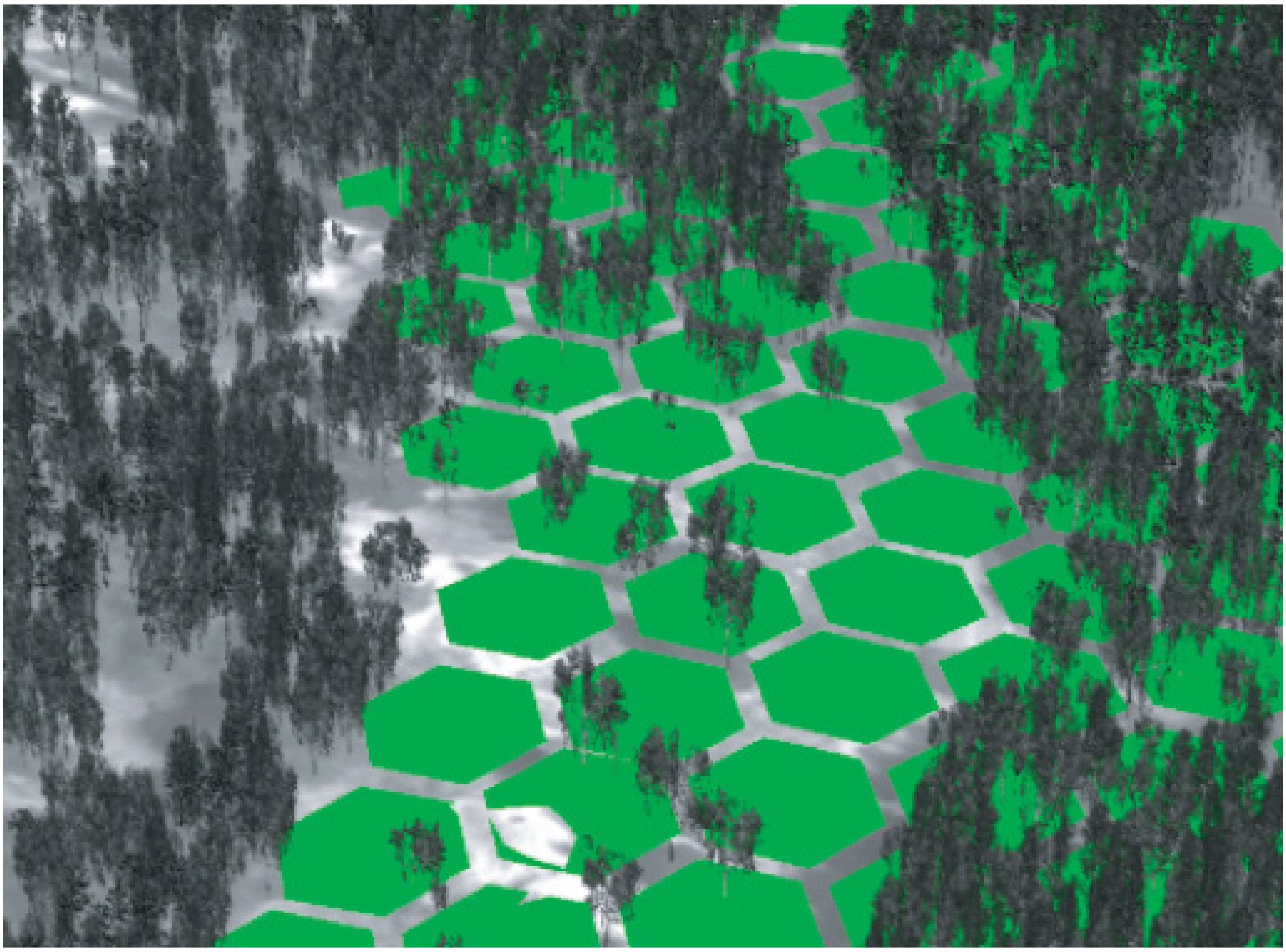
Path and Sensor Planning Framework Applicable to UAV Surveillance with EO/IR sensors

PER SKOGLAR, JONAS NYGÅRDS, RICKARD BJÖRSTRÖM, PETTER ÖGREN,
JOHAN HAMBERG, PATRIK HERMANSSON, MORGAN ULVKLO



FOI-R--1741--SE Scientific report
ISSN-1650-1942 September 2005

Sensor Technology
Systems Technology

Per Skoglar, Jonas Nygårds, Rickard Björström, Petter Ögren, Johan Hamberg, Patrik Hermansson, Morgan Ulvklo

Path and Sensor Planning Framework Applicable to UAV Surveillance with EO/IR sensors

FOI-R--1741--SE Scientific report **Sensor Technology**
ISSN-1650-1942 September 2005 **Systems Technology**

Issuing organisation FOI – Swedish Defence Research Agency Sensor Technology P.O. Box 1165 SE-581 11 LINKÖPING	**Report number, ISRN** FOI-R--1741--SE	**Report type** Scientific report
	Research area code Mobility and Space Technology, incl Materials	
	Month year September 2005	**Project no.** E3969
	Sub area code Unmanned Vehicles	
	Sub area code 2 Unmanned Vehicles	
Author/s (editor/s) Per Skoglar, Jonas Nygårds, Rickard Björström, Petter Ögren, Johan Hamberg, Patrik Hermansson, Morgan Ulvklo	**Project manager** Morgan Ulvklo	
	Approved by Lena Klasén	
	Sponsoring agency FMV	
	Scientifically and technically responsible Jonas Nygårds	

Report title
Path and Sensor Planning Framework Applicable to UAV Surveillance with EO/IR sensors

Abstract
This report presents a path and sensor planning framework for a UAV equipped with gimballed EO/IR sensors. The goal is autonomuos UAV surveillance, i.e. searching for objects along roads or in certain areas and acquiring detailed images and geolocalization of discovered objects.

In particular, an information-theoretic approach to concurrent path and sensor planning is presented. The work is inspired by research in optimal observer trajectory computations for bearings-only tracking, but the method is also extended to handle multiple features, limited field-of-view and scene occlusion. Digital elevation models and hardware accelerated computer graphics models are used for detailed occlusion modelling.

Graph search methods are also proposed for the long term planning, in particular formulations using the Target Visitation Problem and the Weight Constrained Shortest Path Problem are investigated.

Keywords
path planning, sensor planning, information theory, Fisher information, optimal control, localization, area coverage, UAV surveillance, graph search, target visitation problem, constrained shortest path problem

Further bibliographic information	**Language** English
ISSN ISSN-1650-1942	**Pages** 118 p.
	Price According to price list

Utgivare FOI – Totalförsvarets forskningsinstitut Sensorteknik Box 1165 581 11 LINKÖPING	**Rapportnummer, ISRN** FOI-R--1741--SE	**Klassificering** Vetenskaplig rapport
	Forskningsområde Farkost- och rymdteknik, inkl material	
	Månad år September 2005	**Projektnummer** E3969
	Delområde Obemannade farkoster	
	Delområde 2 Obemannade farkoster	
Författare/redaktör Per Skoglar, Jonas Nygårds, Rickard Björström, Petter Ögren, Johan Hamberg, Patrik Hermansson, Morgan Ulvklo	**Projektledare** Morgan Ulvklo	
	Godkänd av Lena Klasén	
	Uppdragsgivare/kundbeteckning FMV	
	Tekniskt och/eller vetenskapligt ansvarig Jonas Nygårds	

Rapportens titel
Ramverk för rutt- och sensorplanering användbar för UAV-spaning med EO/IR-sensorer

Sammanfattning

Denna rapport presenterar ett ramverk för sensor- och ruttplanering för en UAV med EO/IR-sensorer. Det övergripande målet är autonom UAV-spaning, d.v.s. spaning efter mål längs vägar eller särskilda områden samt inhämtning av detaljerat sensordata och positionsinmätning av upptäckta mål.

Huvuddelen av rapporten beskriver en informationsteoretisk ansats för samtidig sensor- och ruttplanering. Metoden är inspirerad av forskningsresultat inom optimal ruttplanering för målföljning med passiv sensor, men utvidgningar är gjorda för att hantera flera objekt, begränsat synfält hos sensorn och ocklusion. Digitala höjdkartor och hårdvaruaccelererad datorgrafik används för detaljerad modellering av ocklusion orsakad av terräng, träd och byggnader.

Grafsökningsmetoder är även föreslagna för den långsiktiga planeringen. Verktyg inom Target Visitation Problem och Weight Constrained Shortest Path Problem tillämpas på UAV-spaningsproblemet.

Nyckelord
ruttplanering, sensorplanering, informationsteori, Fisher-information, optimal styrning, lokalisering, yt-täckning, UAV-spaning, grafsökning

Övriga bibliografiska uppgifter	**Språk** Engelska
ISSN ISSN-1650-1942	**Antal sidor:** 118 s.
Distribution enligt missiv	**Pris: Enligt prislista**

Contents

1 Introduction

When an autonomous UAV is used for surveillance and reconnaissance the problem of controlling the path and sensor must be solved in an intelligent manner. In this report the problem is analysed for a UAV with gimballed EO/IR sensors. An information-theoretic approach is investigated and extensive simulations are presented.

1.1 Background

The need for autonomous capabilities, such as on-board sensor data processing, sensor management, and path planning, will increase in both manned and unmanned platforms designed for future military operations. This as a consequence of the increasing use of sensors in combination with limitations in terms of bandwidth and human operator capacity.

Concurrent sensor and path planning, taking into account both platform and sensor constraints as well as threats and environmental conditions, is a very demanding task. Even more demanding is the capability to dynamically adapt and replan the sensor utilization and the platform trajectory in response to changes in the environment. Our working hypothesis is that integration of the detection-recognition chain with spatial awareness enables intelligent autonomous data acquisition by means of active sensor control and path planning.

1.2 References and Further Readings

The research field of UAV path planning and cooperation attracts a lot of attention and the number of publications is rapidly increasing. However, the problem of concurrent path and sensor planning is still a rather unexploited area. An introduction to the UAV surveillance and reconnaissance problem is given by Ulvklo et al. in [46] where the path and sensor planning challenges in UAV surveillance are discussed. Nygårds et al. [32] gives an overview of research fields and communities related to path planning and/or sensor planning. The report also includes a survey of methods, techniques and approaches to path and sensor planning.

One of the methods, an information-theoretic approach, proposed by Grocholsky in [16], is analysed in depth by Skoglar et al. in [40]. Although information-theoretic approaches have been applied to path planning problems before, the focus on *concurrent sensor and path planning* is new in [40]. A detailed presentation of information theory, information filter and utility functions can be found in Manyika [28] and an alternative approach is sketched in Chapter 11.

With the goal to solve the complete path and sensor planning problem, all parameters considered, the present work focus on target position estimation with bearing-only sensors. This problem still captures many of the important aspects of optimal sensor control. During the last six decades substantial research was published on optimal observer trajectory for bearings-only tracking.

Applications can be found in passive sonar and EO/IR tracking and in aircraft navigation systems. Typically, the problem is to estimate the position, and possibly the velocity, of a target, based on a sequence of noisy measurements from a bearings-only sensor mounted on a moving sensor platform, see e.g. [35, 44, 33] for more recent papers on this subject.

Grocholsky, [16], stated the 2D single object localization problem, together with a proposed method for area coverage surveillance. There he formulated the path planning problem as an optimal control problem. The example in Chapter 4 are from [16], as well as the area coverage idea in Chapter 5.

Optimal control is a keyword in the following chapters and the reader who wants to get a detailed exposé of solutions of the optimal control problem by parametrization of the control signal is referd to the book by Bertsekas, [8].

The gimballed sensor system that serves as pattern in Chapter 8.2 is presented in detail by Skoglar in [39].

1.3 Outline

Chapter 2 gives an introduction to the UAV surveillance and reconnaissance problem. In particular, the path and sensor planning challenges in UAV surveillance are discussed. This chapter is a summary of [46]. Chapter 3 introduces the information theory and the optimal control problem. In Chapter 4 the problem of optimal observer trajectory for bearings-only localization is considered. The problem is formulated as an optimal control problem. The effects of different utility function and time horizon are discussed. Furthermore, the models are extended to n objects and a 3D representation. Chapter 5 considers the area coverage problem and an area model is introduced. Chapter 6 focuses on planning the UAV path on a larger scale. The Traveling Salesman Problem (TSP) and the Weight Constrained Shortest Path Problem (WCSPP) are applied to UAV survelliance. In Chapter 7 a planning hierarchy is introduced based on the NURBS path representation. Some optimization aspects are also discussed. Chapter 8 and 9 consider the problems with limited field-of-view and occlusion, respectively. Models are presented and the method is extended with a sensor gazing representation. Chapter 10 shows the simulation results of some different surveillance missions. Chapter 11 aims at showing that the information filter approach has a much wider scope and validity than it was originally designed for. Finally, Chapter 12 summarizes the results and give suggestions for future work are given.

1.4 Contributions of this report

The present work continues earlier work of Skoglar et al. presented in [40]. To make the report at hand selfcontained the background material is presented in chapters 3, 4 and 5. Chapter 2 summarizes the work of Ulvklo et al. [46]. New material is found in Chapters 6 to 11. Chapter 6 applies graph search methods to the UAV surveillance problem. Chapter 7 is revised and rewritten and new material added, e.g. a planning hierarchy is introduced based on the NURBS path representation and optimization aspects are discussed. The Chapters 8 and 9 are substantially expanded since the subjects were presented in one single chapter in [40]. The Section 8.1 is new, and so is the image projection model in Section 8.4.1. In Chapter 9 more advanced occlusion models are presented in Sections 9.3 and 9.2. The simulations in Sections 10.2 and 10.3 are new. Chapter 11 consider theoretical aspects of the information theoretical approach.

2 Path and Sensor Planning for a UAV with EO/IR Sensor

This chapter gives an introduction to the UAV surveillance and reconnaissance problem. In particular, the path and sensor planning challenges in UAV surveillance are discussed. In the current context, we define path planning as planning of the UAV platform trajectory, i.e. path, velocity, etc., whereas sensor planning is defined as planning of a gimballed EO/IR sensor, including gaze direction, focus, zoom, contrast, etc.

2.1 Signal Processing and Surveillance Tasks

The system-oriented research at FOI puts special emphasis on EO/IR image processing and control mechanisms to enhance the level of autonomy in UAV surveillance. Research topics under consideration include:

- Reduction of the amount of data distributed from a sensor node, such as a UAV with EO/IR sensors, so that a network centric system is not overloaded.
- Development of sensor related network services that use advanced sensor data processing to concurrently solve problems such as area coverage, detection, association, tracking, geolocation, change detection, and classification.
- Improvement of data acquisition and sensor data analysis, using network distributed prior knowledge and complementary sensor data in the low level processing of a sensor node with controllable EO/IR sensors.
- Incorporation of real-time sensor data analysis in path planning and sensor management in order to improve the data acquisition process.
- Optimal utilization of surveillance imagery in precision targeting.

The research strives to develop a system architecture for UAV signal processing that incorporates these aspects. Sensor and platform planning are key components in such an architecture. Basic image processing capabilities required to implement such performance enhancements are described in [46].

2.2 Enhanced Levels of Autonomous Planning

The long term goal of the research on UAV surveillance at FOI is a framework for autonomous sensor management, designed to enhance the capability to handle multiple concurrent surveillance requests and raise the level of autonomy. A number of arguments are given here to describe limiting factors in UAV surveillance of today and point out enhanced capabilities with the introduction of higher level of autonomy in sensor and platform planning.

- The ability to manually control the sensor system and laser designator of some UAV systems is limited due to long round-trip latencies in communication link systems (mostly SATCOM related). Automatic tracking and high precision sensor pointing would significantly improve the performance of such systems.

- A human operator is limited to executing only a few tasks in parallel. Fast scheduling between parallel requests requires autonomous sensor control without the human transition time related to context changes. A typical example is a human operator detecting a number of ground targets, zooming in on one and at the same time losing the others out of sight. A fast man-machine interaction using a multi-target tracking system combined with autonomous sensor control would significantly enhance the performance in such a scenario. This is a critical capability when implementing many weapon engagement concepts based on Network Centric Warfare with online target position updates.

- Cueing and sifting mechanisms capable of detecting, tracking, and geolocating multiple ground targets would be of great importance in time critical situations to help the image analyst focus on relevant parts of the surveillance data. Integration of this detection-recognition chain with spatial awareness makes intelligent data acquisition possible by means of active sensor control and path planning. This is of considerable importance for achieving useful autonomous surveillance systems.

- Geolocation of stationary and moving ground targets can improve significantly by including prior knowledge in the estimation process, such as georeferenced imagery and other landmarks. Further improvements can be achieved by integrating the association process with active sensor management. The system will then autonomously schedule sensor resources between different surveillance and geolocation tasks.

- In some advanced future applications, the synchronization between the platform trajectory and the sensor control system will be critical. A typical UAV application related to this problem is low flight altitude surveillance in urban warfare. Very narrow time slots for the sensor control, due to the high degree of occlusion from buildings, require autonomous sensor planning and control to establish guaranteed ground coverage of the streets.

- Successful interaction between multiple UAV surveillance platforms and/or weapon platforms is strongly dependent on a proper representation of the surveillance scene. Decreased time to initiate engagement quality tracks of moving targets would be a direct benefit of sensor data exchange and autonomous sensor and platform planning.

- The ability to dynamically re-plan surveillance missions to accommodate new or updated requests for information will be a basic requirement of autonomous surveillance systems in the future. Being able to re-plan also permits servicing self-initiated surveillance requests based on target indications or other unexpected events in the imagery. This is a required basic component to enable weapon engagement of high-value time critical targets.

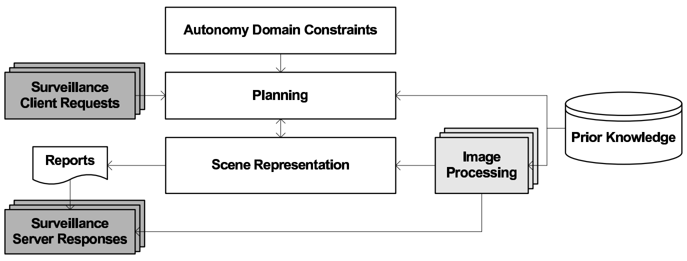

Autonomy Domain Constraints
Surveillance Client Requests
Planning
Prior Knowledge
Reports
Scene Representation
Image Processing
Surveillance Server Responses

Figure 2.1: An overview of necessary signal processing components in autonomous UAV surveillance.

2.3 Surveillance services

Depending on priority, threat level, and flight-time, a new in-flight requested surveillance information service would require allocation of accessible surveillance resources to meet the request for new sensor data. The provider of the surveillance services is the signal processing and control system onboard the UAV itself, or related ground stations, depending on the level of autonomy of the system. However, planning, synchronization and management of surveillance resources over a larger area-of-responsibility is a very demanding procedure.

A client/server approach, designed for managing adaptable surveillance missions, is introduced in [46]. The framework is based on a relationship between information consumers, who dispatch *surveillance client requests*, and a service provider, responsible for *surveillance server responses* (see figure 2.1).

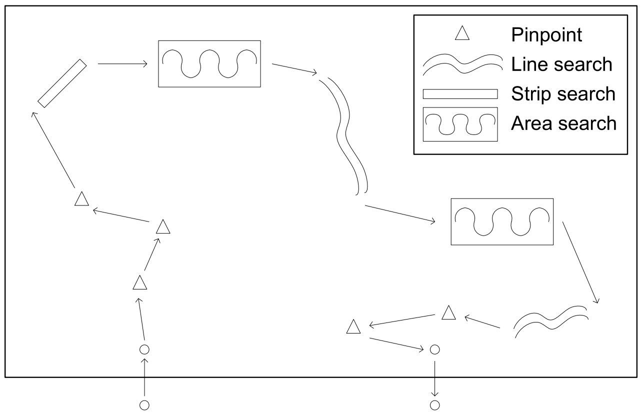


Figure 2.2: Different types of search patterns for surveillance requests in the area of responsibility (AOR).

2.3.1 Surveillance Requests

Surveillance client requests can be categorized using three different criteria: *search pattern*, *task*, and *origin*. The definition of different search patterns we use follows traditional air reconnaissance standards. Given an area of responsibility (AOR), four different surveillance search patterns are defined:

Pinpoint	A limited area around a reference point, known within 100 meters.
Strip search	A task along a straight line between two reference points.
Line search	A task along a communication route, e.g. a railway or road.
Area search	A task over a larger terrain, sea, or urban area.

Figure 2.2 illustrates a surveillance mission with multiple target areas. Given a specific search pattern, four different task categories with associated report requirements are defined:

A: New target	All target features should be reported.
B: Change detection	Subset of all features.
C: Attack planning	Requested features.
D: Damage assessment	Depending on target type.

Only categories A and B are today under consideration in the architecture due to the complexity of C and D. Therefore, there are totally eight combinations of task and search pattern possible in a surveillance request. From a planning point of view, client requests can be generated in three different ways:

Pre-planned	Requests given before take-off of the UAV platform.
In-flight external	New requests given during the implementation of the mission.
In-flight self-initiated	New requests initiated by the system itself. An example is a closer scrutinization of ROIs generated from detection of ground targets.

2.3.2 Surveillance Responses

Depending on available bandwidth and the customer's ability to interpret the imagery, the surveillance client responses consist of surveillance imagery in combination with target meta-data, such as geocoordinates, velocity and target type. Examples are:

- Overviews, such as video mosaics.
- High-resolution multi-view imagery for pinpoint requests.
- High-resolution multi-view imagery of detected ROIs.
- Change detection versus previous flights.
- Recurrent updates of surveillance information.

2.4 Autonomy Domain Constraints

Up to now, autonomous systems have mainly been categorized in terms of capabilities or "intelligence". The Air Vehicle Directorate at AFRL has introduced the notion of *autonomous control level*, ACL, and describe ten such levels, ranging from remotely piloted vehicles to fully autonomous swarms of UCAVs [45, 10]. In [10], Clough argues that the degree of autonomy of a UAV should be expressed in terms of to what extent it can replace humans in the OODA (observe, orient, decide, and act) loop.

In traditional airborne surveillance, air traffic in an AOR is very restricted and controlled. To be able to introduce an autonomous system into the airspace, one must clearly define that system's freedom of action and movement. To that end, we introduce the *Autonomy Domain Constraint*, ADC, which defines the domain of autonomy in terms of freedom of action and movement.

The most restrictive, *zero-level ADC*, permits only basic automatic functions, such as pre-programmed platform control using way-points, and pre-programmed or manual sensor control. The *first level ADC* applies to autonomous control of a sensor system. An example is vision-based feedback for solving multiple concurrent sensing tasks, e.g. multi-target tracking where not all targets are simultaneously in the field of view. Some sort of scheduling mechanism is then required for prioritizing and sequencing concurrent tasks. This level of ADC does not affect the actual platform trajectory, only sensor and signal processing control. In the case of active sensors, e.g. radar or laser, planning may have to take into account the risk of being detected. The *second level ADC* incorporates short-term platform trajectory planning in combination with sensor planning, for single or multiple platforms. The platform adheres to a predefined long term flight plan but modifies it locally in space and time to fit the current task. A flight corridor limits the platform autonomy in space and time. A *third level ADC* also includes long term planning of missions with multiple surveillance client requests. At this level of autonomy, an in-flight generated new request might completely change the ordering between the surveillance tasks of the mission. To summarize:

- ADC 0 Pre-programmed or manual control of platform and sensor.
- ADC 1 Sensor planning and servoing, and pre-programmed platform control.
- ADC 2 Short term platform planning (within free flight corridor) and ADC 1.
- ADC 3 Long term platform planning (within free airspace in AOR) and ADC 2.

2.5 Planning Objectives and Operating Modes

Section 2.3 discusses different objectives, tasks, requests and search patterns. They can be divided into two classes of request modes, *Surveillance & search* and *Tracking & data acquisition.* Surveillance & search in turn can be divided into area search, strip search, line search, and pinpoint. Tracking & data acquisition involves multi-target tracking, precise target coordinate generation, and detailed ROI data acquisition.

To successfully plan and perform these tasks autonomously, some operating modes that facilitate the planning and navigation are also required. Thus, we introduce a *Planning & navigation support* mode that builds and maintains a world model that the planning optimization and navigation estimation are based on. This mode is, for instance, performing probing actions, occlusion estimation, obstacle detection and map building.

Hence, we have three classes of operating modes:

- Surveillance & search
- Tracking & data acquisition
- Planning & navigation support

At the sensor level only one mode is executed at a time. However, several tasks requiring different modes may simultaneously be requesting the sensor resource, and the planning must therefore incorporate some kind of sensor scheduling to allow the system to quickly switch between different modes. The planner does not necessarily require an explicit scheduler; approaches may exist where the scheduling behaviour is a natural part of the planner framework.

2.6 Planning Constraints

The planning optimization process is affected by *planning constraints.* We have identified six classes of constraints:

- *Platform constraints* are associated with the UAV platform, such as dynamic, kinematic, nonholonomic, and fuel constraints.
- *Environmental constraints* define areas where the platform cannot or should not be, and thus include geometric constraints, accessibility constraints, obstacle avoidance, and threat avoidance. Also the autonomy domain constraints, discussed in Section 2.4, belong to environmental constraints.
- *Viewpoint constraints* define areas where visibility is reduced relative some task, for instance due to occlusion, distance, and viewing angle.
- *Sensor constraints* are associated with the gimbal, e.g. dynamic and kinematic constraints, and to the sensor itself, such as field-of-view, resolution, and contrast.
- *Target constraints* are associated with properties that affect the detectability of the target, e.g. target motion and pixels over target.
- *Timing constraints* affect all aspects of planning from platform to sensor.

2.7 Path and Sensor Planning Levels

In Section 2.3 four different search patterns were mentioned. Consider a *line search* example, road surveillance. This involves searching for targets along a road and gathering detailed information (high-resolution images, georeferenced position, etc.) of detected targets. Problems in this surveillance task could be threats and occlusion due to trees, buildings, or terrain masking. The controller must be able to handle uncertainties, such as partially unknown occlusion and road position.

Prior information, e.g. GIS data and prior imagery, Figure 2.3 (a), is useful in the initial planning 2.3 (b), but as the surveillance process progresses it is necessary to look ahead 2.3 (c) and adjust the plan 2.3 (d) due to uncertainties and errors in the prior information. Performance measurements are needed to verify mission success. For instance, a high detection probability can be achieved without necessarily covering every square meter of the road or the ground.

A successful solution to the road surveillance scenarios above should display properties such as probing, caution and reactive behaviour. Probing represents actions to enhance estimation precision in order to improve overall performance in the future. Caution is acting so as to minimize the consequences of erroneous assumptions about the state of the environment. Reactive behaviour means adapting to changes in a dynamic and uncertain environment, e.g. focusing attention on detected targets. Probing and caution are properties of dual controllers as described by, for instance, Maybeck [30].

This discussion motivates a decomposition of planning into the following functional and temporal hierarchy:

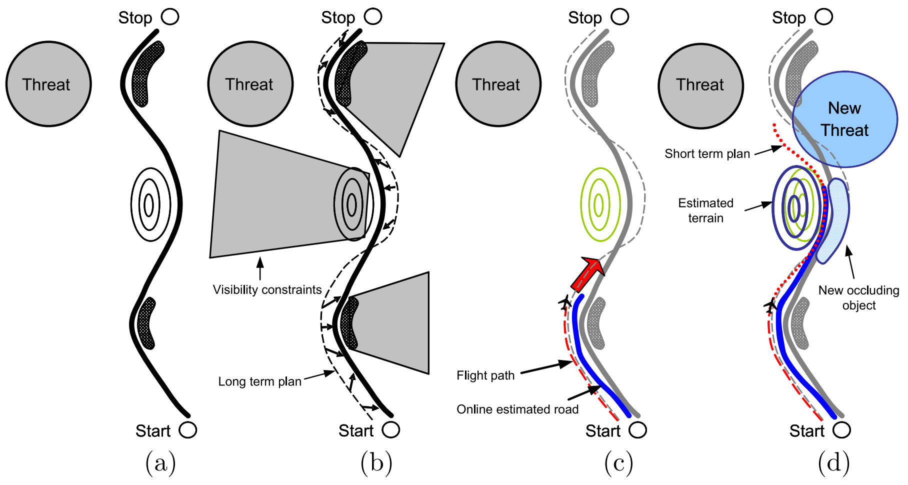


Figure 2.3: Road surveillance scenario. (a) Prior information. (b) Initial plan based on prior information. (c) "Probing", i.e. look ahead and update the world model, is necessary. (d) Replanning is required by new detected visibility and environmental constraints.

1. *Long-term platform path planning* considering prior knowledge, threats, pre-planned surveillance requests and time constraints.
2. *Short-term platform path planning* and *long-term sensor planning*, considering the long-term path plan, trajectory smoothing, detected threats, visibility, occlusion, probing, collision avoidance, and dynamic surveillance requests.
3. *Reactive platform path planning* and *short-term sensor planning*, considering the short-term path plan, trajectory smoothing, occlusion, collision avoidance, and gaze planning.
4. *Reactive sensor planning*, considering focus, zoom, contrast, and gaze in addition to the superior path and gaze plan.

The long-term path planning (level 1) is primarily deterministic and can be computed off-line. This plan might be manually prepared. Also the reactive sensor planning (level 4) may be considered separate from the other levels. However, the levels 2-3 represent a very challenging problem due to their stochastic nature, on-line computational demands, and reliance on sensor data analysis. Also, there is a strong coupling between the sensor and path planning, as well as between the planning levels. Consequently, the planning for levels 2-3 must be considered as one single problem.

In this section we have only considered a line search example. We believe that the discussion here can also be applied to the other search patterns; area, strip, and point.

3 Information Theory and Optimal Control

In this chapter, the optimal control problem based on the information criterion is formulated. First the concept of information is introduced and how it is used in optimal control, and this results in a utility function from information theory, which should be optimized in the optimal control problem.

3.1 Information Theory

Technically, *information* is a measure of the accuracy to which the value of a stochastic variable is known. There are two commonly used formal definitions of information, the *Entropic information* and *Fisher information.*

3.1.1 Entropic Information

Entropic information is defined from *entropy.* The entropy (or Shannon information) $h(\mathbf{x})$ associated with a probability distribution $p(\mathbf{x}) = p_{\mathbf{X}}(\mathbf{x})$, where $\mathbf{X}$ is a random variable, is defined as [27]

$$h(\mathbf{X}) \equiv -E\{\log p(\mathbf{x})\} = -\int_{\mathcal{R}^N} p(\mathbf{x}) \log p(\mathbf{x}) \mathrm{d}\mathbf{x} \tag{3.1}$$

(the continuous case). Entropic information $i(\mathbf{X})$ is simply the negative of the entropy

$$i(\mathbf{X}) \equiv -h(\mathbf{X}), \tag{3.2}$$

i.e. information is maximized when entropy is minimized. When the probability distribution of an n-dimensional state $\mathbf{X}$ is Gaussian, with mean $\mathbf{m}$ and covariance matrix P, the entropic information becomes [16]

$$i(\mathbf{X}) = -h(\mathbf{X}) = -\frac{1}{2} \log\Big((2\pi e)^n \ \det P\Big). \tag{3.3}$$

Of particular interest in estimation theory is the entropy of the posterior distribution $p(\mathbf{x}|\mathbf{Z}^k)$ where $\mathbf{x}$ is the state vector and $\mathbf{Z}^k = \{\mathbf{z}(k), \mathbf{z}(k-1), ..., \mathbf{z}(1)\}$ is the set of all observations up to time k. Using Bayes' theorem

$$p(\mathbf{x}|\mathbf{Z}^k) = \frac{p(\mathbf{z}(k)|\mathbf{x})p(\mathbf{x}|\mathbf{Z}^{k-1})}{p(\mathbf{z}(k)|\mathbf{Z}^{k-1})} \tag{3.4}$$

(assuming $\mathbf{z}(k)$ is independent of $\mathbf{Z}^{k-1}$, conditioned on $\mathbf{X}$) an interesting recursion relation is obtained

$$i(\mathbf{X}|\mathbf{Z}^k) = i(\mathbf{X}|\mathbf{Z}^{k-1}) + E\bigg\{ \log \frac{p(\mathbf{z}(k)|\mathbf{x})}{p(\mathbf{z}(k)|\mathbf{Z}^{k-1})} \bigg\} \tag{3.5}$$

The second term on the right hand side is defined as the information about $\mathbf{X}$ contained in the observation $\mathbf{z}(k)$, or the *mutual information* $I(\mathbf{X}, \mathbf{z}(k))$ of $\mathbf{x}$ and $\mathbf{z}(k)$. Thus, the entropic information following an observation is increased by an amount equal to the information inherent in the observation.

3.1.2 Fisher Information

The (expected) *Fisher information* of a n-dimensional family $p(\mathbf{y}, \theta)$ of distributions at the parameter value $\theta \in \mathcal{R}^n$, is defined as the matrix $\mathcal{I}$ with components

$$\mathcal{I}_{ij}(\theta) \equiv E_\theta\left\{\partial_{\theta_i} \log p \;\; \partial_{\theta_j} \log p\right\} \tag{3.6}$$

where E_θ denotes expectation w.r.t $p(\mathbf{y}, \theta)$. This is well-defined for discrete as well as continuous distributions. For a single continuous distribution $p(\mathbf{y})$ on $\mathcal{R}^n$, this may be applied to the translational family $p_1(\mathbf{y}, \theta) = p(\mathbf{y} - \theta)$ leading to constant (i.e. θ-independent) values

$$\mathcal{I}_{ij} = \int \partial_{\theta_i} \log p_1 \;\; \partial_{\theta_j} \log p_1 \;\; p_1 \; dy = \int \partial_{y_i} \log p \;\; \partial_{y_j} \log p \;\; p \; dy \tag{3.7}$$

This is the case treated in the present report.

Fisher information gives a measure of the amount of information about $\mathbf{X}$ given observations $\mathbf{Z}^k$ up to time k and the probability distribution $p(\mathbf{Z}^k, \mathbf{x})$. The definition of the Fisher information $\mathcal{I}(k)$ is for random state variable $\mathbf{x}$

$$\mathcal{I}(k) = -E\left\{\nabla_x \nabla_x^T \log p(\mathbf{Z}^k, \mathbf{x})\right\} = -E\left\{\nabla_x \nabla_x^T \log p(\mathbf{x}|\mathbf{Z}^k)\right\} \tag{3.8}$$

and for non-random parameters $\mathbf{x}$ the definition is

$$\mathcal{I}(k) = -E\left\{\nabla_x \nabla_x^T \log p(\mathbf{Z}^k|\mathbf{x})\right\} \tag{3.9}$$

The inverse of the Fisher information is the Cramer-Rao lower bound and it is useful in estimation; it bounds the mean square error of any unbiased estimator of $\mathbf{X}$. In the Gaussian distribution case, the Fisher information becomes simply the inverse of the covariance, i.e. $\mathcal{I}(k) = P^{-1}(k|k)$. The relationship between entropic information and Fisher information for a Gaussian distribution is then

$$i(k) = -\frac{1}{2} \log\Big((2\pi e)^n \; \det P(k|k)\Big) = \frac{1}{2} \log\Big((2\pi e)^{-n} \; \det \mathcal{I}(k)\Big). \tag{3.10}$$

In this report we assume Gaussian distributions and the *information matrix* is defined as

$$Y = Y(t) = \mathcal{I}(\mathbf{x}(\mathbf{t})). \tag{3.11}$$

3.2 The Information Filter

The information matrix $Y(t)$ is updated by observations of objects. Let the states to be observed be the vector $\mathbf{x}(t)$ and the observations are in the observation vector $\mathbf{z}(t)$. The observation $\mathbf{z}(t)$ is a function of the observed states $\mathbf{x}(t)$ and some observation noise $\mathbf{v}(t)$ as [16]:

$$\mathbf{z}(t) = \mathbf{h}(\mathbf{x}(t), \mathbf{v}(t)). \tag{3.12}$$

The sensor making the observation is a bearings-only sensor, which can only make observations about the angle to the objects. The function $\mathbf{h}$ is a nonlinear function since it is the observed angle, $\tilde{\varphi}(t)$, which is the real angle, $\varphi(t)$, plus the noise:

$$\mathbf{h}(\mathbf{x}(t), \mathbf{v}(t)) = \tilde{\varphi}(\mathbf{x}(t), \mathbf{v}(t)) = \varphi(\mathbf{x}(t)) + \mathbf{v}(t). \tag{3.13}$$

The observation noise $\mathbf{v}(t)$ is modelled as white noise with covariance R. The nonlinear function is linearized about nominal states $\mathbf{x}_n(t)$ and $\mathbf{z}_n(t)$ as in Grocholsky [16]:

$$\delta\mathbf{z}(t) = H(t)\delta\mathbf{x}(t) + D(t)\mathbf{v}(t). \tag{3.14}$$

Where

$$H(t) = \left.\frac{\partial\mathbf{h}}{\partial\mathbf{x}}\right|_{\substack{\mathbf{x}(t)=\mathbf{x}_n(t)\\ \mathbf{v}(t)=0}} \quad \text{and} \quad D(t) = \left.\frac{\partial\mathbf{h}}{\partial\mathbf{v}}\right|_{\substack{\mathbf{x}(t)=\mathbf{x}_n(t)\\ \mathbf{v}(t)=0}} \tag{3.15}$$

and the new linearized states are

$$\begin{aligned}\delta\mathbf{x}(t) &\equiv \mathbf{x}(t) - \mathbf{x}_n(t)\\ \delta\mathbf{z}(t) &\equiv \mathbf{z}(t) - \mathbf{z}_n(t).\end{aligned}$$

The H-matrix is called the linearized observation matrix and the D-matrix is the linearized observation noise matrix. The reason for the linearization, is that the information filter equations are linear. The linearized filter equations are detailed in Manyika and Durrant-Whyte [28] and simplified in Grocholsky [16], and describes how the *expected information* $I(t)$ is related to the observations, which is

$$I(t) = H(t)^T R^{-1} H(t). \tag{3.16}$$

The information matrix is updated not only by the observed information, but there are losses due to process noise and the system dynamics as well. First consider the system equations

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{w}(t)), \tag{3.17}$$

where $\mathbf{x}(t)$ is the states, $\mathbf{u}(t)$ is known control inputs and $\mathbf{w}(t)$ is the process noise, the latter is assumed to be a zero mean uncorrelated Gaussian process with covariance Q. The function $\mathbf{f}$ is a system of non-linear differential equations, that could be linearized about nominal states $\mathbf{x}_n(t)$ and nominal control signals $\mathbf{u}_n(t)$ as

$$\delta\dot{\mathbf{x}}(t) = F(t)\delta\mathbf{x}(t) + B(t)\delta\mathbf{u}(t) + G(t)\mathbf{w}(t). \tag{3.18}$$

The matrix $F(t)$ is the linearized state transition matrix, $B(t)$ is the linearized input matrix and $G(t)$ is the linearized noise matrix. These are given by

$$F(t) = \left.\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\right|_{\substack{\mathbf{x}(t)=\mathbf{x}_n(t)\\ \mathbf{u}(t)=\mathbf{u}_n(t)\\ \mathbf{w}(t)=0}}, \quad B(t) = \left.\frac{\partial\mathbf{f}}{\partial\mathbf{u}}\right|_{\substack{\mathbf{x}(t)=\mathbf{x}_n(t)\\ \mathbf{u}(t)=\mathbf{u}_n(t)\\ \mathbf{w}(t)=0}} \quad \text{and} \quad G(t) = \left.\frac{\partial\mathbf{f}}{\partial\mathbf{w}}\right|_{\substack{\mathbf{x}(t)=\mathbf{x}_n(t)\\ \mathbf{u}(t)=\mathbf{u}_n(t)\\ \mathbf{w}(t)=0}} \tag{3.19}$$

where

$$\begin{aligned}\delta\mathbf{x}(t) &\equiv \mathbf{x}(t) - \mathbf{x}_n(t)\\ \delta\mathbf{u}(t) &\equiv \mathbf{u}(t) - \mathbf{u}_n(t).\end{aligned} \tag{3.20}$$

Given these matrices, the update law of the information matrix $Y(t)$ is given in [16] as:

$$\dot{Y}(t) = -Y(t)F(t) - F^T(t)Y(t) - Y(t)G(t)Q(t)G^T(t)Y(t) + I(t) \tag{3.21}$$

where $I(t)$ is given in (3.16). This is the continuous version of the prediction and update states of the information filter which is described in [16] and [28]. Since R and Q are positive semi-definite, the process noise cannot gain any information and observation cannot lose information. However, the system dynamics in F could lose or gain information over time.

3.3 Information Matrix and Utility Function

Let

$$Y = Y(t) = \mathcal{I}(\mathbf{x}(t)) \tag{3.22}$$

be the *information matrix* throughout this report. In order to maximize the "information" a *utility function*, giving a scalar value suitable for comparison, is needed. For a discussion of the properties of utility functions, see e.g. [27]. The utility function must combine or weigh the elements or eigenvalues of the information matrix or its inverse. The determinant of the information matrix is an example of a utility function and the problem can be expressed as

$$\max\,(\det Y) \tag{3.23}$$

Two alternatives are maximizing the trace and minimizing the trace of the inverse, i.e.

$$\max\,(\operatorname{tr} Y) \tag{3.24}$$

and

$$\min\,(\operatorname{tr} Y^{-1}) \tag{3.25}$$

respectively.

Let $\{\lambda_1, \ldots, \lambda_n\}$ be the eigenvalues of the information matrix Y. The utility functions in (3.23), (3.24) and (3.25) can then be expressed in terms of the eigenvalues:

$$\det Y = \prod_{i=1}^{n} \lambda_i, \tag{3.26}$$

$$\operatorname{tr} Y = \sum_{i=1}^{n} \lambda_i, \tag{3.27}$$

$$\operatorname{tr} Y^{-1} = \sum_{i=1}^{n} \frac{1}{\lambda_i}, \tag{3.28}$$

respectively. The information matrix has non-negative eigenvalues, since it is defined as the expectation of a non-negative matrix.

In the Gaussian distribution case, the utility function based on the determinant (3.23) is equivalent with maximizing the entropic information, see (3.10). Entropy is a appropriate measure due to its general applicability and resulting preference profiles [27]. However, as can be seen from (3.26), and (3.27), there are drawbacks. A poorly scaled information matrix could give high information even when some eigenvalues are small in comparison. In the object localization case, this corresponds to a position uncertainty that is very small in some directions, but very large in other. Thus, despite large information, the object can not be considered as satisfactory localized. This motivates the use of alternative utility functions, e.g. (3.28). Another utility function that is working on the smallest eigenvalue is

$$\max\,(\min\,(\operatorname{eig} Y)). \tag{3.29}$$

However, this alternative works bad when the problem grows and several objects are included in the model.

3.4 Optimal Control

3.4.1 The General Optimal Control Problem

The general optimal control problem could be expressed simply as "choose the control signal such that the system behaves as good as possible" [15]. In mathematical terms, the problem can be formulated as in Glad and Ljung [15]:

Given initial conditions:

$$\mathbf{x}(0) = \mathbf{x}(t_0) \tag{3.30}$$

System equations:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{3.31}$$

Subject to constraints:

$$\psi(\mathbf{x}(t), \mathbf{u}(t)) = 0 \tag{3.32}$$
$$\mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)) \leq 0 \tag{3.33}$$

The criterion function to minimize is:

$$J(\mathbf{x}(t), \mathbf{u}(t)) = \phi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t))\ dt \tag{3.34}$$

The solution to the optimal control problem would be:

$$\min_{\mathbf{u}(t)} J(\mathbf{x}(t), \mathbf{u}(t)), \tag{3.35}$$

subject to the constraints (3.32) and (3.33).

3.4.2 Optimal Control In Path Planning

The criterion function J in (3.34) is the utility function discussed in Section 3.3, if the utility function is modified such that it is minimized. A maximum problem, could always be converted into a minimum problem,

$$\max g(x) = -\min(-g(x)), \tag{3.36}$$

and in the implementation, the utility functions that required a maximization, like maximizing the determinant, were converted into minimum problems.

The solution to the optimal control problem is affected by the choice of utility function and the idea is to find a control signal, or sequence of control signals, that maximizes information. However, the optimization is done over a pre-defined optimization time, called the *time horizon* denoted by t_f, and it is only of interest to consider the information at the time horizon t_f. Therefore the criterion function is a function of the time horizon only and L in (3.34) is equal to zero, i.e.

$$J = J(t_f) = \phi(\mathbf{x}(t_f)). \tag{3.37}$$

Most optimal control problems require a numerical solution and by parametrizing the control signal into m steps in every optimization, an approximate solution will be found, for details see Bertsekas [8]. The idea is now to find a sequence of control steps u_i that maximizes information at the time horizon t_f as in Grocholsky [16]:

$$\mathbf{u}_i(t) = \mathbf{p}_i \chi_i(t), \quad i = 1, \ldots, m. \tag{3.38}$$

Where χ_i simply holds the control variable over m equal time steps Δt_u as

$$\chi_i(t) = \begin{cases} 1 & \text{if } (j-1)\Delta t_u \leq t \leq j\Delta t_u \\ 0 & \text{otherwise} \end{cases}, \quad \Delta t_u = \frac{t_f - t_0}{m}. \tag{3.39}$$

The optimal control problem in (3.35) is now converted into a nonlinear programming problem [16]:

$$\min_{\mathbf{p}} J(\mathbf{p}) = \phi(\mathbf{x}(t_f)) + \frac{1}{2}\Delta t_x \sum_{i=1}^{m\cdot n_{steps}} (L_i(\mathbf{x}_i, \mathbf{u}_k) + L_{i-1}(\mathbf{x}_{i-1}, \mathbf{u}_k)), \tag{3.40}$$

subject to the constraints in (3.32) and (3.33), where $\mathbf{p} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$ is the parameter vector and $\Delta t_x = \frac{\Delta t_u}{n_{steps}}$, $\{n_{steps} \geq 1\}$ is the time between the evaluations of the states, and k is the control index. The original optimal control problem is now in the form of mathematical programming problem [16]. In the path planning problem, L is zero, but it is given for generality.

There are no equality constraints as in equation (3.32), but there could be inequality constraints as in equation (3.33), with the control signals bounded as

$$u_{min} \leq u_i \leq u_{max}, \tag{3.41}$$

since it is reasonable that the movement of a UAV is restricted by its dynamics.

In Chapters 4 and 5 the optimal control problem is solved by *Matlab*'s Optimization Toolbox. If there are no bounds on the control signal, the problem is considered to be a unconstrained problem, solved by the function *fminunc.* When the problem is constrained, the problem is solved by the function *fmincon.* A constrained problem can also be solved as a unconstrained optimization problem if the bounds are included as penalties in the utility function.

3.4.3 Gradient Determination

The functions in *Matlab*'s toolbox uses the gradient $\nabla_p J$ and the Hessian $\nabla^2_{\mathbf{p}} J$ in order to find the local minimum. The use of the gradient will help the optimizer to reach a minimum in terms of efficiency and reliability. This can be done in two ways, either by letting the optimizer calculate the gradient and Hessian itself as a numerical solution, or by giving the analytical expressions explicitly. The first method requires no knowledge of the partial derivatives with respect to state and control vectors. However, the drawback is the computational load. For the latter method, the details are given here for the gradient, the details for calculating the Hessian are given in Grocholsky [16].

Differentiating (3.40) ($L = 0$) with respect to $\mathbf{p}$ gives:

$$(\nabla_p J) = \frac{\partial \phi(\mathbf{x}(t_f))}{\partial \mathbf{x}(t_f)} \frac{\partial \mathbf{x}(t_f)}{\partial \mathbf{p}} \tag{3.42}$$

The first part of (3.42) is simply the derivative of the criterion function with respect to each state at the time horizon t_f. The second part is derived from (3.31) by applying the chain rule, which gives:

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial \mathbf{x}}{\partial \mathbf{p}} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial \mathbf{p}} + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \mathbf{p}}, \text{ with } \left.\frac{\partial \mathbf{x}}{\partial \mathbf{p}}\right|_{t=t_0} = \frac{\partial \mathbf{x}_0}{\partial \mathbf{p}} \tag{3.43}$$

By the use of a Heun scheme, the first order sensitivities could be calculated as:

$$\frac{\partial \mathbf{x}_i}{\partial \mathbf{p}} = \left[\mathrm{I}_n - \frac{1}{2}\Delta t_x \frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i}\right]^{-1} \left[\left[\mathrm{I}_n + \frac{1}{2}\Delta t_x \frac{\partial \mathbf{f}_{i-1}}{\partial \mathbf{x}_{i-1}}\right] \frac{\partial \mathbf{x}_{i-1}}{\partial \mathbf{p}} + \frac{1}{2}\Delta t_x \left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{u}_k} + \frac{\partial \mathbf{f}_{i-1}}{\partial \mathbf{u}_k}\right) \frac{\partial \mathbf{u}_k}{\partial \mathbf{p}}\right],$$

where I_n is a $(n \times n)$ identity matrix, Δt_x is the time between each state evaluation, and $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$ are the derivatives of the system equations with respect to each state and $\frac{\partial \mathbf{f}}{\partial \mathbf{u}}$ with respect to the control signal.

4 Optimal Path for Bearings-only Localization

In this chapter a single vehicle is considered, and the task is to localize a feature in the xy-plane with a bearings-only sensor. This is done by seeking control action of the vehicle that maximizes the information as described in Chapter 3.

The examples illustrate information as a performance metric and the effect of varied optimization time horizons. Different utility functions are evaluated and the difference between analytical implementation of the gradient and numerical calculation is examined. First the third dimension is ignored and the vehicle and the feature is defined in the xy-plane, then a constant altitude of flight is added. Also the case with n objects is considered, where the path is planned such that all objects are to be localized simultaneously.

4.1 System Models

This 2D single object localization example and the models are from Grocholsky [16]. The vehicle and the object are here assumed to be in the same plane.

4.1.1 Sensor Platform Model

The sensor is attached to a sensor platform (also called vehicle) moving in the xy-plane with constant velocity V. The location $[x \quad y]^T$ and the direction of the vehicle are described by the state $\mathbf{x}_s(t)$, Figure 4.1. The vehicle's heading

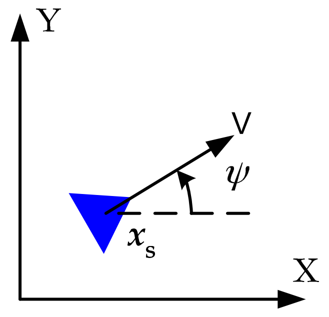


Figure 4.1: 2D vehicle model

is modelled by ψ, the angle between the head of the platform and the X-axis. The rate of change of the platform heading $\dot{\psi}$ is the control variable. The equations describing the sensor platform are summarized as

$$\mathbf{x}_s(t) = \begin{bmatrix} x(t) \\ y(t) \\ \psi(t) \end{bmatrix}, \quad \mathbf{x}_s(0) = \mathbf{x}_s^0$$

$$u(t) = \dot{\psi} \tag{4.1}$$

$$\dot{\mathbf{x}}_s(t) = \begin{bmatrix} V\cos(\psi(t)) \\ V\sin(\psi(t)) \\ u(t) \end{bmatrix}.$$

In real life applications the coordinates of the vehicle is given by GPS/INS and navigation uncertainty should be included in the vehicle model. However, adding this to the model yields a much more complex problem to solve and is saved for future work.

4.1.2 Feature Model

The feature is represented by a stationary point $\mathbf{x}_f = [x_f \quad y_f]^T$ in the xy-plane. The uncertainty of the location is captured in the covariance of a two dimensional Gaussian distribution $P_f(t)$. In the information filter, this is represented by an information matrix $Y(t)$ as the inverse of the covariance as

$$Y(t) = P_f^{-1}(t). \tag{4.2}$$

Since a Gaussian distribution is assumed, the entropic information and the Fisher information are the same according to (3.10), and there is no need to distinguish between them.

The feature process model is

$$\dot{\mathbf{x}}_f(t) = \omega(t), \tag{4.3}$$

where the process noise $\omega(t)$ is a zero mean Gaussian process with uncorrelated covariance $Q(t)$. The process noise may have been omitted, if the object is stationary, but is included to allow flexible estimator design. The impact of incorrect information will be lower and the effect of the new information gained will be stronger.

The position uncertainty can be represented by an ellipse about the estimated location of the object. When planning the next optimization step, the path is calculated under the assumption that the true location of the object is the estimated location. This is not true in a real life application, since the object position is not known and could be in the outer range of its uncertainty ellipse, and the path would then not be optimal. This problem must be considered in future work. Instead of having a Gaussian distribution representing the uncertainty, one could instead use a sum of Gaussian with different weights which gives a more accurate model.

4.1.3 Sensor Model

The vehicle carries a sensor that makes observations of the feature. The observation is the bearing of the feature, i.e. the relative angle to the feature, which could be calculated as the angle from the X-axis to the feature θ minus ψ as shown in Figure 4.2. The observation model is then, from (3.12) and (3.13),

$$z(t) = h(\mathbf{x}_f, \mathbf{x}_s) \tag{4.4}$$

$$h(t) = \theta(t) - \psi(t) + \nu(t) = \arctan_2\left(\frac{y_f - y_s}{x_f - x_s}\right) - \psi(t) + \nu(t) \tag{4.5}$$

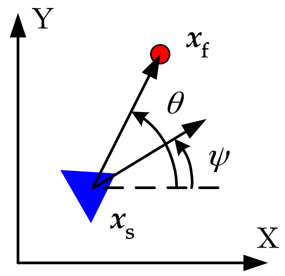


Figure 4.2: 2D observation model

where $\nu(t)$ is a zero mean uncorrelated Gaussian process with variance $R = \sigma^2$. Taking the Jacobian with respect to the feature state gives the linearized relationship between the sensed output and the states according to (3.15):

$$\begin{aligned} H(t) &= \nabla_{\hat{\mathbf{x}}_\mathbf{f}} \mathbf{h}(\mathbf{x}_f, \mathbf{x}_s) \\ &= \left[\frac{-(\hat{y}_f - y_s(t))}{(\hat{x}_f - x_s(t))^2 + (\hat{y}_f - y_s(t))^2}, \frac{\hat{x}_f - x_s(t)}{(\hat{x}_f - x_s(t))^2 + (\hat{y}_f - y_s(t))^2}\right] \\ &= \frac{1}{\hat{r}(t)}\left[-\sin\hat{\theta}(t), \cos\hat{\theta}(t)\right], \end{aligned} \tag{4.6}$$

with $(\hat{x}_f, \hat{y}_f)$ as the estimated feature location, and $(\hat{r}, \hat{\theta})$ estimation of (r, θ) respectively. The resulting observed information is derived according to (3.16) as

$$I(t) = H^T(t)R^{-1}H(t). \tag{4.7}$$

4.1.4 System Equations

The state of the system consists of the platform model and the information matrix representing the uncertainty of the feature. The update of the vehicle states $\mathbf{x}_s$ is given in (4.1). The update of the information matrix is given by (3.21). A comparison between the feature model in (4.3) with the general expression in (3.18), yields $F(t) = 0$ and $G = I$ (and $B = 0$). The update of information is now reduced to

$$\dot{Y}(t) = -Y(t)QY(t) + I(t). \tag{4.8}$$

The rate of change in information is some loss due to process noise and the gain of information by observation. The matrices $Y(t)$ and $I(t)$ are symmetric and Q is a diagonal matrix as

$$Y(t) = \begin{bmatrix} \mathrm{Y}_x & \mathrm{Y}_{xy} \\ \mathrm{Y}_{xy} & \mathrm{Y}_y \end{bmatrix}, \; I(t) = \begin{bmatrix} \mathrm{I}_x & \mathrm{I}_{xy} \\ \mathrm{I}_{xy} & \mathrm{I}_y \end{bmatrix} \text{ and } Q = \begin{bmatrix} \mathrm{Q}_x & 0 \\ 0 & \mathrm{Q}_y \end{bmatrix}. \tag{4.9}$$

The feature information matrix is symmetric and it is therefore sufficient to calculate three of four values, which means that the states representing the information would be

$$\mathbf{x}_{info} = \begin{bmatrix} \mathrm{Y}_x \\ \mathrm{Y}_{xy} \\ \mathrm{Y}_y \end{bmatrix}. \tag{4.10}$$

The equations derived for the evolution of the feature information combined with the equations of the vehicle dynamics describe fully the system state and the augmented system equations become

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\mathbf{x}}_s \\ \dot{\mathbf{x}}_{info} \end{bmatrix} = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\psi}(t) \\ \dot{\mathrm{Y}}_x(t) \\ \dot{\mathrm{Y}}_{xy}(t) \\ \dot{\mathrm{Y}}_y(t) \end{bmatrix} = \begin{bmatrix} V\cos(\psi(t)) \\ V\sin(\psi(t)) \\ u(t) \\ -\mathrm{Y}_x^2(t)\mathrm{Q}_x - \mathrm{Y}_{xy}^2(t)\mathrm{Q}_y + \mathrm{I}_x(t) \\ -\mathrm{Y}_x(t)\mathrm{Q}_x\mathrm{Y}_{xy}(t) - \mathrm{Y}_{xy}(t)\mathrm{Q}_y\mathrm{Y}_y(t) + \mathrm{I}_{xy}(t) \\ -\mathrm{Y}_{xy}^2(t)\mathrm{Q}_x - \mathrm{Y}_y^2(t)\mathrm{Q}_y + \mathrm{I}_y(t) \end{bmatrix}. \tag{4.11}$$

The task is to reduce the uncertainty of the feature by maximizing information. Introduce for example the determinant as utility function $J(t)$ according to (3.26) as

$$J(t_f) = \det(Y(t_f)) = \mathrm{Y}_x(t_f)\mathrm{Y}_y(t_f) - \mathrm{Y}_{xy}^2(t_f). \tag{4.12}$$

and maximize it at the time horizon t_f.

4.1.5 Analytical Gradient

If the gradient would be given analytically, recall Section 3.4.3, the gradient of the utility function in (4.12) would be

$$\frac{\partial\phi(\mathbf{x}(t_f))}{\partial\mathbf{x}(t_f)} = \begin{bmatrix} 0 & 0 & 0 & \mathrm{Y}_y & -2\mathrm{Y}_{xy} & \mathrm{Y}_x \end{bmatrix}. \tag{4.13}$$

The gradient of the systems equations (4.11) is needed, and is given by

$$\frac{\partial\mathbf{f}}{\partial\mathbf{x}} = \left[\begin{matrix} 0 & 0 & -V\sin(\psi) & \dots \\ 0 & 0 & V\cos(\psi) & \dots \\ 0 & 0 & 0 & \dots \\ \frac{4(\hat{y}_f-y)^2(\hat{x}_f-x)}{\sigma^2r^6} & \frac{4(\hat{y}_f-y)^3}{\sigma^2r^6} - \frac{2(\hat{y}_f-y)}{\sigma^2r^4} & 0 & \dots \\ -\frac{4(\hat{y}_f-y)(\hat{x}_f-x)^2}{\sigma^2r^6} + \frac{(\hat{y}_f-y)}{\sigma^2r^4} & -\frac{4(\hat{y}_f-y)^2(\hat{x}_f-x)}{\sigma^2r^6} + \frac{(\hat{x}_f-x)}{\sigma^2r^4} & 0 & \dots \\ \frac{4(\hat{x}_f-x)^3}{\sigma^2r^6} - \frac{2(\hat{x}_f-x)}{\sigma^2r^4} & \frac{4(\hat{y}_f-y)(\hat{x}_f-x)^2}{\sigma^2r^6} & 0 & \dots \end{matrix}\right.$$

$$\left.\begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -2\mathrm{Y}_x\mathrm{Q}_x & -2\mathrm{Y}_{xy}\mathrm{Q}_y & 0 \\ -\mathrm{Y}_{xy}\mathrm{Q}_x & -\mathrm{Y}_x\mathrm{Q}_x - \mathrm{Y}_y\mathrm{Q}_y & -\mathrm{Y}_{xy}\mathrm{Q}_y \\ 0 & -2\mathrm{Y}_{xy}\mathrm{Q}_x & -2\mathrm{Y}_y\mathrm{Q}_y \end{matrix}\right] \tag{4.14}$$

where $[x \quad y]^T$ is the vehicle's position, $[\hat{x}_f \quad \hat{y}_f]^T$ is the feature's estimated position and r is the distance between the vehicle and the target, and

$$\frac{\partial\mathbf{f}}{\partial u} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T. \tag{4.15}$$

These are the equations needed to calculate the gradient $\nabla_{\mathbf{p}}J$ according to (3.42). One could realize that if the model is further complicated, it would be very difficult to derive the expressions needed for the analytical gradient.

4.2 Information vs. Detection and Identification

Throughout this report, the task of the UAV is to detect, localize and identify objects on the ground. To represent the ability to identify targets, the concept of information is introduced in the sense "the more information about an object, the more likely is the identification". When an object is said to be identified, there are sufficient number of pictures of the object that it could be identified with methods from image analysis. There are no real images in this work, instead the amount of information is connected to the uncertainty of the target's geometrical location. The terms"identify" and"detect" an object would now be that the information value of the object is larger than pre-defined threshold values.

4.3 Simulations

The initial conditions in (4.1) give the starting position of the vehicle and the starting angle as the angle between the heading of the vehicle and the x-axis. Let the vehicle start at the origin, and specify a desired starting angle. In the simulation examples here the starting angle was set to $\pi/2$ [rad]

$$\mathbf{x}_s(0) = \begin{bmatrix} 0 & 0 & \pi/2 \end{bmatrix}^T .$$

There must also be some starting information, since in order to plan how to localize a target some information is needed. In other words, there is a need to know that the target exists, otherwise it is hard to plan the path. The settings used in the simulations are:

$\mathbf{x}_f = [10 \quad 10]\ m$	Feature location (stationary).
$V = 1\ m/s$	Speed.
$R = \sigma^2, \quad \sigma = 2.5^\circ$	Observation noise.
$Y(0) = \mathrm{I}_{2\times 2} \cdot 10^{-3}$	Starting information.
$Q = \mathrm{I}_{2\times 2} \cdot 10^{-6}$	Process noise.
$m = 2$	Number of parameterized control signals in each optimization step.
$t_f = 1\ s$	Optimization time horizon.
$\lvert u_i \rvert \leq 1\ rad/s$	Control signal bounds.
$J(t_f) = \det(Y(t_f))$	Utility function.

These values are chosen for simplicity and are not values for real applications, it is the principle that is interesting. The resulting path is presented in Figure 4.3 and the observed information and the parameterized control signal are shown in Figure 4.4. The control signal is bounded and then the optimization is solved by the function *fmincon* in *Matlab Optimization Toolbox* [29].

The solution is calculated as following. In each step, a control signal, parameterized into m equal long parts, will be calculated such that the information at the time horizon t_f is maximized, given the system equations and subject to constraints. The first optimum will be a trajectory from the origin of length $V \cdot t_f = 1\ m$, and this point is reached by the optimal control signal consisting of $m = 2$ steps. The first part of the control signal u_1 is valid between the time 0 and $t_f/2$ and the next part u_2 is valid between $t_f/2$ and t_f. After the first optimum is reached, and the states $\mathbf{x}$ have been updated, a new optimization is done. The uncertainty of the feature location is plotted as an ellipse about the feature, and the procedure is terminated when the uncertainty is so low that the object could be said is localized. The total number of optimizations are the number of 'x':s in the figure, a total of 25, that is the number of optimizations required for localizing the target in this case. The target is said to be localized if the information is higher than a pre-defined value.

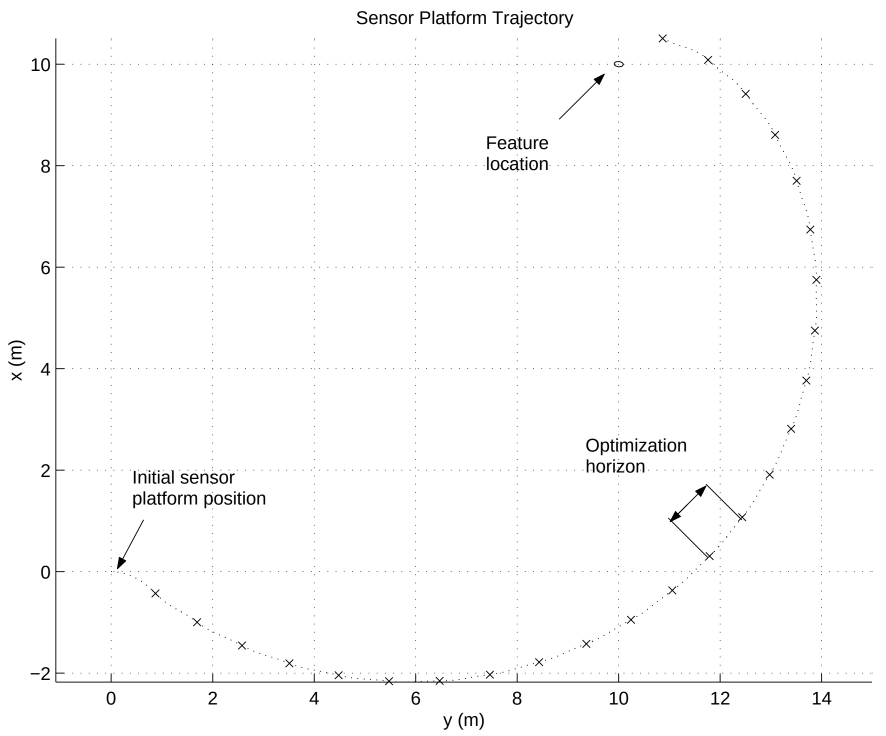


Figure 4.3: Trajectory of the sensor platform. Each 'x' marks a new optimization. The feature location's uncertainty is an ellipse about its true location.

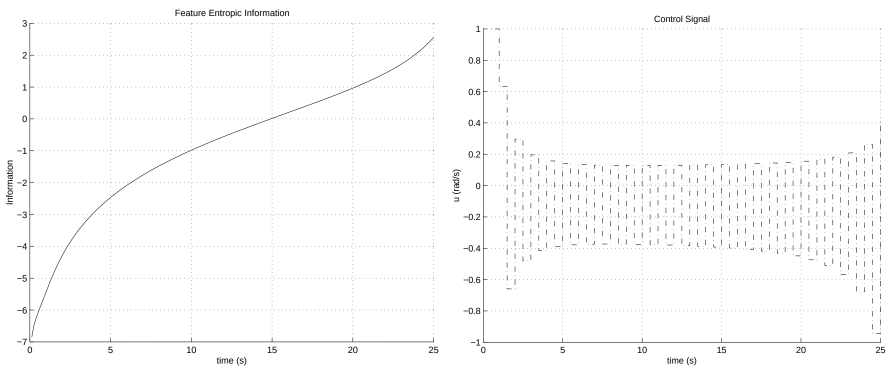


Figure 4.4: Feature entropic information and the parameterized control signal.

The path is shaped as a spiral, which is reasonable since the sensor is a bearings-only sensor. In the beginning, the planning steps are approximately in the orthogonal direction with respect to the object. In addition, the sensor platform is also moving closer to object.

The path is the planned path for the sensor platform. However, since there are no uncertainties of the platform's position, it will also be the performed path by the platform. The control signal varies all the time and it is undesirable, since the actuators would be worn out. However, it is understandable that the solution gives a varying control signal, since it plans very short ahead. To avoid this problem, one could formulate penalty functions on varying control signals or use a different control signal parametrization.

The calculation time is highly dependent on the number of control parameterizations m. The calculation time is also dependent on the tolerance chosen for the optimization and the ODE solver.

The use of information as a performance index is intuitive, since the more information you have of a feature, the more certain you are of its location. It is also a convenient criterion for the optimization process, since when the information is maximized, the uncertainty of the feature location is minimized.

4.3.1 Plotting the Criterion Function

To illustrate the complexity of the optimization problem, the criterion function could be plotted over the first optimization step, that is over the first parameterized control signal $[u_1\ u_2]$.

Recall the criterion function used here as the determinant of the information matrix from (4.12)

$$J(t_f) = \mathrm{Y}_x(t_f)\mathrm{Y}_y(t_f) - \mathrm{Y}_{xy}^2(t_f).$$

This function is plotted for two cases of bound on the control signal, first bounded as $-10\ rad/s \leq u_i \leq 10\ rad/s,\ i = 1,2$, plotted in Figure 4.5. The optimizer is trying to find the maximum, and the problem is that there are several local maxima, and there is a risk that the optimizer will find a local maximum instead of the global. For this example, tighten the bound on the control signal to $-1\ rad/s \leq u_i \leq 1\ rad/s,\ i = 1,2$, for the same situation yields the situation in Figure 4.6. The problem of many maxima is not solved by this, but there are fewer maximums than for the case with looser bounds.

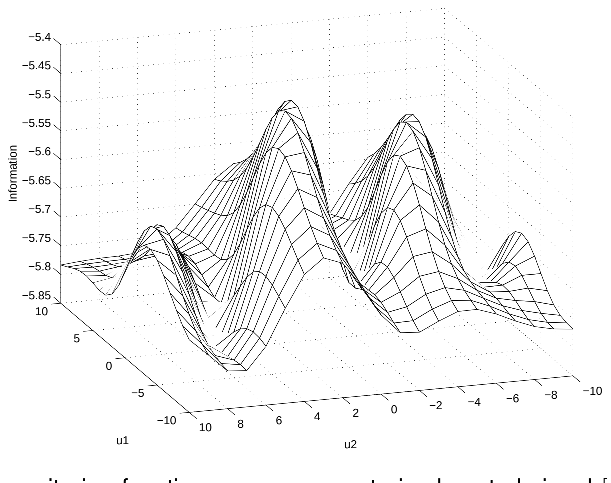


Figure 4.5: The criterion function over a parameterized control signal $[u_1\ u_2], -10 \leq u_i \leq 10$.

The reason why the criterion function was plotted is to illustrate that the optimization is not perfect, but since it is an application of engineering, one would be satisfied with a good solution and not the best in the theoretical sense. The Matlab Optimization Toolbox functions *fminunc* and *fmincon* use line search when they cannot solve the problem otherwise. The line search algorithm has problem with local maxima, and instead one should implement and test other optimization methods, see Chapter 7.

4.3.2 The Effect of Optimization Time Horizon

The sensor platform's trajectory is affected by the time horizon t_f in the optimization. Figure 4.7 shows a comparison between three different time horizons,

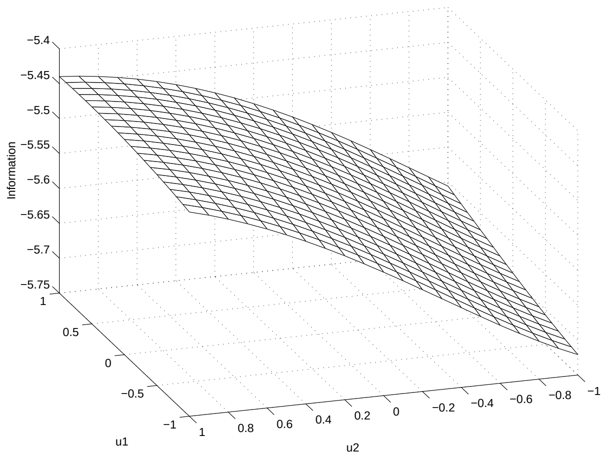


Figure 4.6: The criterion function over a parameterized control signal $[u_1\ u_2], -1 \leq u_i \leq 1$.

(1 s) (16 optimization steps), (4 s) (4 optimization steps) and (8 s) (2 optimization steps). Thus, the total time would be 16 s for all cases. The cases can be summarized in Table 4.1.

Case	1	2	3
Time horizon $t_f(s)$	1	4	8
Number of optimizations	16	4	2
Total time (s)	16	16	16
Control parameters m	2	4	8

Table 4.1: Details of the three cases used to investigate the effect of optimization time horizon.

As can been seen, the platform with long optimization time travels more directly to the feature. It will not gain as much information in the beginning as in the case with short optimization time, but since it plans further in the future, the information at the time horizon will increase as seen in Figure 4.8. According to Figure 4.8, a long time horizon is preferable compared to a short time horizon. But there are problems with a long horizon. The calculation time was different for the three cases. The computation time of the second and third case was 30% and 350% longer than for the first case, respectively.

4.3.3 The Effect of Prior Information

In the simulations, the starting information was set to 10^{-3} in both the x- and y-direction. The starting information is a measure of how much is known about the object when planning the first step in terms of uncertainty. When knowing more about the object, i.e. the starting information was set to 1 in both directions, the resulting path is seen in the left part of Figure 4.9. It will travel in the direction of the object at first, since it will not gain that much information by taking a step orthogonally. Instead it needs to move closer, which it does until one point where it turns and continues like in the first simulation in Figure 4.3.

In the simulation shown in the right part of Figure 4.9 the starting information was set to 1 in the x-direction and 10^{-3} in the y-direction. This is the

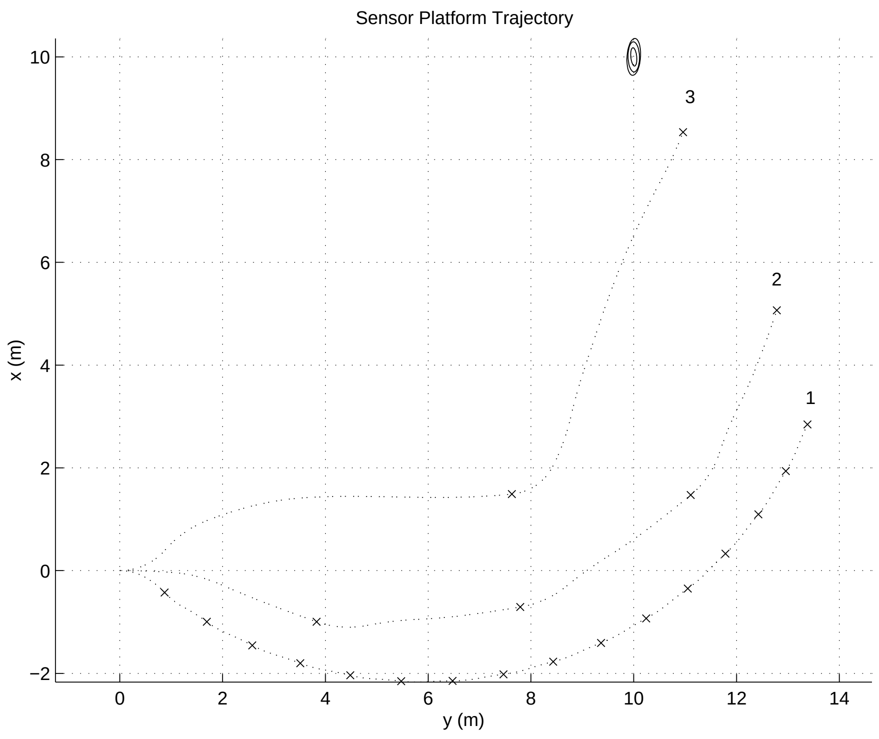


Figure 4.7: The trajectories for platforms with different optimization times for the three cases in Table 4.1.

case when knowing one coordinate of the object's position and it is needed to reduce the uncertainty in the other direction. The vehicle is trying to reduce the uncertainty in the y-direction by moving in the direction which is orthogonal to the initial uncertainty, that is in the y-axis. The vehicle continues until the uncertainty in the y-direction is about the same as the x-axis and it turns to reduce the uncertainty in both directions.

4.3.4 Evaluation of the Utility Function

In the simulations below, three different utility functions are evaluated. The details of the different simulations are given in Table 4.2. The first one is maximizing the determinant of the information matrix, and tried for two cases where the gradient of the utility function is given analytically, case 1, and by letting the optimization function approximate the gradient numerically, case 2. The optimization is made by the function *fmincon*, where it is possible for the user to choose whether the gradient is given explicitly or not. Case 3 is the path from the trace of the information matrix's inverse and case 4 is maximizing the minimum of the eigenvalues. The resulting paths are shown in Figure 4.10.

All paths have the same basic shape, a spiral. The time horizon was set to 1 s for simplicity and except for the utility function and gradient, all other simulation parameters are the same. The entropic information (i.e. $\det Y$) values are plotted in Figure 4.11 and not very surprising $\det Y$-utility gives the largest final information value. However, this is an unfair comparison tool and it is difficult to decide which utility function that solves the localization problem best. However, the analytical and numerical implementation of the gradient in the $\det Y$ case can be compared. The analytical implementation is better, since information is higher, but the difference is small. The major advantage of

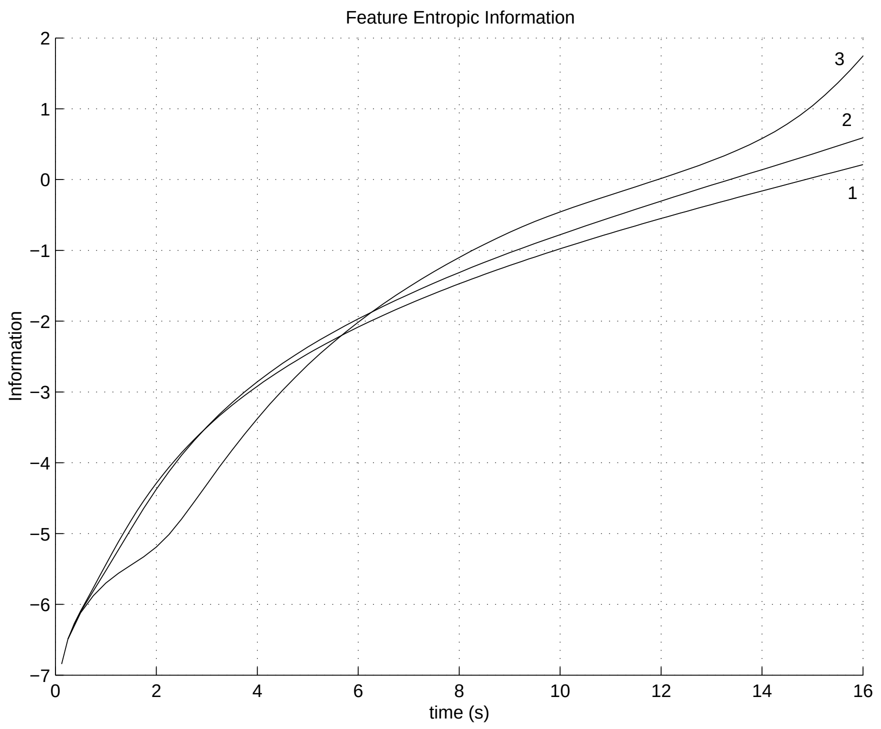


Figure 4.8: The information for platforms with different optimization times for the three cases in Table 4.1.

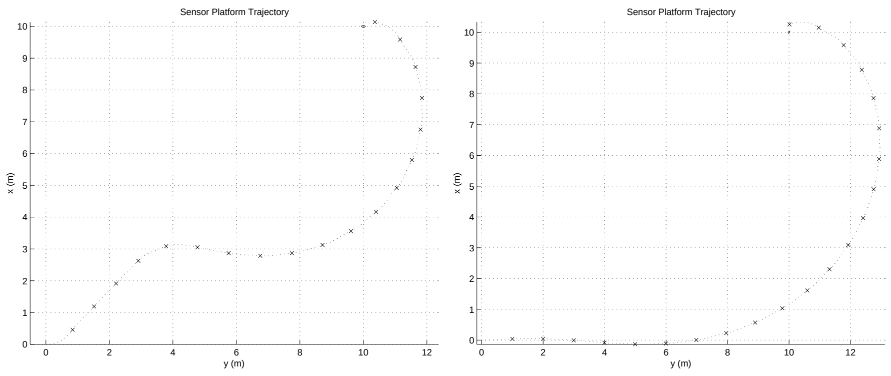


Figure 4.9: *Left:* The effect of high starting information. *Right:* High starting information in x-direction and low starting information in y-direction.

the analytical case is the computational time, but when the model gets more complicated, it is difficult or impossible to derive the analytical expression explicitly.

4.4 n Objects

Following the methods of this chapter, it seems that the vehicle could successfully localize an object. To make things more interesting, the vehicle must be able to handle the case of n objects on the ground.

Case	1	2	3	4
Utility function	$\max(\det Y)$	$\max(\det Y)$	$\min \operatorname{tr} Y^{-1})$	$\max(\min(\operatorname{eig} Y))$
Gradient computation	Analytical	Numerical	Numerical	Numerical

Table 4.2: Details of the four cases used to investigate the effect of different utility functions.

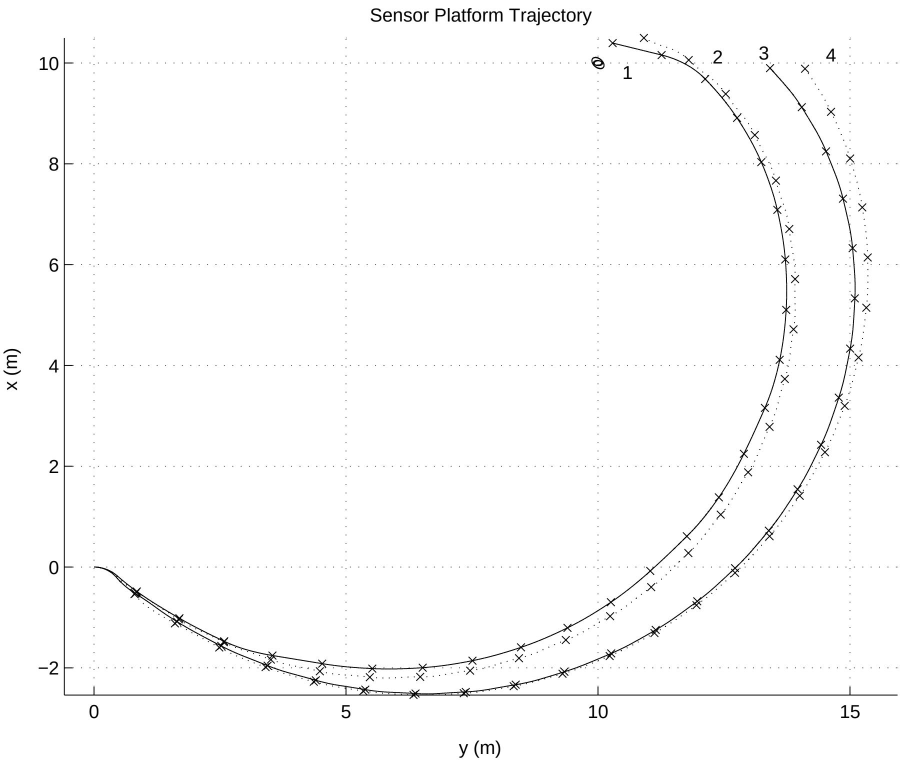


Figure 4.10: Sensor platform trajectory calculated from different utility functions: 1) $\max(\det Y)$ with gradient analytically, 2) $\max(\det Y)$ with gradient numerically, 3) $\min(\operatorname{tr} Y^{-1})$, 4) $\max(\min(\operatorname{eig} Y))$

4.4.1 Modelling two Objects

First the model should be extended to the case with two objects. A symmetric information matrix Y is now of (4×4) as:

$$Y(t) = \begin{bmatrix} \mathrm{Y}_{x_1} & \mathrm{Y}_{x_1 y_1} & \mathrm{Y}_{x_1 x_2} & \mathrm{Y}_{x_1 y_2} \\ \mathrm{Y}_{x_1 y_1} & \mathrm{Y}_{y_1} & \mathrm{Y}_{x_2 y_1} & \mathrm{Y}_{y_1 y_2} \\ \mathrm{Y}_{x_1 x_2} & \mathrm{Y}_{x_2 y_1} & \mathrm{Y}_{x_2} & \mathrm{Y}_{x_2 y_2} \\ \mathrm{Y}_{x_1 y_2} & \mathrm{Y}_{y_1 y_2} & \mathrm{Y}_{x_2 y_2} & \mathrm{Y}_{y_2} \end{bmatrix}. \tag{4.16}$$

With just one more object, there were seven new states introduced. There is now three states representing the vehicle, same as in (4.1) and ten states from the information matrix above. The question arises whether some states, especially the cross states $\mathrm{Y}_{i_1 j_2}$, could be zero. Recall the information rate of change from (4.8) as

$$\dot{Y}(t) = -Y(t)QY(t) + I(t), \tag{4.17}$$

and the same observation model as for one object from (4.4) now extended as

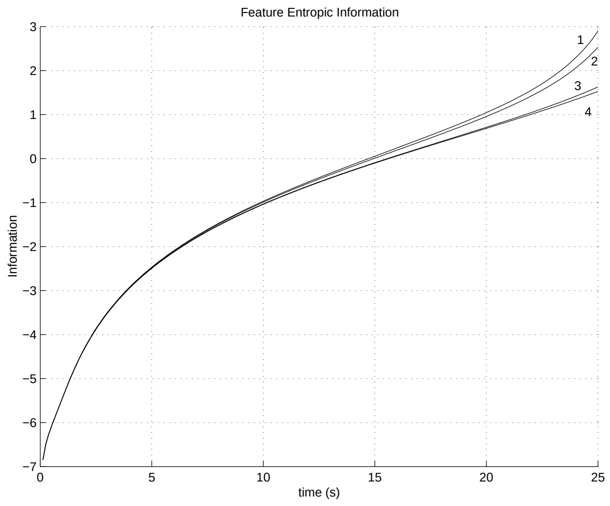


Figure 4.11: Information from the different utility functions: 1. $\max(\det Y)$ with gradient analytically, 2. $\max(\det Y)$ with gradient numerically, 3. min(traceY^{-1}), 4. max(min(eig$Y$))

$$\begin{aligned} \mathbf{z}(t) &= [h_1(\mathbf{x}_{f_1}, \mathbf{x}_s), h_2(\mathbf{x}_{f_2}, \mathbf{x}_s)]^T \\ h_1(t) &= \theta_1(t) - \psi(t) + \nu_1(t) \\ h_2(t) &= \theta_2(t) - \psi(t) + \nu_2(t). \end{aligned} \tag{4.18}$$

The linearized observation matrix $H(t)$ is

$$\begin{aligned} H(t) &= \nabla_{\hat{\mathbf{x}}_\mathbf{f}} \mathbf{h}(\mathbf{x}_f, \mathbf{x}_s) \\ &= \begin{bmatrix} -\frac{\sin\theta_1(t)}{r_1(t)} & \frac{\cos\theta_1(t)}{r_1(t)} & 0 & 0 \\ 0 & 0 & -\frac{\sin\theta_2(t)}{r_2(t)} & \frac{\cos\theta_2(t)}{r_2(t)} \end{bmatrix}. \end{aligned} \tag{4.19}$$

The R matrix is the covariance matrix of the two noise processes $\nu_1(t)$ and $\nu_2(t)$ as

$$R = \begin{bmatrix} \mathrm{var}(\nu_1) & \mathrm{cov}(\nu_1, \nu_2) \\ \mathrm{cov}(\nu_1, \nu_2) & \mathrm{var}(\nu_2) \end{bmatrix}. \tag{4.20}$$

Combining (4.19) and (4.20) gives the observed information according to (4.7) as

$$I(t) = H^T(t) R^{-1} H(t) \tag{4.21}$$

If the two noise processes $\nu_1(t)$ and $\nu_2(t)$ are independent, there are no covariances and the observed information matrix $I(t)$ would become a block matrix as

$$I(t) = \begin{bmatrix} \mathrm{I}_1(t) & 0 \\ 0 & \mathrm{I}_2(t) \end{bmatrix}, \tag{4.22}$$

where $\mathrm{I_i}(t), i = 1, 2$ are (2×2) matrices representing the observed information from object i. From this, a model with n objects could be easily be derived by just extending the observed information with block matrices. This is not sufficient for the cross states $\mathrm{Y}_{i_1 j_2}$ in (4.16) to be zero. The term in the update law corresponding to the loss due to the process noise

$$-Y(t) Q Y(t)$$

must also be considered. The process noise is modelled with a block matrix, each block containing process noise for the two objects, extended from (4.9) to

$$Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} = \begin{bmatrix} \mathrm{Q}_{x_1} & 0 & 0 & 0 \\ 0 & \mathrm{Q}_{y_1} & 0 & 0 \\ 0 & 0 & \mathrm{Q}_{x_2} & 0 \\ 0 & 0 & 0 & \mathrm{Q}_{y_2} \end{bmatrix}. \tag{4.23}$$

If the cross states $\mathrm{Y}_{i_1 j_2}$ are set to zero, then the term $Y(t)QY(t)$ would be a block matrix as

$$Y(t)QY(t) = \begin{bmatrix} Y_1(t)Q_1Y_1(t) & 0 \\ 0 & Y_2(t)Q_2Y_2(t) \end{bmatrix}. \tag{4.24}$$

Since both terms, $I(t)$ and $Y(t)QY(t)$, in the update of the information matrix in (4.17) are block matrices with non-diagonal blocks as zeros, there will not be any information update from the cross states $\mathrm{Y}_{i_1 j_2}$. With no update of those states, they will not affect the optimization and therefore those states could be set to zero. Then the information matrix in (4.16) would be a diagonal matrix, with (2×2) block matrices on the diagonal representing the information of each object as

$$Y(t) = \begin{bmatrix} \mathrm{Y}_1(t) & 0 \\ 0 & \mathrm{Y}_2(t) \end{bmatrix}, \tag{4.25}$$

where

$$Y_i(t) = \begin{bmatrix} \mathrm{Y}_{x_i} & \mathrm{Y}_{x_i y_i} \\ \mathrm{Y}_{x_i y_i} & \mathrm{Y}_{y_i} \end{bmatrix}.$$

The matrix in (4.25) could easily be extended to n objects, with adding block matrices for each object.

4.4.2 Simulation with two Objects

In these simulations, the noise processes $\nu_1(t)$ and $\nu_2(t)$ are independent of each other, and the resulting path is examined. An information matrix is constructed as in (4.25), and from this the determinant is taken as utility function, and information could be maximized.

The observed information is inverse proportional to the squared distance and angle dependent, and there is a risk that the optimizer will only consider the nearest object, since it will gain much information by getting closer all the time. In order to avoid such a problem, the object is removed from the model when localized. Thus, the states representing the information from the objects are removed from the system equations and the new information matrix is simply the information matrix from the non-localized object.

In the following simulation two features were placed in $[10 \quad 10]$ and $[10 \quad 0]$, and the vehicle starts in the origin with heading in the Y-axis direction. The optimization time horizon t_f is set to 1 s in the first simulation and to 8 [s] in the second simulation, and the utility function is the determinant of the information matrix. The noise is the same for the two objects. The resulting paths are shown in Figure 4.12, where the short time horizon to the left and the long time horizon to the right.

From the two paths, it is very tempting to make the conclusion that the long time horizon is always better than the short time horizon. But if the threshold for localization is set higher, the long time horizon must plan a long time ahead close to an object. In Figure 4.13 a path is shown, where the problem with a long time horizon is seen. The first two points are the same as the right part of

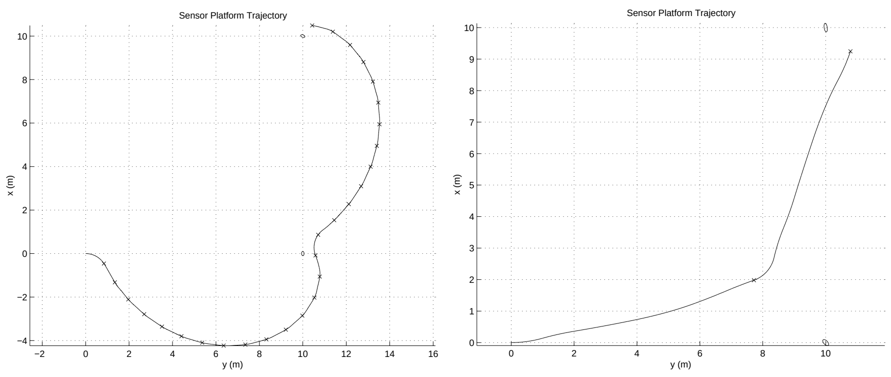


Figure 4.12: Left: Simulated sensor platform trajectory with two objects, time horizon $1\ s$. Right: Simulated sensor platform trajectory with two objects, time horizon $8\ s$.

Figure 4.12, but then the path must be planned a long time ahead close to the target, until it is localized and then turns back to the other object, which is not very effective. The short optimization horizon is not much more effective, but it takes less time to calculate. These issues motivate the use of suitable replanning policies, see discussion in Section 7.5.1.

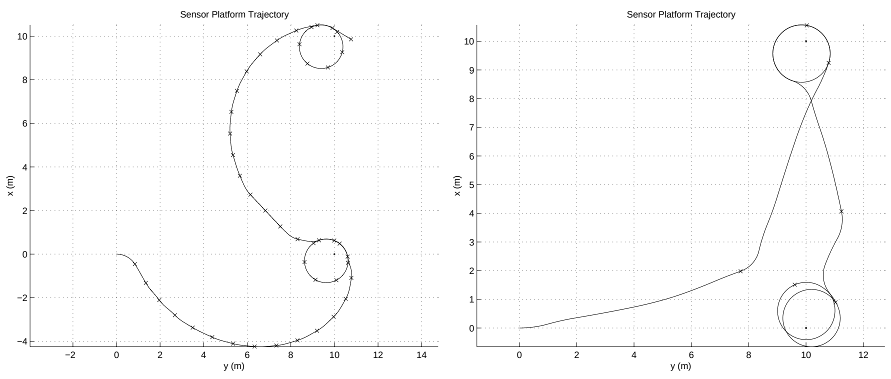


Figure 4.13: With long time horizon (right), the UAV must turn back to the initially nearest object compared to the short time horizon (left).

4.4.3 Extension to n Objects

The model can now be extended to handle n objects, since it is about adding and removing states to the model and hence to the information matrix. When modelling n objects, each object i has its own (2×2) information matrix $\mathrm{Y_i}$. Then a global information matrix could be constructed by putting these on the diagonal as

$$Y = \begin{bmatrix} \mathrm{Y}_1 & 0 & \dots & 0 \\ 0 & \mathrm{Y}_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathrm{Y}_\mathrm{n} \end{bmatrix}. \tag{4.26}$$

A simulation is made with five objects randomly placed in the xy-plane. The same parameters is used as for the simulation with two objects and the result is shown in Figure 4.14.

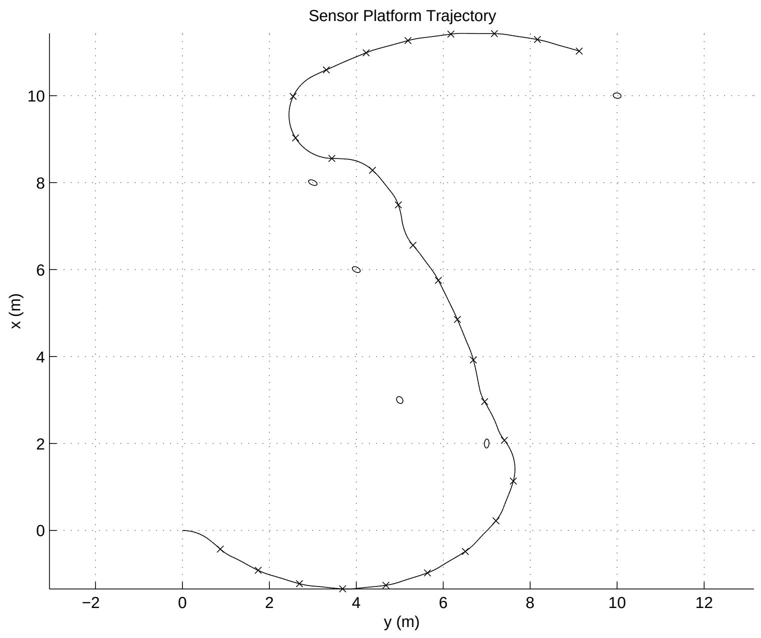


Figure 4.14: Sensor platform trajectory with five objects.

4.5 3d Modelling

The previous simulations have been in two dimensions, but since a UAV has a certain height of flight, it is natural to extend the models to three dimensions. A new control signal of the vehicle that controls the flight altitude could be introduced, but to keep the optimization complexity down we assume that the flight altitude is constant.

There is also a singularity in the 2D model, since the observed information is inverse proportional to the squared distance to the feature. This means that the vehicle could not be close to the object. The problem was solved by removing object when localized or by bounding the control signals, so the vehicle never could come too close. In this section, the singularity is avoided by first deriving the observed information relative a local reference frame and then transform the information into global reference frame.

4.5.1 3d Object

As described previously, the uncertainty of an object's location is represented by its information matrix $Y(t)$. The concept is taken into three dimensions with an extended information matrix:

$$Y(t) = \begin{bmatrix} \mathrm{Y}_x & \mathrm{Y}_{xy} & \mathrm{Y}_{xz} \\ \mathrm{Y}_{xy} & \mathrm{Y}_y & \mathrm{Y}_{yz} \\ \mathrm{Y}_{xz} & \mathrm{Y}_{yz} & \mathrm{Y}_z \end{bmatrix}. \tag{4.27}$$

By modelling an object in three dimensions, the uncertainty in the z-direction will also be considered.

The information matrix is symmetric as before and it is sufficient with six states to represent an object. The update of the information matrix is still:

$$\dot{Y}(t) = -Y(t)QY(t) + I(t), \tag{4.28}$$

but both the process noise Q and the observed information I are now (3×3)-matrices.

4.5.2 Information in 3d

Let the UAV fly in the xy-plane and the distance to the ground, z_s, is kept constant. An object on the ground could be described by the angle α same as θ in the two dimensional case and the angle β, which is the angle from the xy-plane to the object, shown in Figure 4.15.

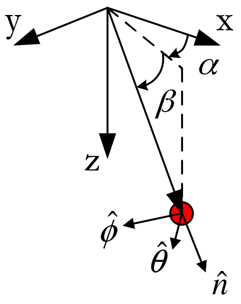


Figure 4.15: 3d modelling of a feature described by the angles α and β. Local coordinates for observing information.

Introduce a sensor direction $\hat{n}$ as the vector from the vehicle to the feature. At this moment, the sensor is considered to"point" at every direction and a camera will be introduced later on in Chapter 8. For now, the sensor has unlimited field of view. A coordinate system is set in the end of the vector at the object, as in Figure 4.15. The information is observed orthogonal to the direction of the camera, that is in the $\hat{\theta}$ and the $\hat{\phi}$-direction. In other words, the variance of the noise process in the $\hat{n}$-direction is infinite. The variance in the $\hat{\theta}$- and the $\hat{\phi}$-directions are σ_θ^2 and σ_ϕ^2 respectively. The inverse of the covariance matrix R could be set as

$$R^{-1} \approx \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{\sigma_\theta^2} & 0 \\ 0 & 0 & \frac{1}{\sigma_\phi^2} \end{bmatrix}, \tag{4.29}$$

if the noise processes are uncorrelated. The information could now be transformed into the global Cartesian coordinates, by using the angles α and β. The angles α and β are the rotation about the z-axis and the y-axis, respectively. The transformation matrices are described as

$$T_2(\theta) = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \text{ and} \tag{4.30}$$

$$T_3(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.31}$$

The resulting rotation matrix which transforms the local coordinates into the global coordinates would be

$$T = T_3(-\alpha)T_2(-\beta) \tag{4.32}$$

The information expressed in the local reference system can now be transformed to the global reference system

$$I_{global} = T I_{local} T^T. \tag{4.33}$$

This yields in a (3×3) information matrix, used in the update law in (4.28), without a singularity. The H in this case is simply the identity matrix in the local coordinates, since the feature location and the observed location is the same point.

A simulation with the same properties as for the two dimensional model in Figure 4.3 is shown in Figure 4.16 for two different heights. In the left picture the height of flight z_s is 1 m. Hence, the β-angle is small and the simulation result is similar to the 2D case. The flight height affects more when it is higher, and in the right picture of Figure 4.16 it was set to 10 m. The rotation matrix about the y-axis differs now from the identity matrix, and there will be essential amount of information in all states in the information matrix (4.27) and this will affect the eigenvalues of the information matrix and hence the utility function. The scalar measure of the utility function will be lower with increasing flight height and could be interpreted as the higher the UAV flies, the less information is seen about the object since it will be farther away. The resulting path with higher flight height is longer, since there will be less information gained.

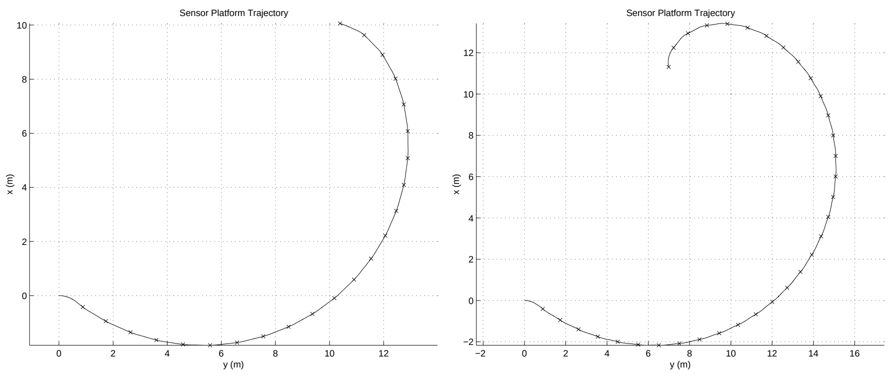


Figure 4.16: The trajectory for an UAV with different flight heights. Left: $z_s = 1\ m$. Right: $z_s = 10\ m$. Feature located in $(10, 10)$.

5 Area Exploration

A common task in surveillance and reconnaissance is area coverage. The area is represented by a number of grid points, and with each grid point is associated a quantity which may be measured under the influence of noise, so an information matrix could be constructed. The flight path is calculated from utility functions derived from the information matrix.

5.1 Area Coverage

The area coverage is usually done after a pre-defined path. The UAV is sweeping the area back and forth until the whole area has been covered. However, we want the UAV to search for objects in the area and once an object is found the UAV should localize it. By calculating the next step from an information matrix consisting of information about both the area and the objects, the UAV is able to react on objects on the ground. The vehicle will now take the best possible step at the time horizon in terms of information.

Consider for example a game of battleships. In those games, you are sweeping your opponents area, and if you hit something, you know that there is a ship and you continue to search that part of the area until the ship has been sunk. Instead of ships and sinking, change the terms to objects and localizing and that will be the case of the UAV.

5.1.1 Area Model

The area is represented by a number of *grid points*, thus the area is discretized. The information from an object was modeled in (4.27) by a (3×3) information matrix, that is by six states. It would be possible to use the object model as the grid point model. Grid points are not going to be localized as the objects, since the grid points are set out and therefore completely known. However, the (3×3) information matrix is suitable since it will capture the properties of how well the grid point has been seen, including if it has been seen from significantly different directions.

The downside of using the six states model for the grid points is, of course, that the complexity of a discretized area with N grid points will grow large to $6N$ states. An alternative is to let each grid point be represented by just one state. This one state has no physical interpretation. We lost the strive for seeing the grid points from significantly different directions, but we keep the complexity down, the total number of states will be N. Hence, calculation time will be much shorter.

In this report we use the one state grid point model. Thus, the state vector $\mathbf{x}$ consists of the three states representing the vehicle, that is $\mathbf{x}_s$ and the N states representing the grid points $\mathbf{x}_{gp}$ as

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_{gp} \end{bmatrix}. \tag{5.1}$$

The state vector consists now of $(3 + N)$ states.

The information matrix consists of the states representing the information and is constructed in the same manner as for n objects as

$$Y_{gp}(t) = \begin{bmatrix} Y_{gp_1}(t) & 0 & \dots & 0 \\ 0 & Y_{gp_2}(t) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & Y_{gp_N}(t) \end{bmatrix}, \tag{5.2}$$

where the index $\mathrm{gp_i}$ denotes grid point i. The noise in each observation is considered uncorrelated and the information matrix becomes a diagonal matrix. The dimension of the information matrix is $(N \times N)$.

5.1.2 Limited Sensor Range

The sensor has a limited range, due to its resolution and the atmospheric conditions. A simple mathematical representation of these limitations is a exponential variation in the measurement uncertainty. Introduce a distance r_{max} which is the maximum range of the sensor and introduce an exponential penalty function such that the covariance of the noise becomes

$$R = \sigma_0^2 \exp\left(k_R (\frac{r}{r_{max}})^2 \right), \tag{5.3}$$

Thus, the expected information is proportional to the inverse of the covariance as

$$R^{-1} = \frac{1}{\sigma_0^2} \exp\left(-k_R (\frac{r}{r_{max}})^2 \right), \tag{5.4}$$

where σ_0 is the standard deviation at zero range, r is the distance between the sensor and the grid point, r_{max} is the maximum range of the sensor and k_R is a constant, $k_R = 4.6$ is used in this chapter. The penalty function is the exponential function of r

$$\exp\left(-k_R (\frac{r}{r_{max}})^2 \right),$$

which does not penalize at zero distance, and at distance r_{max} there will hardly be any information observed, and is plotted in Figure 5.1. Other functions could be used and for examples see Grocholsky [16].

5.1.3 Observed Information

The observation model for an area exploration is according to Grocholsky [16]:

$$\mathbf{z}(t) = \mathbf{T}(x, y) + \mathbf{v}(t), \tag{5.5}$$

where $\mathbf{T}(x, y)$ is a function describing the terrain characteristic and $\mathbf{v}(t)$ zero-mean uncorrelated Gaussian sequence with variance R as a function of the distance r as in (5.3).

In this model the terrain characteristics are said to be the states representing the grid points. For simplicity, we make the assumption that these states could be observed directly by the camera. This means that there is no transformation between the measurements and the states. The reason for making this assumption is that the linear observation matrix H then becomes simply the identity matrix according to (3.14) as

$$\begin{aligned} \mathbf{T}(x, y) &= \mathbf{x}_{gp} \\ \mathbf{z}(t) &= \mathbf{x}_{gp} + \mathbf{v}(t). \end{aligned} \tag{5.6}$$

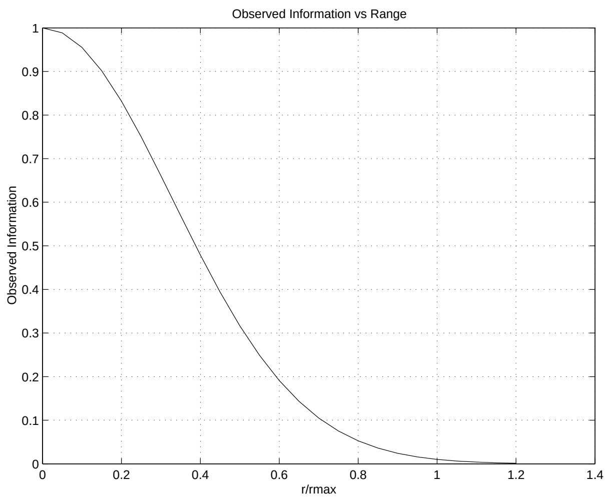


Figure 5.1: Exponential modelling of range dependent measurement errors in a mathematical model of a realistic bearings-only sensor.

The information observed from the grid points follows the usual update law as

$$\begin{aligned} I(t) &= H^T(t)R^{-1}H(t) = R^{-1} \\ &= \frac{1}{\sigma_0^2}\begin{bmatrix} \exp\Big(-k_R(\frac{r_1}{r_{max}})^2\Big) & 0 & \dots & 0 \\ 0 & \ddots & & 0 \\ \vdots & & & 0 \\ 0 & \dots & 0 & \exp\Big(-k_R(\frac{r_N}{r_{max}})^2\Big) \end{bmatrix}. \end{aligned}$$

Each grid point is stationary, but as for stationary objects, a small process noise is included to improve numerical conditioning, as described in Section 4.1.2. The model for a grid point i is

$$\dot{\mathbf{x}}_{gp_i} = \omega(t), \tag{5.7}$$

where $\omega(t)$ is a zero mean Gaussian process with uncorrelated covariance $Q(t)$. The observed information matrix $I(t)$ is also an $(N \times N)$ matrix and the update law of the information matrix is given by

$$\dot{Y}(t) = -Y(t)QY(t) + I(t).$$

5.1.4 Comments on the Utility Function

A problem with using the determinant as utility function is the numerical difficulties. If the area consists of N grid points with low starting information 10^{-m}, the first determinant calculated would be in the range of 10^{-mN}, e.g., if we have 121 grid points with starting information 10^{-3} the determinant is 10^{-363} which is considered to be zero by Matlab. However, maximizing $\log(\det Y)$ would also maximize the entropic information, recall the definition of entropic information from (3.10)

$$i(\mathbf{x}) = \frac{1}{2}\log\Big((2\pi e)^{-n} \det Y\Big).$$

The numerical problems are smaller when using $\log(\det Y)$, but there is still a risk when the number of states is large.

When calculating trace of the inverse information matrix, a summation is done instead of a multiplication and the numerical difficulties could be avoided.

5.1.5 Simulations

In the simulations a quadratic area (10×10) was created, and discretisized with a distance of one between each grid point. This means that there are $11^2 = 121$ points representing the area.

The complexity of the problem is now increased, compared to the simulations in Chapter 4. There are now 121 states for the area plus the original three representing the vehicle, to a total of 124 states, to be compared with for example nine states when localizing an 3D object. There is again need to evaluate the performance of the different utility functions.

Figure 5.2 shows two area searches with different bounds on the control signal. The vehicle starts at the origin with heading upwards and the time horizon is set to 1 s, otherwise the calculation time would be too long. For the same reason the control signal is parameterized into two steps $m = 2$. The background is the information level of the surface, the brighter the more information. The area search is terminated when the information for all grid points is higher than a predefined threshold value.

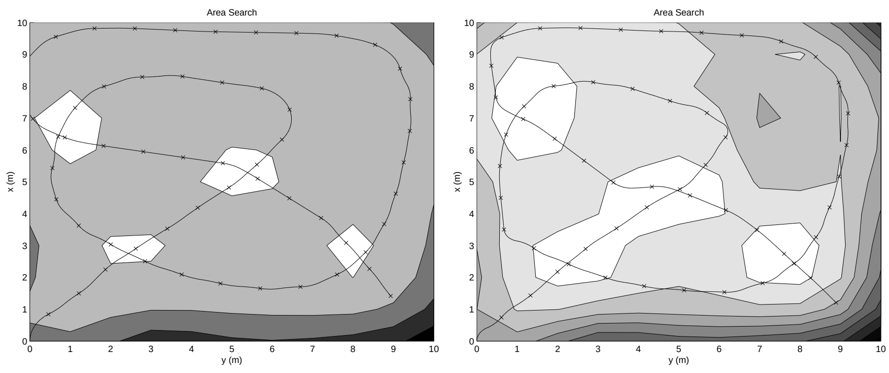


Figure 5.2: Trajectory of sensor platform for an area search with different bounds on the control signals. Left: $-1 \le u \le 1$. Right: $-\pi \le u \le \pi$

The paths are about the same for the two simulations, but with the tighter bound on the control signal, the vehicle will not turn as sharp. The flight path will begin on the diagonal since there is the most information to gain, but when the sensor's maximum range reaches the outer rim of the area, it turns and continues the search in the direction of maximum information.

5.2 Combined Object and Grid Point Model

It is now possible to define a model which treats the problem of searching an area with n objects. The UAV decides where to go, by solving the optimal control problem while flying, with respect to the grid points and the objects.

5.2.1 Global Information Matrix

The locations of the objects are unknown to the sensor initially and first when an object is in the sensor's range, the information matrix from the grid points are extended with the information matrix from the object. So the global information matrix would be

$$Y_{tot} = \begin{bmatrix} \mathrm{Y}_{gp} & 0 \\ 0 & \mathrm{Y}_{obj} \end{bmatrix} \tag{5.8}$$

where Y_{gp} is defined in (5.2) as

$$Y_{gp}(t) = \begin{bmatrix} \mathrm{Y}_{\mathrm{gp}_1}(t) & 0 & \dots & 0 \\ 0 & \mathrm{Y}_{\mathrm{gp}_2}(t) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \mathrm{Y}_{\mathrm{gp}_\mathrm{N}}(t) \end{bmatrix}, \tag{5.9}$$

and

$$Y_{obj} = \begin{bmatrix} \mathrm{Y}_{obj_1} & 0 & \dots & 0 \\ 0 & \mathrm{Y}_{obj_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathrm{Y}_{obj_n} \end{bmatrix}. \tag{5.10}$$

All Y_{gp_i} are scalar values, and the Y_{obj_i} are (3×3) matrices.

As mentioned earlier in (5.4), the observations of grid points are range dependent. This is also true for objects, and the same exponential behaviour is applied to the objects. The information observed from objects is computed in local coordinates, and a measurement provides no information in the direction of the camera. The new range dependent inverse covariance matrix is the same as in (4.29) with the penalty function added as

$$R^{-1}(t) \approx \begin{bmatrix} 0 & 0 & 0 \\ 0 & \frac{1}{\sigma_\theta^2} & 0 \\ 0 & 0 & \frac{1}{\sigma_\phi^2} \end{bmatrix} \exp(-k_R(\frac{r}{r_{max}})^2). \tag{5.11}$$

5.2.2 Simulations

Since the grid points and the objects are modelled differently, there is a problem to get balance in the simulations. The first simulation is an area search where five objects have been placed randomly. The path resembles the paths in Figure 5.2, which could be explained by instead of concentrating on an object, there is more information to be gained by searching the unexplored area. It seems that the model for the grid points gives higher information than the model for the objects, and therefore there is more information to gain by searching the area.

However, one could set different priorities for objects and grid points. The priorities could be chosen such that the information values from the grid points could be compared to the information values from the objects.

As comparison, a simulation with a weighing matrix W is including in the utility function as

$$J = \mathrm{trace}(W\ P), \tag{5.12}$$

where P is the covariance matrix Y^{-1}. The states representing the objects are weighted higher than the states representing the grid points. The resulting path is shown in Figure 5.4 and the vehicle strives to localize a target within the sensor's range. However, the question of how to choose weights remains open.

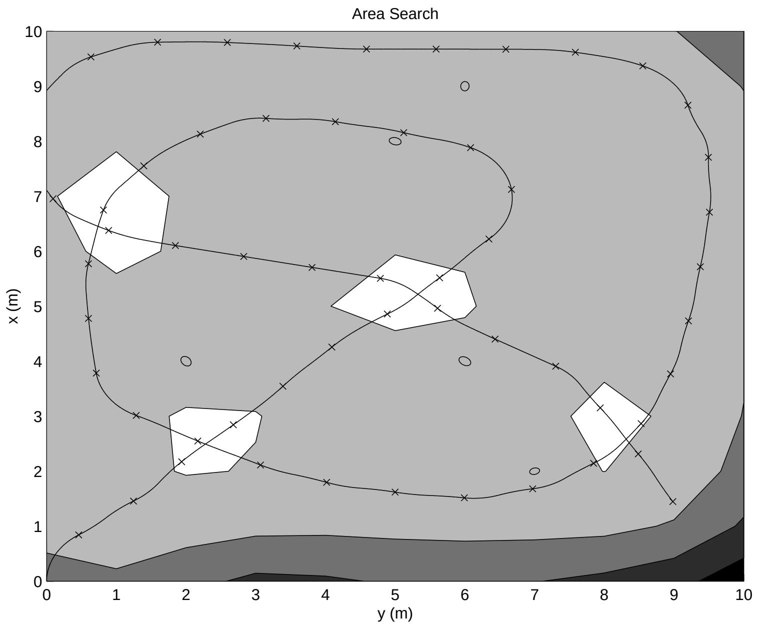


Figure 5.3: Trajectory of sensor platform for an area search with five objects. $-1 \leq u \leq 1$

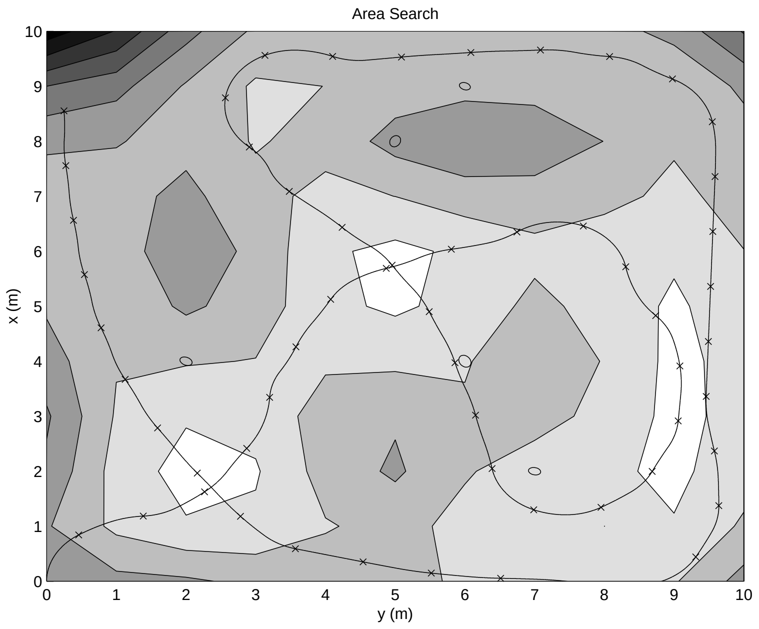


Figure 5.4: Trajectory of sensor platform for an area search consisting of five objects with weighted utility function. $-1 \leq u \leq 1$

6 Long Term Planning Using a WCSPP/TSP Framework

In this chapter we focus on planning the UAV path on a larger scale. To be more precise, we investigate those problems where the possibility of the sensor switching from point A to point B and then back to point A again, can be ignored. This situation obviously occurs when it can be assumed that only one point of interest is within sensor range at each time instant. As will be shown however, temporarily ignoring switching can also facilitate high level planning of strip- and linesearch in the presence of threats and no-fly zones.

The outline of this chapter is as follows, in Section refsec:tsp1 we review some computational tools, including the Traveling Salesman Problem (TSP) and the Weight Constrained Shortest Path Problem (WCSPP). In Section 6.2 we then describe our problem formulation, discuss modelling issues and present some simulation results.

6.1 Computational Tools

In this section we will review some optimization problems and the algorithms used to solve them. Most of these problems are computationally very difficult to solve to optimum and hence we have to settle for approximate solutions.

6.1.1 Traveling Salesman Problem

The Traveling Salesman Problem (TSP) can be described as follows. Given n cities and distances $d_{ij} \in \mathbb{R}_+$ between each pair of them, find the shortest path that visits all cities once. To be more precise we write the following.

Problem 6.1 (TSP) *Let $d \in \mathbb{R}_+^{n\times n}$ and π be a cyclic permutation of n objects, then the TSP can be defined as*

$$\min_{\pi} \; \Sigma_{i=1}^{n-1} d_{(\pi(i),\pi(i+1))} + d_{(\pi(n),\pi(1))}. \tag{6.1}$$

The time required to solve a TSP grows very fast with the size, i.e. number of cities, of the problem. In fact, the TSP is NP-hard, see e.g. [34]. This means that finding the true optimum is very hard, but if we are willing to settle for a good, but not optimal solution, approximation algorithms or heuristics can be used.

In a UAV path planning problem where you want to visit a number of locations, minimizing time, is often a relevant objective. However, in some cases one part of the mission is more urgent and therefore preferably performed early. Similarly, some parts of the mission might be very dangerous, or might be likely to alert the enemy, and thus be reasonable to perform last. The Target Visitation Problem below is one way to incorporate ordering into the objective function of a path planning problem.

6.1.2 Target Visitation Problem

The Target Visitation Problem (TVP), [17], is an extension of the TSP to include ordering. Given a matrix $e \in \mathbb{R}_{+}^{n \times n}$ reflecting the benefit of visiting point i before point j, we define the following problem.

Problem 6.2 (TVP) *Let $e, d \in \mathbb{R}_{+}^{n \times n}$ and π be a cyclic permutation of n objects, then the TVP can be defined as*

$$\max_{\pi} \Sigma_{i=1}^{n-1} \Sigma_{j=i+1}^{n} e_{(\pi(i),\pi(j))} - d_{(\pi(n),\pi(1))} - \Sigma_{i=1}^{n-1} d_{(\pi(i),\pi(i+1))}. \qquad (6.2)$$

As can be seen, the TVP is a TSP with an ordering part added. Figures 6.1 to 6.3 illustrate the formulation. First Figure 6.1 shows 20 randomly placed cities and a random tour visiting all of them. Figure 6.2 then depicts an approximate TSP tour. Finally, we form a TVP by requesting city nr 19, top right, to be visited early by adding 100 to row 19 in e (default $e_{ij} = 1$). Similarly, we want city nr 9, lowest right, to be visited late and thus add 100 to column 9 in e. The result is shown in Figure 6.3.

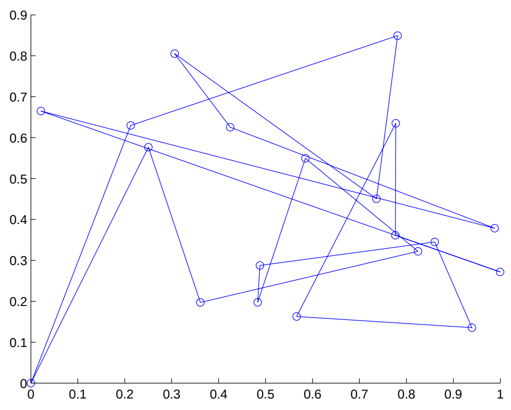


Figure 6.1: A random tour starting at $(0, 0)$ and visiting 20 cities.

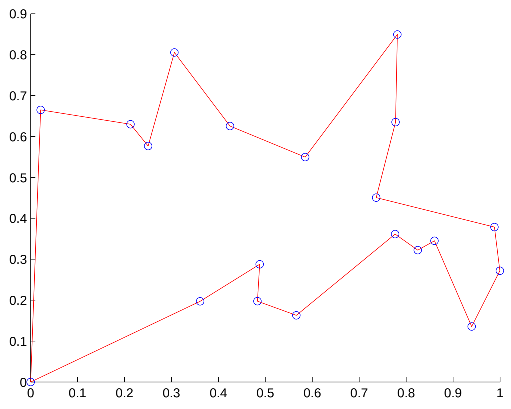


Figure 6.2: The approximative TSP solution given by the algorithm.

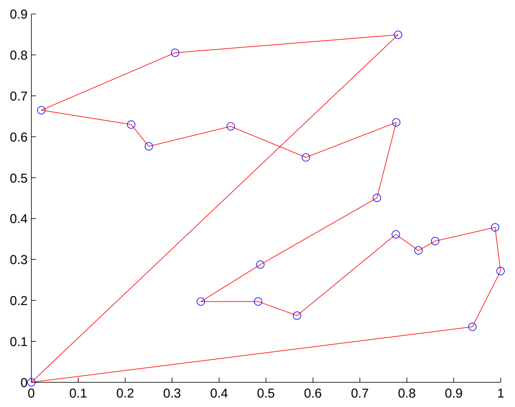


Figure 6.3: An approximate TVP solution including ordering preferences of two cities.

Following [17], and due to of the fact the TSP is NP-hard, we use a heuristic approach to find approximate solutions to the TVP. The proposed algorithm is composed of two steps, that are quite straightforward and easy to code.

The first step uses a randomized greedy procedure to obtain a reasonable starting guess. The second step then improves on this starting guess by doing a local search, in terms of pairwise swapping of two cities. Finally, these two steps are iterated, either a fixed number of times, or a number of times depending on the computation time available.

In detail, the first step works as follows. Given a starting city we must chose the next city to visit. This is done by randomly choosing a city from the closest $\frac{1}{k}$ fraction of the cities that remains to be visited. From this city, again the $\frac{1}{k}$ fraction of the remaining cities is found and the next city is picked by random. This procedure is iterated until all cities are visited. In the extreme case of $k = n$, where n is the number of cities, this corresponds to a standard greedy search. If, on the other hand, $k = 1$, a totally random path is picked each time. In [17], $k = 4$ is suggested and therefore also used here.

To review the second step we first define a 2-exchange neighbourhood.

Definiton 6.1 *A 2-exchange neighbourhood of a tour is the set of tours that differ from the given tour by swapping one pair of cites.*

The second step is performed by doing exhaustive search in a 2-exchange neighbourhood of the starting guess. If a better solution is found, exhaustive search is performed in a 2-exchange neighbourhood around that solution, and so on, until no improvement can be found.

Running the algorithm we get the solutions depicted in Figures 6.2 and 6.3.

6.1.3 Constrained Shortest Path Problems

Another useful tool in path planning is the Shortest Path Problem (SPP) and it's extension, the Weight Constrained Shortest Path Problem (WCSPP). In order to formulate them we first review some graph notation, then state the SPP and finally look at the definition of a WCSPP.

Definiton 6.2 *A graph $G = V, E$ is a collection of vertices $V = \{1 \dots n\}$ and edges $E \subset V \times V = \{(i, j),\ i, j \in V\}$. A path P is a sequence of vertices $P = (v_1, v_2, \dots, v_m)$ such that the set of consecutive pairs, $E(P) = \{(v_i, v_{i+1}),\ i =$*

$1 \dots m-1\}$ *is a subset of* E. *Each edge in a graph can furthermore be assigned weights* $w_{ij} \in \mathbb{R}_+$ *and costs* $c_{ij} \in \mathbb{R}_+$.

We are now ready to state a general SPP.

Problem 6.3 (SPP) *Given a graph* $G = (V, E)$, *costs* $c_{ij} \in \mathbb{R}_+$ *and start and destination vertices* s *and* d. *The Shortest Path Problem (SPP), is defined as follows:*

$$\begin{aligned} \min_P \quad & \Sigma_{(i,j)\in E(P)} c_{ij}, \\ s.t. \quad & s, d \in P. \end{aligned} \tag{6.3}$$

That is, find the path P *from* s *to* d *such that the sum of costs* c_{ij} *is minimized.*

At this time we note that the SPP can be reliably solved in polynomial time by any dynamic programming type of algorithm, such as Dijkstra or A*, [34]. The constrained extension WCSPP is defined as follows.

Problem 6.4 (WCSPP) *Given a graph* $G = (V, E)$, *weights* $w_{ij} \in \mathbb{R}_+$, *costs* $c_{ij} \in \mathbb{R}_+$ *and start and destination vertices* s *and* d. *The Weight Constrained Shortest Path Problem (WCSPP), is defined as follows:*

$$\begin{aligned} \min_P \quad & \Sigma_{(i,j)\in E(P)} c_{ij}, \\ s.t. \quad & \Sigma_{(i,j)\in E(P)} w_{ij} \le W, \\ & s, d \in P. \end{aligned} \tag{6.4}$$

That is, find the path P *from* s *to* d *such that the sum of costs* c_{ij} *is minimized while the sum of weights* w_{ij} *is below the bound* W.

The WCSPP is in general much more complex than the SPP, it is in fact also, as the TSP above, an NP-hard problem [12]. Again, this means that large problem instances are in practice very hard to solve to optimality in reasonable time and we have to use approximation algorithms. Note however, that it is claimed in [12] and [51] that some exact algorithms perform well on problems of reasonable magnitude.

In this paper we settle for an approximate solution and use bisection search and a standard shortest path algorithm, such as Dijkstra, [34] to get a good feasible solution and an upper bound on the gap to optimum.

To apply bisection search, we need the Lemmas below. In the following, let $f, g : \mathbb{R}^n \to \mathbb{R},\ a, b \in \mathbb{R}$. Furthermore, let x_a be a minimizer of $\min(f(x) + ag(x))$ and $g_{opt}(a) = g(x_a)$, $f_{opt}(a) = f(x_a)$. x_b is defined in a similar manner.

Lemma 6.1 *If* $b > a$, *then*

$$g_{opt}(b) \le g_{opt}(a),$$

i.e. g_{opt} *is monotonically decreasing.*

Proof By definition $f(x_a) + ag(x_a) \le f(x_b) + ag(x_b)$ and $f(x_b) + bg(x_b) \le f(x_a) + bg(x_a)$. This makes $f(x_b) + bg(x_b) + (a-b)g(x_a) \le f(x_b) + ag(x_b)$, and using $b > a$ we get $g(x_b) \le g(x_a)$. ■

Lemma 6.2 *By definition,* x_a *is a minimizer of* $\min(f(x) + ag(x))$.
x_a *is also a minimizer of*

$$\min f(x), s.t.\ g(x) \le g(x_a).$$

Proof By contradiction. ■

Lemma 6.3 *Let x^* be a minimizer of*

$$\min f(x), s.t.\ g(x) \leq W. \tag{6.5}$$

If $g_{opt}(a) \leq W \leq g_{opt}(b)$, then $f_{opt}(a) \geq f(x^*) \geq f_{opt}(b)$. That is, x_a is a feasible solution to (6.5) and the gap to the optimal value is bounded as $f_{opt}(a) - f(x^*) \leq f_{opt}(a) - f_{opt}(b)$.

Proof The proof is a straightforward application of Lemma 6.1 and 6.2. ■

Applying bisection search on a and b, making the gap $|b-a|$ smaller while $g_{opt}(a) \leq W \leq g_{opt}(b)$ holds now reduces the gap $f_{opt}(a) - f_{opt}(b)$. Thus we get a good feasible solution and an upper bound on the optimality gap. Note that f_{opt} and g_{opt} are monotonic, but not strictly monotonic. And since WCSPP is a combinatoric problem the gap will decrease monotonically, but not tend to zero.

6.2 Modelling and Problem Formulation

This section describes how the tools of Target Visitation Problem (TVP) and Weight Constrained Shortest Path Problem (WCSPP) can be applied to some UAV surveillance missions. First we explore some modelling issues, and then formulate high and medium level planning problems.

6.2.1 Minimizing Accumulated Risk

This section describes how the sums in (6.4) can be used to model accumulated risk in the WCSPP framework.

A natural constraint or objective function is

$$\max_P \ \Pi_{(i,j)\in E(P)}(1 - R_{ij}), \tag{6.6}$$

where R_{ij} is the risk of losing the aircraft when flying from i to j. The product then reflects the combined probability of surviving all the path segments. Note that this is not a sum of costs as in the WCSPP. In many papers, such as Beard et al.[6] and Zabarankin et al.[51], the sum of kill probabilities, sum of inverse squared threat distances, radar exposure, or some other measure of badness is minimized. Here however, we propose to look for the solution that minimizes the actual risk estimate, by an elaborate choice of edge costs described in the following Lemma.

Lemma 6.4 *Let $c_{ij} = log\left(\frac{1}{1-R_{ij}}\right)$. Then*

$$\max_P \ \Pi_{(i,j)\in E(P)}(1 - R_{ij}), \tag{6.7}$$

has the same solution as

$$\min_P \ \Sigma_{(i,j)\in E(P)} c_{ij}.$$

Proof Note that log is a strictly increasing function and let "$\Leftrightarrow$" denote that two optimization problems have the same solution. Then max $\Pi_{e_i\in Path}(1 - R_{ij}) \Leftrightarrow \max log(\Pi_{e_i\in Path}(1-R_{ij})) \Leftrightarrow \max \Sigma_{e_i\in Path} log(1-R_{ij}) \Leftrightarrow \min \Sigma_{e_i\in Path} - log(1-R_{ij}) \Leftrightarrow \min \Sigma_{e_i\in Path} log(1-R_{ij})^{-1}$, which is exactly the proposed c_{ij}.
■

To find the R_{ij} of the previous paragraph we let the threat level at each point be represented by a function $f : \mathbb{R}^2 \to [0,1]$. This number represents the probability of being shot down when flying one kilometer at that particular threat level. In detail, the probability of surviving a flight path of length l with the constant threat level of $f(x) \equiv p_T$ is

$$p_s = (1 - p_T)^l.$$

With $R_{ij} = 1 - p_s$, this makes the cost of a path segment equal to

$$c_{ij} = \log(1/(1 - R_{ij})) = \log(1/p_s) = -\log((1 - p_T)^l) = -l \log(1 - p_T). \quad (6.8)$$

We now explore different picture quality measures.

6.2.2 Reconnaissance Quality Modelling

In this section we propose a very coarse approximation to the information maximization objective function used in the short term path and sensor planning, described in Chapter 4-5.

The quality of the reconnaissance can be measured in many ways. One choice is assigning a cost to each surveillance object. This cost should then increase with the distance and the amount of occlusion between the UAV and the object at the time when the picture is taken. A first choice of cost might be

$$c_k = (o_k + 0.1)d_i, \quad (6.9)$$

where o_k is some measure of occlusion and d_k is the distance from the UAV to object k.

A more elaborate choice is to maximize the probability that all pictures are useful

$$\max \Pi p_k,$$

where p_k is the probability that the picture of object number k is useful. With this choice, all pictures must be at least of reasonable quality. Again, taking the logarithm and multiplying by -1 we get the cost $c_k = \Sigma \log \frac{1}{p_k}$. Of course, p_k is again a function of distance and occlusion.

We now look at the point target reconnaissance mission.

6.2.3 High Level Planning of Point Targets, Road Segments and Area Search

This section describes planning on a high level where the distances involved are much larger than the sensor range. In other words, to see a target we need to approximately fly right over it.

Our example mission is set in the 10 by 10 kilometer area depicted in Figure 6.4. As can be seen in the Figure, there are 7 point targets, 3 areas and 6 roads to be included in our path. By converting the areas and roads into reasonably spaced points the problem can be cast into our TVP-framework. Without any ordering preferences we get the path depicted in Figure 6.5. Notice how the start and end points of the roads are visited consecutively, corresponding to the whole road segment being covered. This was accomplished by setting $d_{ij} = 0$ for those particular edges in the graph. We did not introduce any ordering preferences in the objective function for this example, i.e. we let $e_{ij} = 0$ for all i, j.

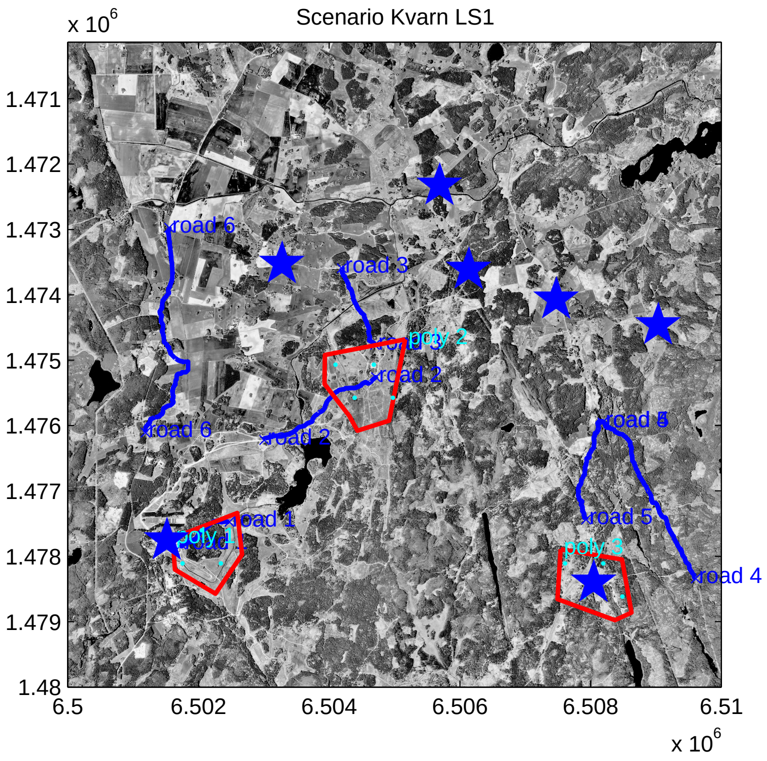


Figure 6.4: The 10 by 10 kilometer area with marked roads, regions and points of interest.

6.2.4 Medium Level Planning of Line and Strip Search with Threat Constraints

In this section we investigate path planning under threat constraints. We take into account coarse sensor range and occlusion phenomenon, but ignore the issue of multi-view images discussed in Chapter 4. In other words, for flight path planning purposes, we assume that one reasonably close and un-occluded view per target is good enough.

Furthermore, we assume that the destination/target ordering can be performed before the choice of detailed flight path. This assumption trivially holds in the case of a line or strip search and in cases similar to the one illustrated in Figures 6.6 and 6.7, where the ordering can be determined by a TVP step as described above.

Given the ordering, the task is now to formulate a WCSPP capturing surveillance quality as well as threat exposure. We create a graph, where each node (x,y,p) represents a position (x,y) of the UAV, as well as a surveillance object (p). To determine the number of nodes we discretize the plane into a uniform grid, and use the given ordering of targets. The edges of the graph are given by neighbouring UAV positions and destinations next to one another in the ordering.

Next we need to determine the cost c_{ij}, end weights w_{ij}, of the WCSPP. We let the costs be determined according to equation (6.8) by the threat levels of the UAV positions. The weights on the other hand should represent picture quality. For an edge between different UAV-positions, but the same surveillance object we set the cost to zero. Only when changing object, i.e. taking a picture,

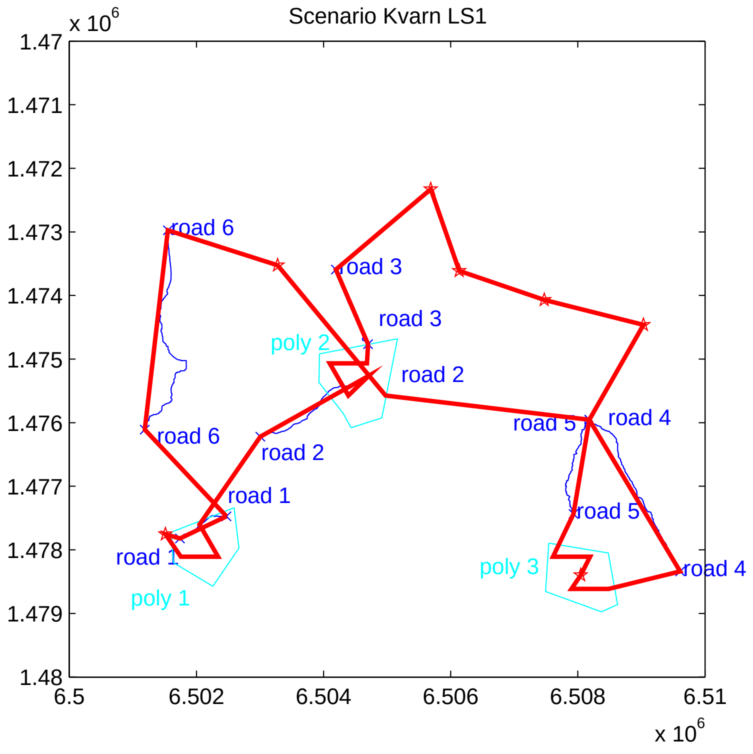


Figure 6.5: Approximate TSP-route through the 30 points of interest.

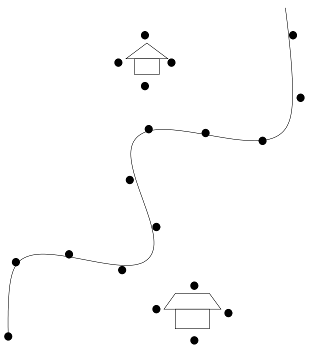


Figure 6.6: Surveillance area with market points of interest.

there is a cost set according to equation (6.9).

Solving the WCSPP now gives us a flight path and sensor usage with as low threat exposure as possible, while keeping a given level of surveillance quality. Figure 6.8 depicts an example scenario including a road to be surveillanced, two occluding forest areas to the east of the road, a continuous threat zone along the road with a very high risk area in the west. It also shows the resulting UAV path using a medium quality bound. Note how the high risk area is avoided and the road near the forest is viewed from the west.

With a low quality bound the risk area is almost completely avoided, as can be seen in Figure 6.9. Note how no surveillance is performed while detouring the most dangerous area. Increasing the quality bound we get a path closer to the

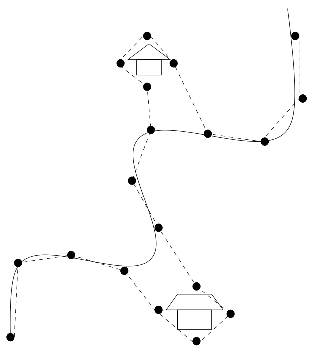


Figure 6.7: An approximate TSP route through the points of interest.

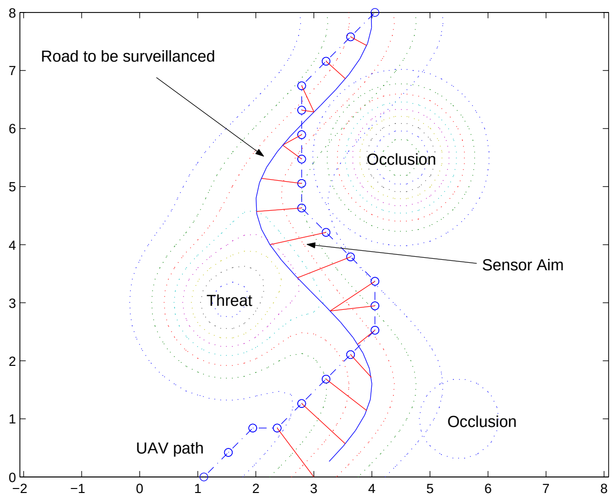


Figure 6.8: Medium quality bound.

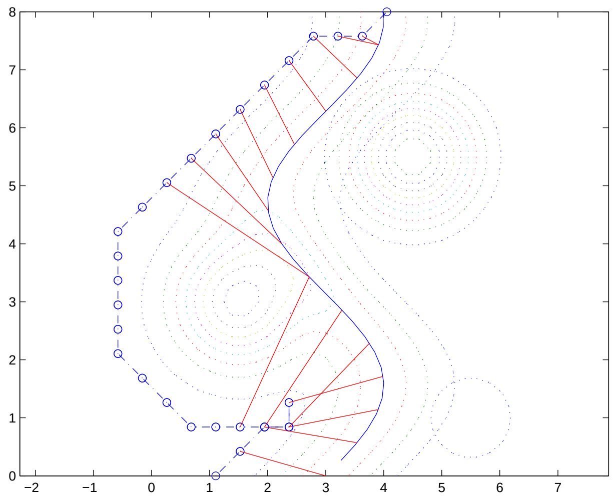


Figure 6.9: Low quality bound.

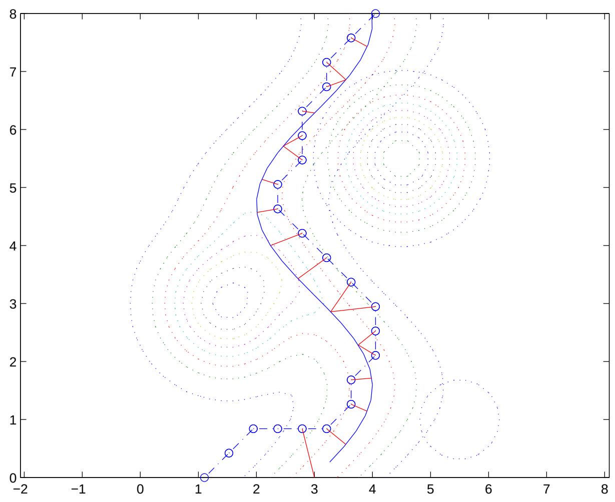


Figure 6.10: High quality bound.

road, as shown in Figure 6.10. Finally, Figure 6.11 illustrates the possibility of choosing start and final position far away from the surveillance objects. Note that the threat zone is avoided at the beginning of the mission, while the occluding forest is overflown at the end. These examples conclude the chapter on long term planning.

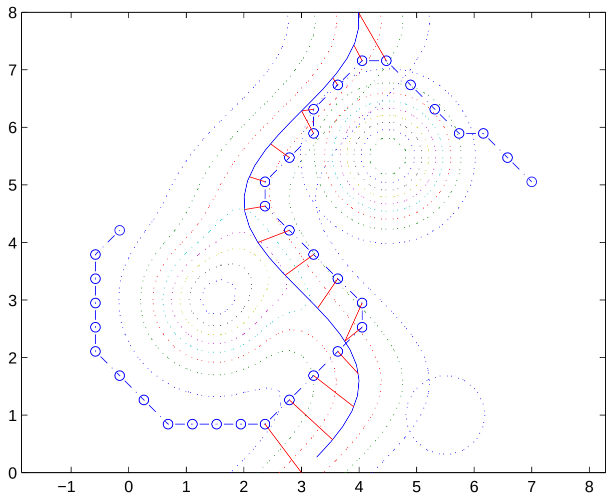


Figure 6.11: Start and goal point far from surveillance object.

7 Optimization, Path Representation and Planning Hierarchy

This chapter will consider some different aspects of the information theoretic planning problem. We will, among others, discuss the optimization problem, different path representations and a planning hierarchy. They are all different sides of the problem, but strongly connected. How the flight trajectory is represented and the planning problem is formulated are strongly affecting the optimization process. A planning hierarchy is introduced that divides the problem into smaller sub-problems.

7.1 Optimization Methods and Tools

Finding the solution to an optimal decision-making problem, i.e. the decision which incurs the lowest cost, is in many applications a problem that has to be solved by some numerical optimization method. A rough classification of the optimal decision-making problem at hand is often useful when deciding which optimization method (and tool) to use. Examples of features of the optimal decision-making problem which affects the choice of method include for instance:

- Is the problem convex? If not, is global optimization necessary?
- Are gradients expensive to evaluate? Are derivative free optimization methods preferable?
- Are there one or several decision makers in the problem?
- Other features such as number of variables and number, and types, of constraints.

As was first explained in Chapter 3, our (short term) path and sensor planning problem can be formulated as a problem in nonlinear programming, with constraints. As was exemplified in Figure 4.5, our optimization problem is generally not convex since the criterion function (utility function) in general has several local optima. However, in many cases a method which solves for a local optimum, i.e. greedy search methods or methods based on evaluating the gradient, results in a solution that is good enough for our engineering purposes. Examples of gradient based optimization tools solving for a local optimum are the *Matlab Optimization Toolbox* [29] functions *fmincon* (for constrained optimization) and *fminunc* (for unconstrained optimization) which have been utilized in the path and sensor planning examples presented in the present report. Another example of a gradient based optimization tool is SNOPT (see [41, 14]), a software package for solving large-scale optimization problems (linear and nonlinear programs). Some preliminary tests have been performed using SNOPT as optimization tool in our applications. Although the performed tests clearly indicate that SNOPT is more effective than the corresponding

MATLAB tools for our applications, more tests have to be performed to draw firm conclusions on the benefits of using SNOPT.

As was mentioned above, gradient based optimization tools like SNOPT or the MATLAB functions can in some cases provide satisfactory results. However, in other cases, for instance when a "bad" initial condition is given to the solver, the gradient based methods can give a solution which is a local optimum but which is not close enough to the global optimum to be acceptable. In such cases, methods (and tools) for finding the global optimum is of interest. An example of a situation when a gradient based optimization tool can run into difficulties is shown in Figure 7.1. In the figure we show examples of criteria functions obtained by using two different choices of occlusion models (see Chapter 9). The dashed curve shows a relatively smooth criteria function obtained by using an angle and range dependent occlusion model (see Section 9.1). The more rugged solid curve shows a criteria function using a digital elevation model (DEM) for occlusion (see Section 9.2). Since the criteria function for the DEM case has a large number of local optima it is clear that a gradient based method can have difficulties finding a solution which is close to the global optimum.

A comparison of some methods and tools for global (and derivative free) optimization is given in [9]. Generally the methods for global optimization are (much) slower than gradient based methods. Some preliminary tests have been performed using a Simulated Annealing [23] tool called Adaptive Simulated Annealing (ASA) [24] but further testing, using other options and other scenarios, is necessary to assess the tool.

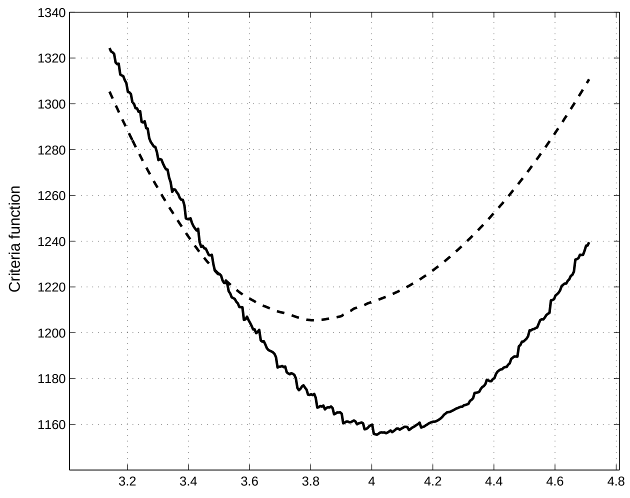


Figure 7.1: Example of criteria functions using two different occlusion models. The dashed line criteria function has been obtained using an angle and range dependent occlusion model. The solid line criteria function has been obtained using a digital elevation model for occlusion. Gradient based method can have difficulties finding an acceptable solution for the solid line criteria function.

7.2 Recursive Filter Algorithm

In Chapter 4 ODE solvers were used to solve the path and the expected information equations, (4.1) and (4.8) respectively. If the trajectory is given, a recursive filter can be used to compute the evolution of the information as

$$Y(k+1) = (Y^{-1}(k) + Q(k))^{-1} + I(k) = Y(k)(1 + Y(k)Q(k))^{-1} + I(k) \quad (7.1)$$

In the remaining chapters of this report, this recursive filter is used to compute the expected information.

7.3 Parallel Computation

In Chapters 8 and 9 more complex methods of the expected information computation are introduced to handle more realistic problems. The computational load is then also increased. If we assume that all features are uncorrelated, the expected information of each feature can be calculated independently and the design of computational nodes is quite straightforward.

Consider the information matrix Y divided into N submatrices Y_i

$$Y = \begin{bmatrix} Y_1 & 0 & \dots & 0 \\ 0 & Y_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & Y_N \end{bmatrix} \quad (7.2)$$

Let $\Lambda(k),\ k = 1, 2, ..., n$ be the trajectory that will be evaluated. Assume that we have N computational nodes. Given the submatrix $Y_i(0)$ and Λ, the node i computes the expected information $I_i^{\Lambda}(k),\ k = 1, 2, ..., n$ and then the information matrix $Y_i(n)$ by using (7.1). For many utility function choices, the utility computation can also be separated. For instance, if the utility function is

$$J = \log \det Y = \sum_i J_i \quad (7.3)$$

the term

$$J_i = \log \det Y_i \quad (7.4)$$

is computed by node i.

The proposed parallel computation structure has been implemented with very satisfying result. Due to communication delays and planning initializing, not much time is saved in small scale simulations. However, in large scale simulations, the decrease of the computational time is almost proportional to the number of nodes. The main challenge in the future work will probably be the update and distribution of large models of the environment.

7.4 Path Representations

The signal, or path, representation is very important to the result of the optimization process. Unfortunately, there are no easy answers to how this representation should be defined. In this section we discuss this problem and present some alternatives.

7.4.1 Discussion

In general, the fewer optimization parameters the better. It is not very efficient to let the sample time of the observations decide the resolution of the optimization parameter. Instead we need a compact parameterized signal representation that allows us to compute the signal with arbitrar resolution given a number of optimization parameters. However, there are several alternatives how these parameters will define a signal. In previous chapters we used a piecewise constant signal, but there are other alternatives of which B-Spline and interpolation methods are presented below.

The next question is in which basis the signal is defined, i.e. which quantity the signal is representing. In previous chapters the signal represents the rate of change of the heading. The flight trajectory is then computed by using the dynamic model of the vehicle. An alternative is to let the optimization parameters represent the flight trajectory directly.

The former representation is better from a vehicle control point of view. The signal is a vehicle control signal and dynamical constraints are naturally expressed in this basis. In general, these constraints are constant bounds if simple dynamic models are used. A disadvantage is that a feedback loop, i.e. an auto-pilot, is needed if we want the vehicle to follow a given trajectory or visit a particular point. Furthermore, obstacle and threat avoidance constraints are more difficult to include.

The latter representation is better from a sensor observation and planning point of view. The flight path is given (almost) directly, thus, there is no need for an auto-pilot and threat and obstacle constraints are relatively easy to include. Furthermore, assume that the signal is created with a method based on basis functions with limited range, e.g. B-Splines (see below). This means that an optimization parameter is only affecting a finite stretch of the flight path. Changing an optimization parameter in the rate of change of heading representation, will affect the entire flight path after that very moment. However, a disadvantage with optimization parameters representing the flight path directly is that dynamical constraints of the vehicle are non-linear.

7.4.2 B-Splines

B-splines provides a flexible and computationally fast design to a large variety of shapes, see Figure 7.2. A B-spline curve [48] is defined as

$$\mathbf{C}(t) = \sum_{i=0}^{n} \mathbf{P}_i N_{i,p}(t) \tag{7.5}$$

where $P_0, ..., P_n$ are control points and $N_{i,p}(t)$ are basis functions defined as

$$N_{i,0}(t) = \begin{cases} 1 & if \quad t_i \leq t < t_{i+1} \\ 0 & otherwise \end{cases} \tag{7.6}$$

$$N_{i,p}(t) = \frac{t - t_i}{t_{i+p} - t_i} N_{i,p-1}(t) + \frac{t_{i+p+1} - t}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(t) \tag{7.7}$$

The knot vector T is a nondecreasing sequence

$$T = \{t_0, t_1, ..., t_m\} \tag{7.8}$$

with $t_i \in [0, 1]$. The knots $t_{p+1}, ..., t_{m-p-1}$ are called internal knots. The degree is defined as $p = m - n - 1$.

In a nonperiodic B-spline the first $p + 1$ knots are equal to 0 and last $p + 1$ are equal to 1. For a uniform B-spline the internal knots are equally spaced. A B-spline with no internal knots is a Bézier curve. A curve is $p - k$ times differentiable at a point where k duplicate knot values occur.

Nonuniform Rational B-Splines (NURBS) are a generalization of non-rational B-splines with weighted control points. NURBS offer a mathematical form for both standard analytical shapes and free form shapes.

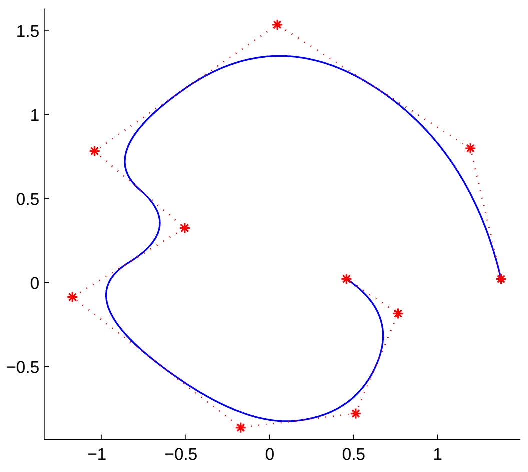


Figure 7.2: Example of a uniform B-Spline of degree 2 and its control points.

Curvature

In two dimensions, let a plane curve be given by Cartesian parametric equations $x = x(t)$ and $y = y(t)$. Then the (extrinsic) curvature κ is defined [50] by

$$\kappa = \frac{d\phi}{ds} = \frac{\frac{d\phi}{dt}}{\frac{ds}{dt}} \tag{7.9}$$

where ϕ is the tangential angle and s is the arc length. It can be shown [50] that the curvature is

$$\kappa = \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{3/2}} \tag{7.10}$$

7.4.3 Interpolation

An alternative approach is to use an interpolation method to compute the path. Examples of methods are linear and cubic spline interpolation.

A cubic spline [49] is a spline constructed of piecewise third-order polynomials which pass through a set of m control points. The second derivative of each polynomial is commonly set to zero at the endpoints, since this provides a boundary condition that completes the system of $m - 2$ equations.

Simulation Results with Cubic Spline Optimization

In this section we use cubic spline interpolation and let the optimizer search for a number of control points (path points). Note that the resulting path is considered as the flight path and not the rate of change of heading as in Chapter 4. The result is compared with the method in Chapter 4.

The path points must be placed in such a way that the path is feasible and the control signals corresponding to the path are bounded. This is done by introducing geometrical bounds on the path points as shown in Figure 7.3. The path points are the bigger points, and the smaller points are the interpolated points in between. The first constraint is that the path points must be at a certain distance d from each other. The second constraint is that the angle φ is bounded. φ_i is the angle between two lines, one line going through path points $i - 2$ and $i - 1$ and one line going through path points $i - 1$ and i.

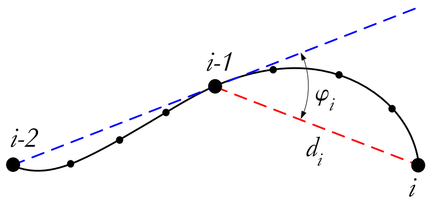


Figure 7.3: Geometrical bounds on the path. The bigger points are the path points and the smaller points interpolated (e.g. spline) points in between. The distance d between the path points is bounded, as well as the angle φ in the intersection of two splines.

Consider the example in Chapter 4, but now optimizing path points instead of the control signal. The information is in three dimensions, with flight height 1 m and it would be the same situation as the left plot of Figure 4.16. The feature and sensor are modelled as in Chapter 4. The states representing the vehicle are removed from the model and the spline is used. The position could be taken directly from the spline and the angle ψ could be calculated by using the path points.

In the first simulations, the parameters describing the spline are

$d = 1\,m$	Distance between the path points.
$\varphi = 30°$	Angle bound between the path points.
$n_s = 3$	Number of path points in each optimization step.

The distance between the path points is kept constant to resemble constant velocity. The other simulations parameters are

$[x_f \quad y_f] = [10 \quad 10]\,m$	Feature location
$Y(0) = \mathrm{I}_{3\times3} \cdot 10^{-3}$	Starting information
$Q = \mathrm{I}_{3\times3} \cdot 10^{-6}$	Process noise
$R = \sigma^2, \quad \sigma = 2.5°$	Observation noise

The resulting trajectory, Figure 7.4 (left), is shaped as a spiral like the simulation in Figure 4.3 in Chapter 4. Thus, the cubic spline optimization gives about the same result as for the control signal optimization, which is expected since the same information model is used. Figure 7.4 (right) shows two simulation results where the distance is set $d = 2$ m and $d = 3$ m, respectively.

The simulations are so far very similar to the simulations in Chapter 4. Arguing for the use of splines, it is very easy to place the path points with different distance in between. The next simulation is the same as for in Figure 7.4 (left) but the distance is allowed to vary within 20%, i.e. the path point distance is between 0.8 m and 1.2 m. This would be harder to implement in the original model. The resulting path is shown in the left part of Figure 7.5.

In Figure 7.5 (right) the length between the path points varies with the distance to the target. This opens up for two interpretations. The first is that the vehicle will no longer travel with constant velocity, but instead travel fast when it is far away and slow when it is close to the object. The other interpretation is that the velocity is still constant and instead the number of observations varies with the distance to the target.

To summarize, the cubic spline interpolation method is appropriate, the resulting trajectory is similar to the result of the optimal control problem. The interpolation methods are flexible, the properties of the path can easily be

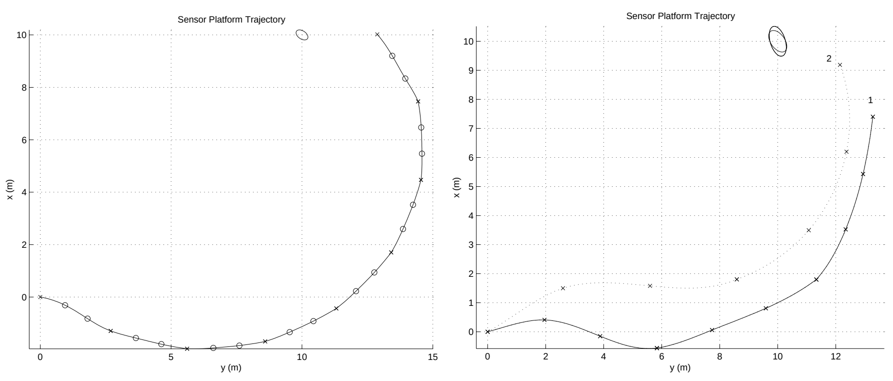


Figure 7.4: *Left:* The optimal trajectory with constant distance 1 m between the path points. Each path point is marked with an 'x' or a 'o', the 'x' points indicates a new optimization. *Right:* The resulting trajectory for distance between path points: 1) $2\ m$ and 2) $3\ m$.

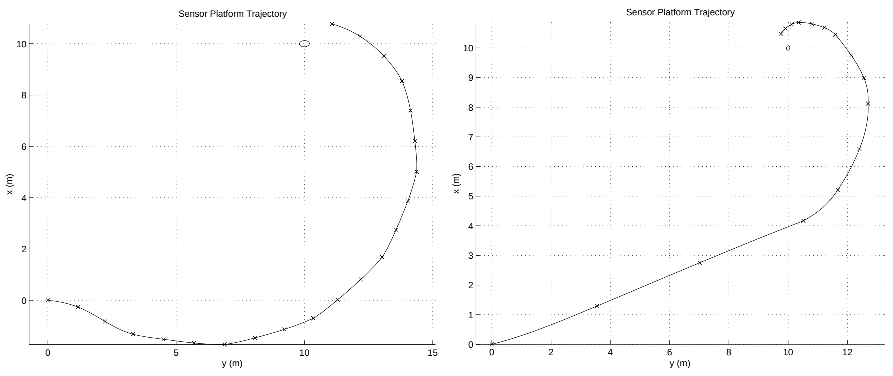


Figure 7.5: *Left:* The resulting trajectory for a path with distance varying 0.8 and $1.2\ m$ between two spline points. All path points are marked with an 'x'. *Right:* The optimal trajectory with varying distance in each subspline.

changed and it is easy to introduce forbidden flight zones or certain points that the UAV must pass.

There is still, of course, the problem with many optima, and the number of local optima increases with more path points or less tight bounds. The starting values are also important for a successful result. However, the optimization with cubic spline is faster than the corresponding optimal control problem.

7.5 General Path Planning Hierarchy

A problem is that all algorithms solving realistic problems become computationally very demanding due to the "curse of dimensionality". A monolithic planner is probably an unrealistic goal, thus a hierarchical decomposition is required. The issue then is how to decompose the problem into sub-problems that guarantee that the overall objective is achieved, see the discussion in Section 2.7 that proposes a decomposition of planning into a functional and temporal hierarchy.

Even the most simple bearings-only localization problem in Chapter 4 is non-linear and non-convex. Adding more features, more degrees of freedom of the vehicle, an actuated sensor gimbal, and a sensor with limited field of view (see Chapter 8) will increase the complexity and the number of local optima enormously. However, the utility function will be smooth if the field of view model is "nice", but if a realistic occlusion model is included (Sections 9.2 and 9.3) the utility function will not be smooth, causing huges problems for gradient based optimization routines, see Section 7.1.

Our working hypothesis is that our very complex problem can be solved by successive refinement of the plan. First the planning horizon is long and the system and environmental models are coarse. At the next planning level, the planning horizon is shorter and the models more detailed and so on. Furthermore, at each planning level a suitable optimization routine are applied.

7.5.1 Planning Levels

In this section we propose a planning hierarchy that divides the problem into a number of planning levels, see Figure 7.6. The objective is to find a "sufficiently good" plan by successive refinement.

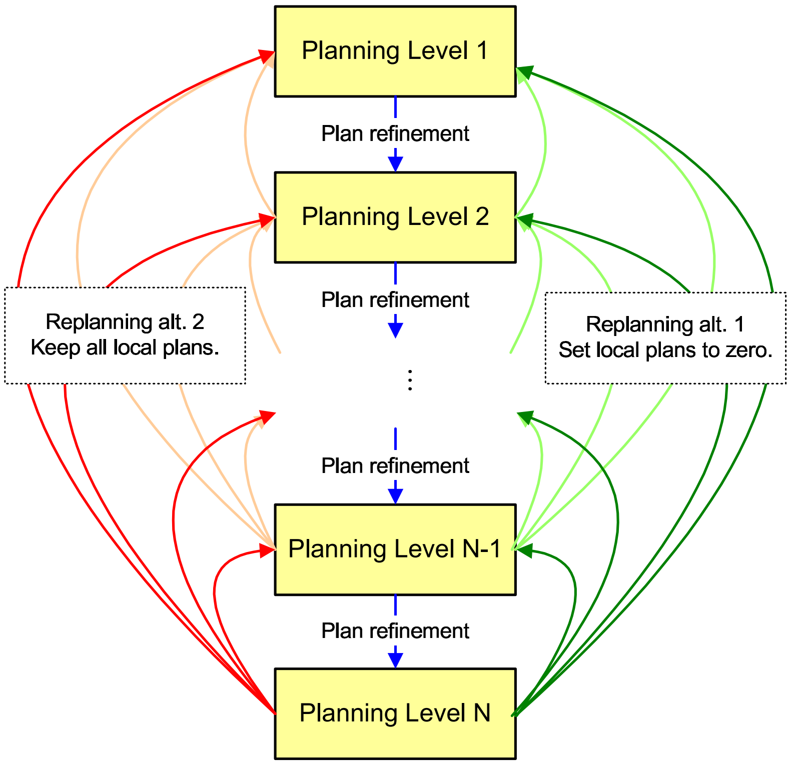


Figure 7.6: Proposed planning hierarchy.

Each level in the planning hierarchy has one *local* plan. The *total* plan is the sum, in some sense, of all *local* plans at all levels 1 to N. The utility function is always based on the total plan.

Let the local plan at level 1 be a coarse long term plan. Then, the local plans 2 to N can be considered as deviations from plan 1. In general, the lower planning levels have longer planning horizon and should capture the low frequency behaviour and the higher planning levels should capture the high frequency behaviour at the cost of shorter planning horizon. Thus, the models of the system and the environment are more detailed at the higher planning levels.

Initially, all local plans are zero. A coarse plan is first created at planning level 1 and since all plans except plan 1 are zero, the total plan is then equal to plan 1. In the next step, local plan 2 is then considered. The total plan is now the sum of the local plans 1 and 2. This is repeated for all planning levels until level N. The final total plan is then executed.

In real world applications our knowledge could never be complete. The plan at time t_0 is based on the knowledge about features and environment at time t_0. However, during the execution of the plan, new knowledge is obtained through observations of the world. Thus, we need to *replan*. For instance, at time t_0 a new plan for the time span $[t_0, t_0 + t_T]$ is generated. During the execution of this plan, a new object is discovered at time t_1. Then a new plan is generated for $[t_1, t_1 + t_T]$, see Figure 7.7.

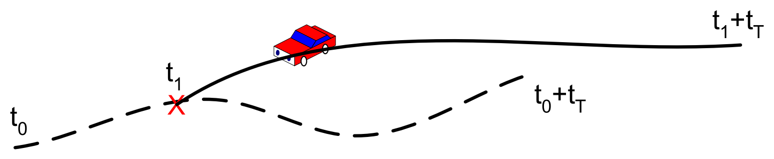


Figure 7.7: Replanning at time t_1.

In our planning hierarchy, we have two replanning ways. For both alternatives, we can go "back" to an arbitrarily planning level k and change the local plan k. In the first alternative, we keep the all local plans, illustrated by the green arrows to the right in Figure 7.6. The second alternative is a complete replanning where all local plans $k + 1, ..., N$ are set to zero, illustrated by the red arrows to the left in Figure 7.6. However, the latter alternative requires some constraints to avoid discontinuities in the total plan.

The proposed hierarchy results in a very flexible planning process. A plan contains of a number of parameterized signals. Each signal represent a quantity we need to optimize, e.g. the flight path, the sensor aiming direction, sensor zoom, etc. At each planning level we decide which signal(s) to optimize and the degree of freedom of the signal(s), e.g. when optimizing the flight path it is possible to restrict the path to a plane. In general, it is advantageous to plan the movements of the vehicle at the "earlier" planning levels and sensor properties at the "later" planning levels. Compare with the discussion in Section 2.7.

7.5.2 NURBS Implementation

Consider a path planning problem. Each planning level has k NURBS curves. For simplicity, we assume $k = 1$ in this section, but the method can be extended to handle more than one curve. The planning levels $i, \ i = 2, 3, ..., N$ have the following options:

n_i	number of optimization points (parameters)
d_i	control points divider, relative plan $i - 1$
j_i	planning dimensions
r_i	observation sample resolution

Furthermore, each planning level has one occlusion model and one field-of-view model.

All NURBS curves are uniform, i.e. the knots are equally spaced. Curve i has the same knots as curve $i - 1$ plus $d_i - 1$ new knots between every knot in curve $i - 1$. The curve of plan i has

$$m_i = (m_{i-1} - 2)d_i + 2, \ i = 2, 3, ..., N \tag{7.11}$$

number of control points (assuming that the degree of the NURBS is 2). Plan 1 is a long term plan and all other plans are initially set to zero.

A total plan is obtained by summarizing all plans, $i = 1, 2, ..., N$. Firstly, new knots are inserted in all curves $i = 1, 2, ..., N - 1$, so all curves have the same knot vector as curve N. There exist algorithms for inserting new control points for specific knots values without changing the shape of the curve. Then the control points are summarized for each knot value and a new curve can be created based on these control points, see Figure 7.8

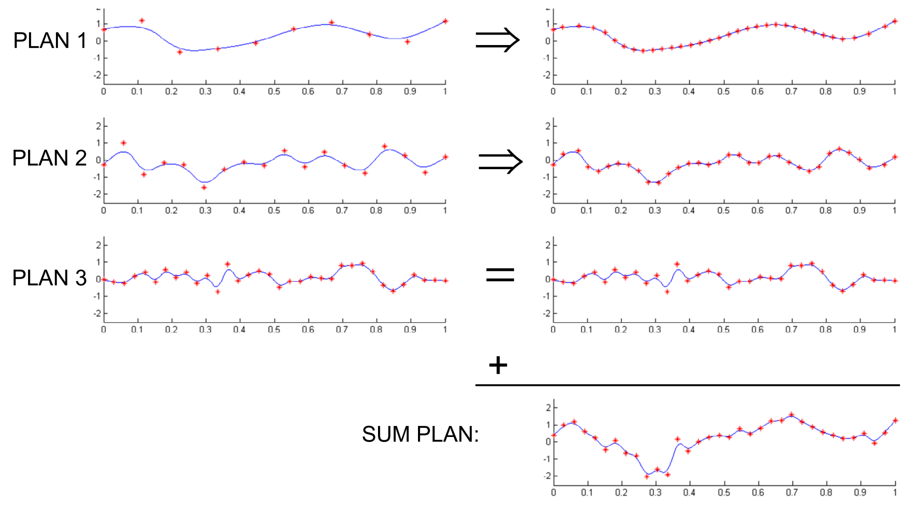


Figure 7.8: New control points are inserted in plans $i = 1, 2, ..., N-1$, so all plans have the same number of points. Then the control points of all plans are added to create the total plan.

The optimization process is as follows. An optimization routine is applied to each planning level $i = 2, 3, ..., N$ sequentially (we assume that plan 1 is constant). The optimization evaluation is always performed on the total curve, not on the curve i directly. The optimization parameters are n_i points in the curve i. The indexes in j_i decides the degree of freedom of the curve. The number of observations is defined by r_i. Initially, all curves $i = 2, 3, ..., N$ are zero, but when optimizing plan i all curves $i = 1, 2, ..., i - 1$ are non-zero, in general.

Now assume that we need to replan at the control point k in the summation plan. Then all control points $1, 2, ..., k, k + 1, k + 2$ are locked, i.e. their values will not be changed any more. Control points $k, k + 1, k + 2$ are also locked to assure a continuous summation plan. If we decide to replan at planning level i, the optimization parameters will be the n_i control points in curve i next to the control point $k + 2$ in the total curve. The start values are either the current curve (keep all curves) or the zero curve (all curves $i, i + 1, ..., N$ are zero), see Figure 7.6.

In our case, the flight path is described directly by the total curve. Planning level 1 is the long term planning based on a WCSPP/TSP framework, see Chapter 6. The level 1 plan is created based on prior knowledge of the surveillance area and the mission. Planning levels 2 to N are based on the information theoretic approach. To assure that the plan is feasible, the curvature of the plan is calculated, see Section 7.4.2. The curvature is used as a penalty in the optimization process.

8 Limited Field-of-View and Sensor Planning

So far in this report we have assumed that the sensor is able to see in all directions all the time. Furthermore, we have assumed that all objects and grid points are visible from all positions. The only limitation has been a distance dependent noise, i.e. observations contain more information the closer the feature is. In this chapter we introduce a more realistic sensor model with limited field-of-view. We also assume that the sensor pointing direction is controllable. In the next chapter we also introduce occlusion in order to be able to handle more realistic scenarios.

8.1 Expected Information Computation

We include the effects of limited field of view and occlusion by increasing the measurement noise R, i.e. decreasing the expected information for each feature. The expected information I_i for feature i is computed as before as

$$I_i = H^T R_i^{-1} H \tag{8.1}$$

However, we divide the product R_i^{-1} in two parts. The first part $\tilde{R}_i^{-1}$ is only dependent on the sensor properties. The second part f_i is a function of the sensor position $\mathbf{x}_s$ and aiming direction $\mathbf{x}_c$, and this part represents the information degeneration due to range, limited field of view, occlusion, etc. Thus,

$$f_i(\mathbf{x}_s, \mathbf{x}_c) = f_i^{range}(\mathbf{x}_s, \mathbf{x}_c) f_i^{fov}(\mathbf{x}_s, \mathbf{x}_c) f_i^{occ}(\mathbf{x}_s, \mathbf{x}_c) \tag{8.2}$$

where $f_i^j(\mathbf{x}_s, \mathbf{x}_c) \in [0, 1]$. An example of $f_i^{range}(\mathbf{x}_s, \mathbf{x}_c)$ has already been introduced in Section 5.1.2 in the exponential penalty function modelling the limited sensor range. This chapter will propose methods for modelling the limited field of view, thus, the computation of $f_i^{fov}(\mathbf{x}_s, \mathbf{x}_c)$. In Chapter 9 the occlusion and the computation of $f_i^{occ}(\mathbf{x}_s, \mathbf{x}_c)$ are considered.

8.2 Experimental Sensor System

The sensor system that serves as pattern is quickly presented here. The sensor system is a gimballed EO/IR system with an IR QWIP sensor and a CCD video sensor, see Figure 8.1. Field-of-view of the QWIP sensor is $20° \times 15°$. The gimbal has two actuated axes, pan and tilt. Pan axis is able to rotate $360°$, and the tilt axis is able to rotate approximately $\{10°, -90°\}$.

8.3 Sensor Pointing Path

There are some alternatives of how the sensor aiming can be represented. The discussion in Section 7.4.1 is also relevant here. If a gimballed sensor is considered, it is natural to represent the aiming direction with pan and tilt angles. The gimbal kinematic and dynamic constraints are then defined rather easily.

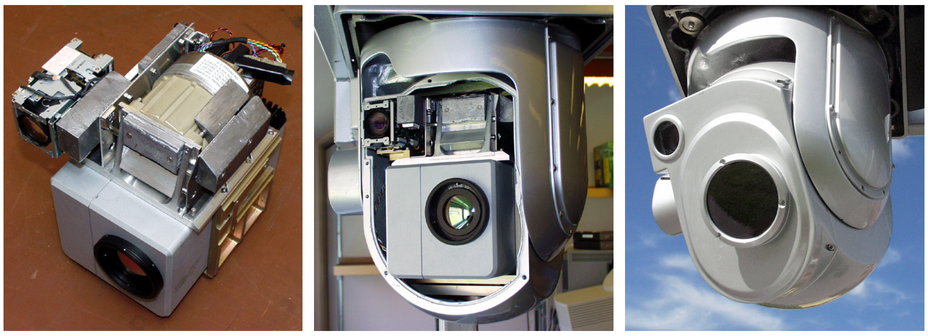


Figure 8.1: The gimbal system. Left: Inner gimbal consists of an IR camera and a colour CCD. Middle: The gimbal with demounted front. Right: Gimbal with mounted front.

However, a sensor pointing path, representing the positions that the sensor is pointing at, has some advantages from the planning point of view and will be used in this report.

8.3.1 Cubic Spline Interpolation Implementation

The sensor pointing path is obtained in the same way as the vehicle path described in Section 7.4.3. The optimizer searches for n_s vehicle points and n_c sensor pointing points at the same time, see Figure 8.2. The vehicle and sensor

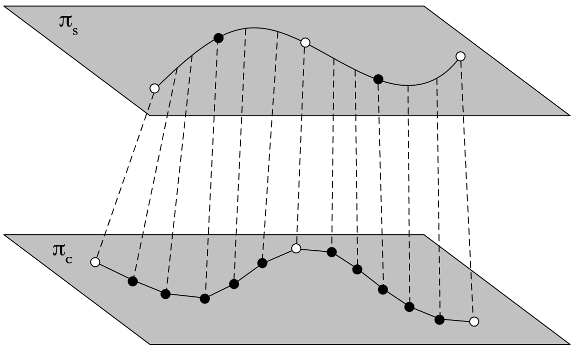


Figure 8.2: The cubic spline interpolation method with the vehicle path in the upper plane and the sensor gazing path in the lower plane.

points are in two parallel planes, π_s and π_c respectively. It is not necessary that $n_s = n_c$. An interpolating method is used to compute the complete vehicle and sensor pointing trajectories. If not stated otherwise, a linear interpolating method is used to compute the sensor pointing trajectory. It is possible to add constraints on where to put the sensor points, as for the vehicle points, but in the simulations in this report no constraints for the sensor points were applied. The results are quite reasonable even without these constraints.

8.3.2 NURBS Implementation

The NURBS implementation presented in Section 7.5.2 is extended with a sensor pointing curve. A curve is added to each planning level and the complete sensor pointing curve is computed as described in Section 7.5.2. The number of control points and knots are the same for both flight and sensor paths at each planning level.

8.4 Limited Field-of-View

The sensor has a limited field-of-view (FOV) and therefore it is only possible to see features (objects and grid points) that are inside the sensor footprint on the ground, Figure 8.3.

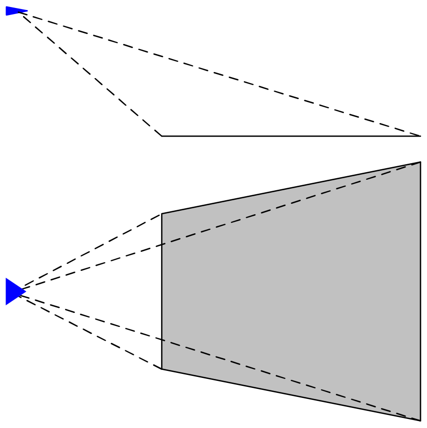


Figure 8.3:

8.4.1 Image Projection Model

Given a pinhole camera model, Figure 8.4, a point $\mathbf{P} = [X \quad Y \quad Z]^T$ is projected on a virtual image plane onto the image point $\mathbf{p} = [x \quad y]^T$ according to

$$\mathbf{p} = -\frac{c}{Z}\begin{bmatrix} X \\ Y \end{bmatrix}. \tag{8.3}$$

If the image size is $n_x \times n_y$ pixels and the field of view is $\alpha_x \times \alpha_y$, the camera constant c is defined as

$$c = \frac{n_y}{2\tan\frac{\alpha_y}{2}}. \tag{8.4}$$

By using this model, it is possible to project the features onto the image plane and, hence, decide if the features are visible or not.

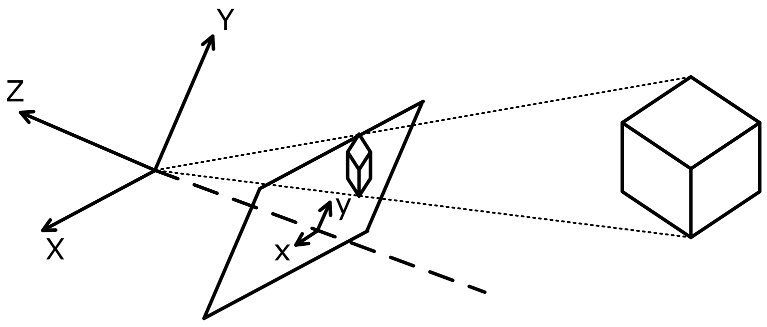


Figure 8.4: The pinhole camera model.

Inversely, the image plane can be projected on a ground plane and the resulting polygon is the sensor footprint, Figure 8.5.

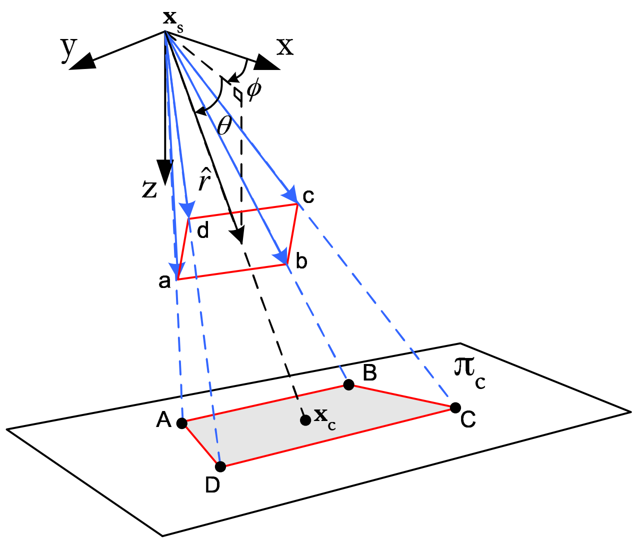


Figure 8.5: Projection of plane $abcd$ on the plane π_c.

8.4.2 Edge Function

The optimization routines based on gradient search will probably fail if we state that all features inside the image plane $abcd$ (or sensor footprint $ABCD$) are fully visible and all features outside image plane are not seen at all. The optimization requires a smooth transition between visibility and invisibility, or in other words, the optimizer requires a non-zero gradient on the information surface to know in which direction the information utility increases.

In practice, a smooth and continuous threshold function is defined for each edge. Let w be the perpendicular distance to the line going through the edges, i.e. $w > 0$ is "outside", $w = 0$ is on the line/edge and $w < 0$ is "inside", see Figure 8.6. The threshold function is near 1 if $w \ll 0$ near 0 if $w \gg 0$. For

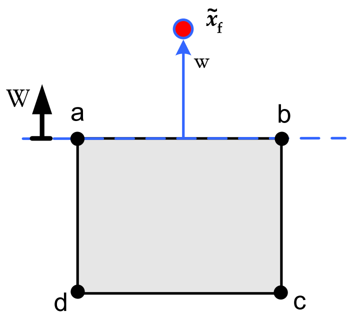


Figure 8.6: Definition of W relative the image plane. The red dot $\tilde{\mathbf{x}}_f$ is a feature in $\mathbf{x}_f$ projected on the (extended) image plane.

instance,

$$f_{edge}(w) = f_{atan}(w) f_{exp}(w) \tag{8.5}$$

where

$$f_{atan}(w) = 1/2 - \frac{1}{\pi}\arctan(k_1 w) \tag{8.6}$$

and

$$f_{exp}(w) = \begin{cases} 1, & w < 0 \\ \exp(-k_2 w^2), & w \geq 0 \end{cases} \tag{8.7}$$

The threshold behaviour comes from the arctan function, the exponential function is added to force the values "faster" to zero when w increases. The parameters k_1 and k_2 affect the slope and may also be dependent on w.

Finally, by taking the product of all four edge threshold functions a new smooth and continuous threshold function f_{fov} is obtained with values near 1 inside $abcd$ and near 0 outside, i.e.

$$f_{fov}(\mathbf{x}_s, \mathbf{x}_c) = f_{edge}(w_{ab})f_{edge}(w_{bc})f_{edge}(w_{cd})f_{edge}(w_{da}) \tag{8.8}$$

Figures 8.7 shows some examples of the sensor footprint and figures 8.8 shows the visibility value in the point $[30 \quad 0]^T$ as a function of $\mathbf{x}_c$. In these examples the threshold functions are applied to the sensor footprint $ABCD$.

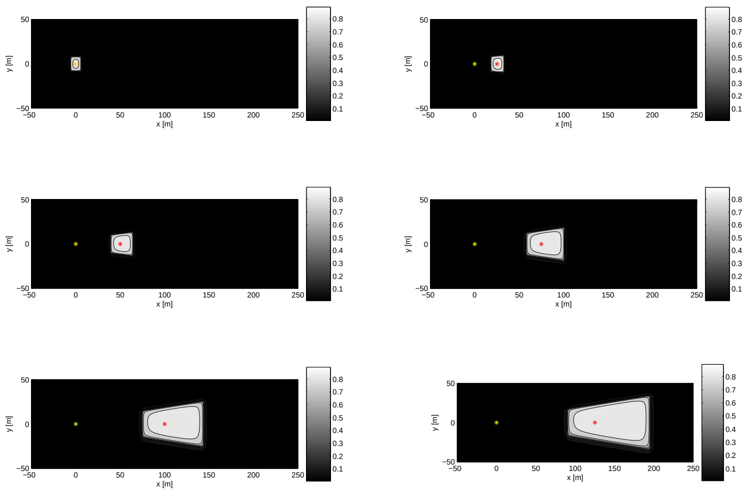


Figure 8.7: Sensor footprint examples. Sensor located in $[0 \quad 0]^T$ at the altitude of 50 m. Sensor aiming points are $\mathbf{x}_c = [0 \quad 0]^T$, $\mathbf{x}_c = [25 \quad 0]^T$, $\mathbf{x}_c = [50 \quad 0]^T$, $\mathbf{x}_c = [75 \quad 0]^T$, $\mathbf{x}_c = [100 \quad 0]^T$, $\mathbf{x}_c = [125 \quad 0]^T$.

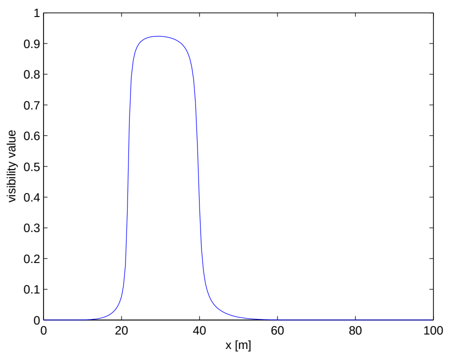


Figure 8.8: Visibility value in the point $[30 \quad 0]^T$ as a function of $\mathbf{x}_c = [x_c \quad y_c]^T$ where $x_c \in \{0, 100\}$ and $y_c = 0$.

9 Occlusion Modelling

In high-altitude surveillance the 3d structure of the ground may be ignored and the ground can be approximated with a plane. However, in low-altitude surveillance, especially in urban environments, the 3d structure give rise to occlusion that can not be ignored, Figure 9.1. In this chapter we introduce

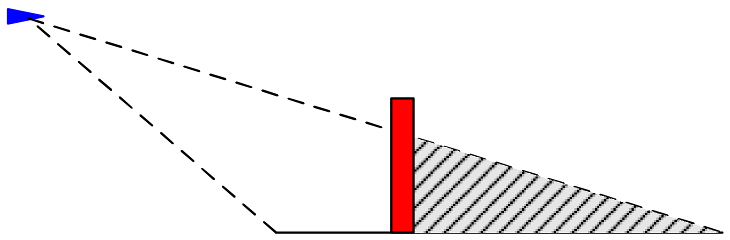


Figure 9.1: Occlusion.

occlusion and some models of occlusion in order to be able to handle more realistic scenarios. The effects of occlusion is included by decreasing the expected information as described in Section 8.1.

In this report some preliminary results are based on a simple 2d angle-dependent occlusion model. The 2d model is smooth and computationally efficient. On the other hand, the model is a very simplified description of the occlusion, but for some scenarios it will be faithful enough. For more complex scenarios, models based on digital elevation models (DEM) and even more detailed geometric models rendered using graphics hardware, OpenGL and OSG tools are needed. However, the computational complexity is increased, and the utility function will be non-smooth. Thus, the requirements on the optimization routines become even harder.

9.1 Occlusion Model Based on the Angle of Incidence

One simple model of occlusion is to consider the angle of incidence to a feature in the horizontal plane, i.e. the angle of depression is ignored.

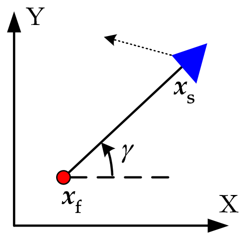


Figure 9.2: γ, angle of incidence.

Each feature has a visibility function $f_{occ}(\gamma)$ where γ is the angle of incidence, see Figure 9.2. Furthermore, the function is bounded as $0 \leq f_{occ}(\gamma) \leq 1$ and defined for $\gamma \in [-\pi, \pi)$. f_{occ} near one means that the feature is seen and f_{occ} near zero that the feature is occluded. It is possible to achieve an occlusion model that also depends on the angle of depression by modification of the range parameters r_{max} and k_R introduced in Chapter 5.

Figure 9.3 shows four examples of the visibility function $f_{occ}(\gamma)$ and the corresponding symbol that will be used in the next chapter. Figure 9.4 shows an example of how the occlusion may be defined in a road surveillance example with foliage occlusion.

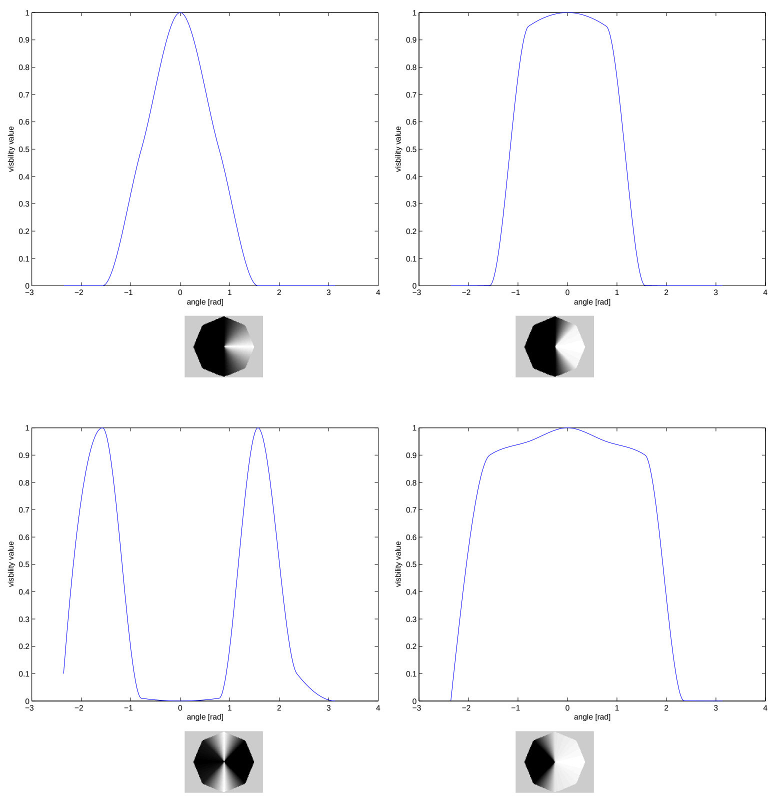


Figure 9.3: Four examples of the visibility function $f_{occ}(\gamma)$.

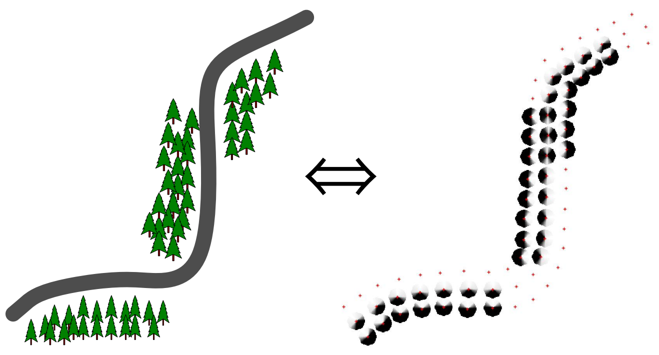


Figure 9.4: An example of how the 2d occlusion models may be defined in a road surveillance example. A number of grid points are placed along the road. Each grid point has an occlusion model, indicated by the circular symbol.

9.2 Digital Elevation Models

Typically current maps have some associated elevation data usually organized as a matrix of heights, a digital elevation model (DEM). Using laser data collected over the terrain in the scenario studied in this report a DEM with a resolution of 2x2 m squares was constructed, Figure 9.5. The DEM is a combination of three height maps Z_{max} with the maximum, Z_{min} with the minimum and Z_{mean} with the mean of the measurements within each 2x2 m cell. From these three matrices a new model was constructed using the mean value whenever it was within 1 m from the min-value to get good representation of the ground and otherwise the one extreme closest to the mean to achieve a good representation of occluders and free space. The visibility is modelled in

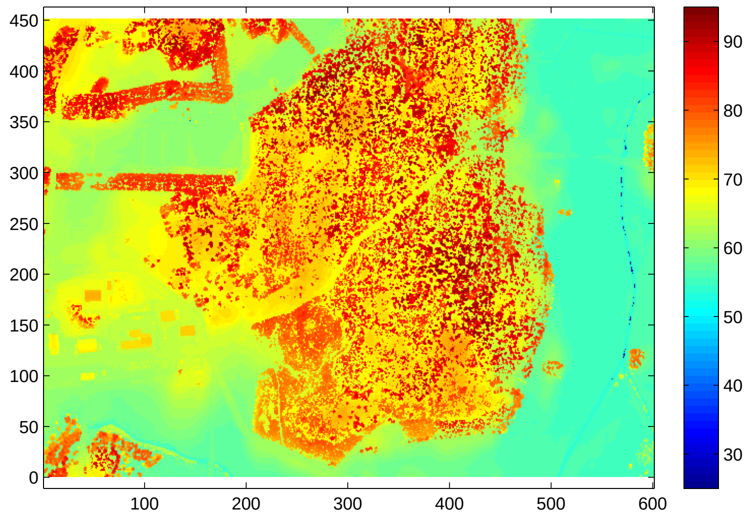


Figure 9.5: Digital elevation model of selected scenario. Each cell represents a 2x2 m squares using the maximum elevation in the cell if the mean elevation is closer to max than min to not underestimate occlusion from tree cover.

the DEM by comparing the line of sight by the interpolated height from the DEM along a line with the same ground coordinates as the line of sight. In Figure 9.6 a typical sight-line is illustrated on the DEM. The corresponding height is illustrated in Figure 9.9

If the z-coordinate of the line of sight $z(i)_{LOS}$ exceeds the interpolated height $z(i)_{DEM}$ of the cell c_i it is considered free otherwise the occlusion is modelled as proportional to the difference as:

$$occ_i = \exp\left(-\frac{(z(i)_{DEM} - z(i)_{LOS})^2}{2d^2}\right) \tag{9.1}$$

where d is a tuning parameter to model the amount of occlusion as a function of how close to the treetop the line of sight passes through the cell. Note, this model of occlusion is an ad hoc model and has not been physically derived. The occlusion is then calculated as the product of the occlusion in all cells

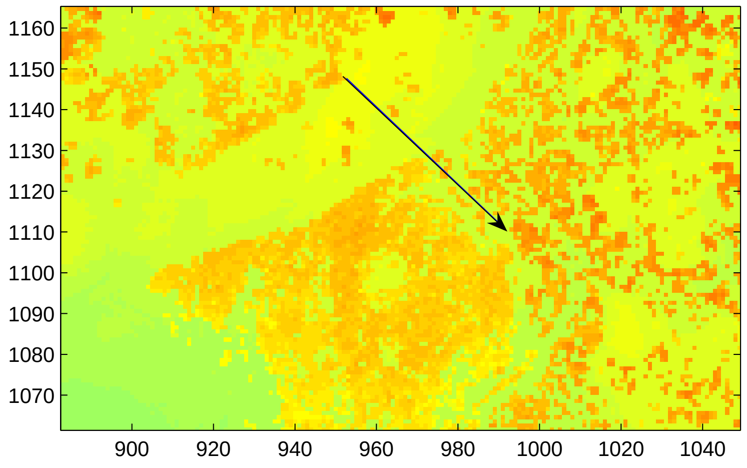


Figure 9.6: A sight-line superimposed on a zoom-in of the DEM of the scenario.

along the line of sight

$$f_{occ}(\mathbf{x}_s, \mathbf{x}_c) = \prod_{i=1}^{n} occ_i \tag{9.2}$$

Since the occlusion from an individual tree introduces non-linearities complicating the optimization two more DEM:s are introduced. They are based on grayscale morphology using spherical construction elements. The result of the morphological operations dilation and erosion can be thought of as the height of the center position of a sphere resting above the surface (dilation) or below the surface (erosion). The operation of first eroding and then dilating the DEM gives the opening of the DEM representing an optimistic view removing geometries smaller than the sphere, Figure 9.7. The other operation closing are based on the opposite sequence of dilation followed by erosion giving a pessimistic model of the visibility, Figure 9.8.

In this report the occlusion result from the two views are combined by a tuning parameter in an ad hoc way to avoid the local minima in an algorithm similar to continuation methods [3]. In continuation methods the solutions of a family of “smoother” problems (occlusion on the eroded/dilated DEM) related to the original problem are used as initialization to the real problem, thus avoiding many or, in the best case, all local minima.

In future work, they can also be used to represent the effect of *Planning & Navigation Support*, see Section 2.5. The idea is that if the geometry or navigation is uncertain Murphy’s law would cause the expectation of future information beyond the short control horizon to equal the pessimistic view. However, if sensor resources are allocated to model an area beyond the short control-horizon it will allow future control actions that eliminate the occlusion of singleton trees making the optimistic view the correct expected future observations. For the short control horizon the final optimization is of course always based on the original DEM

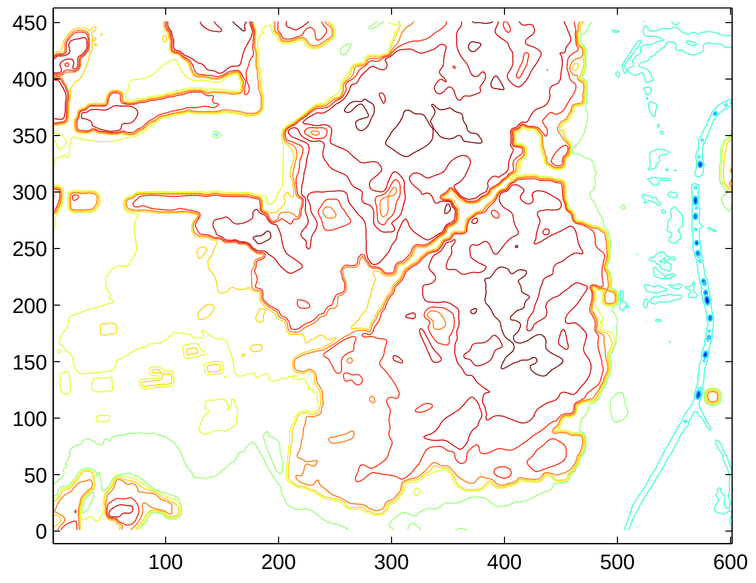


Figure 9.7: Contours for every 5 m in height, for an optimistic occlusion model, produced using grayscale erosion followed by dilation with a spheroid of 20m radius and a height of 7m.

9.3 Hardware Accelerated Computer Graphics Models

Under the influence of the computer-gaming industry a performance-race is underway for the graphics boards of today. Given a good geometric model, good performance is to be expected for the calculation and removal of hidden surfaces in a visual image. Parts of the model used in this report are illustrated in Figure 9.10.

The need for a modelling of occlusion in the sensor is thus possible to meet using calls to the graphics board. One remaining problem to achieve high performance is the bottleneck of communicating result to and from the graphics board so the best results are achieved if as much as possible can be calculated on the board. Similar problems receiving interest within the computer graphics (CG) community is shadow-casting and occlusion modelling for fast walkthroughs. Real-time rendering of shadows gives important clues to the structure of a scene and contributes to the realism of the images [21]. For this reason, methods for shadow calculation directly on the graphics board exist today. However, the geometric problem of computing shadows of an extended light source is equivalent to the problem of calculating the visibility of the same surface from positions along the extended light source, and actually some algorithms is based on that by rendering the scene from the point of the light source [38]. The purpose of occlusion modelling for fast walkthroughs is to avoid sending much of the geometry to the rendering pipeline to speed up rendering of large models, for some examples see [43], [11], [42], [20] and [22] . Post processing of this sort would obviously also give a speedup of the visibility calculations needed for the present information-gathering problem.

Due to limited resources within the present project these extensions have not been tested. The possibilities have instead been explored using a sim-

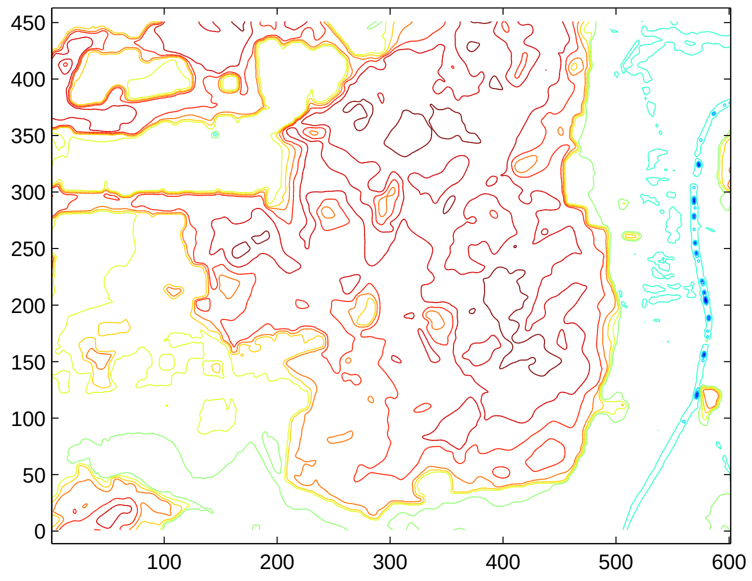


Figure 9.8: Pessimistic occlusion model, where the contours represent steps of 5m in height. The DEM was produced using grayscale dilation followed by erosion with a spheroid of 20m radius and a height of 7m. Notice that the road between the two woods is almost completely covered except for the glen in the south-west.

pler interface to the graphics board based on an open source toolkit[1] using an in-house developed Matlab interface [7]. This interface does not achieve the real-time properties expected in a final system but gives an indication of future possibilities. The model used is a high fidelity model based on measurements from an airborne laser with normally about 16 measurements/m^2. The measurements have been post-processed to extract ground, trees [37] and buildings [19]. Using this model gives indications on future possibilities when given high resolution a priori models. In principle the algorithm is based on directly simulating the view of the sensor as in Figure 9.10. By adding coloured patches to the otherwise grayscale model the amount of the patches visible from the viewpoint is easily extracted as in Figure 9.11. A second view using only the scene patches gives an unoccluded view, where each patch is colour coded to achieve direct identification as in Figure 9.12. Given the pixel count (or area) $Au(\mathbf{x}_s, \mathbf{x}_c)$, of the unoccluded patch, and the pixel count ("weighted area") $Ao(\mathbf{x}_s, \mathbf{x}_c)$ of the possibly occluded patch, the occlusion function is directly calculated as the ratio:

$$f_{occ}(\mathbf{x}_s, \mathbf{x}_c) = \frac{Ao(\mathbf{x}_s, \mathbf{x}_c)}{Au(\mathbf{x}_s, \mathbf{x}_c)} \tag{9.3}$$

The corresponding views for a more realistic vantage point are given in Figures 9.13, 9.14 and 9.15. Note that in the currently presented algorithm, self occlusion is not modelled. The patches are assumed to be so small and on reasonably flat surfaces so that no terrain within the patch can seriously occlude other areas of the same patch.

[1]Open Scene Graph http://www.openscenegraph.org/

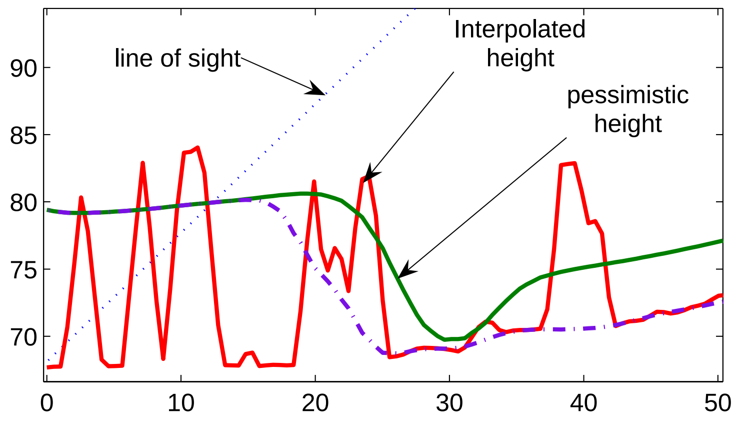


Figure 9.9: A plot of the terrain profile for a given line of sight. The height profiles are based on the original DEM, the optimistic DEM (dash-dotted) and the pessimistic height.

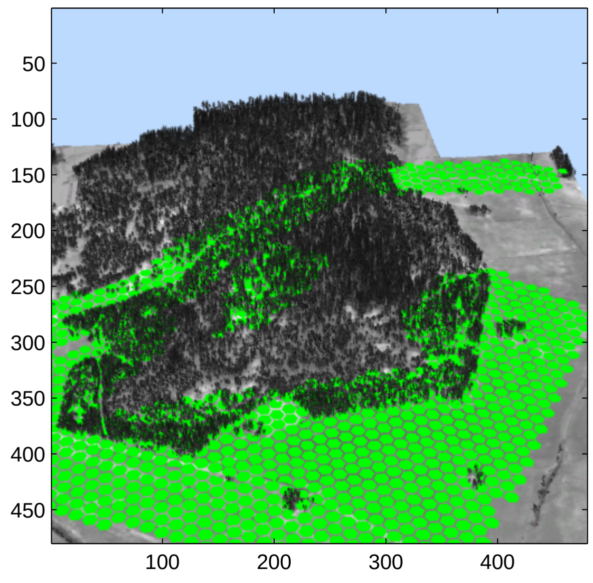


Figure 9.10: A rendering of a high resolution model with patches representing a surface cover task superpositioned on the ground surface, viewed from a high vantage point.

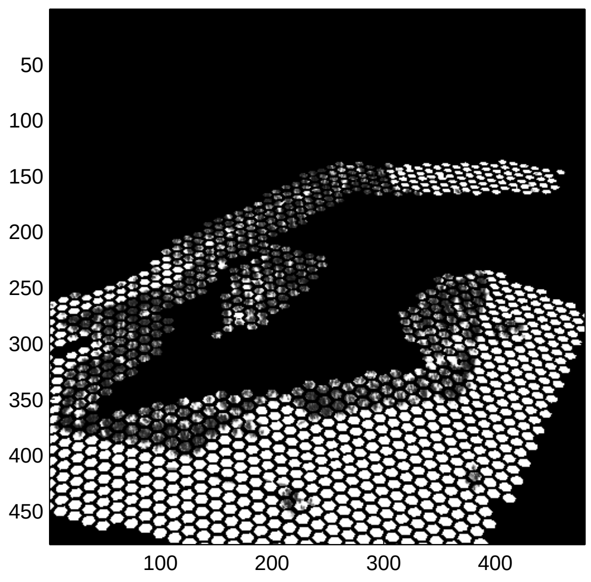


Figure 9.11: The occlusion of surface patches using a high resolution model with patches representing a surface cover task superpositioned on the ground surface, viewed from a high vantage point.

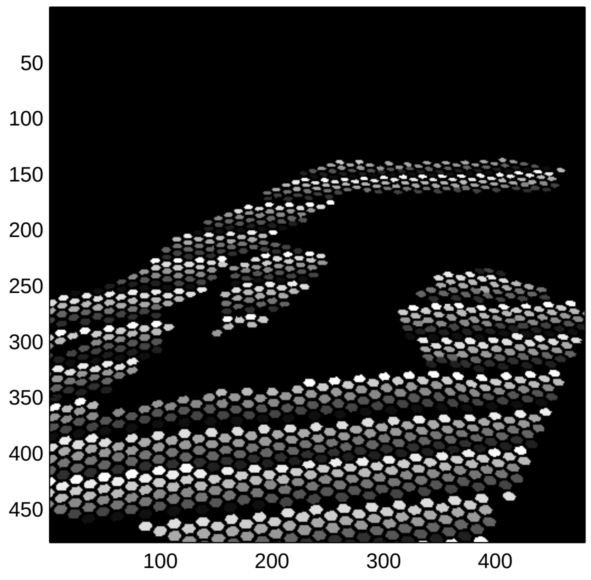


Figure 9.12: A rendering of colour coded surface patches representing a surface cover task. The red channel is illustrated in gray scale.

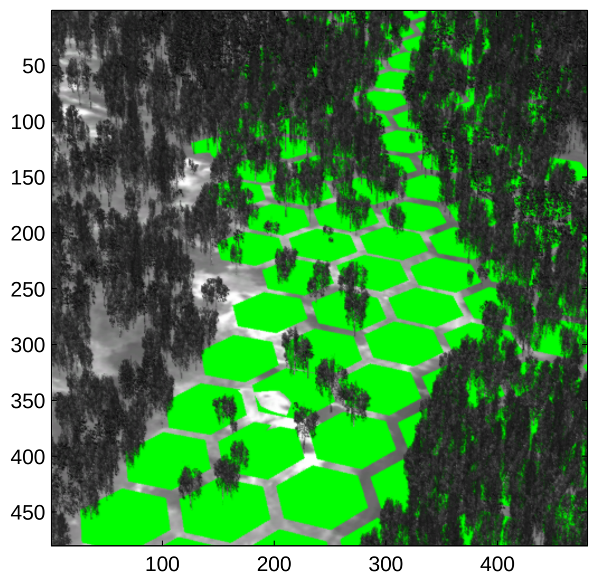


Figure 9.13: A typical view of a high resolution model with patches representing a surface cover task superpositioned on the ground surface.

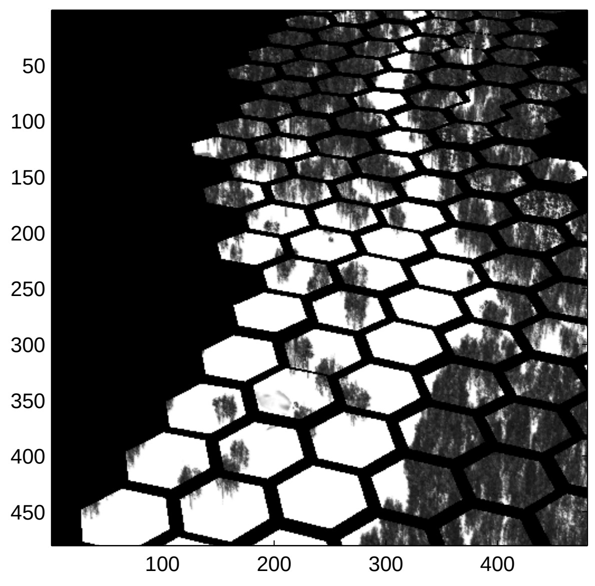


Figure 9.14: The occlusion of surface patches in a typical view using a high resolution model with patches representing a surface cover task superpositioned on the ground surface.

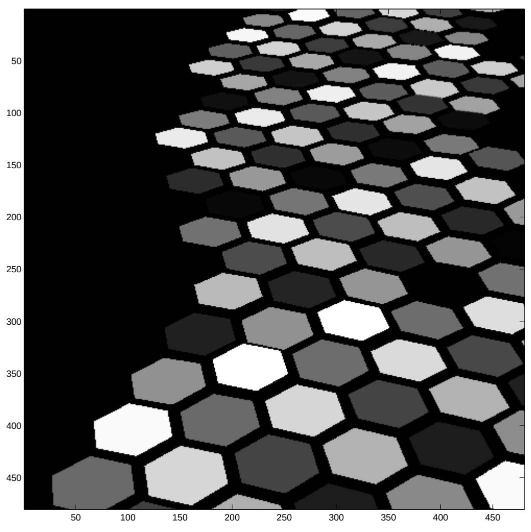


Figure 9.15: A rendering of colour coded surface patches representing a typical view from a surface cover task. The red channel is illustrated in gray scale.

10 Simulation Results

In this chapter we put the pieces presented in this report together. We use concurrent path and sensor planning to solve a number of surveillance missions. The simulation results are highly dependent on the simulation settings. The purpose of this chapter is not to show how different parameter settings affect the result.

10.1 Road, Building and Objects Surveillance; Cubic Spline Implementation

The task in this section is to survey a road segment and a building, and to detect and localize an object on the road, see Figure 10.1. A number of grid

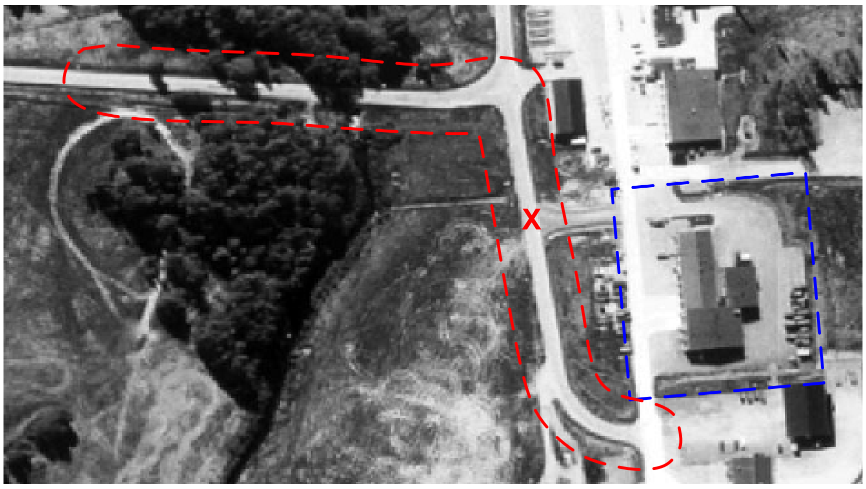


Figure 10.1: Road surveillance scenario. A road segment (red lines) and a building (blue rectangle). In addition, there is an object on the road.

points along the road and around the building are set out. The number of grid points is a compromise between long computational time and fair area coverage. An occlusion model is connected to each grid point, and points on the edge of a wood and around the building are partly occluded, see Figure 10.2.

In this simulation a replanning is done when the object is "detected" and "localized". An object is "detected" and "localized" if its information value (here the determinant) is above some thresholds $I_{detected}$ and $I_{localized}$, respectively. The logarithm-determinant utility and *fminunc* in Matlab Optimization Toolbox 2.2 are used.

Figure 10.3 shows the path of the vehicle and the information values at each grid point. Figure 10.4 shows, besides the vehicle path, the gaze direction of the sensor at some points. Flying around the building is reasonable according

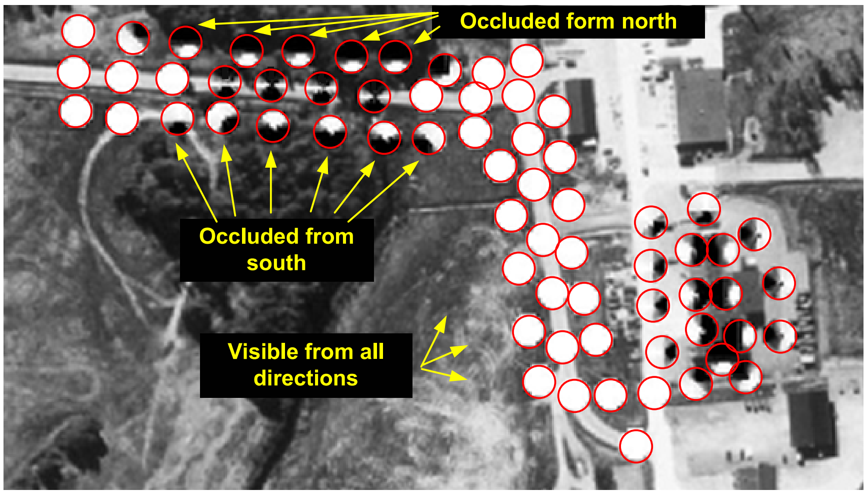


Figure 10.2: An occlusion model is associated with each grid point.

to the occlusion definitions of the grid points, the sensor can not “see” all grid points from just one direction.

Figures 10.5-10.6 show two snapshots of the simulation. The sensor footprint is illustrated as a contour plot, white represents the area that the sensor is “seeing” and the position of the vehicle is indicated by the star/circle. In Figure 10.6 the sensor is gazing at the detected object. Note the covariance ellipse of the object. Note that this covariance is based on the information state of the object, i.e. it is related, but not equivalent, to the position uncertainty. The yellow pentagram shows the position of the vehicle when the object was “detected” and a replanning was performed.

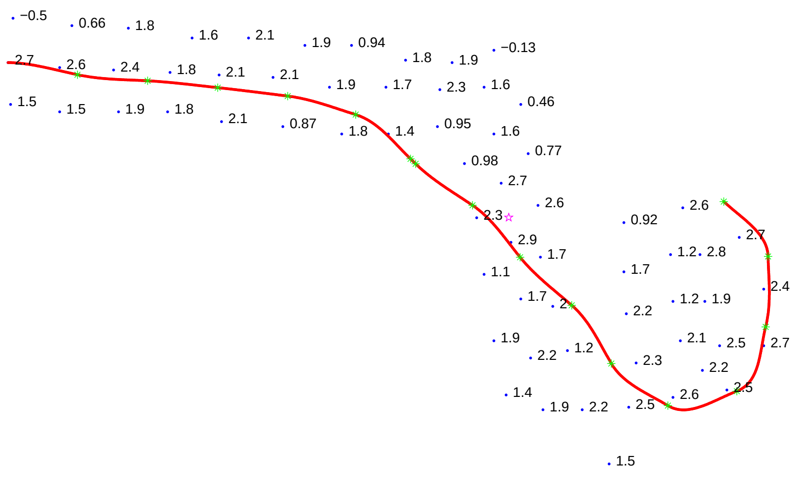


Figure 10.3: Vehicle path and information values at each grid point.

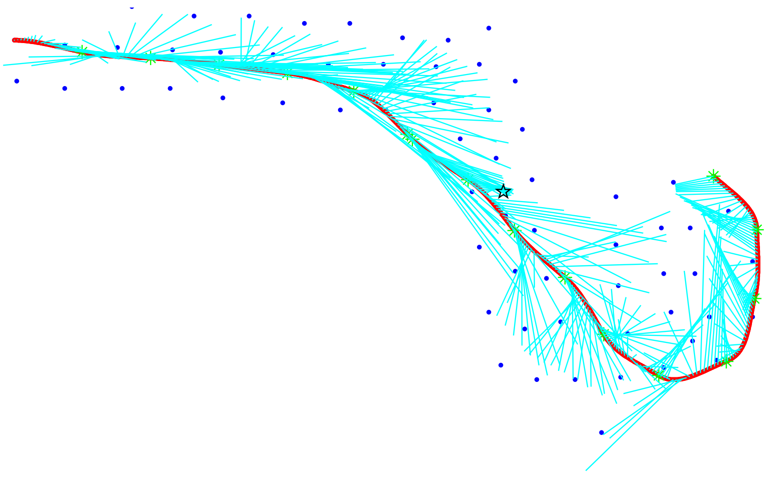


Figure 10.4: The gaze direction of the sensor at some points.

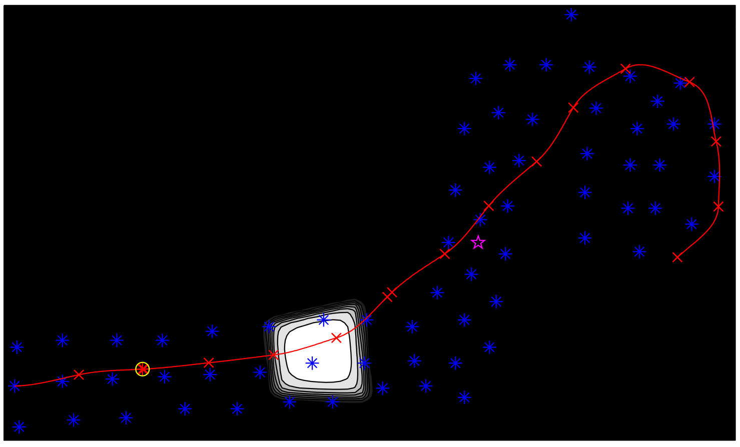


Figure 10.5: Sensor footprint illustrated as a contour plot.

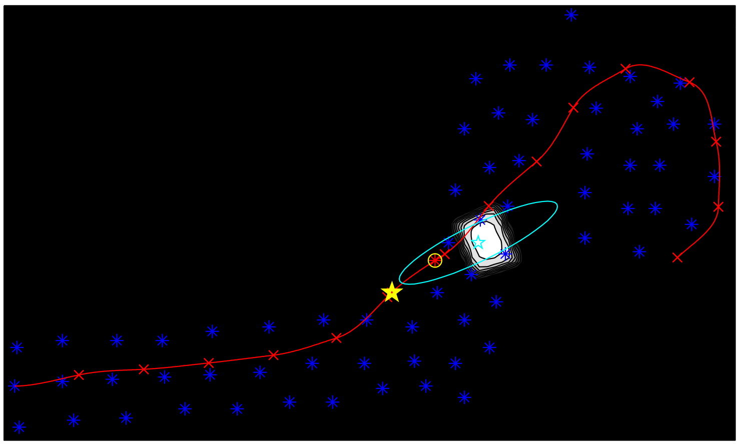


Figure 10.6: Sensor gazing at the object.

10.2 Road and Objects Surveillance; Cubic Spline Implementation

This simulation is similar to the simulation in the previous section. The mission is to survey a road segment, some edge of woods and five objects. An occlusion model is connected with each feature. In addition, each feature has also a range function with an individual maximum range parameters. In general, features in the wood have short maximum range and features on a field have long maximum range. Figure 10.7 shows the position of all features and their occlusion symbols. Unlike the previous simulation, the objects are known beforehand.

At each planning step, the optimization is over one point of the flight path and two points of the sensor path. The logarithm-determinant utility and *fminunc* in Matlab Optimization Toolbox 3.0 are used.

The simulation result can be seen in Figures 10.8 and 10.9. Figure 10.8 only shows the vehicle path, in Figure 10.9 the sensor pointing directions are also given. Two snapshots of the simulation are shown in Figures 10.10 and 10.11.

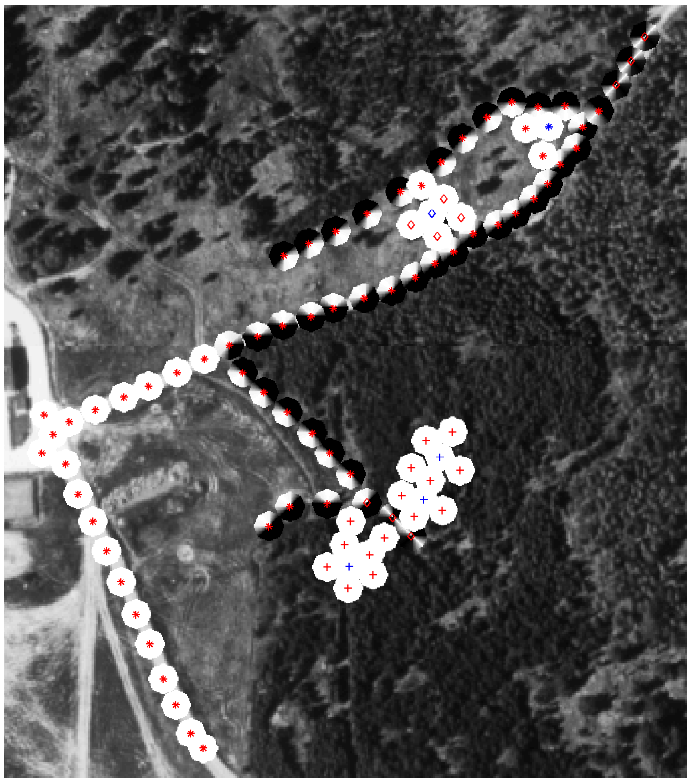


Figure 10.7: Road and objects surveillance scenario. Feature positions and their occlusion symbols. Plus, diamond and star signs represent short, medium, long maximum range, respectively.

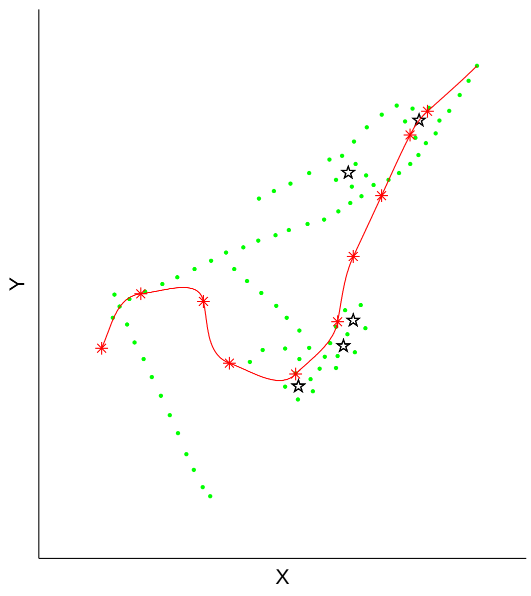


Figure 10.8: The vehicle path is in red, the (re)planning locations are marked with red stars. Green dots are the grid points and pentagrams are the objects.

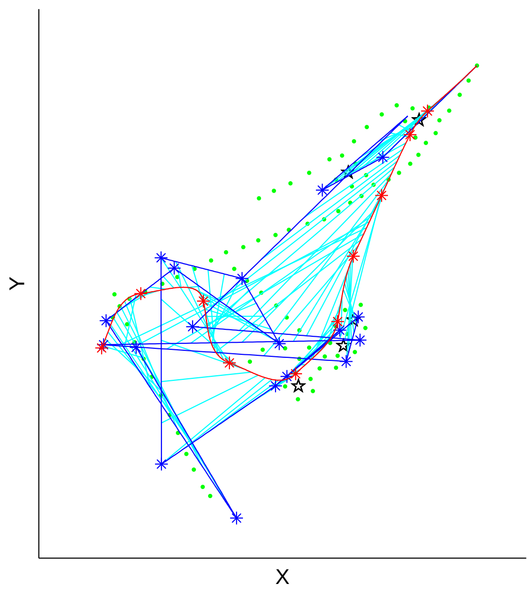


Figure 10.9: The vehicle path is in red and the sensor pointing path in blue. Some of the aiming directions between the two paths are also plotted as cyan lines. Green dots are the grid points and pentagrams are the objects.

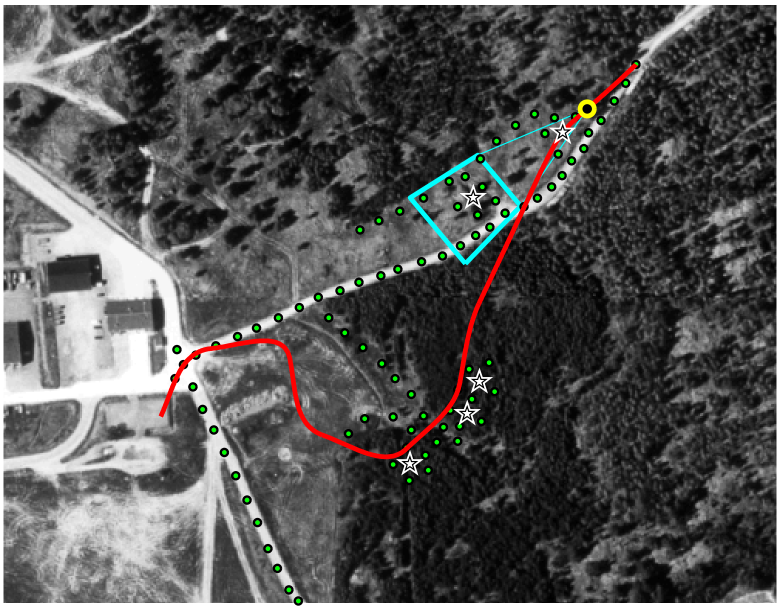


Figure 10.10: Simulation snapshot 1.

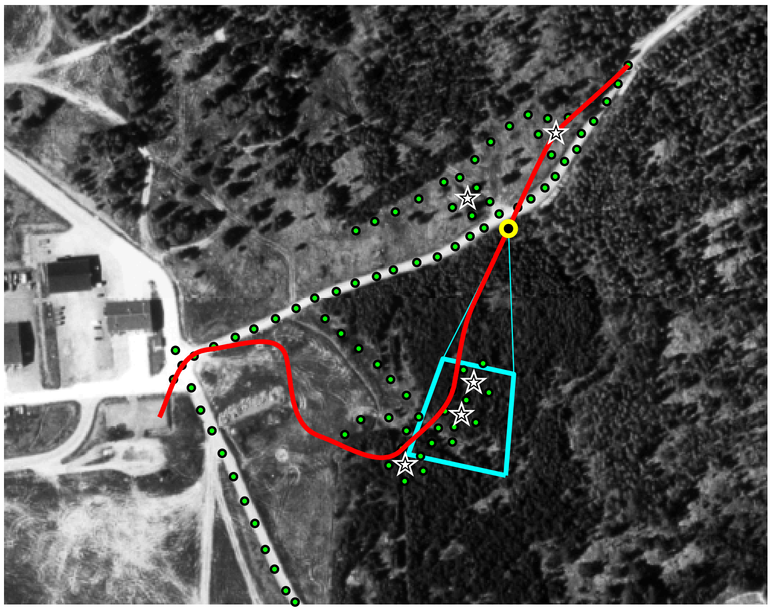


Figure 10.11: Simulation snapshot 2.

10.3 Road and Area Surveillance; NURBS implementation

In this section an example of the NURBS implementation of the planning hierarchy, presented in Section 7.5.1, is given. The area grid points are the same as in the previous section, but no objects are included. Two curves are used, one curve representing the vehicle trajectory and one curve representing the sensor aiming directions. We have three planning levels, of which the first level is given by a long term plan, see Chapter 6. Recall the parameters in Section 7.5.1.

n_i	number of optimization points
d_i	control points divider, relative plan $i-1$
j_i	planning dimensions

In this simulation we have the following values

$$n_2^1 = n_2^1 = n_2^1 = n_2^1 = 4$$
$$d_2 = d_3 = 2$$
$$j_2^1 = j_2^2 = j_3^1 = j_3^2 = [x, y]$$

where super index 1 is the vehicle curve and super index 2 is the sensor pointing curve. Thus, the "resolution" is doubled at each planning level and the planning of curves is done in a plane, i.e. the z (altitude) component is constant. The number of optimization parameters at both planning level 2 and 3 are $n * \dim(j) * \#curves = 4 * 2 * 2 = 16$.

Both planning level 2 and 3 are using a field-of-view model based on the image projection model in Section 8.4.1. No occlusion model is used at planning level 2, i.e. the features are fully visible from all directions. Planning level 3 has an occlusion model based on a DEM model, see Section 9.2. The logarithm-determinant utility and *fminunc* in Matlab Optimization Toolbox 3.0 are used. However, note that *fminunc* is not suitable for problems with the DEM-occlusion model. At first hand, the reader should consider this simulation as an example of the proposed NURBS planning hierarchy.

Figure 10.12 shows the long term plan (red curve), the vehicle curve and the sensor pointing curve are initially the same. The positions of the area grid points are marked with green dots. The planning results from level 2 and 3 can be seen as thick curves in Figures 10.13 and 10.14, respectively. The red curves are the vehicle path and the blue curves are the sensor pointing path on the ground. The thin curves are the planning result from the previous planning level or planning step. Replanning points are marked with black circles, the path stretch before and near this point are locked. Figures 10.15 and 10.16 shows the planning result of the second planning step and Figures 10.17 shows the third planning step.

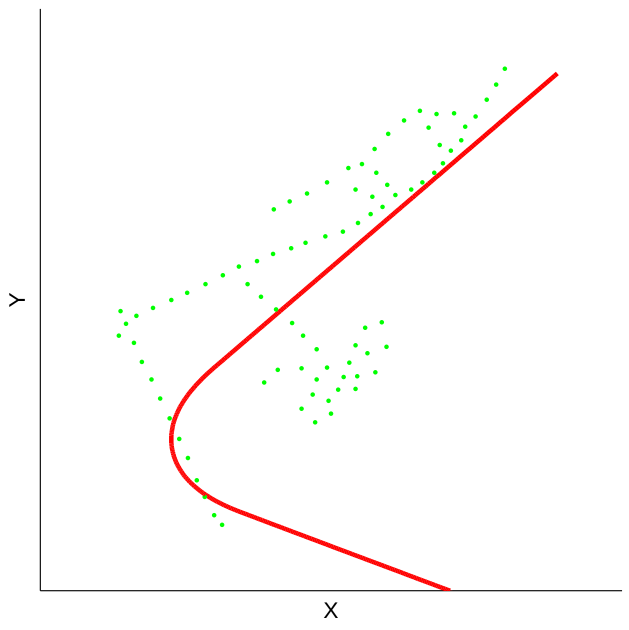


Figure 10.12: Planning level 1. Long term plan result.

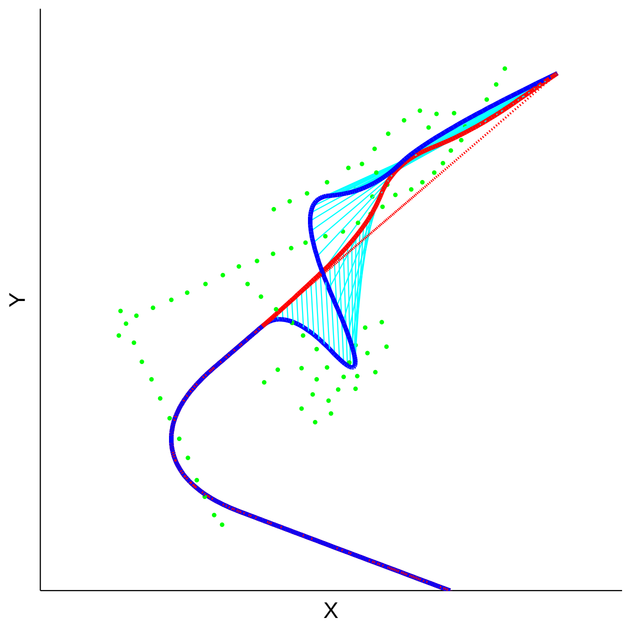


Figure 10.13: Planning step 1. Result from planning level 2.

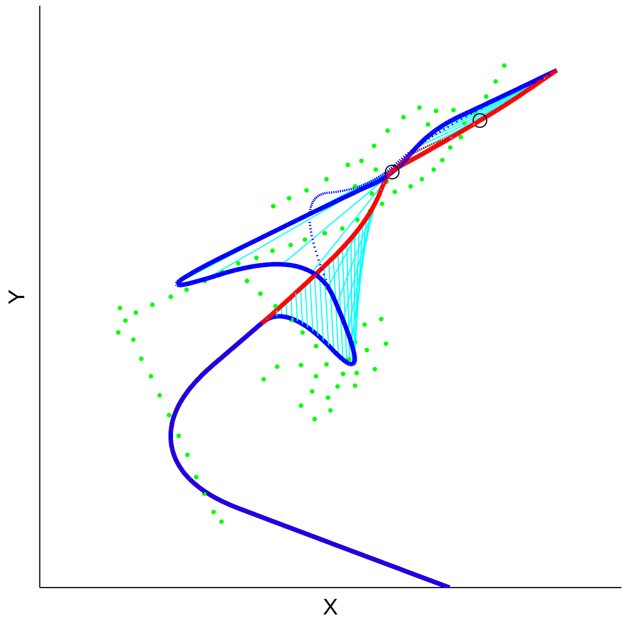


Figure 10.14: Planning step 1. Result from planning level 3.

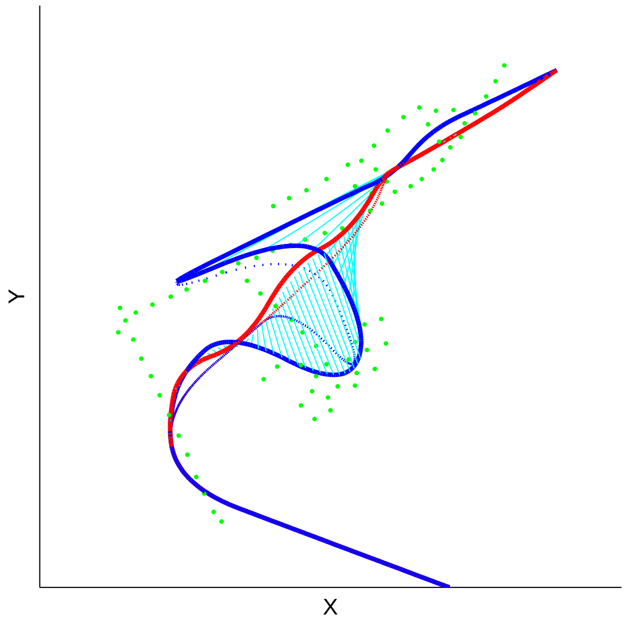


Figure 10.15: Planning step 2. Result from planning level 2.

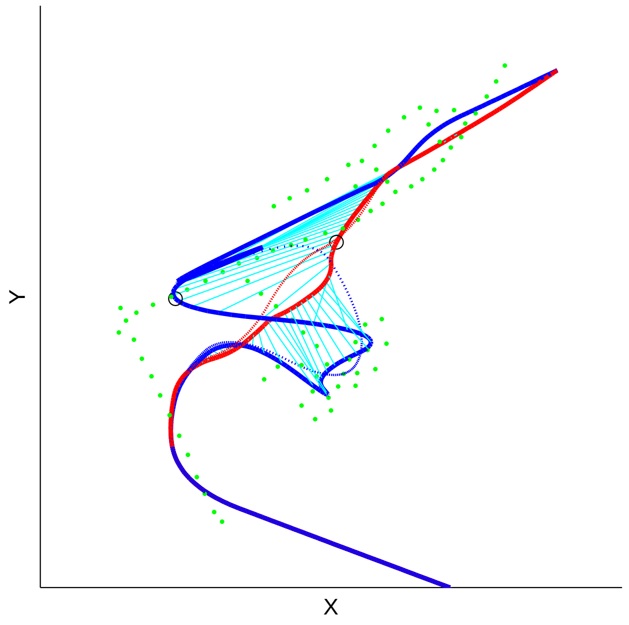


Figure 10.16: Planning step 2. Result from planning level 3.

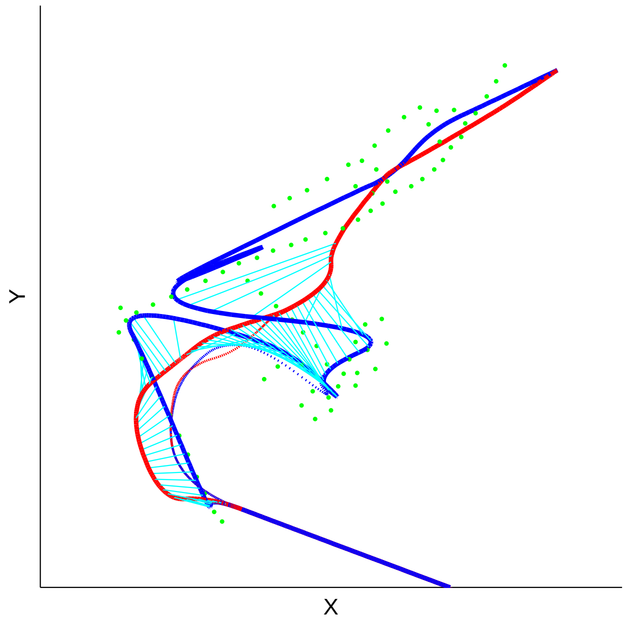


Figure 10.17: Planning step 3. Result from planning level 2.

11 Theoretical Aspects of the Method

In this chapter, which is to a high degree independent of the rest of this report, the general problem of information based guidance is addressed from the point of view of likelihood based inference and differential geometry. It is aimed at showing, or at least indicating, that the information filter approach has a much wider scope and validity than it was originally designed for [16]. In the usual approach, the information filter is motivated by reference to a normally distributed error model, and the Fisher information matrix enters essentially as the inverse of the covariance matrix (which is correct for a translational, normal measurement model). The variables, however undergo a variety of nonlinear transformations in the course of the derivation, and hence it is less clear what the final formulas will tell us. The form of the information filter, on the other hand, coincides with that of the ubiquitous Extended Kalman Filter, rewritten in terms of the inverse of the variance. The theoretical rationale for the Extended Kalman Filter is however rather meager, though [26] gives some results. As an alternative to the normal/variance/Kalman approach we here present a likelihood oriented approach, which so to speak takes the term "information filter" literally.

This approach is still somewhat heuristic, to the extent that it assumes that the likelihood function is sufficiently well represented by the location of its maximum (which is assumed to be unique) and by the Hessian at that point ("the observed information"). (This will indeed hold asymptotically.) It is also assumed that the "best" planning for future information gain is to maximize the most likely expected future information. From a decision theoretical point of view, this is only one choice (albeit quite natural) among many possible.

In this chapter, the presentation is rather demanding, giving only a very superficial overview of underlying mathematics and assuming familiarity with some differential geometric concepts. For a good background in differential geometrical terminology, the reader may want to consult a text like [1]. A good book on likelihood based inference is [36]. After a section on the different approaches to statistics, the geometry of parametric models is presented. This is a quite new subject ("statistical geometry" or "informational geometry"), which may be studied in [18, 4, 31, 25]. A recent good book on optimal control is [2].

11.1 A Short Overview of Statistics

11.1.1 Frequentists, Bayesians and Likelihoodists

In spite of the well-known rigor of modern probability theory with its axiomatics and highly technical architecture, there is at present no consensus about the 'practical' interpretation of probability and its application to real-life problems in statistics. Three main schools of statisticians may be identified: frequentists, Bayesians and likelihoodists [13, 36]. Their differences may be illustrated by their respective approaches to, say, interval estimation of an unknown parameter.

The frequentist approach is to construct some (random) interval from a random variable. For a given sample this gives a certain *confidence interval*, with a 'confidence' defined as the probability with which the random interval covers the true value of the parameter. This is hence an average property of the *method*. It is easy to construct examples where a confidence interval with high confidence in some (improbable) outcomes may be such that one may know with certainty that the true parameter value is outside the interval. An (ultra-)orthodox frequentist would however still report this confidence interval. Furthermore, an orthodox frequentist would refrain from drawing rather safe conclusions from data on the grounds that the sampling method *on an average* would be unsufficient for this conclusion. The frequentist approach has a nice and clean foundation in probability theory, but is perhaps best aimed at judging methods in advance, before one has any actual data.

The Bayesian approach is akin to consider the unknown parameter itself as a random variable, to the extent that it is assigned a probability distribution. Since we think of this parameter as a fixed but unknown value, this implies an intepretation of probability different from that of the frequentists (and likelihoodists). Indeed, the Bayesian approach is to think of probability as representing our state of knowledge. This 'state of knowledge' may be objective or subjective. In the objective case, the probability distribution is constructed from known facts according to well-defined rules, independent of any choices (such as parametrizations), states of mind or mood that we may have. A subjective Bayesian probability may allow such dependencies. In the interval estimation case, the parameter is assigned an apriori probability distribution, independent of any outcomes, which represents our prior knowledge (or lack thereof) of the parameter. When a sample is drawn, the probability distribution is updated to a posterior probability distribution by means of Bayes's theorem. (This procedure may be repeated recursively.) Finally, a Bayesian interval can be constructed so that it contains a prescribed posterior probability. The Bayesian approach is formally simple and clean, representing all kinds of uncertainties in a unified way as probabilities. This is however also its great weakness, since representing complete ignorance in terms of probability leads to strange paradoxa.

The likelihoodist approach would be to compute the likelihood function of the parameter, given an outcome. A likelihood interval is an interval such that the (normalized) likelihood function is less than a prescribed value outside the interval. This depends on (the model and) the *outcome* alone and is thus independent of the sampling method's average properties or any choice of prior probability. When more data are obtained, the likelihood function is updated according to simple rules (loglikelihoods are added) so this approach is also recursive. It has also been suggested that subjective prior likelihoods may serve as representation of subjective or vaguely percieved prior knowledge, thereby providing an alternative to subjective Bayesianism.

In most real-life applications the three schools of statistics lead to comparable results, at least asymptotically for large samples. This chapter is written mainly with the likelihoodist perspective.

11.1.2 The Envelope Paradox

You are offered to choose one of two envelopes, one of which contains twice the amount of money as the other [36]. You are not told the size of these amounts or which envelope contains the larger amount. You make a choice and when you open your envelope, you find that the amount of money inside is x. At this point you are offered the possibility of shifting to the other envelope. Since the other envelope contains either $\frac{x}{2}$ or $2x$, both possibilities being equally likely, it is tempting to compute an expected value as $y = \frac{1}{2}\frac{x}{2} + \frac{1}{2}2x = \frac{5x}{4}$ which is $> x$. So, you should indeed select the other envelope instead, regardless of the actual value of x. This holds even if you never opened the first envelope (since x is arbitrary). Obviously this reasoning is paradoxial, since if you do not open the first envelope and then change to the second envelope, you are back in a situation equivalent to that, when you had just chosen your first envelope.

This paradox has been analysed by many people, as has the rather similar St. Petersburg paradox. Bayesians assign prior probabilities (in different ways) to the amounts of money in the envelopes.

The likelihoodist solution is simple: the possibilities $\frac{x}{2}$ or $2x$ being equally *likely* (equal likelihood) does not make them "equally probable", in the sense of probability theory. The two cases are like two possible values of an unknown parameter, without any prior probabilities defined. There is therefore no rationale for considering the above average value y as an *expectation*, and the paradox evaporates.

11.1.3 Inference Problems and Decision Problems

In statistics two main types of problems may be identified, inference problems and decision problems [47]. Suppose we have a sample x_1 from a random variable with probability density $p(x, \theta)\, dx$. A typical inference problem is to give a point estimate $\hat{\theta}(x_1)$ of the unknown paramenter θ. The estimate should have "good statistical properties" in some sense made precise by standard statistical theory. But it should also be objective, in the sense that whatever inference is made, it should not depend on irrelevant parameters. This is the paradigm of "the scientific method". The conclusions drawn should be invariant under symmetry operations leaving the relevant parameters invariant.

In a decision problem, on the other hand, we may acknowledge that in some problems, it may be for instance more advantageous to make point estimates that are in general larger, say, than the true value. A decision problem hence needs, for its formulation, a choice of *utility* function, the value of which is maximized or at least used as a measure of how good a decision strategy is.

Formally, inference problems may be regarded as particular decision problems, where the utility function satisfies some symmetry or naturality condition. For instance, the maximum likelihood point estimator is characterized by utility being equal to likelihood, while the "corrected" maximum likelihood estimator (achieving "unbiasedness" *in the decisionmaker's preferred coordinates*) relies on a subjective utility function.

In this chaper the view is held, that information-based guidance is of a decision theoretic nature, but that the subjectivity only enters when a utility function is formed from the information matrix. Otherwise, it is an inference problem. Its formulation should therefore enjoy all symmetry properties of a true inference problem, which includes reparametrization invariance of the stochastic variables, reparametrization invariance of the parameters etc.

The systematic study of such reparametrization properties is the subject of a recent subject: statistical geometry [18, 4, 31, 25].

11.2 Statistical Manifolds

In this section, a differential geometric oriented presentation of parametric models is given, *cf.* [18], [31].

11.2.1 Parametric Models without Parameters

A statistical model may be described as follows. Given the n-dimensional manifold M, a subset, S of the smooth nowhere vanishing probability densities on M (i.e. elements ϕ of $\Gamma(M, |\Lambda^n| M)$ satisfying $\phi > 0$ everywhere and $\int_M \phi = 1$) is a 'parametric model'. If S is given a manifold structure (of dimension N, say) such that the pointwise evaluations mapping ρ_{SM}

$$\rho_{SM} : S \times M \to |\Lambda^n| M$$

is jointly smooth, then S is a *statistical manifold*, and the triple (M, S, ρ_{SM}) is a *smooth parametric model*. (The 'parameters' of the model are the local coordinate functions on S).

Our aim is to understand such smooth 'parametric' models in a coordinate-free fashion, so as to identify genuine properties of the triple (M, S, ρ_{SM}), irrespective of any choice of coordinates on either M or S. In more technical terms, we look for invariants of (M, S, ρ_{SM}) under the actions of independent simultaneous diffeomorphisms of S and M.

It turns out that the structure of smooth parametric models is so rich, that (generically) more or less unique standard parameterizations can be defined. Nevertheless, it is often useful to restrict attention to smaller sets of invariants (of certain forms), that do not suffice to determine canonical coordinate systems.

11.2.2 Examples

1) An *ad hoc* model on the unit circle $\mathbb{S}$ is given by

$$\begin{aligned} M &= \mathbb{S}_x (= \mathbb{R}_x / 2\pi\mathbb{Z}) \\ S &= \mathbb{S}_{x_0} \\ \rho(x_0, x) &= \frac{2 + \sin(x - x_0)}{4\pi} |dx| \end{aligned}$$

We see that a simultaneous rotation of both circles M and S leaves the model invariant. The formulas suggest that S may be identified with M via $x \leftrightarrow x_0$, and that this would provide a preferred point estimate of $x_0 \in S$, given $x \in M$. On the other hand, by Moser's theorem below, there exists a diffeomorphism of S whose pullback of the density $x_0 \in S$ is $\frac{1}{2\pi} |dx|$, which does not seem to single out any natural point estimate. Hence, if the identification $x \leftrightarrow x_0$ (or the corresponding point estimate) has any invariant meaning at all, it is a property of the geometry of the whole model (M, S, ρ_{SM}) and not of any single density x_0.

2) The one-dimensional normal family is given by

$$\begin{aligned} M &= \mathbb{R}_x \\ S &= \mathbb{R}_\xi \times \mathbb{R}^+_\sigma \\ \rho(\xi, \sigma, x) &= \frac{e^{\frac{-1}{2\sigma^2}(x-\xi)^2}}{\sigma\sqrt{2\pi}} |dx| \end{aligned}$$

Most of the discussion of the preceding example carries over to this one. This model is an example of an exponential family, which implies a benign behavior. The property of being exponential is invariant, and is related to the vanishing of the curvature of an appropriate Amari connection. We shall return to this example in order to compute, among other things, its Fisher information metric.

11.2.3 Moser's Theorem

The following is a well-known theorem of differential geometry: two nowhere vanishing volume forms on a compact, oriented manifold M are equivalent via the pullback of a diffeomorphism, if and only if they have the same total volume. This can be proved by a Lie transform method [1].

Apart from technicalities (compact and oriented manifold), this theorem tells us that, generally, single probability densities on M have no individual properties. As the second simplest invariants would be properties of *pairs of densities*, we turn our attention to these.

11.2.4 Invariants of a pair of probability densities

Let ρ_1 and ρ_2 be two nowhere vanishing probability densities on the manifold M. Their quotient q

$$\rho_2 = q\ \rho_1$$

is a mapping $q : M \to \mathbb{R}^+$. The push-forward of ρ_1 (as a measure), $q_*\rho_1$, is a probability measure on $\mathbb{R}^+$, whose probability function, F,

$$F : \mathbb{R}^+ \to [0,1]$$

is a function space invariant of the pair ρ_1 and ρ_2. It is our conjecture that there are no further pair invariants. This would imply that any scalar pair invariant may be written in terms of invariants of the form

$$\int\limits_M f(q)\,\rho_1$$

and it is clear that for every function f, this expression is an invariant.

For the special choice

$$f(q) = -\ln q$$

this is the famous Kullback-Leibler divergence and for the choice

$$f(q) = \sqrt{q}$$

this is the scalar product of the fractional densities $\sqrt{\rho_1}$ and $\sqrt{\rho_2}$ as unit vectors in the natural Hilbert space of square-roots of probability densities. The squared Hilbert distance between $\sqrt{\rho_1}$ and $\sqrt{\rho_2}$, a.k.a. the *Hellinger metric* is

$$2 - 2\int\limits_M \sqrt{q}\rho_1$$

11.2.5 Invariants of Infinitesimally Close Pairs

Let $\varepsilon \mapsto \rho_\varepsilon$ be a smooth one-dimensional family of densities, and consider the Taylor coefficients of

$$I_\varepsilon = \int\limits_M f(q_\varepsilon)\,\rho_0$$

It holds that

$$\begin{aligned} I_\varepsilon &= f(1) + \frac{\varepsilon^2}{2} f''(1) \int\limits_M \left(\frac{\rho_0'}{\rho_0}\right)^2 \rho_0 + \\ &\quad + \frac{\varepsilon^3}{6}\left(f'''(1) \int\limits_M \left(\frac{\rho_0'}{\rho_0}\right)^3 \rho_0 + 3f''(1) \int\limits_M \left(\frac{\rho_0'}{\rho_0}\right)\left(\frac{\rho_0''}{\rho_0}\right)\rho_0 \right) + O\left(\varepsilon^4\right) \end{aligned}$$

The family $\varepsilon \mapsto \rho_\varepsilon$ is a curve on S. In terms of local coordinates $\theta^A, (A = 1..N)$, the curve is given by $\varepsilon \mapsto \theta^A(\varepsilon)$. Introducing the *Fisher information metric*

$$g_{AB} = \int\limits_M \frac{\rho'_A}{\rho}\frac{\rho'_B}{\rho}\rho$$

and the *skewness tensor*

$$T_{ABC} = \int\limits_M \frac{\rho'_A}{\rho}\frac{\rho'_B}{\rho}\frac{\rho'_C}{\rho}\rho$$

the above formula takes the form

$$\begin{aligned} I_\varepsilon &= f(1) + \frac{\varepsilon^2}{2} f''(1)\, g_{AB}\dot\theta^A\dot\theta^B + \frac{\varepsilon^3}{2} f''(1)\, g_{AB}\dot\theta^A \frac{\overset{0}{D}\,\dot\theta^B}{dt} + \\ &+ \frac{\varepsilon^3}{6}\left(f'''(1) + \frac{3}{2} f''(1)\right) T_{ABC}\dot\theta^A\dot\theta^B\dot\theta^C + O\left(\varepsilon^4\right) \end{aligned}$$

where $\frac{\overset{0}{D}\dot\theta^B}{dt}$ is the acceleration w.r.t. the Levi-Civita connection of the Fisher metric.

$$\frac{\overset{0}{D}\,\dot\theta^B}{dt} = \ddot\theta^B + \overset{0}{\Gamma}{}^B_{AC}\,\dot\theta^A\dot\theta^C$$

where $\overset{0}{\Gamma}{}^A_{BC} = \frac{1}{2} g^{AD}\left(\frac{\partial}{\partial\theta^C} g_{DB} + \frac{\partial}{\partial\theta^B} g_{CD} - \frac{\partial}{\partial\theta^D} g_{BC}\right)$ are the Christoffel symbols for the Levi-Civita connection.

With the introduction of Amari's α-connections

$$\begin{aligned} \overset{\alpha}{\Gamma}{}^A_{BC} &= \overset{0}{\Gamma}{}^A_{BC} - \frac{\alpha}{2} g^{AD} T_{DBC} \\ (\alpha &\in \mathbb{R}) \end{aligned}$$

we may also write

$$\begin{aligned} I_\varepsilon &= f(1) + \frac{\varepsilon^2}{2} f''(1)\, g_{AB}\dot\theta^A\dot\theta^B + \frac{\varepsilon^3}{2} f''(1)\, g_{AB}\dot\theta^A \frac{\overset{-1}{D}\,\dot\theta^B}{dt} + \\ &+ \frac{\varepsilon^3}{6} f'''(1)\, T_{ABC}\dot\theta^A\dot\theta^B\dot\theta^C + O\left(\varepsilon^4\right) \end{aligned}$$

where $\frac{\overset{-1}{D}\dot\theta^B}{dt} = \ddot\theta^B + \overset{-1}{\Gamma}{}^B_{AC}\,\dot\theta^A\dot\theta^C$ is the acceleration w.r.t. Amari's -1-connection.

Summing up, we see that *any* invariant measure of divergence, has an $O\left(\varepsilon^4\right)$-expansion in terms of the Fisher metric and the skewness tensor.

11.2.6 Exponential Families

In the important case, when the smooth parametric model (M, S, ρ_{SM}) has the following properties

- S is an open subset of $\mathbb{R}^N_\theta$
- $\rho_{SM}(\theta, x) = e^{c_0(x) + \theta^A c_A(x) - \psi(\theta)}\,|d^n x|$ for some functions c_A on M.

it is said to constitute an *exponential family* [5].

It is obvious that this functional form is invariant under an *affine* change of θ^A, and it is in fact elementary to show that no other reparametrization can preserve this form. Put in other words, to the exponential family is associated a *flat affine connection.* It turns out that this connection is the +1-Amari connection, which for this reason is also called *the exponential connection.*

11.2.7 Examples

Returning to our earlier examples, for the family

$$\rho\left(x_0, x\right) = \frac{2 + \sin\left(x - x_0\right)}{4\pi} \left|dx\right|$$

we may compute the Fisher metric $g = \left(1 - \frac{\sqrt{3}}{2}\right) dx_0 \otimes dx_0$ and the skewness $T = 0$. The identification $x \leftrightarrow x_0$ is the maximum likelihood point estimate. The $\overset{\alpha}{\Gamma}$-geodesic coordinate x_0 on S hence translates into a preferred coordinate x on M, so the given parameterizations (x and x_0) of M and S may be reconstructed from properties of the model itself, up to a common additive term.

For the family

$$\rho\left(\xi, \sigma, x\right) = \frac{e^{\frac{-1}{2\sigma^2}(x-\xi)^2}}{\sigma\sqrt{2\pi}} \left|dx\right|$$

we have

$$\begin{aligned} g &= \frac{1}{\sigma^2}\left(d\xi \otimes d\xi + 2d\sigma \otimes d\sigma\right) \\ T &= \frac{2}{\sigma^3}\left(4d\sigma \otimes d\sigma \otimes d\sigma + d\xi \otimes d\xi \otimes d\sigma + d\sigma \otimes d\xi \otimes d\xi + d\xi \otimes d\sigma \otimes d\xi\right) \end{aligned}$$

The Fisher metric is that of the standard hyperbolic plane [31], while the nontrivial skewness tensor in this case leads to a flat exponential connection, confirming that the normal family is an exponential family. Affine coordinates for this family are

$$\begin{aligned} \theta^1 &= \frac{-1}{2\sigma^2} \\ \theta^2 &= \frac{\xi}{\sigma^2} \end{aligned}$$

The Fisher metric is invariant under the full hyperbolic group, while only ξ-translations, simultaneous rescaling of σ and ξ and combinations of these are symmetries for both g and T.

Finally, for the Cauchy family

$$\rho\left(\xi, \sigma, x\right) = \frac{1}{\pi} \frac{\sigma}{(x - \xi)^2 + \sigma^2} \left|dx\right|$$

we have

$$\begin{aligned} g &= \frac{1}{2\sigma^2}\left(d\xi \otimes d\xi + d\sigma \otimes d\sigma\right) \\ T &= 0 \end{aligned}$$

Again, the Fisher metric is that of the hyperbolic plane. Both the metric and the (trivial) skewness tensor are invariant under the full hyperbolic group:

$$z \mapsto \frac{az + b}{cz + d}$$

where a, b, c and d are real with $ad - bc = 1$ and the complex coordinate z is given by

$$z = \xi + i\sigma$$

The higher invariance of the Cauchy family reflects the fact that if X is a Cauchy random variable, so is

$$\frac{aX + b}{cX + d}$$

This invariance of the Cauchy family under the group of projective transformations, makes it particularly interesting in camera modelling.

Cauchy distributed errors may also be used at the level of theoretical investigations of information based guidance, since they lead to nontrivial maximum likelihood estimators. (Simply taking the arithmetic mean of a m-fold sample from a Cauchy variable will not decrease the width of the distribution density function.) In short, the Cauchy family gives a very transparent illustration of the fact that the asymptotic formulas for the distribution of the maximum likelihood estimator is a quite different thing than the central limit theorem.

11.2.8 Transformational Families

A statistical manifold (associated to the statistical model (M, S, ρ_{SM})) is a transformational family with respect to the group G of transformations on S if there is a representation τ of G as transformations on M, such that

$$\rho_{SM}\left(\gamma\left(s\right), \tau_{\gamma}\left(m\right)\right) = \rho_{SM}\left(s, m\right)$$

identically in $\gamma \in G$. The case when $S = G$ (and the action on S is group multiplication) is particularly simple. The Cauchy family is of course an extreme example, having a three-dimensional transformation group for a two-dimensional family. For the distributions of interest in this report, there is a one-dimensional transformational symmetry: a measurement error is *added* to the true value. In the literature, there is sometimes introduced a distinction between what is called a location model and a scale model, the former having an additive group action, the latter a multiplicative one. A logarithmic reparametrization, however, turns one into the other, so there is no real difference. In fact any one-dimensional transformation model may be represented as a location model.

11.2.9 The Fréchet-Darmois-Cramér-Rao bound

Let (M, S, ρ_{SM}) be a parametric model and f a real valued function on M and let the expectation of f be F. The "expectation" depends on the actual distribution, so F is a function on S. Let Y be an arbitrary vector field on S, later to be chosen conveniently. Taking the Y Lie derivative [1] of the identity

$$\int_M (f - F)\rho dx = 0$$

we arrive at

$$\int_M (f - F)(L_Y \ln \rho)\rho dx = L_Y F$$

The Cauchy-Schwarz inequality then gives

$$\int_M (f - F)^2 \rho dx \int_M (L_Y \ln \rho)^2 \rho dx \geq (L_Y F)^2$$

Here $\int_M (L_Y \ln \rho)^2 \rho dx$ may be written in terms of the Fisher information metric as $g_{AB} Y^A Y^B$. If, finally Y is chosen so that $\frac{(L_Y F)^2}{g_{AB} Y^A Y^B}$ is maximized, i.e.

so that $Y^A = g^{AB}\frac{\partial F}{\partial \theta^B}$ (where g^{AB} are the components of the inverse metric), we arrive at the general lower bound on the variance $V = \int_M (f - F)^2 \rho dx$ of f (which by construction is an unbiased estimator of F).

$$V \geq g^{AB} \frac{\partial F}{\partial \theta^A} \frac{\partial F}{\partial \theta^B}$$

The result holds whenever the integrals considered exist, and the parameter dependency is smooth.

11.2.10 Invariant Priors

Our main formal objection to the Bayesian approach is that "complete ignorance" is represented by a "uniform" probability density, though the "uniformity" is in the eye of the beholder, to the extent that this property is not invariant under reparametrization. There is, however, a preferred density defined on the S manifold, namely the Fisher information metric volume form $\sqrt{\det g}\,|d\theta|$.

This is the so-called Jeffreys' prior.

In the case of exponential families, or generally when some Amari α -connection is flat (or more generally, when its induced connection on the bundle of volume forms is flat), one may construct other natural reference densities. The Barndorff-Nielsen formula below shows that Bayesian and likelihoodist parameter estimation become asymptotically equivalent, provided that the Bayesian prior corresponding to ignorance is Jeffreys' prior.

11.2.11 Barndorff-Nielsen's Formula

We give a simplified version of the Barndorff-Nielsen formula [36, 5]: for large samples the maximum likelihood estimator $\hat{\theta}$ is approximately distributed as

$$p(\hat{\theta}) \sim \sqrt{\det\left(\frac{1}{2\pi} g(\hat{\theta})\right)} e^{-\left(\ell(\hat{\theta}) - \ell(\theta)\right)} \left|d\hat{\theta}\right|$$

Here θ is the true parameter value, and $\hat{\theta}$ is the maximum likelihood estimator. The log-likelihood function is denoted by ℓ. Furthermore, as the sample size grows, $\ell(\hat{\theta}) - \ell(\theta)$ is approximately quadratic close to those points where it is not too large (i.e. when $p(\hat{\theta})$ is significantly nonzero), so the maximum likelihood estimator is asymptotically normal. This is true even for Cauchy variables, for which the central limit theorem does not hold (in the Gaussian sense). The full Barndorff-Nielsen formula uses the observed information instead of the above expected information in the determinant factor, and also takes into account so-called ancillary variables as well as a normalization correction.

11.3 Information Based Guidance

We begin by presenting the information based guidance problem in discrete time. Consider a random process consisting of independent random variables X_t, whose probability densities depend on two parameters, θ and u.

$$p(x, \theta, u)\,|dx|$$

(all variables are multidimensional, and in general take values in manifolds) The unknown parameter θ is fixed in time, while the control variable $u = u_t$ may be chosen at each instant. The problem is to sequentially choose u_t, based on earlier observations of the X process, so that some given likelihood-based utility function is maximized. Since future values of X are unknown, the utility function for the whole process will be based on *expected future likelihoods.* On the other hand, since the parameter θ is also unknown, the best we can do is to compute the expectation w.r.t. the probability density corresponding to the best estimate we have at the actual instant. So the utility function is an expression in terms of the *most likely expectation of the future likelihood.*

What utility functions are reasonable? In principle, the pure likelihoodist approach would be to aim at a situation where the normalized likelihood function is less than a prescribed value in a region as small as possible. Some measure of this smallness has to be chosen, such as the "diameter" in some metric sense.

In general however, the log-likelihood function quickly assumes the parabolic shape mentioned in the discussion of the simplified Barndorff-Nielsen formula. So asymptotically (and in practice, often from the very start on) the relevant data for the likelihood function are the position of its maximum and the Hessian there, which equals the Fisher information metric. So instead of considering "diameters" of general set shapes, we may work with expressions in terms of the Fisher metric.

11.3.1 The Utility Function

Let $\tilde{g}$ denote a "reference" Riemannian metric on S (the manifold of the θ-variable). It is meant to represent the importance of a precise parameter determination in different θ-directions at the θ-point in question. It thus defines an importance scale against which we may judge our information (represented by the *observed* Fisher information matrix $\mathcal{J}$). Directions in which our information is higher than the reference (i.e. the specification) are the directions spanned by those eigenvectors of $\tilde{g}^{-1}\mathcal{J}$ which correspond to eigenvalues greater than unity. Suitable utility functions are expressions in terms of the spectrum of $\tilde{g}^{-1}\mathcal{J}$. In this report, following [16], we use the determinant of $\tilde{g}^{-1}\mathcal{J}$. This function has the property, that its maximization is independent of the scaling metric $\tilde{g}$. It may, however, lead to an unwanted behaviour, as bad precision in one direction may be compensated for, by a higher precision in another direction.

In the applications (given in the other chapters) in this report, the S-manifold is invariably the physical Euclidean plane, over which a UAV moves. The $\tilde{g}$ is taken as the Euclidean metric itself, represented by the unity matrix in cartesian coordinates and thereby gets hidden in the matrix formulas. Varying metrics may be helpful, even in this otherwise very streamlined application, distinguishing areas in the plane where high measurement precision is necessary, and areas where it is not.

The choice of the reference metric $\tilde{g}$ and the actual utility function (in terms of the spectrum of $\tilde{g}^{-1}\mathcal{J}$) are, so to speak, the decision theoretic elements of the method, while the rest is "pure" inference oriented. Likewise, if a larger decision theoretic problem is posed, it will typically provide the choices of the reference metric and the utility function.

11.3.2 Computational Simplification Induced by Symmetry

Suppose that the measurements X are given by an expression in the actual values of the parameters θ and u plus an additive noise term E.

$$X = F(\theta, u) + E$$

the noise having the probability density $p(e)\,|de|$. This is not so restrictive as it may seem at first, since any one-dimensional symmetry may be transformed into a translational symmetry by a change of coordinates. Then, in the computation of the *expected* Fisher information metric g, we have relations like

$$\frac{\partial \rho}{\partial \theta^A} = -\frac{\partial p}{\partial e}\frac{\partial F}{\partial \theta^A}$$

leading to the formula

$$g_\rho = g_p \frac{\partial F}{\partial \theta} \otimes \frac{\partial F}{\partial \theta}$$

This means that the information matrix for such a symmetry situation, and when the measured quantity is scalar, reduces to the computation of the constant scalar "information matrix" of p (or, rather the translational family defined by p).

More complicated transformation models lead to analogous simplifying formulas.

If, furthermore, the function $F(\theta, u)$ is invariant under some simultaneous group action on the unknown parameter θ and the controlled variable u, further simplifications obviously follow in the computations of the (u-dependent) Fisher information metric.

In the case of a pure bearings only measurement, the unknown point has the cartesian coordinates (x, y), the position of the measuring camera is (ξ, η) and its optical axis is in the direction ϕ. The point has as camera coordinates $(\tilde{x}, \tilde{y})$ given by

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi & \xi \\ \sin\phi & \cos\phi & \eta \\ 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

This expression clearly has the property of being invariant under simultaneous left-premultiplication by any matrix of the form

$$\begin{pmatrix} \cos\alpha & -\sin\alpha & \beta \\ \sin\alpha & \cos\alpha & \gamma \\ 0 & 0 & 1 \end{pmatrix}$$

of both the matrices

$$\begin{pmatrix} \cos\phi & -\sin\phi & \xi \\ \sin\phi & \cos\phi & \eta \\ 0 & 0 & 1 \end{pmatrix}$$

and

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

reflecting the Euclidean invariance of the problem.

If it is assumed that the bearing angle $\arctan\frac{\tilde{y}}{\tilde{x}}$ is measured with an additive error E distributed as $p(e)\,|de|$, we see from the above that the Fisher information metric in terms of the parameters $\tilde{x}$ and $\tilde{y}$ is given by $g_p\ \tilde{\mathbf{s}}\ \tilde{\mathbf{s}}^T$, where

$$\tilde{\mathbf{s}} = \frac{1}{\tilde{x}^2 + \tilde{y}^2} \begin{pmatrix} -\tilde{y} \\ \tilde{x} \end{pmatrix}$$

This rank 1 matrix is then further transformed (by means of the chain rule) into the Fisher information matrix in terms of the true parameters x and y by means of the formula $g_\rho = g_p\ \mathbf{s}\ \mathbf{s}^T$ where

$$\mathbf{s} = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \tilde{\mathbf{s}}$$

11.3.3 The Information Dynamics

Since the measurement errors are independent each measurement adds an independent log-likelihood term to the total log-likelihood function $\ell(\theta, t)$. At the maximum likelihood point $\hat{\theta}(t)$ (which for simplicity is assumed to be uniquely defined) it holds that $\ell'_\theta(\hat{\theta}(t), t) = 0$ identically in t. If we now consider the case of time continuous measurements, this identity may be differentiated with respect to t.

By doing so, solving for $\dot{\hat{\theta}}(t)$ and expressing the result in tensor notation, we get

$$\dot{\hat{\theta}}^A = -\mathcal{J}^{AB} \ell''_{\theta^B t}(\hat{\theta}, t)$$

where $\mathcal{J}^{AB}$ denotes the inverse (the "formation", in Fisher's terminology) of the observed information matrix. Finally, differentiating the formula for the Fisher information $\mathcal{J}_{AB} = \ell''_{\theta^A \theta^B}(\hat{\theta}, t)$ and substituting the dynamics $\dot{\hat{\theta}}(t)$ of the estimate, we arrive at

$$\dot{\mathcal{J}}_{AB} = \ell'''_{\theta^A \theta^B t}(\hat{\theta}, t) - \ell'''_{\theta^A \theta^B \theta^C}(\hat{\theta}, t) \mathcal{J}^{CD} \ell''_{\theta^D t}(\hat{\theta}, t)$$

This is the formula, which should be used in the general case. Now, for typical dynamical estimation problems, the log-likelihood function quickly approaches an approximately quadratic form (asymptotically so, in any coordinates, and more quickly in particular "linearizing" coordinates), where it holds that $\ell'''_{\theta^A \theta^B \theta^C} = 0$ and that $\ell'''_{\theta^A \theta^B t}$ is independent of θ. The dynamics for the Fisher information then simply becomes

$$\dot{\mathcal{J}}_{AB} = \ell'''_{\theta^A \theta^B t}(\theta, t)$$

So the total observed information increases by additive contributions in the form of a "rate of information observed at time t" giving an expression for $\ell'''_{\theta^A \theta^B t}$. Information based guidance, then, is the technique of selecting the control variables, so that the *expected* future observed information leads to a maximal utility function.

In the case of the planar bearings-only problem, one choses the successive ξ, η and ϕ values in such a way, that the object (at (x, y)) is likely to be seen from several different angles.

Now, the term $\ell'''_{\theta^A \theta^B t}$ representing the rate of increase of the observed information in the approximation mentionned above (the ignored term being essentially a geometric correction to make the formula truly invariant) is the Hessian of the rate of increase of of the log-likelihood, which we identify with the expression $\log p(X_{obs}(t), \theta, u(t))$ from our measurement error model discussed above. In short we think of the continuous measurement of the process X as providing "one measurement" (of the above form) per unit time. The contribution to the observed information $\mathcal{J}$ being the Hessian of $\log p(X_{obs}(t), \theta, u(t))$ computed at $\hat{\theta}(t)$.

Note that the contribution to the observed information may be negative for some (unexpected) measurements. This is the case when the new evidence speaks against the earlier estimate. We also mention that the simplification (using the "Hessian" term only) is practical and simplifies presentation, but is not essential to the fundamental step of identifying the rate of observed information as the "Hessian" of the individual measurement's log-likelihood. (The omitted term is part of a geometrically defined hessian, in terms of an Amari-like connection.) In any case, some assumption similar to $\ell'''_{\theta^A \theta^B \theta^C} = 0$ or a coordinate independent version of it has to be done, in order to have a closed system of differential equations for the filter.

What we have described so far in this subsection is the actual dynamics of the estimate and the observed information due to successive actual measurements. When future motion is planned, the evolution of the maximum likelihood estimate cannot be known in advance, and the best thing we can do (according to the likelihood philosophy) is to use the present estimate for calculating estimates of future quantities. Similarly, the values of the hessians of of future observed contributions to the likelihood function cannot be known in advance, but the *expected* information from future measurements may be calculated as in the previous subsection.

So apart from the actual evolution of our estimation state, driven by the successive measurements, we have an expected future dynamics given by the Fisher metric g. It is given by $\dot{\hat{\theta}} = 0$ and

$$\dot{\mathcal{J}}_{AB} = g_{AB}(\hat{\theta}, u(t))$$

11.3.4 The Optimal Control Problem

So far, we have referred to the variables u as control variables, which may be chosen for the optimization of the utility function. These are typically restricted by static conditions (bounds, inequalities..). There may also be dynamical conditions, such as bounds on $\dot{u}$. Generally, we may think of the u-variables as outputs from a general nonlinear control system of the type

$$\begin{aligned} \dot{z} &= f(z, v) \\ u &= h(z, v) \end{aligned}$$

where z represents the state on some suitable manifold, The full (expected future) dynamics of the state variables $\mathcal{J}$ and z is then given by

$$\begin{aligned} \dot{z} &= f(z, v) \\ \dot{\mathcal{J}} &= g(\hat{\theta}, h(z, v)) \end{aligned}$$

It is now clear how to formulate an optimal control problem. Given initial values (at time t_0) of z and $\mathcal{J}$ and the value of the estimate $\hat{\theta}(t_0)$, the control v is chosen (among those satisfying appropriate static constraints) so as to maximize the final time utility $V(\mathcal{J}(t_{end}))$. The form of the utility function has been discussed above.

More generally, we may add other terms to the criterion function, such as an integral constraint on, say, $L(z, v)$. In the surveillance application discussed in this report, this Lagrangian may contain terms penalizing the sensor platform's presence in dangerous areas (a penalty on z) or penalizing quick camera movements (a penalty on v).

In any case, this now has the form of a classical optimal control problem, and the methods of maximum principle, dynamic programming or direct numerical methods may be applied. This is discussed at length in other chapters of this report.

Now, at time t_0, the solution of the optimal control problem will give us the full expected optimal trajectory of the system, and in particular the optimal choice of the control signal v at time t_0. As time proceeds, the actual measurements give other data, than those anticipated in the optimization (changing both the information matrix and the estimate $\hat{\theta}(t)$ itself), so the optimal trajectory calculation has (in principle) to be redone continuously, taking into account the actual measurements. (In practice, of course, the re-optimization neither can nor needs to be done on-line "continuously".)

Summarizing, optimization in terms of the *expected* information dynamics gives us the present choice of control variables to put into the *observed* informations dynamics (the actual dynamics of the system), which in turn provides a new initial state for the "next" optimization problem.

11.3.5 Moving Targets

In this chapter we have formulated the information based guidance problem as one of parameter estimation. It is clear that this is appropriate if the purpose of the system is to localize static targets, or constructing a map. If the target points move according to a known (practically stable) dynamics, the problem still has the same form, the parameters being, say, the initial position of the moving target.

If the dynamics is not fully known, a worst case approach may be appropriate. (It is usually the case that there is a known bound on the acceleration of a target.) A stochastic model of the target's motion is often assumed in the literature. This leads to the complicated problem of nonlinear filtering, and the powerful methods of Fisher information do not apply. In some situations, modelling unknown motions of the target by a stochastic process may be a reasonable representation of what is known about the target, in other cases a detailed stochastic model is an unjustified ad hoc assumption.

A heuristic approach to taking into account unknown dynamics of the target is to postulate an aging of the information. Old data can be given lower weight in the calculation of the likelihood function, much as if old data had been obtained by noisier measurements. This would lead to modified information dynamics both for the observed information, and for the expected information. In the case of expected information, a modified dynamics may read

$$\dot{\mathcal{J}} = -\kappa \mathcal{J} + g(\hat{\theta}, u(t))$$

and similarly for the observed information (with the same κ). The "dissipative" term $-\kappa\mathcal{J}$ in the dynamics will typically have the effect that there will be a bound on how precise the parameters may be determined. The same effect is obtained with a quadratic dissipative term of the form $-\kappa\mathcal{J}K\mathcal{J}$, where K is a positive symmetric matrix characterizing the "aging".

By such heuristic "aging" terms in the dynamics, one may motivate the Riccati equation form of the information filter, used by [16] and in the rest of this report, within the Fisher information framework. The particular quadratic form of aging, and its accompanying Riccati equation form where originally motivated by reference to the Kalman filter, and more generally to the extended Kalman filter.

11.4 Large Problems From Smaller Constituents

In the present chapter, the discussion has been kept at a quite abstract level, the only concrete example being an idealized planar bearings-only problem, determining the position of one single point. We will now indicate how this problem formulation allows generalizations to larger problems, by systematic operations. The discussion will be in terms of this particular problem, but the principles and the systematics is general.

11.4.1 Several Target Points

Suppose that there are N target points and that the sensor can distinguish between these (due to e.g. different "color"). The space of unknown parameters is now the cartesian product of N Euclidean planes. A point in this space represents the simultaneous positions of all the N target points. Each point gives rise to an individual bearings component in an N-dimensional measurement vector X. The camera platform's position and heading are our control variables. The utility function may be constructed from the (large) information matrix and an appropriate scaling metric (some target points may be more important than others).

11.4.2 Several Camera Platforms

What has been said about several target points, also holds for several camera vehicles (with full communication). If there are N_1 cameras, each target point will lead to N_1 components in the measurement vector. There will be $3N_1$ control variables, corresponding to the cameras' positions and headings.

If the Lagrangian is such that large motions are penalized, the cameras will typically move in a way that is globally economic, each camera's motion being determined so that the total cost becomes minimal.

11.4.3 Occlusion, Aperture Limitations etc.

Occlusion, limited aperture, finite range of sight and spatially varying optical properties are all examples of effects that break the symmetry of the idealized problem formulation. Otherwise the general problem setting still applies. An interesting question is whether or not (or to what extent) such problems may be simplified or better understood by insights into the corresponding idealized group invariant problem.

11.4.4 Target Semantics

In the whole discussion above, the canonical problem is one of determining the positional coordinates of a set of distinguishable target points. The principles of information based guidance may however be used for much more general situations than those. In any application, though, one must have some prior model of the world. Suppose we have a family of possible "photographic pictures", e.g. two possible types of vehicles, each seen from all possible angles. For a given noise model, we may calculate likelihoods for different original pictures, given some measurements. By adjusting camera parameters and by utilizing any motional freedom for the sensor platform, we may use information based guidance to classify the target (as well as determining its position).

12 Conclusions

12.1 Summary

This report is concerned with UAV surveillance, i.e. a UAV with a gimballed EO/IR-sensor searches for objects and once found, the object should be localized. In particular an information-theoretic approach is presented.

Technically, *information* is a measure of the accuracy to which the value of a parameter or stochastic variable is known. There are two commonly used formal definitions of information, the *Entropic information* and *Fisher information.* An information matrix is constructed from the states representing the features, e.g. objects and area grid points. A utility function is defined based on the information matrix and the planning process is to find optimization parameters (control signals or path points) that maximize the utility function at the planning horizon.

Different parameters are discussed, such as different utility functions, time horizon and prior information. The methods are applied to one object localization, n objects localization, as well as area exploration. The method is also extended with a sensor gimbal model and occlusion models.

A general planning hierarchy is proposed based on a NURBS signal representation. With this planning hierarchy the optimization problem is solved in a number of steps. In general, the solution at one level serves as the start solution in the next level and the models, e.g. of the environment, are more detailed at every level. The short term planning is using the information theoretic approach, but for the long term planning a graph search method is proposed.

12.2 Conclusions

The information-theoretic approach is very promising, but several optimization issues remain open. However, these issues are not primarily due to the choice of approach, but stem from the very complex nature of the problem that we are trying to solve. The information-theoretic approach has several advantages. It is straight forward to generate utility measures in term of information that captures the nature of the problem in an intuitive way. Information is also connected to probability distributions, which are used to describe the uncertainties of the feature estimation. For instance, in the object localization example, the information is appropriate since the more information you have about an object, the better localization you get. The information-theoretic approach is also very suitable for decentralized and/or multi-sensor applications. Assuming that sensor observations are conditionally independent, the fusion step from different sensors becomes remarkably straightforward.

From the theoretical point of view, the information-theoretic approach has the advantage of having simple transformational properties. This allows its formulation also when highly nonlinear mappings are involved, as is the case for the sensor systems discussed in this report.

The problem considered in this report is very complex since it is non-convex and non-linear. Therefore, there is no guarantee that the optimization routine finds the global optimum. However, in many applications it is not required that the exact global optimum is found, a reasonably good feasible solution is often enough. In the quest to find such a solution, the elements of problem formulation, path/signal representation, utility function and choice of optimization algorithm are all equally important.

12.3 Future Work

There are many questions and problems left for the future work:

- The optimization routines are crucial. More algorithms should be reviewed and evaluated.
- More research and evaluations of different utility functions and path representations must be done.
- The proposed planning hierarchy must be thoroughly evaluated and tested. Robustness must be guaranteed.
- Models of environment (occlusion) based on hardware accelerated computer graphics has been developed, but not thoroughly tested.
- Better vehicle and sensor models should be developed (this is connected to the choice of path representation). The navigation uncertainties should be taken under considerations.
- More research in the cooperating platforms problem.
- Geometrical corrections to the filter update equation, corresponding to a geometrical version of the current assumption that $\ell'''_{\theta^A\theta^B\theta^C} = 0$.
- A theoretical investigation of the filter's performance based on the asymptotical properties of the likelihood function.

Bibliography

[1] R. Abraham and J. E. Marsden. *Foundations of Mechanics.* Benjamin/Cummings Publishing Company, 1978.

[2] A. Agrachev and Yu. Sachkov. *Control Theory From tha Geometric Viewpoint.* Springer Verlag, 2004.

[3] Eugene L. Allgower and Kurt Georg. *Numerical continuation methods: an introduction.* Springer-Verlag New York, Inc., New York, NY, USA, 1990.

[4] S.-I. Amari and H. Nagaoka. *Methods of Informational Geometry.* American Mathematical Society, 2000.

[5] O. E. Barndorff-Nielsen and D. R. Cox. *Inference and Asymptotics.* Chapman & Hall, 1994.

[6] W. Beard, T. McLain, M. Goodrich, and E. Anderson. Coordinated target assignment and intercept for unmanned air vehicles. *IEEE Transactions on Robotics and Automation*, pages 911–922, December 2002.

[7] F. Bennet and S. Fenelius. Sceneserver - a 3D software assisting developers of computer vision algorithms. Teknisk rapport/Technical report FOI-R–0831–SE, Defence Research Establishment, Sweden, 2003.

[8] D. P. Bertsekas. *Dynamic Programming and Optimal Control: 2nd Edition.* Athena Scientific, 2001.

[9] Ulf Brännlund and Stefan Feltenmark. Derivatafri optimering för Flight-Clearance-beräkningar. Technical Report FOI-R–0678–SE, Swedish Defence Research Agency, 2002.

[10] B. T. Clough. Metrics, schmetrics! How do You determine a UAV's autonomy anyway? In *Proceedings of the 2002 PerMIS Workshop.* NIST, August 2002. NIST Special Publication 990.

[11] Laura Downs, Tomas Akenine-Möller, and Carlo Séquin. Occlusion horizons for driving through urban scenery. In *Proceedings of the 2001 Symposium on Interactive 3D graphics*, pages 121–124. ACM Press, 2001.

[12] Irina Dumitrescu and Natashia Boland. Algorithms for the weight constrained shortest path problem. *International Transactions in Operational Research*, pages 15–30, January 2001.

[13] A. W. F. Edwards. *Likelihood.* Cambridge University Press, 1972.

[14] Philip E. Gill, Walter Murray, and M. A Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. In *SIAM J. Optim.*, volume 12, pages 979–1006, 2002.

[15] T. Glad and L. Ljung. *Reglerteori. Flervariabla och olinjära metoder.* Studentlitteratur, Lund, 1997.

[16] B. Grocholsky. *Information-Theoretic Control of Multiple Sensor Platforms.* PhD thesis, The University of Sydney, 2002. Available from http://www.acfr.usyd.edu.au.

[17] D.A. Grundel and D. E. Jeffcoat. Formulation and solution of the target visitation problem. *Proceedings of the AIAA Intelligent Systems Technical Conference, Chicago, Illinois*, September 2004.

[18] S. S. Gupta, editor. *Differential Geometry in Statistical Inference.* Inst. of Math. Stat., 1987.

[19] R. Hamren and M. Brandin. Classification of ground objects using laser radar data. Master's thesis, Linköping University, 2003. LiTH-ISY-EX-337.

[20] S. Har-Peled, Y. Wang, and P. K. Agarwal. Occlusion culling for fast walkthrough in urban areas. In *Eurographics 2001*, 2001.

[21] Jean-Marc Hasenfratz, Marc Lapierre, Nicolas Holzschuch, and François Sillion. A survey of real-time soft shadows algorithms. *Computer Graphics Forum*, 22(4):753–774, Dec. 2003. State-of-t.

[22] JunHyeok Heo, Jaeho Kim, and KwangYun Wohn. Conservative visibility preprocessing for walkthroughs of complex urban scenes. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 115–128. ACM Press, 2000.

[23] L. Ingber. Simulated annealing: Practice versus theory. *Mathl. Comput. Modelling*, 18(11):29–57, 1993.

[24] Lester Ingber. Web page of Lester Ingber. http://www.ingber.com/, (2005-09-30).

[25] R. E. Kass and P. W. Vos. *Geomerical Foundations of Asymptotic Inference.* Wiley, 1997.

[26] A. J. Krener. The convergence of the extended kalman filter. In A. Rantzer and C. I. Byrnes, editors, *Directions in Mathematical Systems Theory and Optimization.* Springer Verlag, 2003.

[27] J. Manyika and H. Durrant-Whyte. *Data Fusion and Sensor Management: A Decentralized Information-Theoretic Approach.* Prentice Hall, 1994.

[28] J. Manyika and H. Durrant-Whyte. *Data Fusion and Sensor Managment: A Decentralized Information-Theoretic Approach.* Ellis Horwood, London, 1994.

[29] MathWorks, Inc. *Optimization Toolbox User's Guide*, June 2004. Version 3.0, Fifth printing.

[30] P. S. Maybeck. *Stochastic Models, Estimation and Control*, volume 1–3. Academic Press, 1979.

[31] M.K. Murray and J.W. Rice. *Differential Geometry and Statistics.* Chapman & Hall, 1993.

[32] J. Nygårds, P. Skoglar, J. Karlholm, M. Ulvklo, and R. Björström. Towards concurrent sensor and path planning - A survey of planning methods applicable to UAV surveillance. Scientific Report FOI-R–1711–SE, ISSN 1650-1942, Swedish Defence Research Agency (FOI), Divison of Sensor Technology, SE-581 11 Linköping, Sweden, 2005.

[33] Y. Oshman and P. Davidson. Optimization of observer trajectories for bearings-only target localization. *IEEE Transactions on Aerospace and Electronic Systems*, 35(3):892–902, July 1999.

[34] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization Algorithms and Complexity.* Dover Publications, 1998.

[35] J.M. Passerieux and D. van Cappel. Optimal observer maneuver for bearing-only tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 34(3):777–788, July 1998.

[36] Y. Pawitan. *In All Likelihood.* Oxford University Press, 2001.

[37] Å. Persson. Extraction of individual trees using laser radar data. Technical Report FOI-R–0236–SE, Defence Research Establishment, Sweden, 2001.

[38] William T. Reeves, David Salesin, and Robert L. Cook. Rendering antialiased shadows with depth maps. *Computer Graphics (Proceedings of SIGGRAPH '87)*, pages 283–291, 1987.

[39] P. Skoglar. Modelling and control of EO/IR-gimbal for UAV surveillance applications. Scientific Report FOI-R–0893–SE, ISSN 1650-1942, Swedish Defence Research Agency (FOI), Divison of Sensor Technology, SE-581 11 Linköping, Sweden, June 2003.

[40] P. Skoglar, J. Nygårds, R. Björström, and M. Ulvklo. Information-theoretic approach for concurrent path and sensor planning for a UAV with EO/IR sensors. Scientific Report FOI-R–1685–SE, ISSN 1650-1942, Swedish Defence Research Agency (FOI), Divison of Sensor Technology, SE-581 11 Linköping, Sweden, 2005.

[41] Stanford business software inc. Web page. http://www.sbsi-sol-optimize.com/asp/sol_product_snopt.htm, (2005-09-30).

[42] A. James Stewart. Hierarchical visibility in terrains. In Julie Dorsey and Philipp Slusallek, editors, *Eurographics Rendering Workshop 1997*, pages 217–228, New York City, NY, 1997. Springer Wien.

[43] Seth Teller and Carlo Séquin. Visibility preprocessing for interactive walkthroughs. *Computer Graphics (Proceedings of SIGGRAPH '91)*, pages 61–69, 1991.

[44] O. Trémois and J.-P. Le Cadre. Optimal observer trajectory in bearings-only tracking for maneuvering sources. *IEE Proceedings Radar, Sonar and Navigation*, 146(1):31–39, February 1999.

[45] Unmanned Aerial Vehicles roadmap 2002-2027. Office of the Secretary of Defense, Departement of Defense, USA, December 2002. http://www.acq.osd.mil/usd/uav_roadmap.pdf.

[46] M. Ulvklo, J. Nygårds, J. Karlholm, and P. Skoglar. Image processing and sensor management for autonomous UAV surveillance. In *Airborne Intelligence, Surveillance, Reconnaissance (ISR) Systems and Applications XXVII, Proc. SPIE*, volume 5409, April 2004. To be published.

[47] N. N. Čencov. *Statistical Decision Rules and Optimal Inference.* American Mathematical Society, 1982.

[48] E. W. Weisstein. B-spline. From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/B-Spline.html (2005-09-30).

[49] E. W. Weisstein. Cubic spline. From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/CubicSpline.html (2005-09-30).

[50] E. W. Weisstein. Curvature. From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/Curvature.html (2005-09-30).

[51] M. Zabarankin, S. Uryasev, and P. Pardalos. Optimal risk path algorithms. *R. Murphey and P. Pardalos (Eds.) Cooperative Control and Optimization, Kluwer Academic Publishers*, pages 273–303, 2002.

FOI
Swedish Defence Research Agency
P.O. Box 1165
SE-581 11 LINKÖPING

Tel: +46 13 37 80 00
Fax: +46 13 37 81 00

www.foi.se