all combinations of tasks and services. A drastically different approach is to design tasks and services as agents which negotiate over allocations in a decentralised manner.

This function also performs re-prioritisation of tasks. The reason for reconsidering the prioritisation conceived by the task management function (in Section 4.1) is that high priority tasks, e.g., might not be satisfiable in the near future given the available services. In that case, a task which originally was assigned a high priority by task management has to yield in favour of low priority, but satisfiable, tasks. An example is the *contract net protocol* depicted in Figure 4.3. Here the allocation scheme function does not reside in a single node somewhere within the system. Rather it is collectively implemented by the nodes of the system.

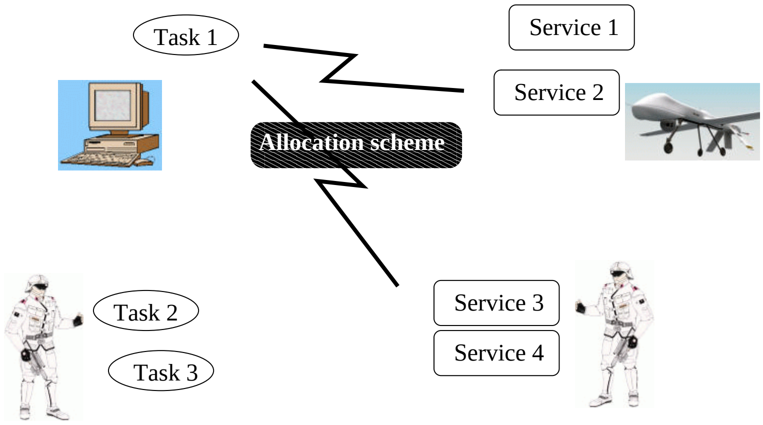


Figure 4.3: An example of a distributed allocation scheme using the contract net protocol.

Service Management and Resource Deployment Function

The function of service management and resource deployment has two main chores: populating the service space with feasible services, and making sure that resources fulfil their obligations towards initiated services.

Available services are, as stated in Section 2.3, services which can be achieved with available resources. Services which are probabilistically achievable (based on the system's belief in their availability and reliability) are also conceivable. Note that flexible sensing resources may concurrently serve multiple services, e.g., by distributing their capabilities over different services.

4.2 LSIAf Compared to Other Architectures

In this section, we relate LSIAf to other known structures of sensor management and information acquisition. We first relate LSIAf to control architectures and then we present a summary in Table 4.1 of a comparison with architectures with respect to the functions and spaces of the framework. The details of the comparison are described in Appendix C.

Centralised and Decentralised Control

A fundamental issue when designing a sensor management system is whether the control should be centralised or decentralised. The distinction between the two is described in Section 2.3.

Many efforts in sensor management have assumed centralised control, perhaps because the focus in that line of research has been on optimal solutions rather

Table 4.1: A summary of how different architectures contribute to the functions and spaces of the LSIAf.

Architecture	**Task Origin**	**Task Man. and Space**	**Alloc. Scheme**	**Service Man. and Space and Resource Space**
dMARS	*unspecified*	percept interpreter and task prioritisation	plan execution	*unspecified*
GPGP/TÆMS	*unspecified*	task decomposition	scheduling	*unspecified*
OAA	*unspecified*	client agents	facilitator agent	service agents
$\mathcal{M}/\mu$	*unspecified*	macro manager	macro manager and micro manager	micro manager
FMDNN	*unspecified*	goals	management node	sub-ordinate management nodes
Goal lattice	*unspecified*	goal lattices	scheduler	*unspecified*
Multi-platform SM	platform operators and tasks	tasks created if complying with platform policy	sensor to task assignment and scheduling	platform sensors (typically radar)
ASN	common mission objective	implicit	team decision-making	*unspecified*

than on control structures. Decentralised control is, however, highly relevant for decentralised fusion and the emerging field of network centric warfare (see Alberts et al., 1999 for a general introduction to NCW), and will likely receive more attention in the future. In NCW, sensors are assumed to form extensive networks and share information among themselves. In suchlike applications, the centralised control approach is simply unreasonable due to its lack of scalability.

Both centralised and decentralised control can be expressed in LSIAf. In the case of centralised control, typically, tasks are created and managed in a centralised fashion. The allocation scheme is also centralised. It uses the knowledge the system has acquired to consider all possible sensing actions (or more frequently, for complexity reasons, the optimal action is approximated through a search in the space of coherent sensing actions). Resources are inherently distributed (unless the application concerns only a single platform), but the attached service representation and management reside in the centralised node. Efforts that employ a centralised control concern, e.g., sensor placement (Penny, 1998) and target-tracking (Schmaedeke, 1993). In both of the efforts, the task space only contains one fixed task. In the former article, it is the task of achieving the best detection probability, and in the latter that of the best expected improvement of the target state estimate.

In systems with decentralised control, tasks are created and maintained locally in distributed information nodes (i.e., nodes which require information for various missions). Likewise, services are created and maintained in sensor nodes. Hence, the allocation scheme will here have to find a suitable allocation and configuration of services that are distributed. One means to handle this problem is the contract net protocol (see Figure 4.3 and Smith, 1981) which represents tasks and services as agents. *Managers* (task agents) announce tasks to be solved and *contractors* (service agents) respond with bids reflecting the cost (expressed in, e.g., time or energy) the enclosing system will suffer if selecting the service.

An example with decentralised sensor control is offered by Dodin and Nimier (2001) who present a multi-target tracking problem for decentralised fusion on multiple platforms. Expressed in LSIAf, there is really only one task (which also is constant): maximising expected measurement accuracy on the system on all targets. The services are the platforms themselves which each can take measurements of one target at a time. For complexity reasons, the allocation scheme in this case approximates the optimal solution. The resulting allocation of sensor platforms to targets is achieved after some cooperative work performed by the sensor platforms.

4.3 Plan Recognition and Decision Support

In this section, we use the LSIAf for an application where the information need originates from multiple plan recognition processes. Plan recognition is the estimation of the current plan of some antagonistic actor observed in a *mission-relevant* environment.[2] The plan recognition models and inference used here is due to the efforts of Suzić.

The purpose of plan recognition is here to support some *information consumers* acting (e.g., performing a mission) in an uncertain environment. The consumers are, furthermore, assumed to be interconnected through a network that connects a set of nodes. The network structure facilitates, e.g., *reliability* (through the redundancy of multiple communication paths between nodes in the network) and *flexibility* (through information exchange between arbitrary nodes in the network). Each node of the network is assumed to have at least communicational and computational skills. The individual success of a consumer is dependent on the result of the its (local) plan recognition for each observed actor, but the resources used to acquire the information that fuels the plan recognition process are shared among consumers.

We do not make any assumption about the network concerning, e.g., topology, communication protocols and information security. Many of those questions have to be settled by the designer of the network, based on available technology, resources, and possibly of the designing organisation's policies. However, we do require that the network is capable of conveying information throughout itself and that it (somehow) can collect sensor measurements and perform tracking based on this information. An approach to that problem is offered by Brännström et al. (2004).

Network nodes should also be prepared to share fused and inferred information with interested nodes, and to assign tasks to *sensing resources* (i.e., sensors) and perform sensor reconfiguration. At this point, we do not make an attempt to describe how this could be performed.

An instance of the general problem discussed above, which we simulate and present in this chapter, is the scenario depicted in Figure 4.4. It concerns an extensive geographic environment including two consumers located in the middle of the view (`a1` and `a2`). The two consumers have individual goals (e.g., for `a1` it is to defend the city in which it is located), but belong to the same network. They both perform plan recognition based on information about actor states provided by the network. In the scenario there are nine (hostile) actors, i.e., *platoons*. Groups of three platoons belong to a *company*. There are two companies near the perimeter to the north (labelled `cn1` and `cn2` respectively) and one in the far south of the view (`cs`). There are two types of resources modelled. One type is the UAV observer which can travel quickly but can only give state estimates from a distance (to ensure its own security). The other one is the ground soldier who is limited in speed but who can hide itself close to the road and make comparatively precise state estimates of a passing actor. The network has duties such as collecting measurements to track hostile actors and to configure and engage sensors in information acquisition tasks (i.e., sensor management).

[2] The part of the observable environment that can have any effect on the mission held by the observer.

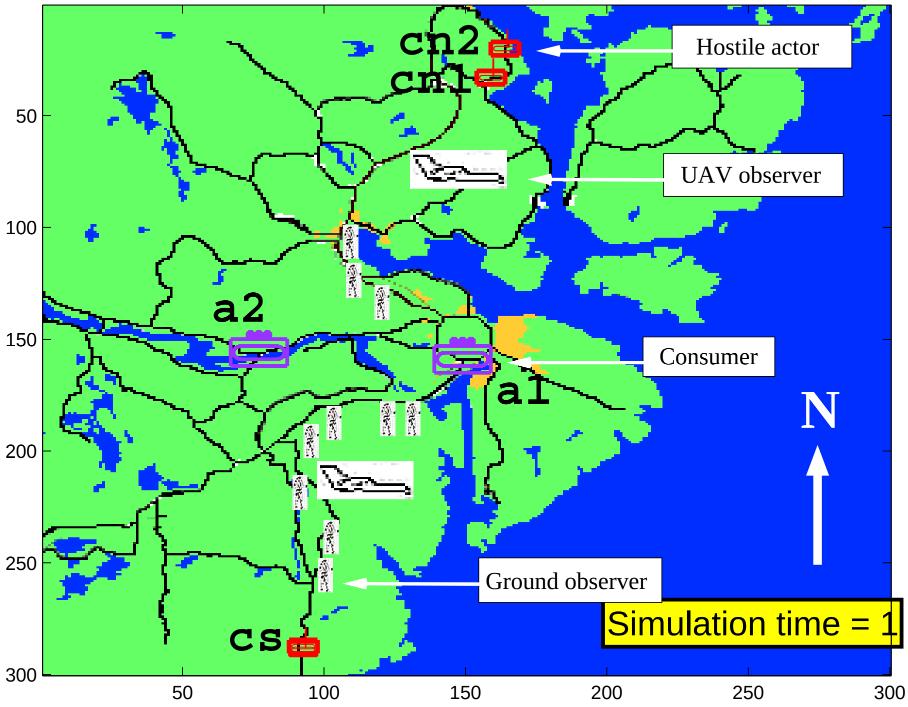


Figure 4.4: Scenario for use of plan recognition

The scenario, containing the hostile actors and their movements and the locations of the consumers, is fixed in all of our simulations. The sensor composition, i.e., the number of sensors, their locations, and their properties, however, may vary from simulation to simulation.

In the scenario, the cs travels north, initially constituting a *threat* (we define the concept of threat in Section 4.7) to both of the consumers (a1 and a2). Later a bridge between cs and a1 is destroyed and cs has to retreat.

The companies in the north are initially stationary, not constituting an imminent threat, but become more aggressive in the later part of the scenario when they both continue to travel south towards the city.

The objective of a consumer is to know as much as possible about the varying *threats* derived from plan recognition estimates of the hostile actors. Threat estimation is defined in Section 4.7.

For the scenario above, the LSIAf functions and spaces manifest themselves in the following way:

- **Task origin space (TO):** It is populated by two origins, i.e., the two consumers $a_1, a_2 \in \mathbf{TO}$. As the state of the environment evolves over time, the two consumers are in constant need of new information (i.e., state estimates) of the hostile actors.

- **Task management and space:** Tasks are created every time step in the simulation from the information need expressed by the consumers. The tasks in the task space $\mathbf{T} = \{\tau_i\}_i$ are simple and basically only expressed in terms of the actor to observe (α) and a priority value (*prio*), i.e., $\tau_i = (\alpha, \mathit{prio})$.

- **Resource space (R):** Contains UAVs and ground soldiers. Both types can make observations, but they have different properties (i.e., the resource space is heterogeneous): UAVs are fast, but make uncertain observations. Ground soldiers on the other hand are slow, but their observations are more certain.

- **Service management and space:** In our implementation, for each resource in **R** there is a corresponding service s in the service space **S**. The service management makes sure that the resources are properly configured (e.g., have flight paths) when associated with tasks.

- **Allocation scheme:** Connects created tasks to available services, based on their priorities. Connected pairs of tasks and services are referred to as *allocations*, e.g., $\mathit{alloc} = (t, s) \in \mathbf{Allocs}$. A service may refuse to perform a task if the cost of using the resource for the task at hand is too high. Services may also be *preempted*, i.e., removed from one task and connected to another.

The next section will reveal some details of the implementation of the plan recognition process.

4.4 Plan Recognition Using Dynamic Bayesian Networks

In the work of Suzić (2003b), uncertain sensor information, terrain information, and uncertain a priori knowledge about the enemy are inferred to obtain estimations of enemy plans on different abstraction levels. We utilise a hierarchical Dynamic Bayesian Network (DBN) model (see, e.g., Murphy, 2002) for this purpose.[3]

A Bayesian network, to begin with, is a graphical model of a joint multi-variate probability distribution which reduces the complexity of full table of joint probabilities to a set of interdependent, less complex, tables. The variables, in our case, represent plans and factors (some of which are sensor observable) that we (or, as would be preferred, a domain expert) believe affect the actors selection of plans. The graph representation consists of nodes for variables and directed edges between conditionally dependent variables.

[3] We agree with Murphy that *temporal* Bayesian network is probably a more suitable name since *dynamic* suggests that the network itself (with its nodes and edges) is non-static (which is not the case). Also, recently, an interest in (truly dynamic) Bayesian networks with changing structures has arisen.

What distinguishes a DBN from an ordinary Bayesian network is that it has a temporal dimension, i.e., that some variables are affected by their previous estimates. In Figure 4.5, we present a simplified DBN that has been used for our plan recognition. The DBN models a military hierarchy and produces probability distributions over plausible plan alternatives from observations of the actions and states of tanks; a priori knowledge of military organisation; and environment factors such as traversability, cover and weather. The temporal aspect of this DBN is expressed in the company and platoon plan variables from a previous time step (e.g., `Company Plan (t-1)` which was the inferred plan distribution in a previous time step $t-1$).

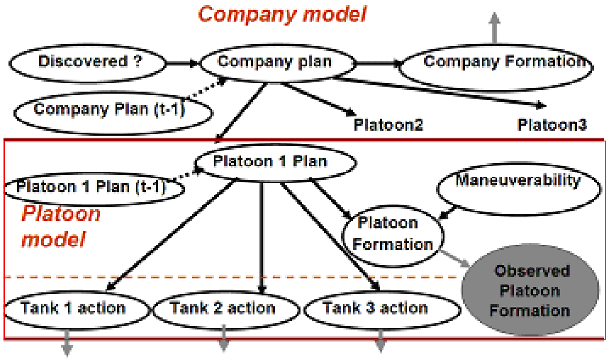


Figure 4.5: A simplified DBN used for plan recognition. The ellipses represent stochastic variables and the arrows causal relationships between variables. By courtesy of Robert Suzić.

We, furthermore, call our DBN *hierarchical* since it models both plans of platoons and plans of their aggregate, the company. Note that this hierarchy is also a hierarchy of information, from low-level information close to sensors, to more abstract high-level information (i.e., plans on company level cannot in general be inferred with confidence by just observing a single platoon).

Our methodology combines a set of fuzzy functions that insert sensor data as soft evidence into the DBN. The DBN represents knowledge about the enemy actors, their doctrine, and the environment. The DBN estimates plans and the result is a discrete distribution π over plausible plans for each enemy unit on each abstraction level (i.e., on platoon and company levels). On the platoon level, e.g., the plan space contains the plans: `attack`, `defence`, `reconnaissance` and `march`. Their corresponding probabilities are π(`attack`), π(`defence`), etc. Hence, a plan is a label that can be added to an actor's state, and that gives the consumers some idea of what the actors might be up to.

In our application, for the scenario in Figure 4.4 both consumers (`a1` and `a2`) receive information from the network of observations and perform their own (local) plan recognition independently.

In the next section, we will discuss how we represent state uncertainty of enemy actors.

4.5 Particle Filter-Based State Estimation

We represent uncertainty in actor state with particle filters. The particle filter offers us several useful properties: (i) the position uncertainty representation is more expressive (more distributions may be represented); (ii) the state estimate is robust to spurious observations, and (iii) Monte Carlo simulated threat estimation can be performed by simply drawing particles from the particle set.

In the particle filter algorithm, the posterior state probability $p(\mathbf{X}_t|\mathbf{z}_{0:t})$ at time t, where $\mathbf{X}_t$ is the actor state and $\mathbf{z}_{1:t} \triangleq (\mathbf{z}_1, \ldots, \mathbf{z}_t)$ is a sequence of observations (from time 0 to t), is inferred and approximately represented by a set of N particles $\{\mathbf{x}_t^{(i)}\}_{i=1}^N$. The discrete approximation $\hat{p}(\mathbf{X}_t|\mathbf{z}_{0:t})$ of $p(\mathbf{X}_t|\mathbf{z}_{0:t})$ is then expressed as

$$\hat{p}(\mathbf{x}_t|\mathbf{z}_{0:t}) = \frac{1}{N}\sum_{i=1}^{N} \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)}), \tag{4.1}$$

where $\delta(\mathbf{x})$ should be interpreted to be the Kronecker delta which equals 1 when $\mathbf{x} = 0$ and 0 otherwise.

The inference at each time step is generally performed in two steps: *importance sampling* and *selection*. In the importance sampling step, for each particle $\mathbf{x}_t^{(i)}$ a sample $\tilde{\mathbf{x}}_{t+1}^{(i)}$ is drawn from the *state transition* distribution $p(\mathbf{X}_{t+1}|\mathbf{x}_t^{(i)})$. The purpose of the state transition distribution is to predict the future actor state distribution. Each sample is, furthermore, associated with a weight related to its *likelihood* given a new observation $\mathbf{z}_{t+1}$, i.e., $\tilde{w}_{t+1}^{(i)} = p(\mathbf{z}_{t+1}|\tilde{\mathbf{x}}_{t+1}^{(i)})$. The weights are then normalised. In the subsequent selection step, N particles are drawn (with replacement) from the weighted set of samples. This set is the posterior at time $t+1$ and used as the prior in the next time step.

The particle filter algorithm we use is the one by Doucet et al. (2001, pp. 11) slightly modified. The reason we chose to modify it is because observations may sometimes be infrequent. As a consequence, the particles will drift randomly (possibly away from the area of interest) and also, if a new observation is made it may have little effect since the closest particle may be far away from the observation (i.e., in that case even the most likely particle is unlikely). To counter these problems, we add three modifications to the original algorithm: (i) we let the particles (through the transition model used in the prediction step) be attracted by the consumers; (ii) the particle filter is re-initialised whenever observations are too far away from the nearest particle (i.e., a fraction of the particles are relocated to the vicinity of the observation); and (iii) we discount the weights of particles corresponding to states where an actor should have been observed (i.e., the utilisation of so called *negative information*, Sidenbladh, 2003). The rationale behind these modifications is that we use some extra a priori information already available (i.e., a generic estimation of the opponents' intentions and that a correct observation should be reflected in the configuration of the particle set) to heuristically improve the performance of the particle filter.

In the following sections, we describe the details of our transition and likelihood models.

State Transition Model

We describe the state of each particle $\mathbf{x}_t^{(i)}$ by four values. The first two values represent the position of the particle in $\mathtt{x}$ and $\mathtt{y}$-coordinates. The following two are direction, ϕ, and speed, $|v|$.

$$\mathbf{x}_t^{(i)} = [\ x \quad y \quad \phi \quad |v|\ ] \tag{4.2}$$

An actor is influenced by physical and doctrinal properties. Our transition model takes the following factors into account: *previous state* $\mathbf{x}_t^{(i)}$, *terrain properties*, *strategic sites* (such as own forces). The samples $\{\tilde{\mathbf{x}}_{t+1}^{(i)}\}_{i=1}^{N}$ from the state transition distribution are drawn according to Algorithm 4.1.

Algorithm 4.1: State transition model

```
(1)  foreach particle in ${\mathbf{x}_t^{(i)}\}_{i=1}^{N}$
(2)      Randomly choose a strategically important place $c$ from
         the set of consumers in the task origin space TO
(3)      path ← find the shortest road path from $\mathbf{x}_t^{(i)}$ to $c$
(4)      $\hat{\mathbf{e}}_g \leftarrow$ calculate the direction of $c$ from $\mathbf{x}_t^{(i)}$ in path
(5)      $\hat{\mathbf{e}}_v \leftarrow (\cos(\phi), \sin(\phi))$
(6)      $\hat{\mathbf{e}}_r \leftarrow (\hat{\mathbf{e}}_g + \hat{\mathbf{e}}_v) / \|\hat{\mathbf{e}}_g + \hat{\mathbf{e}}_v\|_2$
(7)      $(x_{\text{new}}, y_{\text{new}}) \leftarrow \hat{\mathbf{e}}_r \cdot |v| + (x, y)$
(8)      $(x_{\text{new}}, y_{\text{new}}) \leftarrow$ most_likely_neighbor$(x_{\text{new}}, y_{\text{new}})$
(9)      $\tilde{\mathbf{x}}_{t+1}^{(i)} \leftarrow [\ x_{\text{new}} \quad y_{\text{new}} \quad \phi + \Delta_\phi \quad |v| + \Delta_{|v|}\ ]$
```

We run our state transition algorithm for each time step, i.e., we propagate particles for each time step with respect to strategic sites (attractors), particles' previous state $\mathbf{x}_{t-1}^{(i)}$ and local terrain properties.

Each particle propagation is *independent* of other particles, but is influenced by a global property that we call *gravity*. In our military scenario, we say that particles are attracted to *strategically important places* (lines *2-4*), e.g., the consumers. To calculate the gravity vector, $\hat{\mathbf{e}}_g$, we calculate the shortest terrain path. Therefore $\hat{\mathbf{e}}_g$ is not in the direction of the straight line between the particle and the consumer. Instead, $\hat{\mathbf{e}}_g$ is calculated based on positions of the particle, terrain and strategic site. Our shortest path is the shortest distance between the particle and the randomly chosen strategic site, given terrain restrictions. The shortest path consists of a number of terrain nodes and the traversability costs associated with edges between nodes. After calculation of the shortest path, the gravity vector is pointing in the direction of the shortest path's first node.

In the next step of the algorithm (lines *5* and *6*), a direction vector, $\hat{\mathbf{e}}_v$, is added to $\hat{\mathbf{e}}_g$. This calculation gives us a resulting vector, $\hat{\mathbf{e}}_r$. Given the resulting vector's direction and speed, the new position of the particle is calculated in line *7*.

The new position of the particle is adjusted to local terrain properties. Each particle position in our discrete terrain representation can be associated with a traversability cost. We specify the size of the local surrounding as a parameter. We place a particle at the position of minimum cost in the specified surrounding (line *8*). In line *9*, we add white noise to the particle state (Δ_ϕ and $\Delta_{|v|}$).

Finally, particles that end up outside the area of interest are replaced by copies of other predicted particles, where the selection probability over predicted particles is uniform.

Likelihood Model

Inspired by Lichtenauer et al. (2004), we propose the following likelihood function for sensor observations given a sample $\tilde{\mathbf{x}}_t^{(i)}$:

$$p(\mathbf{z}_t|\tilde{\mathbf{x}}_t^{(i)}) \propto \begin{cases} \epsilon + 1, & d{<}d_s \\ \epsilon + e^{-(d-d_s)^2/(2\sigma_s^2)}, & d{\geq}d_s \end{cases} .$$

Here $\mathbf{z}_t$ is the observation at time t. The parameters of the likelihood distribution, d_s and σ_s, are sensor specific and related to the accuracy of the sensor, and d is the Euclidean distance (in the $\mathtt{x}$ and $\mathtt{y}$-coordinates) between the particle and the observation. The shape of the likelihood distribution is shown in Figure 4.6. Therefore, all particles within a circular distance with radius d_s of the observation will receive the same weight. For instance, in our experiments, we use two types sensors with different levels of accuracy. The small $0 < \epsilon \ll 1$ improves the particle filter's ability to recover a track when observations are scarce.

When we have several concurrent observations (from different sensors), say k^+, of the same platoon, we let the joint likelihood be the product of all the individual likelihoods for each observation, i.e.,

$$p(\underline{\mathbf{z}}_t|\tilde{\mathbf{x}}_t^{(i)}) = \prod_{j=1}^{k^+} p(\mathbf{z}_t^j|\tilde{\mathbf{x}}_t^{(i)}). \tag{4.3}$$

The conditional independence expressed in Equation 4.3 is correct under the assumption that the detection of a sensor is only dependent on the true state of the actor, not on the states of any other sensor.

To incorporate negative information ('missing' observations, i.e., observations that should have occurred if our particle set state hypothesis was correct), we introduce a symbol $\underline{\mathbf{z}}_t^-$ that represents all the sensors (say k^- such sensors) that didn't

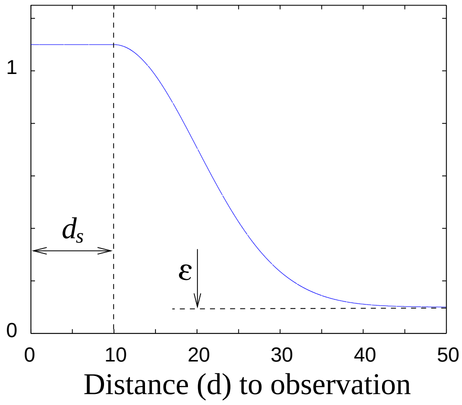


Figure 4.6: Likelihood model

make an observation. Thus, instead of weighing the particles with the likelihood $p(\underline{\mathbf{z}}_t|\tilde{\mathbf{x}}_t^{(i)})$, we calculate the value of $p(\underline{\mathbf{z}}_t \cup \underline{\mathbf{z}}_t^-|\tilde{\mathbf{x}}_t^{(i)})$, i.e., the likelihood of $\tilde{\mathbf{x}}_t^{(i)}$ given the observations $\underline{\mathbf{z}}_t$ and the missing observations for sensors $\underline{\mathbf{z}}_t^-$. The likelihood of a missing observation for a sensor l can be denoted $p(\neg D^l|\tilde{\mathbf{x}}_t^{(i)})$, where $\neg D^l$ is a symbol that means that $\tilde{\mathbf{x}}_t^{(i)}$ did not cause a detection by sensor l. As this is the probabilistic complement to a detection by l, we conclude that

$$p(\underline{\mathbf{z}}_t^-|\tilde{\mathbf{x}}_t^{(i)}) = \prod_{l=1}^{k^-} p(\neg D^l|\tilde{\mathbf{x}}_t^{(i)}) = \prod_{l=1}^{k^-} \Big(1 - p(D^l|\tilde{\mathbf{x}}_t^{(i)})\Big) . \tag{4.4}$$

We model the detection probability (for simplicity) to be linearly decreasing with the distance to a (hypothesised) target state $\tilde{\mathbf{x}}_t^{(i)}$,

$$p(D^l|\tilde{\mathbf{x}}_t^{(i)}) = \begin{cases} 0, & \text{if } d > d^+ \\ (d^+ - d)/(d^+ - d^-), & \text{if } d^- \leq d \leq d^+ \\ 1, & \text{if } d < d^- \end{cases} , \tag{4.5}$$

where d is the Euclidean distance between the sensor and the geographical position of the sample $\tilde{\mathbf{x}}_t^{(i)}$, d^+ is the maximum range of the sensor (beyond which the sensor is not expected to be able to detect anything), and d^- is the maximum distance from the sensor within which a detection is certain.

Hence, the combined likelihood model for a sample $\tilde{\mathbf{x}}_t^{(i)}$ given k^+ observations and k^- missing observations becomes

$$p(\underline{\mathbf{z}}_t \cup \underline{\mathbf{z}}_t^-|\tilde{\mathbf{x}}_t^{(i)}) = \prod_{j=1}^{k^+} p(\mathbf{z}_t^j|\tilde{\mathbf{x}}_t^{(i)}) \prod_{l=1}^{k^-} \Big(1 - p(D^l|\tilde{\mathbf{x}}_t^{(i)})\Big) . \tag{4.6}$$

We don't investigate this approach to deal with negative information any further. A more formal treatment of negative information is offered by Koch (2004).

In the next section, we will discuss how the state uncertainty representation of the particle filter can be used for plan recognition.

4.6 Robust Plan Recognition

Multi-agent stochastic plan recognition deals with stochastic outcomes of actions, uncertain observations, and incomplete knowledge of multiple actors (Bui et al., 2002). In military applications, the use of plan recognition could be valuable when utilising sensor data and is of decisive importance in achieving information superiority and predictive situation awareness. One reason is that sensors themselves are not able to reveal the actors' true intentions. Military commanders have to act agilely and they might not have much time (especially on the tactical level) to interpret all data. In some cases, the difficulty to recognise different patterns is caused by space-time separation, and limited capability to correlate patterns. In some cases, behaviour of individual actors can be classified as harmless, but put in some greater context such as actors' mutual interrelations, environment and their assumed doctrines, the threat might be identified as much higher. The methodology of plan recognition helps mitigating these difficulties by estimating plans.

In his original multi-agent stochastic plan recognition (Suzić, 2003b), it was assumed that a unique, unimodal, qualified guess on actors states, containing actors' positions and velocities, could be obtained at each time step. Later, we relaxed the problem by dropping the assumption of continual observations (Suzić & Johansson, 2004). The new approach was to infer plans only in cases when observations were received; between observations we assumed that the latest plan alternative was still valid. The latter approach implies that when new observations arrive, the new plan estimate could greatly differ from the previous plan estimate.

In our latest joint article (Johansson & Suzić, 2005), we introduced the particle filter described in Section 4.5 that maintains a state estimate, even when observations are lacking, by using our state transition model, i.e., $p(\mathbf{X}_t|\mathbf{x}_{t-1})$. The particle filter produces a discrete multi-modal state representation with each particle as a mode (Equation 4.1). This uncertain representation, unfortunately, cannot be used directly with our plan recognition model since the DBN requires exact state observations for its inference. One way could be to select an "average" particle of the particle set and bring that state estimate to the plan recognition model. However, bearing in mind that particles could spread in different directions, we do not consider such an approach to be sufficiently robust. The average of particle sets could represent states with no particles, i.e., unlikely states; and the expressiveness of the particle set (for representing generic probability distributions) is largely ignored.

Another way could be to find plan distributions for each particle and average over the acquired plan distributions. That is, however, prohibitively computationally costly when the particles are numerous. What we instead try to do is to collect a representative set of L particles from the particle set by drawing them randomly with replacement. Each member of the set is then, simply put, used to generate a plan distribution.[4] We denote the set generated plan distributions at time t by $\mathbf{\Pi}_t$ and obviously $|\mathbf{\Pi}_t| = L < N$.

We can visualise the set of plan distributions $\mathbf{\Pi}$ for decision-makers, as we do in the experiments in Section 4.8. It is, however, not a suitable representation to use as a previous state plan distribution in the DBN (e.g., the `Company Plan (t-1)` variable in Figure 4.5) as it requires single precise probability distributions (see the discussion on credal networks in Section 4.9 for a generalised graphical model that tries to encompass this situation).

In this work, we test two approaches to summarise the set of plan distributions $\mathbf{\Pi}$ to a single distribution π. The first approach is the average plan distribution that we denote π^{avg} and the second approach is an estimate based on maximum entropy principle (Grünwald & Dawid, 2004 and Jaynes, 2003), π^{me}. The maximum entropy principle has received some attention among researchers and its rationale is that it selects the distribution that is the least specific (i.e., least biased).

The first approach is based on finding the average distribution of probability for each plan alternative, h, given all plan distributions,

$$\pi^{\text{av}}(h) \triangleq \frac{1}{|\mathbf{\Pi}|} \sum_{\pi' \in \mathbf{\Pi}} \pi'(h), \tag{4.7}$$

where $\pi^l(h)$ is the probability of plan alternative h for $\pi' \in \mathbf{\Pi}$.

The second approach is choosing the estimate that has the maximum entropy compared to the other distributions in $\mathbf{\Pi}$,

$$\pi^{\text{me}} \triangleq \arg \max_{\pi' \in \mathbf{\Pi}} \texttt{Entropy}(\pi'), \tag{4.8}$$

where $\texttt{Entropy}(\pi')$ is the Shannon entropy,

$$-\sum_h \pi'(h) \log_2 (\pi'(h)) \, .$$

In Section 4.8, we compare the two approaches (Equation 4.7 and 4.8) in an experiment.

This section describes our approach to deal with the uncertainty representation of the particle filter for plan recognition. We call it *robust* since we acknowledge and deal with the state uncertainty represented by the particle filter, rather than just a single most likely state hypothesis. A second reason to call it robust is that we want to invite the methodology of robust Bayesian analysis (David Rios Insua, 2000), which explicitly deals with sets of probabilities, to our research. This idea is further discussed in Section 4.9 and by Suzić (Johansson & Suzić, 2005). The next section deals with information acquisition using the same uncertainty representation.

[4] In practice, it is a little bit more complicated since the DBN requires inputs from all three platoons in a company. Hence, for each estimated plan distribution, we need to draw a particle for each platoon.

4.7 Information Acquisition Through Perception Management

As previously stated, unless the plan recognition is supported by an appropriate information acquisition function, it will not produce useful results. The problem context of large-scale information acquisition then provides the following interesting challenges: there are multiple task origins, heterogeneous sensors, long time-intervals between initiated sensing action and result of the sensing actions, and resource constraints. We also add the requirement that we want to acquire the information that is the most relevant for the mission of the enclosing system (i.e., what plan estimates are the most relevant for the overall system objectives).

In the following discussion, we argue that an optimal solution to information acquisition is unachievable both in general and for our problem at hand. In the subsequent sections, we describe the details of our approximative solution which deals with several of the challenges of large-scale information acquisition.

The Challenge of Optimal Information Acquisition

To express the information acquisition problem, we make use of the service configuration space concept introduced in Section 2.4. The concept captures the possible actions in a service configuration space $\mathcal{C}$. In Section 4.3, we mentioned that we assume that every resource (UAV or ground soldier) is related to one unique service. Thus, if the set of UAV related services is $S_{\mathrm{UAV}} = \{s_{\mathrm{UAV},i}\}_i$ and the set of ground soldier services is $S_{\mathrm{GS}} = \{s_{\mathrm{GS},i}\}_i$, the service configuration space becomes

$$\mathcal{C} = \left(\times_i \, \mathcal{C}_{s_{\mathrm{UAV},i}}\right) \times \left(\times_i \, \mathcal{C}_{s_{\mathrm{GS},i}}\right), \tag{4.9}$$

where $\mathcal{C}_s$ corresponds to the control options of service s. As system designers, we are entitled to define the coarseness of the control options (modes or parameters) of each service. Here, for simplicity, $\mathcal{C}_s$ is made coarse. It simply contains the platoons of the companies mentioned in Section 4.3 (and the option of *no target* assignment), i.e., the selection of service configuration $\mathbf{c}$ results in an assignment of sensors to targets. Hence, due to the coarseness of $\mathcal{C}$, given a target assignment, a UAV will, e.g., have to solve its path planning problem itself (since that subproblem is not addressable with $\mathcal{C}$).

Before trying to express the properties of a “best” service configuration, we should stress that an optimal selection exists neither in general nor in our application. Basically, we want the information acquisition to support the data and information fusion processes to establish the best possible situation awareness (or environment representation, defined in Section 3.4). What constitutes a good situation awareness could be debated, but one that allows the overall system to efficiently achieve its objectives is a good choice.

If the contents of the environment representation is limited to the whereabouts of a single target, the information acquisition can be specified as the selection of a service configuration that minimises the expected uncertainty in the state estimate of the target. However, an optimal selection might still be unachievable, e.g., because

1. the number of possible configurations is too great for a timely evaluation, or
2. the consequences of a service configuration on the environment representation are too costly to estimate accurately, or
3. the environment might be evolving at such a pace (resulting in changing objectives and changes in available resources) that what would have been an optimal decision when the configuration selection deliberation was initiated is suboptimal once it has finished.

Furthermore, the decision situation rarely involves only a single objective. Multiple objectives make it even more difficult to express an optimal solution. Multiple objectives naturally enter into the decision situation when there is more than one user of the system. Even if there is not a group of users, there might be different mission objectives that have to be met concurrently, e.g., maximised information value and minimised resource usage. Multiple objectives are often summarised in a scalar utility function through a weighted linear combination which can be optimised if time allows. However, such a simplification is typically ambiguous (there are infinitely many possible weight assignments) and is only an approximation of the original problem. Are the units of the objectives even comparable?

Multiple objective decision-making is addressed elsewhere (e.g., Cohon, 2003) and we will not present any more solution alternatives in this chapter.[5]

The example problem in Section 4.3 is intentionally complex and possesses two of the complicating properties just mentioned. It has multiple objectives since it has to consider the utilities of the two information consumers and the cost of using resources. The second property is the evaluation of service configurations (number two in the list above). It is especially important as it appears in virtually any information acquisition application for information fusion.

We exemplify the problem with the latter property in the following way. Assume that we can summarise the objectives of our application in an acceptable way as the expected utility over the space of service configurations $\mathcal{C}$ given in Equation 4.9. Given a system environment representation, ER, we can also assign a utility, $u_{\text{SOC}}(\text{ER})$, to it reflecting its usefulness to the system objectives control (SOC). We can then, in terms of decision theory and Pearl (1988, pp. 316), express a best configuration $\mathbf{c}^*$ as the $\mathbf{c} \in \mathcal{C}$ that maximises the perception management utility, $u_{\text{PM}}(\mathbf{c})$, i.e.,

$$\mathbf{c}^* = \arg\max_{\mathbf{c}\in\mathcal{C}} u_{\text{PM}}(\mathbf{c}) = \sum_{\mathbf{o}\in\mathcal{O}} p(\mathbf{o}|\mathbf{c}, \text{ER})\, u_{\text{SOC}}\left(\underbrace{\text{DFP}(\mathbf{o}, \text{ER})}_{ER'}\right), \tag{4.10}$$

[5] Our approach to coordinate agents in Chapter 5 is another way to deal with multiple objectives, though.

where

$p(\mathbf{o}|\mathbf{c}, \mathrm{ER})$ is the probability of observation $\mathbf{o}$ given service configuration $\mathbf{c}$ and environment representation ER. In our work, an observation corresponds to a tuple $(\mathrm{target}, \mathrm{state}, \mathrm{sensor})$, i.e., which target was observed, what was its state (including its position and direction) and which sensor was used. This information is used by the particle filter tracker described in Section 4.5.

$\mathrm{DFP}(\mathbf{o}, \mathrm{ER})$ is the updated environment representation, ER', when observation $\mathbf{o}$ has been processed by the data fusion process (DFP) of the system. In our case, the DFP is the plan recognition process, and ER foremost contains the estimated plan distributions and particle filter state estimates. For simplicity, in Equation 4.10 we assume that the result of $\mathrm{DFP}(\cdot)$ is deterministic, which is not necessarily the case.

It is clear from Equation 4.10 that to optimise u_{PM}, we have to utilise a replica of the DFP to generate all hypothetical updated environment representations ER'. In our case, and in many other applications, it is time consuming just to perform the DFP on the "real" acquired observations, let alone multiple simulated ones.

Hence, a characteristic of information acquisition for information fusion is that the effect of a service configuration is costly to estimate. This problem for decision-making has recently been identified in another context. Ostwald et al. (2005) have a similar problem, where sensor measurements are processed by algorithms to classify meteorological phenomena. The authors approximate the meteorological algorithms with Bayesian networks. Interestingly, in our case the computationally costly process that we would like to replace is a (dynamic) Bayesian network, and we therefore seek a different approach.

In the following sections, we explain how we design a approximative solution by describing details of the implemented task management, service management, and allocation scheme.

Task Origin and Task Management Implementation

Tasks are generated by members of the task origin space, i.e., the consumers `a1` and `a2` (from our discussion in Section 4.3) who require information about the observed actors for their current mission. In the current work, each consumer maintains an estimate of the plans of each known actor. The consumers formulate tasks themselves, one for each opponent platoon. Typically, the consumers never have sufficient information about the actors and would like to know more about each one of them. Still, for the network (system) to perform efficiently, with its limited sensing resources, the consumers need to prioritise their tasks.

Here, we try to model the prioritisation of a consumer that assigns priorities to its tasks. The consumer is not primarily interested in a precise estimation of plan distributions. Instead it should favour information that has the most relevance to its mission. A consumer has, for instance, little use of knowing the plans of some hostile actor precisely if that actor only has little impact on the consumer's mission. We decompose this mission-related prioritisation into three elements: (degree of) *hostility*, *time-separation* and *impact*. The first one concerns to what degree the actor's plan is hostile towards the consumer, the second to what degree the actor is separated in time from the consumer (all other properties equal, closer actors should have higher priority), and the third concerns to what degree the actor can cause harm to the consumer's mission.

One way to try to capture this is to use *fuzzy set theory*, where the membership of an element to a set is not a binary condition (in or not in). We tentatively propose a "high threat" fuzzy set, HT, expressing the membership degree of an actor state $\mathbf{x}_t^{(i)}$ to the fuzzy set. The fuzzy set HT is now a conjunction of the three parts, i.e.,

$$\texttt{HT} = \left(\texttt{Host}_\texttt{c} \cup \texttt{Host}_\texttt{p}\right) \cap \texttt{STimeS} \cap \texttt{GI}, \tag{4.11}$$

where $\texttt{Host}_\texttt{p}$ is the "hostile platoon" fuzzy set and $\texttt{Host}_\texttt{c}$ is the "hostile company" fuzzy set. The underpinning explanation of the disjunction of the two hostility degrees is that the hostility of a platoon should not be less than the hostility inferred on the superordinate company. $\texttt{Host}_\texttt{p}$ and $\texttt{Host}_\texttt{c}$ are calculated as normalised and weighted linear combinations of the associated plan distributions (here, denoted $\pi(\mathbf{x}_t^{(i)})$), making the membership degree one when the probability of the plan alternative with the highest weight is one, i.e.,

$$\texttt{Host}_\texttt{p}(\mathbf{x}_t^{(i)}) = \frac{\mathbf{w}_p^T \ \pi(\mathbf{x}_t^{(i)})}{\max(\mathbf{w}_p)},$$

where $\mathbf{w}_p$ is a column vector of weights for platoon plan alternatives. The calculation of $\texttt{Host}_\texttt{c}$ is analogous except for the change of weights.

STimeS expresses the degree to which the separation in time between the consumer and actor is small. This value is based on a function that calculates the actor's least expensive (in terms of traversability) route from its current position to the consumer (also discussed in Section 4.5).

Finally, the GI fuzzy set expresses to what degree the actor can have a great impact on the consumer's mission. In this work, we do not distinguish between the impact of the hostile actors and always use $\texttt{GI}(\mathbf{x}_t^{(i)}) = 1$.

Using the standard fuzzy set operators (see, e.g., Klir & Yuan, 1998, pp. 3), Equation 4.11 mathematically conforms to

$$\texttt{HT} = \min\left(\max\left(\texttt{Host}_\texttt{c}, \texttt{Host}_\texttt{p}\right), \texttt{STimeS}, \texttt{GI}\right).$$

A calculation of HT is based on a single sample $\mathbf{x}_t^{(i)}$ from the particle set of the corresponding actor.[6] What we also want to do is to capture the statistical properties (i.e., expected value and standard deviation) of the HT membership degree given the state uncertainty expressed by the particle set.

[6] Or actually, a single sample from all the platoon actors in the same company.

Calculating the HT membership degree for each of the particles of the state uncertainty representation of an actor is computationally costly (as the number of particles is typically high). To alleviate this problem, we perform a Monte-Carlo simulation estimation of the expected membership degree of HT, $\mu_{\mathtt{HT}}$, and its standard deviation, $\sigma_{\mathtt{HT}}$, by drawing M (typically much less than N) samples from the particle set, $\{\mathbf{x}_t^{(j)}\}_{j=1}^{M}$. The calculations are then the following basic estimates

$$\hat{\mu}_{\mathtt{HT}} = \frac{\sum_{j=1}^{M} \mathtt{HT}(\mathbf{x}_t^{(j)})}{M}, \quad \hat{\sigma}_{\mathtt{HT}}^2 = \frac{\sum_{j=1}^{M} \left(\mathtt{HT}(\mathbf{x}_t^{(j)}) - \hat{\mu}_{\mathtt{HT}}\right)^2}{M-1}. \tag{4.12}$$

The estimates $\hat{\mu}_{\mathtt{HT}}$ and $\hat{\sigma}_{\mathtt{HT}}$ are calculated by each consumer for each mission-relevant actor, and stored in a task structure, $\tau = (\alpha, \mathit{prio})$, where $\mathit{prio} = (\hat{\mu}_{\mathtt{HT}}, \hat{\sigma}_{\mathtt{HT}})$. In our current implementation, we leave it up to the allocation scheme (described later in this section) to order the tasks by comparison. For the comparison to be fair and make sense, we require that all consumers use the same threat calculation (i.e., Equation 4.11) and statistical estimates (i.e., Equation 4.12).

The task management performed for each consumer can be summarised in Algorithm 4.2.

```
Algorithm 4.2: Task management
(1)   foreach actor $\alpha$
(2)      calculate $\hat{\mu}_{\mathtt{HT}}$ and $\hat{\sigma}_{\mathtt{HT}}$ according to Equation 4.12
(3)      $\mathit{prio} \leftarrow (\hat{\mu}_{\mathtt{HT}}, \hat{\sigma}_{\mathtt{HT}})$
(4)      if there already exists a $\tau$ s.t. $\tau.\alpha == \alpha$
(5)         $\tau.\mathit{prio} \leftarrow \mathit{prio}$
(6)      else
(7)         create a new task $\tau' = (\alpha, \mathit{prio})$
```

The only line in the algorithm that requires an explanation is line 4. It checks whether there is already a task τ concerning actor α. If so, line 5, updates its priority (i.e., the priority of a task is allowed to change over time).

Allocation Scheme Implementation

The allocation scheme implementation (Algorithm 4.3) maintains a set of prioritised tasks **T** (possibly updated as described in the task management part above), references (and means to contact) to the services **S**, and connections between tasks and services, i.e., allocations **Allocs**.

Some of the tasks concern the same hostile actor, this is because several consumers may be interested in information about the same actor. In this implementation, the allocation scheme merely considers the maximum value over all tasks that concern the same actor.

Algorithm 4.3: Allocation scheme

```
(1)   $\mathbf{T}_t \leftarrow$ sort tasks according to preference relation $PR$
(2)   **foreach** sorted task $\tau_t$ in $\mathbf{T}_t$ at time $t$
(3)       $cur\_alloc \leftarrow$ get_current_alloc($\tau_t$)
(4)       $best\_alloc \leftarrow$ get_best_alloc($\tau_t, \mathbf{S}$)
(5)       **if** $curr\_alloc == best\_alloc$
(6)           continue with next task in $\mathbf{T}_t$
(7)       **while** $best\_alloc$ is not empty
(8)           $s \leftarrow best\_alloc.service$
(9)           $cur\_alloc' \leftarrow$ get_current_alloc($s$)
(10)          **if** ($s$ not occupied) or ($\tau_t$ is preferred to $cur\_alloc'.task$ according to $PR$)
(11)              **if** $s$ occupied
(12)                  remove previous allocation for $s$
(13)              **if** $cur\_alloc$ is not empty
(14)                  remove $cur\_alloc$
(15)              add new allocation $best\_alloc$
(16)              **break**
(17)          $\mathbf{S} \leftarrow \mathbf{S} \setminus \{s\}$
(18)          $best\_alloc \leftarrow$ get_best_alloc($\tau_t, \mathbf{S}$)
```

Line 1, in the algorithm, orders the presented tasks according to the selected preference relation (which will be described below in this section). The task with the highest priority will be served first. Line 3 finds the current allocation of τ if there is one (thus, we allow for a task to change to a more beneficial service). Line 4 finds the best (most beneficial) allocation for τ_t, i.e., the allocation that gets the best payoff based on a calculation of utility and cost.[7] If the cost for the service is too high, the service might reject τ. Line 5 checks whether τ is already connected to its most preferred service. If so, it continues with the next task.

Line 10 finds the current allocation *cur_alloc′* of the service s in *best_alloc* if it exists. If s is occupied in another allocation and if τ_t has a lower priority than the task of s's current allocation *cur_alloc′*, the program continues on line 21 where the next best allocation is found (if any). Otherwise *best_alloc* is employed. Line 13 realizes the preemption property of the algorithm, i.e., that an allocation may be removed if there is a task that is in more need of a service than the one currently allocated.

We previously explained that the priority stored for a task is actually the tuple $(\hat{\mu}_{\mathtt{HT}}, \hat{\sigma}_{\mathtt{HT}})$. Optionally, we could have combined these two into a summarised value by a weighted sum in the task management function. Here, instead, we allow the network to decide what is more important, expected threat or standard deviation.

[7] The utility is based on the quality of the expected observation and the time it is anticipated to take before the observation can be made. The cost is based on the cost of initiating and running the sensor (e.g., in terms of fuel to transport a UAV).

To do so, we tentatively introduce two preference relations PR_μ and PR_σ, that represent both desires, respectively.

PR_σ is constructed to prefer tasks with a high associated threat variance, i.e., tasks for which the sensors can improve the predictive situation awareness by lowering the threat variance through observations.

Formally, PR_σ is defined to prefer a task τ_1 to another τ_2 (denoted $\tau_1 PR_\sigma \tau_2$), if $\tau_1.prio.\sigma - \tau_2.prio.\sigma > \delta$. If $0 \leq \tau_1.prio.\sigma - \tau_2.prio.\sigma \leq \delta$, $\tau_1 PR_\sigma \tau_2$ only if $\tau_1.prio.\mu > \tau_2.prio.\mu$. PR_μ is defined analogously.

PR_σ, which orders tasks according to the size of the threat standard deviation, appears to be the most appropriate choice from a pure information acquisition perspective. The reason is that the information acquisition can improve the predictive situation awareness by decreasing the threat variance of actors, but makes no improvement by confirming known threats.

Service Management and Resource Deployment

The resource deployment part of our implementation performs a simple path planning for the UAV sensors and sends them on their way. The path planning, as designed, reuses the particle set approximation of the state uncertainty for an actor by applying the state transition algorithm (Section 4.5) to predict the configuration of the particle set when the UAV is likely to be able to make an observation. A path for the UAV is then constructed by drawing path nodes from the predicted particle set. This idea is depicted in Figure 4.7. This approach is a heuristic necessary to realise the simulation. We do not make an attempt to evaluate its efficiency.

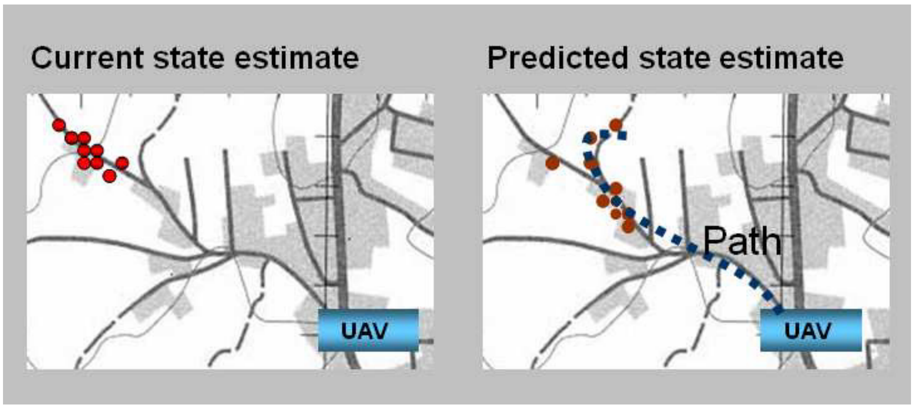


Figure 4.7: The picture on the left depicts the state uncertainty of an actor when a UAV is assigned the task to observe the actor. The picture on the right shows the future uncertainty prediction and flight path (blue squares) planned given the prediction. (By courtesy of Robert Suzić.)

It is the responsibility of the service management to check whether services have completed or reached the end of its path and if so make the service available (even to tasks with low priority).

Note that, although not implemented here, some of the responsibilities of service management and resource deployment (such as path planning) could be deferred to the resources themselves.

4.8 Experiments

Evaluating the proposed framework and implementation of the scenario described in Section 4.3 is difficult considering its simulation complexity (i.e., it involves a multitude of parameters concerning plan recognition, information acquisition, and their interconnection) and the intricate interplay between plan recognition and information acquisition. The proposed problem space is also uncommon and there is little to compare to in the literature. Hence, the two experiments shown here aim at testing the appropriateness of LSIAf and design choices (such as the particle filter uncertainty representation), and exposing pertinent problems and issues.

The first experiment visualises the plan recognition estimate of the company plan alternative `attack` for company `cs`. The simulation in question is the scenario and sensor composition depicted in Figure 4.4. The second experiment shows the result of the same scenario, but with a different sensor composition. The second experiment gives an example of different sensing actions depending on the choice of preference relation (Section 4.7).

Experiment 4.1: Attack Probability Estimation

Figure 4.8 shows a comparison between the best possible (`bp`) estimates of the `attack` probability (probably the most interesting plan alternative for a decision-maker) and two of our robust plan recognition estimates. The `bp` estimate is achieved given continual and accurate observations of all actors and is shown for comparison. The first robust plan recognition estimate represents the average (`av`) attack plan estimate from the set of posteriors (Equation 4.7). Here, we show uncertainty intervals (the green bars) of the posteriors as well. The intervals are $[\min \mathbf{\Pi}_t(\texttt{attack}), \max \mathbf{\Pi}_t(\texttt{attack})]$, where $\mathbf{\Pi}(h) = \{\pi(h) | \pi \in \mathbf{\Pi}\}$. We also show the maximum entropy estimate (`me`) calculated using Equation 4.8 (dash and dot in the figure). Both estimates are dependent on the implemented information acquisition, described in Section 4.7, which uses the PR_μ task preference relation.

In Figure 4.8, initially `cs` is observed by both UAVs and ground observers and the estimated attack probability is close to `bp` and the uncertainty interval is small. The increasing attack probability attracts the interest of the sensors and the uncertainty interval is kept relatively small. By the end of the scenario, near time step 80, the attack probability has decreased (because `cs` is moving away from the consumers) and the interest of the sensors declines (due to decreased threat) resulting in fewer observations of `cs`. This, together with the bias of the state transition

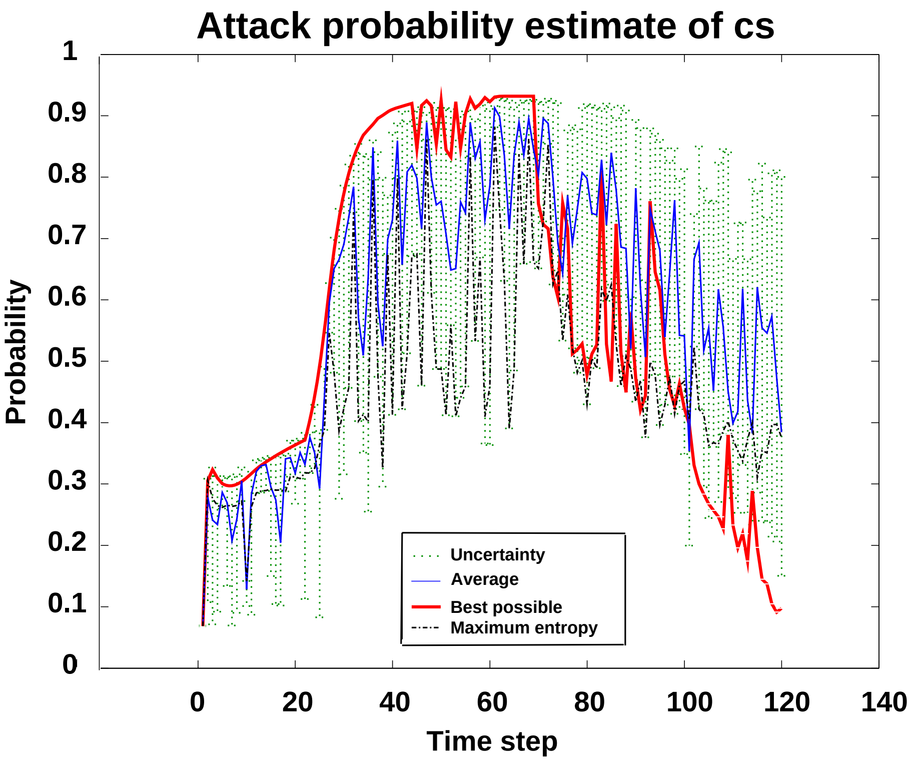


Figure 4.8: Attack probability estimate for actor `cs` for predictive situation awareness.

model (in Algorithm 4.1 which pulls particles towards the consumers), explains why the uncertainty interval fails to cover the `bp` estimate in the last time steps.

For this experiment, the `av` rather appears to better approximate `bp` than `me`. The `av` approximation considers the whole set of distributions unlike the `me` estimate.

Experiment 4.2: Threat-Driven Information Acquisition

The second experiment is run from scenario time step 130 to 160. The purpose is to give an example of the difference between the two task preference relations in Section 4.7, PR_μ and PR_σ. The sensor composition, which can be seen in Figure 4.9(a), differs from that in the first experiment. The target tracking uses 40 particles for each target (i.e., $N = 40$ in Equation 4.1) and 20 representative samples are drawn which are used both for plan recognition and threat estimation (i.e., L and M on page 87 and in Equation 4.12, respectively, are both 20). In

Experiment 4.2, the plan recognition uses the average probability distribution (in Equation 4.7) to summarise the set of plan distributions.

Figure 4.9(b) shows how the best possible threat estimates (given perfect information about the actors' states) for the three companies. The value shown for a company is the maximum threat value of all of its three platoons. The threat level of company `cs` is constant in the experiment since `cs` is stationary between time steps 130 and 160. A noteworthy event happens around time step 145 when the threat levels of companies `cn1` and `cn2` surpass the one of company `cs`.

Figure 4.10(a) shows the mean threat and variance during a simulation using task preference relation PR_σ. Figure 4.10(b) shows how the target allocation to the UAV sensor changes during the time interval. If the UAV is tracking company `cs`, say, it means that it is looking for one of the platoons in company `cs`. In the beginning of the simulation, the UAV switches between the companies `cn1` and `cn2`. At the same time, company `cs` is not being observed and its threat standard deviation therefore remains high.

Figure 4.11(a) shows the mean threat and variance during a simulation using task preference relation PR_μ. In this case, the UAV sensor initially prefers company `cs` due to its higher mean threat. By the end of the simulation, the expected threat of companies `cn1` and `cn2` close in on the expected threat of `cs`. This is consistent with the best possible estimate in Figure 4.9(b), and the result is that the UAV switches between platoons of different companies.

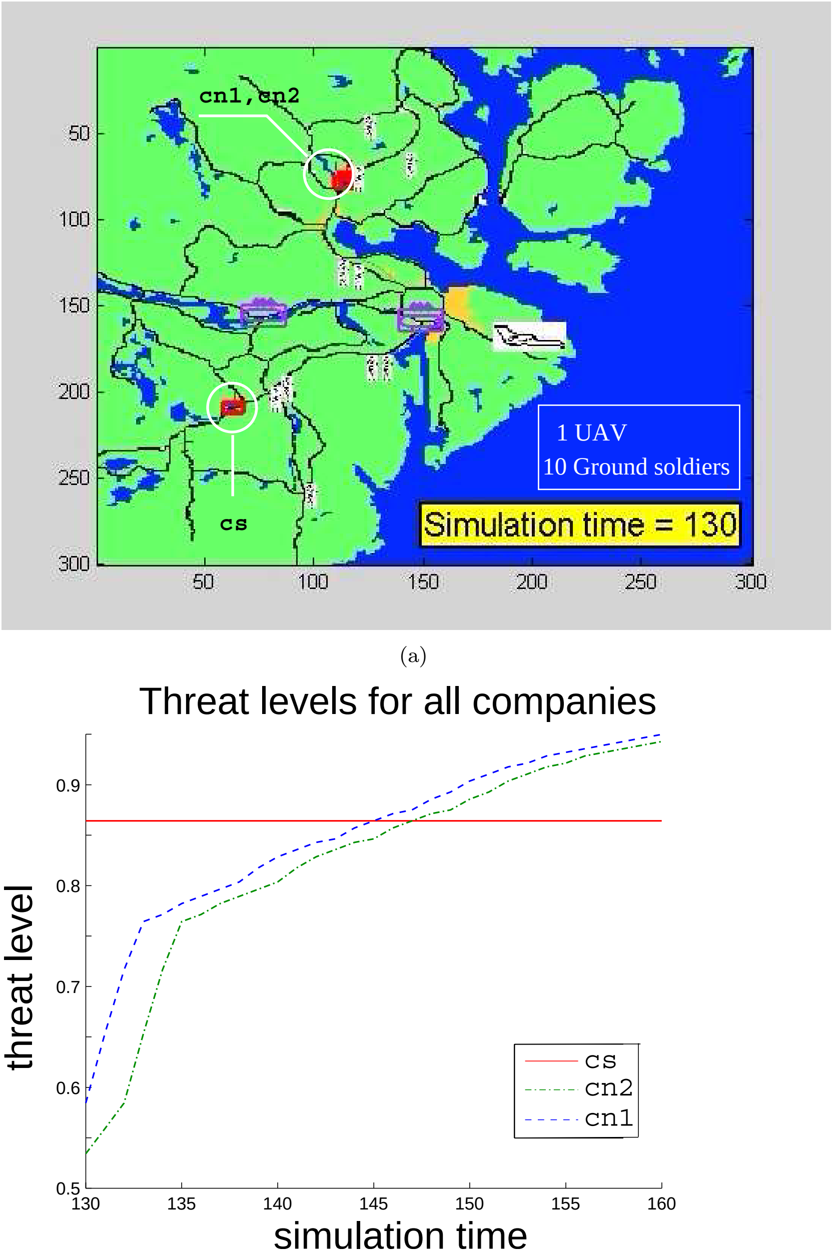
It is difficult to recommend one of the two task preference relations based on this simple comparison alone. One difference between the two simulations, though, is that the use of PR_μ causes an underestimation of the threat of `cn1`

4.9 Summary and Discussion

The large-scale information acquisition framework (LSIAf) presented in this chapter is the result of our recent research (Johansson & Suzić, 2005; Suzić & Johansson, 2004; Johansson & Suzić, 2004). The structure of the framework is designed based on the findings in Chapter 3 regarding the properties of LSIA.

Key aspects of the framework, including multiple objectives for high-level information (e.g., agent plans), heterogeneous sensing resources, long-term sensing actions, sensor preemption, have been implemented and evaluated in simulation experiments. Pertinent issues that were not considered for the current implementation include decentralised control and management of task and service dependencies. The framework should be further tested and discussed and will probably be subject to revision in the future.

In the following sections, we discuss some of the issues concerning the framework implementation and experiments in more detail.



(a)

(b)

Figure 4.9: a) The initial environment state in the second experiment. b) The best possible threat estimate (achievable if all the actors' states are known exactly.)

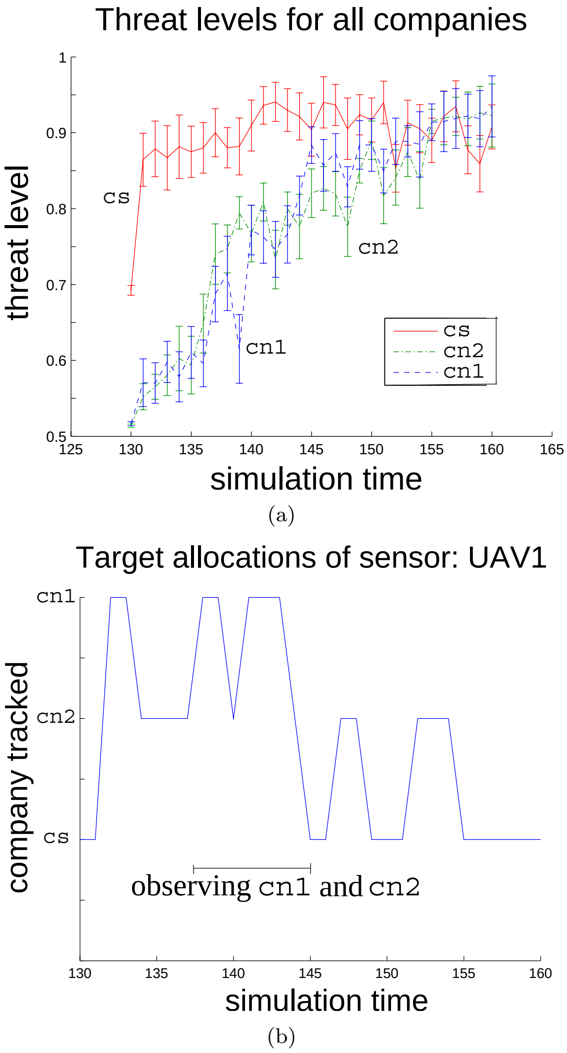
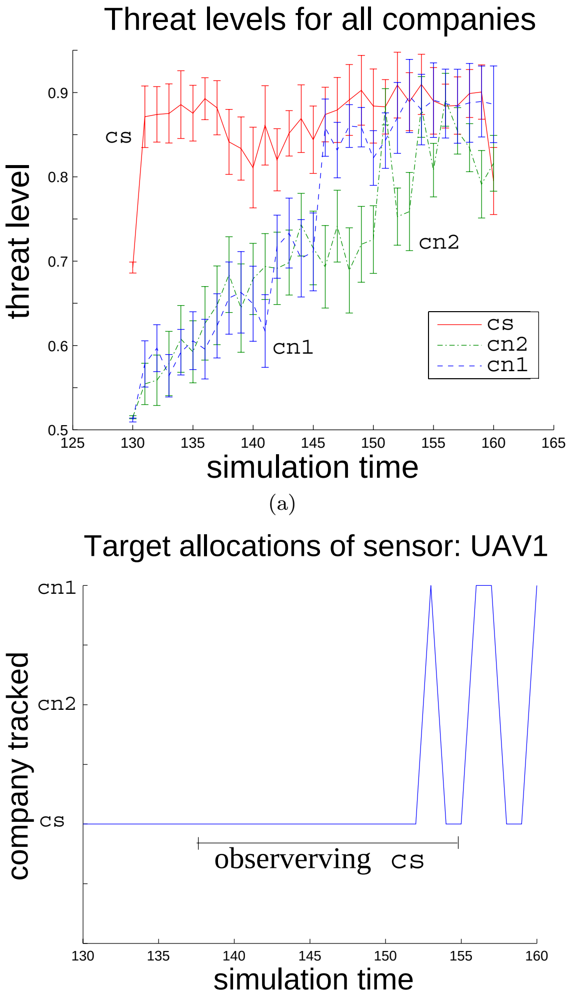


Figure 4.10: a) Mean threat estimates and standard deviation b) Task allocation using task preference relation PR_σ

The Particle filter

For the experimental implementation presented in this chapter, we connected plan recognition to information acquisition in terms of the LSIAf. The (reasonable) assumption that sensing resources are limited and have to be managed and, hence,



(a)

(b)

Figure 4.11: a) Mean threat estimates and standard deviation b) Task allocation using task preference relation PR_μ

that observations are not made continually, encouraged us to integrate particle filters for representation of state uncertainty. The particle filter proved to be useful for several reasons: the state uncertainty representation is expressive (e.g., compared to a Kalman filter); the state estimate is robust to spurious observations; and the particles are a convenient representation of a probability distribution when applying Monte-Carlo simulations.

Since observations of the actors are sparse in our simulation, we had to make two unusual modifications to a standard particle filter algorithm: (i) the transition model of the particle filter is biased to draw particles towards the consumers (possibly overestimating the progress of the actors); and (ii) the particle filter is reinitialised whenever an observation is too far (further than some distance threshold) away from the the closest particle.

There are also a few challenges related to the use of particle filters in our approach. For instance, the interplay between the plan recognition process and the dynamics of the particle filter is delicate. The consequences can be seen in the right part of the graph in Figure 4.8. There the uncertainty of the attack probability of company `cs` is great although some observations are made. The result is partly a consequence of particles being attracted by the consumers, since the inference in the dynamic Bayesian network is sensitive to changes in direction.

Robust Plan Recognition

We have been tempted to apply the robust Bayesian methodology to the set of plan distributions $\mathbf{\Pi}$ acquired in Section 4.6 to establish robust plan recognition. In robust Bayesian analysis (David Rios Insua, 2000), a single prior distribution is replaced by a set of plausible priors resulting in a set of posteriors. Such an approach is also called *global* robust Bayesian analysis. To make it manageable, we would have to replace our set of distributions with a convex set of probability distribution parameters, but we have yet to determine if our set of distributions $\mathbf{\Pi}$ is a valid basis for such a parameter set. Even if we could make a valid transformation from $\mathbf{\Pi}$ to a convex set, the research on generalised Bayesian networks, so called *credal* networks (Cozman, 2005), is still too immature to be applied to our current research.

Optimal Information Acquisition for Information Fusion

One of our objectives in this chapter is to represent and conduct information acquisition for high-level fusion, i.e., information fusion. Our discussion in Section 4.7 lead to the important conclusion that there is an inherent difficulty in information acquisition for information fusion. One reason is simply that a fusion process might be computationally costly and to estimate the utilities of sensing actions the fusion process has to reused over and over again. Hence, approximative solutions are often necessary.

The approach we present in Section 4.7 is a greedy approximation which iteratively constructs a service configuration space selection, $\mathbf{c}$. Instead of trying to evaluate the expected effect of a service configuration on the environment representation (i.e., $u_{\mathrm{SOC}}(\mathbf{c})$ for all $\mathbf{c} = (c_i)_i$), which is very time consuming, we assign priorities to information acquisition tasks. Next, we order the tasks according to their priorities (i.e., threat estimates which we assume are apart of the system's environment representation) using one of the preference relations (PR_μ or PR_σ).

Assume that we have a task τ_j which has the highest priority according to one of the preference relations. Then the allocation scheme (Algorithm 4.3) allocates a most suitable service s_i to it (unless none is feasible). This allocation corresponds to selecting $c_i = \tau_j$ for the current service configuration $\mathbf{c} = (c_i)_i$.

Finally, what then are the properties of the imagined u_{SOC} (or $\succeq_{\text{SOC}}$[8]) that is approximately optimised? When using PR_σ, e.g., we imagine a $\succeq_{\text{SOC}}$ that prefers environment representations with small threat uncertainties for actors. The rationale is that a service can probably make the greatest difference by observing the actors with the highest threat standard deviation, Conversely, if the threat standard deviation for an actor is low or negligible, we cannot hope to improve it and should not waste resources on it.

Additional Challenges

We end the chapter with a list of a few questions that were raised during the implementation of the framework.

- What is the cost of ignoring an actor? After a while (without observations), the state uncertainty of an actor might become very large, and soon also the priority of the actor might increase. At that point it might be difficult (not to mention costly) to find the actor again.

- Some issues concern processing and information distribution in networks. Where should observations be sent and where should they be fused? Moreover, where (in the network) should computations be performed? It seems reasonable that the node that requests information makes sure that the characteristics of the information it needs is known to the observer nodes (e.g., in a simple case a relevant characteristic could be the location of observed data). To counter these problems, plenty can be learned from the distributed processing and sensor network communities.

- Long-term (or farsighted) sensing actions induces additional difficulties as the estimated observations from a service has to be based on not the current environment state estimate but a predicted future one. We address this issue to some extent when we use the particle filter to predict future states for UAV path planning (Figure 4.7). Hence, we predict future states for resource management after selecting a service configuration. However, predicted future states should also be integrated into the process of selecting a service configuration.

- It is plausible that the activation of a service in the LSIAf causes a request for information necessary to prepare or perform the service. Is the LSIAf

[8] The preference relation over service configurations for a system objectives control, $\succeq_{\text{SOC}}$, was introduced in Section 2.4.

expressive enough to efficiently encompass the recursive situation where a service might be acting as a member of the task origin space?

- Deployment costs (e.g., fuel costs, time to observation, etc) might be tremendously difficult to estimate in advance. Resources needed to make an estimation might not even be available.

Part III

Issues in Large-Scale Information Acqusition

Chapter 5

Decentralised Control of a Mobile Sensor System

In this chapter, we focus on one of properties of large-scale information acquisition (LSIA), *decentralised control*. As an example, we consider decentralised control for a mobile sensor system.[1] A mobile sensor system consists of a team of mobile platforms (for the sake of the following discussion, we could also refer to them as computational agents) equipped with various sensors. A mobile sensor system could be used for different kinds of sensing tasks, e.g., search and rescue operations (Jennings et al., 1997), and intruder detection and target tracking applications (Pirjanian & Matarić, 2000). In this chapter, we address the issue of coordinating the actions in a mobile sensor system using *bargaining theory* (Nash, 1950).

The mobile sensor system can be thought of as one composite sensor, and in that sense it has some powerful properties. One property is *mobility* which significantly extends the sensing range of the composite sensor. Another property is *distributed components* which allows the mobile sensor system to behave amorphously and, hence, be more applicable to time-varying tasks. The mobile sensor system can, e.g., split up in subsets performing subtasks in parallel, or observe objects from various and changing angles.

Due to their sensing capabilities and the aforementioned properties, mobile sensor systems have a natural role to play in information fusion. It is an inherent property of information fusion that, unlike most robotics research, the observed environment tends to be highly responsive to the mobile sensor system's actions and possibly deliberatively antagonistic or even hostile.

Information fusion applications are often assumed to rely on a large-scale system of sensors. In Chapter 3, we present some desirable properties of the necessary *large-scale information acquisition* (LSIA) for general applications. Those properties

[1] A synonym would be multi-robot system, but since this paper focus on sensing mobile sensor system appears to be more appropriate.

include the ability to deal with *multiple and time-varying objectives*, *decentralised control*, and *long-term and continual planning* for sensor resources.

Out of the different aspects of LSIA, we focus on decentralised control in this chapter. A system fulfils the requirements of decentralised control if it does not have a single (bottleneck) agent controlling the whole distributed system. Decentralised control is an issue that has been studied in, e.g., the symbiotic fields of mobile robotics and agent theory.

In this chapter, we propose a *coordination protocol* based on *recurrent negotiations* to realise decentralised control. The purpose of the coordination protocol is to prescribe how agents benevolently should coordinate their actions to handle conflicts. Although, in this chapter, conflicts mostly concern actions, other types of conflicts are conceivable (e.g., goal disparities in planning, constraints in resource allocation, and task inconsistencies, Jennings et al., 1998, pp. 290). The protocol dictates pairs of agents to recurrently engage in bilateral negotiations to select joint actions through a mechanism based on bargaining theory (a sub-discipline of game theory in economics). Computational and communication aspects are discussed and the protocol is compared to a global centralised and a non-coordinated solution in an intruder detection scenario.

Bargaining theory offers a number of alternative solutions. We here critically discuss the most famous one, the Nash bargaining solution (NBS), from the perspective of coordination of benevolent agents. We identify shortcomings of the NBS and show that it and the product maximising mechanism (which is derived from the NBS) may, for some problems, be challenged by other bargaining solutions. A comparison is made with the so called *egalitarian solution.*

Section 5.1 mentions some prominent literature in the related fields of research, i.e., information fusion, agent theory, game theory, and mobile robotics. Section 5.2 characterises the coordination problem we are addressing. Section 5.3 describes our recurrent negotiation coordination protocol. The coordination protocol is, furthermore, depending on a negotiation solution specified in bargaining theory. Two solutions from bargaining theory are presented and analysed in Section 5.4. Section 5.5 discusses the proposed coordination protocol from the perspective of evaluation criteria suggested in the multi-agent literature. Section 5.6 offers the results of two simulation experiments involving a set of sensor-equipped agents exploring a shared environment. The first compares an egalitarian implementation of the coordination protocol to a fully coordinated (centralised) solution and a solution with independent (non-coordinated) agents. The second experiment compares the two bargaining solutions for a specific problem. Finally, Section 5.8 summarises and concludes the chapter.

5.1 Related Work

Whereas the literature on sensor management in information fusion is extensive (see surveys such as those by Xiong & Svensson, 2002; Ng & Ng, 2000), surprisingly little attention has been given to decentralised sensor control which is required to facilitate LSIA in its most expressive form. It has been suggested that the reason for this is to a large extent academic; that general features of problems, such as upper and lower efficiency bounds on algorithms, are most appropriately studied in a centralised rather than in a decentralised context. Notable recent exceptions, though, are the efforts by Dodin and Nimier (2001) where multiple sensors share the task of tracking multiple targets and sharing observations, and the *active sensor network* (Makarenko & Durrant-Whyte, 2004). Unlike those efforts, where coordination is indirectly achieved through shared and fused information, we are more interested in explictly representing actions and negotiating about joint actions.

The interdisciplinary concept of coordination is throughly discussed by Malone and Crowston (1994, pp. 90) who succinctly define coordination as *the management of dependencies between activities*. Their comprehensive study on "coordination theory" spans results from computer science, economics, organisation theory, as well as biology. In our case, *activities* are represented by the individual agents and *dependencies* are represented by the effect individual agents' actions have on other agents.

In computer science, the decentralised control aspect and agent coordination have been given much attention in a sub-discipline known as distributed artificial intelligence (DAI). Coordination in DAI has been defined as the state or process of a set of agents for which the action selection of an individual agent fits well (in the sense of, e.g., sharing common resources and avoiding deadlocks) with the actions of the others (Weiss, 1999, pp. 589). An overview and classification of coordination techniques for software agents is provided by Nwana et al. (1996) who list some reasons for coordination: preventing chaos (i.e., when the un-coordinated interactions between agents may have detrimental outcomes); meeting system constraints (the agents may share a resource such as electricity or bandwidth); heterogeneous capabilities (i.e., the agents capabilities might have to be orchestrated to achieve a system goal); dependencies between actions (i.e., the action of one agent might be a necessary precondition for another agent to perform an action); and, efficiency (e.g., information acquired by one agent might, if shared, improve the performance of some other agent). Our interest in coordination, in this chapter, lies primarily with efficiency, e.g., by avoiding redundant work.

The field of DAI generally concerns the study and construction of interacting agents and can be decomposed into two subfields: one dealing with *cooperative* multi-agent systems (Coop-MAS) and the other with *competitive* (Comp-MAS).[2] Although both fields adhere to the same agent definition (basically, that agents are independent computational entities capable of perceiving and acting), they assume different contexts and analysis criteria. The former is assumed to have a single

[2] In the classical DAI terminology, the two subfields are called *distributed problem solving* and (simply) *multi-agent systems*, respectively. However, it might appear inconvenient to reserve *multi-agent systems* for competitive agents only. Hence, we stick to the terminology used by, e.g., Leyton-Brown (2003).

designer, who has the opportunity to furnish all its agents with behaviours and strategies. Evaluation of a Coop-MAS system focuses on how well it fulfils the goals elicited by its designer. The latter field assumes that agents have different designers with varying objectives. For Comp-MAS, evaluation concerns, e.g., the willingness of agents to participate in the negotiation. Interestingly, the distinction between these two is mainly in the evaluation; in applications, ideas from both fields may blend (Kraus, 2001a, pp. 3). In particular, negotiations from Comp-MAS research can be used to resolve conflicts in Coop-MAS, as in the case in this chapter.

Coop-MAS is discussed by, e.g., Durfee et al. (1989). An overview of research on negotiation for Comp-MAS is provided by Sandholm (1999) and more detailed descriptions are presented by Kraus (2001b).

Social laws (Shoham & Tennenholtz, 1995) are one technique within Coop-MAS to achieve coordination. The idea behind social laws is basically to beforehand specify context dependent restrictions on the action space of individual agents in order to prevent certain, otherwise plausible, predicaments (e.g., collisions and deadlocks). Typical examples of social laws include traffic rules such as stopping at a red light in an intersection and consistently sticking to the right (or left) side of the street. A set of pre-specified social laws may simplify planning and coordination of the agents. Coordination through social laws is typically communication-less and efficient as long as all agents interpret the situation in the same way. In the kind of inaccessible environments that we consider here, that is a strong assumption.

Robotics, and multi-robot systems in particular, is in theory a natural application domain for DAI techniques. However, the integration of DAI techniques and multi-robot systems has so far been limited, partly due to the severe sensing, communications and manipulation restrictions of state-of-the-art robotics (in DAI, which frequently concerns agents in software environments, such restrictions are often neglected).

Interestingly, coordination in multi-robot systems tends to have slightly different characteristics than in agent theory.[3] Coordination in mobile robotics is mostly cooperative (a notable example is the RoboCup robotic soccer competition with teams of cooperating soccer robots, Asada et al., 1999), and the research has to take account of issues such as dynamic environments and physical constraints. As a contrast, research in "pure" agent theory, with the advent of the Internet, tend to study more competitive agents in computational environments.

A comprehensive survey on cooperative interaction in mobile robotics is provided by Cao et al. (1997). Issues and definitions are also discussed in an early article by Matarić (1995). A more recent effort was made by Farinelli et al. (2003) who provide a coordination taxonomy for multi-robot systems. According to their taxonomy, our effort in this chapter belongs to the *strongly coordinated and distributed*

[3] Also note that there are different definitions of coordination (sometimes slightly conflicting) appearing in the robotics literature. Here, we stick to the definition used in DAI.

class.[4] An example of an effort within robotics that is recognised as belonging to this class is the ALLIANCE architecture by Parker (1998). ALLIANCE, along with several other multi-robot architectures, strongly addresses the many physical issues that comes with the mobile robotics problem domain: real-time performance, motion planning and collision avoidance, communication protocols (not to be confused with coordination protocols), estimation of world state, detection of the result of one's own actions, and estimation of internal state (e.g., battery power, position, etc.). ALLIANCE is, furthermore, tailored to make sure that the tasks that a team of robots are performing are completed even if some robots become dysfunctional. Additionally, ALLIANCE is behaviour-based and it does not support direct negotiation. Instead, robot actions are based on the current assumed task, environmental conditions, the state of other robots (state information is assumed to be explictly communicated rather than sensed), and the state of the robot itself. Another approach is the Active Sensor Network architecture presented by Makarenko and Durrant-Whyte (2004) for which both negotiation-less and negotiation-based coordination have been implemented.

Finding a joint action for a set of autonomous agents can generally be seen as a *multi-objective optimisation problem*, where each agent represents an objective. This is a problem that appears in many parts of science, and it has been studied with assumptions about, e.g., objective constraints and priorities. One approach to the problem, not considered here, is to use *evolutionary algorithms* (Coello, 1999). The evolutionary approach is most appropriate for large decision spaces. In our case, the aim is to keep the decision space as small as possible. The evolutionary approach is, furthermore, stochastic in nature; a property which complicates the coordination of negotiating agents. The theory of multi-objective optimisation problem has been applied to action selection of a single agent, where conflicts arise among the different objectives of the agent (Pirjanian & Matarić, 2000).

5.2 The Coordination Problem

What we consider here is a set of *benevolent* agents. They are benevolent in the sense that they are willing to share information truthfully with each other and to consider the preferences of other agents concerning shared resources.[5] Even though the agents are benevolent, it is inherent that actions of one agent may affect the performance of one or more other agents. For instance, the agents might be sharing some resource such as physical space (in robot motion planning) or a peripheral device (in a computer network). Hence, some coordination protocol to promote efficient results is necessary.

[4] The mobile sensor systems we are considering is strongly coordinated because it has an explicit coordination protocol. It is, furthermore, distributed (or decentralised) because (as previously declared) there is no single control agent that makes all the decisions for all agents.

[5] A distinct definition of benevolent agent does not yet appear to exist (Mohamed, 2000, pp. 9).

In the rest of this section, we will describe the properties of the agent environment that we consider and their consequences. The environment is assumed to be *physical* and *antagonistic*. By physical we mean that it is both *dynamic* and *inaccessible*, in the taxonomy of Russell and Norvig (1995) which we presented in Section 2.1. Dynamic environments are such that they evolve with or without agent actions. By antagonistic, we mean that the environment can not only respond intelligently to the actions of the mobile sensor systems, but it may also try to act and plan to hinder the completion of the tasks belonging to the mobile sensor systems. These environment characteristics have two fundamental impacts on the the coordination problem: the communication bandwidth and the time for deliberation are both limited.

Communication is limited because of the risk of revealing one's presence to a presumable intruder and because of the risk of congestion. For example, an antagonistic environment capable of performing communications intelligence might easily intercept radio communication and use it against the cooperating team of agents. If instead transceivers based on infrared light are used, the range of communication is limited and heavily dependent on the landscape and vegetation. Time for deliberation is limited since the mobile sensor system should operate under real-time constraints.

In summary, we want to design a coordination protocol that solves conflicts that arise between benevolent agents acting in a common environment.

Given the properties of the problem domain (i.e., that it is physical and antagonistic), we are looking for a coordination protocol that is *simple*, *efficient* and *timely*. We end up with a recurrent multi-objective optimisation problem and propose a coordination protocol to deal with it in the following section.

5.3 A Recurrent Negotiation Protocol

As the agents pursue their individual or common tasks, they will have to coordinate their actions among themselves to efficiently use shared resources.

A coordination protocol can be divided into two levels: an *organisational level* and a *negotiation level*. The organisational level dictates which agents should negotiate and when. Limiting the number of agents in an *encounter*[6] has the advantage that the negotiation problem to solve also becomes limited. To respond to the changing knowledge of the environment state and desires of the agents, they recurrently engage in negotiations during the lifespan of their tasks. The negotiation level describes the actual bargaining mechanism that the agents apply in each encounter to reach a unanimous deal, i.e., joint action.

[6] The general term for the event when agents interact and consider interrelations (negotiation is an example of such an event) is called an encounter (Parsons & Wooldridge, 2002). Formally, it is often characterised by a tuple including a set of agents, a set of interaction outcomes and the utility functions of the agents.

The rest of the section describes the properties of a single agent in a team of mobile sensors; describes the bargaining based coordination mechanism for negotiation encounters; and discusses the organisational characteristics in more detail. Section 5.4 then describes two negotiation solution concepts from bargaining theory.

The Individual Agent

An individual agent in the mobile sensor system has a task (which might be shared or individual) and acts autonomously to complete it.

In the experiments in Section 5.6, the agents share the same task, to locate a presumed intruder in some physical environment. Agent actions involve moving in the environment and making observations. An agent makes its own observations and establishes its own individual understanding of the situation. The individual understanding, or situation picture, then constitutes a basis of desires and preferences of the decision-making of the agent.

Furthermore, an agent may have an opinion about the candidate actions of other agents. Using the coordination protocol described here, it can express its views on the actions proposed by the agents it negotiates with. The coordination protocol allows an agent to, e.g., transfer belief of the usefulness of joint actions or to call for assistance.

In our current implementation, once an agent has coordinated its action with one of the other agents, it will honour its commitment to the coordinated action until it has been completed. Even so, negotiating with others, while being committed to an action, is still fruitful since the agent has the opportunity to affect the action selection of its negotiation counterpart. A relaxation of this assumption would be to consider the consequences of agents abandoning commitments under certain circumstances, but that is beyond the scope of this study. That issue is further discussed by Nguyen and Jennings (2005) and Jennings (1993).

Negotiations

This section prescribes the process of our bargaining encounters. The initiation of an encounter is discussed in the next section. As we explained in Section 5.2, the coordination protocol we wish to design should allow the agents to act efficiently in a dynamic environment.

For the following discussion, we define $\mathcal{A} = \{a_i\}_{i=1}^{|\mathcal{A}|}$ to be a set of agents and D_i to be the set of actions or decisions for agent a_i. A deal δ is the outcome of an agent encounter and is defined by the actions assigned to all involved agents, e.g., $(d_1, \ldots, d_{|\mathcal{A}|})$. As will be explained, we here focus on *bilateral* negotiations (i.e., involving only two agents). We denote the agent which initiates a negotiation a_{n_1} and its counterpart a_{n_2}, where $1 \leq n_1, n_2 \leq |\mathcal{A}|$ and $n_1 \neq n_2$.

We note that the benevolence assumption has some important properties: first, the truthful sharing of information among the agents allows them to find solution deals using a limited amount of communication. In fact, we wish to design the mechanism in such a way that once information about agent state (denoted I) and preference has been shared, both agents can individually find a solution deal $\delta^{\top} = (d_{n_1}, d_{n_2})$ and execute its own action (d_{n_1} or d_{n_2}, respectively) of the deal without any further communication.

Second, the benevolence assumption mitigates at least some of the criticism towards game-theoretic solutions for multi-agent systems[7] raised by Nwana et al. (1996).

Our protocol for a complete agent encounter is described in Algorithm 5.1:

Algorithm 5.1: A negotiation encounter

```
(1)   An agent $a_{n_1}$ initiates communication with another $a_{n_2}$.
(2)   $a_{n_1}$ and $a_{n_2}$ exchange relevant status information $I_{n_1}$ and
      $I_{n_2}$, if necessary.
(3)   $a_{n_1}$ and $a_{n_2}$ calculate their individual utility mappings ($u_{n_1}$ and $u_{n_2}$) over all joint action deals
      $\delta_{n_1 n_2} = (d_{n_1}, d_{n_2}) \in D_{n_1} \times D_{n_2} = \Delta_{n_1 n_2}$.
(4)   $a_{n_1}$ and $a_{n_2}$ exchange utility functions.
(5)   Both agents determine the outcome deal $\delta^{\top}$, by calling
      Bargaining($\Delta_{n_1 n_2}, (u_{n_1}, u_{n_2})$).
(6)   If Bargaining does not yield a unique best deal, $a_{n_1}$ selects one deal $\delta^{\top}$ from the set of best deals $\Delta^{\top}$ and notifies
      $a_{n_2}$ about its selection.
(7)   Both agents implement their individual parts of $\delta^{\top} =$
      $(d_{n_1}, d_{n_2})$.
```

In the first line, agent a_{n_1} decides to coordinate its actions with some other agent a_{n_2} (more on the initiation of a negotiation encounter is explained in the following section). If a_{n_2} is busy (perhaps due to an ongoing negotiation with another agent) it ignores or refuses the connection from a_{n_1}. a_{n_1} may then negotiate with some other agent instead or try to reconnect with a_{n_2} at a later time. If a negotiation can not be initiated, a_{n_1} will have to make an uncoordinated action based on its own preferences alone.

If a_{n_2} is available for communication, status information will be exchanged (line two). The status information should contain relevant properties (feasible actions, agent state, current task or behaviour, requests for assistance) for the agents' evaluation of the consequences of the other agent's actions. For some applications, all necessary information may already be available (e.g., the sets of actions D_i might be fixed and known for all i). In line three, the agents (using the status information of both agents) by themselves calculate their own utilities over all joint deals $\Delta_{n_1 n_2}$. The agents share their utilities in line four. In the fifth line, the agents individually

[7] One part of the criticism concerned that it is unrealistic that agents know each other's valuation of joint actions. However, under the benevolence assumption, we can assume that the agents are truthfully willing to share information about their beliefs and internal states.

calculate the outcome deal $\delta^\top$ using a Bargaining algorithm. In the following sections, we will present some possible bargaining solutions. Finally, in the sixth line, both agents implement their individual part of $\delta^\top$.

In some cases (depending on the utility functions of the agents), the outcome of Bargaining might be ambiguous (the reason for this is further discussed in Section 5.4). An ambiguous solution is a solution set with more than one deal which we can denote $\Delta^\top \subseteq \Delta_{n_1 n_2}$. This happens, e.g., when both agents, for whatever reason, are completely indifferent between all deals and assign the same value to all of them.

If there is more than one solution to the bargaining problem, the agents could, in an ad-hoc fashion, select the first deal in the set. The agents should have calculated the deals in Δ_{12} in the same order. Therefore, they should select the same deal $\delta^\top$ from $\Delta^\top$. There are of course other possibilities: agent a_{n_1} (or agent a_{n_2} for that matter as long as the protocol specifies one of the agents) could select one deal randomly from the equal ones and announce its selection to the other agent. This option obviously has the drawback of the extra time and energy required to exchange a_{n_1}'s selection. Another randomised alternative would be for the initiating agent a_{n_1} to uniformly select a random real value R, i.e., $R \sim U(0, 1)$, and transmit the outcome r together with u_{n_1}, just in case ambiguities arise. If then $\Delta^\top$ contains more than one deal, both agents can deterministically select the $\lceil r \cdot |\Delta^\top| \rceil$-th deal from $\Delta^\top$, without any further communication.

The computational cost for each agent running the algorithm is $O(|\Delta_{12}|K + B)$, where K is the application specific cost for evaluating a deal, and B is the cost of the chosen bargaining mechanism. Examples of bargaining mechanisms and their costs are given in Section 5.4.

Organisational Structure

Organisational structuring aims at making agent cooperation more efficient by defining roles, responsibilities, and communication patterns for agents. For instance, involving agents without conflicting interests in negotiations might be inefficient. Instead, agent roles or rules as premise or precondition for which agents should engage in negotiations can be used. The opportunities with organisational structuring are more thoroughly discussed by, e.g., Durfee (2001, pp. 130) and Durfee et al. (1989, pp. 122).

The organisational level of our coordination protocol dictates *when* a bargaining encounter should commence and *which* agents should be involved. A careful consideration of the organisational structure is important since involving too many agents in an encounter or engaging in bargainings too frequently will allocate too much resources (time and computation) to coordination. Conversely, neglecting the potential advantages of coordination, the agents may repeatedly interfere with each other's activities.

In the previous section, we assumed that agents negotiate in pairs. To motivate this choice of organisational structure, we study the complexity of larger groups of negotiating agents. The set of joint actions $\Delta_{\mathcal{A}'}$ grows exponentially with the number of participating agents ($\mathcal{A}' \subseteq \mathcal{A}$) since $|\Delta_{\mathcal{A}'}| = D_{\max}^{|\mathcal{A}'|}$, where $D_{\max}$ is the maximum number of actions for an agent, i.e., $D_{\max} = \max_i |D_i|$.

To bridle the beast of computational complexity, we suggest that encounters concern only bilateral negotiations. For bilateral negotiations, i.e., with $|\mathcal{A}'| = 2$, the running time for each agent is restricted by a joint action set with maximum size $D_{\max}^2$. Another practical reason to suggest bilateral negotiations is to avoid forcing agents into negotiations when they, and the system as a whole, can not benefit from them.

We have decided how many agents should engage in each negotiation, but we also need to decide which agents should negotiate. In the intruder detection application considered in this chapter, deciding which agents should negotiate is dependent on the physical locations of the agents. For instance, two agents in each other's vicinities might have opinions about where the other one should or should not go. One agent might have opinions about the other's actions because it has detected obstacles, fully explored some part of the region which would be unnecessary for the other one to re-explore, or because some of the other's actions might interfere with its activities. Nearby agents could also request help from each other. Therefore, we suggest that an agent only engages in negotiation encounters with its geographical neighbours.

In other applications, it might be more useful to define the concept of neighbour in more abstract terms. Agents might, for instance, be neighbours due to the possession of similar resources, or conflicting or interacting tasks.

One problem that we do not address here is practical ways of maintaining negotiation neighbours. As the agents move about, agents who initially were neighbours might not be after some time. One way to suppress the problem is to require that agents keep the same negotiation partners throughout the life-time of the application. That idea, however, might yield inefficient results. In the experiments in Section 5.6, we make the assumption that the agents are somehow aware of who their closest neighbours are.

Deciding when to negotiate is a trade-off between degree of coordination and amount of communication. Frequent negotiations will yield a lot of communication, but well coordinated actions. On the other hand, negotiating too seldom will yield uncoordinated actions. In the experiments in Section 5.6, the simulation time is discretised and each agent is negotiating in each new time step.

5.4 Bargaining Theory and Negotiation Solutions

In this section, we will give some background and fundamentals of game theory and focus on a sub-discipline known as bargaining theory. Bargaining theory models agents that are trying to reach mutually beneficial agreements when their joint actions have an impact on the utility of the individual agent. Bargaining theory is well known to the DAI community (see, e.g., Zlotkin & Rosenschein, 1996a;

Sandholm, 1999) and we will discuss and use it to find negotiation solutions for our coordination protocol.

Game theory comprises a diverse collection of tools for analysing the conflicts of interacting agents (Osborne & Rubinstein, 1994).[8] An interdependent decision situation including at least two agents is, for historical reasons, called a *game*. Since the pioneering work of John von Neumann, Oskar Morgenstern and John Nash in the 1940s and 50s, game theory has flourished yielding Nobel prize awards in economics. Apart from economics, conflicts modelled through game theory is also studied in social science, e.g., by Kahan and Rapoport (1984).

For computer scientists, unfortunately, the literature on game theory in economics and sociology is not always immediately applicable as its focus is on other aspects than that of computing. Typically, in economics one is interested in finding some stable monetary exchange or in proving that a given problem (game) has stable solutions. In computer science, on the other hand, the issue of computational complexity is of vital importance. Furthermore, in economics the resulting utility of agents is in focus. In computer science, utilities are often merely a means to express preference and the actual actions corresponding to the utilities are the relevant outcomes.

In spite of the aforementioned properties, game theory, as a tool for modelling situations of conflicting interests, has become relevant for research in DAI. Decentralised solutions are in focus in DAI, motivated partly by the emergence of shared communication networks (such as the Internet, where a multitude of independent users co-exist) and partly by the existence of problems that are distributed in nature. Examples of the latter are cases where the success of a task is dependent on actions performed in spatially separated places or when intractable computational complexity precludes centralised solutions.

The agents of a particular game are assumed to be *rational* in the sense that they try to optimise their own individual gain for joint actions given their knowledge of the preferences of the other agents. In spite of its diversity, common components of games are a set of agents $\mathcal{A}$, a set of actions or decisions for each player $\{D_i\}_{i=1}^{|\mathcal{A}|}$, and a set of utility functions $\{u_i\}_{i=1}^{|\mathcal{A}|}$ over game outcomes. An *outcome* of a game is usually an *action profile* $\left(d_1, \ldots, d_{|\mathcal{A}|}\right)$ with $d_i \in D_i$, i.e., the resulting actions of each agent in the game. We will use the notation presented here frequently throughout the remainder of the chapter.

A famous solution concept from game theory for interacting agents is the *Nash equilibrium* (see, e.g., Osborne & Rubinstein, 1994, pp. 11). A Nash equilibrium solution corresponds to an action profile which is stable in the sense that no agent has an incentive to unilaterally choose another decision as that would result in a lower payoff. In spite of the pleasant stability property of the Nash equilibrium it has some serious deficiencies for our current research. First, for some games no pure Nash Equilibrium exists (whereas for others there are multiple). Second, there might exist other action profiles (non-Nash equilibria) which all agents would

[8] In the game theory literature, agents are mostly called *players*.

prefer (i.e., which dominates the Nash equilibria), as exemplified by the *Prisoners' dilemma* game (Sandholm, 1999, pp. 203). The solution concept of *axiomatic bargaining theory*, on the other hand, does not suffer from these deficiencies.

The following subsections discuss two bargaining solutions and possible interpretations in computer science.

The Nash Bargaining Solution

Axiomatic bargaining theory models the encounter of (at least) two agents who try to find a mutually beneficial solution when the action of one agent affects the outcome of the other. Kalai (1985) calls it "the theory of consensus."

The bargaining problem, as presented in economics, models the situation where the utilities of (originally) two agents, a_1 and a_2, are interdependent. If the agents can coordinate their actions, i.e., if they can strike a *deal* or *agreement* δ on what joint action to select (i.e., $\delta \in D_1 \times D_2$) or what fraction of a shared resource to allocate to each agent (i.e., $\delta \in [0,1] \times [0,1]$), they can avoid unfavourable situations. Consider, e.g., two uncoordinated vacuum cleaning robots moving to the same room, where they might constantly be in each other's way while trying to clean the floor that the other might already have cleaned, hence, significantly decreasing the performance of the multi-robot cleaning team.

A bargaining game B is characterised by a tuple $< u^{\perp}, U >$, where $U \subseteq \mathbb{R}^2$ is a continuous set of joint utilities, i.e., $(u_1, u_2) \in U$, for the agents, and $u^{\perp} = (u_1^{\perp}, u_2^{\perp})$ is called the *disagreement utility* and corresponds to the joint utility of the agents if no agreement is reached. The utility of the resulting deal given a game $B =< u^{\perp}, U >$ and a solution concept f (or bargaining mechanism) is denoted $u^{\top} = f(B)$.

Nash (1950) presented a solution concept now known as the *Nash bargaining solution* (NBS). It suggests a list of axioms (in Table 5.1) that should hold for – what has been called – a "fair" deal of negotiating agents. The axioms restrict the game in such a way that, if they are all accepted, the unique solution simply becomes the maximised product of the agents' utilities

$$u^{\top} = f_{\text{NBS}} \left(< u^{\perp}, U >\right) \triangleq \max_{u=(u_1,u_2) \in U^+} \left(u_1 - u_1^{\perp}\right)\left(u_2 - u_2^{\perp}\right), \qquad (5.1)$$

where $U^+ \subseteq U$ is the set of *individually rational utilities*, i.e., those no worse than $u^{\perp}$ for both players.

The solution concept is, furthermore, based on the assumption that U is compact and convex. For some games this is intuitive, e.g., consider the utility space of the `pie-splitting` game in Figure 5.1. In that game, two agents bargain about how to split a pie. The utility of a deal for an agent is the share of the pie it is awarded. The utility space U here is confined by the axes and the diagonal from $(0,1)$ to $(1,0)$ which contains all *Pareto optimal* solutions.[9] The set of Pareto optimal solutions

[9] The several terms used in this chapter that include the word Pareto were named after the Italian economist Vilfredo Pareto (b. 1848, d. 1923).

Efficiency The solution $u^{\top}$ should be Pareto optimal, i.e., there should be no other utility pair that at least one of the agents would prefer while the other is indifferent.

Invariance of equivalent utility representations Only an agent's (ordinal) preference over possible deals should have an impact on the resulting deal, not the cardinalities (i.e., numeric values) of utilities.

Anonymity (symmetry) If the agents' utility space is symmetric, i.e., if there for all utility pairs $(x, y) \in U$ exists a symmetric pair $(y, x) \in U$, and $u_1^{\perp} = u_2^{\perp}$, then the resulting deal should be equally valued by both agents, i.e., $u_1^{\top} = u_2^{\top}$. The anonymity axiom suggests that if the agents have equal opportunities in terms of utility then the deal should be of equal value to both agents.

Independence of irrelevant alternatives Given a game $B_U =< u^{\perp}, U >$ and its solution $u^{\top} = f(B)$, the solution of another game $B_V =< u^{\perp}, \mathbf{V} >$ should also be $u^{\top}$ if $\mathbf{V} \subseteq U$ (and $u^{\top} \in \mathbf{V}$). That is, the removal of alternatives not in the solution should not affect the solution.

Table 5.1: The Nash bargaining axioms

is known as the *Pareto frontier*. If the agents can not come to an agreement, they will get the shares of the pie represented by $u^{\perp}$, and the rest of the pie will go to waste.

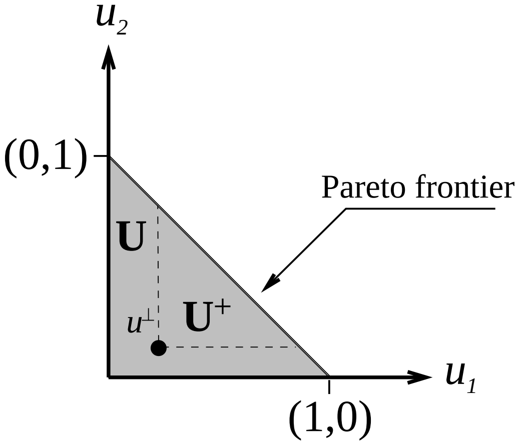


Figure 5.1: The utility space U of the `pie-splitting` game.

In other games, the set of deals is inherently discrete, e.g., games where a deal is a distribution of indivisible commodities between the agents. An example of this situation is given in Nash's original article (Nash, 1950). For instance, in an agent encounter with two agents, each possessing some kind of tools, a beneficial exchange of tools can lead to both agents performing their individual tasks more efficiently. In such cases, a convex and compact space of utilities can still be achieved by considering all *lotteries* that include the deals. For instance, consider the set of deals $\Delta = \{\delta_1, \ldots, \delta_7\}$ in Figure 5.2, with corresponding utility pairs $(u_1(\delta_i), u_2(\delta_i))$. Lottery $L = [p, \delta_1; (1-p), \delta_2]$, with $p \in [0, 1]$, is the mixed deal (or probabilistic deal) that the pure deal δ_1 is jointly selected by the agents with probability p and the pure deal δ_2 is selected with probability $(1-p)$. The utility of a lottery for an agent a_i is simply the expected utility over the pure deals, i.e., $u_i(L) = pu_i(\delta_1) + (1-p)u_i(\delta_2)$. The solid line between deals δ_1 and δ_2 in Figure 5.2 represents the utilities of all lotteries between δ_1 and δ_2 for all values of p.

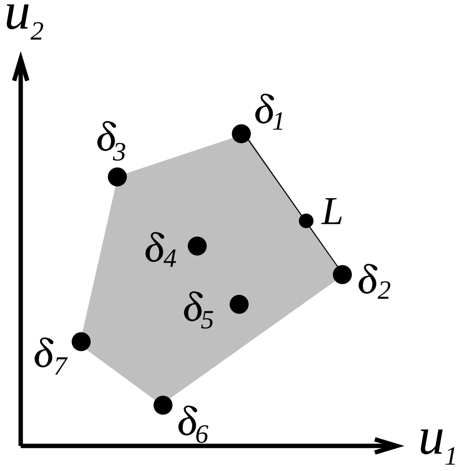


Figure 5.2: A convex and compact utility space can be achieved by constructing lotteries of the pure deals in Δ.

If lotteries such as L are acceptable deals for the interaction between agents, then one can construct a convex space of utilities and apply the NBS to find a solution.

An interpretation of the NBS for competitive agents in DAI, known as the *product maximising mechanism* (PMM), has been proposed by Zlotkin and Rosenschein (1996a). PMM is true to the NBS in that it maximises the products of utilities, but has the important differences that it expresses the outcome in actions instead of their utilities and that it does not require compact and convex utility sets. Hence, the PMM can output a pure deal even if there is a mixed deal that would give a higher expected utility product (if the system designer thinks it is appropriate). Furthermore, the PMM specifies that if there is more than one deal that maximises the product of utilities, the one amongst them that maximises the sum of the utilities is chosen. If there are still more than one that maximises both the product and sum, one of them is selected randomly.

Our implementation of the PMM for two agents a_{n_1} and a_{n_2} is shown in pseudo code in Algorithm 5.2. It is our product maximising version of the Bargaining function (called in line 5 in Algorithm 5.1). The worst case running time of PMM is $\Omega(2|\Delta_{n_1 n_2}|) = \Omega(2D_{\max}^2)$ which happens when the agent utilities of all deals have to be both multiplied and summed.

Algorithm 5.2: The product maximising mechanism
PMM$(\Delta_{n_1 n_2}, \mathcal{P}_{n_1 n_2})$
Input: Deals $\Delta_{n_1 n_2}$, Preferences $\mathcal{P}_{n_1 n_2} = (u_{n_1}, u_{n_2})$
Output: A unique solution deal $\delta^\top$

```
$\Delta^\otimes \leftarrow \arg\max_{\delta \in \Delta_{n_1 n_2}} u_{n_1}(\delta) u_{n_2}(\delta)$
if $|\Delta^\otimes| = 1$
    return $\Delta^\otimes$
$\Delta^\oplus \leftarrow \arg\max_{\delta \in \Delta^\otimes} (u_{n_1}(\delta) + u_{n_2}(\delta))$
return $\Delta^\oplus$
```

In the following section, we will discuss some shortcomings of the NBS and PMM in DAI in general, and for benevolent agents in particular.

Shortcomings of the NBS for Negotiating Benevolent Agents

When considering NBS as a candidate for a negotiation solution mechanism among benevolent agents, the first thing we have to be aware of is that ever since the seminal article on bargaining theory (Nash, 1950), research in economics has been focusing on the utility values ultimately attributed to each agent. The actual deal that led to those utilities is not considered important. However, when designing mechanisms for cooperative agents in computer science the situation is quite the opposite; the utilities merely express the agent's preferences over deals and have typically no meaning once the bargaining encounter has ended. The deal agreed upon, on the other hand, should be efficiently implemented. Consequently, a game form which explicitly states the candidate deals is desired.

First, we restate the bargaining game as $< \delta^\perp, \Delta, \{u_i\} >$, similar to Osborne and Rubinstein's (1994, pp. 299) notation, to include the possible deals in its characterisation. Inspired by Sandholm (1999, pp. 221), we then rewrite Equation 5.1 to select the desirable deal $\delta^\top$ so that

$$\delta^\top = \arg\max_{\delta \in \Delta^+} \left(u_1(\delta) - u_1(\delta^\perp)\right) \left(u_2(\delta) - u_2(\delta^\perp)\right), \tag{5.2}$$

where, analogously to U^+ in Equation 5.1, $\Delta^+ \subseteq \Delta$ is the set of deals whose utilities are at least as great as those for $\delta^{\perp}$ for both agents, i.e.,

$$\Delta^{+} = \left\{ \delta \in \Delta | u_1(\delta) \geq u_1(\delta^{\perp}) \wedge u_2(\delta) \geq u_2(\delta^{\perp}) \right\}.$$

At least two concerns should be raised towards designing a bargaining mechanism for benevolent agents based on Equation 5.2: i) *the multiplicity of optimal deals* $\delta^{\top}$ and ii) *the consequences of lotteries on cooperation.*

For several reasons there might be multiple solutions to Equation 5.2. First, with no specific restriction on the individual utilities of the agents for each pure deal, there might be several deals that yield the same utility for both agents resulting in potentially multiple solutions. Second, the risk of multiple optimal solutions also arises from the concept of lotteries itself. In Figure 5.3, for example, where the three deals δ_1, δ_2 and δ_3 are aligned, there are multiple lotteries all corresponding to the solution $\delta^{\top}$. At least there are a $p \in [0, 1]$ and a $q \in [0, 1]$ such that $u_i(L_{12}^p) = u_i(\delta^{\top})$ and $u_i(L_{32}^q) = u_i(\delta^{\top})$ for $i \in \{1, 2\}$, where L_{12}^p is the lottery $[p, \delta_1; (1-p), \delta_2]$ and L_{32}^q is the lottery $[q, \delta_3; (1-q), \delta_2]$. Also note that there are infinitely many lotteries that include all three pure deals, e.g., $[w_1, \delta_1; (1-w_1), [w_2, \delta_3; (1-w_2), \delta_2]]$ for some $(w_1, w_2) \in [0, 1] \times [0, 1]$. Hence, unless the bargaining mechanism unambiguously precludes all but one of the possibly many solutions, extra communication between the agents is necessary to establish the mixed (or occasionally pure) deal to implement.

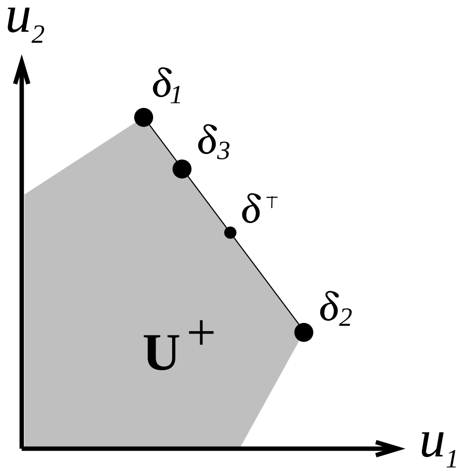


Figure 5.3: When focusing on deals rather than utilities, the problem of multiple solutions arises. The alignment of the deals δ_1, δ_2 and δ_3, makes it possible to construct the mixed deal $\delta^{\top}$ in several ways.

We saw that the use of lotteries might confuse a bargaining mechanism by introducing multiple solutions, among which there is no obvious preferred one. Admittedly, the phenomenon depicted in Figure 5.3 could be considered to be rare. The possibly negative effects of simply selecting one of the candidate lotteries are presumably negligible. However, the following discussion provides some more examples of difficulties with lotteries.

A complication with lotteries as a solution to a bargaining encounter is that they typically can not be directly implemented by the agents. If the bargaining concerns an infinitely divisible resource (e.g., money, electricity, etc.) that should be distributed over some targets (e.g., investments), a lottery could be a solution that could be immediately implemented (i.e., the lottery then represents a distribution of the resource over the targets). If, however, a lottery can not be immediately implemented, both agents have to make sure that the other implements the same pure deal, as the consequences otherwise might be disastrous. Compare to the "Chicken game", as an example, where two drivers are approaching each other in the middle of a road (Rasmusen, 2001, pp. 74). Neither driver is interested in yielding for any other driver, since yielding would incur a presumed loss of prestige. An outcome δ_{yc}=`(yield, continue)` for two prestigeful agents a_1 and a_2, would be beneficial for a_2 but embarrassing for a_1 and vice versa for the outcome δ_{cy} = `(continue, yield)`. The optional outcomes δ_{yy}=`(yield, yield)` and δ_{cc}=`(continue, continue)` corresponds to a draw (which is less prestigious than to frighten the other driver to yield) and an imminent death for both drivers (which both drivers consider to be the worst possible outcome), respectively. If the deal agreed upon is a lottery $L = [p, \delta_{yc}; (1-p), \delta_{cy}]$, then some extra *correlation*[10] is necessary, e.g., a common randomising device (Rasmusen, 2001, pp. 74) such as the flip of a single coin observed by both agents, to make sure that both implement the same deal. The disaster in this case corresponds to a_1 implementing deal δ_{cy} and a_2 deal δ_{yc}.

The inherent need for extra coordination is not the only reason to question lotteries as bargaining deals. Consider once again the "Chicken game". In Figure 5.4, the four pure deals of the game and the convex utility space U of all lotteries are shown. In the convex utility space, required by the NBS, the best deal is, as in the example before, the lottery $L = [p, \delta_{yc}; (1-p), \delta_{cy}]$. Note, however, that although the product of expected utility is maximised for the lottery deal L, the coordinated deal that the agents actually implement is either δ_{yc} or δ_{cy}. Therefore, the lottery solution will arbitrarily (randomly) favour one of the agents. The pure deal δ_{yy}, which in the game corresponds to no agent losing prestige and which is undominated (and therefore Pareto optimal) by the pure deals in L, is ignored. Hence, even though δ_{yy} might be considered to be a more "fair" deal than any of δ_{yc} and δ_{cy} it is disregarded as a solution.

A lottery deal would perhaps be acceptable as a solution if the same game was played repeatedly, i.e., some fraction (similar to p) of the games resulting in the deal δ_{yc} and some in δ_{cy}. However, the outcome of the game is inherently dependent on the utilities of the two agents, which typically change between every

[10] Correlation as in *correlated strategy* and *correlated equilibrium* is a term from game theory. To maintain a consistent terminology in this chapter, we will write *coordination* instead.

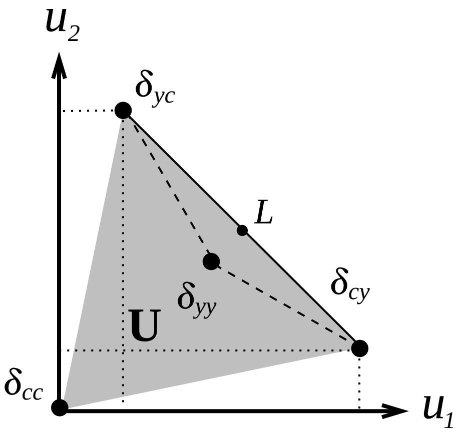


Figure 5.4: In the NBS solution, the lottery deal L is preferred to the pure deal δ_{yy}.

encounter in dynamic environments (as the problem changes and the agents acquire more information). Hence, the exact same game can not be anticipated with every encounter between two agents.

In summary, we claim that the NBS (and the PMM to some extent) could be challenged as the preferred conflict resolution mechanism among bargaining benevolent agents. The reasons are briefly, as explained above, the multiplicity of candidate deals, the spurious fairness of lotteries, and the tendency to randomly favour one of the agents.

The Egalitarian Bargaining Solution

In this section, we explain the details of another bargaining mechanism based on the so called egalitarian solution.

Over the years, other solution concepts for the bargaining problem have been proposed. Some of Nash's axioms (Table 5.1), e.g., the independence of irrelevant alternatives axiom, have been questioned, and others have been introduced. Some of these alternative ideas are presented by Kalai (1985), Dagan et al. (2002), and Thomson (1994).

To address the mentioned shortcomings of the NBS for bargaining benevolent agents, we adapt one of the alternative bargaining solutions for computer science.

Formally, a solution f_{EGT} (see, e.g., Kalai, 1985, pp. 91) is *egalitarian* (EGT) if there are weights $\lambda_1, \lambda_2 > 0$ such that for every game $< u^{\perp}, U >$, $f_{EGT}(< u^{\perp}, U >)$ is Pareto optimal in U and satisfies

$$\lambda_1 \left(f_{EGT,1}(< u^{\perp}, U >) - u_1^{\perp} \right) = \lambda_2 \left(f_{EGT,2}(< u^{\perp}, U >) - u_2^{\perp} \right), \qquad (5.3)$$

where $f_{EGT,i}$ is the utility $u_i^\top$ for agent a_i of f_{EGT}. The egalitarian solution satisfies the axioms of *Pareto optimality*, *symmetry* and *strong monotonicity* (Thomson, 1994, pp. 1251). The first two were valid also for the NBS and were described in Table 5.1. The final axiom, monotonicity, says that a solution f is monotonic if for every two bargaining pairs $< u^\perp, U >$ and $< u^\perp, U' >$ when $U \subseteq U'$ then $f_i(< u^\perp, U >) \leq f_i(< u^\perp, U' >)$ for all i (Kalai, 1985, pp. 92). A consequence of the monotonicity axiom is that none of the agents should be worse off if the set of deals increases. Note that this is not guaranteed for NBS.

When translating the equality condition in Equation 5.3, we ignore the disagreement utilities, i.e., we set $u_1^\perp = u_2^\perp = 0$, and let $\lambda_1 = \lambda_2 = 1$.

Furthermore, for the same reasons mentioned before, we do not require the set of utilities U or the corresponding set of deals Δ to be compact. Especially, we do not attempt to express lotteries of deals when the set of deals is discrete. The desirable set of deals is now reinterpreted from Equation 5.3 as deals that belong to the Pareto frontier Δ^π and for which

$$\textsc{diff}(\delta) \triangleq u_1(\delta) - u_2(\delta) = 0. \tag{5.4}$$

However, since the utilities of the deals in the Pareto frontier Δ^π typically are points (and not compact sets) we can not be guaranteed to find a solution to Equation 5.4. Instead, we try to minimise the difference of utilities and end up with the following requirement on a desirable deal:

$$\delta^\top = \arg\min_{\delta \in \Delta^\pi} \textsc{diff}(\delta) = \arg\min_{\delta \in \Delta^\pi} |u_1(\delta) - u_2(\delta)|. \tag{5.5}$$

Algorithm 5.3 is our egalitarian version of the Bargaining function in Algorithm 5.1. It assumes that both agents, say a_{n_1} and a_{n_2}, are aware of each other's preferences, represented by their utility functions u_{n_1} and u_{n_2}, and is performed by both agents individually. We denote the agent that initiated the negotiation by a_{n_1}.

```
Algorithm 5.3: The egalitarian mechanism
EGALITARIAN($\Delta_{n_1 n_2}, \mathcal{P}_{n_1 n_2}$)
Input: Deals $\Delta_{n_1 n_2}$, Preferences $\mathcal{P}_{n_1 n_2} = (u_{n_1}, u_{n_2})$
Output: A unique solution deal $\delta^\top$
(1)   $\Delta^\pi_{n_1 n_2} \leftarrow$ FIND_PARETO_FRONTIER($\Delta_{n_1 n_2}, \mathcal{P}_{n_1 n_2}$)
(2)   $\Delta^{\textsc{diff}} \leftarrow \arg\min_{\delta \in \Delta^\pi_{n_1 n_2}} \textsc{diff}(\delta)$
(3)   if $|\Delta^{\textsc{diff}}| = 1$
(4)       return $\Delta^{\textsc{diff}}$
(5)   $\Delta^{\textsc{diff}1} \leftarrow$ FIND_ADV_DEALS($\Delta^{\textsc{diff}}, u_{n_1}$)
(6)   return $\Delta^{\textsc{diff}1}$
```

In the first line of the algorithm, the Pareto frontier $\Delta^{\pi}_{n_1 n_2}$ of the set of deals $\Delta_{n_1 n_2}$ is found. The Pareto frontier always exists, but typically has more than one element. Appendix D provides an algorithm for finding the Pareto frontier.

In the second line of the algorithm, the deals (hopefully there is only one) Δ^{DIFF} from the Pareto frontier that minimise the function $\text{DIFF}(\delta)$ are selected. Line three checks whether there is only one deal that minimises the difference function in line two. If there is only one such deal it will be returned. If there is more than one, the agents look (in line five) for the deals (among Δ^{DIFF}) that are most preferred by agent a_{n_1}, i.e., the agent that started the negotiation. The set $\Delta^{\text{DIFF}1}$ will contain those deals δ from Δ^{DIFF} where $u_{n_1}(\delta) > u_{n_2}(\delta)$. Hence, the mechanism prescribes an asymmetry in that the agent who initiates the negotiation will have an advantage in case Δ^{DIFF} is ambiguous. The asymmetry, i.e., that one of the agents gets an extra advantage, is the price to pay to elude the, otherwise, extra communication needed to jointly select deals from Δ^{DIFF}. It could happen that also $\Delta^{\text{DIFF}1}$ is ambiguous. If so, the agents can select one of the remaining deals according to some rule that is specified by the mechanism. Examples of such rules, e.g., to select the first alternative in $\Delta^{\text{DIFF}1}$, are given in Section 5.3.

The worst case time complexity of Algorithm 5.3 is $\Omega(|\Delta_{n_1 n_2}|^2 + 2|\Delta_{n_1 n_2}|)$. The first term originates from finding the Pareto frontier in Algorithm 5.3 (see Appendix D). The second term comes from finding the Pareto efficient deals with the smallest difference in utility and then finding those that are favoured by agent a_{n_1}.

Due to the restriction of a discrete space of pure deals, only the first axiom (Pareto optimality) is valid for the egalitarian mechanism in Algorithm 5.3. Symmetry is sacrificed for the practical reasons just explained. Monotonicity is, furthermore, not guaranteed if the deal set is expanded with Pareto efficient deals whose difference in utility is smaller than the currently best one. Hence, although some of the axioms are not fully respected, the essence of the egalitarian approach is preserved, i.e., that of yielding solutions for agents that are efficient and of similar value to them.

5.5 Mechanism and Protocol Evaluation Criteria

In this section, we evaluate the coordination protocol described in Section 5.3 with respect to a number of evaluation criteria. Mechanism evaluation criteria are listed by, e.g., Kraus (2001a, pp. 2), Sandholm (1999, pp. 202), and Zlotkin and Rosenschein (1996a, pp. 207). Some criteria, such as *stability*, are mostly useful for competitive MAS. Some criteria that we consider to be relevant in our context are:

1. **Negotiation time** As the environment is typically dynamic and, hence, changes while agents are negotiating, the negotiation should end as quickly as possible. In the worst case, by the time the negotiation reaches its conclusion, the information state it was based upon might have become obsolete.

2. **Simplicity** A mechanism should require as little communication and computational efforts for the agents as possible.

3. **Efficiency** Basically, a bargaining deal δ should only be accepted as an outcome of a negotiation encounter if it is Pareto optimal.[11]

4. **Distribution** Preferably, a mechanism should require no external mediator to solve the negotiation conflict. One of the reasons is that a centralised third party will become a bottle-neck and the whole negotiation mechanism will be sensitive to failure.[12]

5. **Symmetry** The symmetry criterion proposes that the mechanism should not treat the agents differently or arbitrarily favour any of them. An asymmetric mechanism requires some overhead work (an extra mechanism) to assign roles to agents in the encounter (Zlotkin & Rosenschein, 1996b, pp. 181).

We evaluate according to the negotiation time and simplicity criteria in the following way. As the agents are benevolent, they are willing to truthfully share information with each other. Hence, there is no need for an agent to waste time trying to figure out what the state of its negotiation counterpart really is. Each agent can, given its own desires and the desires of the opponent, find a unique compromise joint action and implement it without further interaction. Furthermore, the time spent in each negotiation is decreased by only allowing two agents to participate in each negotiation. What is potentially not simple about the mechanism is that it requires an exhaustive utility evaluation of all deals. If the number of deals is an issue, the evaluation of a random sample of deals might suffice (if the agent that initiates the negotiation randomly selects the indices of the deals that should be considered and transmits this information to its counterpart).

The computational complexity of each agent for an encounter of the type described in Section 5.4 is $O(|\Delta_{12}|K + B)$. For PMM B is $\Omega(2|\Delta_{12}|)$ and for EGT $\Omega(|\Delta_{12}|^2 + 2|\Delta_{12}|)$.

Both PMM (and NBS) and EGT are efficient since they select a deal from the Pareto frontier. The coordination protocol is, furthermore, designed to be distributed. PMM is symmetric but EGT is not. Although symmetry is desired, it is lost to some extent in a trade-off with simplicity. The agent role assignment for a negotiation encounter is, furthermore, uncomplicated. The agent that initiates the encounter assumes the role of agent a_1 (i.e., the one who will get to decide the deal in case of ambiguity) and the responding agent automatically assumes the role of a_2. A similar asymmetry could, of course, also be introduced for PMM.

[11] Note that a non-Pareto optimal solution is acceptable in some research when it satisfies some minimum requirements by each agent (a so called *satisficing solution*, Pirjanian & Matarić, 2000) and the alternative of finding a Pareto-optimal one is too costly.

[12] Some mechanisms require a centralised mediator, e.g., the *price tâtonnement* market mechanism (Sandholm, 1999, pp. 227).

5.6 Experiments

In this section, we present two experiments involving the coordination protocol described in Section 5.3. The first experiment compares the coordination protocol (an egalitarian implementation) to a fully centralised approach that exhaustively evaluates all possible joint actions for all agents, and an individual approach where no coordination at all occurs. The purpose is to compare solutions quality and time efficiency of the three solution types. The second experiment gives examples of a situation where the solutions of the NBS (and PMM) give unsatisfactory results.

Experiment 5.1: Verifying the Efficiency of Coordination

In the first experiment, we compare the coordination protocol (`Coordinated`) to two extremes: the centralised approach (`Centralised`) and a solution where the agents act independently (`Individual`). Our purpose is to demonstrate the efficiency of `Coordinated` compared to `Centralised` in terms of computational cost and its efficiency compared to `Individual` in terms of the quality of their solutions.

In the simulation described here, the common objective of a set of mobile agents is to explore a common environment as quickly as possible. The simulation environment is highly simplified to facilitate comparisons based on Monte-Carlo simulations. Admittedly, since the problem is static (the grid and initial configuration is known), it could be solved more efficiently by allowing the agents to plan their paths beforehand and execute them without intervention. However, we use this simple problem as a benchmark for comparisons.

The grid world environment is depicted in Figure 5.5(a). The mobile sensors (observer agents), that move vertically or horizontally across the grid, are shown as circles, 'o'. Next to a circle is a plus '+' which shows in which direction the observer is looking. The position of a target (used in the next experiment in Section 5.6) is marked with a star '*'.

To make the comparisons less dependent on the environment, we assume that the grid world has the properties of the surface of a torus. Hence, the grid world wraps around both vertically and horizontally. For instance, an agent moving left from grid position $(1, 8)$ will end up in position $(10, 8)$.

Each agent has a field-of-view which is limited to the grid position it is in, one step ahead in the viewing direction, and the diagonally adjacent grid points in its viewing direction (see Figure 5.5(b)).

The resulting deal $\delta^\top$ of a negotiation is a joint action pair $(d_1^\top, d_2^\top)$. An action is defined as a one-step-movement in one direction (north, south, east or west) and an observation in the resulting field-of-view. The utility of a deal $\delta = (d_1, d_2)$ for an agent a_i, i.e., $u_i(\delta)$, negotiating with another agent a_j, we tentatively define as

$$u_i(\delta) = C - so_i(\delta) - oo_i(\delta), \tag{5.6}$$

where C is a constant (large enough so that $u_i > 0$), $so_i(\delta)$ is the number of grid points that a_i sees (given action d_i) that it has seen before, and $oo_i(\delta)$ is the number of grid points that a_j sees (given its action d_j) that a_i has already seen. Hence, high utility for an agent a_i is given to deals that lead to a state where a_i does not review previously observed grid points and where its bargaining counterpart a_j does not observe grid points that a_i has already seen.

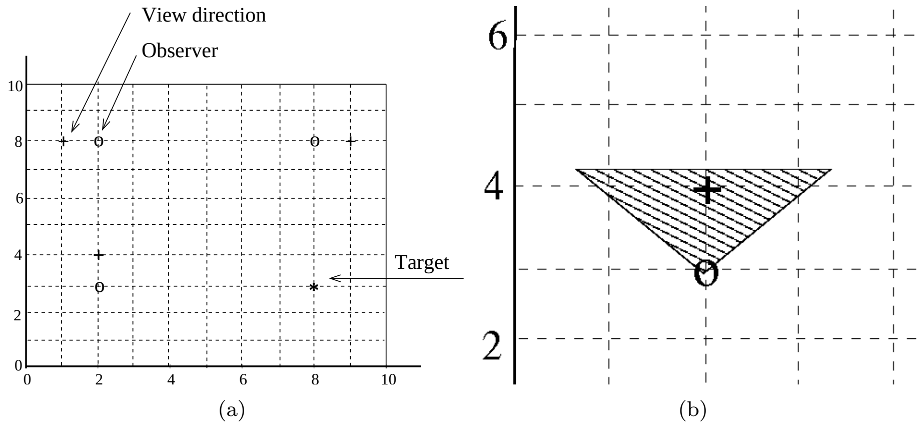


Figure 5.5: a) The grid world b) The field of view of an agent is limited to its current position, and one step ahead in its viewing direction. It can also observe the diagonally adjacent grid points in the viewing direction.

The simulation proceeds in a series of discrete time steps. In the beginning of each time step, the agents try to connect to their neighbours (in the current implementation, the two agents that are geographically closest to an agent are considered to be its neighbours). The order in which the agents connect to each other is random. An agent will stick to the action, say $d^{\top}$, it selected in its first encounter of the time step. If it engages in any other negotiation, it will only accept deals which contain its selected action $d^{\top}$. In the next simulation step, when agents have performed their actions, they are free to choose any feasible action again.

The simulation software we have developed has a number of configuration parameters: grid size $gs = (x\_sz, y\_sz)$, number of observers *noo*, and the maximum number of time steps before the search is halted *mnts*.

The two approaches used for comparison can now be described in the following way:

`Centralised` The centralised approach enumerates and evaluates all $|D|^{|\mathcal{A}|}$ joint actions in each time step. A joint action (that involves all agents) that maximises the number of new grid points observed by the team is selected.

`Individual` In the individual approach, each agent tries to explore the whole grid by itself. The actions an agent selects are only dependent on the grid points it has observed. There is, therefore, no interaction at all between the agents. The agents are, furthermore, not assumed to interfere with each other's activities.

For each of the three approaches, we applied a Monte-Carlo simulation (to estimate the average performance for each approach) where we drew the initial positions for the observers randomly from a uniform distribution over the grid, making sure that an observer did not end up in the same position as another observer. The initial positions were drawn 1000 times for each approach and simulation configuration and result is shown in Table 5.2.[13] We consider the exploration task of the agents to have been completed once they collectively have explored fifty percent of the grid world.

In the table, the **avg sol time** is the time (in real-time seconds) spent by Matlab to find negotiation solutions to the coordination problem. **std sol time** is the standard deviation of the Monte Carlo sample set. In the case of the `Centralised` approach, the number of seconds shown is the exact time spent by Matlab finding the optimal joint action. In the case of `Coordinated` and `Individual`, however, the total amounts have been divided by the number of observers as they are supposed to act more or less in parallel. In the `Individual` approach this is true, but due to the interaction between agents in the `Coordinated` approach, the actual time spent will be longer.

The fifth column in Table 5.2, **avg sim time**, shows the fraction of simulation time steps required before the objective was reached (only counting those simulations where it was actually reached). The next column, **std sim time**, is the corresponding standard deviation.

As expected, the completion frequency for `Coordinated`, shown in the final column, i.e., the ratio of simulations where the agents managed to explore half of the grid world, falls between `Centralised` and `Individual`.

In Figure 5.6, a visual comparison of the development of the grid coverage is shown. The simulation parameters are $gs = (20, 20)$, $noo = 5$, $mnts = 25$ and 200 Monte Carlo simulations were run for each approach. The main difference of the graphs is the diversity in grid coverage. The fifty percent grid coverage level is marked for each of the approaches, and all three have a best simulation where that level of coverage is reached after about 12 time steps. Their worst simulations, however, differ greatly. `Centralised` does not have a single simulation below fifty percent coverage after 15 time steps, `Coordinated` has no simulation worse than 19 time steps, and `Individual` has plenty worse than 20 time steps.

The large diversity of `Individual` in Figure 5.6(c) is due to the varying random initial conditions of the samples and the randomised selection of optimal actions. From the perspective of grid coverage, the coordination protocol used for `Coordinated` makes the performance (in terms of grid coverage) less dependent on the aforementioned factors that greatly affect `Individual`.

[13] The Matlab simulations were run on an Intel-based PC with 1500 MHz processor and 256MB RAM.

Table 5.2: Results for the three approaches for different experiment configurations

Configuration	**Approach**	**avg sol time (s)**	**std sol time (s)**	**avg sim time**	**std sim time**	**complet freq (%)**
10 by 10 grid	`Centralised`	0.0189	0.0081	8.0020	0.0447	100
2 agents	`Coordinated`	0.0349	0.0142	8.0480	0.2230	100
20 time steps	`Individual`	0.0041	0.0031	9.4490	1.3933	100
15 by 15 grid	`Centralised`	0.0201	0.0059	18.1810	0.3952	100
2 agents	`Coordinated`	0.0371	0.0092	18.2190	0.4574	100
30 time steps	`Individual`	0.0036	0.0023	21.4775	2.3838	99.9
15 by 15 grid	`Centralised`	0.1049	0.0319	12.0770	0.2667	100
3 agents	`Coordinated`	0.0344	0.0101	12.9920	0.9924	100
30 time steps	`Individual`	0.0038	0.0012	15.0100	2.1402	100
20 by 20 grid	`Centralised`	0.1115	0.0253	11.1850	0.7440	100
3 agents	`Coordinated`	0.0356	0.0076	23.5936	1.5536	99.9
30 time steps	`Individual`	0.0037	0.0010	26.3600	2.1897	87.5
30 by 30 grid	`Centralised`	0.1185	0.0173	51.3795	2.3086	97.5
3 agents	`Coordinated`	0.0367	0.0052	53.8099	2.5513	93.1
60 time steps	`Individual`	0.0037	0.0010	57.1059	2.1215	42.5
30 by 30 grid	`Centralised`	0.6072	0.1011	38.1950	2.1657	100
4 agents	`Coordinated`	0.0332	0.0054	41.9260	3.0305	100
60 time steps	`Individual`	0.0037	0.0013	47.4020	3.7711	99.5
50 by 50 grid	`Centralised`	3.8295	0.5779	90.0500	4.8036	80
5 agents	`Coordinated`	0.0388	0.0166	94.9921	3.4094	63.2
100 time steps	`Individual`	0.0035	0.0010	97.5000	2.5635	16

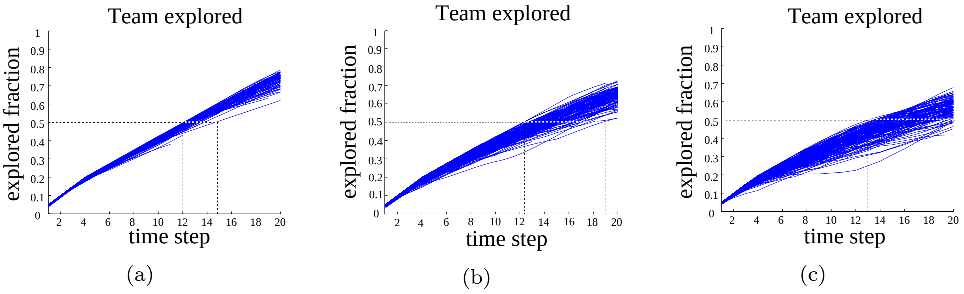


Figure 5.6: a) `Centralised` b) `Coordinated` c) `Individual`

Table 5.2 and Figure 5.6 show the expected result; that the performance of the coordinated approach falls between an entirely centralised and an entirely individual approach. The coordinated approach is much less computationally costly than the centralised one, and more efficient than the individual one.

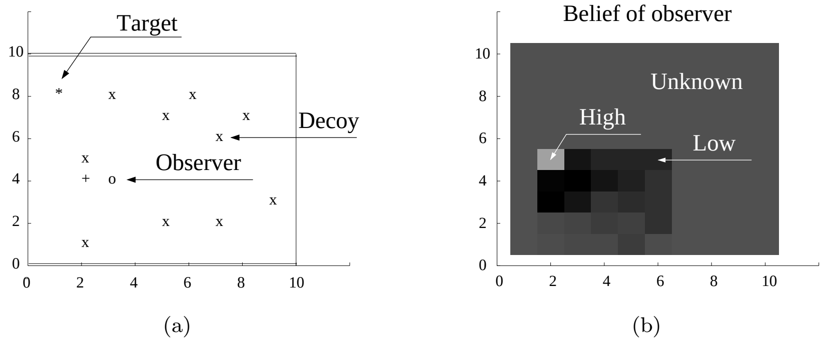
For this exploration problem, the NBS and PMM have similar performance to `Coordinated`.

Experiment 5.2: Targeting the Shortcomings of NBS

By the second experiment, we want to give an example where the "arbitrariness", i.e., the tendency to arbitrarily favour one of the agents described in Section 5.4, of the NBS and PMM has some undesired effects on the outcome. We, thus, want to compare negotiation protocols implemented with the following of bargaining strategies: the Nash bargaining solution (`NBS`), the product maximising mechanism (`PMM`) and the egalitarian solution (`EGT`). The difference between our implementations of the `NBS` and the `PMM` is that the `NBS` finds lottery solutions (after which a pure deal is selected from the lottery) and the `PMM` only considers the pure deals.

For this experiment, we re-design the simulation world from the previous experiment. First, we introduce a target, marked with a '*' in Figure 5.7(a), which during the simulation moves from the upper part of the grid to the lower. If the target reaches the first row of the grid without being detected and classified, we say that it has managed to escape the grid. Furthermore, the grid no longer wraps around vertically, meaning that the agents can not move upwards from the top row and downwards from the first row.

The field of view of the agents is the same as in Experiment 5.1. If the target is within the field of view of an agent its sensors will detect it with a high probability (95% in the simulations). However, a single detection is not enough to classify the target as a fraction of the grid is occupied by *decoys*, which the agents' sensors also are sensitive to. A decoy in the field of view causes a detection with 50% probability. An agent can distinguish which grid cell a detection originates from.



(a) (b)

Figure 5.7: a) A grid with an agent, target and decoys b) Certainty grid

The decoys, and the density of decoys, will be important to show the performance difference between EGT and NBS.

The belief state of an agent (i.e., its "guess" about the whereabouts of the target) is represented by a *certainty grid* (see, e.g., Elmenreich et al., 2001). The momentary certainty grid of the agent is shown in Figure 5.7(a) and its corresponding belief is shown in Figure 5.7(b). Each grid cell contains the belief of the agent that the target sought is located in that cell. In the figure, grey colour represents ignorance (i.e., that the agent does not know what (if anything) is located in that cell), a darker tone represents a disbelief in the target being in the corresponding cell, and a lighter tone a higher probability that the target is in fact present in the cell. In the figure, the observer's belief in the presence of the target in cell $(2, 5)$ has increased due to the detection of the decoy in that position.

The belief of the cells is updated using conventional Bayesian updating. Let $\mathbf{c}$ denote a coordinate in the grid and t the simulation time step. Then the updated probability distribution in coordinate $\mathbf{c}$ is

$$P_{\mathbf{c}}^{t}(H_i) = \frac{P(O_{\mathbf{c}}^{t}|H_i)P_{\mathbf{c}}^{t-1}(H_i)}{\sum_j P(O_{\mathbf{c}}^{t}|H_j)P_{\mathbf{c}}^{t-1}(H_j)}, \tag{5.7}$$

where $P_{\mathbf{c}}^{t-1}(H_i)$ is the prior belief of hypothesis H_i in grid coordinate $\mathbf{c}$. $P(O_{\mathbf{c}}^{t}|H_i)$ is the likelihood of observation $O_{\mathbf{c}}^{t}$ in coordinate $\mathbf{c}$ at time t given hypothesis H_i. An observation $O_{\mathbf{c}}^{t}$ is either classified as *detection*, *no detection* or *no observation.* If grid coordinate $\mathbf{c}$ was not observed by the agent, the update in Equation 5.7 will not be used. The hypotheses $H = \{H_i\}_i$, in our case are, *target present* (H_1), *decoy present* (H_2), and *no presence* (H_3). For simplicity, no observations (or likelihood distributions) are shared between the sensors, and hence only local observations contribute in Equation 5.7.

Since the target is moving, a grid coordinate that was previously believed to be empty might suddenly contain the target. To accommodate this fact, we let the belief for each coordinate decay over time towards ignorance (i.e., a uniform distribution, $\forall i P(H_i) = |H|^{-1}$, over the hypotheses). The decay is expressed in the following way:

$$P_{\mathbf{c}}(H_i) \leftarrow P_{\mathbf{c}}(H_i) + \left(|H|^{-1} - P_{\mathbf{c}}(H_i)\right)\beta, \tag{5.8}$$

for some $\beta \in (0, 1]$, and is performed in every time step.

Figure 5.7(b) shows the certainty grid of one of the observers in Figure 5.7(a). The light grey regions are those which the agent has not observed yet and the darker ones have been observed, but nothing has been detected.

If the belief of target presence $P_{\mathbf{c}}(H_1)$ increases above 85%, the classification is considered to have been successful, if in fact the target is located in cell **c**. If the target isn't there, the simulation continues. A decoy can also be classified. If the belief in a decoy in a cell $P_{\mathbf{c}}(H_2)$ exceeds 65%, the concerned agent maintains the belief for the rest of the simulation that cell **c** is occupied by a decoy.

We tailor a utility function u_i (for an agent a_i currently negotiating with another agent a_j) to demonstrate the potential drawbacks of the `PMM`. In this design, an agent evaluates a candidate deal δ based on three criteria:

- Criterion $\mathcal{C}_1$: *Deal δ allows agent a_i to observe some interesting grid cell*
- Criterion $\mathcal{C}_2$: *Deal δ results in agent a_j moving closer to agent a_i*
- Criterion $\mathcal{C}_3$: *Deal δ allows agent a_j to observe a previously not explored grid cell*

In the simulations, criterion $\mathcal{C}_1$ is satisfied if the deal δ means that agent a_i will observe a grid cell whose current belief is more than 0.4. Criterion $\mathcal{C}_2$ is satisfied whenever δ results in a_j moving closer to a_i. Satisfying this criterion might be beneficial for classifying the target if it happens to be close to a_i. $\mathcal{C}_2$ is crucial for the results shown later on in this section. The third criterion is satisfied if deal δ results in a_i observing a previously unexplored grid cell.

The utility function for agent a_i is then defined in this way:

$$u_i(\delta) = \begin{cases} 30, & \text{if } \delta \text{ satisfies } \mathcal{C}_1 \wedge \mathcal{C}_2 \\ 15, & \text{if } \delta \text{ satisfies } \mathcal{C}_1 \wedge \neg\mathcal{C}_2 \\ 13, & \text{if } \delta \text{ satisfies } \neg\mathcal{C}_1 \wedge \mathcal{C}_3 \\ 10, & \text{if } \delta \text{ satisfies } \neg\mathcal{C}_1 \wedge \neg\mathcal{C}_3 \end{cases} . \tag{5.9}$$

Equation 5.9 is the basis for the utility function used in the experiment. The implementation also includes small differences in utilities for field-of-views with different degrees of belief. Using u_i, a pair of agents (a_1 and a_2) might be exposed to bargaining encounters where they have to decide between the two Pareto efficient deals δ_1 and δ_2. Say that $u_1(\delta_1) = 30$, $u_2(\delta_1) = 10$, $u_1(\delta_2) = 15$, $u_2(\delta_2) = 15$. In this case, `PMM` and `NBS` will choose δ_1 (since $30 \cdot 10 > 15^2$) and `EGT` will choose δ_2 (since $|15 - 15| < |30 - 10|$). The drastic deal δ_1 (which means that a_2 is dragged away towards a_1) is obviously a worse solution than δ_2, if a_2 happens to be observing the target before the negotiation starts. Also note, however, that δ_1 might be a good choice if a_1 is actually observing the target. We will see that this difference in behaviour might lead to poor performance for NBS and PMM for recurrent negotiations.

Let us consider a grid with size $gs = (15, 10)$. We try the three mechanisms (EGT, NBS, and PMM) for difference decoy densities. The result for simulations with three observers, $noo = 3$, is shown in Figure 5.8. The fraction of simulations where a correct target classification was achieved is plotted against the decoy density.

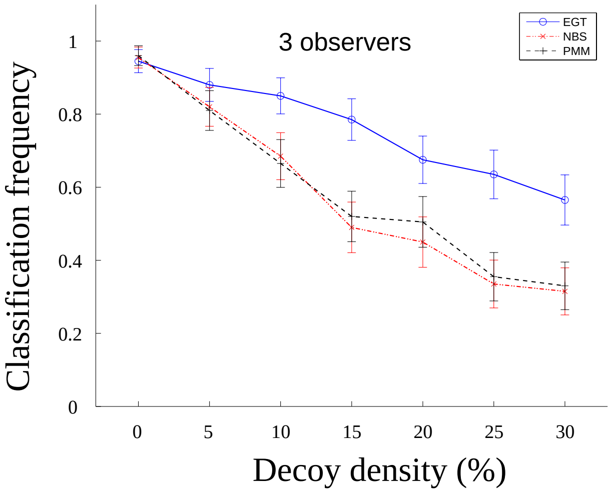


Figure 5.8: Classification degree for various decoy densities.

The confidence intervals (i.e., the vertical bars in the figure) of the 200 sample estimates are estimated in the following way. Assume that $p_{m,q}$ is the true classification degree of mechanism m for decoy density q. Then the number of successful classifications in a batch of n simulations can be seen as a stochastic variable $X_{m,q}$ with the binomial distribution $Bin(n, p)$. Already for a fairly small n, the binomial distribution with parameters n and p can be approximated by the normal distribution $N(np, \sqrt{np(1-p)})$. What we plot in Figure 5.8 is actually our estimate of $\frac{X_{m,q}}{n}$ with distribution $N(p, \sqrt{p(1-p)/n})$, which we denote $\hat{p}$ (dropping the m and q for brevity). We end up with confidence intervals

$$I_p = (\hat{p} - \lambda_{\alpha/2}\hat{d}, \hat{p} + \lambda_{\alpha/2}\hat{d}),$$

where the estimated standard deviation $\hat{d} = \sqrt{\hat{p}(1-\hat{p})/n}$. In the figure, 95% confidence intervals are shown and, hence, $\alpha = 0.05$ and $\lambda_{0.025} = 1.96$.

As can be seen in Figure 5.8, the relative performance of the EGT over NBS and PMM increases with the decoy density. The reason for the result is mainly that when the decoy density is high there are more situations, such as the one explained above, where (arbitrarily) disturbing one agent in favour of another is the preferred choice of the PMM and NBS. An agent requires multiple observations to classify a target and since it might be directed away from the candidate target it is observing, it is more likely to fail than when using EGT which prefers compromises instead.

5.7 Discussion

From a LSIA perspective, the coordination protocol presented and discussed should be considered in a wider sense. It is not limited to coordinating sets of homogeneous robots, which we can call *intra*-service coordination, i.e., coordination of service components (if we think of the set of robots as a service). Its usefulness is generally that it allows decentralised information acquisition to coordinate actions (or service configuration selections) based on the current environment state. Another application of coordination for large-scale information acquisition is *inter*-service coordination, e.g., sharing resources between services (cf Nash original example described on page 120) or avoiding detrimental joint actions. A concrete example of the latter is the following:

Example One service is realised by a camera that can take photos in different directions. Another service corresponds to a mobile platform that can measure the moisture in its current location. Say that the "camera service" has been given the task to acquire an image in some direction for object recognition. The camera service sends a description of its task to neighbouring services within radio communication range. The "moisture service" detects a conflict: if the mobile platform enters the room it will occlude the view of the camera service. Hence, a negotiation is initiated for inter-service coordination, perhaps resulting in the platform waiting for the camera to grab the requested image.

To more clearly relate action coordination to the information acquisition topic of this thesis, we describe it in terms of the service configuration space presented in Section 2.4. Each agent $a_i \in \mathcal{A}$ constitutes a service with service attribute $c_i \in \mathcal{C}_i$, where $\mathcal{C}_i$ is the set of possible actions (i.e., D_i) for a_i. The complete configuration space is then $\mathcal{C} = \times_{i=1}^{|\mathcal{A}|} \mathcal{C}_i$, but since the control is decentralised each agent can only control its own attribute. If there were no coordination between the agents, each agent would order its actions according to its own local belief of the environment (perhaps including its belief of the other agents). We can denote the ordering of each agent a_i by $\succeq_{a_i}$. Now, since the agents, described in this chapter, engage in recurrent negotiations, they might end up with other orderings. Each agent still only controls its own attribute $c_i \in \mathcal{C}_i$, but its *coordinated* ordering of actions might be different and denoted $\succeq_{a_i|A_i}$ instead, where $A_i \subseteq \mathcal{A} \setminus a_i$ is the set of agents that agent a_i has negotiated with prior to finally selecting (and executing) an action.

5.8 Summary

In this chapter, we present a coordination protocol based on recurrent negotiations for benevolent agents. Our interest in such a coordination protocol stems from its potential to realise decentralised control for LSIA (see Section 3.12). The type of environment the agents are assumed to interact within is physical and antagonistic. The physical property prescribes that agents have a limited time for negotiation and the antagonistic property that the agents should be careful communicating since communication might reveal the agents' intentions. To counter the otherwise exponential time complexity of solving negotiations, we require that the protocol only involves bilateral negotiations. For practical reasons, we also require that an agent should not try to engage in negotiations with all other agents during the same time interval.

The coordination protocol relies on a bargaining solution to resolve conflicting interests between agents and reach coordinated joint actions. A conventional choice of bargaining solution is the product maximising mechanism, which derives from the Nash bargaining solution (NBS). We critically analyse the NBS and provide experimental examples of problem configurations where other bargaining solutions (here the egalitarian solution) might be appropriate candidates. We experimentally found the bargaining approach to be faster than a centralised approach, and to yield more efficient results than an individual approach.

Incidentally, we noted some subtle difficulties with transferring the NBS from economics to our computational problem. For instance, the characterising axioms of solutions in bargaining theory are useful to compare different solutions, but care has to be taken when translating the solution to algorithms in computer science as some properties (i.e., axioms and underlying assumptions) might be undesirable, unimportant or lost in the translation. For instance, in Section 5.4, we argue that the assumption of lottery solutions might be undesirable in the computational domain; the uniqueness of optimal outcome utilities is unimportant when focusing on deals (as the uniqueness of an optimal deal can not be guaranteed, as also discussed in Section 5.4); and in our translation of the egalitarian solution, the monotonicity axiom is somewhat violated when only pure deals are considered. Additionally, computational complexity has to be considered.

A lesson learned from this study on bargaining theory is that in multi-objective optimisation problems, such as the one we study here, there are no absolute right or wrong solutions in general, i.e., different solutions are suitable for different situations. For example, as shown in the experiments, compromises (suggested by the egalitarian solution) are sometimes better than arbitrarily favouring one of the agents (which might be the result of using the Nash bargaining solution), but for other problems compromises are detrimental.

Admittedly, more could be said about coordination mechanisms than is discussed here. To get a deeper insight into coordination mechanisms, other bargaining solutions could be studied (several examples are provided by Kalai, 1985) and a comparison to the theory of social choice could be conducted. Mariotti (1998) touches upon the close relationship between bargaining theory and social choice theory. Social choice for multi-agent theory is discussed by Endriss et al. (2003, 2005) who provide an alternative (but similar) egalitarian interpretation and solution.

Chapter 6

Decentralised Control Experiments

In this chapter, we explain how the egalitarian coordination protocol presented in Chapter 5 can be applied in a mobile robotic application. The objective is to show that the coordination protocol can be of use not only in simulations.

In Section 6.1, we briefly describe the experiment problem. We describe the components of the robotic platforms used in Section 6.2. Finally, the experiments and their results are shown in Section 6.3 and the chapter is concluded in Section 6.4.

6.1 Cooperative Detection in an Office Environment

The application we are considering involves a set of mobile robots that jointly explore a common office environment. Their task is to detect an intruder in an otherwise desolate office (typically because the time of executing the application is after working hours).

If a robot detects something it will call for another member of the robot set to get a second opinion. If both robots believe that an intruder has been detected, that is what they will report. The robots also try to make sure that there is always a robot in the corridor to make sure that an intruder will not use the corridor to go unnoticed while the robots are exploring the rooms.

6.2 Equipment

For this experiment, we need robots with four fundamental components: *mobility*, *sensing modalities*, *communication* and *computational ability*. The robotic platform we have chosen is the ER1 by Evolution Robotics, Inc. It provides our system with mobility. Sensing modalities and communication ability is provided by a sensor module called Embedded sensor board (ESB).[1] The computational ability is

[1] The ESB was designed by the CST group at Freie Universität (FU) in Berlin and is sold by ScatterWeb GmbH.

provided both by the micro controller on the ESB and a (IBM Thinkpad) laptop that is carried by the ER1 (Figure 6.1). The laptop has RedHat Linux 7.3 installed.

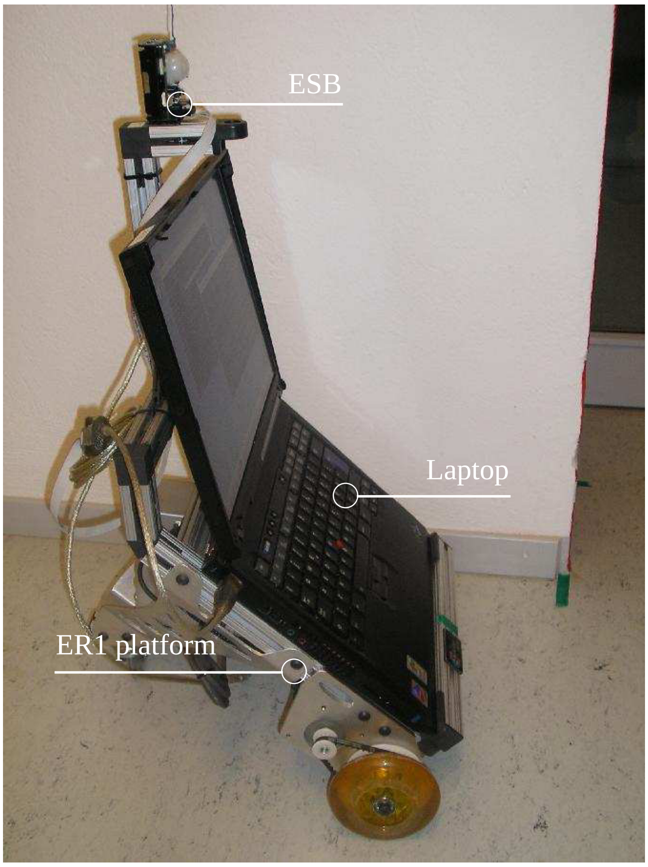


Figure 6.1: The robotic system used in our experiments

The ESB software is based on the ScOS operating system.[2] It collects sensor measurements every second and sends them to the laptop through its serial interface. Additionally, the ESB can send and receive radio messages to and from another ESB (connected to another robot).

The major part of the application software, and the negotiation implementation in particular, runs on the laptop. A robot device server called Player[3] has been installed on the laptop along with a specific driver for the ER1 to provide a con-

[2] ScatterWeb operating system

[3] `http://playerstage.sourceforge.net`

venient interface to the ER1 control electronics.[4] The laptop software performs the following duties:

- maintains the cognitive state of the robot (including its location relative to a map, its percepts, and the phases of negotiations);
- motion planning (to navigate the robot from one part of the environment to another);
- parsing messages from the ESB and formatting of ESB commands;
- odometry (i.e., estimating the current pose of the robot);
- motor control.

We explain details of the ESB sensor node and the ER1 robot platform below.

The Embedded Sensor Board

A picture of the ESB identifying most of its components is shown in Figure 6.2. The "brain" of the device is its MSP430F149 micro controller. It has a 60 kB program memory and 2 kB RAM. It also has an EEPROM memory for storing data such as configurations and routing tables.

The ESB provides two main functionalities: *communication* and *sensing*. Communication is performed by three means: radio, (wired) serial, and infrared. The radio communication is handled by a TR-1001 transceiver. Its operation frequency is 868 MHz and its transmission range is 0.1 to 300 meters in open space. The serial interface is shown in the bottom left part of the figure. The associated serial chip is right above it and is capable of transferring data at the maximum rate of 115.2 kb/s. The serial interface can also connect to a mobile phone. The ESB can also send visual and acoustic signals (i.e., one-way communication) through its three light emitting diodes (LEDs) and beeper, respectively.

There are five sensors available:

Movement A pyro-electric infrared sensor (PIR) that detects changes when a heat emitting source (such as a human being) passes its field of view. The PIR is covered by a Fresnel lens (not shown in Figure 6.2) which splits its field of view and makes it more sensitive to moving sources of infrared radiation.

Light intensity Responds to infrared light in the range from 800nm to 1100nm. The light intensity sensor, like the PIR sensor, is located under the Fresnel lens.

Temperature Measures temperatures from -55° C to $+125^\circ$ C, with a $\pm 2^\circ$ C accuracy.

[4] The ER1 driver was written by David Feil-Seifer.

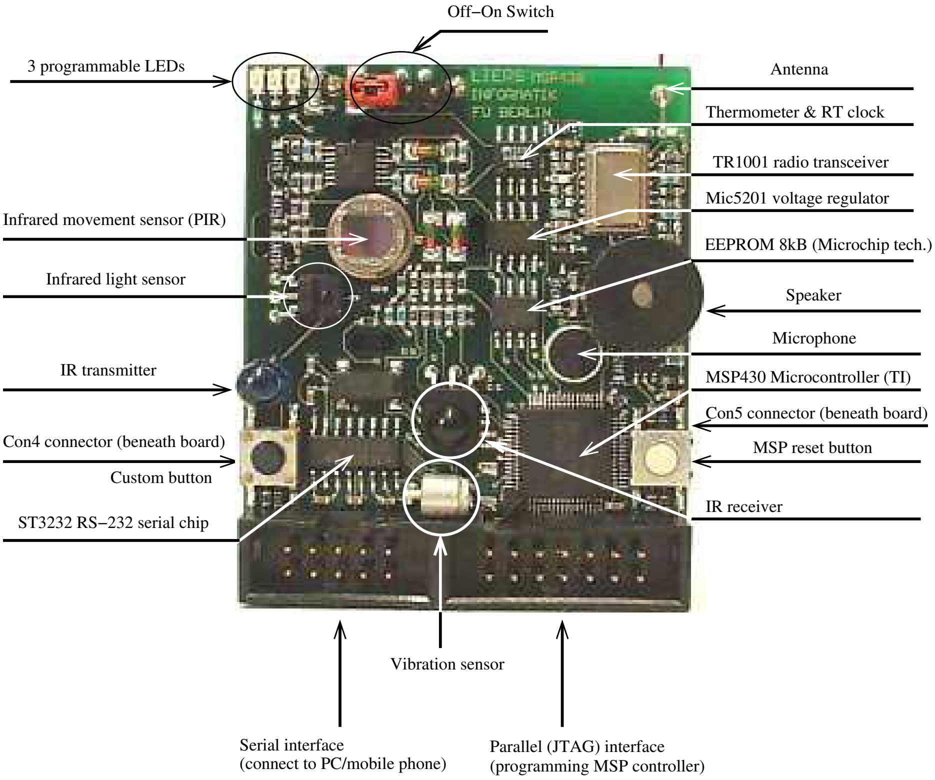

Off−On Switch
3 programmable LEDs
Antenna
Thermometer & RT clock
TR1001 radio transceiver
Mic5201 voltage regulator
Infrared movement sensor (PIR)
EEPROM 8kB (Microchip tech.)
Infrared light sensor
Speaker
Microphone
IR transmitter
MSP430 Microcontroller (TI)
Con5 connector (beneath board)
Con4 connector (beneath board)
MSP reset button
Custom button
IR receiver
ST3232 RS−232 serial chip
LIERS MSP430 INFORMATIK FU BERLIN
Vibration sensor
Serial interface
(connect to PC/mobile phone)
Parallel (JTAG) interface
(programming MSP controller)

Figure 6.2: An overview of the ESB device

Vibration Monitors vibrations (seismic effects) that the ESB might be subjected to.

Microphone Used to detect sound up to 120 dB.

The software running on the ESBs used in this experiment collects data and reports it once every second. To be able to show all the different kinds of measurements at the same time, the normalised instead of the absolute values are shown in Figure 6.3.

We should offer a few comments on the data shown in the figure; in this case, the ESB was placed in a room where a human was moving about. In the beginning of the time interval, we see a small reaction by the PIR sensor (the black curve). This indicates that the PIR sensor has detected slow motion or that the human is some meters away from the sensor. Notice also the reduction of the light intensity in the time steps between 16 and 25 seconds. This effect corresponds to the human occluding some light source. Hence, although the movement sensor does not react between 19 and 22 seconds, the robot might draw the conclusion that the human is still there, though just not moving.

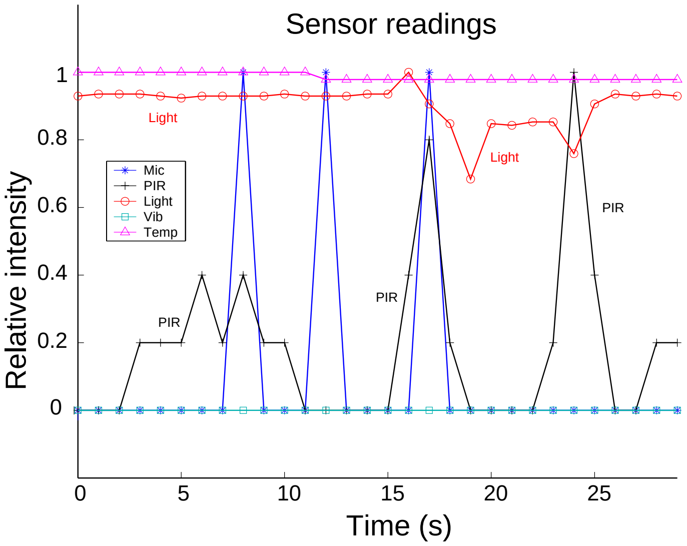


Figure 6.3: Data collection from an ESB

Detection Using the PIR Sensor

Out of the five sensors, the one we have found to provide the most useful information is the PIR movement sensor. In our initial tests, the PIR sensor turned out to be very reliable (hardly any false positives were noted), and has a limited range (unlike, e.g., the microphone which might pick up very remote sound sources).

The PIR sensor is sensitive to changes in the perceived infrared energy. If a deviation from the average is noted, a detection will be signalled. If it was not for the Fresnel lens, the PIR sensor would only detect someone (i.e., some infrared emitting source) entering and leaving the field of view of the sensor. The Fresnel lens fragments the field of view of the sensor so that a target might leave and enter the view repeatedly while moving in front of the sensor.

Theoretically, a PIR sensor might fail to detect a target for two reasons (if otherwise working correctly): the target is moving (i) too slow or (ii) too fast. If the target moves very slowly, the sensor will include it in the background radiation, and if it moves too fast there may be too little time for the sensor to notice the change in infrared emittance. The velocities and distances for which a target can be detected is called the *target velocity range*. Figure 6.4 shows the properties of a different PIR sensor, but illustrates this situation. If the target is in the shaded region (i.e., if it has a velocity and distance to the sensor that belongs to the region) it should be detected. If it is outside the region, e.g., if it is moving slowly at a distance or fast very close by, it might go undetected. In our practical tests, however, we did not manage to elude the PIR sensor at a distance of less than five meters.

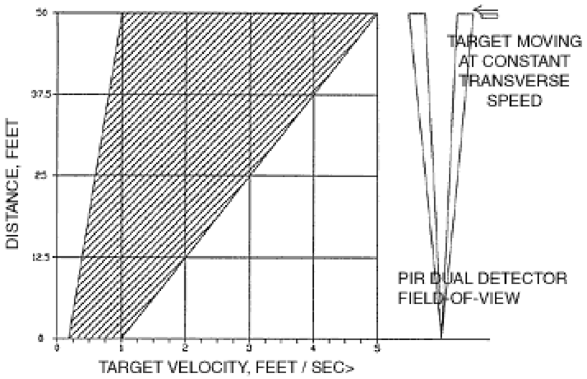


Figure 6.4: Target velocity range for a PIR sensor (by courtesy of Fuji & Co)

The ER1 Robotic Platform

The ER1 is a small (47cm W x 38cm H x 47cm D) mobile robotic platform. It carries a 12 V battery which allows about 2 hours of usage. The ER1 also carries a user laptop which could contain the manufacturer's software which provides a range of functionalities in combination with camera and microphone sensors. In our experiments, however, we do not make use of the manufacturer's sensors and software. Instead, we use the ESB device described in Section 6.2 and customised software.

6.3 Experiments

With the experiments, we want to show that the egalitarian coordination protocol described in Chapter 5 can be used in practise. The experiments also give examples on how the robots react to variations in the environment. In the first experiment, two robots of the type described in Section 6.2 coordinate their actions to explore a small office environment. Based on their negotiations, they explore different parts of the environment.

We only had two robots available for this experiment. In simulations, however, we have used three or more agents (see Section 5.6 on page 128).

The office environment is depicted in Figure 6.5. The map is overlaid with the topological map (represented by a graph) which is used by the robots to navigate in the physical environment. In the current implementation, the robots are “blind” while travelling between nodes, but they halt, collect and analyse sensor data when they reach a new node.

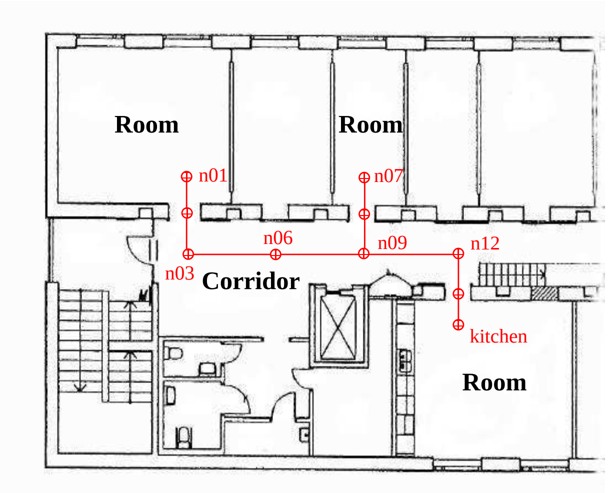


Figure 6.5: Map of the office environment overlaid with the topological map (with node names) used by the robots

Naturally, the division of the office space could have been performed off-line avoiding repeated negotiations. However, the robots’ ability to respond to dynamic events in the environment is more important. Hence, we compare the result in the first experiment to the situation in the second experiment with a suddenly appearing target.

In the second experiment, the initial configuration of the robots is the same as in the first experiment (i.e., they have the same starting poses) and the same coordination protocol. The appearing target is detected by one of the robots who, through a negotiation, manages to convince the other robot to assist it.

Both experiments end prematurely; the first experiment ends when the robots have jointly explored the whole office environment, and the second when both robots manage to detect the target. The reason for the sudden termination is that we instead focus on distinctive elements of the practical experiments, i.e., sensing, mobility, and communication (which were trivial in the simulation experiments in Chapter 5). If we would like to improve the robot set to work indefinitely, we could add components such as those developed for the simulations, e.g., the belief decay in Equation 5.8.

Coordination Details

In Chapter 5, we wrote about two levels of a negotiation protocol: the *organisational* and *negotiation* levels. The organisational level specifies what robots should negotiate and when. In our experiments, we only use two robots so both robots know what other robot to negotiate with at all times. Basically, the robots should only engage in negotiations when they have to, e.g., when one has detected something or when someone wants to enter a room (and wants to make sure that some other robot is covering the corridor). However, since the robots do not have appropriate sensor resources for detecting each other, the robots negotiate frequently to avoid collisions.

The robots bring a repertoire of six possible actions to a negotiation encounter: four movement actions (up, down, right, and left in the map), to stay in its current position, or to assist another robot. Every negotiation encounter results in a deal of joint actions being reached (unless the communication fails during the negotiation) and each robot makes a note that it is committed to perform its individual action. A commitment for a move action expires when the robot has reached the next node in the graph (corresponding to its committed action). A commitment for a robot a_i to assist another robot a_j is a long-term commitment that will not expire until a_i has visited the position of a_j and reported whether it can detect the target. As long as a commitment has not expired, a_i and a_j will not engage in another negotiation.

In practise, a negotiation between an initiating robot a_i and a responding robot a_j proceeds in the following way:

1. Robot a_i initiates the negotiation by addressing its (known) negotiation peer a_j. a_i sends its current position on the map along with its action alternatives and a randomly drawn value r between 0 and 255. The r-value is used to discriminate between equally suitable solutions and was explained on page 115 in Chapter 5. a_i sends this message repeatedly until a reply has been received (it will only send a limited number of messages before aborting, though).

2. Unless a_j is busy (e.g., moving) or unreachable, it will respond to the negotiation request from a_i by sending its own current map position and its currently available actions. a_j will send its reply repeatedly until it receives the utilities from a_i.

3. a_i receives the position and actions of a_j. a_i then constructs all possible deals (joint actions) and starts to evaluate their consequences. Since a_i has been told the position of a_j it can estimate the consequences of its own and a_j's actions. Some evaluation criteria are given in Table 6.1. a_i sends its utilities repeatedly to a_j until it responds with its utilities.

4. a_j receives the utilities from a_i and both a_i and a_j can find the egalitarian solution which they both implement.

Consequence	**Evaluation**
a_i and a_j collide	Assign minimal score
a_i and a_j both end up outside the corridor	Decrease total score
a_i stays in the current position while observing something	Increase total score
a_i decides to assist a_j while currently not observing the target itself	Increase total score
a_i ends up in a position which it has not previously explored	Increase total score
a_i ends up in a position which it has previously explored	Decrease total score
a_j ends up in a position which has previously been explored by a_i	Decrease total score

Table 6.1: Examples of criteria that a robot a_i is considering while evaluating joint action deals with a robot a_j.

Experiment 6.1: Coordinated Exploration

In the first experiment, the robots, a_1 and a_2, will not detect any target. However, they will still coordinate their actions to avoid collisions, to efficiently explore the common office environment, and to avoid leaving the corridor unattended.

The evolution of the exploration task is shown in the map in Figure 6.6 and details for all negotiations are given in Table 6.2. The table is divided in six negotiations ($n = 1, \ldots, 6$) and contains information about which robot initiates the negotiation, and what information is exchanged between the two agents. The utilities for both robots are shown and correspond to the possible deals (i.e., joint actions). If the actions of the initiating robot are $D_i = \{d_{i1}, \ldots\}$ and the actions of the responding robot $D_r = \{d_{r1}, \ldots\}$, then the deals (as well as the utilities) are ordered in the following way: $((d_{i1}, d_{r1}), (d_{i1}, d_{r2}), \ldots, (d_{i2}, d_{r1}), (d_{i2}, d_{r2}), \ldots)$. Hence, e.g., in the first negotiation ($n = 1$), $u_1\left((stay, down)\right) = 28$ and $u_2\left((stay, down)\right) = 32$.

In the second negotiation encounter, the division of the office space is in practise decided. The outcome is here that a_1 moves to the right in the corridor. The robots are just as comfortable deciding that a_2 should move to the left, but the selection of r (in this case 49) makes them select right. In the succeeding third negotiation, a_1 moves to the left as this is preferred by a_2 (which is already exploring the corridor to the right). Hence, the robots successfully manage to divide the office space between them. a_1 continues to node n01 and a_2 to node kitchen.

Figure 6.7 shows a few snapshots from the experiment.

Experiment 6.2: Cooperative Detection

In the second experiment, we will see how the result differs from Experiment 6.1 when a target enters the office environment. The second experiment has the exact

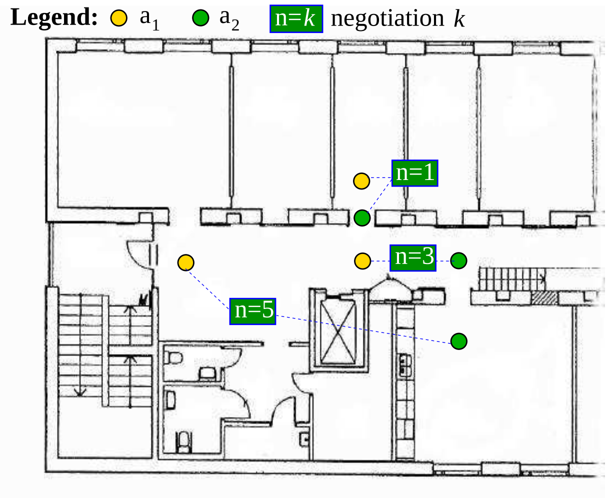


Figure 6.6: The evolution of Experiment 6.1

same initial configuration, and the first three negotiations have identical outcomes as in Experiment 6.1. The negotiation details are given in Table 6.3. The evolution of the experiment is depicted in Figure 6.8. In the second negotiation encounter, the outcome could have been different (i.e., a_2 could have explored the left part of the corridor instead of the right), but once again the selection of r decided the outcome.

Before the fourth negotiation encounter, a_1 (which at that time is located in node n06) makes a detection. The succeeding negotiation outcome is a_1 staying in node n06 and a_2 committing to go n06 to confirm the detection. The experiment is depicted in Figure 6.9. For the sake of the experiment, the target remained in node n06 and a_2 could confirm its presence.

6.4 Summary and Discussion

We successfully implemented the egalitarian coordination protocol, described in Chapter 5, on a set of two mobile robots (denoted a_1 and a_2) carrying radio communication and sensing devices. Details of the negotiations for two experiments are given. The sudden appearance of a target in front of a_1 in Experiment 6.2, resulting in the different outcomes of the fourth negotiations (for Experiment 6.1 and Experiment 6.2), shows how the implemented egalitarian negotiation adapts the joint action selection. In Experiment 6.1, a_2 continued to explore the unexplored space around node kitchen, but in Experiment 6.2 the benevolent disposition of a_2 urges it agree on a deal where it assists a_1.

A few issues concerning the implementation that differs from the simulations in Chapter 5 should be noted:

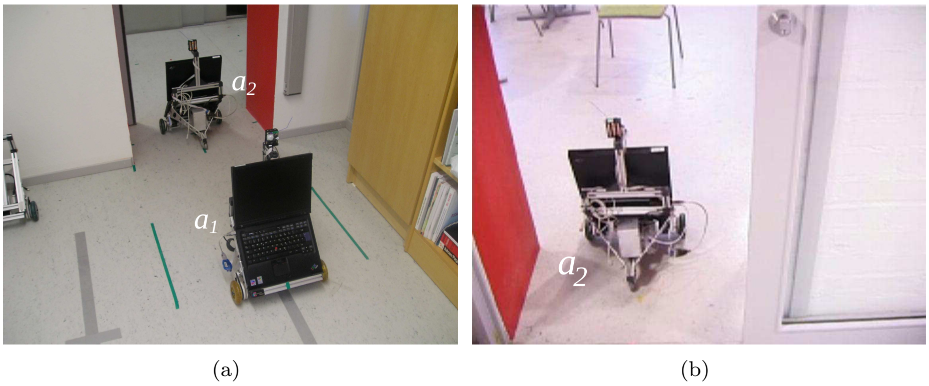
- The robots are “blind” while moving as the software is devoted to controlling the motors and no effort is spent on processing sensor data.

- Although no communication failures were experienced in this small experiment setting, it is an inherent challenge. In our current implementation, communication failures result in aborted negotiations. In that case, the robots that failed to communicate and coordinate their actions, reason about their actions without input from the other.

- The concurrency of the individual robots is another challenge. Preferably the robots should be able to sense, move and negotiate all at the same time. However, robots changing their own state while negotiating (e.g., by moving and sensing) might want to renegotiate if crucial observations are made. Since we already have the restriction that a robot can only negotiate and sense when it is not moving, the robots synchronise whenever a negotiation starts. This means that a robot that wants to negotiate has to wait for another to reach its destination and collect sensor data. This requirement reduces the flexibility of the robot set, but simplifies its implementation.

Robot events for Experiment 6.1
n = 1
a_1: sending request: p = n07 r = 134 a = [stay,down,assist]
a_2: sending reply: p = n08 a = [stay,down,up,assist]
a_1 and a_2 exchange utilities
u1 = [28,28,0,23,0,32,0,0,35,0,0,0]
u2 = [28,32,0,30,0,31,0,0,23,0,0,0]
Joint action selected: a_1 = down, a_2 = down
n = 2
a_2: sending request: p = n09 r = 49 a = [stay,up,right,left,assist]
a_1: sending reply: p = n08 a = [stay,down,up,assist]
u1 = [28,0,28,35,0,0,27,0,28,32,28,0,28,32,28,0,23,0,0,0]
u2 = [27,0,28,22,0,0,28,0,31,31,32,0,31,31,32,0,30,0,0,0]
Joint action selected: a_1 = down, a_2 = right
n = 3
a_1: sending request: p = n09 r = 197 a = [stay,up,right,left,assist]
a_2: sending reply: p = n12 a = [stay,down,left,assist]
u1 = [28,28,0,23,28,28,27,0,32,32,31,0,0,32,0,0,35,0,0,0]
u2 = [27,31,0,30,27,31,27,0,28,32,28,0,0,31,0,0,22,0,0,0]
Joint action selected: a_1 = left, a_2 = down
n = 4
a_2: sending request: p = n13 r = 197 a = [stay,down,up,assist]
a_1: sending reply: p = n06 a = [stay,right,left,assist]
u1 = [28,28,32,31,28,28,32,0,28,28,32,0,23,0,0,0]
u2 = [28,27,28,23,32,31,32,0,28,27,28,0,31,0,0,0]
Joint action selected: a_1 = left, a_2 = down
n = 5
a_1: sending request: p = n03 r = 115 a = [stay,right,up,assist]
a_2: sending reply: p = kitchen a = [stay,up,assist]
u1 = [28,28,23,28,28,0,32,32,0,31,0,0]
u2 = [28,29,32,28,29,0,28,30,0,23,0,0]
Joint action selected: a_1 = up, a_2 = up
n = 6
a_2: sending request: p = n13 r = 137 a = [stay,down,up,assist]
a_1: sending reply: p = n02 a = [stay,down,up,assist]
u1 = [28,28,32,31,28,28,27,0,28,28,32,0,23,0,0,0]
u2 = [28,28,28,23,28,28,23,0,29,29,29,0,32,0,0,0]
Joint action selected: a_1 = up, a_2 = up

Table 6.2: Robot events for Experiment 6.1



(a) (b)

Figure 6.7: a) The initial configuration of a_1 (at node n07) and a_2 (at node n08). b) a_2 entering the kitchen between negotiation four and five.

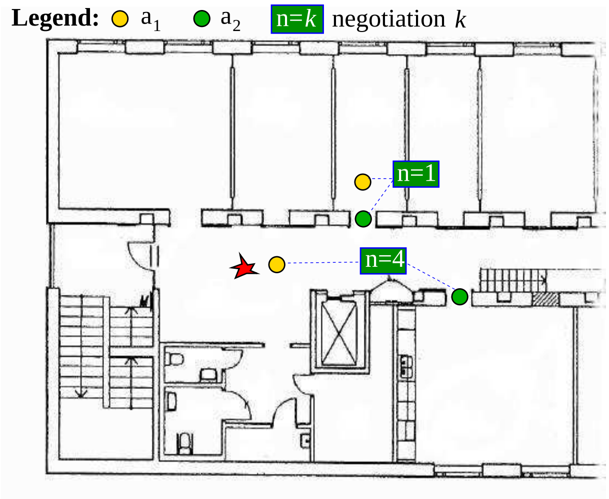
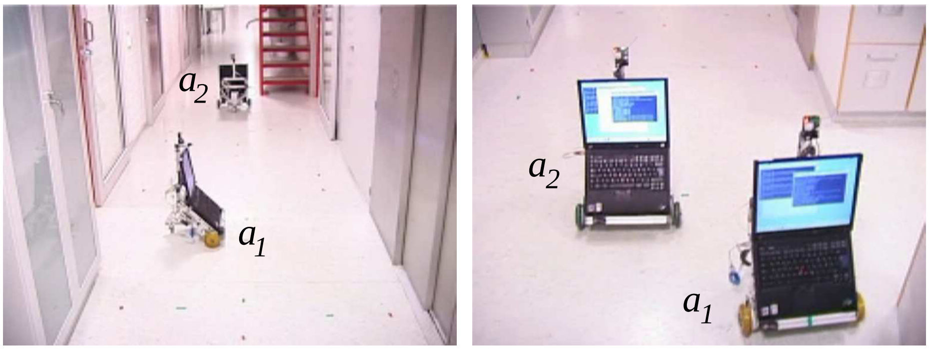


Figure 6.8: The evolution of Experiment 6.2

Robot events for Experiment 6.2
n = 1
a_1: sending request: p = n07 r = 123 a = [stay,down,assist]
a_2: sending reply: p = n08 a = [stay,down,up,assist]
a_1 and a_2 exchange utilities
u1 = [28,28,0,23,0,32,0,0,35,0,0,0]
u2 = [28,32,0,30,0,31,0,0,23,0,0,0]
Joint action selected: a_1 = down, a_2 = down
n = 2
a_2: sending request: p = n09 r = 87 a = [stay,up,right,left,assist]
a_1: sending reply: p = n08 a = [stay,down,up,assist]
u1 = [28,0,28,35,0,0,27,0,28,32,28,0,28,32,28,0,23,0,0,0]
u2 = [27,0,28,22,0,0,28,0,31,31,32,0,31,31,32,0,30,0,0,0]
Joint action selected: a_1 = down, a_2 = right
n = 3
a_1: sending request: p = n09 r = 228 a = [stay,up,left,right,assist]
a_2: sending reply: p = n12 a = [stay,down,left,assist]
u1 = [28,28,0,23,28,28,27,0,32,32,31,0,0,32,0,0,35,0,0,0]
u2 = [27,31,0,30,27,31,27,0,28,32,28,0,0,31,0,0,22,0,0,0]
Joint action selected: a_1 = left, a_2 = down
n = 4
a_2: sending request: p = n13 r = 191 a = [stay,down,up,assist]
a_1: sending reply: p = n06 a = [stay,right,left,assist]
u1 = [28,23,27,28,28,23,27,0,28,23,27,0,33,0,0,0]
u2 = [28,27,28,23,32,31,32,0,28,27,28,0,31,0,0,0]
Joint action selected: a_1 = stay, a_2 = assist

Table 6.3: Robot events for Experiment 6.2



(a) (b)

Figure 6.9: a) a_1 (at node n09) and a_2 (at node n12) jointly exploring the corridor. b) a_2 arriving at node n06 to verify the presence of a target.

Part IV

Summary

Chapter 7

Summary and Discussion

The presentation has focused on large-scale information acquisition and its relationship to data and information fusion. More specifically, our studies include:

- an interpretation of the data fusion model in an agent context (Chapter 2);
- a characterisation of large-scale information acquisition (LSIA) and perception management (Chapter 3);
- an address to the intricate connection between high-level information fusion artefacts and information acquisition (Chapter 4);
- a proposition of a recurrent coordination protocol to realize one of skills required for large-scale information acquisition, viz decentralised control, with both simulation and robotic experiments (Chapter 5 and Chapter 6).

We summarise these contributions more thoroughly in Section 7.1 and give directions for future work in Section 7.2.

7.1 Contributions

In this thesis, we introduce and set out to characterise the concept of large-scale information acquisition for data and information fusion. For historical reasons, we call a system performing data and information fusion activities a "data fusion system." The purpose of a data fusion system is to integrate (fuse) multiple pieces of data (typically from different sources) to acquire a better *understanding* of a system-relevant environment. A better understanding, in this case, manifests itself as, e.g., state estimates of higher accuracy (exploiting data redundancy) or resolved ambiguities (utilising complementary data).

As is discussed in Chapter 1, an interest in LSIA is justified by both the observed lack of research on a *holistic* view on the data fusion process, and the proliferation of network enabled sensing devices. We identify a number of properties that could be considered in the context of LSIA (which are frequently disregarded in the data and information fusion research field). The sensing resources used could be large in number, heterogeneous (i.e, provide different kinds of data), complex (have different modes or initiate autonomous activities to acquired the requested data), and distributed. Also, algorithms for LSIA, may have to deal with decentralised control (to robustly utilise multiple sensors) and multiple and varying objectives (many users of the data fusion system with objectives that change over time).

In the literature, a function that realises information acquisition is frequently denoted *sensor management*. We, however, believe that this term is somewhat limiting for LSIA, as it primarily suggests optimising the usage of the sensor devices themselves, rather than the value of the information ultimately acquired. Instead, in Chapter 2, we introduce a concept called *perception management*.

To succinctly and uniformly express control of resources, we first introduce the concept of (perception) *service*. A service, compared to a sensing resource, more explictly represents the available sensing opportunities of perception management. A service can involve both sensing resources and basic control of the resources and initial processing of acquired data. A service configuration space $\mathcal{C}$, representing all possible service configurations is presented, which can be used to reason about information acquisition.

From a literature survey focusing on the information aspect of information acquisition, we extract three different perception activities: *incorporation*, *monitoring* and *discernment*. Two aspects of perception activities also emerge: *facilitation* and *focus of attention*. The perhaps most distinguishing feature of perception management is its explicit duty to facilitate observations (e.g., avoiding using interfering sensors, and repositioning a camera for a better view), and to some extent managing the system's focus of attention (e.g., resolving information need into sensing actions).

Perception management also encourages an agent perspective on the data fusion process, where the data fusion process acts as a decision support for an overall system control. The data fusion process is all about *perceiving* and its perception management skill concerns *acting*, i.e., the possibly two most important characteristics of an agent. Apart from explictly inviting the wealth of agent theory research into the data and information fusion research, it also highlights that the resource usage of perception management is constrained by the overall control of the system.

To facilitate further research on LSIA, we propose a framework (in Chapter 4) that captures the aforementioned properties of LSIA. The framework suggests five information spaces where information need, information acquisition tasks, resources and services (which encapsulate resources) are represented.

We apply the framework in a context where we connect information need to information acquisition. In our case, information need is represented by a plan recognition process which estimates plans of some observed hostile *actors* in a military context. The plan recognition process is based on a dynamic Bayesian network, and the underlying state uncertainty (which is affected by observations) is represented and maintained by particle filters. Our implementation deals with many of the issues related to LSIA, e.g., high-level information need, multiple objectives, and heterogeneous sensors.

A few interesting issues came to our attention during the implementation:

- Our implementation is not decentralised; e.g., prioritisation of tasks and task-to-service assignment was all performed by a single processing node. Instead, decentralisation is dealt with in another chapter. The desire to decentralise the data fusion system raises some critical questions that remains to be answered, e.g., "Where (in the network) should data be fused?" One fascinating approach to this problem is offered by (Brännström et al., 2004) who allow a fusion agent to travel between nodes in a network distributing fusion results to any user who is interested in subscribing.

- Many optimisation problems need to consider multiple objectives. The information acquisition problem is no different. In our application, such objectives include the desires of those requesting information as well as the needs of the sensors (e.g., not to use sensors which are too far away from the targets). Hence, there is, at times, difficult to even express what an optimal solution might be. In our implementation, we integrate some of the objectives, but in end have two left. Currently, we leave it up to the user of the system to decide which one to prioritise. A tentative comparison between focusing on either of the two objectives is shown in the experiments section of Chapter 4. Approaches to managing multiple objectives is addressed by, e.g., Cohon (2003), Brynielsson and Wallenius (2003), Coello (1999) and Pirjanian (1998).

- In data and information fusion in general, and in plan recognition in particular, the value of sensing actions can only be estimated by evaluating the data fusion process on anticipated observations. This is problematic since it is typically costly to run the data fusion process over and over again on hypothetical observations (it is often considered too costly just to run it on the actual data). It might even be a pointless exercise as the data fusion system is evolving continuously. Hence, some approximation of the data fusion process or a simplified approach to control sensors is desired. In our simulations, we do the latter. We make the simplifying, and reasonable, assumption that the plan recognition of a hostile actor is likely to be improved by observing the particular actor. This might appear true in general, but is not since an improved state estimation might have no effect on the plan estimate.

  A second problem is that in a distributed context, the information need and the process that assigns tasks to sensors might be separated. The latter then requires some instructions on how to evaluate available sensing actions for a particular information acquisition task.

- Information need might not always lead to sensing actions. The information might already be available somewhere in the network. Thus, costly sensing actions might be avoided by careful information management.

- Long-term sensing planning is another interesting and under-researched topic. In a set of sensors (e.g., the one with ground soldiers and UAVs which we use in our simulations), the sensors may have different time delays before an observation can be made. The sensor manager then has to consider the uncertainty about when an observation can be made and when the sensing resource can become available again.

In the final chapters of the thesis, we address one of the properties of LSIA, *decentralised control*, in more detail. Decentralised control is an approach to deal with scalability and robustness in a network of controllable devices. We study the aspect of decentralised control which concerns *action coordination* of sensor platforms, i.e., managing dependencies between sensors such that conflicts are avoided and cooperation established whenever possible. The multi-robot example, we give, can be thought of as coordination within a single service (consisting of the set of robots). Important for large-scale information acquisition is also coordination between different services serving different objectives.

We assume that the multi-robot coordination is conducted in an antagonistic and physical environment. Those environment characteristics suggest that solutions with limited inter-agent communication and fast negotiation should be favoured.

We propose a coordination protocol that involves recurrent negotiations. Our approach to the negotiations is from a *bargaining theory* perspective. We identify shortcomings of the in computer science most commonly applied bargaining solution, the Nash solution. We demonstrate in simulation a problem instance where the Nash solution is inferior to an *egalitarian* one. However, we can not conclude that other solutions, such as the egalitarian solution dominates the Nash solution in general. They are preferable in different situations.

Furthermore, solutions from bargaining theory in economics can not (in general) be seamlessly transferred to applications in computer science. In the thesis, we mention, e.g., that the lottery concept used in bargaining theory to ensure convex and compact utility spaces does not represent suitable solutions in general for coordinating agents.

We have also implemented and tested the coordination protocol on set of mobile robots.

Finally, we believe that issues related to large-scale information acquisition, as presented in this thesis, will receive increasing attention in the near future.

7.2 Future Work

Although extensive, our approach to LSIA is not complete and many future directions can be envisioned. In the following sections, we give some interesting directions for future research for the agent-based data fusion model (presented in Chapter 2), our framework for large-scale information acquisition (Chapter 4), and the recurrent coordination protocol (Chapter 5), respectively.

The Agent-Based Data Fusion Model and Perception Management

As previously indicated, we do not claim that the model or domain map of information acquisition for data and information fusion presented in Chapter 2 is complete. Future work in this field could consider some of the following directions:

Learning and coevolution In environments where a data fusion process 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 by, e.g., Mitchell, 1997; Kaelbling et al., 1996).

Learning is especially important in environments inhabited by antagonistic intelligent adversaries. In such cases, also the adversaries may be learning and adapting their behaviour to their observations. The process of agents learning from each other is called *coevolution*. Their adaption imposes constraints on the decision-making for sensing actions of the data fusion process. Important to learning is the representation of the mental state of other agents. *Perceptual state* described by Steinberg and Bowman (2001) and *recursive modelling method* by Gmytrasiewicz and Durfee (2000) are used for that purpose.

Perception services Managing services instead of resources appears to have some important advantages. 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. We address this issue to some extent in Chapter 4.

Prioritisation of information acquisition 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* by Hintz and McIntyre (1999). Internal priorities arise from the available resources, e.g., a task with high external priority may get a low internal priority because the available resources cannot satisfy the task. The notions of external and internal priorities and ways of integrating them require further exploration. We begin to address this issue in Chapter 4.

Information acquisition tasks A perception manager may respond to stimuli of the other levels of a data fusion process, sensing resources and other agents, as explained in Section 2.3. 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 A perception manager 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 the work by Budenske and Gini (1997) who describe the process of *explication* which denotes the transformation from tasks to utilisation of sensing resources. It should also be explored how resources can be used for treating several tasks simultaneously. We begin to address this issue in Chapter 4.

Data mining *Data mining* techniques should be considered. Hand et al. (2001) define data mining 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 data fusion process running data mining algorithms on acquired and stored data to create and refine rules and models, supporting the further work of the data fusion process. This issue has already been discussed thoroughly by Steinberg (1999), but should explicitly enter the agent-based data fusion model.

Control of data fusion methods One part of JDL Level 4 fusion, process refinement (see Section 2.2), is controlling and, possibly, exchanging methods for fusion of information. The properties of this activity should be investigated.

Management process models Process models of information acquisition (i.e., sensor management) of other works (e.g., Grahn et al., 2005; Xiong & Svensson, 2002; Ng & Ng, 2000) should be considered, and possibly integrated with this study.

The Framework

Our implementation which connects high-level information (plan estimates) need to information acquisition, in Chapter 4, includes some of the ideas presented by our large-scale information acquisition framework. More issues and details could of course be considered. One of the most important issues is decentralised control (which we instead deal with in Chapter 5) which will have a profound effect on how information is spread in the system and how service configurations are selected.

Another issue is the set of plan estimate distributions that is acquired from the particle filter state uncertainty representation. Currently, we select a single representative distribution from the set. However, we see the opportunity to retain more information from the set of distributions; perhaps using the theory of robust Bayesian analysis.

The delicate interplay between the plan recognition process and the particle filter also has to be studied closely. Currently, the particle filter transition model sometimes causes the plan recognition process to overestimate the threat posed by an actor.

The Coordination Protocol

We identify some challenges with our negotiation protocol that is presented and tested in Chapter 5 and Chapter 6. One is to detect potential conflicts and decide when and with whom to negotiate with. In our simulations, the agents constantly engage in negotiations. This approach is talkative and might be inefficient for some applications. In the same simulations, an agent chooses its negotiation partners by their geographical distance. Other kinds of distance measures to decide which "neighbours" to negotiate with could be evaluated for different applications. In any case, the process of deciding which agents to negotiate with incurs a computational overhead and is difficult in general.

Another challenge is the rigidness of bilateral negotiations in multi-agent environments. Once a negotiation is complete, the involved agents are committed to perform the joint action that the negotiation yielded. Such a commitment constrains an agent in future negotiations until the joint action is completed and the commitment dismissed. Hence, commitments should ideally only be made when absolutely necessary. A relaxation could perhaps be to agree on a set of individual actions, where there is perhaps a most preferred one and the others acceptable for both agents. That would give the agents a broader repertoire of possible actions in further negotiations. The possibility to break commitments and consequences should also be considered (consider, e.g., the situation where two agents have made commitments with other agents and where their individual commitments result in them causing each other severe damage).

A suitable direction for future research is to compare and characterise different bargaining solutions. The selection of bargaining solution could possibly be the duty of some external agent responsible for the set of coordinating sensors.

Part V

Appendices

Appendix A

Sensing Resource Properties

When designing a perception management function, it is useful to ascribe properties to sensing resources or services to make them amenable to formal mathematical treatment. 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 further described by Hall (1992, Ch. 1).

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. Two views of sensors are discussed by Blackman and Popoli (1999, pp. 978): *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., human beings.[1]

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.

Provocative versus Non-provocative Perception Services

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 (Ng & Ng, 2000; Blackman & Popoli, 1999). There are two reasons for this distinction: first, control of an active sensor is generally more versatile and offers the ability to "provoke" the environment for richer information acquisition, 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.

Notice that the second reason mentioned in the previous paragraph is indistinct 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 *non-provocative 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 A.1(a)) and, correspondingly, a *provocative perception service.* 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 "footprints" or clues in the environment. In problem domains where hostile agents can interpret the footprint and take advantage of it (e.g., in command and control applications), it is essential to characterise and estimate such footprints.

A perception manager using active perception-resources should weigh the advantages of using the resources against the risk of being detected.

Admittedly, the conventional distinction between active and passive sensors is widely accepted, and, hence, we settle for this remark and don't pursue this idea any further.

Control Space Constraints

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

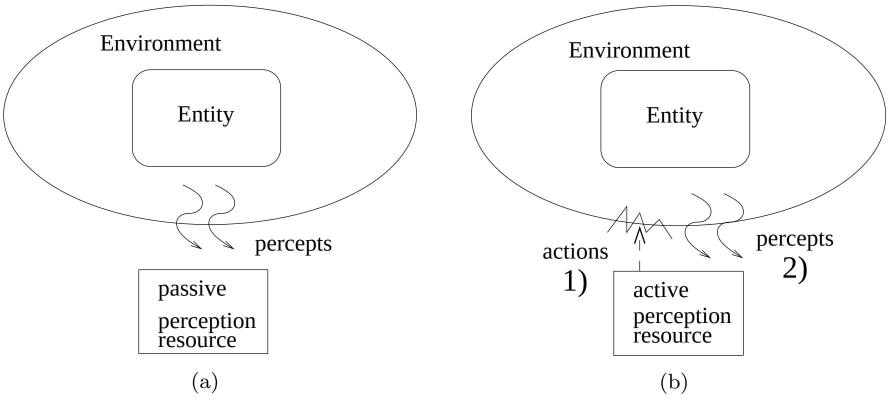


Figure A.1: (a) Usage of passive perception-resource (b) Usage of active perception-resource

Whereas the previously mentioned properties are static, i.e., belonging to the structure of a sensor, control space constraints describes objectives that must be considered while managing the resources. For instance, in some works, one wants to minimise energy consumption (Perillo & Heinzelman, 2003) or the number of sensors transmitting observations (Kalandros & Pao, 1998). We can often express control space constraints as unsuitable regions of the control space (cf to *configuration space* for motion planning in robotics, Latombe, 1993).

Availability and Redeployability

Perception resources are not always available to 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 objectives 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, e.g., an ordinary multi-meter which can not both measure current and voltage simultaneously);

- 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 objectives control.

We imagine that some resources are inexpensive and used only once, perhaps, e.g., sonobuoys dropped for tracking a target (Svensson et al., 1997). 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 information acquisition task at hand.[2] 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 summary 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 an additional cost involved. Some properties that might affect the value of using a resource are *reliability*, *condition* and *response time*.

Reliability

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

Perception resources may fail, either temporarily or permanently. Furthermore, sometimes a sensor might be sending 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 can be trusted. A perception manager could monitor the status of the sensors, and use unreliable sensors less frequently or not at all. Information source reliability is modelled by, e.g., Haenni and Hartmann (2004).

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.

Response Time

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

Appendix B

Systems of Perception Resources

In Appendix A, we discuss 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 data fusion process 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 and Heterogeneous Resource Sets

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 information acquisition 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 include the efforts by Penny and Williams (2000) and Nash (1977).

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 perception management makes an intelligent selection as task allocation becomes more tricky (Cao et al., 1997). This task allocation problem has mostly been dealt with in the field of distributed artificial intelligence.

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 to improve the performance of the information acquisition (this type of cooperation is called *cueing*). A few situations where such aid is useful are described by Waltz and Llinas (1990, Ch. 5):

- 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;[1]

- 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 by Durrant-Whyte (1988). There, sensors are directly dependent on observations of other sensors to form observations of their own.

[1] Interchange of entities between sensing resources is called “hand-off”.

Appendix C

Details of Related Architectures

We briefly listed some architectures related to information acquisition in Table 4.1 and how they relate to the functions and spaces of the large-scale information acquisition framework (LSIAf). In this appendix, we make a detailed description of these architectures, which are

- Selected agent architectures
- $\mathcal{M}/\mu$ Architecture
- Data Fusion and Resource Management Dual Node Network
- Goal Lattices
- Multi Platform-Based Architecture
- Active Sensor Network

We find that all of them can be expressed (at least to some extent) in the terms of our LSIAf.

Agent Architectures

There are several reasons to study agent architectures when investigating the LSIAf. First of all, we present perception management and information acquisition from an agent perspective (the agent context was described in Section 2.1) hoping to learn from the extensive field of agent theory. We can also choose to think of the LSIAf as an agent framework (it has objectives represented by the task space and it can sense and act with the members of the resource space). As far as we know, there are no other agent frameworks for information gathering that decouples information need from sensing resources and that emphasises the aspects stressed here, e.g., multiple objectives and distributed sensing resources.

It might also be useful to think of different components of an implemented framework as agents (see, e.g., Chapter 5, where we consider sensor platforms to be agents and devise a coordination protocol).

The field of agent theory has spawned a lot of architectures of various flavours. The purpose of the architectures is mainly to convey a common vocabulary (ontology) and to emphasise certain aspects of agent reasoning and behaviour. These architectures frequently also specify inter-agent communication protocols and strategies for reasoning and, ultimately, behaviour and decision making.

Many agent architectures are based on the BDI framework where the acronym is the first letters in the words *beliefs*, *desires*, and *intentions*. BDI concerns the internal representation of an agent and originates from a psychological model that considers behaviour to be a consequence of interactions between these three so called attitudes. Beliefs represent the information base an agent has about its environment. Desires are goals or options the agent considers to be feasible and desirable. Intentions are the subset of desires the agent is currently committed to perform. BDI architectures concern the representation, update and process of the attitudes.

An implemented BDI-based architecture is dMARS (d'Inverno et al., 1997) which has its roots in the procedural reasoning system from the late 1980s. It extends the BDI components with a *plan library* and an *interpreter function*. The library contains plans or recipes that the interpreter can apply to achieve its intentions. A plan contains an *invocation condition*, a *pre-condition* and a *body*. The invocation condition specifies the circumstances under which the plan should be considered. An explanatory example is given by d'Inverno et al.: for a plan `make-tea` the invocation condition might be `thirsty`. A pre-condition specifies the circumstances under which the plan could be executed. Once again, for the `make-tea` plan a pre-condition might be `have tea-bags`. The body contains a sequence of actions and subgoals that have to be fulfilled to accomplish the plan. An action might be `add water to cup` and a subgoal might be `get boiling water`. A subgoal typically results in another plan being invoked.

Compared to LSIAf the interpreter of dMARS, which is the active part of the architecture, implements task management and allocation scheme. The interpreter generates tasks (desires) by analysing percepts fulfilling the duties of task management. It, furthermore, selects some prioritised tasks (intentions) and execute plans to deal with them, i.e., completing the duty of allocation scheme. Note that external objectives could be represented as percepts in the dMARS architecture.

The GPGP/TÆMS (Generalised partial global planning/Task analysis, environment modelling, and simulation) is another architecture (Lesser et al., 2004) that was introduced in the early 1990s. GPGP is an agent coordination mechanism framework[1] and the associated TÆMS offers a hierarchical task network formalism. The formalism represents tasks (with properties such as quality, duration, and deadlines) and relationships between tasks in multi-agent systems. GPGP/TÆMS

[1] We discuss coordination further in Chapter 5.

specifies, among other things, the scheduling of tasks, and the inter-agent coordination that might result in re-scheduling of tasks whenever dependencies are detected. One interesting feature is that GPGP/TÆMS also allows the results of tasks to be non-deterministic.

From a LSIAf perspective, GPGP/TÆMS contributes to the task space and management parts (dealing with relationships between tasks) and the allocation scheme with the scheduling of tasks.

Another notable agent architecture is the OAA (Open Agent Architecture) by SRI (Martin et al., 1999), first presented in the mid 1990s, that has been used in various practical applications (e.g., multi-robot systems and, recently, in a support tool onboard a space shuttle). The architecture concerns communication and coordination aspects of agent interaction rather than the internal representation of an agent. OAA, furthermore, provides a framework with various agent roles. An agent typically provides a service (i.e., assumes the role of being a service provider) for actions (e.g., to send an E-mail if it is an E-mail service) or data (acting as a database), but also acts as a client when using other services. Facilitators are a special kind of agent which receives and maintains notifications of services and responds to requests for data or actions by alerting the appropriate services agents. Hence, the facilitators are an example of an allocation scheme function.

For further reading, a detailed comparison of some agent systems is provided by Ricordel and Demazeau (2000).

The $\mathcal{M}/\mu$ Architecture

The Macro/micro architecture explained by Blackman and Popoli (1999, Ch. 15), in their comprehensive target tracking handbook, is a hierarchical sensor management architecture with two distinct layers. It distinguishes two levels of functionality: the macro level which manages the overall information need and coordination of sensors (involving, e.g., distribution of tasks to sensors), and the micro level which deals with how a given particular task is best accomplished by a particular sensor. The authors suggest that the sensor management function is divided into a centralised macro sensor manager (that enjoys the benefits of global information and coherent sensor control) and several sensor located micro sensor managers.

LSIAf effectively encompasses the Macro/micro architecture. The responsibilities of the "macro manager" (e.g., task prioritisation and scheduling) belongs to the task management and allocation scheme functions of LSIAf. The micro managers can be considered to be services (that encapsulates both sensors and tailored control processes) that the macro manager accesses. The sensor located services (i.e., micro managers) naturally makes sure that the underlying sensing resources perform the assigned task. Hence, the micro managers perform the work of the service management and resource deployment function.

Whereas the Macro/micro architecture is designed to support the common situation that a set of sensors on a single platform has to be managed, LSIAf is less restrained (to encompass also, e.g., decentralised control).

Data Fusion and Resource Management Dual Node Network

The hierarchical *data fusion and resource management dual node network* (FMDNN) by Bowman and Steinberg (2001) is an architecture that tightly integrates data fusion and resource management (see Section 2.3 about resource management). The research concerning FMDNN covers both the concepts of internal activities in network nodes and interaction between nodes, and as well as how to engineer the network.

The resource management part of the network implements information acquisition, but is also responsible for all other system resources, such as defence and attack measures in a military system. The FMDNN, hence, acknowledges the dependence (and possibly conflicts), that we point out in Section 2.1, between activities for information acquisition and other system activities.

The hierarchical network allows high-level goals to be decomposed into concrete actions through layers of management nodes. A management node typically implements the following functions: develop candidate response plans to serve higher-level goals, evaluation and selection of plans, and the generation of the necessary control commands to implement the selected plan(s). The response plans might include sub-goals which are dealt with by lower-level nodes assignment of resources.

Task management for FMDNN would translate information need into goals. Furthermore, the FMDNN integrates the task management, allocation scheme, and service management. The allocation scheme appears distributed among the nodes of the network. It divides goals into sub-goals in at some level in the network, goals which it has to deal with (possibly further decomposition) at a lower level of the network. The services are the management nodes (and resources) that are allocated to goals and sub-goals.

Goal Lattices

Hintz and McIntyre (1999) have presented the idea of *goal lattice* for sensor management in a number of publications (see the survey in Section 3.10 for details). The goal lattice provides a formalism to hierarchically represent goals (from abstract mission-goals down to concrete sensing actions). This idea explicitly introduces mission goals (or system objectives) to sensor management and allows goal priorities to be changed dynamically. Recently, the concept of goal lattices spanning over parts of distributed systems has been proposed (Hintz, 2004).

The work of Hintz and McIntyre for information acquisition extends beyond the goal lattice, though. The work also includes the details of a sensor scheduling algorithm that can exploit the dynamic task prioritisation of the goal lattice (Zhang & Hintz, 1996) and a comprehensive sensor management architecture (Schaefer & Hintz, 2000) that subsumes both the scheduler and the goal lattice.

In the LSIAf, the goal lattice methodology can be exploited for task management and the scheduler for the allocation scheme function.

Multi Platform-Based Architecture

In a series of articles, Strömberg et al. present their *multi platform-based sensor management architecture* which utilises ideas from both agent theory and the $\mathcal{M}/\mu$ architecture (see, e.g., Strömberg et al., 2002; Strömberg & Lantz, 2003). The fundamental setup is that of a set of sensor-equipped platforms interacting to serve tasks that appear on the platforms. A task is a process that runs on a platform and has properties such as *priority* (which might be continously updated) and *creator id*. A task might have been initiated by the operator of the platform; it might have been initiated by an operator on another platform; or by some process or other task running on the local platform.

The $\mathcal{M}$-level management of a platform implements a market metaphor where the tasks act as consumers and platform sensors as producers. The purchasing power of a task agent is relative to the assigned priority of the task. Given a new task, a sensor is selected considering, e.g., the available capacity and the expected information gain of each sensor. The μ-level management consists of scheduling sensing actions to serve assigned tasks.

Some sensing actions are best handled automatically, rather than by the platform operator, since they concern activities in the range of milliseconds. The operator is, instead, offered a high-level influence over the sensors by assigning behaviours to the sensor system, so called *sensor management policies*. The operator should, e.g., have the opportunity to restrict the usage of some of its active sensors as they might reveal the presence and location of the platform to an adversary. Task requests (both internal and external) might be rejected if they are inconsistent with the current policy.

The task origin space is populated by the operators of the platforms and the functions that are running on the platforms. Tasks, that occupy the task space, appear on platforms if they comply with the constraints of the platform. The service and resource spaces merge into one, and the service management maintains information about the load of the sensors. Finally, the allocation scheme assigns tasks to sensors and performs the scheduling.

Active Sensor Network

Durrant-Whyte has been doing research on decentralised fusion and control for about two decades. Recent research concerns an architecture called Active Sensor Network (ASN, Makarenko & Durrant-Whyte, 2004; Makarenko, 2004; Makarenko et al., 2004). The application domain is somewhat similar to that of the platform-based sensor management by Strömberg et al. in that it involves multiple cooperative mobile platforms. However, the ASN architecture has a stronger assumption about the objective of the team of platforms. The general objective is the collaborative monitoring of some dynamic phenomenon in the environment. A common understanding about the phenomenon is achieved through a decentralised data fusion scheme.

Two different approaches to team decision making has been proposed so far. In one approach, the platforms make decisions based only on the common understanding of the environment, and in the other approach, platform decisions are a result of negotiations.

From the perspective of LSIAf, the task origin space has a single objective and that is to maintain (and improve if necessary) an understanding of the state of the mission-relevant phenomenon. Allocation scheme is expressed through the decision-making approaches explained above.

Appendix D

The Pareto Frontier

Algorithm D.1 describes how the Pareto frontier $\Delta^{\pi}_{\mathcal{A}}$ of set of deals $\Delta_{\mathcal{A}}$, for a set of agents $\mathcal{A}$, can be found in a straightforward way.[1] The incremental algorithm has a time complexity of $O(|\mathcal{A}|\,|\Delta_{\mathcal{A}}|^2)$ or, equivalently, $O(|\mathcal{A}|\,D_{\max}^{2|\mathcal{A}|})$. $D_{\max}$ is the maximum number of actions of any agent at any time and $|\Delta_{\mathcal{A}}| = D_{\max}^{|\mathcal{A}|}$. Other algorithms, e.g., the Simple Cull and some more efficient on large problems, can be found in the literature (see, e.g., Yukish, 2004).

To define the concept of Pareto frontier, we first define *dominance* and *Pareto optimality.* For convenience, let the joint binary preference relation $\succ$ for bargaining agents $\mathcal{A}$ with utility functions $\mathcal{P}_{\mathcal{A}} = (u_i)_{i=1}^{|\mathcal{A}|}$ be defined in the following way:

$$\delta_1 \succ \delta_2 \iff \bigwedge_{i=1}^{|\mathcal{A}|} u_i(\delta_1) \geq u_i(\delta_2), \quad \delta_1, \delta_2 \in \Delta_{\mathcal{A}}, \tag{D.1}$$

where at least one of the inequalities is strict, i.e., $\exists i \ u_i(\delta_1) > u_i(\delta_2)$. For the following discussion, it is important to notice that the relation $\succ$ is *transitive.*[2]

Now, a deal δ_1 dominates another δ_2 if $\delta_1 \succ \delta_2$. Furthermore, a deal δ is said to be Pareto optimal if there is no other deal $\delta' \in \Delta_{\mathcal{A}}$ such that δ' dominates δ. The Pareto frontier $\Delta^{\pi}_{\mathcal{A}}$ contains all Pareto optimal deals of $\Delta_{\mathcal{A}}$.

[1] We use the notation from Chapter 5.

[2] The transitivity property implies that $\delta_1 \succ \delta_2 \wedge \delta_2 \succ \delta_3 \rightarrow \delta_1 \succ \delta_3$. Tsoukiàs and Vincke (1992) provide a succinct overview of binary relation properties.

Algorithm D.1: The incremental Pareto frontier algorithm

```
FIND__PARETO__FRONTIER($\Delta_{\mathcal{A}}, \mathcal{P}_{\mathcal{A}}$)
Input: Deals $\Delta_{\mathcal{A}} = \left\{\delta_1, \ldots, \delta_{|\Delta_{\mathcal{A}}|}\right\}$, Preferences $\mathcal{P}_{\mathcal{A}} = (u_i)_{i=1}^{|\mathcal{A}|}$
Output: The Pareto frontier $\Delta_{\mathcal{A}}^{\pi}$ of $\Delta_{\mathcal{A}}$
(1)    if $|\Delta_{\mathcal{A}}| = 0$
(2)        return $\emptyset$
(3)    $\Delta_{\mathcal{A}}^{\pi} \leftarrow \{\delta_1\}$
(4)    foreach $\delta \in \Delta_{\mathcal{A}} \setminus \delta_1$
(5)        $dominated \leftarrow$ false
(6)        copy of $\Delta_{\mathcal{A}}^{\pi} \leftarrow \Delta_{\mathcal{A}}^{\pi}$
(7)        foreach $\delta^{\pi} \in$ copy of $\Delta_{\mathcal{A}}^{\pi}$
(8)            if $\delta^{\pi} \succ \delta$
(9)                $dominated \leftarrow$ true
(10)               break
(11)           else if $\delta \succ \delta^{\pi}$
(12)               $\Delta_{\mathcal{A}}^{\pi} \leftarrow \Delta_{\mathcal{A}}^{\pi} \setminus \delta^{\pi}$
(13)       if $dominated =$ true
(14)           continue
(15)       $\Delta_{\mathcal{A}}^{\pi} \leftarrow \Delta_{\mathcal{A}}^{\pi} \cup \delta$
```

The first two lines of Algorithm D.1 check for the extreme case that there are not any deals at all. If so, the algorithm returns the empty set. Otherwise, the Pareto frontier is built incrementally based on the deals the algorithm has seen so far. In line three, naturally the first element $\delta_1 \in \Delta_{\mathcal{A}}$ is added to the Pareto frontier since it is undominated by all (actually none) other elements seen so far. Lines four to fifteen iterate over all remaining elements of $\delta \in \Delta_{\mathcal{A}}$. For each δ there are two possibilities: it is dominated by some or none of the elements of the current Pareto frontier. A copy of the current Pareto frontier is made in Line six, since the actual Pareto frontier might be changed (in Line twelve) during the course of the algorithm. Line eight handles the first case; if δ is dominated by any element of the Pareto frontier δ^{π} then it can not, by definition, belong to the Pareto frontier. δ can not remove any current member of $\Delta_{\mathcal{A}}^{\pi}$ either since, if there existed such a dominated deal $\delta^{\pi\prime} \in \Delta_{\mathcal{A}}^{\pi}$ (i.e., $\delta \succ \delta^{\pi\prime}$), then $\delta^{\pi} \succ \delta^{\pi\prime}$ (due to transitivity). However, then the assumption that $\delta^{\pi\prime}$ belongs to the Pareto frontier has been refuted, and, hence, no deal of $\Delta_{\mathcal{A}}^{\pi}$ can be dominated by δ.

The second case is handled in the following lines. In lines eleven and twelve, deals in the current Pareto frontier that are dominated by δ are removed. As δ now is undominated, it is added to the current Pareto frontier in line thirteen.

Now we derive the worst-case and best-case complexities. The worst-case running time occurs when all deals belong to the Pareto frontier. Then each δ has to be compared to a growing number of elements in $\Delta_{\mathcal{A}}^{\pi}$. The cardinality of $\Delta_{\mathcal{A}}^{\pi}$ is, in the worst case, as great as the number of elements that have been considered so far. Note that each deal comparison with the preference relation (Equation D.1) requires $|\mathcal{A}|$ utility comparisons. We end up with the following expression for the worst case running time:

$$\begin{aligned}\textstyle\sum_{i=1}^{|\Delta_\mathcal{A}|-1} i|\mathcal{A}| &= |\mathcal{A}|\sum_{i=1}^{|\Delta_\mathcal{A}|-1} i = \left\{\text{since } \sum_{i=1}^{n} i = \frac{n(n+1)}{2}\right\} = \\ &= |\mathcal{A}|\frac{(|\Delta_\mathcal{A}|-1)|\Delta_\mathcal{A}|}{2} = O(|\mathcal{A}|\,|\Delta_\mathcal{A}|^2).\end{aligned} \tag{D.2}$$

In the best case, no deal (except for the first) is put into the Pareto frontier. This means the for each δ only one comparison is made, i.e., to δ_1. Still $|\mathcal{A}|$ utility comparisons have to be made in order to determine that δ_1 dominates a δ. The best case running time becomes:

$$\sum_{i=1}^{|\Delta_\mathcal{A}|-1} |\mathcal{A}| = |\mathcal{A}| \sum_{i=1}^{|\Delta_\mathcal{A}|-1} 1 = |\mathcal{A}|\,(|\Delta_\mathcal{A}|-1) = \Omega(|\mathcal{A}|\,|\Delta_\mathcal{A}|). \tag{D.3}$$

Appendix E

Glossary

Accessible A service is accessible to a perception manager if the manager is question has been designed to reason about the service and configure it. (See *available* and *feasible*)

Activity A process that a system uses to support or realise information acquisition.

Actor In this thesis (to avoid confusion), an actor is defined to be an *agent* that is observed by the data fusion system we are considering. Other definitions exist in the agent literature. (See *agent*)

Agent Something (in our case typically a process) that can perceive and act. (See *actor*)

Available A service is available to a perception manager if the resources necessary to realise the service are currently available.

Configuration space (SCS) The service configuration space $\mathcal{C}$ is the mixed space (a mixture of discrete and continuous dimensions) representing the complete space of options for information acquisition.

Cooperation A set of agents cooperate if they coordinate their actions to achieve a common goal. (See *coordination*)

Coordination Various alternative definitions can be found in the multi-robot literature. Here, we use a definition (similar to Huhns & Stephens, 1999, pp. 83) that says that coordination is a state or process of a set of agents that makes agents select their individual actions with respect to the other agents in the set so that, e.g., resource conflicts are avoided.

Data The contents (e.g., numeric or symbolic) of an measurements. (See *observation*, *measurement*, and *percept*)

Data fusion The process of combining data or information to estimate or predict entity states (Steinberg & Bowman, 2001). Note that data fusion is by some authors said to encompass the activities of information fusion. (See *information fusion*)

Data fusion process (DFP) A process whose main purpose is to synergetically combine data from different sources (i.e., to perform data fusion). Apart from "pure" data and information fusion, it is also concerned with all activities that facilitate data fusion, e.g., data association, management of sensing resources, etc. (See *data fusion* and *information fusion*)

Data incest The result of repeated use of identical information in the fusion process (McLaughlin et al., 2004).

Environment The environment is the source of all observations (possibly except for observations of internal states) of an agent. Depending on the agent's problem domain and its available sensing resources, a problem-related subset of the environment will be considered to have different properties (those properties are listed in Section 2.1). (See *environment representation*)

Environment representation (ER) The environment representation (introduced in Section 3.4) is a computational data structure (or structures) maintained by an agent that describes the agent's environment is a way that it can interpret and reason about. (See *agent* and *environment*)

Farsighted A sensor management process is farsighted if does not just consider the reward of a single sensing action but a sequence of actions (Nedich et al., 2005). (See *myopic*)

Feasible A service is feasible for a perception manager if certain constraints originating, e.g., from the manager's time and accuracy requirements on observations, and from the service's ability to serve the manager without causing damage to itself, are met. (See *accessible* and *available*)

Information We consider a piece of information to be data extended or replaced with relationships to the context of the observed situation. For instance, a sequence of observations (data) may form the information entity of a track. (See *data*)

Information acquisition The skill of a system to actively perceive its environment. Perception is typically achieved by controlling available sensing resources. (See *large-scale information acquisition* and *environment*)

Information fusion Information fusion involves the combination of information from JDL levels two and three. Note that to some researchers, information fusion also encompasses data fusion. (See *data fusion* and *opportunistic information fusion*)

Information gathering An agent-theoretic term for the activity that an agent engaged to update its environment representation. In this thesis, we consider this term to be similar to *information acquisition*. (See also *sensor management* and *perception management*)

Large-scale information acquisition (LSIA) Comprehensive information acquisition that relies on a large number of sensing resources. Its many properties are discussed in Chapter 3. (See *information acquisition*)

Measurement A measurement is basically data originating from a single sensor. (See *observation* and *percept*)

Myopic A sensor manager is myopic if it only considers one sensing action at a time (and disregard the reward of future actions). (See *farsighted*)

Observation In this thesis, an observation is an assertion about the state the environment. It might be based on a single measurement och might be inferred from several measurements. (See *percept*, *measurement* and *environment*)

Observer An observer is an agent. We use the term observer when we focus on the perception qualities of an agent.

Opportunistic information fusion The concept of discovering and exploiting shared sensors for data and information fusion (Challa et al., 2005). (See *data fusion* and *information fusion*)

Organic data Data owned by a network *node* which originates from measurements made by node itself (Llinas, 2003).

Percept A percept is an agent's perceived interpretation of an observation expressed in a form (perhaps relying on an ontology) that it can understand (Weyns et al., 2003). (See *observation*, *measurement* and *environment*)

Perception management Our extension of the sensor management concept. It especially invites an agent context for information acquisition (described in Chapter 2) and includes related activities such as facilitation (described in Chapter 3). (See *sensor management*)

Perception manager A process devoted to perform perception management. (See *perception management*)

PIR sensor An electronic component that has a surface which is sensitive to infrared radiation. The component generates a electrical signal whenever the infrared radiation that reaches the sensor changes. The sensor is useful for detecting moving, infrared radiating, sources.

Pose The subset of an agent's state that concerns its spatial position and orientation.

Sensor management An activity devoted to controlling sensors for some objective. We argue that sensor management is mostly associated with a local control of sensor parameters to improve the tracking performance of a specific platform. We propose a generalisation, *perception management*. (See *perception management*)

Situation awareness A representation and interpretation of the state of the environment as a basis for decision making.

Bibliography

Abidi, M. A., & Gonzalez, R. C. (Eds.). (1992). *Data Fusion In Robotics and Machine Intelligence.* Academic Press, San Diego.

Adrian, R. (1993). Sensor management. In *Proceedings, AIAA/IEEE Conference on Digital Avionics System*, pp. 32–37. IEEE Computer Society Press.

Alberts, D. S., Garstka, J. J., & Stein, F. P. (1999). *Network Centric Warfare: Developing and Leveraging Information Superiority* (2 edition). CCRP Publication Series.

Albus, J. S. (1999). The engineering of mind. *Information Sciences*, *117*, 1–18.

Aloimonos, Y., Weiss, I., & Bandopadhay, A. (1987). Active vision. In *Proceedings of the first IEEE International Conference of Computer Vision*, pp. 35–54.

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

Arkin, R. C. (1998). *Behavior-Based Robotics.* MIT Press.

Arnborg, S., Artman, H., Brynielsson, J., & Wallenius, K. (2000). Information awareness in command and control: Precision, quality, utility. In *Proceedings of the 3rd International Conference on Information Fusion*, pp. 25–32. International Society of Information Fusion.

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

Asada, M., Kitano, H., Noda, I., & Veloso, M. (1999). RoboCup: Today and tomorrow - What we have learned. *Artificial Intelligence*, *110*(2), 193–214.

Bajcsy, R. (1988). Active perception. *Proceedings of the IEEE*, *76*(8), 996–1005.

Bar-Shalom, Y., & Fortmann, T. E. (1988). *Tracking and Data Association.* Academic Press.

Benameur, K. (2000). On the determination of active and passive measurement strategies. In *Proceedings of the 3rd International Conference on Information Fusion*. International Society of Information Fusion.

Benaskeur, A. (2002). Sensor management in command and control. In *Proceedings of the 7th International Command and Control Research and Technology Symposium*.

Bender, P. S. (1983). *Resource Management - An alternative view of the management process*. John Wiley and Sons.

Berenji, H. R., Vengerov, D., & Ametha, J. (2003). Co-evolutionary perception-based reinforcement learning for sensor allocation in autonomous vehicles. In *Proceedings of the 12th IEEE International Conference on Fuzzy Systems*. IEEE.

Bergman, N. (1999). *Recursive Bayesian Estimation Navigation and Tracking Applications*. Ph.D. thesis, Linköping University.

Blackman, S., & Popoli, R. (1999). *Design and Analysis of Modern Tracking Systems*. Artech House, Norwood, MA.

Blasch, E. P., & Hanselman, P. (2000). Information fusion and information superiority. In *Proceedings of the IEEE National Conference on Aerospace and Electronics*, pp. 290–297. IEEE.

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

Bowman, C. L., & Steinberg, A. N. (2001). A systems engineering approach for implementing data fusion systems. In Hall, & Llinas (Hall & Llinas, 2001), chap. 16, pp. 16–1 – 16–39.

Brännström, M., Lennartsson, R., Lauberts, A., Habberstad, H., Jungert, E., & Holmberg, M. (2004). Distributed data fusion in a ground sensor network. In Svensson, P., & Schubert, J. (Eds.), *Proceedings of the 7th International Conference on Information Fusion*, Vol. 2, pp. 1096–1103. International Society of Information Fusion.

Brynielsson, J., & Wallenius, K. (2003). A toolbox for multi-attribute decision-making. Tech. rep. TRITA–NA–0307, Department of Numerical Analysis and Computer Science, Royal Institute of Technology, Stockholm, Sweden.

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

Bui, H. H., Venkatesh, S., & West, G. (2002). Policy recognition in the abstract hidden markov model. *Journal of Artificial Intelligence Research*, *17*, 451–499.

Cao, Y. U., Fukunaga, A. S., & Kahng, A. B. (1997). Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, *4*, 7–27.

Castañón, D. A. (1997). Approximate dynamic programming for sensor management. In *Proceedings of the 1997 Conference on Decision and Control*, pp. 1202–1207.

Challa, S., Gulrez, T., Chaczko, Z., & Paranesha, T. (2005). Opportunistic information fusion: A new paradigm for next generation networked sensing systems. In *Proceedings of the 8th International Conference on Information Fusion*. International Society of Information Fusion.

Charles L. Ortiz, J., Hsu, E., desJardins, M., Rauenbusch, T., Grosz, B., Yadgar, O., & Kraus, S. (2001). Incremental negotiation and coalition formation for resource-bounded agents. Preliminary report. AAAI Fall Symposium on Negotiation Methods for Autonomous Cooperative Systems.

Coello, C. A. C. (1999). An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In Angeline, P. J., Michalewicz, Z., Schoenauer, M., Yao, X., & Zalzala, A. (Eds.), *Proceedings of the Congress on Evolutionary Computation*, Vol. 1, pp. 3–13, Mayflower Hotel, Washington D.C., USA. IEEE Press.

Cohon, J. L. (2003). *Multiobjective programming and planning*. Dover Publications, Inc.

Contat, M., Nimier, V., & Reynaud, R. (2001). An ordering method to select a sensor and its mode in a multitaget environment. In *Proceedings of the 4th International Conference on Information Fusion*. International Society of Information Fusion.

Contat, M., Nimier, V., & Reynaud, R. (2002). Request management using contextual information for classification. In *Proceedings of the 5th International Conference on Information Fusion*, pp. 1147–1153. International Society of Information Fusion.

Cook, D. J., Gmytrasiewicz, P., & Holder, L. B. (1996). Decision-theoretic cooperative sensor planning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(10), 1013–1023.

Coradeschi, S., & Saffiotti, A. (2003). An introduction to the anchoring problem. *Robotics and Autonomous Systems*, *43*(2–3), 85–96.

Cozman, F. G. (2005). Graphical models for imprecise probabilities. *International Journal of Approximate Reasoning*, *39*(2-3), 167–184.

Dagan, N., Volij, O., & Winter, E. (2002). A characterization of the Nash bargaining solution. *Social Choice and Welfare*, *19*(4), 811–823.

David Rios Insua, Fabrizio Ruggeri, D. R. I. (2000). *Robust Bayesian Analysis (Lecture Notes in Statistics)*. Springer-Verlag.

Denton, R., Alcaraz, E., Llinas, J., & Hintz, K. (1994). Towards modern sensor management systems. In Levis, A. H., & Levis, I. S. (Eds.), *Science of Command and Control: Part III - Coping with change*, chap. 13, pp. 119–134. AFCEA International Press, Fairfax, Virginia.

d'Inverno, M., Kinny, D., Luck, M., & Wooldridge, M. (1997). A formal specification of dMARS. In Singh, M. P., Rao, A. S., & Wooldridge, M. (Eds.), *Intelligent Agents IV: Proceedings of the Fourth International Workshop on Agent Theories, Architectures, and Languages*, pp. 155–176. Springer-Verlag.

Dodin, P., & Nimier, V. (2001). Distributed resource allocation under communication constraints. In *Proceedings of the 4th International Conference on Information Fusion*. International Society of Information Fusion.

Doucet, A., Freitas, N. D., & Gordon, N. (Eds.). (2001). *Sequential Monte Carlo Methods in Practice*. Springer Verlag.

Doucet, A., Vo, B.-N., Andrieu, C., & Davy, M. (2002). Particle filtering for multitarget tracking and sensor management. In *Proceedings of the 5th International Conference on Information Fusion*, pp. 474–481. International Society of Information Fusion.

Durfee, E. H., Lesser, V. R., & Corkill, D. D. (1989). Cooperative distributed problem solving. In Barr, A., Cohen, P. R., & Feigenbaum, E. A. (Eds.), *The handbook of Artificial Intelligence*, Vol. 4, chap. 17, pp. 83–147. Addison Wesley, USA.

Durfee, E. H. (2001). Distributed problem solving and planning. In Luck, M., Marík, V., Stepánková, O., & Trappl, R. (Eds.), *EASSS*, Vol. 2086 of *Lecture Notes in Computer Science*, pp. 118–149. Springer.

Durrant-Whyte, H. F. (1988). *Integration, Coordination and control of Multi-sensor robot systems*. Kluwer Academic Publishers, Norwood, MA.

Elmenreich, W., Schneider, L., & Kirner, R. (2001). A robust certainty grid algorithm for robotic vision. In *Proceedings of the Sixth IEEE International conference on Intelligent Engineering Systems*.

Endriss, U., Maudet, N., Sadri, F., & Toni, F. (2003). Resource allocation in egalitarian societies. In Herzig, A., Chaib-draa, B., & Mathieu, P. (Eds.), *Secondes Journées Francophones sur les Modèles Formels d'Interaction (MFI-2003)*, pp. 101–110. Cépaduès-Éditions.

Endriss, U., Maudet, N., Sadri, F., & Toni, F. (2005). Negotiating socially optimal allocations of resources: An overview. Tech. rep., Imperial College London, Department of Computing.

Farinelli, A., Iocchi, L., & Nardi, D. (2003). An analysis of coordination in multirobot systems. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1487–1492. IEEE.

Gmytrasiewicz, P. J., & Durfee, E. H. (2000). Rational coordination in multi-agent environments. *Autonomous Agents and Multi-Agents Systems*, *3*(4), 319–350.

Gonsalves, P. G., Cunningham, R., Ton, N., & Okon, D. (2000). Intelligent threat assessment processor (ITAP) using genetic algorithms and fuzzy logic. In

Proceedings of the 3rd International Conference on Information Fusion. International Society of Information Fusion.

Gonsalves, P. G., & Rinkus, G. J. (1998). Intelligent fusion and asset management processor. In *Proceedings of the 1998 IEEE Information Technology Conference*, pp. 15–18. IEEE.

Grahn, P., Grönwall, C., Herberthson, M., Kaijser, T., Lantz, F., Strömberg, D., & Ulvklo, M. (2005). Sensor control in NCW - problem description and important areas. Tech. rep. FOI-R–1860–SE, ISSN 1650-1942, Defence Research Establishment, Division of Command and Control Warfare Technology, P.O. Box 1165, SE-581 11 LINKÖPING, SWEDEN.

Grünwald, P. D., & Dawid, A. P. (2004). Game theory, maximum entropy, minimum discrepancy and robust bayesian decision theory. *Annals of Statistics*, *32*(4), 1367–1433.

Grupen, R., & Henderson, T. C. (1989). Autochthonous behaviors - mapping perception to action. In Henderson, T. C. (Ed.), *Traditional and Non-Traditional Robotic Sensors.* Springer-Verlag, Berlin.

Haenni, R., & Hartmann, S. (2004). A general model for partially reliable information sources. In Svensson, P., & Schubert, J. (Eds.), *Proceedings of the 7th International Conference on Information Fusion*, Vol. 1, pp. 153–160. International Society of Information Fusion.

Hager, G. D. (1990). *Task-Directed Sensor Fusion and Planning - A Computational Approach.* Kluwer Academic Publishers, Norwell, MA.

Hall, D. L. (1992). *Mathematical Techniques in Multisensor Data Fusion.* Artech House, Norwood, MA.

Hall, D. L., & Llinas, J. (Eds.). (2001). *Handbook of Multisensor Data Fusion.* CRC Press.

Hand, D., Mannila, H., & Smyth, P. (2001). *Principles of data mining.* MIT Press.

Henderson, T., & Shilcrat, E. (1984). Logical sensor systems. *Journal of Robotic Systems*, *1*, 169–193.

Hernandez, M. L., Kirubarajan, T., & Bar-Shalom, Y. (2002). Efficient multisensor resource management using Cramér-Rao lower bounds. In *Proceedings of the SPIE International Conference on Signal and Data Processing of Small Targets.* SPIE.

Hintz, K. J. (2004). Implicit collaboration. In Kadar, I. (Ed.), *Proceedings of SPIE, Signal Processing, Sensor Fusion and Target Recognition*, pp. 89–94. SPIE.

Hintz, K. J., & McIntyre, G. (1999). Goal lattices for sensor management. In *Proceedings of the SPIE International Symposium on Aerospace/Defence Sensing & Control*, Vol. 3720, pp. 249–255. SPIE.

Horling, B., Vincent, R., Mailler, R., Shen, J., Becker, R., Rawlins, K., & Lesser, V. (2001). Distributed sensor network for real time tracking. In *Proceedings of the 5th International Conference on Autonomous Agents*, pp. 417–424, Montreal. ACM Press.

Hovland, G. E., & McCarragher, B. J. (1997). Dynamic sensor selection for robotic systems. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 272–277. IEEE.

Howard, R. A. (1966). Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, *2*, 22–26.

Huhns, M. N., & Stephens, L. M. (1999). Multiagent systems and societies of agents. In Weiss (Weiss, 1999), chap. 2, pp. 79–120.

Humphry, M. (1997). *Action Selection using Reinforcement Learning*. Ph.D. thesis, University of Cambridge.

Jaynes, E. T. (2003). *Probability theory: The logic of science*. Cambridge University Press.

Jennings, J. S., Whelan, G., & Evans, W. F. (1997). Cooperative search and rescue with a team of mobile robots. In *Proceedings of the International Conference on Advanced Robotics*, pp. 193–200.

Jennings, N. R. (1993). Commitments and conventions: The foundation of coordination in multi-agent systems. *The knowledge engineering review*, *8*(3), 223–250.

Jennings, N. R., Sycara, K., & Wooldridge, M. J. (1998). A roadmap of agent research and development. *Journal of Autonomous agents and multi-agent systems*, *1*(1), 275–306.

Johansson, L. R. M. (2005). Recurrent negotiations for multi-agent coordination in antagonistic environments. Submitted to the Journal of Artificial Intelligence Research, December 2005.

Johansson, R., & Suzić, R. (2004). Bridging the gap between information need and information acquisition. In Svensson, P., & Schubert, J. (Eds.), *Proceedings of the 7th International Conference on Information Fusion*, Vol. 2, pp. 1202–1209. International Society of Information Fusion.

Johansson, R., & Suzić, R. (2005). Particle filter-based information acquisition for robust plan recognition. In *Proceedings of the 8th International Conference on Information Fusion*. International Society of Information Fusion.

Johansson, R., & Xiong, N. (2003). Perception management - an emerging concept for information fusion. *Information Fusion*, *4*(3), 231–234.

Kaelbling, L. P., Littman, M. L., & Moore, A. P. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, *4*, 237–285.

Kahan, J. P., & Rapoport, A. (1984). *Theories of Coalition Formation*. Lawrence Erlbaum Associates, 365 Broadway, Hillsdale, New Jersey 07642, USA.

Kalai, E. (1985). Solutions to the bargaining problem. In Hurwicz, L., Schmeidler, D., & Sonnenschein, H. (Eds.), *Social goals and social organization*, chap. 3, pp. 77–105. Cambridge University Press, USA.

Kalandros, M., & Pao, L. Y. (1998). Controlling target estimate covariance in centralized multisensor systems. In *Proceedings of American Control Conference*, pp. 2749–2753.

Kannan, R., Sarangi, S., Ray, S., & Iyengar, S. S. (2002). Minimal sensor integrity in sensor grids. In *Proceedings of the 5th International Conference on Information Fusion*, pp. 959–964. International Society of Information Fusion.

Karan, M., & Krishnamurthy, V. (2003). Sensor scheduling with active and image-based sensors for maneuvering targets. In *Proceedings of the 6th International Conference on Information Fusion*, pp. 1018–1023. International Society of Information Fusion.

Kastella, K., & Musick, S. (1996). The search for optimal sensor management. In *Proceedings of SPIE*, pp. 318–329. SPIE.

Klein, K., & Sequeira, V. (2000). View planning for the 3D modelling of real world scenes. In *Proceedings of the IEEE Conference on Intelligent Robots and Systems*, pp. 943–948. IEEE.

Klir, G., & Yuan, B. (1998). Basic concepts and history of fuzzy set theory and fuzzy logic. In Ruspini, E., Bonissone, P., & Pedrycz, W. (Eds.), *Handbook of Fuzzy Computation*, chap. A1.1, pp. A1.1:1–9. Institute of Physics Publishing.

Koch, W. (2004). On 'negative' information in tracking and sensor data fusion: Discussion of selected examples. In Svensson, P., & Schubert, J. (Eds.), *Proceedings of the 7th International Conference on Information Fusion*, pp. 91–98. International Society of Information Fusion.

Kraus, S. (2001a). Automated negotiation and decision making in multiagent environments. In Luck, M., Marík, V., Stepánková, O., & Trappl, R. (Eds.), *EASSS*, Vol. 2086 of *Lecture Notes in Computer Science*, pp. 150–172. Springer.

Kraus, S. (2001b). *Strategic Negotiation in Multiagent Environments.* MIT Press.

Kreucher, C., Kastella, K., & III, A. O. H. (2003). Multi-target sensor management using alpha-divergence measures. In Zhao, F., & Guibas, L. J. (Eds.), *IPSN*, Vol. 2634 of *Lecture Notes in Computer Science*, pp. 209–222. Springer.

Latombe, J.-C. (1993). *Robot motion planning.* Kluwer Academic Publishers, Norwell, MA, USA.

Lee, S., & Ro, S. (1999). Robotics with perception and action nets. In Ghosh, B. K., Xi, N., & Tarn, T. J. (Eds.), *Control in Robotics and Automation - Sensor-based integration*, chap. 12, pp. 347–380. Academic Press, San Diego, CA.

Lesser, V., Decker, K., Wagner, T., Carver, N., Garvey, A., Horling, B., Neiman, D., Podorozhny, R., NagendraPrasad, M., Raja, A., Vincent, R., Xuan, P., & Zhang, X. (2004). Evolution of the GPGP/TAEMS Domain-Independent Coordination Framework. *Autonomous Agents and Multi-Agent Systems*, *9*(1), 87–143.

Leyton-Brown, K. (2003). *Resource allocation in multiagent systems*. Ph.D. thesis, Stanford University.

Lichtenauer, J., Reinders, M., & Hendriks, E. (2004). Influence of the observation likelihood function on particle filtering performance in tracking applications. In *Proceedings of the 6th International Conference on automatic Face and Gesture Recognition*. IEEE.

Llinas, J. (1988). Toward the utilization of certain elements of ai technology for multi sensor data fusion. In Harris, C. J. (Ed.), *Application of artificial intelligence to command and control systems*, chap. 3.1, pp. 126–184. Peter Peregrinus Ltd, London, UK.

Llinas, J. (2003). Studying the complexities in distributed object tracking systems. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, Vol. 2, pp. 2035–2041. IEEE.

Llinas, J., Bowman, C., Ragova, G., Steinberg, A., Waltz, E., & White, F. (2004). Revisiting the JDL data fusion model II. In Svensson, P., & Schubert, J. (Eds.), *Proceedings of the 7th International Conference on Information Fusion*, Vol. 2, pp. 1218–1230. International Society of Information Fusion.

López, J. M., Rodríguez, F. J., & Corredera, J. C. (1995). Fuzzy reasoning for multisensor management. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, Vol. 2, pp. 1398–1403. IEEE.

Lowe, D. (1998). The relevance and application of information fusion to financial analysis. In Bedworth, M., & O'Brien, J. (Eds.), *Proceedings, 3rd International Conference on Information Fusion*, pp. 5–12.

Lozano-Pérez, T. (1983). Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, *C-32*, 108–120.

Luo, R. C., & Kay, M. G. (1992). Data fusion and sensor integration: State-of-the-art 1990s. In Abidi, & Gonzalez (Abidi & Gonzalez, 1992), chap. 2, pp. 7–135.

Mahler, R. (2003). Multisensor-multitarget sensor management: A unified Bayesian approach. In *Proceedings of the 2003 SPIE AeroSense Conference*, Vol. 5096. SPIE.

Mahler, R. P. (2004). The levels 2, 3, 4 fusion challenge: Fundamental statistics. In Svensson, P., & Schubert, J. (Eds.), *Proceedings of the 7th International Conference on Information Fusion*, Vol. 1, pp. 535–536. International Society of Information Fusion.

Makarenko, A., Brooks, A., Williams, S., & Durrant-Whyte, H. (2004). A decentralized architecture for active sensor networks. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1097–1102. IEEE.

Makarenko, A., & Durrant-Whyte, H. (2004). Decentralized data fusion and control in active sensor networks. In Svensson, P., & Schubert, J. (Eds.), *Proceedings of the 7th International Conference on Information Fusion*, pp. 479–486. International Society of Information Fusion.

Makarenko, A. A. (2004). *A decentralized architecture for active sensor networks*. Ph.D. thesis, University of Sydney.

Malone, T. W., & Crowston, K. (1994). The inderdisciplinary study of coordination. *ACM Computing Surveys*, *26*(1), 87–119.

Manyika, J., & Durrant-Whyte, H. (1994). *Data Fusion and Sensor Management - a decentralized information-theoretic approach*. Ellis Horwood.

Mariotti, M. (1998). Nash barganing theory when the number of alternatives can be finite. *Social choice and welfare*, *15*(3), 413–421. ISSN: 0176-1714 (Paper copy).

Martin, D. L., Cheyer, A. J., & Moran, D. B. (1999). The open agent architecture: A framework for building distributed software systems. *Applied Artificial Intelligence*, *13*(1-2), 91–128.

Matarić, M. J. (1995). Issues and approaches in design of collective autonomous agents. *Robotics and Autonomous Systems*, *2-4*(16), 321–331.

Mavrantza, R., Gountromichou, C., Vamvoukakis, C., Kouli, M., Karfakis, I., Santamouri, M., Seymour-St., K., Argialas, D., Karathanassi, V., Papadimatou, K., Michailidou, E., & Rokos, D. (2002). 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*.

McCarragher, B. J. (1998). Control problems in robotics and automation. In Siciliano, B., & Valavanis, K. P. (Eds.), *Control Problems in Robotics and Automation*, chap. Discrete Event Theory for the monitoring and control of robotic systems, pp. 209–225. Springer Verlag London Limited, Great Britain.

McIntyre, G. (1998). *A comparative approach to sensor management and scheduling*. Ph.D. thesis, George Mason University, Fairfax, FA.

McLaughlin, S., Krishnamurthy, V., & Evans, R. J. (2004). Bayesian network model for data incest in a distributed sensor network. In Svensson, P., & Schubert, J. (Eds.), *Proceedings of the 7th International Conference on Information Fusion*, Vol. 1, pp. 606–613. International Society of Information Fusion.

Miller, B., Malloy, M. A., Masek, E., & Wild, C. (2001). Towards a framework for managing the information environment. *Information, knowledge, systems management*, *2*, 359–384.

Mitchell, T. (1997). *Machine learning.* McGraw-Hill.

Mohamed, A. M. (2000). *Benevolent agents.* Ph.D. thesis, University of Southern California.

Muir, P. (1990). A virtual sensor approach to robot kinematic identification. In *Proceedings of the 1990 IEEE International Conference on Systems Engineering*, pp. 440–445. IEEE.

Murphy, K. P. (2002). *Dynamic Bayesian Networks: Representation, Inference and Learning.* Ph.D. thesis, University of California, Berkeley.

Musick, S., & Malhotra, R. (1994). Chasing the elusive sensor manager. In *Proceedings of the National Conference on Aerospace and Electronics*, pp. 606–613. IEEE.

Naisbitt, J. (Ed.). (1982). *Megatrends - Ten new directions transforming our lives.* Warner Books.

Nash, J. M. (1977). Optimal allocation of tracking resources. In *Proceedings of the IEEE Conference on Decision and Control*, Vol. 1, pp. 1170–1180. IEEE.

Nash, J. F. (1950). The bargaining problem. *Econometrica*, *18*, 155–162.

Nedich, A., Schneider, M., & Washburn, R. (2005). Farsighted sensor management strategies for move/stop tracking. In *Proceedings of the 8th International Conference on Information Fusion.* International Society of Information Fusion.

Ng, G. W., & Ng, K. H. (2000). Sensor management - what, why and how. *Information Fusion*, *1*, 67–75.

Nguyen, T. D., & Jennings, N. R. (2005). Managing commitments in multiple concurrent negotiations. *Electronic Commerce Research and Applications*, *4*, 362–376.

Nilsson, N. J. (1998). *Stimulus-Response Agents*, chap. 2, pp. 21–35. Morgan Kaufmann Publishers Inc.

Nwana, H. S., Lee, L., & Jennings, N. R. (1996). Co-ordination in software agent systems. *British Telecom Technical Journal*, *14*(4), 79–88.

Osborne, M. J., & Rubinstein, A. (1994). *A Course in Game Theory.* MIT Press.

Ostwald, J., Lesser, V., & Abdallah, S. (2005). Combinatorial auction for resource allocation in a distributed sensor network. *The 26th IEEE International Real-Time Systems Symposium, (RTSS'05)*, 266–274.

Paradis, S., Chalmers, B. A., Carling, R., & Bergeron, P. (1997). Towards a generic model for situation and threat assessment. In Raja, S. B. (Ed.), *Proceedings of SPIE, Digitization of the Battlefield II*, pp. 171–182. SPIE.

Paradis, S., Roy, J., & Treurniet, W. (1998). Integration of all data fusion levels using a blackboard architecture. In Bedworth, M., & O'Brien, J. (Eds.), *Proceedings, 3rd International Conference on Information Fusion*, pp. 195–202.

Parker, L. E. (1998). ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, *14*(2), 220–240.

Parsons, S., & Wooldridge, M. (2002). Game theory and decision theory in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, *5*(3), 243–254.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc.

Penny, D. E. (1998). The automatic management of multi-sensor systems. In *Proceedings of the International Conference on Information Fusion*, pp. 748–755.

Penny, D. E., & Williams, M. (2000). 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*, pp. 598–609. SPIE.

Perillo, M. A., & Heinzelman, W. B. (2003). Optimal sensor management under energy and reliability constraints. In *Proceedings of the IEEE Wireless Communication and Networking Conference*.

Phister, P. W., Busch, T., & Plonisch, I. G. (2003). Joint synthetic battlespace: Cornerstone for predictive battlespace awareness. In *Proceedings of the 8th International Command and Control Research and Technology Symposium*.

Piccerillo, R. A., & Brumbaugh, D. A. (2004). Predictive battlespace awareness: Linking intelligence, surveillance and reconnaissance operations to effects based operations. In *Proceedings of the 9th International Command and Control Research and Technology Symposium*.

Pinz, A., Prantl, M., Ganster, H., & Kopp-Borotschnig, H. (1996). Active fusion - a new method applied to remote sensing image interpretation. *Pattern Recognition Letters*, *17*, 1349–1359.

Pirjanian, P. (1998). *Multi-objective Action Selection and Behavior fusion using voting*. Ph.D. thesis, Aalborg University.

Pirjanian, P., & Matarić, M. (2000). Multi-robot target acquisition using multiple objective behavior coordination. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2696–2702. IEEE.

Rasmusen, E. (2001). *Games and information - an introduction to game theory* (Third edition). Blackwell Publishing Ltd.

Ricordel, J.-M., & Demazeau, Y. (2000). From analysis to deployment: a mulitagent platform survey. In Omicini, A., Tolksdorf, R., & Zambonelli, F. (Eds.), *ESAW*, Vol. 1972 of *Lecture Notes in Computer Science*, pp. 93–105. Springer.

Rimey, R. D. (1993). Control of selective perception using Bayes nets and decision theory. Tech. rep. TR468, Computer Science Department, The University of Rochester. citeseer.nj.nec.com/rimey93control.html.

Roy, S. D. (2000). *Active Object Recognition through Next View Planning*. Ph.D. thesis, Indian Institute of Technology.

Russell, S., & Norvig, P. (1995). *Artificial Intelligence - A Modern Approach* (First edition). Prentice Hall, Englewood Cliffs, New Jersey.

Sandholm, T. W. (1999). Distributed rational decision making. In Weiss (Weiss, 1999), chap. 5, pp. 201–258.

Schaefer, C., & Hintz, K. J. (2000). Sensor management in a sensor rich environment. In *Proceedings of the SPIE International Symposium on Aerospace/Defence Sensing & Control*, Vol. 4052, pp. 48–57. SPIE.

Schmaedeke, W. (1993). Information based sensor management. In *Proceedings of SPIE, Signal processing, sensor fusion, and target recognition II*, pp. 156–164. SPIE.

Schmaedeke, W., & Kastella, K. (1998). Information based sensor management and IMMKF. In *Proceedings of the SPIE International Conference on Signal and Data Processing of Small Targets*, pp. 390–401. SPIE.

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

Shannon, C. E. (1948). A mathematical theory of communication. Tech. rep., Bell System Technical Journal.

Shoham, Y., & Tennenholtz, M. (1995). On social laws for artificial agent societies: off-line design. *Artificial Intelligence*, *73*(1–2), 231–252.

Sidenbladh, H. (2003). Multi-target particle filtering for the probability hypothesis density. In *Proceedings of the 6th International Conference on Information Fusion*, pp. 800–806. International Society of Information Fusion.

Smith, R. G. (1981). The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, *C-29*(12), 1104–1113.

Soliday, S. W. (1999). A genetic algorithm model for mission planning and dynamic resource allocation of airborne sensors..

Starr, A., & Desforges, M. (1998). Strategies in data fusion - sorting through the tool box. In Bedworth, M., & O'Brien, J. (Eds.), *Proceedings, 3rd International Conference on Information Fusion*, pp. 85–90.

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

Steinberg, A. N. (2005). An approach to threat assessment. In *Proceedings of the 8th International Conference on Information Fusion*. International Society of Information Fusion.

Steinberg, A. N., & Bowman, C. L. (2001). Revisions to the JDL data fusion model. In Hall, & Llinas (Hall & Llinas, 2001), chap. 2, pp. 2–1 – 2–19.

Strömberg, D., Andersson, M., & Lantz, F. (2002). On platform-based sensor management. In *Proceedings of the 5th International Conference on Information Fusion*, pp. 600–607. International Society of Information Fusion.

Strömberg, D., & Lantz, F. (2003). Operator control of shared resources in sensor networks. In *Proceedings of the 6th International Conference on Information Fusion*, pp. 575–582. International Society of Information Fusion.

Suzić, R. (2003a). Generic representation of military organisation and military behaviour: UML and Bayesian networks. In *Proceedings of the NATO RTO Symposium on C3I and M&S Interoperability*.

Suzić, R. (2003b). Representation and recognition of uncertain enemy policies using statistical models. In *Proceedings of the NATO RTO Symposium on Military Data and Information Fusion*.

Suzić, R., & Johansson, R. (2004). Realization of a bridge between high-level information need and sensor management using a common DBN. In *The 2004 IEEE International Conference on Information Reuse and Integration (IEEE IRI-2004)*. IEEE.

Svensson, P., Johansson, K., & Jöred, K. (1997). Submarine tracking using multi-sensor data fusion and reactive planning for the positioning of passive sonobuoys. In *Proceedings of Hydroakustik 1997, FOA, Stockholm, Sweden*.

Tarabanis, K. A., Allen, P. K., & Tsai, R. Y. (1995). A survey of sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, *11*(1), 86–104.

Thomson, W. (1994). Cooperative models of bargaining. In Aumann, R. J., & Hart, S. (Eds.), *Handbook of game theory with economic applications*, Vol. 2, chap. 35, pp. 1237–1284. North-Holland Publ. Co., Amsterdam, The Netherlands.

Tsoukiàs, A., & Vincke, P. (1992). A survey on non conventional preference modeling. *Ricerca Operativa*, *61*, 5–49.

Waltz, E., & Llinas, J. (1990). *Multisensor Data Fusion*. Artech House, Norwood, Massachussets.

Weiss, G. I. (Ed.). (1999). *Multiagent Systems - A Modern Approach to Distributed Artificial Intelligence*. MIT Press.

Weyns, D., Steegmans, E., & Holvoet, T. (2003). A model for active perception in situated multi-agent systems. In M. d'Inverno, C. S., & Zambonelli, F.

(Eds.), *Proceedings of the first European Workshop on Multiagent Systems (EUMAS)*, pp. 1–15.

White, F. E. (1998). Managing data fusion systems in joint and coalition warfare. In Bedworth, M., & O'Brien, J. (Eds.), *Proceedings, 3rd International Conference on Information Fusion*, pp. 49–52.

Wooldridge, M. (1999). Intelligent agents. In Weiss (Weiss, 1999), chap. 1, pp. 27–78.

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

Yukish, M. A. (2004). *Algorithms to identify Pareto points in multi-dimensional data sets.* Ph.D. thesis, Pennsylvania State University.

Zhang, Z., & Hintz, K. J. (1996). OGUPSA sensor scheduling architecture and algorithm. In *Proceedings of the SPIE International Symposium on Aerospace/Defense Sensing & Control*, Vol. 2755, pp. 296–303. SPIE.

Zimmermann, H.-J. (Ed.). (1991). *Fuzzy Set Theory - And Its Applications.* Kluwer Academic Publishers.

Zlotkin, G., & Rosenschein, J. S. (1996a). Mechanism design for automated negotiation, and its application to task oriented domains. *Artificial Intelligence*, *86*(2), 195–244.

Zlotkin, G., & Rosenschein, J. S. (1996b). Mechanisms for automated negotiation in state oriented domains. *Journal of Artificial Intelligence Research*, *5*, 163–238.

Index